2002년도 대한전기학회 하계학술대회 논문집 2002.7.10-12

텔넷 기반 멀티 스레디드 게임 서버의 구현

김연정* 정옥란 조동섭
이화여자대학교 컴퓨터학과

Design and Implement of the Multi-threaded Game Server using Telnet

Yeun-Jung Kim*, Chung Ok-Ran, Dog-Sub Cho
Dept. of Computer Science and Engineering, Ewha Womans University

Abstract - 초고속 인터넷과 고성능 컴퓨터의 보급은 게임이 발전할 수 있는 환경을 만들어 주었다. 게임은 단순한 오락의 수준을 아닌 21세기 핵심 산업으로 꼽힐 정도로 많은 관심을 받으며 다양하게 개발되고 있다. 그러나 많은 관심은 많은 사용자를 만들었고 이 많은 사용자는 서버 접속률과 진행속도를 떨어뜨리는 제한 요소가 되고 있다. 이러한 다수의 사용자를 효과적으로 관리하고 게임을 진행시키기 위해서는 각 게임에 맞는 Server의 구현을 필요로 한다. 이 논문에서는 다양한 게임을 구현하는데 필요한 서버의 요건들을 네트워크 환경중심으로 알아보고 실제로 하나의 게임 서버를 디자인하여 구현하여 보았다.

1. 서 론

컴퓨터 기술의 발달은 다양한 게임을 개발할 수 있는 여건을 만들어 주었다. 과거의 게임은 주로 사람과 컴퓨터가 대결을 이루었다. 컴퓨터와 사람 사이의 게임에서 있어서 가장 큰 문제점은 컴퓨터가 사람만큼 고도의 인식과 판단을 하지 못한다는 것이다. 이러한 게임의 다양성과 지능성에서 한계를 가지고 있다. 이런 단점을 극복하기 위해 사람과 사람의 대결이 가능하게 하는 네트워크를 이용한 다양한 온라인 게임들이 등장하기 시작했다.

네트워크/온라인 게임에서 네트워킹의 가장 중요한 역할은 게임 플레어를 동기화이다. 즉 서로 다른 컴퓨터끼리의 멀티플레이에서 가장 중요한 목적은 진행되고 있는 게임 플레이 상태를 각 컴퓨터에서 동일하게 나타나도록 하는 것이다. 이를 정상적으로 수행 되어야만 정상적인 게임플레이를 할 수 있다.

게임 개발자는 게임플레이를 동기화 하는 방식에 대해 정확히 디자인해야 한다. 그것은 현재 프로그램 구조와 네트워크 환경을 모두 검토하고 결정해야 한다.

본 본문은 두 명의 게임머와 그들의 플레이를 바라보며 누가 이길 것인지 배팅을 하며 게임을 풀어가는 게임머로 이루어진 형깃의 퀴즈 게임서버를 구현하면서 게임 플레이의 동기하의 문제를 다루어 보도록 하였다. 우선 2장과 3장에서는 게임 서버 모델과 게임에 사용되는 프로토콜에 대하여 알아보고 구현된 게임에 가장 적합한 네트워크 환경을 선택한다. 4장에서는 구현된 서버의 구조를 설명하고 5장에서는 실제 구현 결과 및 평가에 대해 알아본다. 마지막 6장에서는 앞으로의 보완점에 대해 기술하면서 결론을 맺는다.

2. Game Server model

게임 서버는 게임의 성능을 좌우하는 매우 중요한 부분이며 서버의 모델을 어떤 것으로 선택하는가도 성능향상에 많은 영향을 미친다. 많은 게임 서버의 모델이 제한되고 있다. 대표적이면서 게임에 사용되는 서버 모델에는 클라이언트-서버모델과 피어-투-피어 모델 그리고 분산 서버 모델이 있다.

2.1 Client-server model

게임공간을 공유하는 컴퓨터중 하나가 게임 서버 역할을 하고, 나머지 컴퓨터들이 게임 클라이언트 역할을 하는 네트워크 구조이다. 게임 클라이언트는 게임 서버와 네트워크연결을 하고 게임 서버는 모든 게임클라이언트와 일대일 연결되어있다. 게임 클라이언트는 게임 플레이를 사용자와 직접 상호작용 한다. 한편 게임서버는 연결되어 있는 다른 게임 클라이언트 간에 중계역할을 한다. 클라이언트-서버를 사용할 경우 모든 클라이언트는 게이머의 입력을 서버에게 보내며 서버는 이러한 입력을 기반으로 게임의 모든 진행 결과를 전체 클라이언트에게 브로드캐스팅 한다.

