Multi-platform Online Game Design and Architecture

JungHyun Han^{1,*}, Ingu Kang¹, Chungmin Hyun¹,
Jong-Sik Woo², and Young-Ik Eom³

¹ Department of Computer Science and Engineering,
Korea University, Seoul, Korea

² Korea Game Development and Promotion Institute,
Ministry of Culture and Tourism, Seoul, Korea

³ School of Information and Communications Engineering,
Sungkyunkwan University, Suwon, Korea

Abstract. Contemporary Multi-player online games (MOGs) support only a single game platform. Our research goal is to provide an MOG service to ‘users from different platforms.’ This paper presents the architecture of the world-wide first multi-platform online game, which consists of 3D game engines and a multi-platform game server. The experimental results through the prototype implementation show the feasibility of simultaneously supporting multiple game platforms.

1 Introduction

Recently, 3D *multi-player online games* (MOGs) have experienced an amazing success, especially in East Asia. In Korea, for example, a large number of MOGs are currently in service that can accommodate tens or hundreds of thousands simultaneous players, and there is also a long line of MOGs which are scheduled for commercial release. Initially, MOGs were running on only PCs, but now are running on game consoles (e.g., PlayStation2 and Xbox), arcade game machines, or mobile devices (including mobile phones, PlayStation Portable and Nintendo DS). However, such an MOG can currently be serviced for a single platform. Our research goal is to provide an MOG service to ‘users from different platforms,’ e.g. to allow a PC user and a mobile phone user to interact in an MOG.

This paper presents the architecture of the ‘world-wide first’ *multi-platform* online game, supported by the Korean government. The game service environment is illustrated in Fig. 1, which encompasses six types of platforms. The core components of a multi-platform MOG service consists of 3D game engines[1][2] and multi-platform game server. The game platforms are classified into *high-end* and *low-end* machines, depending on their computing capabilities, and we have developed both of the high- and low-end 3D game engines. The game server has also been developed, and an FPS (first-person shooting) MOG has been developed as a prototype implementation of the multi-platform game.

* Corresponding Author.

Fig. 1. Multi-platform online game

Fig. 2. Falcon3D game board

2 Game Design

Three challenges in a multi-platform game design can be described as follows: (1) the hardware performance gap between low-end and high-end platforms, (2) the network latency gap between wired and wireless networks, and (3) the user expectation gap between different platforms.

First, the hardware performances of the low-end and high-end platforms are significantly different. However, we have to also note that the *hardware accelerators* for mobile 3D graphics are coming to market.

Second, the latencies of the wireless and wired networks are also different. For example, a game player through the wireless network may not be able to perform *realtime* actions, which are nicely supported in the wired environment.

Third, game players from different platforms usually have different expectations. For example, a game player on the arcade machine would expect *immersion* or *virtual reality* while mobile users would not. However, it is important to provide the same or similar *look-and-feel* of the game to all game players, regardless of the platforms through which they are connected.

In designing multi-platform game contents, the above-mentioned features should be carefully considered. To tackle the first feature, we designed light-weight mobile 3D game engine that can yet incorporate upcoming mobile 3D acceleration hardware. We have handled the second and third features by assigning different *roles* to wired users and wireless users.

3 Game Engines for Multi-platform Support

Computing capabilities of PCs, arcade machines and game consoles are comparable, especially in hardware 3D acceleration. It is a trend that a high-end

game engine supports PC and game consoles, and a good example is Criterion's Renderware engine[3]. In contrast, our high-endx game engine currently supports PC and arcade hardware, and can incorporate game consoles in the future.

The 4th author of this paper has been in charge of developing a *networked* arcade game board, named Falcon3D[4], in Korea Game Development and Promotion Institute. See Fig. 2. Falcon3D supports both of Direct3D and OpenGL. It also supports TCP/IP for networking, and is compatible with the popular JAMMA (Japanese Amusement Machine Manufacturers' Association) wiring standard for arcade machines. With Falcon3D, we have been able to develop an MOG, which can be played either on PCs or on arcade machines.

Recently, mobile phones are being equipped with 3D acceleration chips[5][6]. As a standard interface for the 3D acceleration chips, OpenGL ES has been proposed[7], and our mobile engine has been built on it. The mobile engine's *high-level* API adopts *scene-graph*[8], which hierarchically organizes geometric models, transformations, textures, light sources, rendering state information, levels of detail (LOD), etc.

4 Multi-platform Game Server

The multi-platform game server's architecture is illustrated in Fig. 3. The front-end system, called the *ubiquitous game framework*, is composed of the *game container* which provides the game service API and the actual game logic is mounted on, and the *gateway server* working as a request-response broker. The back-end system consists of game database, billing server, CRM server, etc.

Together with the game engines discussed in Section 3, the game server enables developing a multi-platform game, which is always accessible if any of PCs, arcade game machines, desktop or mobile game consoles, PDAs and mobile phones is available.

Fig. 3. Multi-platform game server

5 Prototype Implementation and Conclusion

We have designed and implemented an FPS (first-person shooting) game, which currently can be connected through PCs, arcade machines, PDAs, and mobile phones. The user from a PC or an arcade machine plays on the ground in the FPS game world. In contrast, the mobile users can play the game only in a stealth fighter. The mobile users are not supposed to do realtime actions, but do non-realtime actions such as bomb-dropping. However, the PC/arcade-machine users can do all kinds of realtime shooting actions. Fig. 4 and Fig. 5 show the snap-shots of the FPS game.

Fig. 4. Game snap-shots from PC

Fig. 5. Game snap-shots from PDA

Unlike a few existing *turn-based* (not *realtime*) MOGs mostly in the casual board game genre, the authors aim at an MOG, where PC/arcade-machine users are connected through wired Internet and play a realtime FPS game whereas mobile users are connected through wireless network and play special roles which do not require realtime actions. The experimental results show the feasibility of supporting multiple platforms simultaneously. The authors believe that this work is the first of its kind in the field of multi-platform MOGs.

Acknowledgements

This work was sponsored and funded by Korea Game Development & Promotion Institute as Korean government project. (Ministry of Culture and Tourism)

References

1. Bishop, L., Eberly, D., Finch, M., Shantz, M., Whitted, T., "Designing a PC Game Engine," *IEEE CG&A*, Vol. 18, No. 1, Feb. 1998, pp. 46-53.
2. Eberly, D., *3D Game Engine Architecture*, Morgan Kaufmann Publishers, 2005.
3. Renderware, <http://www.renderware.com/>
4. Falcon3D, <http://www.uniondigital.com/>
5. ATI IMAGEON 2300, <http://www.ati.com/kr/products/imageon2300>
6. NVIDIA GeForce 3D 4500, http://www.nvidia.com/page/geforce_3d_4500.html
7. OpenGL ES, <http://www.khronos.org/opengles/>
8. Sowizral, H., Rushforth, K., Deering, M., *The Java 3D API Specification*, Addison Wesley, 1998.