Multiplayer Mobile Games: Business Challenges And Opportunities

Version 1.0; April 16, 2004
Copyright © 2004 Nokia Corporation. All rights reserved.

Nokia, Nokia Connecting People, and Nokia N-Gage are trademarks or registered trademarks of Nokia Corporation. Java and all Java-based marks are trademarks or registered trademarks of Sun Microsystems, Inc. Other product and company names mentioned herein may be trademarks or trade names of their respective owners.

Disclaimer

The information in this document is provided “as is,” with no warranties whatsoever, including any warranty of merchantability, fitness for any particular purpose, or any warranty otherwise arising out of any proposal, specification, or sample. Furthermore, information provided in this document is preliminary, and may be changed substantially prior to final release. This document is provided for informational purposes only.

Nokia Corporation disclaims all liability, including liability for infringement of any proprietary rights, relating to implementation of information presented in this document. Nokia Corporation does not warrant or represent that such use will not infringe such rights.

Nokia Corporation retains the right to make changes to this specification at any time, without notice.

License

A license is hereby granted to download and print a copy of this specification for personal use only. No other license to any other intellectual property rights is granted herein.
## Contents

1 Introduction ........................................................................................................................................... 5
2 Challenges Facing Multiplayer Mobile-Game Providers ................................................................. 6
  2.1 Providing Multiplayer Games ........................................................................................................ 6
    2.1.1 Remote-server games ............................................................................................................. 6
    2.1.2 Quasi peer-to-peer games ..................................................................................................... 7
    2.1.3 What this means for mobile .................................................................................................. 8
  2.2 Who Pays? ....................................................................................................................................... 9
  2.3 How Much Does The Consumer Pay? .......................................................................................... 9
  2.4 Latency .......................................................................................................................................... 10
  2.5 Different Development Skills Are Required .............................................................................. 11
  2.6 Addressing The Challenges ......................................................................................................... 12
3 Fixing The Business Model .................................................................................................................. 13
  3.1 Treating The Installation Fee As A Subscription ...................................................................... 13
  3.2 Using SMS As The Revenue Generator ..................................................................................... 13
  3.3 Sharing Airtime Or Data Traffic Revenue .................................................................................. 14
  3.4 The Business Model In The Future .............................................................................................. 14
4 Common Server Issues ....................................................................................................................... 16
  4.1 Server-Side Platforms .................................................................................................................. 16
  4.2 Player Matching ............................................................................................................................ 16
5 Multiplayer Games Benefit All ............................................................................................................. 18
6 Conclusion ........................................................................................................................................... 19
7 References ............................................................................................................................................. 20
### Change History

<table>
<thead>
<tr>
<th>Date</th>
<th>Version</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>April 16, 2004</td>
<td>V1.0</td>
<td>Initial document release</td>
</tr>
</tbody>
</table>
1 Introduction

Mobile phones are social devices. People use them primarily to keep in touch with friends, family, and business contacts. Prior to the invention of computers, games also were primarily social. People got together to play games with friends, and socialization — “table talk” — was, except for highly cerebral games such as chess, a big part of the appeal of game playing.

In the world of conventional PC and console games, games have not been social. The earliest home devices were inherently single-user in nature (although some console games have always been playable by two people with two controllers), but the spread of the Internet has brought about a big change. PC games are moving away from strictly solo experiences toward multiplayer gaming. Massively multiplayer games (MMGs) such as EverQuest and Lineage are among the top revenue-generating games in the PC market — not because of high unit sales but because of ongoing subscription revenues. Games that are inherently for solo players — graphic adventures and single-player strategy games, for example — have lost market share. But games that provide online play, such as real-time strategy (RTS) and first-person shooter (FPS) games, have become more popular. While most PC gamers, most of the time, continue to play alone, multiplayer online play is an increasingly important part of the market.

Console games have been slower to become connected, and console gamers still face higher costs (in terms of additional hardware purchases and ongoing subscription fees) for online play. But games such as Final Fantasy XI and Halo (playable in deathmatch mode over the Xbox Live service) are growing in popularity.

