Towards Situation-aware Cross-platform Ubi-Game Development

JungHyun Han[1], Hoh Peter In[1], and Jong-Sik Woo[2]
[1]Department of Computer Science and Engineering
College of Information and Communications
Korea University, Seoul 136-701, Korea
[2]Korea Game Development and Promotion Institute
Ministry of Culture & Tourism, Seoul, Korea
{jhan, hoh_in}@korea.ac.kr

Abstract

This paper presents cross-platform 3D game engines, game server and situation-aware middleware, which are essential components of ubiquitous game environments. These components enable situation-aware cross-platform game development, where a seamless game service can be provided if any of the supported platforms is available.

Index Terms – ubiquitous game, 3D game engine, game server, situation awareness, situation-aware middleware..

1. Introduction

A key point in *ubiquitous computing* is that it should be as easy to interact with computers as with everyday objects in the physical world. The goal of *ubiquitous entertainment* is similar: wherever a user resides, the entertainment contents should be available to the user.

Recently, we have noticed a proliferation of 3D multi-player online games (MOGs), which can accommodate tens or hundreds of thousand simultaneous players. The dominant platform for such MOGs has been PC. However, game consoles (such as Playstation2 and Xbox), arcade game machines and mobile devices (including cell phones and PDAs) can now be networked. Therefore, a user can play an MOG with a PC at home or game parlor, with a motion-based rider at theme parks, and with a cell phone or other wearable computing devices while moving. The goal of the research presented in this paper is to provide a seamless game service while the user moves between various platforms.

The cross-platform multi-player online game environment is illustrated in Fig. 1, which encompasses five types of platforms. The core components of such a ubiquitous game environment include 3D/multimedia clients, cross-platform game/multimedia server, and situation-aware middleware. Section 2 presents the 3D game engines and the cross-platform game server, and Section 3 focuses on the situation-aware middleware.

2. Cross-platform Game Server

The platforms are classified into high-end and low-end, and both of the high-end and low-end 3D game engines have been developed. The high-end PC/arcade-machine/console integrated engine is based on scene-graph, and currently supports OpenGL® and DirectX®. For rendering, an abstract API is built upon the two low-level APIs, and the abstract API can support other low-level APIs of arcade game boards and game consoles. The low-end mobile engine is also based on scene-graph, and currently supports OpenGL® ES, which is the standard for embedded accelerated 3D graphics [1].

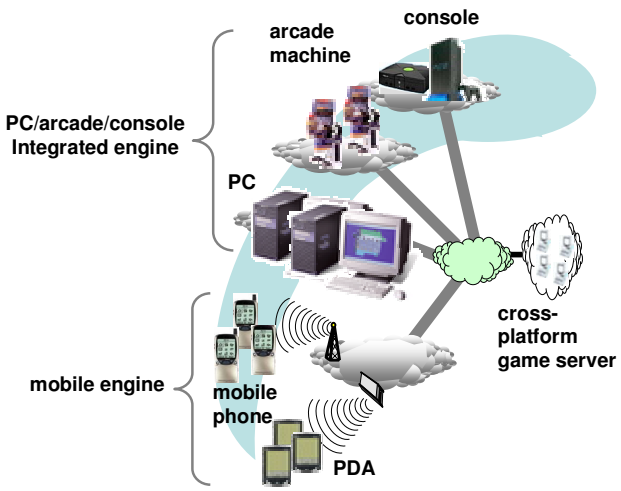

Fig. 1: Cross-platform multi-player online game

Using these game engines, a cross-platform game can be developed that is always present if any of PCs, game consoles, arcade game machines, PDAs and cell phones is available.

The cross-platform game server consists of the front-end system, which we call the *ubiquitous game framework*, and the back-end system, which is composed of game database, billing server, CRM server, etc. The main components of the ubiquitous game framework are the game container which provides the game service API and the actual game logic is mounted on, the gateway server which works as a request-response broker, and the smart distributed server which dynamically allocates and deallocates resources based on a virtual network.

For the cross-platform contents development purpose, two factors should be seriously considered. First, the hardware performances of the low-end and high-end platforms are (and will be) significantly different even though hardware accelerators for mobile 3D graphics are coming to market. Second, the bandwidths of the wireless and wired networks are also significantly different. Game designers should consider these differences, and for example, different roles can be assigned to PC users and mobile users in an MOG.

Proceedings of the 11th Asia-Pacific Software Engineering Conference (APSEC'04)
1530-1362/04 $ 20.00 IEEE

3. Situation-Aware Middleware: An Approach

A conceptual architecture of the situation-aware middleware based on Reconfigurable Context-Sensitive Middleware (RCSM) is proposed in [2]. Ubiquitous applications require use of various contexts to adaptively communicate with each other across multiple network environments, such as mobile ad hoc networks, Internet, and mobile phone networks. However, existing context-aware techniques often become inadequate in these applications where combinations of multiple contexts and user actions need to be analyzed over a period of time. Situation-awareness is considered as an essential property to overcome this limitation, and situation-aware applications can respond to both current and historical relationships of specific contexts and user actions, in addition to being context-sensitive [3].

