White Paper on Mobile Gaming Evolution

Approved – 10 Jun 2008

Open Mobile Alliance

OMA-WP-MGPC-20080610-A

Contents

1. SCOPE....4

2. REFERENCES....5

3. TERMINOLOGY AND CONVENTIONS....6

3.1 CONVENTIONS....6
3.2 DEFINITIONS....6
3.3 ABBREVIATIONS....6

4. INTRODUCTION....7

5. MOBILE GAME PERFORMANCE CHARACTERISTICS....9

6. DEVICE CAPABILITIES AND FEATURES....12

6.1 MEMORY....12
6.2 MULTIMEDIA....12
6.3 GRAPHICS....13
6.4 DISPLAY....14
6.5 KEYBOARD / KEYPAD....15
6.6 ERGONOMICS....15
6.7 CONNECTIVITY....16

APPENDIX A. CHANGE HISTORY (INFORMATIVE)....17

Tables

Table 1: Recommended capabilities for mobile games....9

Table 2 Minimum Graphics Performance....10

Table 3 Graphics Features for Benchmarking....11

1. Scope

This white paper aims to stimulate the market for mobile gaming by driving content and devices to better meet gaming consumer expectations. The Open Mobile Alliance focus is on achieving industry alignment of mobile handset feature characteristics that impact game user experience and porting costs to increase quality and quantity of premium mobile games. The guidelines captured in this white paper can be used by content providers, mobile network operators, and handset manufacturers to create devices, content, and services for the mobile gaming market. As such, it is not in the scope of this paper to define a universal classification system and device appraisal scheme across multiple handset manufacturers.

This white paper is not intended to define standard APIs for game developers or ensure binary portability of games. There are several industry initiatives such as Khronos Group (with OpenKODE® and OpenGL® ES), WIPI, Java™ and a host of proprietary solutions such as Microsoft's Direct3D™ Mobile that provide these programmatic interfaces and environments for mobile games. Rather, this white paper provides guidelines for fundamental architectural and platform considerations that significantly impact application performance regardless of the actual set of APIs that are presented to the game developer.

2. References

[OGLES1x]	"OpenGL® ES 1.x for fixed function hardware." URL: http://www.khronos.org/opengles/1_X
[OGLES2x]	"OpenGL ES 2.X and the OpenGL ES Shading Language for programmable hardware." URL: http://www.khronos.org/opengles/2_X
[OPENKODE]	"OpenKODE – Khronos Open Development Environment." URL: http://www.khronos.org/openkode

3. Terminology and Conventions

3.1 Conventions

This is an informative document, which is not intended to provide testable requirements to implementations.

3.2 Definitions

OpenGL® ES	Cross-platform API for full-function 2D and 3D graphics on embedded systems developed by Khronos Group

3.3 Abbreviations

CDMA EV-DO	Code Division Multiple Access, Evolution – Data Optimized
CPU	Central Processing Unit (applications processor)
FPU	Floating Point Unit
GPU	Graphic Processing Unit (2D/3D Hardware)
HSA	High Speed Access (eg High Speed Downlink or Uplink Packet Access)
HW	Hardware
OMA	Open Mobile Alliance
MB	One million bytes (actually 1,048,576)
RAM	Random Access Memory
SW	Software
QCIF	Quarter CIF (Common Intermediate Format)
QVGA	Quarter VGA (Video Graphics Array)
UWB	Ultra Wide Band
PAN	Personal Area Network

4. Introduction

Mobile games have become a key handset application, competing with ringtones as the most popular (and revenue generating) form of online content for mobile handsets. Informa predicts that global revenue from mobile games will increase from US$2.4 billion in 2005 to US$7.2 billion by 2010 (Source: Informa Telecoms & Media, Mobile Games, 2006). While a majority of gaming sales today is attributable to simple, 2D casual games, there is a clear market shift to high-end 3D gaming to create a user experience equivalent to portable gaming consoles and reap the benefits of a higher average unit price per game.