It took a long time for PCs — and even longer for consoles — to get connected. The earliest PCs and home game systems were available in the late 1970s, but not until the late 1980s did commercial online services become available. And not until the late 1990s did use of the commercial Internet become widespread. Thus, it was natural that solo-play games dominated initially.

Mobile phones are different though. Mobile phones are, by nature, connected. They are social devices; games are social environments. It is logical that, on mobile phones, multiplayer games should be popular. Indeed, a priori, one might expect multiplayer games to dominate.

Yet today virtually all revenues generated by mobile games are derived from single-player games.

Why? What are the technical and business obstacles that have slowed the development of a market for multiplayer mobile games? What needs to be done — both in technical terms and in terms of changes to the value chain — to make a success of multiplayer mobile gaming? Is there a business case for multiplayer mobile games? If so, what does it look like?

Some of these questions can be answered only in the future. The purpose of this white paper is to pose the questions to developers currently working to make a reality of multiplayer mobile games and to suggest fruitful paths for future development.
2 Challenges Facing Multiplayer Mobile-Game Providers

A variety of technical and business challenges face multiplayer mobile-game providers.

2.1 Providing Multiplayer Games

There are essentially two ways to provide a multiplayer game experience. The first method is via a remote server. The second is what this document calls quasi peer-to-peer. (The term "quasi" is used because some games that use this method are not true peer-to-peer applications.)

2.1.1 Remote-server games

When a game is offered via a remote server, the player runs a client application on his or her machine. The application connects over the network to the server. Each player communicates with the server separately, and the server is responsible for receiving all players' actions, determining what happens as a result, and returning new game-state information to each player. Figure 1 illustrates this type of networking, which is often called a hub-and-spoke configuration.

![Diagram of remote-server configuration]

Figure 1: The remote-server configuration is also known as a hub-and-spoke configuration.

There are obvious security advantages to this approach: Game data are processed and verified by the remote server, and there is a limit to how much advantage a player can gain by hacking his client software. More important, however, this is the only way to supply MMGs. When thousands of people are playing in the same game world, large amounts of bandwidth are required to receive all players' actions and responses. A substantial server farm and a complicated back-end setup that integrates databases, player information, game updates, and, of course, billing systems are also needed.

For MMGs, in particular, this means that the game operator bears substantial ongoing costs for bandwidth, hardware, and maintenance. Players understand this and that it is necessary to pay a subscription fee to keep the game online, and they bear this fee willingly. This is why such games as EverQuest and Lineage are successful.

It is, however, possible to use a remote-server configuration to provide other sorts of games. Blizzard Entertainment does so for online play of Diablo II, for example, mainly for security reasons. Pogo, Uproar,
and Yahoo! Games provide play of classic multiplayer board and card games this way, defraying their (more modest) expenses by displaying advertisements to the players. And other game providers, such as Codo Games Zone, which offers Laser Squad Nemesis, provide smaller-scale strategy games in this fashion, charging a more modest yearly or biannual subscription fee.

2.1.2 Quasi-peer-to-peer games

In a true peer-to-peer game, illustrated in Figure 2, each player’s machine sends data to and receives data from all other players’ machines. Each machine checks to ensure that it is in sync with the others. This approach is rarely used in PC gaming, however. More often, one player’s machine acts as a server for the others, and a hub-and-spoke configuration is used. The other players’ machines communicate only with the server, but chat may be sent directly from machine to machine.

![Diagram of true peer-to-peer configuration](image)

Figure 2: In a true peer-to-peer configuration, each player’s machine sends and receives data from all other players’ machines.

