2005 Mobile Games White Paper

Presented at the Game Developers Conference 2005
by the IGDA Online Games SIG

Donald Wisniewski (white paper co-editor)
Derrick Morton (white paper co-editor)
Brian Robbins (SIG chairperson)
John Welch (SIG chairperson emeritus)
Steven DeBenedictis
Elonka Dunin
Jon Estanislao
Daniel James
Greg E. Mills
Gordon Walton
Jeferson Valadares
Table of Contents

I. CREDITS ................................................................................................................................. 3

II. INTRODUCTION .................................................................................................................... 4

III. MARKET OVERVIEW .......................................................................................................... 5

   A. MOBILE GAMING MARKET SIZE .................................................................................. 5
   B. RECENT MARKET TRENDS ............................................................................................. 7
   C. MOBILE VALUE CHAIN ..................................................................................................... 9
   D. CONSUMER TRENDS ....................................................................................................... 14

IV. THE FUTURE OF MOBILE GAMING .................................................................................. 16

   A. Q&A WITH GREG BALLARD, CEO OF SORRENT ......................................................... 16
   B. TRIP HAWKINS, CEO OF DIGITAL CHOCOLATE ........................................................... 17

V. MOBILE GAME CONTENT ................................................................................................... 18

   A. GAME GENRE BREAKDOWN AND ANALYSIS ............................................................... 18
   B. BRANDS IN MOBILE GAMES ....................................................................................... 18
   C. INTERACTIVE GAMING / CONNECTIVITY FEATURES .................................................. 19
   D. WHAT ARE THE CHALLENGES IN LAUNCHING A SUCCESSFUL MOBILE GAME? .... 20

VI. MOBILE DEVICES ................................................................................................................ 21

   A. MOBILE DEVICE EVOLUTION ....................................................................................... 21
   B. DEVICE TRENDS ............................................................................................................. 22

VII. MOBILE GAME TECHNOLOGY .......................................................................................... 24

   A. INTRODUCTION ............................................................................................................... 24
   B. THE MAJOR TECHNOLOGY PLATFORMS ..................................................................... 24
   C. OTHER OPERATING SYSTEMS (OS) ............................................................................. 28
   D. NEW GAMING FEATURES AND TECHNOLOGIES ......................................................... 29

VIII. DEVELOPING MOBILE GAMES ......................................................................................... 31

   A. PLANNING/DESIGN ......................................................................................................... 31
   B. PUBLISHER RELATIONS ................................................................................................. 34
   C. FIRST PLAYABLE/PROTOTYPE/DEMO .......................................................................... 35
   D. GAME BALANCING ......................................................................................................... 36
   E. DEVELOPMENT ................................................................................................................ 38
   F. MOBILE DEVELOPMENT TOOLS .................................................................................. 42
   G. TESTING AND PUBLISHER/IP APPROVAL ................................................................. 44
   H. PORTING/LocalIZATION ................................................................................................. 45
   I. CERTIFICATION AND CARRIER SUBMISSION ............................................................... 47

IX. FROM THE FIELD: STARTING A MOBILE DEVELOPMENT COMPANY ............................... 50

   A. MOTIVATION TO GET STARTED .................................................................................... 50
   B. TYPES OF WIRELESS GAME DEVELOPERS ................................................................. 51
   C. FINANCE YOUR COMPANY ............................................................................................. 52
   D. MARKET CONDITIONS ................................................................................................... 52
   E. REALLY GETTING STARTED .............................................................................................. 53

X. CLOSING ................................................................................................................................. 54

XI. VERSION HISTORY ............................................................................................................... 55
## I. Credits

<table>
<thead>
<tr>
<th>Editors</th>
<th>Section Editors</th>
</tr>
</thead>
<tbody>
<tr>
<td>Donald Wisneiwski</td>
<td>Arvind Bhatia (Market Overview)</td>
</tr>
<tr>
<td>Senior Vice President, Publishing</td>
<td>Principle Analyst, SWS Securities, LLC.</td>
</tr>
<tr>
<td>Superscape, Inc.</td>
<td><a href="mailto:ABhatia@swst.com">ABhatia@swst.com</a></td>
</tr>
<tr>
<td><a href="mailto:donald@superscape.com">donald@superscape.com</a></td>
<td></td>
</tr>
<tr>
<td>Derrick Morton</td>
<td>Jill Braff (Consumer Profile)</td>
</tr>
<tr>
<td>General Manager, Mobile Games RealNetworks,</td>
<td>VP Marketing, Sorrent, Inc.</td>
</tr>
<tr>
<td>Inc.</td>
<td><a href="mailto:jill.braff@sorrent.com">jill.braff@sorrent.com</a></td>
</tr>
<tr>
<td><a href="mailto:dmorton@real.com">dmorton@real.com</a></td>
<td></td>
</tr>
<tr>
<td>Arvind Bhatia (Market Overview)</td>
<td>Jon Estanislaao (Mobile Game Development)</td>
</tr>
<tr>
<td>Principle Analyst, SWS Securities, LLC.</td>
<td>Senior Manager, Business Development and</td>
</tr>
<tr>
<td></td>
<td>Emerging Platforms, Activision, Inc.</td>
</tr>
<tr>
<td></td>
<td><a href="mailto:jestanislaao@activation.com">jestanislaao@activation.com</a></td>
</tr>
<tr>
<td>Scott Edison (Content)</td>
<td></td>
</tr>
<tr>
<td>Director-Mobile; Oberon Media</td>
<td></td>
</tr>
<tr>
<td><a href="mailto:scott.edison@oberon-media.com">scott.edison@oberon-media.com</a></td>
<td></td>
</tr>
<tr>
<td>Lou Fasulo (Value Chain)</td>
<td></td>
</tr>
<tr>
<td>Product Manager, Games&amp; Applications, Cingular</td>
<td></td>
</tr>
<tr>
<td>Wireless</td>
<td><a href="mailto:lou.fasulo@cingular.com">lou.fasulo@cingular.com</a></td>
</tr>
<tr>
<td>Paul Mangalik (Mobile Devices)</td>
<td></td>
</tr>
<tr>
<td>Paul Beadon (Technology Platforms)</td>
<td></td>
</tr>
<tr>
<td>CTO &amp; Sr. VP Reasearch and Development,</td>
<td></td>
</tr>
<tr>
<td>Superscape</td>
<td></td>
</tr>
<tr>
<td><a href="mailto:paul.beadon@superscape.com">paul.beadon@superscape.com</a></td>
<td></td>
</tr>
<tr>
<td>Greg Ballard (The Future of Mobile Gaming)</td>
<td></td>
</tr>
<tr>
<td>CEO, Sorrent</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Contributors</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>John Newcomer (Planning and Design &amp; First</td>
<td>Johanna Wilson (Porting/Localization)</td>
</tr>
<tr>
<td>Playable, Prototype, and Demo)</td>
<td>Chief Executive Officer, OpenPath Products</td>
</tr>
<tr>
<td>Vice President of Game Design, In-Fusio</td>
<td><a href="http://www.openpathproducts.com">www.openpathproducts.com</a></td>
</tr>
<tr>
<td>Robert Boehm (Development)</td>
<td>Richard A. Smith (Porting/Localization)</td>
</tr>
<tr>
<td>Partner Manager, Exit Games GmbH</td>
<td>Chief Technology Officer, OpenPath Products</td>
</tr>
<tr>
<td></td>
<td><a href="mailto:Richard@openpathproducts.com">Richard@openpathproducts.com</a></td>
</tr>
<tr>
<td>Harald Behnke (Development)</td>
<td>Michael Nasalroad (Certification and Carrier</td>
</tr>
<tr>
<td>Chief Executive Officer, Exit Games GmbH</td>
<td>Submission), Premium Services Certification –</td>
</tr>
<tr>
<td></td>
<td>Games, Sprint</td>
</tr>
<tr>
<td>Christof Wegmann (Development)</td>
<td>Kim Chau (Mobile Devices)</td>
</tr>
<tr>
<td>Chief Technology Officer, Exit Games GmbH</td>
<td>Brian Wolkenberg (Mobile Devices)</td>
</tr>
<tr>
<td>Drew Fisher (Testing and Publisher/IP Approval)</td>
<td></td>
</tr>
<tr>
<td>General Manager, Barking Lizards</td>
<td>Motorola</td>
</tr>
</tbody>
</table>
II. Introduction

Mobile Games are one of the fastest growing segments of the game industry. However, mobile games are also some of the most challenging products to develop and distribute. Unlike most areas of the game business, mobile games must be created to run on hundreds of handsets, must pass quality assurance tests from dozens of organizations, and are marketed and sold unlike any other entertainment product.

In the summer of 2004, the International Game Developers’ Association (IGDA) Online Games Special Interest Group (SIG) solicited volunteers interested in contributing their expertise to a document which would present a snapshot of the major aspects of the mobile game market in 2005. The volunteers were drawn from all areas of the business including marketing, distribution, product development and technology. While the IGDA is an organization dedicated to the concerns of game developers, the organization felt it was important that developers have a grasp of all aspects of the market so that the products they develop find their way into the hands of consumers and are as successful as possible.

This White Paper is primarily geared toward those members of the game development community who may be considering entering the mobile game industry. For the purposes of the White Paper, we have defined mobile games as those that are delivered via wireless networks to devices whose primary function is a mobile phone. The goal of this White Paper is to give individuals with no knowledge of mobile games an overview of the challenges involved in their development, distribution, and marketing.

This work was created and written by volunteers on behalf of the community at large. The White Paper content is based on the individual input of the contributors and does not necessarily reflect the opinions of the IGDA or the companies at which the individuals work. The volunteers worked diligently to provide accurate information but there may be inaccuracies and information that has become outdated since this White Paper was originally written. The information was obtained from publicly available sources, including company websites, company annual reports and SEC filings, and news sites dedicated to games.

The findings of this paper were presented at the 2005 Game Developers Conference (GDC) in San Francisco, California. This White Paper is the complete version produced by the IGDA Online Games SIG. It is available for download at no charge courtesy of the IDGA via www.igda.org/online. There you can also find other documents created by the Online Games SIG such as the Web and Downloadable Games Whitepaper. We encourage anyone with an interest in games to participate in the exchange of ideas and information that is the purpose of the group.

This information is provided for informational purposes. If you use any of the information you are responsible for its use and must reference this document as a source. We also request that you reference the URL where the complete document can be found at www.igda.org/online.
III. Market Overview

Strong growth and innovation in the wireless communications industry has led to the emergence of the wireless entertainment market. Factors such as the rapidly increasing availability of multi-media capable phones, unprecedented growth in the number of wireless subscribers and deployment of next-generation wireless networks are responsible for the enormous growth being experienced in the wireless entertainment market.

A. Mobile Gaming Market Size

While sources may disagree on the current and future size of the market, all of them are predicting huge growth ahead. The table below highlights the various projections for the market size and the growth opportunity.

<table>
<thead>
<tr>
<th>Source</th>
<th>2003</th>
<th>2008</th>
<th>CAGR %</th>
</tr>
</thead>
<tbody>
<tr>
<td>Global Gaming Revenue</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ARC ($MM)</td>
<td>$1,100</td>
<td>$8,400</td>
<td>50%</td>
</tr>
<tr>
<td>Screen Digest ($MM)</td>
<td>$1,100</td>
<td>$4,200</td>
<td>31%</td>
</tr>
<tr>
<td>North American Gaming Revenue</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>IDC ($MM)</td>
<td>$160</td>
<td>$1,700</td>
<td>60%</td>
</tr>
<tr>
<td>ARC ($MM)</td>
<td>$119</td>
<td>$1,200</td>
<td>59%</td>
</tr>
<tr>
<td>US Mobile Gaming Revenue</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Instat/MDR ($MM)</td>
<td>$91</td>
<td>$204</td>
<td>$1,790</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>


Although Japan and Korea represent the vast majority of revenue generated by mobile game sales, Screen Digest predicts that The Americas and Europe will play an increasing role in the growth going forward and will lead the world in mobile gaming revenues over the next four years.

<table>
<thead>
<tr>
<th>Percent of Worldwide Gaming</th>
</tr>
</thead>
<tbody>
<tr>
<td>2003</td>
</tr>
<tr>
<td>---</td>
</tr>
<tr>
<td>Europe</td>
</tr>
<tr>
<td>The</td>
</tr>
<tr>
<td>Japan</td>
</tr>
<tr>
<td>South</td>
</tr>
<tr>
<td>Other</td>
</tr>
</tbody>
</table>

Source: Screen Digest
Over the last several years there have been two significant changes in the landscape for wireless gaming, and both are related. First, is the potential size of the market and second is the availability of games.

The potential market for mobile gaming has grown considerably since 2000. According to ITU, by mid-year 2004, total worldwide subscribers totaled 1.5 billion, double the size it was in 2000 and up from 1.3 billion by the end of 2003. Nokia believe the worldwide subscriber base will grow to 2.0 billion by 2007. In the U.S., according to the CTIA, there are approximately 173 million mobile subscribers. U.S. mobile phone penetration is estimated to be approximately 58%\(^1\).

### Mobile Subscribers: Source ITU

<table>
<thead>
<tr>
<th></th>
<th>1998</th>
<th>2003</th>
<th>CAGR</th>
<th>% of Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Asia</td>
<td>108,320.6</td>
<td>543,153.4</td>
<td>38.1%</td>
<td>40.5%</td>
</tr>
<tr>
<td>Europe</td>
<td>104,382.0</td>
<td>441,234.9</td>
<td>33.4%</td>
<td>32.9%</td>
</tr>
<tr>
<td>Americas</td>
<td>95,066.8</td>
<td>288,219.9</td>
<td>24.8%</td>
<td>21.5%</td>
</tr>
<tr>
<td>Africa</td>
<td>4,156.9</td>
<td>50,803.2</td>
<td>65.0%</td>
<td>3.9%</td>
</tr>
<tr>
<td>Oceania</td>
<td>5,748.5</td>
<td>17,256.3</td>
<td>24.6%</td>
<td>1.3%</td>
</tr>
<tr>
<td><strong>World</strong></td>
<td><strong>317,674.8</strong></td>
<td><strong>1,340,667.7</strong></td>
<td><strong>33.4%</strong></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>1998</th>
<th>2003</th>
<th>CAGR</th>
<th>% of Asia</th>
</tr>
</thead>
<tbody>
<tr>
<td>China</td>
<td>23,863.0</td>
<td>269,000.0</td>
<td>62.3%</td>
<td>93.3%</td>
</tr>
<tr>
<td>Japan</td>
<td>47,307.6</td>
<td>86,658.6</td>
<td>12.9%</td>
<td>30.1%</td>
</tr>
<tr>
<td>South Korea</td>
<td>14,018.6</td>
<td>33,591.8</td>
<td>19.1%</td>
<td>11.7%</td>
</tr>
<tr>
<td>India</td>
<td>1,195.4</td>
<td>26,154.4</td>
<td>85.4%</td>
<td>9.1%</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>86,384.6</strong></td>
<td><strong>415,404.8</strong></td>
<td><strong>36.9%</strong></td>
<td>79.7%</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>1998</th>
<th>2003</th>
<th>CAGR</th>
<th>% of Europe</th>
</tr>
</thead>
<tbody>
<tr>
<td>Germany</td>
<td>13,913.0</td>
<td>64,800.0</td>
<td>36.0%</td>
<td>14.7%</td>
</tr>
<tr>
<td>Italy</td>
<td>20,489.0</td>
<td>55,918.0</td>
<td>22.2%</td>
<td>12.7%</td>
</tr>
<tr>
<td>United Kingdom</td>
<td>14,878.0</td>
<td>49,677.0</td>
<td>27.3%</td>
<td>11.3%</td>
</tr>
<tr>
<td>France</td>
<td>11,210.1</td>
<td>41,483.1</td>
<td>30.0%</td>
<td>9.4%</td>
</tr>
<tr>
<td>Spain</td>
<td>6,437.4</td>
<td>37,506.7</td>
<td>42.3%</td>
<td>8.5%</td>
</tr>
<tr>
<td>Turkey</td>
<td>3,560.1</td>
<td>27,887.5</td>
<td>51.4%</td>
<td>6.3%</td>
</tr>
<tr>
<td>Russia</td>
<td>747.2</td>
<td>17,608.8</td>
<td>88.1%</td>
<td>4.0%</td>
</tr>
<tr>
<td>Poland</td>
<td>1,928.0</td>
<td>17,400.0</td>
<td>55.3%</td>
<td>3.9%</td>
</tr>
<tr>
<td>Netherlands</td>
<td>3,351.0</td>
<td>12,500.0</td>
<td>30.1%</td>
<td>2.8%</td>
</tr>
<tr>
<td>Czech Republic</td>
<td>965.5</td>
<td>9,708.7</td>
<td>56.7%</td>
<td>2.2%</td>
</tr>
<tr>
<td>Greece</td>
<td>2,047.0</td>
<td>8,936.2</td>
<td>34.3%</td>
<td>2.0%</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>79,472.3</strong></td>
<td><strong>343,626.0</strong></td>
<td><strong>34.0%</strong></td>
<td>76.1%</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>1998</th>
<th>2003</th>
<th>CAGR</th>
<th>% of Americas</th>
</tr>
</thead>
<tbody>
<tr>
<td>United States</td>
<td>69,209.3</td>
<td>158,722.0</td>
<td>18.1%</td>
<td>29.2%</td>
</tr>
<tr>
<td>Brazil</td>
<td>7,368.2</td>
<td>46,373.3</td>
<td>44.5%</td>
<td>8.5%</td>
</tr>
<tr>
<td>Mexico</td>
<td>3,349.5</td>
<td>25,928.3</td>
<td>50.6%</td>
<td>4.8%</td>
</tr>
<tr>
<td>Canada</td>
<td>5,365.5</td>
<td>13,221.8</td>
<td>19.8%</td>
<td>2.4%</td>
</tr>
<tr>
<td>Venezuela</td>
<td>2,009.8</td>
<td>6,463.6</td>
<td>26.3%</td>
<td>1.2%</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>87,302.3</strong></td>
<td><strong>250,709.0</strong></td>
<td><strong>23.5%</strong></td>
<td>91.8%</td>
</tr>
</tbody>
</table>

\(^1\) International Telecommunications Union – www.itu.int
For mobile gaming, the total subscriber base, while an indicator, is not as relevant as the installed subscriber base with data enabled handsets. IDC estimates that in 2004 there were 692 million 2.5G mobile devices shipped capable of data transmission.\(^2\) This figure doesn’t include 3G devices, yet represents almost one-third of the worldwide subscriber base.