There are several factors influencing this shift, including increasing availability of mobile handsets with hardware-accelerated 2D and 3D graphics, greater presence of high level operating systems that enable provisioning and execution of high quality native games, and increased consumer demand for high end native and managed games across geographic regions. Platform fragmentation, varying levels of security, and lack of deployment infrastructure for premium games are some of the key issues facing game developers and publishers today. Game publishers are finding that game behaviours are highly unpredictable when porting from one platform to another. As a result, game publishers are not able to meet consumer expectations consistently across a range of mobile handsets. Further, behavioural differences among handsets often force fundamental design changes in games (not just handset customizations that can be accounted for at build time).

The time and expense of per-device porting inhibits content delivery and creates an unacceptable time lag from device shipment to content availability. Due to the fragmented market, it is very difficult for game developers to pre-develop games (e.g. before commercial devices are available) and easily deploy these games across a range of devices in a mobile operator's portfolio. As a result, little can be done until a specific device is in the hands of the developer. By the time the port is complete and the game is shipping, a significant fraction of the device's lifetime in the market has passed, and the developer has lost significant profit opportunity.

The Open Mobile Alliance, with members across the mobile value chain, is uniquely positioned to develop market driven specifications that allow all parties to invest with confidence in innovative mobile games and services that offer consumers the most exhilarating experience possible. This white paper is the first step along that path. It establishes a set of guidelines and common terminology for the various facets of the mobile gaming user experience. These guidelines are materialized in a set of performance capabilities, each of which are intended to represent a logical grouping of mobile games that share similar performance requirements and functional capabilities, for example, simple two-dimensional (2D), sprite-based games that comprise the majority of mobile games in the market today, the more advanced three-dimensional (3D) games running in managed environments like Java™ and .NET, and the emerging premium 3D games running within a high level operating system on hardware accelerated devicesTo achieve consistent gaming behaviour across multiple platforms there should be minimum features and capabilities that the game can take advantage of, depending on the type of game under consideration.

It is important to note that the evolution of performance characteristics and certain device features defined in this white paper may advance at different rates. For example, graphics performance may and most likely will increase at a faster pace than available RAM requirements. The main purpose of the game performance descriptions is to provide technology trend guidelines and better visibility to existing and future technology to aid game development. Therefore, it is not intended to define a generic classification system for evaluation of devices across device manufacturers.

Benchmarking based on defined specifications is a recommended approach to measure performance characteristics of the full game across various platforms. It is important to note that the performance characteristics defined in this white paper must be measured holistically. Simple performance metrics such as polygons per second or CPU Mhz are, at best, only loosely correlated with game performance. They do not provide enough information to allow game developers to target specific devices effectively, or to provide a consistent user experience across multiple devices.

As devices introduce more advanced features, game publishers will be able to to take advantage of these features and capabilities to deliver a more desirable gaming experience to the market place and in turn draw new entrants into the mobile gaming arena. Increased revenue opportunities, competition, and a wealth of premium mobile gaming titles will benefit all members of the mobile value chain:

- **Consumers** benefit from a broad library of premium quality games, including mobile versions of leading console titles, and a consistent user experience across handsets.
- **Game Publishers** benefit from higher return on investment (due to reduced porting effort and higher average unit price for premium games).

- **Mobile Operators** benefit from revenue sharing with developers/publishers on higher margin games, consumer attraction and retention, and increased utilization of 3G networks.
- **Operating System Vendors** benefit from higher appeal and more compelling value proposition of their product offering to consumers, and increased attach rate.
- **Handset Manufacturers** benefit from increased phone replacement rate driven by graphics technology evolution that induces consumers to upgrade their handsets to improve their gaming experience, and increased name recognition through mobile game branding opportunities.

5. Mobile Game Performance Characteristics

For the purposes of this white paper, the broad range of mobile games are grouped into the following categories based on performance characteristics:

Casual games include 2D and very simple 3D software-rendered games which make up the majority of mobile games in the market today. Highly optimized, game-specific graphics software integrated into the game executable is normally the best practice for bringing these types of games to the market. Typically casual games have lower memory and display requirements, and they are mainly software rendered.

