Distributed video game streaming system for pervasive gaming

A. Jurgelionis[1], F. Bellotti[2], A. De Gloria[3], P. Eisert[4], J.-P. Laulajainen[5], A. Shani[6]

[1, 2, 3]University of Genoa, Genoa, Italy; [2]Fraunhofer HHI, Berlin, Germany; [5]VTT Technical Research Centre of Finland, Oulu Finland; [4]Exent Technologies Ltd., Tel Aviv, Israel;

E-mail: {[1]jurge, [2]franz, [3]adg}@elios.unige.it, [4]eisert@hhi.fraunhofer.de, [5]jukka-pekka.laulajainen@vtt.fi, [6]ashani@exent.com

ABSTRACT

This paper presents a new distributed video game streaming system. Newly developed streaming protocols and system architectures enable transfer of the game graphics in real-time across the wired / wireless network. Initial tests have shown that the Games@Large system is capable of running video games of different genres, also including First Person Shooter games that are very popular and usually highly interactive and demanding for high end device performance. Further, the experiments prove that a Games@Large server can be hosted on a single PC and support multiple game executions and streaming with a high visual quality to concurrently connected clients via a wireless / wired network. In particular, a Quality of Service solution is exploited, which improves system performance also in the presence of competing traffic.

1 INTRODUCTION

Mobility and digital home entertainment appliances have generated the desire to play games not only in front of a home PC but everywhere inside the house and on the move. Several low-cost consumer electronics end-devices are already available at home. As a result of TV digitalization, set-top boxes (STBs) have entered the home and, as a new trend, mini-laptops are gaining popularity. Although these devices are capable of executing software, modern 3D computer games are too heavy for them. Video games are typically executed on Windows platforms with DirectX API and require high performance CPUs and graphics hardware. For pervasive gaming in various environments like at home, hotels, or internet cafes, it is beneficial to run games also on mobile devices and modest performance CE devices avoiding the necessity of placing a noisy workstation in the living room or costly computers/consoles in each room of a hotel. One of the latest advancements in gaming technology that enables such a pervasive gaming is cloud-based streaming game systems.

This paper presents a new video game streaming system called Games@Large which enables pervasive accessibility of video games from low-end devices in various environments like at home, hotels, or internet cafes. Novel system architectures and protocols are used to transfer the game graphics in real-time across the wired / wireless network to concurrently connected end devices. In general, the system executes games on a server PC, located at a central site or at home, captures the 3D graphic commands, streams them to the end device, and renders the commands on the local screen allowing the full game experience. For end devices that do not offer hardware accelerated graphics rendering, the game output is locally rendered at the server and streamed as H.264/AVC video to the client. Since computer games are highly interactive, extremely low delay has to be achieved for both techniques. This interactivity also requires the game controllers' commands to be captured on the end device, streamed to the server, and injected into the game process.

Simultaneous execution of video games on a central server and two novel streaming approaches of the 3D graphics output to multiple end devices enable the access of games on low cost set top boxes and handheld devices that natively lack the power of executing a game with high-quality graphical output.

The paper is organized as follows: Section 2 describes the G@L system and its main components: 3D, video and audio streaming protocols as well as QoS enabled network architectures. Section 3 presents experimental results for both video and graphics streaming.

2 G@L SYSTEM

The G@L framework [1] depicted in Figure 1 enables cross-platform streaming of video games from a PC to other computers, mobile and consumer electronics (CE) devices in homes and other environments such as hotels, internet cafés and elderly houses. A central PC is utilized to execute multiple games and stream them with a high visual quality to multiple concurrently connected clients via a wireless/wired local area network (LAN) supporting Quality of Service (QoS) [2].

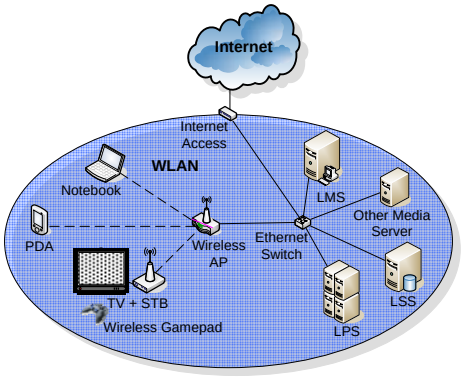

Figure 1: Games@Large framework

The G@L framework consists of several components that we briefly introduce below.