B. \textit{Recent Market Trends}

1. \textbf{Traditional video game publishers jump in}
   
   Over the past few years, video game publishers such as THQ and Electronic Arts have begun to spend on mobile gaming as a way of increasing revenue. Some publishers such as Activision have taken both a development and licensing approach to mobile gaming to test the waters. Disney has also indicated early success as it will now dedicate more resources to developing games for mobile device. 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. \textbf{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. The following table shows the number of games recently offered by five of the top US carriers. Many publishers are now starting to provide some of their more popular catalog titles on mobile platforms. In some cases, the wireless games pattern themselves after the console titles, such as \textit{Tony Hawk’s Underground} and \textit{Tiger Woods PGA Tour 2004}, and the “classics” such as \textit{Galaga} or \textit{Pitfall}. There are also unique movie games and adaptations of parlor and card games to the mobile space like \textit{Texas Hold’em}, \textit{Pub Pool} and \textit{Darts}. There are even unique attempts at multiplayer and community based products.

<table>
<thead>
<tr>
<th>Number of Games Offered</th>
</tr>
</thead>
<tbody>
<tr>
<td>Verizon</td>
</tr>
<tr>
<td>380</td>
</tr>
</tbody>
</table>

\(^2\)\textit{idc.com --}
Directly related is the dramatic increase over the past two years in the number of games available from the various wireless publishers.

<table>
<thead>
<tr>
<th></th>
<th>Mforma</th>
<th>Jamdat</th>
<th>THQ</th>
<th>Airborne</th>
<th>Disney</th>
<th>Sorrent</th>
<th>Gameloft</th>
<th>Dbi</th>
</tr>
</thead>
<tbody>
<tr>
<td>Total Apps</td>
<td>118</td>
<td>73</td>
<td>66</td>
<td>48</td>
<td>35</td>
<td>42</td>
<td>40</td>
<td>35</td>
</tr>
<tr>
<td>Average Price</td>
<td>$3.21</td>
<td>$2.76</td>
<td>$3.53</td>
<td>$3.03</td>
<td>$3.16</td>
<td>$4.62</td>
<td>$3.64</td>
<td>$3.67</td>
</tr>
</tbody>
</table>

**Average Price** $3.45

*Source: WGR*

### 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. An example of a cycle of improving quality driving and being driven by improving technology can be seen in the evolution of the Internet. As more people gained broadband access more complex content became readily available. As more content was offered, more people wanted broadband access. The same cycle is occurring now and should continue to occur with mobile devices and gaming. 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. To put this in perspective, according to Gamewinners.com (11/1/04) Sony has sold just over 32 million PlayStation2 consoles in the U.S. since October 2000, whereas well over 60 million mobile phones were sold in 2005 alone. One indication of the expected continued growth of mobile games is the venture capital funding being received by proven players as well as new start ups and a significant increase in consolidations. Below is a partial list of recent equity investments and mobile game acquisitions.

<table>
<thead>
<tr>
<th>Date</th>
<th>Company</th>
<th>Funding (in millions)</th>
<th>Investors</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nov-04</td>
<td>Digital Bridges</td>
<td>$18</td>
<td>Apax, Argo</td>
</tr>
<tr>
<td>Aug-04</td>
<td>Mforma</td>
<td>$19</td>
<td>Draper</td>
</tr>
<tr>
<td>Aug-04</td>
<td>Digital Chocolate</td>
<td>$13</td>
<td>Sutter Hill</td>
</tr>
<tr>
<td>Aug-04</td>
<td>In-fusio</td>
<td>$27</td>
<td>Insight Ventures</td>
</tr>
<tr>
<td>Jul-04</td>
<td>Sorrent</td>
<td>$20</td>
<td>BA Ventures</td>
</tr>
<tr>
<td>May-05</td>
<td>Sorrent</td>
<td>$20</td>
<td>Granite Global</td>
</tr>
</tbody>
</table>
### Mobile 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 x7000 in 3D must be completely re-designed to work within the severe constraints of a Nokia 3220 in 2D. 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. Seattle-based PopCap is the big success story here, with its *Zuma* and *Bejeweled* having been published by JAMDAT and Sorrent respectively.

#### 2. Porting Service

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 such as Tira Wireless. This is likely to become the trend due
to the high cost of porting without a process that addresses handset diversity.

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. Pure-play mobile publishers include JAMDAT, Gameloft, Mforma,
Sorrent, Digital Bridges (now “iPlay”), Mobliss, Superscape and Digital Chocolate. THQ was the sole
representative of the console publishers for several years (depending on how you view the
Ubisoft/Gameloft relationship), but now the rest of them are starting to get involved.

4. Handset Manufacturer
Nokia put the first mobile game, Snake, onto one of its handsets in 1997, thereby creating the first mobile
game console. 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 and DoJa. 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
The eight-hundred pound gorillas in the mobile gaming industry are the wireless carriers, who are the
equivalent of retail outlets in the traditional video game space. This group includes the large multinational
carriers such as Vodafone, T-Mobile and Orange as well as the larger national carriers such as Cingular,
Verizon, Sprint and Nextel. 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. It is important for developers and publishers to understand carrier culture, which includes their
business practices and, most of all, that staffing in the gaming organizations can quite reasonably be
compared to skeleton crews. The more hand-holding a publisher requires from its carrier account the less
likely they are to earn the label of “partner”. A critical point to understand is that carriers today – at least in
the U.S. – are not looking to create additional partnerships. Going back to the “skeleton crew” point, they
feel they are well-served by the existing publishers. This will have the effect of stabilizing part of the value chain, which has both positive and negative results. 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 Hllumoto. 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.

7. **The Money Trail**
Knowing how billing for mobile games works and where the money changes hands will help developers build realistic budgets and cash flow expectations. The high-level money trail is as follows:
Consumer views the game storefront for his or her carrier on their handset – Media Net for Cingular Wireless subscribers, for example, and Get It Now for Verizon customers. When the user purchases a game it is charged to their wireless bill.

Approximately 30 to 90 days later, the game publisher receives its share of the sales revenue. This amount can run anywhere from 90% of revenue on Japanese carriers to 50% of revenue from carriers in Europe and often depends on the perceived value of the brand, the publisher and carrier co-marketing efforts. In the United States the average publisher revenue share ranges from 65-80%.

Developer royalties are then paid with thirty or forty-five days of the end of the month or quarter in which the publisher is paid. This means the developer should expect no royalties until six or seven months after the first game is sold.

This situation is no different than the cash flow model in internet or retail game distribution, and it is the reason publishers are expected to pay an advance in order to earn the right to (exclusively) distribute a game.

8. Billing Models

The wireless purchase environment facilitates commerce in a few very important ways:

- Charge to phone bill: consumers can put their game purchases on their phone bills and therefore don’t need to enter credit cards or other personal information. This “trusted source” for billing is very important for the end user.
- Relatively low price: a game price is generally low, facilitating impulse purchases.
- Store open 24/7: gamers can purchase, download and play games whenever and wherever they want.

This makes purchasing of games very easy. The same mechanism that facilitates the purchase has various options. For example:

<table>
<thead>
<tr>
<th>Offer Models</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>One Time Purchase</td>
<td>Pay a one time price for unlimited play.</td>
</tr>
<tr>
<td>Time Defined</td>
<td>Purchase an authorization for the use of game for a defined time period. Normally beginning from the time of the initial purchase.</td>
</tr>
<tr>
<td>Usage Defined</td>
<td>Purchase an authorization for use of the game for either a set time or number of sessions.</td>
</tr>
<tr>
<td>Subscription</td>
<td>Agree to pay a monthly recurring charge for access to content.</td>
</tr>
<tr>
<td>Multi-pack</td>
<td>Purchase an authorization for use of a number of products for an unlimited or time defined period.</td>
</tr>
<tr>
<td>Free Trials</td>
<td>Authorization for access to a product for a set time or number of gaming sessions.</td>
</tr>
<tr>
<td>Pay for Play</td>
<td>Purchase an authorization for a single gaming session.</td>
</tr>
</tbody>
</table>

Today, one-time purchase and monthly subscription are the dominant models, but these billing and access models create some interesting possibilities for publishers and carriers to tailor purchases to increase revenues. In the near future, consumers will have the option of purchasing new content for current games already on their handset through a mechanism called micro-payments. For example, a fan of Tony Hawk’s Underground could effectively purchase a new skate park and download it with along with new equipment or new players. As the technologies enabling these models are rolled out and publishers have an incentive to provide new content on a regular basis, the depth of the mobile gaming experience will deepen considerably.
9. Brands & Marketing

Marketing of mobile games is significantly more challenging than other forms of consumer product marketing. In this emerging market, it is difficult to identify marketable segments of the consumer group and normally inefficient and costly to reach them. Nonetheless, the goal is to increase awareness, which in turn will stimulate the interest in sales, and ideally the sale itself. There is no easy medium for driving awareness and educating consumers about the availability of mobile games, but the mediums best suited for it seem to be Web and print – both allow for more information about the product to be communicated to the end user. For example, SMS ordering of games via special short codes might be the easiest method of access, but it is only supported by a few carriers in the US.

Carriers can bring a lot of marketing muscle to the games they choose to market by providing exposure at point of purchase, in bill inserts and strategic placement in the storefront, or “deck”.

When the consumer arrives at the storefront, they are presented with many options to choose from. Games are normally listed in a 25 character line of text on the screen. The purchase environment does not allow for much, if any at all marketing of each game, such as screenshots, gameplay videos, customer feedback, etc. When a consumer only has this one line of text to choose from, purchases tend to be driven more by brands and naming conventions, which both leverage and create consumer expectations.

Chances are your favorite game was never a movie or book: WarCraft, Halo, Doom, The Sims, Half Life, and Grand Theft Auto. However, if you look at a recent listing of the top 10 video games what you find is sequels, movie brands and the latest season’s sports title update, so this trend towards branded games is not unique to mobile:

<table>
<thead>
<tr>
<th>Rank</th>
<th>Title</th>
<th>Publisher</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Need For Speed: Underground 2</td>
<td>EA</td>
</tr>
<tr>
<td>2</td>
<td>Grand Theft Auto: San Andreas</td>
<td>Rockstar</td>
</tr>
<tr>
<td>3</td>
<td>Call of Duty: Finest Hour</td>
<td>Activision</td>
</tr>
<tr>
<td>4</td>
<td>The Incredibles</td>
<td>Disney</td>
</tr>
<tr>
<td>5</td>
<td>FIFA 2005</td>
<td>EA</td>
</tr>
<tr>
<td>6</td>
<td>Prince of Persia: The Warrior Within</td>
<td>Ubisoft</td>
</tr>
<tr>
<td>7</td>
<td>Goldeneye: Rogue Agent</td>
<td>EA</td>
</tr>
<tr>
<td>8</td>
<td>The Getaway: Black Monday</td>
<td>Sony</td>
</tr>
<tr>
<td>9</td>
<td>Pro Evolution Soccer</td>
<td>Konami</td>
</tr>
<tr>
<td>10</td>
<td>Halo 2</td>
<td>Microsoft</td>
</tr>
</tbody>
</table>

The top 40 titles “brand profiles” on Cingular were as follows:

<table>
<thead>
<tr>
<th>Month (2004)</th>
<th>% Titles with Brands</th>
<th>% Revenues from Branded</th>
</tr>
</thead>
<tbody>
<tr>
<td>November</td>
<td>65% (26/40)</td>
<td>61%</td>
</tr>
<tr>
<td>October</td>
<td>67.5% (27/40)</td>
<td>69%</td>
</tr>
<tr>
<td>September</td>
<td>62.5% (25/40)</td>
<td>70%</td>
</tr>
</tbody>
</table>

The data above shows that unbranded content can generate revenues on par with brands given that other success factors are equal for all titles such as porting, production quality and fun factor (which, of course, in practice they are not). However, it’s important to look at what exactly these “unbranded” games are. Solitaire may, on the surface, seem to be unbranded; the fact is that a lot of people love to play solitaire. It is, in effect, a brand in the public domain. Would a game called “Joe’s Pizzeria”, a really fun game about making pizza, see the same sales numbers as a good Solitaire game? Probably not.

Other elements which are unique to mobile game sales include:
The deck clearly does not provide a good merchandising environment (i.e., there is no box art to capture shoppers’ imaginations).

The mobile game experience itself is shallow which does not lend itself to a try-and-buy model. If the mobile gaming ‘fix’ or fun element is satisfied in a 2-4 minute demo, why would the consumer buy that game?

Mobile game reviews are not easily accessible in the purchase medium. A gamer can easily read reviews on their favorite gaming site before making a purchase at their favorite online retailer. However, the combined factors of slow connection speed, less than optimal browsing experience and low price points are not conducive to this behavior for mobile game purchases on mobile phones.

No platform in history has had so many for-pay games developed for it so quickly. (Internet gaming content grew more quickly but was largely based on free content until recent years.) Within the span of just a few years hundreds of games have been created and the competition for the customer is intense. Add to that the fact that the merchandising of the products is still finding its footing trying to reach customers who for the most part are largely unaware that they exist and you begin to get a sense of the challenge.

The “brand mania” perpetuated by the small, monopolized carrier deck is considered by many people to be the fundamental flaw of the mobile marketplace. It makes the case for creating break-away games like Doom, WarCraft, The Sims, and Grand Theft Auto that much harder to make. It is such break-away games that have, historically, fueled the growth of the overall video game market. In console and PC gaming we have a thriving reviews business as well as downloadable demos; purchase decisions are premeditated and based on recommendations from friends. The internet download model is based fundamentally on the try-before-you-buy model. The carriers tried offering free trials briefly, and ran away quickly when sales plummeted. A solution needs to be found that puts the best games in the hands of the consumers who want that sort of game. The shot-in-the-dark model that carriers inflict on consumers today does not maximize consumer satisfaction and therefore stifles the true revenue and growth potential of the mobile games market.