Rich media games utilize either software or hardware rendering for more advanced three-dimensional (3D) games. Rich media games provide better visual quality by taking advantage of hardware acceleration features, such as bilinear filtering and improved perspective correction. Rich media games have more graphics and multimedia capabilities and use optimized software or hardware-based renderers.

Premium games utilize Floating Point Unit hardware and increased CPU/GPU horsepower to deliver more sophisticated and higher quality graphics as compared to rich media games. Premium games are evolving to include 3D games with cinematic realism, enabled by vertex and pixel shader functionality, similar to the capabilities of higher end gaming consoles today. Advanced native games typically use programmable pixel shaders and require 3D hardware acceleration.The following tables represent a logical grouping of mobile games that share similar performance needs within a broad range of game types from casual to premium games. It is therefore possible that adjacent groups may overlap in some of the mentioned characteristics. The performance characteristics and capability sets in Table 1, Table 2 and Table 3 define key performance characteristics for game graphics performance and features. It is important to note that benchmaking should be a holistic approcach for a full game and the individual characteristics are not necessarily indicative of better game performance. However, data below may serve as guidance for game developers to provide insight on some of the key features and performance characteristics of low to high tier games. It is also desirable that a device which runs a high tier games is capable of running lower tier games.

	Casual Games -------→ Rich Media Games ------→ Premium Games			
Screen resolution	QCIF	QVGA	QVGA	QVGA (VGA desired)
Memory size for games*	2 MB	8 MB	16 MB	32 MB
Graphics API	Optimized SW Renderer	OpenGL® ES 1.x or equivalent / Optimized SW Renderer	OpenGL® ES 1.x or equivalent	OpenGL® ES 2.0 or equivalent
Hardware	CPU only	CPU and optional GPU	CPU, FPU, and GPU	CPU, FPU, and GPU

Table 1: Recommended capabilities for mobile games

- If the device architecture dictates separate heap and video RAM (dedicated to the graphics processor), it is recommended that the device provide equal size memory for both heap and video (twice the RAM shown in Table 1).

The following table describes the minimum graphics performance needed to achieve a consistent gaming user experience across devices. Detailed descriptions of the values show in this table are detailed below.

Row (a) represents the total polygon budget for developers for each mobile game category. This number is the total polygon count prior to rendering. *Note: Average vertices per polygon is assumed to be approximately 1.7. Average fill rate per polygon is roughly 15 pixels.*

Row (b) represents the actual number of polygons displayed to the user after backface and view frustum culling. Backface culling typically results in 50% reduction in polygons. View frustum culling typically results in 20% reduction in polygons.

Row © is the target frame rate that should be supported by devices given the specified polygon count.

Row (d) represents the actual graphics performance that should be achieved with a full game implementation

	Casual Games -------→ Rich Media Games ------→ Premium Games			
(a) Average number of polygons per frame	2,500	5,000	12,500	37,500
(b) Average number of visible polygons per frame	1,000	2,000	5,000	15,000
I Frame rate	20 frames per second	30 frames per second	30 frames per second	30 frames per second
(d) Minimum 3D Graphics Performance (b*c)	20K visible polygons per second	60K visible polygons per second	150K visible polygons per second	450K visible polygons per second

Table 2 Minimum Graphics Performance

The following table provides information on the types of graphics features that are typically used in games for each performance category. These features should be incorporated into benchmarking content used to measure and report the device performance criteria in Table 2.

The graphics features in Table 3 are defined as:

- Lighting: objects within the game reflect one or more light sources
- Texturing: applying one or more textures to objects in the game to give a more realistic appearance
- Vertex Skinning: used in skeletal animation to apply skin to game characters. This method allows smooth animation of complex figures and geometries. Typically only about 50% of the scene involves skinning. The matrix pallet size represents the number of bones in the character.
- Depth complexity: A measure of the number of overlapping surfaces in a scene. For example: an image of a wall has depth complexity of one; an image with a box in front of a wall has depth complexity of two and so on.
- Alpha blending: enabling transparency for parts of an object
- Mipmapping: using optimized bitmaps for texturing to increase rendering speed and reduce artifacts
- Shading: programmable instructions for determining the final pixel coloring during rendering
- Filtering: method used to smooth textures when displaying an object at different sizes. Anisotropic filtering reduces blurriness in objects displayed at an oblique angle.