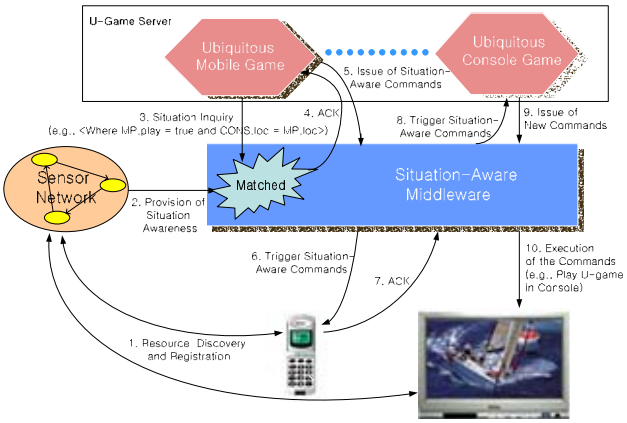

Fig. 2: Operations with the situation-aware middleware

This paper proposes to innovatively apply the situation-aware middleware technologies to the cross-platform game design. The overall concept of operations is shown in Fig. 2. A game-playing scenario for a cross-platform online game running on both mobile phones and game consoles can be described as follows.

Both of the mobile phones and game consoles are networked to a single server, and the corresponding modules in the server are 'Ubiquitous Mobile Game' and 'Ubiquitous Console Game,' as shown in Fig. 2. Suppose that a ubiquitous mobile game player enters home and wants to continue the game on a game console which has a better display device, TV monitor. The Sensor Network discovers the mobile phone and registers the related information of the mobile phone (Step 1 in Fig. 2). The Sensor Network then informs the Situation-Aware Middleware that the mobile phone enters home (Step 2). The game server inquires the middleware whether the console and the mobile phone are at a single location (Step 3), and it is acknowledged by the middleware (Step 4). Then, the game server issues situation-aware commands to the mobile phone through the middleware (Steps 5 and 6). An example of the commands is inquiring the user whether to move to the game console. When the mobile phone user agrees on moving to the console (Step 7), the middleware triggers the game server (Step 8) and finally the cross-platform game is now playing on the game console (Steps 9,10).

For implementing the above scenario, Situation-Aware Interface Definition Language (SA-IDL) has been defined for communication between the game server and the middleware. The middleware interprets the SA-IDL and generates executable codes for various target platforms. In this way, the details of the target platforms are hidden from the game server developers, and the developers need to just describe the situations and related actions.

Situation-Aware Manager (SAM) has also been developed based on IBM TSpaces [5] to monitor raw context-aware data from the sensor network and interpret them for situation-awareness. SAM in the middleware notifies the game server that the interpreted situation matches the pre-defined one (Steps 2-4 in Fig. 2).

4. Conclusion

Ubiquitous game has recently emerged and explored the merger of ubiquitous computing and computer game. A recent report on the ubiquitous game can be found in [2]. However, the previous works have focused on various sensing technologies, for example, capturing physical interaction, the blurring of physical and virtual worlds, etc. Unlike the previous works, this paper has presented the first attempt to develop core components of situation-aware cross-platform multi-player online games such that an adaptive, seamless service can be provided when a game player moves between platforms. Key contributions are introduction of situation-awareness in the game domain and a new middleware design for the cross-platform game engine. Beside the proposed techniques, the others need to be developed such as user intention, task and resource collaboration models, and open, yet standard communication protocols, and adaptive multi-QoS support.

5. Acknowledgment

This work is supported by the ubiquitous computing and network infrastructure development project of the 21st Century Frontier Research Program. Hoh Peter In is the corresponding author.

6. References

[1] OpenGL ES, http://www.khronos.org/opengles/
[2] S. Yau, F. Karim, Y. Wang, B. Wang, and S. Gupta, "Reconfigurable Context-Sensitive Middleware for Pervasive Computing," IEEE Pervasive Computing, Vol. 1, No. 3, July-September 2002, pp. 33-40.
[3] Hoh P. In, C. Kim, and S. Yau, "Q-MAR: An Adaptive QoS Management Model for Situation-Aware Middleware," EUC 2004, LNCS 3207, pp. 972-981, 2004.
[4] Staffan Björk, Jussi Holopainen, Peter Ljungstrand, Regan Mandryk, Special Issue on Ubiquitous Games, Personal and Ubiquitous Computing, Vol. 6, 2002.
[5] TSpaces, http://www.almaden.ibm.com/cs/TSpaces