D. Consumer Trends

Millions of people already have a mobile gaming platform in their pocket. Mobile gaming is on the rise. This much is known. But how many people, exactly, are playing games? How many people play vs. pay? Who is buying these games? Where do they live? What do they play?

Unfortunately, there is a dearth of sales data at this point in the industry. For competitive reasons, carriers are reluctant to share research, gaming habits and sales. Mobile analysts focus on industry numbers: companies, revenue, handsets, technologies, but not on consumers.

According to a new mobile data company, MMetrics, in November 2004 over 10 million mobile subscribers downloaded and paid for games on their phones. In contrast, 47.1 million played games that were already embedded on the handset. Clearly, more attention needs to be focused on the purchase path that converts these active players into active buyers. Consumers frequently report that buying is inhibited by poor merchandising – they simply don’t know what mobile games are available for their phones, or by their carrier. People still learn about games through word-of-mouth primarily, not realizing how much information is available on the web, from their carriers and gaming publications.

In February 2004, Sorrent and the U30 research company conducted a quantitative research study with 752 respondents, ages 9-35, split evenly between men and women, and the findings made it clear that mobile gaming is not new. In fact, mobile gaming has been a frequent “time killer” since Snake was loaded onto handsets in 1997.

In terms of demographics for mobile gamers, most people assume that it’s one or the other; either core gamers playing console-like ports or a mass consumer market bored in their ‘downtime’. The common thinking seems to be that mobile gamers are people who have no choice but to play a retro game for 3-5 minutes while they wait in line, are stuck in an airport or stuck in class.
In truth, the data shows that both “hard-core” and the casual are playing mobile games – and due to the rapidly growing installed base there is room for many types of mobile gamers – and games.

The study revealed frequent mobile gameplay habits that reveal more than just casually killing down time: people are playing at home (more than 60%); playing for long periods of time (15-20 minutes) and playing frequently (more than 65% play more than once a day).

What the Sorrent research says:

- 60% play games once a day or more
- 30% play games more than 3 times per day
- More than 60% typically play games at home
- More than 70% play for longer than ten minutes at a time – including a subset who play over 20 minutes (and as much as two hours) at a time
- More than 60% are between ages 18-26 and single
- More than 70% are sharing mobile gameplay on their phones with friends or family members
- Most people who download games learn about them through “word-of-mouth”

There is also enormous cross over from console to mobile with more than 80% of mobile gamers owning a console. Almost 20% of PC and console video game owners have purchased a game for their cell phone, according to a Ziff Davis research study. They spend an average of 17 minutes playing mobile games per session and 4.4 hours per week. Part of this is no doubt linked to the fact that video gaming has become part of the entertainment fabric of young people.

It is common belief that as the mobile market continues to grow; much of the growth will come from people who are under the age of 25. For these young people they will never see a mobile phone as simply a device for making voice calls. As a result, some of the carriers in the U.S. have focused entirely on this demographic in their marketing. They recognize that by focusing on people 18-24 years old they not only can project a hipper image, but also capitalize on the fastest growing market for texters, gamers and ringtone downloaders around. Mobile data grows ARPU and develops brand loyalty.

Young people are constantly downloading new content -- ringtones, wallpapers, games -- as it keeps them in the know. Mobile is the ultimate lifestyle accessory - synonymous with personalization, entertainment and reflective of their fickle, ever changing tastes.
IV. The Future of Mobile Gaming

We spoke with two of the leaders in the mobile game business regarding their vision for the future.

A. Q&A with Greg Ballard, CEO of Sorrent.

Q: How can developers expect the business climate for them to evolve from a publisher’s point of view?

A flight to quality: I believe the carriers when they say quality will become more important in 2005. I believe it in part because publishers like us intend to focus on even greater quality than we delivered last year. I believe it because this has been the trend in the larger video game market over the last few years, and the mobile market will reflect – on a greatly accelerated basis – these same dynamics.

More will be better and ultimately will drive the adoption of gaming on handsets by consumers. This means that developers have got to push themselves to be prepared to deliver quality – in game play, innovation and graphics – even if that means they need to demand more resources to do it.

A trend toward publishers creating or buying in house studio capability: we have already seen this trend developing in 2004, and it will increase in 2005. Publishers have begun to realize that they can deliver high quality titles on reliable timetables only if they have a certainty of their “supply chain”. The uneveness in the capabilities of many studios has created a sense of uneasiness among publishers who rely extensively on outside developers. And the sudden acquisition of some of the most successful developers by several leading publishers has sent tremors of nervousness through the publishing community. Who will be next and how will that affect my ability to deliver titles that are central to my financial forecast?

These are the very dynamics that have convinced Sorrent to stay with its exclusively internal development model. It has convinced other publishers to begin at least the process of bringing many titles in house. For the remaining developers, many of whom are highly capable, the time has come to start developing longer term relationships with publishers and to consider whether the time has come for them to also consider being acquired.

It’s not just about games anymore: Several industry leaders have spoken about the move toward non-game centric applications, efforts that will seek to define the unique usage of the mobile phone for entertainment. Many other companies have been actively working to develop such applications. In 2005, we expect to see a number of applications that use gaming as a core feature but offer other mobile tools to enable interaction with a community of like minded people.

To remain relevant in this next phase of creative exploration, developers need to push their creative teams to think “outside the screen”. What else can we do to entertain people with this as the platform? What has worked in Asia and how can that be imported to the US and Europe? It is just as likely that these questions can be answered by a highly motivated, intensely creative development group than by one of the producers at a larger publisher. Make this an important part of your creative strategy for this year. The killer app has not yet been discovered.

Q: What will happen with development budgets and game complexity?

We are already seeing the pressures to increase development budgets, and we expect to see the average dollars spent to develop a title nearly double this year. This trend is a function, we believe, of the need to develop titles that are richer, more complex, and able to utilize the increased capability of the hand sets.

But there are two other trends that will accelerate these cost increases. First, we include porting and customization costs in our development expenses. There is no doubt that this year will see more handsets introduced into the market and therefore, more ports required to support the launch of a title.
Even though the cost of each handset port can be managed down over time, the effective cost of the overall ports per title will increase due to the proliferation of handsets.

Second, the advent of 3D technology will significantly increase the cost of development. 3D development is by itself expensive – more detailed art, more expensive technology development. But during the transition from 2D to 3D, it will be necessary to manage twin development costs, an undertaking with enormous implications for managing development budgets.

Q: How will handheld devices such as Sony PSP, GBA and other emerging hybrids impact the mobile game business?

The initial reports of the new Nintendo and Sony handheld devices have been extremely favorable. Gamers have reported a nearly religious experience playing with the high powered, graphic intense devices. There is no way that mobile phones can match the game playing experience these devices are delivering today.

Mobile phone devices will always lag in the entertainment experience that purpose-built game devices can deliver. But they need not compete in capabilities in order to have their own place in the entertainment eco-system. Mobile phone-based games will be successful because they can be played at times when a user may not have thought to grab his or her PSP or when carrying a device like that would simply have been inconvenient or impractical. The point of cell phone based gaming is that it permits users to play games when they otherwise would not have a gaming device with them. Mobile phones are ubiquitous and therefore make gaming ubiquitous.

Other gaming devices are not similarly spontaneous. They may provide a superior gaming experience, but only for someone who plans in advance to bring his or her game device and then has a setting that permits full-on game play. For all those other game playing times, the cell phone is the perfect device.

B. Trip Hawkins, CEO of Digital Chocolate

Advice on Becoming a Mobile Game Developer

“Applications need to look for the true heartbeat of mobile. Hollywood thinks it’s a tiny TV. Silicon Valley thinks it’s a PC in your pocket. The video game industry thinks it’s a gimpy GameBoy. Mobile does not need to be the stepchild of another medium. Surely the heartbeat of mobile is elsewhere. It’s a new medium.

Our industry has been talking about multiplayer and massively multiplayer as the nirvana of gaming business models for a long time. As the handsets and networks are more capable of robust gaming experiences, casual, massively multiplayer games will launch on carrier networks and grow the market to new levels.

Mobile gaming has the potentially to reach that nirvana and as the market grows potentially we will see a mobile version of every top PC and console video game made.”
V. Mobile Game Content

Mobile phones offer a unique game platform due to their compact size, personal nature and connectivity. Users can play mobile games by themselves or with others all over the world. Consumers can learn about, shop for, and buy games anywhere there is wireless coverage. The “store” is open 24 hours a day, seven days a week, and travels with the user wherever they go. This completely changes the standard consumer purchasing dynamic and presents significant new opportunities for game play and new genres.

A. Game genre breakdown and analysis

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:

- **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.

B. Brands in Mobile Games

Given the maturity of the console business, you would expect mobile to follow suit, but in actuality mobile remains more of a general consumer category with games like Jamdat Bowling, Jeopardy, Who Wants to be a Millionaire?, and Tetris continuing to dominate downloads. Here are the main brand categories in mobile gaming:

- **Game Show Games**: A perennial favorite that does very well in all types of game categories. This genre also lends itself very well to a subscription model (monthly payment plan) because the games have high replay ability and an ever-expanding catalog of content. In addition to single player versions, Game Show Games allow for great multiplayer, turn based games which provide just another compelling reason for users to keep subscribing.
Movie Games: By and large, movie-themed mobile games have been disappointing, leaving carriers a bit skeptical of this category. There are often licensing related issues that may delay launch of a game, which can jeopardize promotional plans and sabotage sales. If you are considering a movie-themed game product it’s important to be in step with the studio, get all necessary approvals in a timely fashion, confirm operator’s interest, and above all develop a fun game with good play dynamics which truly captures the flavor of the movie.

Board Games: Many popular classics have already been adapted to mobile, and there are plenty more board games that can work on the handset. Generally these games are great for low-end devices, as they require little in the way of animation, sound, etc. and are received well by the casual consumer. The brands are often well-known, but contrary to movie games they lend a great deal of positive value to game design/play because they bring along implicit game play mechanics that are often understood already by consumers.

Console Games: Becoming more and more popular, console conversions are showing up on mobile handsets everywhere. They present a unique challenge, though, as players and defiantly reviewers expect the game to approximate the console title. This is a very high standard to meet.

Private Label Brands: Emerging markets always offer an opportunity to formulate new brands, but these opportunities are quickly disappearing. Given the fact that the entire purchase decision is made by picking the game from a 25 character name on the handset, brands heavily influence the decision. For better or for worse, consumers will always be more comfortable looking to brands to deliver a good entertainment experience.

Adult Games: So far US carriers have been cautious and have not ventured into this space. There is clearly demand, so expect users to look off-deck for this type of content. In fact, adult games might prove to be catalysts that help drive alternative distribution models.

C. Interactive Gaming / Connectivity Features

One major difference in mobile gaming is the simple fact the console travels with you wherever you go. This presents opportunities for truly unique game play idea to be added;

Competitive gaming (asynchronous): Many trivia and game-show format games employ this type of competition already. Participants join a “round” which may include a handful or a hundred competitors who are all given the question or puzzle at the same time. Winners move on until there is one winner left, or the number of questions is exhausted. One advantage of this type of connected gaming is that is it not dependent upon network speeds.

Competitive gaming (head to head): As network speeds and handset capabilities improve, we will see more true multiplayer games. Jamdat’s Bejeweled Multiplayer is a good example that pits players against each other in a real-time contest. This game requires some very specific handset characteristics, and was creatively designed in such a way that users are not aware of any delays caused by network latency.

LBS gaming: Location Based Services have been the buzz on and off. The only commercial deployments of games which leverage location are in Europe. The concept may very well work in the U.S., but there is hesitation on the part of U.S. carriers to provide competitive play that includes user location as it could introduce liability and violate privacy. There are also many technical issues in making this implicit capability a practical reality.

Tournaments for prizes: More publisher / operators are looking to build a community of players and to extend their 1:1 relationship with their consumers. One way to provide incentives for these activities is to host tournaments and supply prizes. There many complications with these games such as sever hosting, security, state laws regarding giveaways, and reporting. It’s an area that will truly increase download and/or subscriptions that will be expanding.

Social Games: Another way to build communities are with social/interactive games that focus on generating SMS interactivity.
D. **What are the challenges in launching a successful mobile game?**

There are many issues associated with launching a successful mobile game, but attention to details and proper due diligence will assist you in avoiding major pitfalls. The challenges that are most common include:

1. **Distribution**

The vast majority of mobile games sold are found and downloaded through the carrier “decks”. These are closely managed catalogs of game content that provide the most direct path to the mobile user. In order to have a hit mobile game, it is important to have broad distribution and engage several major carriers. This most likely involves having a direct carrier relationship. Establishing a direct relationship with carriers can be a challenging proposition. U.S. carriers are very selective when it comes to direct deals, and the largest ones have said directly that they are not looking to form new direct relationships, so in almost all cases it is necessary to bring new content through an established publisher or aggregator.

Regardless of how you get there, there is a direct correlation between the placement of each game and its commercial success. A game placed in a promotional category such as “What's New” or “Top Games” will generally yield 3-5 times the sales of a game placed within genre categories. These positions are limited and generally are available for top games only and for a limited period of time. After this limit expires, these games are normally placed in the appropriate genre category. With so many new games becoming available each month, even the large publishers are starting to see their older, less successful games drop off the deck entirely. The lesson here is that “shelf space” is as real a concern in mobile as it is in brick-and-mortar retail.

2. **Porting**

Unlike the console game space, where there are a handful of important platforms, in the mobile gaming space there are hundreds of devices which each having unique characteristics and challenges. It is not necessary to port to every device, but exactly which minority of handsets make up the majority of the sales is sometimes hard to find out. This makes understanding which devices are important based on your carrier partners’ goals, sales volume, and the quality of game experience offered by each device crucial. Those publishers who are able to navigate the slew of new handset launches and support these handsets as they are released will see their share of mobile game sales grow dramatically.

In his interview above, Greg Ballard communicated that Sorrent's strategy to address this issue is to develop and port games using internal studios, so as to control more of the variables. Alternatively, other publishers are choosing to outsource porting, which can be an effective way to get games to market on time for companies who do not have the means to invest in large internal development studios. Unfortunately there seems to be a shortage of porting services which has resulted in pent up demand and delayed game launches.

3. **Testing/Certification**

Each carrier has its own process for testing and certification of new game content. The best way to get up to speed with the policies and processes is to join the developer programs for major carriers. Here are some helpful links:

- **Alltel** [http://portal.alltel.net/alltel/portal/excess/apps/developersCenter](http://portal.alltel.net/alltel/portal/excess/apps/developersCenter)
- **Nextel** [http://developer.nextel.com/portal/index.jsp](http://developer.nextel.com/portal/index.jsp)
- **Sprint** [http://developer.sprintpcs.com](http://developer.sprintpcs.com)
- **Verizon** [http://www.vzwdevelopers.com](http://www.vzwdevelopers.com)

Regardless of the carrier or technology, you should expect each game will take a minimum of 30 days to be reviewed, tested, and certified by the carrier. It is not unusual for games to sit “in queue” for 90 days or more before being placed on deck. Publishers are giving guidance that new games today take seven to eight months from “go” to launch.
VI. Mobile Devices

Perhaps the biggest difference in the mobile game business from the developer perspective is the wide range of devices and platforms that the games are commonly deployed on and their idiosyncrasies. To further understand these issues it is important to understand how the mobile device has developed.

A. Mobile Device Evolution

1. Color Displays

When color displays were added to phones, display sizes and resolutions also began to grow. The result has been a marked annual improvement in displays, which are perhaps the most critical element in presenting games. The technology brought games to life, allowing developers to create graphics which jump off the screen. From a consumer perspective, the color display had two important effects. First, color brought phone displays to same technology generation as displays on other game devices so mobile games no longer appeared archaic. Second, color displays alone drove an entire global phone upgrade cycle. Finally, the new phones included not only color displays, but many of other the milestones listed below as well. The net result is that color displays were the driving feature in seeding the current market with phones capable of playing mobile games as we know them today.

2. Connectivity

The dawn of messaging connectivity via SMS was an important milestone because it enabled the first generation of games on phones and presaged the coming of multi-player games. By messaging with an application server, a user could play a single-player game, compete against a real or virtual opponent, or participate in tournament. Later, messaging evolved to allow in-game communication and sharing of high scores during more advanced multi-player games.

