Implementation and Evaluation of Front-End Protocol of Network Game Infrastructure

LATE Abran Kra Florence† Shinji Kono☆,††

We have proposed a Network Game Infrastructure using Agent system. This system has two kinds of network protocols, front-end protocol and inter-agent protocol. We describe an implementation of the front-end protocol which connects user game machine such as PlayStation 2 or Gameboy advance and an agent in the closet agent in the Internet. The front-end protocol contains Agent Packet, Game List and Object Table.

In order to evaluate the performance of the front-end protocols, the turnaround time between Agent and Playstion 2, the number of transmission of playstation 2 for agent packets, the number of reception of playstation 2 for agent packets should be measured.

1. Introduction

Agent systems are emerging as a potential solution to the problem of constructing flexible network game software such as Network Game Infrastructure. A characteristic of such systems is that whole system behaviour patterns emerge from the combination of many details in many agents, in sometimes intricate ways. The Network Game Infrastructure has two kinds of network protocols, Front-End Protocol and Inter-agent Protocol.

This paper introduces a Front-End Protocol which connnects user game machine such as Playstation 2 or Gameboy advance and an agent in the closet agent in the internet. The Front-End Protocol contains Agent Packet, Game List and Object Table.

In the following section, we first describe the Front-End Protocol, present the Target Application, describe the outline of PlayStation 2, and present the Simulation results. We conclude in the last section.

† Interdisciplinary Information Engineering, Graduate School of Engineering and Science, University of the Ryukyus.
☆ 琉球大学工学部情報工学科
Information Engineering, University of the Ryukyus.
†† 科学技術振興事業団さきがけ研究 21(機能と構成)
PRESTO, Japan Science and Technology Corporation.

2. Front-End Protocol

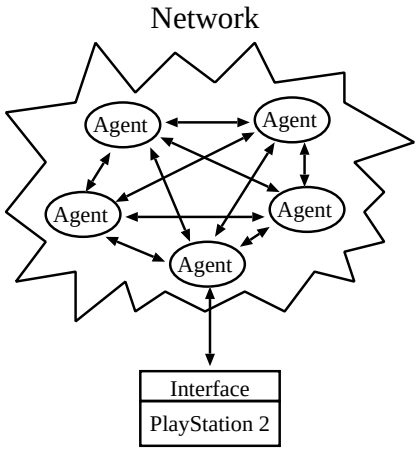

図 1 Communication system using Agent

Front-End Protocol is an interface between the Agent system and the game machine. It contains Agent Packet, Game List and Object Table.

2.1 Agent Packet

Agent Packet is for communication in agent system. It is consists of:

- Packet Header which contains basic informa-

tion of the packet such as

```
total_size //size of the packet
ID         //id from the sender
             of packet
game_id    //game ID
```

These items of information, are used with the basic section of the system such as sending and receiving of the packet and the routing of Network Manager.

- Packet Tag attaches semantically in Object Data. It contains:

```
data_size  //size of object
object_type //type of object
object_id   //ID of object
offset     //offset value regarding
             object
command   //appointment of processing
            for this packet
```

- Object Data is the data of object such as 3D object and texture. Object Data must be the character string of the ASCII cord/code. Encode to the text data and decode routine must be prepared from the binary data. You can set the Agent Packet to the way of the data format which is originally the text format of XML.

2.2 Game List

Game List is used for the information interchange of agent and the game presentation. Below is an example of Game List from the shooting game:

```
shooting ./game_data/shooting shooting-ver3
dummy1 ./game_data/dummy1 dummy1
dummy2 ./game_data/dummy2 dummy2-debug
```

2.3 Object Table

Object Table manages the object in agent system. It contains:

```
id           //object table ID
object_type  //type of object which is
               registered to this table
attribute   //variable for various
              attribute flags
void *object //pointer to actual data
               of object
```

3. Target Application

For our research, we chosed shooting game as a model for the implementation and evaluation of this agent system. The game was implemented in 3D space and 2D space. The Torus network was adopted as a network topology. A torus network is a grid network with boundary connections as shown in Fig.2. The boundary connections decrease internodal distances and eliminate edge effects. The regularity of torus networks enables development of simple routing schemes.

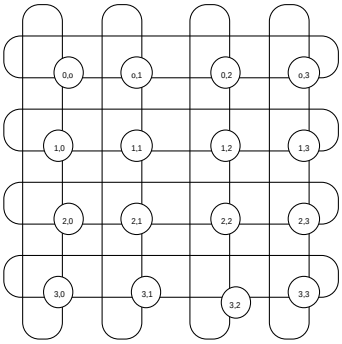

図 2 Torus Network

3.1 3D object and 2D image data

2D image data must be the character string of the ASCII cord/code. Encode to the text data and decode routine must be prepared from the binary data.

We use LightWave:3D modeling tool to model the 3D object. The Light Wave file(.lwo) can not be use on Playstation 2, we need to convert this file to the convernient file by using a converting script which was developed at this laboratory. The data of Light Wave format are converted with the Perl script.