	Casual Games -------→ Rich Media Games ------→ Premium Games			
Example scene description (representative content)	Classic 2D arcade game or 3rd person perspective (likely to mix 2D and 3D elements)	Sports or racing game or first person shooter	Similar content as other mobile game categories but with increased realism	
Lighting	No lighting	None or One directional light	One directional light	Two lights or more
Texturing	Single textures	Perspective-correct	Mostly single	Extensive use of

	only, mostly not perspective-correct. Texture size is 64x64.	texturing and single textured triangles only (no multitexturing). Texture size is 128x128.	textured triangles. Sparing use of multitexturing in high end games. Texture size is 256x256.	multitexturing. Texture size is 256x256 (minimum) or 512x512 (recommended).
Vertex Skinning	No	No	Yes Matrix palette size is 9 matrices (4x3) and 2 bone influences per vertex Uniform scaling	Yes Matrix palette size is 20 matrices (4x3) and 2 bone influences per vertex Uniform scaling
Depth Complexity	1.5	2	3	3
Alpha blending	No	Not likely	Yes	Yes
Mipmapping	Recommended, but could be limited by available memory	Yes	Yes	Yes
Shading	No	No	Emulated via multipass rendering or advanced SW rendering	Use of vertex and pixel/fragment shaders
Filtering	No	Not for software, bilinear filtering in hardware	Bilinear filtering	Bilinear filtering or anisotropic filtering

Table 3 Graphics Features for Benchmarking

6. Device capabilities and features

This section describes the factors that significantly impact the overall gaming user experience on mobile handsets. Following sections describe how these factors play into the experience of different tiers of mobile games.

6.1 Memory

Memory may be divided into the following categories, where each category may have an impact on the performance of the game application. When considering memory size, it is important to understand the minimum required to download, store, and execute the mobile game. The actual numbers will vary depending upon the gaming tier, and manufacturers are encouraged to exceed the minimum to optimize user experience.

Video	Dedicated video memory for stand-alone GPUs is a traditional mechanism to improve display performance by relieving latency caused by accessing the main system memory. In this architecture, it is recommended that the GPU has a mechanism to fall back to main system memory if all the video memory is consumed.
RAM	Device needs to guarantee that there is a certain amount of available RAM for game execution. Amount of RAM varies as shown in Table 1 above.
Static mass storage	Agame should be installed on to static mass storage, as it then works even if a memory card is removed or changed. The storage size limits the total size of the game.
Mass storage extension	This could be an internal HDD or memory cards such as mini/micro SD. These provide cheap extension to the total mass storage capacity, but offer usually slower access and transfer speed than static mass storage. Installation packages of games can be stored here, or games can be transferred onto the device utilizing mass storage extensions.

The user is expected to be able to store at least one game on the device. The approximate game size varies by the mobile game category. Casual games are estimated to be about 1MB, Rich Media games are estimated to be about 4 to 5 MB, and Premium games are estimated to be about 10-40 MB.

To allow storage of multiple games on the device, the device needs to provide mass storage, such as a removable memory card or a built-in hard drive.

The RAM requirements in Table 1 reflect the amount of memory that needs to be available when a game is launched, for example, after the mobile device closes down all non-essential applications, or has just completed booting up with one or two typical applications running, such as a music player or browser. Consumers should be educated to shut down all non-essential applications on the device when playing a high-end game, as the limited amount of available RAM may otherwise lower the gaming experience considerably.

6.2 Multimedia

Mobile games today typically embed Codecs or uncompressed audio content as part of the games to deliver optimized multimedia playback. Hardware accelerated codecs supported on the device allow game developers to include compressed audio and video clips, such as cut-scenes and background music, using industry standard formats. However, as larger data transfer amounts are time consuming and/or expensive in handheld devices, multimedia is so far mainly limited to music or sound in general.

The following advanced audio features create an immersive gaming experience.