More advanced data connectivity, such as WAP and HTTP over GSM/CDMA networks (the two major types of signals used to carry mobile phone communications) and later over the more advanced GPRS/1X networks and now over EDGE/UMTS/DO networks, enabled an entire distribution channel for mobile games. These new networks allowed mobile games to be downloaded to the phone as an application for permanent storage and be purchased by the subscriber. As networks improved their coverage, bandwidth, and stability, more and more users were connected to a content store which could be updated and refreshed daily.

3. Development Environments

Development environments such as Java, BREW and Symbian have matured to the point that developers can create their own games and applications on mobile phones with much greater reliability. Previously, some environments were controlled by standards organizations, while others were defined by single firms or a small group of companies. Prior to the proliferation of Brew and Java, SMS games were played via the network, limiting the presentation on the phones. Consistent playback environments allowed real-time game engines to deliver quality graphics, game logic, audio, and interactions: mobile phones were opened up as platforms for game play. Varying implementations of standard environments and multiple proprietary environments has led to platform fragmentation – even the implementation of J2ME is not consistent from handset to handset. So many environments are available in the marketplace today that it has fragmented development efforts to an incredible scale: a single title must be ported across several dozen handsets to fulfill a complete launch in a single region. This fragmentation has led developers to compete on their ability to deploy games on hundreds of devices, and to focus less on game quality/design.

4. Joysticks/D-pads/Nav-key Input Devices

As phones began to support more advanced applications and user interfaces, device makers added buttons beyond those needed for voice calls. Importantly for games, these additions included input devices that serve as joysticks and d-pads (“digital” pads like the ones commonly found on console
controllers). In fact many devices now support 8-way navigation keys and multi-button press, allowing for a different result when two buttons are pressed simultaneously as opposed to serially.

These changes allow input mechanisms on phones to be more similar to those found on traditional gaming devices, and thus more familiar and comfortable to game players. Game play has become more advanced as a well, as players can use multiple keys to move more freely about the game screen.

5. Memory
On-device application memory (e.g., HEAP space, RAM) was made accessible to game developers simultaneously with the launch of J2ME and Brew support. The quantity of memory available to applications has consistently grown over time, and the bottleneck has generally been network capabilities: limits on downloadable games have grown from 30 to 50 kilobytes originally to 300-500 kilobytes for Java and 3.5 megabytes for BREW today; limits for embedded games or games played from memory cards are now 3-5 megabytes. The added memory allows for richer games and deeper games, even allowing some developers to embed their own technology engines in the games (e.g., 3D graphics).

In addition to providing better games, added device memory has allowed users to store more games on their phone at any one time. As recently as 2002, users could store as few as two games on their phone. Current devices can hold over a dozen games. This storage allows users to buy more games, expanding the market and enabling a distribution channel by providing a way for consumers to keep the content they paid for. Importantly, added memory also allows users to store games locally and play even if they are out of coverage, thus transforming mobile phones into entertainment devices.

6. Multi-channel and PCM Audio
Multi-channel and PCM audio (Pulse Code Modulation which converts analog audio to digital) allowed developers to play more than one stream of audio at a time and to include more advanced sound effects by leveraging digital sound codecs. These additions “complete” the user experience and fulfill the definition that most consumers have of a game: color graphics presentation, interaction via an input device, and quality audio. As technology has advanced, audio has evolved from missing, to an annoyance, to starting to be an important part of the game – similar to the evolution of audio on other video game platforms.

7. Software Support for 3D Graphics
The first 3D software solutions for mobile phones have been primitive but they demonstrate the speed with which mobile games have advanced. The first solutions were proprietary embedded engines, but today some developers are bundling their own solution with the games (delivered via memory card or over the air). 3D graphics have certainly generated attention for the mobile game industry and begun drawing in more traditional game publishers/developers. Today this technology and games using it are mostly of interest to carriers and device manufacturers who want to demonstrate the 3D capabilities of emerging mobile devices. Time will tell if 3D becomes a differentiator when it comes to sales.

8. Application Processors and 2D Graphics Processors
Added hardware support for applications and processing graphics were driven by games and multimedia requirements. In addition to off-loading work from core chipsets and allowing games to become more advanced and rich, added hardware support is an indicator of more specialized technologies that are coming to mobile phones. These technologies will take the game experience to the next level, and allow phones to become first-rate portable entertainment devices.

B. Device Trends
The complexity of the mobile device matrix isn’t decreasing. There continue to be new developments that add to the complication of developing games for the mobile market. Several of the most important trends in mobile phones are discussed below, along with their impact on mobile games.
1. **Additional Device Memory**

In a span of three years, application memory allocations have grown from 50 kilobytes to 3+ megabytes, and expectations are that they will keep growing to 5+ megabytes in 2005, and up to 16M or 32M within the next three years. Additional HEAP space and RAM improve the quality and depth of games, allowing developers to add more value to the content. Simultaneously, games this large have much higher development costs and will begin to be compared with content developed on game-first handheld platforms and traditional consoles. New distribution models and technologies will need to be leveraged to manage such large files (e.g., over-the-air, retail, local networks, PC-sync).

Memory for user storage also continues to grow at an exponential rate, especially via the inclusion of removable memory, micro-drives, on-line storage systems. This is a critical support enabler for the use and storage of richer and deeper content as games become larger and larger. Phones will begin to be viewed alongside consumer electronics devices – with implications on how devices, services, and content are marketed and used.

2. **Hardware Support for 3D Graphics**

GPU’s (“graphical processing units”) enable far richer and deeper content by off-loading 3D rendering tasks from core chipsets to special-purpose processors just like in modern consoles and personal computers. In line with other technology items that drive the game experience, GPU’s will also drive up development costs, and hence price points for devices and content – thus segmenting the market. It is important to note that this technology will also be used for other entertainment applications such as video and media.

3. **High Speed, Low Latency Network Connectivity**

Increasingly, mobile phones contain both local and wide-area network capabilities that deliver high-bandwidth, low-latency connections (3G, Bluetooth, WiFi). This connectivity will support new distribution models such as peer-to-peer distribution and content sharing. For example, users will send demo and gift content to each other. New game play modes will also be realized as high-speed, low-latency networks will enable true real-time games between devices and as part of larger networks with consoles, PC’s, etc.

4. **Growing Screen Resolutions**

Screen resolutions and sizes are growing throughout the price tiers. Although screen sizes had typically been 128w x 128h on the low end and 176w x 208h on the higher end, in 2004 QVGA (320w x 240h) displays were prevalent in the high tier and 128x160 displays were available in the low tier. More screen real estate and higher resolutions not only improve the experience in their own right, but they drive demands for other hardware components (memory, chipsets, etc.). As the trend continues, developers will need to design games for large displays and consumers who choose a phone based on the game experience. At the same time, developers will still need to support smaller displays and consumers who do not consider gaming part of their phone purchase decision. New games and genres will also be enabled on mobile phones, and at lower price points (e.g. solid 3D games and first-person shooters that simply do not work on small displays today).

5. **Native Development Environments and Middleware**

Beginning in 2004, native environments with middleware support began to be deployed on mobile phones (e.g., Renderware and X-Forge on Symbian). This trend will continue as more native environments will be deployed and middleware will support additional platforms. These technologies support very rich content experiences, but also enable developers to leverage work done on other platforms (e.g., console, PC). This has the potential to bring more traditional game publishers and developers into the mobile space as they see development hurdles lower and performance rise. It is worth noting, however, that although development hurdles will be lowered, content created on these platforms will be expensive to develop. This will result in higher development costs and retail prices in comparison to development costs for most mobile games, thus making the devices and content attractive to a segment of the market.
VII. Mobile Game Technology

A. Introduction

Technology platforms are those hardware systems and accompanying software components that make up the mobile phone handset application environments and are typically far more constrained than other gaming platforms, such as consoles or personal computers.

The mobile space also differs greatly from more traditional console or PC gaming markets in that the number of platforms are greater, more varied and the delivery mechanisms are more complex and present their own challenges for game design and distribution. Where a console game will only need to target two or three distinct SKUs (“Stock Keeping Units”), a mobile game could easily be built in three different programming languages and need to run on 200-300 different handsets.

Within this section, we will present the three major client-side (on the handset) systems for over-the-air (OTA) delivery of games and execution of content: Java (J2ME), BREW and Symbian. We will also describe the typical handset platform for these Major Operating Systems (OS) and what challenges developers face targeting each of these systems. There is also a brief description of some potential platforms that have yet to reach mainstream numbers of shipments: Linux, SavaJe and Microsoft Smartphone.

The latter part of this section focuses more on technology and covers new features, such as multiplayer, 3D and community and looks at how porting can be managed across multiple, varied platforms.

B. The Major Technology Platforms

1. Java

Java is a programming language, supporting toolset and runtime components created and maintained by Sun, Inc. While Sun oversees the specification of the language, the standardization process they have established enables components to be designed and created by a much wider range of companies and individuals.

Java applications execute on a synthetic processor called a virtual machine (VM) that is a native code application that runs on the actual device processor. This abstracts Java byte code from the host native code, allowing in theory portability of code between different cell phone architectures. Typically, Java applications are interpreted, or read and translated one Java instruction at a time by the VM. As a result, each Java instruction is translated into many hardware processor instructions, inevitably leading to slower execution rates for application code than an all-native code solution. However this is partially offset by the APIs themselves being mainly implemented in native code.

There are also acceleration technologies such as Just-In-Time (JIT) VMs that can speed up execution by retaining and re-using the native versions of previous executions of Java code. There are also more aggressive strategies such as Java-to-Native compilers and processor technologies such as ARM’s Jazelle system that run most, if not all Java code as native code. The end result is that a Java game can be a very satisfying experience for the end user.

The key to any gaming platform is not only the performance of the system, but also the richness of the APIs available to the developer. A process called the Java Community Process (JCP) creates and evolves new Java APIs with the actual work carried out within working groups called Java Specification Request (JSR) Expert Groups. It is these individuals or companies that join a JSR EG and that can contribute to the design of the new APIs. These APIs, once ratified/approved, are then available to be implemented by any company wishing to do so and are fully tested for compliancy. This means that the actual system software can vary significantly from handset to handset if there are multiple software vendors for any given JSR, and it is the compliancy testing that is used to maintain consistency across platforms.
The runtime environment for Java on handsets is served by the Java 2 Mobile Edition (J2ME) and standardizes on two major components, the Mobile Information Device Profile (MIDP) that defines APIs for mobile devices and the Connected Limited Device Configuration (CLDC), which defines the base set of APIs and the dedicated Virtual Machine – called the KVM (Kilobyte VM).

Java was initially marketed with a “write-once, run anywhere” message and although this has proven to be not entirely true, code is extremely portable. Java offers a well-designed programming language, excellent and with widely available tool for support and a huge user base for the developer to target.

Most implementations of the J2ME environment run on what are termed “closed” operating systems. In other words the end user does not generally have access to the file structure underneath, and when access is allowed it is through specific tools, such as application managers or media file browsers. This usually gives a reasonable amount of security for applications, preventing all but the most determined hackers from copying applications from the handsets. Of course there are always exceptions to this rule and some handsets do allow PC based browsers to examine the directories where games are stored.

Java is also deemed a “sandboxed” language; in other words a game runs within a managed, secure environment and actions such as writing files or accessing system features and functions are usually disallowed or severely limited. The fact that the Java code is not executed directly on the host processor allows a large degree of possible intervention when code goes astray – a good Java VM will not let a Java game crash a phone.

Java is and will likely be for the foreseeable future the dominant platform for cell phone game development. The fact that it isn’t tied to any particular OS and that VM and APIs can be obtained from multiple vendors makes it attractive to handset OEMs (“original equipment manufactures, e.g., Nokia). The wide availability of tools and comprehensive documentation mean that developers have multiple sources of technology, as well as having a large pool of developer talent to pull on. APIs continue to be improved and are more comprehensive and media-rich than before, so that high quality gaming experiences are now possible. Finally, Network Carriers have robust provisioning systems, so that delivery of content is not a barrier to use.

In terms of numbers of handsets shipped, Java is by far the most prevalent platform – Nokia stated in a press release in October of 2004 “There are 350 million Java handsets in the global market as of June 2004”. It is expected that hundreds of millions of Java enabled handsets will now ship annually, giving Java a 5 to 1 ratio of volume over its nearest competitor, BREW. This is very sizeable market that can be addressed by developers and consists of devices of varying performance and memory capabilities, as well as different APIs.

2. Developing for Java

A typical Java handset coming to market today is powered by a 32-bit processor, usually an ARM9 variant, and has a color screen, often QVGA size. Available RAM for these handsets is usually 500 kilobytes and up, with a few going over 1.5 megabytes, usually reserved for those with larger screens and perhaps more advanced functionality, such as 3D.

The first incarnation of the Java library set, MIDP 1, was quite limited in features and only allowed simplistic gaming. As a result most manufacturers added additional non-standard libraries for features such as sound or to control handset functions such as the vibrate feature. Developers were then faced with the increasing problem of multiple API’s and faced with the option of building to a least common denominator, which invariably meant a poor gaming experience, or have individual builds for each handset. Most took the latter approach and it wasn’t uncommon for a major title to have required the creation of 200-300 different SKU’s. For this reason, porting of games from handset to handset was, and still is a major issue for Java game developers.

From a code development standpoint, Java has several advantages and some disadvantages over native coding. First, the language itself is well designed and with MIDP 2 and other multimedia APIs available
through new JSR’s, is now feature rich. Access to these features is straightforward in most cases and developers can focus on game design rather than handset specifics for the most part.

Where Java proves challenging is in the wide variation in handset design, performance and support for what should be standard APIs. In spite of conformance tests, implementations of the many features of handsets differ and the Java game developer has to put portability of code at the top of the priority list if they are to make successful returns on their investment.

Memory management too can prove to be a central part of the game development process. While in theory a Java developer does not need to worry about handling memory directly through pointers or allocation APIs, the limited space on a handset and the need to optimize for performance means that a good understanding of how the Java VM memory manager operates is vital for good game dynamics. As an example, generally allocating memory during a game update is not a good thing to do, but sometimes a line of code can be allocating memory without it being immediately obvious; an example would be String functions where a simple assignment could be allocating new memory from the heap. There is some degree of control over when garbage collection takes place (the process of cleaning up discarded Java objects), but often this is not completely under the game developers control and sudden and irregular pauses in gameplay can result which can severally distract the user.

3. BREW

BREW is a runtime platform, a development environment, and application deployment system created by Qualcomm, Inc. that offers an alternative proprietary approach to the semi-open standards of Java. Whereas Java is run on a virtual machine and the byte code for the application is not tied to any specific processor design, BREW applications are compiled into ARM machine code. This approach offers an improvement in performance and finer control of code execution; the downside is the proprietary nature of the platform, the resulting royalties to Qualcomm, and the potential need to recompile and optimize for different hardware platforms.

BREW resolves the potential portability issue by being designed to run on a partially customizable hardware platform defined by the Qualcomm CDMA Technology group (QCT). It also has consistent and rich APIs, a central certification process to test and reliable delivery mechanism. In short, Qualcomm has put in place all of the components needed for the mobile application value chain so that it can offer the same, if not greater, level of portability as Java. The reality is that all games, whether Java, BREW or other system, need to be ported to different handsets as screen size, memory and keyboard layouts vary greatly from one handset to another.

From a coding standpoint, BREW offers a proprietary solution that seems to reduce flexibility by tying the developer into one vendor’s tools and technology, but in reality offers a consistent platform for game development, lowering costs and reducing the time to get a game to market. It also supports some of the largest download file sizes (read this as “bigger, more robust games”) and the most powerful handset designs, with the exception of one or two powerful Java handsets such as Motorola’s E680 or other native code platforms such the Nokia N-Gage. In general, BREW handsets are probably the most powerful platform today for phone-first mobile gaming.

BREW handsets also supports “sandboxing” of applications, so that most errors are trapped before the phone is crashed, but being native code there is always the possibility for games to cause some major handset problems. For this reason, applications undergo “True BREW Testing” to ensure that they are well behaved and not malicious or unstable. This provides a very useful one-stop shop for developers to get their games certified so that they can then be submitted for carrier testing and acceptance.