3.2 Compilation method of 3D object

Considering the expandability of data representation and the fact that it is utilized with various architecture, The XML format is used to express the 3D object data.
In the following is an example of 3D object which is expressed with XML.

```
<?xml version="1.0" encoding="EUC"?>
<OBJECT-3D name="box_tex">
    <surface name="Default" size="36"
                  prim="Triangle">
        <coordinate>
         1.45000004768372, 1.10000002384186,
                        -1.29999995231628
         -1.47500002384186, 1.10000002384186,
```

```
                    -1.29999995231628
 -1.47500002384186, -1.10000002384186,
                    -1.29999995231628
              < Abbreviation >
        </coordinate>
        <normal>
            -0, 0, 1
            -0, 0, 1
            -0, 0, 1
              < Abbreviation >
        </normal>
        <color>
            200.0, 200.0, 200.0, 127.0
            200.0, 200.0, 200.0, 127.0
            200.0, 200.0, 200.0, 127.0
              < abbreviation >
        </color>
        <texture name="CowMask.tga">
            1, 0
            0, 0
            0, 1
              < Abbreviation >
        </texture>
    </surface>
</OBJECT-3D>
```

3D object is formed from polygon, polygon is made from plural apexes. Information of each apex, is the most fundamental information of 3D object. The basic data which is necessary in order to express apex is, coordinate and normal vector, color and texture coordinate.
Information of these apexes is collected to every surface. The various special effects which are administered to the surface of 3D object are done at this surface unit. Depending, it made even with XML to collect at the surface unit.

3.3 Sketch of the application

Portal Manager does the reception processing of Agent which sends connection request newly. There are two types in connection request.

- From Agent to Agent, participation demand for Agent network
- From the game machine to Agent, the connection demands to Agent in order to begin the game.

In each case, connection is done with the procedure shows below.

(1) Agent: receive a request for certification
(2) Agent: when certification succeed, Agent gives the game list
(3) Requester: choose the game which it would like to participate from game list, and return the Id to agent.
(4) Rquester: from the Id which is return, agent sends the data of 3D object, the texture and all data necessary for that game to requester.
(5) Requester: after receiving all game data, game start is declared to agent.

reference Fig.3.

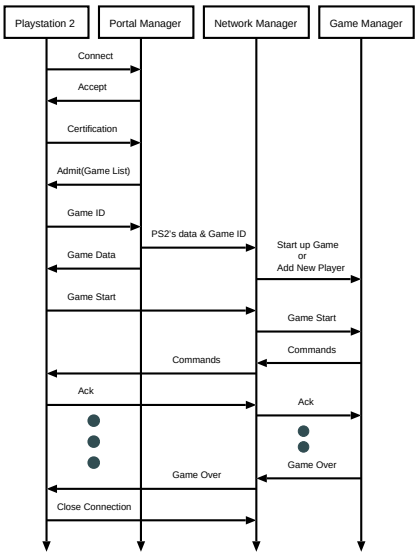

図 3 Entire flow of Agent system

Portal Manager does the reception processing of Agent, describes the socket and delivres the information of requester to Network Manager. After that leaves proccessing.
The packet sent from requester is received by Network Manager.
Network Manager manages the network topology of each game, does the routing of packet communication.
Game Manager is the core of game. It actualizes the game. The physical simulation of 3D object is done in Game Manager. reference Fig. 4.

4. PlayStation 2 System Overview

Here we explain the structure of the playstation 2 : the Emotion Engine (EE). We have to look at the Emotion Engine in the context of the overall design of the PS2 because, the PS2 is in a slightly

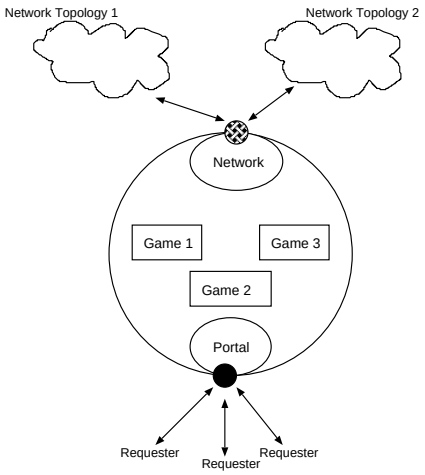

図 4 Entire image of Agent

different situation. The PS2's designers designed hardware whose main purpose is to run one type of application extremely well: the 3D game. Sure, the PS2 can run web browsers, mail clients, and other types of software, but that's all secondary. The main thing the PS2 does is 3D gaming, which means generating the kind of immersive sound and vision that places you in a virtual world. Nearly all of the PS2's hardware is dedicated to providing some specific portion of that audiovisual gaming experience.

Let's take a look at the main parts of the PS2.