Stereo	Self explanatory

Audio (multichannel) mixing	Device can mix several audio tracks simultaneously (e.g., game background music and sound effects).
3D positioning	Ability for individual voices or the entire audio output to appear to originate from a certain position in a 3D space. 3D positioning technologies are further subdivided according to whether the effect is created when using headphones or speakers
3D spatialisation	Processing of audio output so as to sound as if it is in a particular acoustic environment, including different frequency response, path effects, etc

Devices should support streaming of audio content (eg music) from a network source to enhance the user experience because devices have limited space. This needs to be under user control so that it is only used when the user has a flat-rate data subscription. It is recommended that the devices support mixing of streamed audio data with game sound effects.

Games should be able to mix in multiple types of audio content. Some audio content may be DRM protected but the DRM needs to be handled outside the game by another component on the device. For instance, if a game wants to allow the user to play personal audio on the device during game play then DRM protection of the audio would be handled outside of the game proper.

Games today use between one and five audio channels on average, and implementation is inside the game using software. However, games achieve better performance when devices provide hardware audio mixing capability. Audio mixing for content that is external to the game (eg ringtones or background mixing) should be transparent to the game. Game development SDKs and handset manufacturers should remove the burden of handling audio mixing within the game to improve portability across devices. Devices supporting up to eight channels should satisfy the audio requirements for most games.

Devices need to ensure low latency audio rendering to the output device (eg the speaker, headset, etc) to deliver a good user experience. Audio latency creates user confusion because sounds do not appear to map to actions occurring the game. The time from which the game fills the audio buffer to the time the audio is rendered via the output device corresponds to the application's frame rate. For instance, assuming an application running at 30 frames per second, the audio should be rendered within the frame (within 33 miliseconds).

Typical 3D audio effects such as panning sound based on relative position, tweaking frequency based on relative velocity (for Doppler, and changing volume based on relative distance are all fairly simple to achieve in software. Hardware based 3D audio effects such as positioning and spatialization are nice to have but not widely used today.

6.3 Graphics

The implementation of the graphics engine and software stack significantly impacts overall game performance.

Software Renderer	Software that generates 2D and/or 3D images using the main application processor (CPU), not specialized graphics hardware
Hardware Acceleration	Provides high performance 2D and/or 3D rendering using specialized graphics hardware. Actual graphics performance may be impacted by the following factors CPU/GPU performance Presence of Vector Floating Point Unit Bus speed

Highly optimized SW renderers can achieve good quality 3D effects when HW solution is not available. It is worth mentioning that SW renderers may outperform HW accelerator solutions in some circumstances. Causes of this can include:

- Game content that is designed and optimized for SW rendering, and uses the graphics API in ways that are unfriendly to hardware implementations. This is common in legacy mobile applications and in games or middleware written by programmers who lack experience with hardware accelerated graphics.
- HW accelerators that are poorly integrated with the underlying display system and/or the OS and windowing system. This is common when the system integrator is adding hardware acceleration to a legacy system software architecture.
- Poorly architected HW accelerators and/or drivers. This is unusual.

To date, SW implementations of OpenGL ES have not been widely used by game developers due to lack of desirable performance. Instead, native games typically include custom rendering engines that take advantage of game-specific optimizations, and are therefore faster than even highly optimized “correct” software implementations of OpenGL ES (or comparable APIs).

However, it is highly desirable for game developers to take advantage of HW acceleration solutions when the device supports it, and in fact HW acceleration is required to achieve the high level of performance desired for premium games. Usually it is easier for game developers to utilize hardware acceleration than making a highly optimized software renderer. Hardware acceleration is simply enabled by utilizing a supported level of OpenGL ES (or comparable APIs).

Any new product introductions supporting OpenGL ES should use OpenGL ES 1.1 which is backwards compatible with 1.0. Devices supporting OpenGL ES 2.0 should also support OpenGL ES 1.1 for backwards compatibility because the two versions are not compatible. It is expected that for the near future OpenGL ES 1.1 content will exceed the amount of OpenGL ES 2.0 content,therefore, it is necessary to support both versions. It should also be noted that OpenGL ES 2.0 will never be utilized without hardware acceleration; a software- only implementation would be too inefficient for high-end games.