Where BREW really has been impressive though is in the provisioning of games, integration with the network carriers billing systems, and resulting support for the publisher and developer. Once a game is accepted by the carrier and placed on the game deck, the end result is the publisher can easily track downloads and monitor the success of the title.
BREW is the second most prolific platform for gaming, behind Java in numbers of deployed handsets. However, the robust and comprehensive end-to-end platform solution that has been put in place and maintained by Qualcomm means that it is often the easiest to use for game developers and publishers. In the U.S., Verizon sells more games than any other carrier, and Verizon is all BREW. Noting again that BREW handsets are also more consistent and usually more powerful than their Java equivalents, this results in the BREW platform being a very attractive system for mobile games.

Qualcomm and Zelos estimate that there are approximately 54 million BREW handsets worldwide in 2004, of which 26.7 million are in the US. Even though this is a smaller number than Java handsets, it is a rapidly escalating number and, like Java, consists of handsets of varying performance, memory and APIs depending on the BREW version being employed on the devices. In conclusion, BREW is renowned to be the system where developers and publishers can most rapidly get a return on their development costs.

4. Developing for BREW
Writing native code requires more detailed understanding of the underlying hardware if optimal code is to be produced. Key things that the developer has to understand and work with are:

- Processor performance
- Available RAM
- Data and Processor Cache
- Bus Architecture
- Any specialized hardware, such as DSP’s, 3D accelerators, Bluetooth modules etc.

In addition, every game should aim to fulfill the following requirements

- Generally, a well-behaved respected application within the system that releases shared resources when not required and not using processor cycles when idle
- Have a clear user interface, following all handset or carrier guidelines and use keys in an appropriate and logical way
- Respond to system events and save and restore state accordingly

Of course, on top of all this are the given requirements to be a robust, intuitive and above all, fun and addictive game!

On the positive side, the developer is working “on the metal” and can, if so required, have very fine control over many of the important aspects of the application such as memory management; optimal cache line performance and processor register allocation. A good compiler, such as those supplied by ARM, helps with these aspects too, often removing the need to write assembly code.

5. Symbian
The Symbian operating system is the main “Smartphone” OS and is most well known in the form of the Series 60 handsets from Nokia, Siemens and Sendo. It also appears in a pen-driven form as UIQ on the SonyEricsson P800 and P900 handsets. Symbian is also the OS in the most overtly gaming phone to date, the Nokia N-Gage.

In the three months to end of September 2004, it was reported that there were 3.7 million Symbian-based handsets shipped worldwide for the year, up from 1.2 million in the same quarter in the previous year. The total installed base of Symbian handset stands roughly at 9.2 million.

Symbian supports both Java and native programming, with a J2ME implementation on current handsets. Native applications are built to the Symbian API and compiled to ARM binaries, with almost all handsets being ARM 9 based devices. Symbian games are in general, portable between different Symbian handsets, but often need recompiling for different versions of the operating system and certainly need rework if moving from Series 60 to UIQ.
Java games can be downloaded over-the-air (OTA) to handsets as can native Symbian games, but often Symbian games are downloaded instead to a PC from an Internet portal and then installed over wired connections due to their inherently larger file sizes. On the N-Gage, games were available on a locked memory card, which although protected was initially found vulnerable to piracy as the file system was exposed. This vulnerability to piracy, large file sizes, and the much lower installed base of handsets, is the main barrier to the increased adoption of Symbian as a significant games platform.

Symbian is an excellent operating system and has featured some of the most graphically rich games seen in the mobile space, especially on the N-Gage handsets where download limits were not an issue. Many people are excited about the future of Symbian; however, for developers looking to create a game that can be very widely deployed in 2005/2006, look to Java and BREW.

6. Developing for Native Symbian

Symbian makes available a number of tools for application developers and the API itself is very rich and comprehensive. As with BREW, developers are working “on the metal” although in this case games are not subject to the certification that BREW applications go through, nor are they sandboxed like Java. As a consequence, this provides developers with enormous flexibility but it is vitally important for developers to apply rigorous testing of games prior to launch to insure bug free applications.

Symbian applications are generally fast, although the standard compilers are not as efficient at code optimization as some other commercial products. The key issues with Symbian games are the same as BREW games, and it is worth reminding yourself to keep games as small as possible so you can download them over multiple networks, manage memory well and have a well-behaved application.

Symbian platforms are far more consistent than other OSs, so once a developer has authored a game for a Series 60 handset, such as a Nokia 6630, then it will run on most, if not all other Series 60 devices. Another advantage of Symbian is that the heap, both native and Java, is adjusted dynamically and Symbian devices tend to be fast and loaded with memory, so while there may be a megabyte of heap on some Java handsets, a Symbian phone will dynamically increase the heap to fit with the game requirements. Of course it will still run out of memory eventually, but developers have more flexibility than most other OS.

C. Other Operating Systems (OS)

1. SavaJe

SavaJe is a relatively new OS that is mainly sponsored by network carriers, such as Vodafone and Orange. It is in essence a Java OS that runs Java applications, but in this case it is a far more comprehensive OS than the more common VM’s in J2ME handsets, in that it is based on J2SE, the platform normally found on personal computers.

SavaJe handsets should start shipping soon, a deal has been announced with the manufacturer LG, and developing for them will not differ greatly than building games for J2ME handsets.

2. Linux

Linux has found strong sponsorship in the Far East, especially in China, and is also being adopted by NTT DoCoMo in Japan. Linux’s main attraction is that it is free to adopt, although engineering it to fit handsets and maintaining it does have associated cost; however, this could prove to be more economical compared to the licensing costs of other Smartphone OS.

Linux handsets shipping today also support J2ME, so from a game development standpoint it is more efficient to build a standard Java MIDlet rather than dive into the native Linux. This may, of course, change over time but it does prevent the problems of piracy and rogue code that native Linux games would involve until satisfactory DRM solutions are adopted.
3. **Microsoft Smartphone**

Microsoft’s Smartphone OS has yet to reach any significant volumes to warrant the cost of developing games specifically for that platform. In addition, games tend to be large, so the only practical delivery mechanism is through downloading to PC and utilizing a wired connection to transfer games to the device. In short, it isn’t yet a significant market for mobile games, although this may change if handset volumes increase.

D. **New Gaming Features and Technologies**

So far, this section has spoken about general game development. The vast majorities of games to date utilize 2D graphics, have no or very little multi-user capability, and are simple parlor games. However, this is changing rapidly, and there are now a significant number of more ground breaking games that employ 3D graphics, multi-user features such as turn based play, and community features such as high score tables and opponent selection.

Some of the more important new technologies of mobile gaming are:

1. **Multiplayer**

Mobile phones are connected devices, so it seems natural that multiplayer gaming would fit into the networked world. However, there are some significant hurdles to overcome in mobile multiplayer development, with latency being the main issue facing today’s mobile game developer. Most networks, with the exception of the newer 3G networks which have launched widely only in Europe, Japan and Korea, have latency at the minimum of 1-2 seconds, with averages more like 3-5 seconds and no guaranteed delivery of data packets. This doesn’t make multiplayer impossible, but the latency has to be considered with smart game design.

Most new games now offer a centralized high score table to compare results from users in different locations, and some are introducing the concept of “ghost” opponents, whereby the player’s best lap, time or movement is recorded as a sequence, so that other players can compete against them. For example, as well as recording a lap time, games may record the user input so that the lap can be replayed on another handset while the handset owner races against the “ghost”. Not real-time multiplayer, but an interesting way to add a competitive edge to a game.

With 3G, real-time delivery of data packets becomes practical, and so we are likely to see more head-to-head multiplayer gaming. The ultimate goal is to be able to create first-person shooters and racing games, where high frames rates and close competition are the key game play elements – both require high, consistent throughput of data.

2. **Community**

Community gaming takes multiplayer a step further. Instead of one-on-one gaming, a game designer can now consider teams or other groups who commune for the purpose of a gaming or sharing some game-related experience. Some versions of this have already started to appear, with simple SMS messages being used to bring people together for events and GPS (“global positioning system”) tracking being used to re-enact games in real time in the real world, such as the recent playing of a game of Pac-Man on the grid-defined streets of Manhattan.

The community aspects of mobile gaming are a natural part of using a connected device. Link this in with other technologies such as 3D, camera technology, GPS and joystick input, and the mobile phone suddenly becomes a device with huge creative potential for new gaming experience. Treasure and scavenger hunts, which have had some success in Europe but continue to be a niche genre for now, find a natural home on mobile devices, communicating to team mates while finding opponents by physically roaming around a city becomes possible. All of these advances are great for carriers as data traffic is small, but regular and tends to generate high margin.
3.  3D Games

Many developers are pleasantly surprised by the capabilities of new handsets that are hitting the market. While there is still a huge market for 2D games, 3D technology has arrived in the U.S. and Europe after having a head start in Japan.

3D is, of course, a rendering technology and no substitute for good gameplay, but it would be hard to imagine some of today's top console titles not exploiting the advantages of 3D technologies. Equally, therefore, it is hard to visualize the mobile world not taking advantage of the same rich environment. Hardware restrictions are quickly disappearing as evident with the launch of new EVDO handsets from LG on the Verizon network. These handsets probably have more processing power than the first generation of PlayStation.

a)  Java

3D is supported on Java via M3G, the Mobile 3D Graphics API. M3G is available on new handsets from Motorola, Siemens, Nokia and Sony Ericsson, amongst others. M3G also enjoys a good conformance-testing suite, so that there is high degree of compatibility between the M3G enabled handsets.

Writing 3D games is fairly similar to writing 2D games; there are assets, game AI and a user interface. The difference is in the rendering and management of the scene. These are managed by an M3G engine and the assets for the games also have 3D models as well as 2D bitmaps. These models, as well as cameras, lights, animations and surface materials and appearance are usually authored in a dedicated 3D tool, such as 3D Studio Max, and then exported to the standard M3G file format. 3D Studio Max 7 ships with a built-in exporter for M3G and other tools are available from other technology vendors. The M3G format is also published, so developers can build exporters for existing tool chains if they so wish to port games from other platforms, such as the GameBoy or PlayStation.

Most 3D games are first appearing on 3G networks, although a well-crafted game does not necessarily need the download speeds and sizes enjoyed on these fast networks. In actuality, it is always recommended that games are made as small as possible so that users get the game on his handset as fast as possible, even if they have unlimited data packages and fast network connections.

3D porting is often eased by the fact that the scenes are rendered on the fly, so adjusting to different screen sizes is straightforward. Where the issues lay is in 2D UI assets, handset performance and heap size – no changes from 2D game development in other words.

b)  BREW

In the BREW world, things are easier for cutting edge technologies like 3D, as new engines and APIs can be downloaded and installed over the air via the extension mechanism. For this reason, each game can have its own 3D engine, rather than utilizing the generic 3D engine provided by M3G on Java handsets.

This has some advantages in that each game can have a carefully tailored engine to obtain maximum performance; however, there is also downside in that the engine takes up additional space on the handset, restricting further downloads, and development costs can be higher, as each engine tends to also have its own authoring tool costs, even if it is only a variation on an existing level editor for example.

Having said this, some of the best mobile 3D games right now are being built for EVDO BREW networks. The handsets are fast and have plenty of memory and games can rival those from early 3D-capable consoles in terms of visual quality. The ability to download/replace content over the air, the opportunity for enhanced gameplay and extended life also bring the experience closer to the level of a console title.
VIII. Developing Mobile Games

The Mobile Game Development section provides developers with high-level insights into creating a game from the planning stages through development and porting, and ultimately to complete deployment to end users. Those unfamiliar with the mobile game industry will find some unique challenges, including mobile phone technological limitations, porting, and carrier certification. Also, there are some familiar themes dealt with in some detail, including the need for proper planning/design and having an appropriately skilled development team.

All the contributors to this section come from mobile game industry and include publishers, mobile game developers, porting companies, testing companies and wireless carriers. This range of experience provides a clear idea of how development is approached for most mobile games. These contributors provide valuable tips in how to successfully navigate the complicated process to develop a successful mobile game.

A. Planning/Design

The planning and design document stage is what makes the games business a business instead of a hobby. Carriers fully expect to receive top quality games on multiple handsets covering rapidly evolving technology and multiple platforms. Without proper planning, developers will miss the window of opportunity that competitors are more than happy to fill. Without proper attention to design, developers will not push the technology nor give players a creative and fun entertainment experience. Good planning and good design minimize risks in a business where developers must predict what will interest a player 6 to 12 months down the road. Experienced, honest planning and detailed upfront design work are the best way for a team to stay on target with a common vision. When developers start development, they need to know what they are getting into: Where are the likely pitfalls? What are the trade offs? Developers need to have a series of milestone goals from start to finish, and to plan for a level of change and evolution during development.

1. Competitive Analysis

Forecasting the sales requires identifying who the competitors are, and who may likely enter the market with a similar game by the time the developer launches its finished game. If developers are entering a crowded category, it means others have identified its value. It also means the revenue will be split and must differentiate the game in order to make it in the marketplace.

Developers need to take a good look at each existing game similar to the proposed game. List the strengths of those games and what made them successful. List the weaknesses in those games and what made them fail or not reach their potential. Identify if those games include some element that has become so universal that it is expected “prior art” to a player. Even if a developer has a better idea, tradition can be hard to shake because of the comfort of familiarity.

Validate which games have done well on the target handsets. This yields a vision on how other companies have assessed the demographics and technical capability for a particular handset. For example, some handsets may be marketed by the carrier to attract sales to upper middle class business professionals. The demographic for that handset may not be compatible with the game genre, so perhaps that particular handset need not be addressed. However, the desire to cut marginal handsets must be balanced by carriers wanting games that run on 80-90% of their handsets.

2. The Team / Personnel

Each game project has unique characteristics and it is the task of the producer to determine what skills are needed to get the job done on time and with the proper level of creativity and technology. A Microsoft Project chart is a very useful tool to visualize when services need to come together.

Like any video game, the team needs a programmer, designer, artist and sound engineer. In mobile game development, with the exception of the lead programmer, all skills are not needed 100% of the
time. Artists and designers generally handle multiple projects. A designer will devote a large percent of time to creating the Game Design Document (GDD) for one game while working on tuning tasks for a second game on the late alpha or beta stage, and lend advice or tuning expertise to the porting/customization stage of a third game. The producer needs to coordinate how the resources come together on his game without affecting the progress of other games sharing the same resources.

Honesty is the very important part of determining the development team’s technical capability and competency. No one has every skill needed for every project now and in the future. The game may need special server skills or multiplayer expertise or a specific art style. Sometimes a skill can be learned in time to fit a schedule, but many times developers need the experience now. Don’t compromise on the quality of the game because a developer is lacking a skill. Hire, contract a freelancer, farm out a section of the work, or add time to the schedule for training and research.

3. Consumer Target
To kick-off the project, the team needs to know the targeted player. A marketing person gathers data and feedback from the sales team, consumers, designers, carriers and other sources. Not every game appeals to every player. Are the consumer targets casual players? Males, females or both? Players under 12? Players 18-40? Sport players? Action gamers? Arcade style players or simulation players?

Understanding how to attract and entertain the target player determines the accuracy of business projections. Developers should not expect to draw a large audience of males 18-30 with cutey cartoon art. Conversely, games with dull colors and only score-based feedback and rewards will not appeal to a female player.

Developers should keep refining the bait. Do they have a license that will attract the target? Is the name of the game simple, attractive and explanatory to the customer? Will the complexity and depth match player expectations? Developers should not over-design or under-design for the target.

4. The Concept Document
A high level concept document of about 2-5 pages should outline the basic game idea and features. The heart of this document is the “hook.” This is the feature or features that make the game stand above others in the same genre. A hook can be a license, a new technology feature, a clever twist on the theme, an aggressive pricing or bundling proposal, some hole or angle someone missed.