In some games, such as Blizzard’s StarCraft, players initially contact a remote server, which matches players and hands off the game to one player’s machine, and that machine operates as server for the others during game playing. (See Figure 3.) There are several advantages to this method: It allows Blizzard to track usage and how its players use its game; it provides an easy, quick, and immediate way for players to get into a multiplayer game; and the server can check ping times in order to appoint the player with the best connection as the server, improving the experience for all the players. This approach does, however, mean some ongoing support costs for the game provider, which must maintain the player-matching servers and the bandwidth to support them. The cost, however, is much lower than for an MMG: Once a game begins, the players themselves pay their own hardware and bandwidth costs. For games such as StarCraft, the game provider is gambling that online play will spur enough additional retail sales to justify the relatively modest cost of maintaining the ongoing player-matching service. (It should be noted that Blizzard’s Battle.net service also serves revenue-generating ads to players during the player-matching period.)
In other games, there is no central player-matching service. Instead, players must find servers on their own. Quake, for example, works this way. One player might set his or her machine up as a server, contact friends, and ask them to join in playing a game. Some people keep servers active on the Internet at all times, and some companies (for example, GameSpy) maintain servers in order to attract players to their other Internet offerings. This approach has an advantage: The game provider undertakes no ongoing costs to provide online game playing. However, this does make it harder for players to find opponents online. Third-party tools, such as GameSpy Arcade, help solve this problem by performing automated searches for servers, and GameSpy encourages PC game developers to integrate with its software, easing the problem for players.

2.1.3 What this means for mobile

In most cases, the quasi peer-to-peer approach does not work for mobile devices. It works in the PC world because every computer has an Internet Protocol (IP) address, and once the players’ machines know the IP address of the server and the server knows theirs, communication is easy.

Mobile phones generally do not have IP addresses. They can communicate over the air (OTA) to a remote server located on the Internet, the Internet server can respond, and, thus, remote-server games are feasible. However, there is no way for one phone to communicate data directly to another phone, except through a remote server: A phone cannot tell another phone its IP address because it does not have one.

Thus, it is not (in most cases) feasible for one device to act as a server for the others or to play in true peer-to-peer mode.

This is, however, feasible via Bluetooth. Devices, such as Series 60 phones that use Bluetooth wireless technology, can establish a “piconet” with other nearby Bluetooth devices — as much as 10 meters away. Because Bluetooth piconets are limited to eight participants, Bluetooth games are limited to no more than eight players. (For bandwidth reasons, Nokia suggests limiting such games to four players.) Eight (or four) players are quite enough for deathmatch play and many other multiplayer-game styles. And indeed, some successful games — for example, the version of Red Faction for the Nokia N-Gage mobile game deck — take advantage of this capability.
But for devices that do not use Bluetooth wireless technology and for any game played OTA rather than with a local Bluetooth piconet, there must be a server in the mix. This means that someone must pay to buy, operate, and maintain that server, as well as for Internet connections and the bandwidth to transmit game data to players. Generally, “someone” means either the mobile-game publisher or the mobile-game developer.

2.2 Who Pays?

The business model for single-player mobile games is now well established and widespread: The user pays a modest fee — a few dollars or euros — to download and install the game on his or her handset. Users can then play the game to their hearts’ content. The purchase fee is split in some fashion among the operator (or portal site such as Handango), game publisher, and game developer, and sometimes another party — a billing or technology provider — is involved as well.

To say “well established” is perhaps to overstate the case. The model has become widespread only in the last two years, and it took the huge success of NTT DoCoMo to persuade other operators to share a substantial portion of game revenues with providers. But that battle has been won, and worldwide, over every sort of mobile network, the single-player model is in place for devices running Java™ 2 Platform, Micro Edition (J2ME™), Binary Runtime Environment for Wireless (BREW), or Symbian OS.

But multiplayer games are different, at least when they must be provided via a remote server: Providing multiplayer game playing means ongoing costs for the game provider.

In the PC world, some game publishers are willing to pay this kind of cost if it is modest and if it spurs enough additional retail sales. Blizzard, as previously mentioned, does this with StarCraft. But it is more common for game providers to seek an ongoing revenue source that will offset ongoing costs, either via subscription (for example, EverQuest) or online advertising (Pogo, UpRoar, and Yahoo! Games).