6.4 Display

Screen resolution directly impacts the gaming experience. Games with high end graphics features require a high resolution display to enhance the gaming experience. On the other hand, higher resolution requires significantly higher rendering performance, more memory, better data transfer and will consume far more power. The platform needs to be able to support rendering 30fps minimum, and the physical size of the display should be large enough to enable benefit of the higher resolution. There is no point wasting rendering and battery power on pixels smaller than the eye can differentiate. This is especially important in video and gaming use cases, as animation hides small pixels more efficiently that static text or still images, and both use cases are very power consuming. Landscape gaming is generally encouraged which has been the format for PC and arcade games.The following are the most marketed resolutions at the time of writing this document. Many other resolutions have been proposed and over time the available resolutions will evolve.

QCIF	Standard display resolution on mass market handsets
QVGA	Minimum resolution needed for mid- to high-tier devices
VGA	Much improved display resolution provides very crisp images for gaming realism, however it greatly increases device processing requirements and increases power consumption due to the large amount of video data
WVGA	Recommended for landscape displays to provide an immersive gaming experience

Software optimized renderers need access to a backbuffer (and an associated pointer to memory to fill the buffer). The backbuffer is a logical memory buffer used for image manipulation before it is copied to the front buffer and displayed.

Software renderers need an efficient color depth conversion mechanism (eg to support conversion from 16bit to 24bit displays). Color depth conversion can have performance degradation if poorly implemented. Some device architectures do this color depth conversion in the DMA channel that transports data from main memory to display to offload the processing from CPU. This optimization is desirable for software renderers because color conversion can consume up to 25% of performance or greater if poorly implemented.

Software renderers also make use of buffering schemes to optimize performance. One mechanism is to use a backbuffer and DMA to move the contents to the front buffer and simultaneously do color conversion. To maintain the target frame rate, the hardware should enable the backbuffer to be filled at twice the frequency of the target frame rate (and as high as possible).

Another mechanism is to render into one of two or more back buffers. When a buffer (i.e. frame) is complete, the display driver is notified to begin displaying from that buffer at the beginning of the next screen refresh cycle. This method is preferred since it eliminates the need to copy the back buffer, saving bus bandwidth, power, and (potentially) CPU time. However, it requires the renderer to write out data in a format that is compatible with the display hardware. It also requires the display refresh memory to be accessible to the renderer, and large enough to store multiple buffers. Thus it may be incompatible with LCD displays that have embedded memory. Care is required in using this method if the game is not full-screen, e.g. if there is a phone menu or status bar that must be displayed at all times. This requirement can be met by displaying either the game or the non-game items on an overlay plane, or by copying the status bar into each of the back buffers. The latter method may use less power if the overlay method makes redundant memory accesses at display refresh rate.

Remote output to external display is highly desirable for game play and a useful debugging technique, which provides the capability for games to be recorded and played back. Minimum resolution for TV output is QVGA, which is the resolution supported by early game consoles. VGA or D1 resolution for TV output is recommended.

6.5 Keyboard / Keypad

As this is the primary means for the user to interact with the gaming application, it is important that the key matrix satisfies certain criteria such as allowing multiple keypad presses.

Specialized game keys	Some devices provide specialized gaming keys in addition to the normal keys available.
Simultaneous key press capability	Many games require simultaneous presses of 2 or more keys and a directional controller.

Specialized game keys are highly encouraged for landscape game play (at least two gaming buttons on both sides of a landscape display). Shoulder buttons on the side of the handset can also enhance the gaming experience on certain types of games (e.g. for use as trigger buttons).

A directional controller such as a joystick or multi-way rocker goes hand-in-hand with the gaming keys. The more flexibility the device provides for the directional controller the better the gaming experience may become.