Developers should put themselves in the place of both customers - the carrier and the player - and be brutally honest. They may want to make a bowling game because this has been a successful niche, but the carrier already has five bowling games in its catalogue and has rejected fifteen others the public never saw. To take the game, the carrier would have to remove a proven bowling game or add the game as a sixth bowling game and water down the revenue for everyone. At this point, carriers are not seeking more games in a particular category, or more games in general for that matter, because they have a good feel for the size and balance of their catalog and wish to maintain it. Hence, every new game that goes up takes the place of one already there. A good hook makes it clear why a carrier would take this new bowling game at the expense of another new or existing game and why the player would prefer it.

Games are a business, and ideas or characters can be turned into patents and intellectual property (IP) to establish an emerging new brand; however, creating a new brand is a difficult task in mobile because carriers are looking for sales with little promotion on their part. This need for brand familiarity strongly weighs support in favor of highly recognized, proven licenses. There is no packaging for a player to examine before they buy, only a 15-25 character name of the game on the players mobile phone. It is human nature to gravitate to words and names people recognize. Balance that reality with the fact that mobile games have yet to produce a “killer app.” History shows that most killer apps on all other gaming platforms started out as a new emerging brand.
Rapidly evolving handset platforms and networks presents opportunity for technical innovation and novel approaches to player interaction. Breaking new ground can increase the value of the project beyond the surface game for the carrier and the player.

5. **Game Design Document**

The GDD is created to give the team the creative spirit and blueprint to make a game. This is the movie script and it will get into the hands of a variety of people and personality types who need to share a common vision of the game from start to finish. Without a detailed GDD, there is no common goal or direction for the team.

When writing the GDD, it’s important to get the hook and main features in the open early, complete with several screen shots that simulate how the game will look. This helps any type of reader get a good vision of the main differences of the game before entering into the detailed sections.

- **Gameplay Details**: The gameplay details take the reader through everything that happens from the launch of the game through the exit. Report the facts and do not overly embellish. At this point, developers are not trying to convince anyone of the merits of the concept. They are reporting all the details of the game flow clearly and concisely. All the menu pages and the functions of each item should be described and a flow chart should be included so the reader can see all the paths at every decision point.

- **Control Descriptions**: What does the player do? Does he control a character? If so, describe all the functions of the character and show what it looks like. How do the controls map to the handset buttons? This is critical information as there is little forgiveness for a poorly designed control scheme.

- **Playfield Description**: Describe the playfield and explain in detail how all the displays work and what they look like. List any icons in the game, how they function and how they work with the displays.

- **Rules of the Game / Level Design**: How do I play? How does the game progress? How does the game get more difficult? What are the levels? What do the levels look like and what gameplay changes occur on each? Are the levels properly balanced? Are there non-player controlled characters? If so, describe their behavior and show diagrams. List all the tuning variables for each character and each level.

Upon completion of the GDD, create a development timeline to schedule the project accurately. Everyone on the team should be able to calculate how much time they need to complete their section of the blueprint. The producer brings this all together in one timeline so each part of the game is ready when the other team members need them.

Estimating the budget is a factor of correctly assessing human resource needs based on the development timeline. Each company adds their own multipliers for overhead and profit. Why go through all of this? Because an incomplete GDD will yield an inaccurate timeline, which will lead to cost overruns and late delivery.

6. **Technical Design Document**

The Technical Design Document (TDD) is part of the overall GDD. A designer should work in conjunction with a programmer, artist and other technical resources whose expertise is needed to fully understand how the game will be created. It is important to balance creative entertainment value with technical feasibility.

Estimate all memory use in partitions for code, sounds, graphics, specialty API and other pertinent categories so the team is grounded in the reality of handset technical limitations. List the number of animations, frames of art and the average size per frame used for each character and object. This holds the graphic memory in check so the team isn’t focused on characters and animation there is no possible
room to include. Segregate art use that will be expanded for higher end handsets or special download additions.

If there are server functions, describe them and how they work. Describe how any special technical needs like multiplayer interactions will work. How will you deal with latency issues? How will you handle broken connections? Make sure you have allocated enough memory for the specific API’s that will be used.

A marketing person or project manager identifies which handsets and platforms are required when the project is launched. Developers have to make a business decision about the game design. The more handsets that can be covered the better. Carriers are demanding more and more ported handsets as a minimum for launching a title, and developers make more money. If a developer wants to push the technology and memory to make a more in-depth play experience, or maintain a higher quality level for a licensing relationship, some handsets will not be capable. They need to know early on what handsets will have to be cut.

Selection of the master handset to develop the first version of the game is becoming a tougher choice in the mobile industry. Unlike the console business with fixed platforms, mobile is a fluid environment with rapidly changing handsets being introduced constantly. Memory restrictions, processor speeds, features, button placement are all moving targets and a developer needs to re-assess every six months which handset will be the master development platform. A good plan is to use a master handset with capabilities a bit higher than the middle range. Starting with a low end handset and scaling up the art for high-end handsets is a nightmare. It has to be completely redone and resized. Starting with a high end handset and scaling the art down for the low end is easier, but there will be many, many problems having to re-tweak a game that was initially created with a very responsive frame rate and a larger than average screen size. With increasingly more handsets to be supported, developers may need to have more than one master handset to support groups of common handsets. Publishers today report that three or four master builds are common.

Depending on how much of the market is to be covered, approximately two-thirds of the development time and money will go to porting and customizing the game to diverse handsets. It is important to group the handsets by their capabilities: memory restrictions, color depth, screen size, speed. This creates a data base that can be used from game to game. Update the list as new handsets are introduced by the carriers and put them into the appropriate category.

For each category, identify how the game will be adjusted or customized to fit the constraint. Design the game so playfield elements can slide horizontally or vertically to fit different screen sizes. Determine if art needs to be reduced in color or size to fit a handset category and allow background art to expand or contract to fit different screen sizes when possible. Don’t bother with black and white handsets with small screens, as most all carriers are dropping these from their line. Determine if speed is sufficient to make a playable game for the selected genre, and set a minimum frame rate limit the game must reach in order to be acceptable. Keep trade-offs in mind. What can be given up to increase the number of frames per second? Create a technology demo to prove out the concept early on to save months of frustration and wasted resources.

B. Publisher Relations

The most important consideration when working with a publisher is: “If you can’t deliver, don’t take the job.” Smart developers honestly assess their capabilities and then following through. Failures can have catastrophic results for developer and publisher alike. Carriers control the market, and each has only a fixed number of slots in each genre. They want high quality games, games that sell their services and retain their customers. When deliveries are not met and quality falls below expectations, the publisher does not just lose development dollars, they lose a valuable carrier slot to a competitor, market share, credibility with the carrier, previously-projected revenue, and maybe a relationship with a top licensor.

Publishers need to recognize that developers are on a fixed budget with low profit margins. Changes that occur from the initial plans and schedule need to be minimized. If not properly managed, changes can
become significant to the point of putting a small development company out of business. The best publisher/developer relationships are formed when each has a clear understanding of the needs, risks and problems on both sides.

In a healthy open relationship, the publisher and developer both have complete understanding that there are conditions out of their control and that a carrier can change the market conditions with little or no warning: new features will be promoted, new handsets will be launched, new policies introduced, new games by competitors will be launched. This is now a standard part of the mobile game business and six months is a lifetime in this industry. The best defense is to get as much information as possible to forecast these market conditions and work as quickly and efficiently as possible. Since the market conditions are so fluid and contain the carrier wild card, adaptability is key to survival. Fast, efficient work habits coupled with adaptability are what make planning, schedules and information flow between publisher and developer so important essential. Publishers are the risk-takers in the relationship, which is why they have the bigger upside. They should absorb the consequences of changing market conditions and be prepared to offer increased funding to developers when it is warranted by factors beyond the developers’ control.

When working with publishers the first step is to create and gain acceptance for:

- A description of the game and features.
- Screen shots and sketches so everyone can visualize what they are agreeing to make.
- A technical assessment of the game project and a list of potential risks and experimental steps.
- A detailed development schedule.
- Major milestone payment schedule.
- Change in scope cost plan.

The publisher and developer will go over this material together and adjust so everyone is on the same page. There should be no major surprises after this stage. Both parties know what the game entails, where potential risks may occur in the schedule, when various skills are brought in like sound engineers or artists, when licensing materials are needed, when is play testing needed, when are the approval milestones and key delivery dates. It is important to establish some type of day rate or some other type of compensation up front if there is an unavoidable change in the projects scope.

Once initial agreement is made, focus should move to follow through and information flow. The detailed GDD and a Technical Design Document (TDD) are needed first as a blueprint and can be a requirement before a final agreement between companies can be reached, or part of the first milestone of the agreement. When this blueprint is signed off, everyone is good to go.

Information is a two-way street. The publisher needs to keep the developer informed of any changes in market conditions, especially if there is a warning that any part of the game project that may need to be altered. The developer may only be contracted to show builds at key milestones like alpha and beta, but it is best to show interim builds to show steady progress and keep in communication with the publisher to insure expectations are being met. During development there are surprises. Unplanned good ideas come to people, and sometimes original planned ideas don’t work as envisioned. No one wants to lose opportunities if good ideas come during development, and no one wants to be surprised at seeing the game at alpha if it has taken a different direction. If both parties want a good relationship, they need to keep their partners in the loop on these turning points in development and make informed trade-offs and combined decision. Being proactive in this regard is about respect. When developers and publishers work respectfully with each other as partners to make a great game that will be successful in the market, this is the highest probability of success.

C. First Playable/Prototype/Demo

No development plan or schedule can be validated until the concept itself has been validated. An idea may seem good on paper, but if it is not proved to be fun or technically feasible, developers can’t take it to
the bank. A “Proof of Concept” or “Pre-Alpha” milestone is an insurance policy to protect the development investment.

The goal is to extract the key element(s) of the game and create a prototype that the team and publisher can use to visualize the core ideas, verify that these elements will work and be fun. Every game has its own inherent question marks.

For a licensed character game it is important to take at least a couple frames of art and reduce to the proposed size and POV (“point of view”). There will be a threshold between too small to be recognized and too large to be memory efficient. For the game Terminator, I’m Back this was the main action that had to be done before the rest of the game memory calculations could be completed. The game is isometric and the characters need to move in a minimum of four directions. Characters had to be large enough to be easily recognized and the Terminators needed to look physically dominant against Kyle Reese. Without taking this step, there could have been a number of problems meeting memory constraints and publisher expectations.

For a multiplayer card game, developers would check the visual spacing of cards on the screen, verify data transfer and check any server issues. Get two players “talking” to each other so they can experience what will be easy or hard to develop. Try to remove assumptions and guesswork.

For a driving game, developers don’t need finished art for the whole game to prove the concept. It is much more important to test the controls. Make a placeholder car in the size and point of view proposed and then apply steering and acceleration controls. If the road is an important part of the game, get that part of the engine going even as a wire frame.

The bottom line is that every game development project is a gamble. The proof of concept step is the developers’ chance to minimize the risk and recalibrate the investment and schedule required. Sometimes things are harder than originally thought and sometimes shortcuts are discovered. Publishers are universally more open to adjusting schedules and budgets EARLY in the project timeline -- they may do so late in the schedule out of necessity, but they will not soon run back to developers that fail to deliver according to plan close to the end game.

1. Implement Revisions to the Development Plan

In the first prototype of Midtown Madness 3 for Mobile, the plan was to use smaller vehicles in a top down view and try a less complicated control system that reduced math calculations. The combination would make porting easier, increase frames per second (FPS) on lower handsets, and allow more graphic memory for frills and effects. What was learned during the proof of concept was the control system did not have a good enough driving feel for a contact action driving game, and the cars were too small to distinguish direction on a small handset screen or provide a strong emotional connection for the player. The test required the team to increase the vehicle size, which forced cuts and modifications in how the graphic memory was to be used in the rest of the game. The test demonstrated exactly how much driving control was really needed to make the game fun.

Once developers have some facts from the proof of concept test, they can assess if the schedule and development plan needs to be adjusted. Don’t ignore this reality check! Listen to it because it is early enough in the process to establish realistic expectations and negotiate alternatives. It is much better to adjust now than to get surprises and adjust late in the project when there are no alternatives and you are pushed to a deadline.

D. Game Balancing

Good game balancing takes a high level of professional skill. Ultimately, balancing determines the longevity of the product, and what is done on the last 5% of development can determine 50% of total sales. If the elements are not set up to balance the game from the first version of the GDD, don’t expect
the tuning and balancing variables to suddenly appear. Principles are the same as with any form of game design. Keep the following list in mind when developing for mobile.

1. **Know your player**
   The most difficult task in balancing a game is to accommodate varied customers. Mobile has a high percentage of casual players, yet licensing still leans to attracting hardcore gamers. The best way to satisfy the needs of both is to focus on “easy to learn, difficult to master.” It is the “easy to learn” part that many developers miss. A casual player won’t have the patience to read instructions and figure out what buttons to press. Things that are well-known to gamers are not necessarily known to people who don’t frequently play console games or are perhaps only attracted to play a card game or internet style puzzle game. For them, just getting past learning what button to press on the handset to get started is a difficulty level. Balancing a game can’t be done if the target customer is confused as soon as they see the game field.

2. **Protect your depth**
   Mobile games do not have the graphic and code memory that rival the number of characters and levels found in console platforms. The key is to give the player an entertaining experience that is commensurate with the cost of the product. If a player can get through all the levels, play modes and unlock everything in an hour, there won’t be much satisfaction. Pacing is everything. To balance the game properly developers need to estimate how long someone will play the game in a session; how long would one game take to play; and how many days will a player maintain interest in the game. This gives an indication of when one can expect that a player will reach milestones and unlock levels, tracks and goodies. Memory restrictions, development time and cleverness determine how much depth can be put into the game. Don’t water down the game just to say the game has 50 levels. It is better to have quality than quantity in today’s market and even the most casual player can see past false promises and empty depth created by repetition.

   The first level, track or environment is the most important to balance perfectly. This is what everyone sees including carriers and newbies. After one or two plays, the person will decide if they like the game or not… forever. Let the player enter the world, get far enough to learn the rules intuitively, get a taste of some success, then challenge them so they want to try again with the feeling that they have a chance to get farther if they try again and practice. Anticipate the learning and skill curve of the player and verify it through play testing. Hold a carrot out there just far enough away that a player is challenged to improve his skills but not so far that it feels like hitting a brick wall.

   Getting that balance of challenge verses brick wall is tougher when there are limitations. If there are only four quality levels in the mobile game, developers have to take a risk. Make a steeper difficulty ramp than normal to slow players from getting through the whole game, or use a smoother difficulty ramp and risk players mastering the game too quickly. To avoid the frustration of a difficulty ramp that is too steep, developers can sometimes keep the difficulty smoother but throw surprises at the player to get him off his normal play pattern. For example, make the difficulty ramp varied in spots so there is some intense action, followed by a slower point, then more intense action. This can have the effect of throwing a change-up pitch at a batter. When players get a bunch of fastballs, sometimes a slow, off-speed pitch is more effective.

   Remember that balance applies to card games, board games, anything with AI. Instead of levels, the game has a computer opponent and the same principles apply. Skill levels and varied behavior need to ramp up and developers need to control how quickly they want to do it.

3. **Planned rest points**
   Mobile phones are small screens with tiny buttons spaced close together. Plan around these facts when balancing the game. Losing a game because of a cramp from pressing too much or eyestrain from squinting at too much text and tiny icons on a small screen is not fun. Design rest points when balancing
the game. If this is a level game, put in recaps, mission announcements, anything to give a player a chance to rest up the fingers and concentration.

Balancing the game is not just about making a game more difficult; it is about making the game fair. On a small screen it is more important to give a rest point and window of invulnerability when re-entering after a death or crash. This gives the player a fair chance to focus the eyes and get back into the game mentally. Ramp up the action and don’t just hurl everything from the exact point of speed when the death occurred.

In mobile, developers must be clever when balancing the upper ends of the difficulty spectrum in the game. They can’t rely on unplayable speed-ups and overwhelming characters on the screen because many handsets do not have the hardware capability or screen size to reach a threshold that a skillful gamer can’t overcome. In some games, the upper ends are unbalanced because the object limit is overflowed. There are more objects but the game is slower and can be played perpetually. Not a fun thing to discover when porting to slower handsets. Use increased AI variables for enemies; make the environment or road track a challenging layout; limit goodies and extra health opportunities; use greed to tempt players to take risks that pull the player away from their successful game plan.