Server Side: A Windows PC with the Local Processing Server (LPS) running games from the Local Storage Server (LSS) and streaming them to clients. The LPS is responsible for launching the game process after client-side invocation as well as managing execution of multiple games. It performs capturing of the game graphic commands for 3D streaming or, alternatively, already rendered frame buffers for video encoding, as well as capturing and encoding of the game audio. Consequently, it performs real-time streaming of the resulting audio-visual data. The LPS is further responsible for receiving the game controller commands from the end device and injecting them into the game process.
Client Side: The client module, running on a Windows/Linux PC or notebook, or WinCE/Linux STB or a Linux-based handheld, receives the 3D graphic command stream and does the local rendering using the available capabilities (OpenGL or DirectX). For the video streaming approach, H.264 decoding must be supported, instead. The client is also responsible for capturing the controller commands (e.g. keyboard or gamepad) and transmitting them to the processing server [3].
3D Streaming: A new graphics streaming protocol is used for streaming 3D commands to end devices allowing lower performance devices such as interactive TV set-top boxes (STBs) to present high performance 3D applications such as games without the need to actually execute the games locally.
Video Streaming: An alternative video streaming approach has been developed, because the 3D graphics streaming approach cannot be used for several mobile devices that lack the hardware capability for accelerated rendering. Synchronization and transmission are realized via UDP-based RTP/RTCP in a standard compliant way. HE-AACv2 [4] is used for audio streaming [2].
Networking: The network connection to the game client may consist of both wired and wireless links.
In the following sections we describe the streaming protocols and networking solutions that are used for the graphics and audio data transmission between the server and clients.

2.1 3D Streaming

Most of the games that run on Windows use DirectX as a lower graphic layer that provides a graphic abstraction layer above the graphic hardware. In Games@Large we intercept those commands and transfer their execution to the client device.
The audio of the game is also intercepted on the server side and streamed to the client side in the same way for both 3D and video streaming. The audio stream and the visualization of the game are synchronized with each other. .
The game runs on the server in a hidden window. This enables other games and applications to use the screen of the server. Moreover, the same machine can be used as a home PC while streaming a game to another device.
In case the game uses a system cursor, the server adds cursor rendering graphic commands to the stream.
The architecture of the 3D streaming implements a pipeline that consists of several layers on both server and client sides as depicted in Figure 2 below. The layers on the server side are executed in the game context. The Direct3D® interception is a proxy implementation of the d3d9.dll which the game uses to render its graphics and can be replaced easily with different proxies according to the graphic library used by the game. The next two layers combine a lossy compression layer that compresses special data like vertex buffers and textures. For the compression of vertex buffers (colors and coordinates), prediction from neighboring and previous vertices is exploited. The Serializer layer writes various structures describing commands and the graphics state to a buffer that is then compressed by a well known lossless compression algorithm in the Compressor layer. The compressed buffer is sent to the network layer that sends it to the client device.

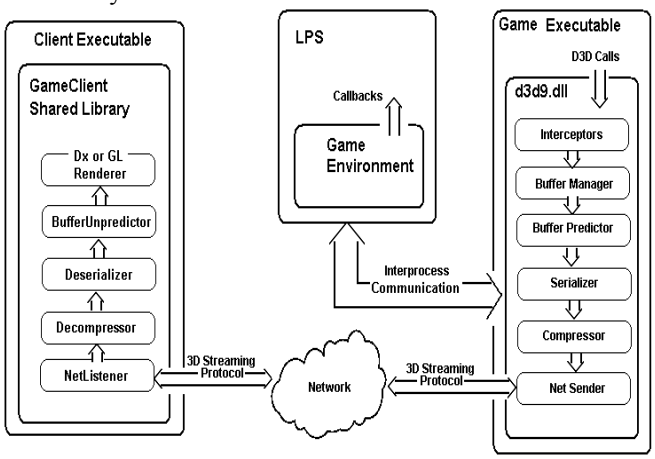

Figure 2: 3D streaming block diagram

On the client side, we have a similar pipeline that reverses the operations of the server pipeline and sends the 3D commands to a rendering layer that uses the graphic library on the end device (DirectX or OpenGL) to render the commands on the local display.
Using the Games@Large 3D streaming, we enable the end device to use a high-resolution display without reducing the quality of the graphics and without the complexity of encoding big frames. The user experience in effected by the time it takes the game to response to user actions. In our architecture, the round trip delay is measured by the time it takes to generate a frame on the server until the next frame is generated after applying the user changes. We use the frame rate as the major measurement of the system. The frame rate is influenced by the bandwidth of the network, the delay of the network and the computation time on the client and the server.