In short, the business demands of multiplayer mobile games do not match well with the current business model for (mainly single-player) mobile games. Multiplayer games involve ongoing costs and require ongoing revenues; the mobile-game business model involves one-time payments and no ongoing revenue stream.

2.3 How Much Does The Consumer Pay?

Games played OTA inherently involve using the air network, and mobile operators charge for that use.

The amounts they charge vary widely, both from operator to operator and by time of day. Thus, a customer who has not used up his or her monthly allotment of minutes might pay an effective rate of $.10 USD per minute of airtime during peak hours, $.01 USD per minute at night — and $.30 USD per minute after exceeding the monthly allotment. But if we assume that $.10 per minute is typical, one hour of game playing will set a customer back $6 USD.

Most mobile games are designed to be played in much less than an hour, but this is still a lot to pay for game playing. Back in the days of such commercial online services as Prodigy, Genie, CompuServe, and America Online (which started as a non-Internet service), people normally did pay $3 USD to $6 USD per hour of connect time, but far fewer people used those services than now use the Internet.

The problem is not just cost; it is also the unpredictability of the charge. Games were popular on the old commercial online services, but some players ran up charges of hundreds — sometimes even thousands — of dollars in a single month. A player hit by a big bill at the end of the month would likely quit the game and never return.
To reach a mass audience, online services needed to start charging flat monthly fees. And to reach a mass audience, multiplayer games likewise had to turn to flat subscription fees.

Today users who wish to play a multiplayer game with a mobile device start by buying — and paying for — the client application. Then, each time they connect, they pay more. And, as with the old commercial online services, they may find themselves facing large bills at the end of the month.

For multiplayer mobile games, in short, there is not only a business model mismatch from a game provider’s point of view; there is also a charge expectation mismatch from the player’s point of view.

2.4 Latency

Latency — the amount of time it takes to get data from one machine to another and to receive a response — is an issue with which all connected games struggle. On a single computer, latency (the amount of time between a user’s input and a response) is calculated in terms of microseconds. On the Internet, latency is about 100 milliseconds.

That is a big difference — so big that some games that work well on a stand-alone computer or console are not feasible over the public Internet. Fighting games such as Soul Calibur and Virtua Fighter offer one example: Action is so quick in these games that even 100 milliseconds of latency is not tolerable.

Even slower-paced action games, such as FPS and RTS games, have problems with Internet latency, particularly when it creeps up into the several-hundred-millisecond range, as it does occasionally. Developers deal with the problem in a number of ways: by using predictive algorithms and fixing what is displayed on-screen when new data are received and by hiding latency in clever ways. (As an example of the latter approach, Quake is designed so that it takes several-hundred milliseconds for a bullet to fly from a gun to its target, providing enough time for the machine to send and receive data over the Internet and display animation that shows whether or not the target was hit.)

OTA, latency is even worse. In theory, latency is about one second. In practice, however, it is several seconds, so developers need to allow for three to five seconds of latency.

Why? The answer is simple. HTTP, the only widely supported networking protocol available on mobile phones, is an excellent protocol in some ways: It guarantees that data will be received. It also involves multiple exchanges of handshaking information between the device requesting data (the handset) and the device sending data (the server). This is how it works: The handset says, “I’m here. I want to communicate with you.” The server says, “OK. Go ahead.” The handset says, “Here’s what I want.” The server says, “Got it. Here’s what you requested.” The handset says, “Thanks, I got it.” The server says, “OK. Good-bye now.” Each of these exchanges of data involves a separate communication OTA, and thus the one-second latency typical of the air network is multiplied.

On the Internet, most games do not use HTTP; they use User Datagram Protocol (UDP). UDP does not guarantee data delivery, but it does not involve extensive handshaking. Most games can tolerate the occasional dropped packet, and users prefer the speedier delivery of data. Use of UDP in mobile games would reduce latency somewhat — by about half. (See Section 7 “References,” Multiplayer Game Performance Over Cellular Networks, for more data on this subject.)