E. Development

There are three technological types of game projects which require different development methods and tools. They are defined by three phases of mobile phone and network technology:

- 2D games: mainstream, commercial breakthrough in ~2000, now with over one hundred million phones deployed
- Multiplayer games: innovation, commercial breakthrough in ~2003, now over ten million deployed
- 3D games: technological edge, commercial breakthrough in ~2005, now over one million deployed

Today, most professional developers make their living from 2D single-player games. Some of them have already done their first 2D multiplayer game where networks and devices have the technology to support it, and many industry pundits argue that this type of “social gaming” will become a very large segment of the business due to the simple fact that your gaming platform is at its roots a communications device (which some people argue may or may not be relevant to using it as a game platform). Last, but not least, the first 3D phones show what a compelling graphical experience of next generation games can be - but even the large developers just started with 3D projects for 2004/2005.

So, here are the main differences in developing 2D, multiplayer and 3D titles.

1. 2D Games

In general, 2D mobile development can be compared to 1980’s type projects. Quite small teams of 3-5 (concept, design, and development) develop games in 3-6 months. The key challenges of 2D mobile development are:

- Develop with phone constraints in mind (processor, screen, controls, and memory)
- Squeezing full games into 50-150 kilobytes
- Providing several languages for worldwide distribution
- Porting from one or more base builds for mass market rollout, e.g., 80-100 handsets x 5 Languages = 400-500 SKUs

Leading game developers and early tool vendors have developed frameworks solving some of the porting pain and are continually striving to come close to the promise of “write once run everywhere”. The tools framework is one of the critical success factors for mobile game developers (see Tools – Framework later in this section).
Example: “Seabattle” – A 2D title ported to 75 handsets with various screen sizes was possible because of its early design considerations and tools framework used.

2. Multiplayer Games

The next evolutionary step from single-player 2D is multiplayer gaming, where the client / server requirements should be an add-on to an already solid 2D game concept. Due to the unique social and ubiquitous nature of the mobile medium, connected games enable a lot of space for game play innovation.

Commercial mobile multiplayer games are complex to develop and to rollout, as they require server-to-server connections to various mobile carrier systems or gateways like game lobby or presence servers. There are already a few multiplayer middleware platforms and services available from independent tool vendors and mobile publishers and carriers are now beginning to investigate standardization for their game offerings (e.g., Sprint requires the use of Game Lobby, a www.M7networks.com product).

The major challenges of multiplayer development are:

- Create concepts and software that handle mobile network latency of 2-4 seconds
- In-game chat and communication
- Balanced game play (turn-based, simultaneous movement, slow update, asynchronous MMORG)
- Make or buy: Choose the right multiplayer platform solution and investigate carrier restrictions
- Even more complex build management for mass production, e.g.,
  50 handsets x 5 languages x 5 systems = 1,250 versions
- Effective QA of distributed applications on various networks
- Cross-carrier access for mass market projects requires extra QA

Example: “Catapults” - A turn-based multiplayer game with integrated chat and subscription billing. It is a J2ME application based on an OMA Games Services multiplayer platform connected to various carriers, e.g., Vodafone or T-Mobile (Developer: Bytedefenders, www.bytedefenders.com. Middleware: Exit Games Neutron).
3. **3D Games**

3D is a major change for the industry, as the end user experience reaches nearly console quality – with a commensurate increase in costs. It is also a dramatic change for mobile game developers, as the core development methods are now very similar to current 3D PC and console productions. With upcoming 3D API standards like JSR 184, porting to different handsets changes in complexity as the graphics scaling is more or less done by the 3D engine which theoretically makes things easier. Unfortunately processing speeds and memory management continue to be a major barrier for development. Be cautious of incompatibilities between 3D middleware tools and mobile phones (e.g., not all polygon attributes work on all devices).

The challenges of 3D mobile development are:

- Manage early adopter technology glitches, especially first generation 3D middleware / API bugs
- Handle huge performance difference between 3D handset vendors
- Manage good 3D usability with limited device controls
- Close the current tool-gap and find file converter, emulator and exporting tools for cross-handset development


Example: “Robot Alliance” – One of the first 3D multiplayer J2ME games, the asynchronous MMOG comes with revolutionary 3D graphics. It offers server based gameplay and in-game messaging system (Developer: Fishlabs, www.fishlabs.net, Middleware: Exit Games Neutron).

**Development Skills**

The development of a mobile game inherits many core methods from classical game development; however, there are some major points of difference. The following points outline the phone-specific differences of game development – porting, usability, performance.
First, for **Software Development** from architecture to coding and testing, here are some mobile must-knows:

- Develop with phone constraints in mind
- While the small screen size is obvious, the limited game controls are often underestimated (e.g., you can do only one key press at a time on most phones today)
- Plan short game sessions due to mobile lifestyle situations and battery life considerations
- Plan phone specific usability tests (e.g., interruptions, pause on call)
- Handle small amount of game assets in a 50-150 KB executable
- Manage the porting challenge (often to over hundred phones worldwide)
- Know the various screen sizes (handset manufacturer developer sites)
- Learn the different programming APIs and avoid API lock-in where possible through a hardware abstraction layer / framework (Java J2ME MIDP 1.0, MIDP 2.0, J2ME JSR or phone vendor API extensions, BREW, etc.)
- Cover different key layouts and UI guidelines
- Plan and test different processor performance on reference phones before you design
- Discover and structure individual (often undocumented) errors per handset type / firmware
- Plan and test different heap and jar sizes per handset type
- Balance size, security, stability and performance
- Avoid object oriented code overhead (5 classes are great, 4 better)
- Shrink code size with obfuscation

For **Artwork** development, all display constraints (size, color depth) are important basics. Moreover, minimizing file size is essential as every byte counts when shipping a 64 KB game (the standard series 40 Nokia build size that is 70% of the gaming market in Europe). It is also a good idea to check on real phones and check in daylight. Many developers build a game, literally in the dark.

The **Level Designer** should finish all levels before porting in 2D projects, as the expensive 2D graphic porting should be done just once – 3D projects are more open to last minute level design changes.

The **Producer** should understand that organizing handsets as early as possible is essential to optimize the game for the relevant target device. Many project problems develop through late arrival of the “real” handset. For 3D games, access even to prototype handsets, test boards and rare or expensive tools are important.

### 4. Resources

Excellent, free-of-cost resources are available online. Check sites for device and OS vendor and targeted carriers, including the following:

- http://www.forum.nokia.com
- http://developer.sonyericsson.com
- http://www.motocoder.com
- http://java.sun.com
- http://brew.qualcomm.com

For 2D Java development, Jason Lam’s MIDP 2.0 book:

- http://www.sourceforge.net/projects/j2megamingbook/

For multiplayer development, platform vendors and publishers with multiplayer technology:

- http://www.terraplay.com
- http://www.gamefederation.com
- http://www.exitgames.net
- http://www.mforma.com
http://www.kayakinteractive.com

For 3D development:

- Motorola and Sony Ericsson currently have some of the best phones and developer tools
- Forum Nokia offers a good JSR 184 white paper

F. Mobile Development Tools

Commonly used development tools are listed in the following table:

<table>
<thead>
<tr>
<th>Type</th>
<th>Solutions</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>IDEs</td>
<td>Eclipse, IntelliJ Idea, JBuilder, …</td>
<td>Many phone emulators integrate into these IDEs</td>
</tr>
<tr>
<td>Version Management</td>
<td>CVS, Perforce, Alienbrain</td>
<td>Essential for mobile development as well.</td>
</tr>
<tr>
<td>Build Management</td>
<td>Ant, Antenna, Custom Tools</td>
<td>Essential to master the porting challenge</td>
</tr>
<tr>
<td>Precompilation</td>
<td>Antenna (antenna.sourceforge.net), VPP (vpp.sourceforge.net)</td>
<td>Best to differentiate handsets.</td>
</tr>
<tr>
<td>Obfuscation</td>
<td>Proguard</td>
<td>Squeeze it!</td>
</tr>
<tr>
<td>Emulators</td>
<td>Nokia, Sony Ericsson, Sharp, LG, Samsung, Siemens, Motorola, …</td>
<td>Better have them all.</td>
</tr>
<tr>
<td>Real Phones</td>
<td>Nokia, Sony Ericsson, Sharp, LG, Samsung, Siemens, Motorola, …</td>
<td>Crucial for testing</td>
</tr>
<tr>
<td>Multiplayer Middleware</td>
<td>Terraplay Move, Exit Games Neutron, Gamefederation GEX, Mforma Connect X</td>
<td>Special middleware services</td>
</tr>
<tr>
<td>3D Tools</td>
<td>3DSMax 7, Swerve Tools, HI Tools, Superscape</td>
<td>3D middleware</td>
</tr>
<tr>
<td>Porting &amp; Deployment</td>
<td>Tira Jump (<a href="http://www.tirawireless.com">www.tirawireless.com</a>)</td>
<td>Porting service</td>
</tr>
</tbody>
</table>

1. Device Porting Framework

As seen from the various porting problems, a device framework is a key tool for mobile developers:

- Ensures high quality (tested and stable code base), mass production (build management), high code reuse, quicker time to market
- Language handling for multiple languages in one file verses separate language files for each of EFIGS (English, French, Italian, German, Spanish)
- Image and menu handling
- Individual code handling (e.g., workarounds for handsets)
- Code obfuscation (saves up to 40% of code size)

There are three options to get access to a porting framework:

- Build your own
- Buy 3rd party (e.g., Tira Jump, others)
- Use publisher in-house framework (Mforma, Jamdat, Superscape, iFone, etc.)

Option one will take the longest but ultimately increases your value to other professional mobile game developers and publishers. If you optimize on working for hire for large publishers, it might be an option just to use their systems.
2. Quality Assurance

Emulators CAN NOT replace the real phones! You must get onto the real handset to experience the FPS, screen and memory issues first hand. Test and re-test the final product on real phones. Due to the nature of this fragmented medium, developers will need access to at least 10 to 20 mobile phones. Mobile carriers are demanding a large number of handsets in order to reach most all of their subscriber base. PC phone emulators are often more buggy than the actual phone and nearly always different from hardware implementation. It’s much more sensible to get the game onto the handset as soon as possible.

For tracking errors managing workflows and responsibilities you can use issue management systems like Bugzilla (bugzilla.mozilla.org) or JIRA (www.atlassian.com/software/jira/).

3. Development Process

Back in 2003, a typical mobile game project took about 3 months with a team of 2-5 people. With big franchises, higher budgets, multiplayer and 3D this is changing to 6-12 months. In each case, planning, development and porting/QA should take 1/3 of time. The game product life cycle is still just about 1-2 years due to the average device exchange cycle of 15 month and changing focus of carriers.

So, the mobile development process is characterized by the following six points:

- relatively short project times (< 6-12 month)
- relatively small teams (< 5-10)
- relatively small budgets (30-150.000 $)
- unstable platform (bugs, new devices during development cycle)
- extremely broad handset range with many small device differences (up to 100 handsets)
- complex rollout and distributed operations and multiple languages (over 100 mobile carriers)

The actual development process has two major phases: creation of the Master Version and Master Handset Versions and porting and deployment to multiple SKUs.

a) Master Version and Master Handsets

With any game project, selling the game play idea should be the focus when code and artwork are integrated for the first time. Something special for mobile is, it’s a good idea to have the master version running on the most limited and extreme handsets (fastest vs. slowest, largest vs. smallest). This provides a good insight for all stakeholders what the final user experience will be like in best and worst case.

Experienced teams develop the master for just one master device class. At the moment Nokia Series 40 devices are often named by mobile game developers as their reference master device class. S40 devices are the largest distributed handset class and make up a significant amount of sales in Europe and in the U.S.

The master version runs through the established stages in software development: alpha, beta and release. The alpha version represents the first version of the game; it is likely to be unstable but useful for demonstrating internally and to select customers. The beta version of the game still needs full debugging, but it is feature complete and satisfies all major requirements of the game. This version is released to a group of beta testers (or, sometimes, to the general public or beta community) for a user test. The gold or release version of a product is the final version of the game, fully tested and de-bugged with all the master release handsets completed. This version is the basis for the porting.

b) Porting and SKUs

The gold master of the game is basis for this phase. Any bug fixes or changes necessary during the porting phase will lead to significant higher cost and delays.

This porting phase includes:
Handset ports
Language ports
Carrier (File format, certificates)
Carrier specific billing systems API

Each port must be fully tested and run through beta and release stages as well. The device porting framework is a key success factor. Results are hundreds of SKUs ready for deployment to the carrier to be passed on to the mobile gamer.

4. Summary

Mobile game development is rapidly changing due to market dynamics and a significant number (over 400 million) of new handsets released yearly. With increasing market size and consumer demand, development time and budgets have nearly double over the last year. Now with the entrance of new technologies like 3D and multiplayer, it is sure to evolve again and continually present opportunities for innovation from small mobile developers. Developing great games for over one billion mobile phones will certainly demand a special perspective. To succeed, think mobile, understand the medium, learn new device platforms as they are launched, and leverage ideas surrounding a ubiquitous network for real game innovation.

G. Testing and Publisher/IP Approval

a) Designing Your Test Plan

There are many books written on the subject of testing. Generally, the process of product testing should begin in the design phase of the development effort. For every feature or function of the game, there should be a test. In the case of mobile development, this is also required. With a decent test plan formulated throughout development, the game stands a much better chance of moving quickly to market and delivering an entertaining product.

The plan is often little more than an organized list of test cases, most being simple PASS/FAIL tests. Gathering the requirements (test cases) for the test plan is where the work and commitment is. Your testing requirements will have several different variable:

- **Internal Test Requirements**: What are we building and how can we be sure it does what it’s supposed to do?
- **Publisher Test Requirements**: Does the game function properly, play well and is the IP well represented? Is it fun? Does it deliver a high enough entertainment level?
- **Customer Test Requirements**: Does the game behave in a manner similar to the standards for all games? Does it meet the carrier requirements? Does it meet all Marketing and Branding requirements?

The **Internal Requirements** for the test plan are the ones developers derive. These can and should include those gathered from publishers and distributors in order to deal with these issues as early as possible. The goal with the internal test plan is to discover defects and design flaws as early in the process as possible where they can be more easily addressed.

The **Publisher Requirements** will cover most if not all of the Internal Requirements. Why have Internal Requirements then? Because there won’t always be a publisher to provide you with a thorough list of tests and you will be changing your procedures for each publisher. The more self reliant you are, the more professional you will be and the more effective and efficient. The developer’s reputation will depend on this.

The **Customer Test Requirements** are those developed and provided by the end customer in this case the carrier. This would be Vodafone, Verizon, Cingular and the Sprints of the world. There are, of course, numerous smaller carriers many without published test requirements.
2. Other Testing Issues
For defect tracking, Bugzilla seems to work well but requires a fair bit configuration and administration. FogBugZ is also a good low-cost commercial product. No matter which defect-tracking tool is used it will become one of the cornerstones of the development environment.

Finally, there is no better test tool than the phone itself. Phones can be purchased from service providers but carry with them a lot of strings in the form of contracts and conditions. The place to buy your mobile phones is Ebay or other online auction/discount websites. Not only can the latest phones be acquired but older phones can be purchased very cheaply. Be sure to buy “unlocked” devices that are not locked to a particular carrier’s network and can be used anywhere.

3. Play Testing
Play testing is similar to standard game development. Is it fun? You should outline a simple test procedure that will give you information from your target user. Be sure to test a large sample to take in account for variations. There are however, a couple of points worth noting.

First, when games are ported to numerous different platforms the CPU speed of each phone can make for a vastly different gaming experience across phones. So be sure to use the extreme slowest and fastest phones for your testing. Also, when testing networked games the developer must be aware that the current state of mobile networks is mediocre at best. It is difficult to get sustained reliability and bandwidth. Carriers have different rules when moving through their WAP gateways that don’t always mean you will have a good user experience. Be sure to factor in a thorough network testing process in order to understand the nature of the target network prior and a full testing procedure to flush out any problems.