2.2 Video Streaming

Video streaming is selected for devices that do not have a GPU strong enough to handle graphic streaming. This may apply to set-top boxes, small screen devices, or even Laptops/PCs with weak or no graphics card. In video streaming mode, the game's 3D scene is rendered on the LPS server at target resolution (see Figure 3) and the resulting 2D image is read back from the graphic card's frame buffer.

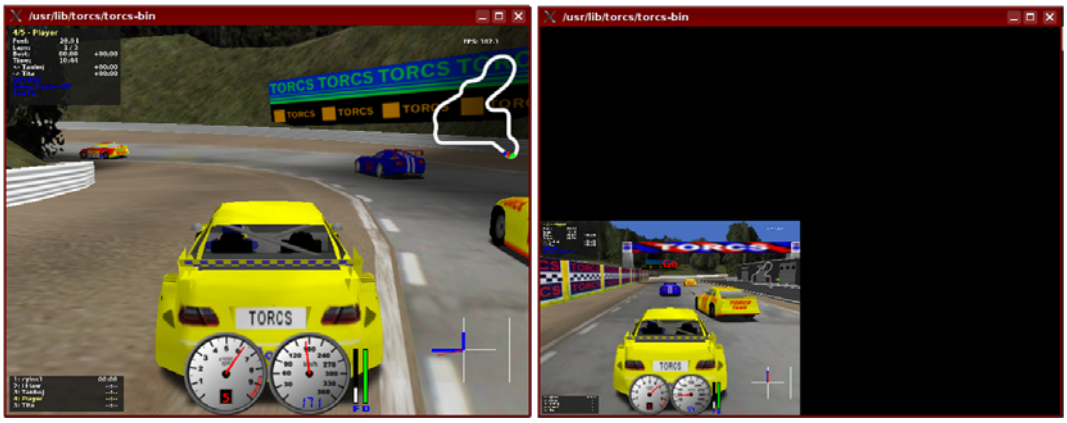

Figure 3: Rendering in resolution adapted to particular end device

The image is then handed to the video streaming module along with additional information from the 3D graphics context, and fed into an optimized H.264/AVC [5] video encoder which exploits the graphics context information to speed up encoding. The resulting H.264 bit stream is finally wrapped into RTP packets and transmitted over the network, Figure 4. Any client implementing the RTP protocol and a standard compliant H.264 decoder can then decode those packets and present the game's output to the user.

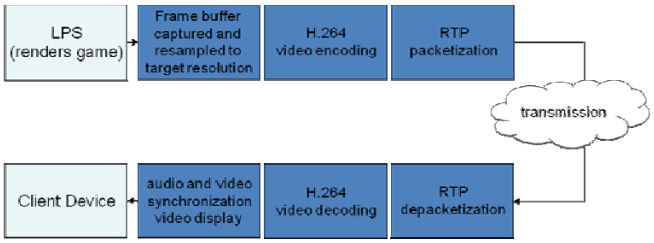

Figure 4: Video streaming block diagram

2.3 Audio streaming

The Games@Large system utilizes the state of the art MPEG Advanced Audio Coding (AAC) [4] audio codec to deliver high quality audio to the user. The game's audio is captured on the LPS and handed to the audio streaming module, which encodes it as AAC compressed audio stream and sends it to the client using the RTP transport protocol, Figure 5. On the client side, any compliant RTP depacketizer and AAC decoder may be used to decode and playback this audio stream.

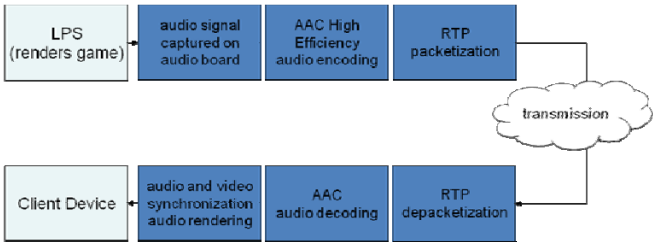

Figure 5: Audio streaming block diagram

2.4 Synchronized real time streaming

Due to the modular architecture of the Games@Large system, the audio, video and graphics streams are processed and transmitted independently. To enable synchronization across those streams, timestamps with a common time base need to be attached to each frame of video, audio or set of graphic commands. This timestamp represents the sampling instant of the media on the LPS. High resolution NTP timestamps, as standardized by RFC 1305, have been proposed for the exchange of timestamp information across the different modules of the Games@Large system.