메세지수는 클라이언트의수에 비례하기 때문에 비교적 많은 수의 클라이언트를 수용할 수 있다. 또한 클라이언트 서버 모델은 여러 플레이어 사이의 동기화의 문제를 단순화 시키므로 멀티 플레이어 게임에 있어서 많은 이점을 가지고 있다.

2.2 Pear to Pear model

주로 실시간 전략 게임이나 액션 게임에서 상대의 컴퓨터가 네트워크 상 어디에 위치하는 지 알아내고 다른 컴퓨터를 거치지 않고 메시지를 주고받을 때 사용된다.

피어 투 피어 모델에서는 클라이언트/서버와 달리 게임공간을 공유하는 모든 컴퓨터는 동등한 역할을 한다. 다시 말하면 클라이언트-서버 모델에서는 게임서버와 게임 클라이언트가 구별되지만 피어 투 피어 모델에서는 모든 컴퓨터가 클라이언트와 서버의 역할을 동시에 한다. 그리고 각 컴퓨터는 게임공간을 고유하는 다른 모든 컴퓨터와 그물처럼 연결된다.

이런 모델에서는 게임 내에서 발생하는 메시지가 다른 모든 컴퓨터에 전송된다. 통신 양은 클라이언트의 수의 증가에 따라 기하급수적으로 증가한다. 그러므로 보통은 네 개 이하인 경우 사용하고 그 이상일 경우는 통신량이 폭발적 증가하므로 사용이 제한된다.

그러나 이 모델은 중간의 경유 없이 메시지를 다른 컴퓨터에 직접 전송하므로 메시지 송신에 걸리는 시간이 짧고 하나의 컴퓨터가 종료되어도 다른 컴퓨터들의 게임 진행은 그대로 유지된다.

2.3 분산 server model

클라이언트-서버 모델의 경우는 많은 수의 클라이언트가 참여할 수 있으나 하나의 게임에 수백내지는 수천명이 참여하는 경우 하나의 서버로는 이를 감당하기는 어려워 작업량을 여러 서버에 나누는 방식이다. 분산 서버 방식에는 부하분산 방식과 맵 서버 방식이 있다. 그러나 부하 분산은 접속 기능만을 여러 대의 서버가 나누어하

본 연구는 2002년도 두뇌한국 21산업에 의하여 지원되었음.

고 게임진행은 메인 서버가 수행가기 때문에 클라이언트의 수가 증가하면 같은 한계를 가지게 된다. 맵서버 방식은 게임 월드를 여러 개의 맵으로 나누고 각 맵을 담당하는 서버가 따로 존재하여 게이머는 자신이 속한 맵을 담당하는 서버에 접속하게 하는 방법이다. 하지만 맵의 경계를 넘게 되면 해당되는 캐릭터를 해당 서버로 보내주어야 하기 때문에 게임 진행이 지연 될 수 있다.

3. 대표적인 게임 프로토콜

프로토콜이란 네트워크 상에서 연결된 컴퓨터끼리 데이터를 주고받을 수 있도록 미리 약속된 전송 규약을 말한다. 다시 말해 컴퓨터 끼리 의사소통을 하기 위한 '언어'라고 할 수 있다.

요즘은 컴퓨터 네트워크의 규모가 증가되고 네트워크를 이용한 정보 전송 수요가 다양화 되고 있다 또한 소프트웨어와 하드웨어 장치가 계속 증가되는 최근의 환경에서 효율적인 정보 전달을 하기 위해서는 프로토콜의 기능이 분화되고 복잡해질 수밖에 없다. 일반적으로 많이 사용되는 프로토콜은 TCP/UDP, IPX와 NetBEUI등 있다.