UDP support is becoming more widespread; Nokia devices that support Mobile Information Device Profile (MIDP) 2.0 (the most recent version of J2ME™ for mobile devices) do allow developers to use UDP (and TCP sockets). Series 60 C++ applications can also use it. But at the present time, we cannot expect UDP to be available on all handsets from all manufacturers.
Whether you use HTTP or UDP, however, latency of one second or more still means that multiplayer action games are not feasible on today’s mobile devices (except via Bluetooth). This is a problem because many of the most popular digital games are action games — FPS, RTS, MMGs, fighters, platformers, and so on.

Even today, however, some types of multiplayer games are feasible for mobile devices. See Section 7, “References,” Overview of Multiplayer Mobile Game Design, for more on this topic.

Will latency improve? Very likely it will, as operators move toward third-generation (3G) networks. This is not guaranteed, however. Much depends on how operators configure their networks and how much priority they give to data delivery relative to voice delivery. However, recent tests by Nokia demonstrate that over a Wideband CDMA (WCDMA) network, latency of a few hundred milliseconds is feasible (see Section 7, “References,” Multiplayer Game Performance Over Cellular Networks).

2.5 Different Development Skills Are Required

A server-driven multiplayer game is, in essence, two software-development projects in one. Software must be developed to run on mobile phones and also the servers that match players and adjudicate multiplayer games. The technologies that run on phones (J2ME™ and C++ Symbian OS applications, for example) are different from those that run on servers (Java™ 2 Platform, Enterprise Edition (J2EE™), Web servers, databases, and so on). Consequently, a mobile-game developer familiar with single-player game development may find the need to acquire new skills (or hire new staff) to produce the server-side code. And because these two pieces of code need to work together and each is developed separately, developers will find that, as a general rule, developing a multiplayer mobile game is somewhat more costly than developing a single-player mobile game.

Some who have looked into this issue have also been daunted by the expense of operating servers for MMGs in the PC world. Operations such as Sony Online Entertainment and EA.com have enormous server farms, and they have made substantial investments in solutions that scale to support thousands or hundreds of thousands of simultaneous users.

Developers looking at multiplayer mobile games should not be deterred by this, however. Supporting a few hundred players simultaneously involved in a mobile game is very different from supporting a hundred thousand players of EverQuest. Each player of a mobile game consumes less bandwidth: The air network would not support data transfer at rates required for MMGs, and mobile games generally require much less data transfer anyway. Players of games such as EverQuest spend hours online in a single game session; mobile-game players typically spend no more than a few minutes at a time. And today, even if a mobile game is widely distributed across many operators, it is unlikely to attract the same number of players as the most successful online operations.

Thus, the solutions required for the big online operations are not required for multiplayer mobile games. One physical server suffices to support even the largest multiplayer mobile game, and in some cases, one server can handle the demands of several different titles. For example, BotFighters, probably the most successful multiplayer mobile game currently on the market, with thousands of simultaneous players in three countries at peak usage times, is served by a single machine.

As the market and audience grow, it may become necessary to implement more scalable and robust solutions for serving multiplayer mobile games, but it is likely that the larger audience will be generating enough additional revenues to fund development of solutions.

In the meantime, developers interested in multiplayer games should be mindful that different skills are required and the cost is somewhat higher, but they should understand that the cost is not enormously higher and that the problems are manageable.
2.6 Addressing The Challenges

In summary, there is a business model mismatch; consumers are likely to feel overcharged; latency is excessive; and it is costlier to develop multiplayer mobile games because they involve both client- and server-side programming. In the face of such obstacles, developers may be tempted to throw up their hands, but multiplayer mobile games are both possible and in deployment today.
3 Fixing The Business Model

Today developers are taking several approaches to ensure a continuing revenue stream through game playing. The problem, however, is that all of them rely to some degree on having the cooperation of operators who have not fixed on a consistent approach to dealing with multiplayer mobile games.

3.1 Treating The Installation Fee As A Subscription

One approach, adopted by JAMDAT Mobile Inc., in Los Angeles, and Unplugged, Inc., in Berkeley, California, uses the existing “pay to install” model for single-player games, but it is modified so that payment gives the user a license of a limited duration — generally, one month.