On the client side, one media stream has to be selected as a master clock. The audio stream provides a continuous and very high resolution timecode, and also accompanies both video and graphics streaming. It is therefore proposed to serve as the master clock that all related media streams may sync onto.

2.5 Networking and QoS

The G@L system requires low-latency and high throughput communication between the LPS and client devices. Network delay should be minimized to maximize the interactivity and controllability of the game play and the network bandwidth should be high enough for good quality gaming.

To ensure smooth game play, decent Quality of Service (QoS) is required from the wireless home network. To fulfil this goal, we use both a MAC-layer mechanism to prioritize some traffic classes and an application-layer mechanism to easily map the application requirements to the MAC-layer QoS. In order to meet the requirement of using low cost components, the choice for the wireless home network has been to use IEEE 802.11 based technologies, because of their dominant position in the market [6].

Priority based QoS can be supported in IEEE WLANs with the Wi-Fi Multimedia (WMM) extensions specified by Wi-Fi Alliance. WMM is a subset of IEEE 802.11e standard and divides the network traffic into four access categories which receive different priorities for the channel access in competition situations. In this way, applications with high

QoS requirements can be supported with better service than others with less strict requirements. In addition to MAC layer QoS support, there is a need for QoS management solutions in a complete QoS solution.

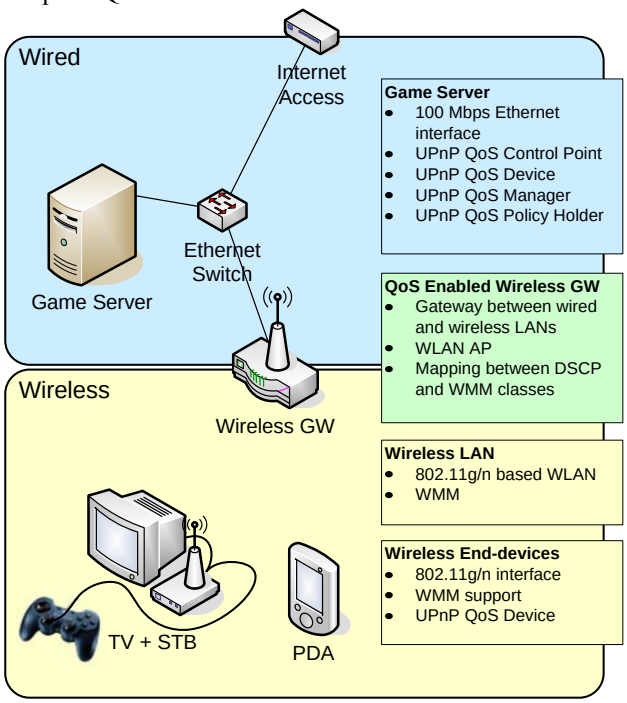

Figure 6: QoS enabled home network for gaming

Our platform relies on UPnP QoS specification. The specification defines services for policy management and network resource allocation. In practice, it acts as a middleware between the applications and the network devices performing the QoS provisioning, Figure 6.

3 INITIAL TEST RESULTS

We have performed a number of different tests in order to evaluate the 3D and video streaming protocol performance, as well as verify how the Games@Large system performs under stress conditions in the LAN. The results of the tests are presented in the following subsections.

3.1 3D streaming results

The statistics below shows the results of some sessions of the game Sprill that was served from a Home PC to a notebook running Windows and another notebook that runs Linux.
Table 1 below shows the statistics of the number of frames per second (independent from resolution) that have been rendered by the client. The average frame rate when running the game natively on Windows client is around 560 frames per second. A Linux client obviously cannot run this game natively.
The frame rate is definitely above "just playable" most of the time, which ensures good gaming experience and game responsiveness.

Table 1: 3D streaming frame rate measurements

	FPS (1/sec)	
	Windows	Linux
AVG	91	42.26
Σ	36	3.72
% of points in range AVG±σ	72.75	83
% of points in range AVG±2σ	95.7	96.7

Table 2 below compares the amounts of raw data representing the original graphics stream and the amount of compressed data that has actually been transferred. The compression ratio is the raw data rate divided by the compressed data rate.