3.1 TCP/UDP

대표적으로 사용되는 소켓 API에서 지원하는 프로토콜은 TCP라는 인터넷 프로토콜이다. TCP는 주로 받는 메시지에 오류가 있을 가능성은 없지만 송수신 지연시간이 생길 수 있다. UDP는 TCP에 의해 송수신 시간은 빠르나 메시지를 못 받거나 데이터 순서가 바뀌어 갈 수 있다.

주로 TCP는 온라인 게임에 많이 사용되고 UDP는 네트워크 게임에서 많이 사용된다.

TCP/IP 프로토콜을 기반으로 하는 게임은 서버 의존도가 크므로 서버의 성능이 전체게임의 수행속도에 지대한 영향을 미치게 된다. 또한 원거리의 사용자간에 게임이 이루어지므로 인터넷에서의 네트워크 속도와 같은 여러 가지 네트워크의 문제점에 대해서도 고려가 필요하다.

아래에 설명할 IPX가 몇 명의 사용자가 같은 게임을 할 수 있는 반면 TCP/IP를 기반으로 했을 경우에는 수백 명에서 수천 명까지 많은 사용자를 함께 한 게임을 할 수 있다.

3.2 IPX

IPX는 데이터그램을 통해 메시지를 전달하는 노벨 네트에어의 프로토콜을 말한다. IPX 경로 제어를 사용하고 있는 라우터를 노벨 Netware 클라이언트와 서버가 통신할 수 있도록 하기 위해 LAN 내부에 접속 된다. IPX는 데이터그램 또는 패킷 프로토콜이다. IPX는 통신프로토콜의 네트워크 계층에서 동작하며, 패킷교환 중에 커넥션이 유지될 필요가 없는 커넥션리스 프로토콜이다. 팻킷 수신 통보는 노벨의 또 다른 프로토콜인 SPX(Sequenced Packet Exchange)에 의해 관리된다.

IPX를 기반으로 하는 온라인 게임의 경우 대부분이 최대 10여명 정도의 사용자가 근거리 통신망을 기반으로 즐기는 경우가 대부분이다. 또한 이러한 게임들의 경우에 특별한 서버가 없이 근거리 통신망에서 사용될 사용자들이 자신들끼리 특정 사용자의컴퓨터를 서버로 지정하고 이 안에서 게임이 진행되며 서로간이 동기를 맞추게 설계되어 있다.

3.3 NetBEUI

NetBIOS를 전통적으로 가장 잘 구현한 프로토콜은 NetBEUI 프로토콜이다. 이것은 Lan Manager 서버를 사용할 때 바른 내부 네트워크를 운용하기 위해 만들어졌으며 주로 50대 미만의 컴퓨터에서 사용된다. NetBEUI 프로토콜은 크기가 매우 작아 브로드캐스트 용으로 사용하기 매우 좋지만 컴퓨터가 많아질수록 효율이 떨어지는 단점을 안고 있다.

4. 멀티 스레디드 게임 서버 설계

게임의 서버는 각 그룹별로 데이터를 주고받으며 데이터를 공유하게 해 주어야한다. 이를 위하여 서버가 게임 상태를 보관하여 게임 상황을 여럿이 공유할 때 적합한 클라이언트 서버 아키텍처를 선택하였다.

주로 온라인 게임에는 TCP와 IPX등이 사용되는데 구현할 game server는 서로 멀리 떨어진 여러 명의 사용자들이 동시에 같은 게임을 참가하게 되어있다. 이런 조건에는 IXP보다는 TCP가 적합하므로 TCP를 기반으로 한 텍스트 처리에 뛰어나며 대부분에서 개발할 때 주로 사용되는 telnet을 프로토콜로 사용하여 구현하였다.

4.1 Architecture.

구현한 게임은 두 명의 게이머(Creator와 Joiner)가 게임을 하면 다른 게이머(Watcher)들이 게임 진행 과정을 지켜보며 자신도 답과 어는 게이머가 맞출지 배팅하여 점수를 얻는 방식의 게임이다. Creator는 게임 그룹을 생성하는 게이머이고 Joiner는 생성된 그룹에 참가하는 게이머이다. 두 명의 주 게이머의 성적과 답은 다른 게이머에게 전달되지만 나머지 게이머의 데이터는 자신에게만 전달된다. 각각의 게임 그룹이 만들어지며 각 그룹은 독립적으로 돌아간다.