Once the period for which the user has paid expires, the game prompts the user to resubscribe and does not allow him or her to play without paying an additional fee.

“Typically, we offer a monthly subscription for a smaller amount, around $3, or a six-month subscription for a larger amount, say, $7,” says Jonathan Zamick, CEO of Unplugged. Most of Unplugged’s multiplayer titles, including versions of backgammon and checkers, as well as Qube, an original multiplayer puzzle game, can be played both in solitaire mode with no air network connection and in multiplayer OTA mode. “We ‘turn off’ only the multiplayer version at the end of the subscription,” says Zamick. “So people who just want to play by themselves can treat the game in the same way as a solo-play title.” A high proportion of downloaders do just that.

According to Zamick, his company launched its first multiplayer titles on BREW carriers because Qualcomm’s billing system allows limited-duration licenses. Unplugged is now working with carriers that support J2ME™ as well. With Java™, however, use of a subscription model depends on whether or not the carrier’s billing solution permits it. “But of course, this is just another carrier-integration issue, and mobile developers already face many of those,” he notes.

3.2 Using SMS As The Revenue Generator

Many of the earliest mobile games were based on Short Message Service (SMS). Players were charged per message, and the revenue was shared in some fashion by the players’ operator and the game provider. In some cases, these games survive. As Glenn Broadway of iomo Ltd., in Eastleigh, Hampshire, England, says, “A product called The Love Game is still generating fairly high levels of revenue for us. This is more like a game than a dating service. The players set up a character, and the game pairs them up with potential matches. Paired players then get the chance to communicate with each other (via WAP) with the game sending out alerts via SMS, often with WAP push links.”

In some cases, it is possible to use an application that uses J2ME™ as the front end for a multiplayer game in which communication is via SMS. In 2000 Siemens launched the mobile version of BattleMail Kung-Fu, a now-defunct online game originally developed for PCs. The original version used e-mail for communication between players, and the mobile version used SMS. It enjoyed some early popularity, although usage trailed off as the most active players discovered they could avoid SMS charges and play the same game for free from their computers.

Another game that enjoys continuing success is BotFighters, from It’s Alive, based in Stockholm, Sweden. BotFighters is a location-based services (LBS) game in which each player controls a combat robot and travels in the real world to find and fight opponents. To launch an attack on another robot, a player needs to be within two kilometers of another player. Players “scan” for opposing robots and launch attacks on them by sending SMS messages to the game’s server.
It’s Alive now has a Java™ client for the game that communicates with the server via HTTP rather than SMS; players who use that version have the option of paying by the message, as they would if they were using the SMS version, or paying a monthly subscription charge of 6 euros ($7.17 USD). According to Tom Söderlund, the game’s producer, most users of the Java™ version continue to pay by the message.

For users of an operator’s network to play the game, the operator must be willing to share LBS information with It’s Alive. So BotFighters is available only in limited regions (the game has been launched in Sweden, Finland, Ireland, and around Moscow). Despite this, it has been both profitable and successful for It’s Alive; BotFighters has more than 20,000 continuing users in Russia alone and around 10,000 in Sweden. “Some of them spend a hundred or more euros a month to play,” Söderlund notes.

3.3 Sharing Airtime Or Data Traffic Revenue

In 1999 and 2000, when WAP games were more prevalent, many operators were willing to share a portion of airtime revenues with game providers. This was a necessity because there was no other way to compensate the providers of such games.

With more modern multiplayer games, however, few operators have been willing to share airtime or data traffic revenues. In at least one case, a game provider has a similar deal with an operator.