Table 2: 3D streaming data Transfer per Second

	Raw Data (KB/sec)		Compressed Data (KB/sec)		Compression Ratio	
	Windows	Linux	Windows	Linux	Windows	Linux
AVG	5836.8	2944.6	1427.3	776.4	4.27	3.9
σ	351.3	916	123.5	236.13	0.307 *	0.354
% of points in range AVG±σ	84	66.7	91	68.6	90	85
% of points in range AVG±2σ	93.7	95.6	97	94.3	97	98

On average, the compression reduces the data amount by ~75%.The standard deviation 0.307, for Windows client, of the compression ratio shows that 90% of the buffers were compressed with ratio 4.27 ± 0.307.
Multiple games' execution and simultaneous streaming from the G@L server to concurrently connected end devices has been demonstrated in [2].
In [2] and [9] it has been proven that the presence of QoS improves system's performance considerably, as it can be seen from Figure 7 and Figure 8. The downlink performance is visualized in Figure 7 in terms of throughput and delay for the case with equal priority, and in Figure 8 for the case where gaming has higher priority. The effect of introducing the background traffic can be seen very clearly in Figure 7 while it is not visible in Figure 8.

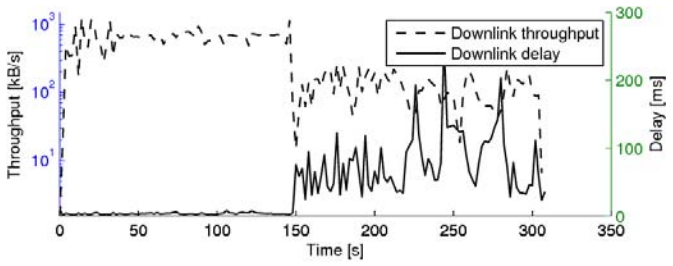

Figure 7: Downlink throughput and delay when both the game and the background traffic share the same priority

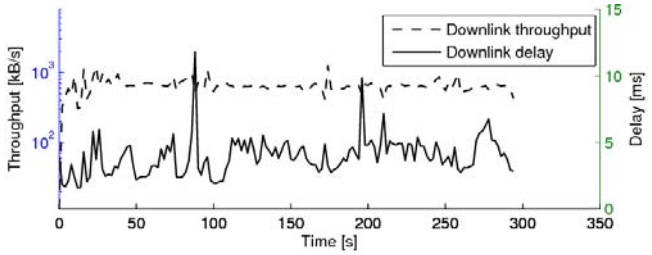

Figure 8: Downlink throughput and delay when the game has higher priority than the background traffic