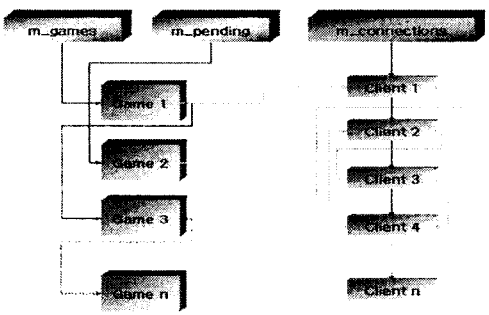

그림 1 연결 리스트를 이용한 데이터 관리

그림 1에서와 같이 연결 리스트를 사용하여 사용자 와 게임 그룹 그리고 그 그룹 내의 사용자등 게임 구현에 사용하는 데이터를 관리하였다.

4.2 기본 구성 요소

게임 서버는 User Session Manager, Billing Manager, Status Manager, System Manager 의 네 부분으로 나누어진다.

User Session Manager는 사용자가 login 하여 게임이 시작하기 전까지의 사용자의 관리를 하고 게임 끝난 다음 종료한 게임의 사용자와 그룹을 각 리스트로부터 제거한다. System Manager는 데이터 전송을 관리한다. Managers는 Quiz 데이터베이스를 가지고 퀴즈문제의 출제를 관리한다. Status Manager는 사용자의 성적을 관리한다. 본 논문에서는 서버의 기본이 되는 User Session Manager, System Manager를 구현하였다.

4.2.1 User Session manager

User Session manager는 각각의 사용자가 자신이 원하는 게임에 참가하고 게임을 시작할 수 있는 준비를 한다. 그리고 게임이 종료되어 사용자가 나가면 그 게임을 게임 리스트에서 삭제하고 전체 사용자 리스트에 각 사용자를 삭제한다.

사용자가 login을 하면 같은 id 사용자가 있는지 검색하여, 없으면 그 사용자를 받아들여 전체 사용자에 등록하고 아니면 다시 입력하도록 한다. login을 마친 사용자는 직접 Quiz게임에 참가 모드나, 직접 참가하지는 않으나 Creator와 Joiner의 진행 보며 Quiz를 푸는 모드를 선택한다. 직접 Quiz 게임에 참가 모드에서는 새 게임그룹을 만들 수 있고 이미 만들어져 다른 참가자를 기다리는 게임에 직업 참가할 수도 있다.

4.2.2 System manager

System manager는 데이터 전송을 관리한다. game group내의 사용자간에 메시지 전송과 사용자들에게 Quiz문제를 전송하고 각 사용자의 답과 배팅 결과를 전송 받는다. 게임이 시작되면 Quiz 데이터베이스로부터 manager를 이용하여 Quiz를 가져와 사용자들에게 전송한다. 그리고 각 사용자가 답을 입력하면 답을 전송받아 그 사용자가 주 게이머이면 다른 사용자들에게 전송하고 아니면 전송하지 않는다. 전송받은 답들은 Status manager에게 전달한다.

4.2.3 Status Manager

각 그룹별로 퀴즈 데이터베이스와 연동하여 퀴즈를 제출 한다. 이렇게 출제된 문제를 System manager를 이용하여 그 그룹의 사용자에게 넘겨준다. 문제를 데이터베이스로부터 가지고 올 때에는 중복되지 않게 제출된 문제를 표시한다. 또한 System Manager를 이용하여 Status Manager에 그 문제에 답을 전송한다.

4.2.4 Billing Manager

System Manager로부터 각 문제에 대한 게이머의 답과 배팅결과(Watcher일 경우)를 전송받아 manager로부터 받은 정답과 비교하여 점수를 매기고 그 결과를 데이터베이스에 저장하고 System Manager에게 전송한다.

일반적인 다중연결리스트의 처리처럼 여기서도 멀티 스레드를 사용해서 클라이언트의 요청을 처리한다. User Session Manager와 System manager는 멀티 스레드를 이용하여 클라이언트를 받아들이고 계속해서 새로운 그룹을 생성하고 게임을 진행 시켜 나간다.