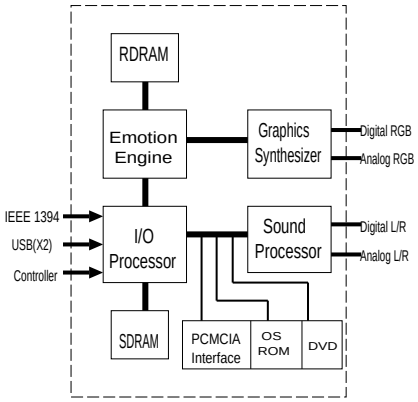

図 5 Constitution of PlayStation 2

In the above picture, you can see that there are four main parts to the device. Let's look at the one at a time. The I/O Processor (IOP) handles all USB, FireWire, and game controller traffic. When you're playing a game on the PS2, the IOP takes your controller input and sends it to the Emotion Engine so that the Emotion Engine can update the state of the game world appropriately. The Emotion Engine is the heart of the PS2, and the part that really makes it unique. The Emotion Engine handles two primary types of calculations and one secondary type:

- Geometry calculations: transforms, translations, etc.
- Behavior/World simulation: enemy AI, calculating the friction between two objects, calculating the height of a wave on a pond, etc.
- Misc. functions: program control, housekeeping, etc.

The Emotion Engine's job is to produce display lists (sequences of rendering commands) to send to the Graphics Synthesizer. The Graphics Synthesizer is sort of a souped-up video accelerator. It does all the standard video acceleration functions, and its job is to render the display lists that the EE sends it. Finally, the Sound Processor is the "soundcard" of the PS2. It lets you do 3D digital sound using AC-3 and DTS.

5. Simulation results

In this section, we first evaluate the performance of the front-end protocols, and then compare it with Linda which is a communication protocol using Centralize Server.

5.1 Evaluation of Front-End Protocol

The Agent was simulated with the communication protocol of Agent.

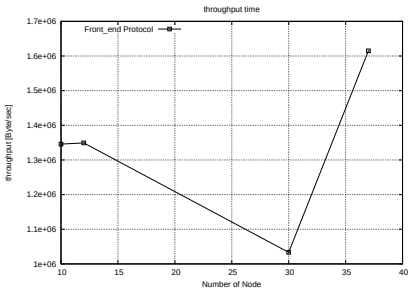

図 6 Throughput time between PlayStaion 2 and Agent packets

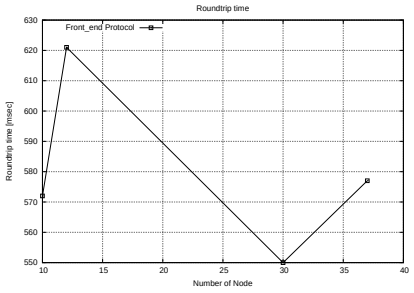

図 7 Response time between PlayStation 2 and Agent packets

5.2 Comparison with centralize server

Linda server was used as a general Centralize Server. Linda is the system which communicates due to the fact that each client reads and writes those where ID and Data which are called the tupple in the server have become set. The access to the data which is stored in the server is done by appointing ID of the tupple. Figure 8 bellow shows the response time between playstation 2 and Linda server.

By comparing the round-trip time of agent system with the round-trip time in Linda, we see that the response time from the agent system is definitely slow.

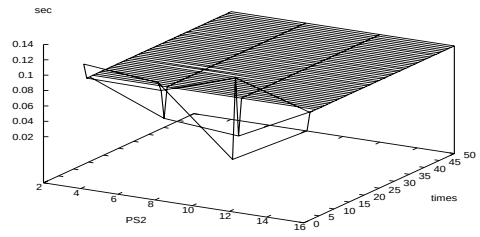

図 8 Round-trip time between Linda PlayStation 2

6. Related works

The techniques for massivelly online game server have been matured from 90's. There are some commercial scalable and seamless massivelly online game middlewares. Some of them adopt agent system, but it is limited on separated and subsidiary traffics for voice chatting or multimedia streaming.

There are several more recent approches that apply agent system to massivelly online game. But there still exists probality of catastrophic failures in, and neither of them argues on the network address translation and restrict wider accesses to a massivelly online game.

7. conclusion

In this paper, we presented a front-end agent protocol for network game. Unlike traditional client-server, our network game infrastructure utilizes high power client machines as local game servers enabling users game machine to play game through agent communication protocol, and thus reduces significantly the outgoing traffic from a central game server. Since outgoing traffic is much larger than incomming traffic for massivelly online game publishers, our scheme will help them to design games in larger scale.

References

[1] UNIX network programming, ohmic corporation, ISBN 4-274-07778-0

[2] Sean Walton, Linux socket programming, Pearson Education and ISBN 4-89471-467-1

[3] SHINJI Kono, Sadoyama Akira, Japanese software scientific meeting, 19th conference dissertation collection September, Proposition of network game framework on PlayStation 2Linux, 2002

[4] SHINJI Kono, Proposition of parallel game object system of PS2 direction, SwoPP 2001, July, 2001

[5] Sadoyama Akira, SHINJI Kono, Autonomous construction of large scale network game infrastructure, Master thesis, 2003.

[6] Jon "Hanibal", A Technical Overview of the Emotion Engine, http://arstechnica.com /reviews/1q00/playstation2/ee-1.html

[7] Kyoung chul KIM,Ikjun YEOM, HYMS: A Hybrid MMOG Server Architecture