3.2 Video streaming results

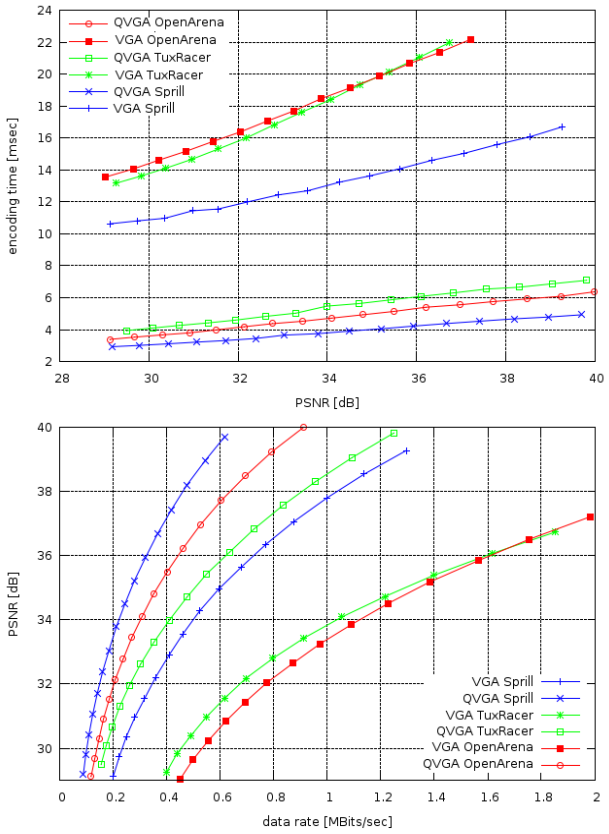

Figure 9: Video streaming performance measurements

In Figure 9 we show the resulting graphs using video streaming for three games, Sprill, Tux Racer and OpenArena, created at two different target resolutions: QVGA (320x240) and VGA (640x480). The measurements were performed on a quad-core 3 GHz computer and represent typical measurements for the evaluation of video streaming performance. The graphs in the two diagrams show the resulting bit-rate for a given resolution and at a certain quality. Also encoding time fat the server is depicted in the upper half of Figure 9. Games with little motion like Sprill result in much better encoding performance than action games, like OpenArena or TuxRacer (also shown in Figure 9).

4 CONCLUSIONS

Lab experiments have shown that the G@L system is capable of running video games of different genres and streaming them with a high visual quality to concurrently connected clients via a wireless / wired network, in particular exploiting a QoS solution, which improves systems performance also in the presence of competing traffic.

In the case of 3D streaming the quality of a gaming experience is typically correlated with the game frame rate, which in the G@L system is proportional to the network throughput. Video games that use high data rates per frame require very high bandwidth and thus, are problematic to be deployed via current networks. This barrier is critical, since we have tested videogame titles that require 80MBit/s for running at 2 frames per second. But several titles require a bandwidth of up to 6 MBit/s for running at over 20 frames per second, which is already manageable with current networks.

For the video streaming, experiments have shown that the encoding time at the server enables the handling of multiple parallel streams on a single server. Further enhancements are expected by exploiting more information on the scene structure obtained from the graphics board. The performance of H.264 allows for satisfying visual quality at bit rates of several hundreds of kilobits.

Another issue for a game to be playable on distributed low-end devices is that it has to be compatible with the device screen size (in particular concerning resolution) and controller capabilities (e.g., some games cannot be controlled with a gamepad or a PDA keypad). Thus, our research argues for new generation mobile devices to have an increase in their Human-Computer Interaction capabilities in order to support more advanced interaction modalities also considering new networked interactive media applications.

Acknowledgements

The work presented in this paper has been developed with the support of the European Integrated Project Games@Large (Contract IST-038453) which is partially funded by the European Commission.

References

[1] Y. Tzruya, A. Shani, F. Bellotti, A. Jurgelionis. Games@Large - a new platform for ubiquitous gaming and multimedia, Proc. BBEurope, Geneva, Switzerland, 11-14 December 2006

[2] A. Jurgelionis, P. Fechteler, P. Eisert, F. Bellotti, H. David, J. P. Laulajainen, R. Carmichael, V. Poulopoulos, A. Laikari, P. Perälä, A. De Gloria, C. Bouras, "Platform for Distributed 3D Gaming," International Journal of Computer Games Technology, vol. 2009, Article ID 231863, 15 pages, 2009. doi:10.1155/2009/231863

[3] I. Nave, H. David, A. Shani, A. Laikari, P. Eisert, P. Fechteler, "Games@Large graphics streaming architecture", ISCE 2008, Algarve, Portugal, April 2008

[4] MPEG-4 HE-AAC, ISO/IEC 14496-3:2005/Amd.2

[5] MPEG-4 AVC (2003), "Advanced video coding for generic audiovisual services", ITU-T Rec. H.264 and ISO/IEC 14496-10 AVC.

[6] J-P. Laulajainen, *Implementing QoS Support in a Wireless Home Network*, IEEE Wireless Communications and Networking Conference (WCNC 2008), Las Vegas, NV, USA, April 2008.

[7] P. Fechteler and P. Eisert, " Depth Map Enhanced Macroblock Partitioning for H.264 Video Coding of Computer Graphics Content", IEEE International Conference on Image Processing (ICIP2009), Cairo, Egypt, November 2009

[8] L. Cheng, A. Bhushan, R. Pajarola, and M. El Zarki, "Realtime 3d graphics streaming using mpeg-4," in Proceedings of the IEEE/ACM Workshop on Broadband Wireless Services and Applications (BroadWise '04), pp. 1–16, San Jose, Calif, USA, July 2004.

[9] A. Laikari, P. Fechteler, P. Eisert, A. Jurgelionis, F. Bellotti, Games@Large Distributed Gaming System, 2009 NEM Summit, 28-30 Sept. 2009