5.구현 결과 및 평가

현재에는 User Session Manager와System Manager의 부분이 구현되었다. 그림 5-1은 실제로 구현한 한 구룹내에 게임서버의 실행화면이다. 한 사용자(Creator)가 게임 그룹을 만들고 다른 사용자(Joiner)가 게임에 참가하였다. 다른 두 사용자(Watcher)가 그룹에 참가하여 진행상황을 지켜보고 있다. 주 게이머(Creator와 Joiner)의 메시지를 다른 모든 게이머에게 전달할 수 있다. 그러나 Watcher는 Creator와 Joiner의 메시지를 보며 자신의 메시지를 Server에 보내고 있다.

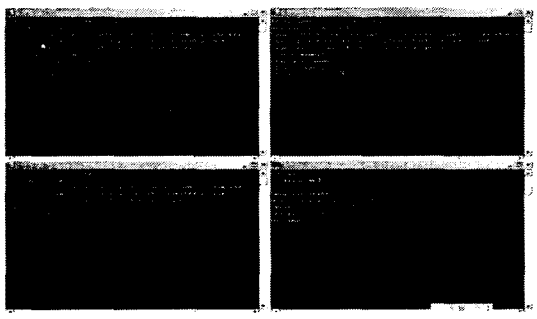

그림 5-1 하나의 그룹 안의 데이터 전달

그림5-2는 두개의 게임 그룹이 생성하여 각각 자신의 그룹 내 참가자끼리 서버를 통해 메시지를 주고받는 실행화면이다. 각 그룹에 두 명의 게이머가 있고 각 그룹의 게이머끼리 메시지를 주고받고 있다.

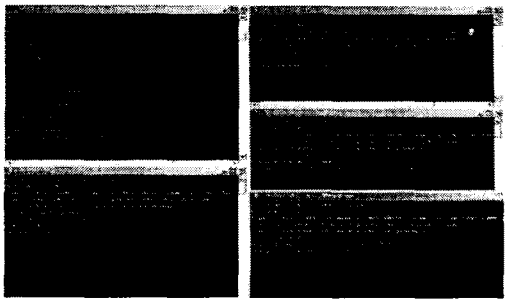

그림 5-2 여러 그룹의 예

6. 결 론

게임플레이에 중점이 되는 동기화에 대해 게임 서버 모델들과 각 서버에 맞는 프로토콜을 중심으로 알아보았다. 그리고 그것을 기반으로 온라인 게임에 많이 사용되는 텔넷 프로토콜을 사용한 게임 서버를 설계하고 구현하였다. 또한 이 게임서버는 클라이언트 서버 모델에 적용할 수 있는 게임 서버로 디자인되고 구현되었다.

게임 서버 개발에서 주요한 요소로 안정성과 확장성 그리고 처리 속도를 들 수 있다. 확장성을 높이는 방법으로는 멀티 스레드 기법을 활용하는 것과 게임 서버와 데이터베이스 등은 분산 처리 가능하도록 처리하는 것이다. 현재까지 구현된 게임서버에서는 클라이언트의 요청를 처리하는데 멀티 스레드가 사용되고 있다. 앞으로 구현될 Status manager와 Billing Manager에서 데이터베이스 이용하여 서버의 작업량을 줄이도록 할 것이다.

[참 고 문 헌]

[1] 이만재, 온라인 게임 엔진 기술 동향, 정보과학회지 제 20권 제 1호, 2002년 1월

[2] 고광현, 온라인 게임 개발 사례 : 헬브레스, 정보과학회지 제 20권 제 1호, 2002년 1월

[3] 신동일, 신동규, " 다수 사용자 기반의 온라인 게임 서버의 설계 및 제작," 전자 공학 회지 , 제 27권 제 9호, pp.954 960, 2001년

[4] 김경석, "온라인 3D 슈팅게임 만들기"

[5] 배재환, "다수 사용자 기반의 온라인 게임 서버의 설계 및 제작," 게임아카데미 강좌 7

[6] 고욱, 첨단 게임 기술 동향, 정보과학지, 제15권 8호, 1997년 8월

[7] Deloura, Mark, Game Programming Gems, Charles River Media, 2000년 8월 1일