4. Conclusion
While it’s difficult to convey the nuances of mobile game testing in a couple pages this should give you an overview of what is needed. The mobile market is growing quickly and there is money to be made. However it is worthwhile to note that this still is a very young market and due to the wide variance of devices, constantly changing network configurations and new technologies being introduced there will be many challenges as it grows. It is critical that developers consider a thorough testing process to ensure that, no matter who their customer is, they can provide a high-quality product.

H. Porting/Localization

1. Introduction: Why Porting?
Porting is the final step in the path to profitability. 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.
2. 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.

3. 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.

Most importantly, these and all tests must be performed on the actual devices. Emulators can help considerably with the initial development, but final development and testing cannot be completed with emulators as emulators do not represent all of the handset nuances.

4. Conclusion
To succeed in the mobile game market, it is necessary to have a great product. But it is equally important to make that product available on many handsets and many carriers in order to establish a market presence. Developing a porting strategy from day one will help to ensure that the applications generate the profits that they deserve.

I. Certification and Carrier Submission

1. Certification Requirements
A game can be considered “defect free” on any platform and on any environment once it passes functional testing. However, due to today’s ever increasing technological and financial challenges of designing and building a game that will capture a user and bring them back for the repeat purchase, usability as well as functional testing becomes critical.

Usability testing starts with consistency. “Action” keys perform the same task through each advancing level, “Soft-keys” display the correct wording relevant to the key’s action, etc... etc.... The same principal applies to many other forms of software on many other hardware platforms. This ensures ease of use for the user during the play of a new game as well as a low learning curve for subsequent games that user might purchase.

Some usability requirements can be debated as well as bypassed completely but the majority should be taken into consideration when the game is in development. Below are the most relevant requirements that every development studio and QA department should consider during the game’s production life-cycle.

2. Menu Options
Menu items should be clear and concise and relate to the game’s overall purpose. With that said every game should have a variation of the following.
- **New Game/Start** allows the user to immediately start the game play with a minimum number of clicks to begin.
- **Save/Load Game** allows the user to save their current progress and begin again from the point of exit at a later time. Of course, this type of functionality would depend on the depth of the game such as an RPG or Action Adventure where each would have multiple levels.
- **Options/Settings** allows users to change the game-play variables such as difficulty and sound/vibrate on or off.
- **Help/Instructions** tells the user how to play the game and what keys perform the necessary actions of the game, and gives any background information about the game that isn’t apparent to the user.
- **Quit/Exit** ends the game.

Other menu options are encouraged when needed such as High Score, Community Login, etc., but the core options should always be available.

3. **Sound & Vibrate**

Sound and vibration within games add to the experience and at times can enhance the overall game-play. However, due to the fact that the device is first and foremost a phone neither sound nor vibrate may be appropriate when in a public setting. Therefore, users should be able to toggle the vibration and sound of any game from within the game’s main menu structure (Options/Settings) as well as during game-play. This can be accomplished through the use of a pause menu or quick keys such as the pound (#) or asterisk (*) key.

4. **Soft-keys**

When soft-keys aren’t necessary they shouldn’t be made visible to the user. This is true during loading screens, splash screens, and so forth. When soft-keys are available their function should be clearly labeled and intuitive. A good example would be “Pause”.

5. **Common Issues**

The three most common issues that arise during final QA include memory allocation problems, canvas sizes, and grammar/punctuation/spelling issues:

- **Canvas size**: The game should fill every possible pixel the phone is capable of displaying. If there’s real estate, make use of it.
- **Grammar/Punctuation/Spelling**: There’s no excuse for poor grammar or typos. This is one of the most common problems that most carriers see that are also the easiest to resolve.
- **Invalid JAD files (Java only)**: Attribute mismatches are the most frustrating of problems because they should have been caught during the publisher’s QA. Again they are also one of the easiest problems to resolve.

6. **Carrier Submission Process**

Once the game has been selected by the carrier, the submission checklist begins. Below are a set of guidelines that should be considered when submitting the physical files to a carrier for testing and certification. Bear in mind that these are general guidelines and that each carrier has their own submission requirements; check with carriers and/or publishers to obtain specific requirements, as they will impact the development process and the devices supported.

- **Naming Conventions**: Make sure that when submitting files to a carrier that any and all required file naming conventions are followed. If the person who receives the submission has to manually rename files, the process will be slowed down and cause unnecessary time delays for the game being published.
- **Device Support**: Ensure that the devices being submitted for are still supported by the carrier. Lower end models or phones that are no longer being used by the customer should not be included. This makes better use of the carrier’s time, the developer’s time, and the development budget.

- **Consistency**: Make every effort possible to include the same feature set of a game for a specific device the same for every other device supported. If a game allows in-game chatting then the game should support it on every other device.

- **File-Size Restrictions**: Make sure that the carrier’s guidelines for the game’s file-size are followed. Just because the device can support a 2 MB download doesn’t necessarily mean the carrier will allow it. Check with the carrier for specifics.

- **Retail Information**: Sell the product. Make sure that the price, description, catalog title, and any other sales information required by the carrier are accurate. Mistakes in this information can result in lost revenue.

After checking and double-checking these guidelines, submit the files to the carrier by e-mail, ftp, or some form of web-based solution. Most problems are simple mistakes that cause unneeded delays and frustration. Pay attention to the guidelines provided by the carrier and follow them.
IX. From the Field: Starting a Mobile Development Company

by Oliver Miao – Co-Founder of CenterScore

Starting your own company is something that is often glamorized. If you have been in the video game industry for any length of time you understand the hard work that goes into starting up a development company. You might remember the days when friends found out you started your own gaming company and they immediately assume that you are rich, you set your own hours, and you are having a lot of fun. Only one out of the three is true.

I was often told that people should only start their own endeavor if they are passionate about it. It’s definitely true. Although my friends and I still have hopes of becoming financially wealthy from our company, the real truth is that until recently, we were each living hand to month for well over a year. Understand this as “let’s play musical chairs with our credit cards so that we can avoid interest on our personal debt.” Likewise, from the beginning, our hours were never flexible – in any job where you have a clients and a team, you need to have consistent hours to get work done and be available whenever the customer wants you to do something.

Yet through the hard work have also come rewards. You have a real sense of accomplishment for building a company together with your team, sometimes your most unique, creative games actually do come to market, and you learn more about accounting and general business practices then ever. Starting a game development company can be thrilling and character building event. You need to remember to ground your dreams in the reality of business, use prudent and diligent business practices, and understand that your work will become an all consuming part of your life for first several years of your startup.

A. Motivation to get started

So why are you going to do this? Why would you put your personal life on hold and start your own company? Let’s look at some standard reasons people have for starting their own game development company.

a) The Buy Out
There has been a lot of activity in the industry over the past 18 months. The way the industry has been heading with publishers becoming more powerful, the avenue to direct publish is becoming very limited. Most of the acquisitions that have occurred in our space are of those companies that began self-publishing their own titles over a year ago and had direct distribution contracts with the carriers - something that is much harder to do now. Consolidation for distribution gains is just about over. So, you need to look to the next logical step in the growth of our business. The next wave of consolidation will come from wireless publishers acquiring high quality development studios with unique engines and porting capabilities. There’s still time for you if you are interested in this.

b) High Royalties and the Killer App
The percentage that developers receive from royalties has seen a steady drop and, although the number of handsets continues to increase, the overall mobile gaming population is still relatively small. So even a huge hit title will not realistically translate into instant retirement for you.

c) Making a Living
So you may not become an instant millionaire, but you should be able to make a steady living on development projects you get from publishers. But don’t expect rates per work month to go dramatically up – the steady influx of new developers will keep a downward pressure on development fees.

d) Creative Freedom
You’ll have much greater immediate financial success developing known projects for a publisher, rather than trying to develop your own franchise or innovative ideas. You may have creative freedom in helping to shape the design of a game, but the basic ideas of the game will more likely than not be dictated by the
publisher. Unlike web games where self-distribution is much more realistic, mobile games are clearly controlled by the publishers and carriers.

e) Having a Major Impact/Involvement on a Project
Given the scale and costs of cell phone games compared to console games, you definitely have the opportunity to have a major impact on a game. In fact often times, games continue to have one lead developer and one artist. Also, even though production cycles are becoming longer, they still average less than 6 months for a game. So you still will be able to move from project to project before it become tedious.

f) Better, More Flexible Hours
Probably not. Nearly every founder I’ve met continues to mention working weekends and long hours. When you’re starting a new company, there’s virtually no way you can get away with working just 40-60 hours a week.

g) Easier then the Console Business
Probably not. But it’s definitely less crowded and with a discipline team, professional business practices, and a knack for sales you’ll have a good shot at landing some contracts.

So if you want to start your own mobile phone game development company, it’s going to be so that you can control your own destiny. And if you get lucky, you may have some creative freedom, pad your pockets in the process, and build an entity that will become part of a larger publishing group at a later time. It’s best not to plan on any exit, and focus instead on building a solid development company.

B. Types of Wireless Game Developers
So where do you get started? Before thinking about whether you have the right skill set, have a client or know people who might want to start a company with you — you need to decide what type of mobile phone game development you want to do.

a) Porting House
One of the easiest and arguably the most stable type of development to start is a company that takes already created games and ports them across a wide spectrum of phones. The founders of FingerTwitch realized this when they first started their company and rode their company to acquisition by Mforma. This may be the least glamorous and least creative type of development work but it can be very steady at the industry grows. You will need capital to purchase the 30-50 handsets necessary to do the work. Also you’ll need several phones with service agreements to do proper testing of your ports. It helps if you have a strong testing and engineering background and are process-oriented.

b) Game Developer with outsourced resources
If you have connections in foreign countries and can leverage the low cost of labor in these countries, you may be able to parlay this into a rapid growth company while leveraging cost margins. Cybiko Wireless is a good example of this model as they set up their development group in Russia.

c) Game Developer – Niche
Is there a genre of games you know well, like sports or racing? Make your reputation known by these types of titles. It may be hard to sign deals for the first few titles you do, but once you have established your reputation in a field, several publishers will ask you to do the same game for them. As you grow your toolset and engine, second generation titles will increase your value, your revenue and your opportunities.

d) Game Developer - Publisher Focused
This is the most common type of game developer. One that likes to work on many different types of games, normally for a certain publisher. Sometimes this relationship is exclusive, therefore giving the publisher the confidence to award you more work. While this can certainly help you enter the market, it can severely limit you.
e) **Original Concept Game Developer**
Publishers love hearing pitches on new game ideas from developers. But remember they tend to make the most money from branded-title games – games like NFL Football or Superman Curling. Unless you’re well funded, I would not recommend starting a company with this focus, at least initially. I have seen a lot of companies try and not succeed at getting publisher distribution for their unique ideas. It’s better to keep the money coming in with development contracts and have a team working on the side developing your original idea.

f) **Self-publishing Developer**
For some, the ultimate goal is to create and distribute your own game. A majority of the major carriers already have publicly stated that they are not planning on working with new developers or new publishers. Some carriers do hold slots for truly unique innovative products, but normally they insist you go through an aggregator to get it to them. They simply do not have the resources to manage a lot of developers.

C. **Finance Your Company**
Once you’ve figured out what type of developer you’d like to be, there’s also the matter of funding to think about.

a) **VCs**
Unless your best friend is a VC, don’t bother trying. Even though the market holds a lot of interest right now, you normally need a proven revenue stream to gain VC money. Game developers are too low on the mobile phone game food chain for a VC to be interested.

b) **Angel Investors**
This is your best possibility for outside funding. With the wireless market of high interest right now, with good due diligence and networking you can find Angles willing to invest if you have a unique game concept, a group of engines, or a solid tool set and plan for growth. If you know investors who would be willing to invest, then by all means work with them. Just keep a controlling interest and do not give up more than 10-30% of your company. Usually you will only be able to gain funding if you already have a solid reputation in the game industry.

c) **Win the lotto**
This is a great way to get some (not much) money to help you get things started. Keep buying those scratch-off cards.

d) **Bootstrap It**
The normal way we all get started. This usually this means living on savings for six months, building a lot of prototypes, dogging a publisher to give you a chance, and then living off advances until you can catch up. Try to get to breakeven 6 months after your first contract. This is the most critical time for your new company and no matter how great it looks, don’t over extend. There will be more opportunities for expansion later.

D. **Market Conditions**
As the market continues to grow, advances and royalties paid to developers, on average, have been steady. In some cases royalty percentages appear to be getting into the high single digits. Publishers and carriers are beginning to recognize the value of high quality, well designed games. Of course there are plenty of exceptions to this rule as some savvy publishers do have double digit royalties but killer terms. Be careful and in the early days of your growth as its more important to have cash in the bank for payroll then the promises of a big royalty 6 months down the line.

Advances on complex projects can be high but the overall average is steady because of the simpler game advances declining. Publishers are now more willing to give simple games out to newer developers but they do not pay much.
If you’re considering getting in the game, a common strategy is to offer very low bids, in an effort to win a first contract. Of course, if you are successful and win the work, the publisher will ask for the same price again and it will be harder for you to make the margins you need to grow.

Competition is coming from everywhere. The rumors of a 5 billion dollar market by ’07 have everyone now looking to get into the business. There are more console or handheld game developers coming into the market everyday, and current developers are taking their costs down by outsourcing to cheaper labor markets. These dynamics put constant downward pressure on development fees. On the flip side, new technologies like 3D and multi-player can increase project fees and size and help smooth out a developer’s pipeline. So it’s suggested you have a smart combination of both.

a) Unique to Cell Phone Development
You need to understand that there are two major areas that are unique to cell phone game development. The very short cycle times of a game project and the number of devices your game needs to run on. This impacts the way you should approach you’re selling and team set up.

The short project cycles create a need for developers to focus a large portion of their time on finding new projects. If you want to start your own company, make sure that someone on your team is focused on putting a majority of their time into developing new business. This is a must if you are to survive the first 18 months. If you don’t have a new project ready to start right when the first one ends, you will be burning cash in unproductive idle time. As your reputation grows you can spend less time at this.

Also, the number of devices you may need to supply has a direct effect on the costs of the project. One concern you should address early on is how to easily port your game across multiple gold master devices and be sure it will work on the low end handsets. Different developers have taken different approaches, with the most successful developers creating specialized porting tools and in some cases, porting teams. Publishers want to be sure when they get a game from you that they can make money from it. Another concern you should be prepared for is the cost of buying several different phones for testing. A good publisher will help supply you with devices, but many will ask you to provide the phones for testing yourself. With a monthly plan factored in over a year, a new phone will usually cost at least $300 and you’ll need a lot of them.

E. Really Getting Started
In writing about starting your own mobile phone development company, I tried to be as realistic as possible. The majority of developers that started between 2001-2003 are still around and many of them are doing well. Some developers that started in early 2004 have also fared well, but beginning in the second half of 2004, I began to see new developers encountering far more difficulties. Publishers and brand owners are setting the bar much higher for a high quality game then every before. New entrants into the space generally lack the experience to understand all the handset tricks and do not have the proper connections to get it right the first time out. So be sure to immerse yourself in every aspect of the business, find others that are willing to share information and play a lot of cell phone games.

So if you’re starting now, I think you can still be a success. But be smart. Think carefully about what type of developer you want to be and what type of progression makes the most sense given your strengths and weaknesses in design, coding, and art. And for tech heads, make sure that you are willing to deal with the business and selling end of the company, too. This is where most developer oriented companies, including our company’s first iteration can fail.

If you have thought things out carefully, have the support of family and friends, are willing to take big risks, can sacrifice time and money, and really want to start a company – you can succeed. Good luck!
X. Closing

We hope you found the information in this document of value. It is the result of the efforts of many volunteers with extensive industry experience that have been willing to donate their time so that others can obtain important information about the mobile game industry.

We invite you to volunteer yourself if you are interested in contributing to subsequent updates of this document in the coming years. In a fast moving industry like mobile games it is certain that there will be new information to share on a continuing basis. Information for volunteering can be found at [www.igda.org/online](http://www.igda.org/online). We also encourage you to view other documents provide by the Online Games SIG such as the Web and Downloadable Games Whitepaper.
XI. Version History

5/17/2005 – Original Release
- Pg. 42 – Minor corrections within the porting framework section
- Pg. 53 – FingerTwitch was purchased by Mforma, not ThumbJive as originally written