4-way digital	Generally this is the minimum requirement on any gaming device
8-way digital	Provide a better gaming experience and is recommended over 4-way
Analog	Provides full range motion and gives a similar experience to a gaming console

Most games require at a minimum an 8-way digital rocker with a center switch, which is the minimum functionality for rich gaming experience that is required on any suitable device for games. However there has been excellent game experience offering using only 4-way directional control. The 8-way controller can either be provided by means of 4-way digital controller via software interpolation or natively in hardware. Analog joystick is highly desirable for any mobile game category and provides a richer gaming experience by offering a full range of motion.

6.6 Ergonomics

Certain device UI features can augment overall game play and user experience.

Motion sensor (gyro)	A device that enables you to play games by tilting the terminal up/down/left/right to substitute for direction keys
Vibration	Provides a “rumble” feature to accentuate certain gaming visual effects such as crashes
Camera	Enables visual interaction and directional controller capability

Vibration is an interesting feature but not widely used today. It is important that handset manufacturers provide clear guidelines to game developers if making this available. Improper or extended use of the vibration mechanism could damage the phone or consume power unnecessarily.

The motion sensor feature provides a very good gaming experience for certain games such as racing and sports genres and others that require motion. For instance, in a racing game tilting the device forward slightly can initiate acceleration while tilting the device back will apply the brakes. Left/right motion would steer the car in that direction. This technology is a good alternative to traditional joystick or keypad controls.

The camera on the handset can also be used for analog-like directional control, utilizing motion recognition on the input stream of the view finder. This document does not address any issues of registered intellectual property of that technology or any other mentioned technology. In addition to the camera use cases mentioned above, the camera can also be used to capture real world images and incorporate them into the game.

6.7 Connectivity

Device connectivity allows the user to communicate with other game players and interact with online game servers.

PAN (ex. Bluetooth)	Short-range connection that may be used for multi-player gaming.
WAN (ex. Wi-Fi, 3G, 2.5G)	In addition to multiplayer gaming this feature may provide access to online community portals.

Connectivity is needed for peer to peer and multiplayer gaming. Bluetooth has been challenging for peer to peer gaming. Wi-Fi may be an alternative solution for multipayer gaming, where low-latency connection is the key for good gaming experience. Multiplayer gaming may be achieved by short range personal network (eg Bluetooth) or a wide area network.

Connectivity needed for downloading game content to the device via high bandwidth wide area connection (Wi-Fi, or cellular). For 2.5G handsets, download should be limited to 500K for good user experience. For 3G networks it is acceptable to support up to 2MB downloads. For 3G High Speed Access, UWB, CDMA EV-DO, Wi-Fi networks, larger downloads (eg 10-25 MB+) should be acceptable.

Interest is increasing in providing multiplayer interactive gaming between mobile and PC gamers. This is likely to involve transferring large amounts of data between clients which will require high bandwidth, low-latency connections like Wi-Fi, 3G High Speed Access, UWB, or CDMA EV-DO, among other challenges.

Appendix A. Change History (Informative)

Document Identifier	Date	Sections	Description
OMA-WP-MGPC-20060821-D	21 Aug 2006	n/a	Initial Draft.
OMA-WP-MGPC-20060822-D	22 Aug 2006	All	Updates during Beijing working group meeting
OMA-WP-MGPC-20061018-D	18 Oct 2006	All	Updates during Athens working group meeting
OMA-WP-MGPC-20061214-D	14 Dec 2006	All	Updates during Washington DC working group meeting
OMA-WP-MGPC-20070116-D	16 Jan 2007	All	Updates during San Francisco group meeting
OMA-WP-MGPC-20070418-D	18 Apr 2007	All	Updates during Frankfurt group meeting
OMA-WP-MGPC-20070507-D	07 May 2007	All	Reformatting text
OMA-WP-MGPC-20070604-D	04 Jun 2007	5	Removed comment
OMA-WP-MGPC-20070625-C	25 Jun 2007	All	Approved to candidate by OMA TP TP ref# OMA-TP-2007-0206R01-INP_MGPC_RRP_for_Candidate_Approval
Approved Version: OMA-WP-MGPC-20080610-A	10 Jun 2008	n/a	Status changed to Approved TP# OMA-TP-2008-0193-INP_GS_MGE_V1.0_RRP_for_Final_Approval