In May 2003, T-Mobile Germany launched TibiaME, developed by CipSoft GmbH of Regensburg, Germany. (Since then, TibiaME has been launched by T-Mobile UK and T-Mobile Austria as well.) TibiaME is something of a milestone in the development of multiplayer mobile games: the first true MMG for mobile devices. It is based on CipSoft’s earlier Tibia, a PC-based MMG originally launched in 1997. Tibia is a combat-oriented MMG in the mold of dikuMUD or EverQuest, with hundreds or thousands of people playing in the same world, fighting monsters and each other. Unlike EverQuest, it is not a high-end, three-dimensional game, however; it uses two-dimensional sprites, and its client software — a 4-MB download — is small by PC standards. The mobile version is even smaller — a 150-KB download expanding to 380 KB once it is installed on a phone. An application of this size and complexity will not work, however, on most Java MIDP phones. TibiaME was coded in C++ for Series 60 phones.

T-Mobile and CipSoft decided on an unusual business model for TibiaME: There is no charge for the client application and no monthly subscription fee.

Instead, users pay for the data traffic they create over T-Mobile’s General Packet Radio Service (GPRS) network. The actual cost to play varies according to the amount of traffic and the specific tariff option that each user selects. The base cost starts at .10 euros ($0.12 USD) per 10 KB, and the cost can be as low at .02 euros ($0.02 USD) per 10 KB for high amounts of traffic. According to Stephan Vogler, CipSoft’s project manager, the typical player generates about 400 KB in traffic per hour of play, for between 4 euros ($4.78 USD) and .80 euros ($0.96 USD) per hour, depending on the player’s deal with T-Mobile. T-Mobile pays CipSoft a fee for each user, based on server usage. Vogler says “This is not a real airtime revenue share, but it is almost equivalent.”

This is still a fairly high cost to pay for play, particularly when the PC version can be played for free (although a premium account requires a subscription). Despite this, however, and despite the limited geographic spread and limited number of mobile users with phones capable of running the game, TibiaME has hundreds of active players and is profitable for CipSoft.

3.4 The Business Model In The Future

As operators move toward packetized networks, airtime charges will, for data services, be replaced by data traffic charges. At present, those charges are still high enough to deter much multiplayer game playing. Most mobile games are designed to be played in short sessions. Because they are time-wasting
entertainment, people who want a lengthy game session generally find a computer or console. But at the
equivalent of, say, $6 USD for an hour of game playing, even a five-minute game sets a player back $.50
USD. One session every day adds $15 USD to the monthly phone bill. Although gamers willingly pay as
much as $15 USD per month for subscriptions to MMGs, the typical MMG player also spends 30 hours per
month in the game — not less than three hours.

Data transfer fees will not go away. Operators invest large amounts of capital in building air networks, and
they need to generate revenues through their use to earn a reasonable return. However, it is likely that
fees will decline over time, just as the price of voice telephony has fallen.

Until prices decline, mobile-game developers are best advised to work to minimize the amount of data
traffic their multiplayer games require, passing deltas rather than complete game states, compressed
data, and so forth. Over packetized networks, charges are no longer calculated by the minute but by the
byte, in effect. By keeping the number of bytes transmitted as low as feasible, developers can minimize
the ongoing cost to the consumer without reducing their own revenues, unless, like CipSoft, they have
operator deals that permit sharing of data revenues.

Because multiplayer mobile-game providers require an ongoing revenue stream to offset ongoing support
costs, it would be nice if mobile operators were more generally willing to share data revenues. However,
many are not, and in general, it is often difficult for operators to determine what service is responsible for
how much data traffic by whom, and to determine how these revenues should be split. No pan-industry
solution to this issue has yet appeared on the horizon.

Thus, at least for the near term, multiplayer mobile-game providers are faced either with a subscription
model, which at least some operators are equipped to provide, or a one-time fee model that assumes
revenues will be adequate to cover operating costs (the Blizzard model). In the medium term, it is
worthwhile for both operators and developers to work to extend the possibility of the subscription model
more broadly to help foster the spread of multiplayer games.
4 Common Server Issues

A number of issues are common to most or all multiplayer mobile games: user authentication and sign on; the need to match players with opponents; and accepting requests from clients and responding to them.

4.1 Server-Side Platforms

A number of companies have developed server-side “platforms” that address these issues and have sought to make them widely available. Among these companies are Digital Bridges, Cash-U Mobile Technologies, and Mforma.

A standard server-side platform, if universally adopted by operators, would in principle make multiplayer mobile-game development easier. It would allow developers to concentrate on game-specific server code, without having to worry about such common issues as user authentication — at least if use of the platform does not require the game developer to sacrifice too much of its revenue stream.

In practice, however, the various platforms are not entirely compatible; few operators have adopted one exclusively; and platform providers want fees that most developers do not find appealing.

In response, major mobile-phone manufacturers, including Nokia, have established the Mobile Games Interoperability Forum (MGIF) (http://www.mgif.org), now part of the Open Mobile Alliance (http://www.openmobilealliance.org), to establish a standard set of APIs for such platforms. Some platform providers have announced that they will make their proprietary platforms compliant with whatever the MGIF specifies in the future. However, this initiative remains at an early stage, and it remains to be seen whether adoption will be widespread.

At present, except when a partner such as an operator or publisher requires them to use a vendor’s platform, most developers write their own code to handle user authentication and so on. Because the cost of doing so, in terms of development time, is not enormous, particularly given the relatively modest server-side demands on current multiplayer games, this is likely to continue for the foreseeable future.

4.2 Player Matching

Some multiplayer-game developers report a problem in finding opponents for players. The problem partly derives from the limited acceptance of mobile games and the division of markets among many operators, and partly from the short play sessions typical of mobile gaming.

At times, players show up at a server for a multiplayer game, and there are no opponents waiting for them. If the wait is longer than a few minutes, the players are likely to depart.

Obviously, the best solution is to ensure that the game is distributed widely enough that there is always a critical mass of ready players. But particularly at this stage in the market, it is important for developers to be aware of the problem and find ways to work around it.

The simplest solution is to provide artificial-intelligence (AI) opponents when live opponents are not available. This is, of course, no different from a solo-play game, but at least it provides a way to avoid frustrating players.

Another is to tackle the problem through design. For example, some games are persistent, allowing a player to attack or otherwise interact with other players even when they are offline.
A third useful technique, which also serves a viral-marketing purpose, allows players to challenge people they know via SMS.
5 Multiplayer Games Benefit All

Although multiplayer mobile games face a number of challenges, fundamentally all parts of the value chain have strong incentives to ensure the growth and spread of multiplayer games.

As operators move to GPRS and 3G technologies, they need to find ways to increase average revenue per unit (ARPU) by encouraging phone users to adopt mobile-data services. As the experience of the Internet shows, games are among the few network data services for which consumers will willingly pay. Although few operators today have made multiplayer games a priority, they are likely to do so in the future to foster network usage. Indeed, in South Korea, one of the most advanced markets for mobile-data services, SK Telecom has already made a priority of multiplayer games.

Game publishers and developers, too, can reap higher revenues from multiplayer games than from single-player games, at least as a business model that supports them spreads, because they generate ongoing revenues.
6 Conclusion

The critical requirement for the success of multiplayer mobile games is the spread of a business model that is acceptable to consumers and that provides ongoing revenues to offset ongoing support costs. Business models that do so are beginning to appear, but operators and developers must work to spread them more widely.

Mobile-phone manufacturers such as Nokia can help by making lower-latency network protocols such as UDP and TCP more commonly available on their devices, as Nokia is beginning to do with its MIDP 2.0 phones.

Operators can help by pricing mobile data at rates that appeal to consumers; by being willing to share revenues with developers to encourage the development of high-quality multiplayer games and spur increases in data transfer revenues; and by working to reduce network latency, particularly as they move to 3G technologies.
7 References

The following documents are available at http://www.forum.nokia.com/games:

* Multiplayer Game Performance Over Cellular Networks
* Overview of Multiplayer Mobile Game Design
* Multiplayer MIDP Game Programming
* Case Study: It’s Alive’s BotFighters
* TibiaME Case Study

Other resources:

* Mobile Games Interoperability Forum (http://www.mgif.org)
* Open Mobile Alliance (http://www.openmobilealliance.org)