제19회 한국정보처리학회 춘계학술발표대회 논문집 제10권 제1호 (2003.5)

인터넷 커뮤니티를 위한 JAVA 기반의 네트워크 게임 환경 설계 및 구현

김종수*, 김태석*, 최길림**, 김삼룡**
동의대학교 컴퓨터정보계열*
경남정보대학 컴퓨터정보계열**

A design and implementation of Java based network game environments design and implementation for Internet community

Jong-soo Kim*, Tea-suk Kim*, Sam-ryung Kim**
Dongeui University*
**Kyungnam College of Information Technology

Summary

Now internet is very popular. Many company services various contents in Web. One of the best way for internet community is network game service for internet member. There are some company to service high quality network game for gathering members. Those company make success for getting fee from their member. Global on-line game market is small but stability growing. Thus MicroSoft invest on line game.

On this paper, one of a java network turn game system called by "Hoola" is designed and implemented. This Application consist of Client/Server and Database module and multi-tier Architecture, because of easily translation to PDA or mobile phone.

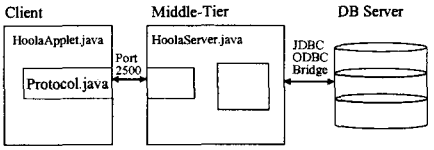
1. 서 론

인터넷은 현재 매우 대중화되어 있다. 많은 회사들이 웹에서 다양한 컨텐츠를 서비스 하고 있다. 인터넷 상에서 커뮤니티를 형성하기 위한 가장 좋은 방법이 네트워크 게임 서비스이다. 현재 국내의 유명한 몇몇 사이트들이 회원 유치를 목적으로 고품질의 네트워크 게임을 서비스하고 있으며, 일부 업체에서는 서비스의 유료화에 성공하여, 상당한 매출을 올리고 있다. 또한 세계적으로 온라인 게임시장은 아직까지 소규모이지만, 성장 가능성이 높기 때문에 마이크로소프트사도 온라인 시장에 진출하고 있다.

본 논문에서는 "홀라"라고 불리는 턴 게임 방식의 자바 네트워크 게임을 설계하고 구현한다. 어플리케이션은 클라이언트와 서버로 구성되었고, 회원이 획득한 점수관리에 대한 데이터베이스 조작에 관해서 다 계층으로 설계한 Java을 Application을 설계 구현한다. 본 솔루션이 JAVA로 구현된 것은 향후 PDA나 Mobile Phone으로 전이가 쉽기 때문이다. 네트워크 게임의 경우, 일반적으로 게임의 참여자에게 순서를 정해주고, 일반사용자에게 일반화되어 있는 게임들이 네트워크게임으로 제작되기 때문에, 시나리오의 작성 비중은 타 게임에 비해 적은 편이다. 또한 Platform에 독립적인 언어인 Java를 사용하였고, 기본적인 Multi-tier Architecture와 MVC 디자인 패턴을 채택하여 사용자 정보와 같은 중요 자원의 보호와, 향후 소프트웨어를 유지보수에 유리하도록 하였다.

2. 시스템 구성

가장 큰 골격은 Multi-tier Architecture구조이고, one-tier, two-tier가 가진 다음과 같은 문제점을 해결해 준다.



[그림1 Multi-Tier Architecture Design]

- 광범위한 데이터를 읽기 쉽게 만들어 준다.
- 네트워크를 통해 쉽게 접근하게 해준다.
- 데이터를 쉽게 은익할 수 있도록 한다.

각각의 tier는 다음과 같이 3개의 tier로 구성되며, 다음과 같은 역할을 한다.

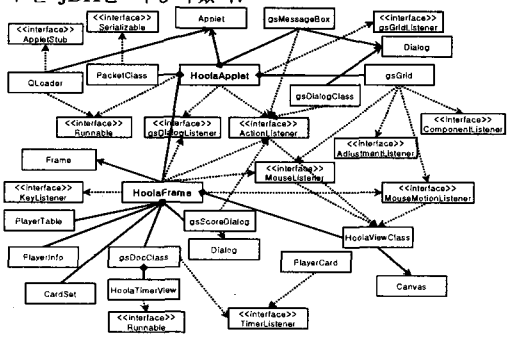
- Presentation : 데이터를 받고, 데이터 가공을 요구한다.

- Business logic : 비즈니스 룰을 수행한다.
- Data & Application : 데이터를 저장하고 액세스 한다.

Client는 네트워크 서버의 ServerSocket이 accept() 메쏘드를 이용하여 기다리고 있는 특정 포트에 접속하기 위한 소켓(Socket)을 생성함으로써, 네트워크 서버에 대한 연결 설정을 시도한다. ServerSocket을 이용하여 특정 포트에서 기다리고(accept) 있다가, 클라이언트가 연결 설정을 시도하면, 해당 클라이언트와 연결을 위한 소켓을 생성하고, 다음으로 이 클라이언트를 관리하기 위한 객체의 Thread 객체를 생성함으로써, Client와 Server의 네트워킹이 가능하도록 한다.

2.1 클라이언트의 주요 클래스

JAVA언어를 이용하여 다중사용자를 위한 네트워크 게임프로그램을 작성한다. 사용하는 Tool은 클라이언트 서버 모두 썬 마이크로 시스템이 제공하는 JAVA개발 도구인 JDK를 사용하였다.



[그림2 Client측 UML Diagram]

- HoolaApplet :

Applet을 확장한 클래스로 서버와 소켓연결을 설정하고, 클라이언트 측 구현에 있어서, 전체적인 소스를 읽어 들인다. 클라이언트는 네트워크에 처음 연결 설정했을 때 대기실로 연결 설정이 이루어지는데, 클라이언트가 대기실에 있을 때를 관리하기 위한 기능을 제공해준다. 클라이언트는 대기실에서 채팅, 방 만들기, 방 들어가기 등의 기본적인 기능을 사용할 수 있고, 방 리스트 보기 및 각 방에 대한 현황 등을 볼 수 있다.

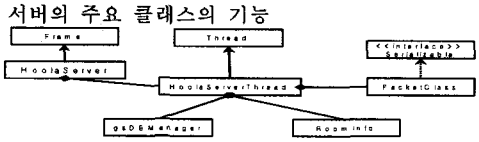
- HoolaFrame

게임 룸으로, 특정 방에 들어있는 클라이언트를 관리하기 위해 사용된다. 이때, 클라이언트는 훌라 게임에 참여할 수도 있으며, 나가기 버튼을 클릭하게 되면, 클라이언트는 다시 대기실로 나갈 수 있다.

- HoolaView

주로 사용자가 가진 패가 위치한 정보를 전해주는 애니메이션을 구현하는 GUI로써, HoolaFrame 클래스에 의해 관리된다. Canvas 클래스를 확장하여 만들었다.

2.2 서버의 주요 클래스의 기능



[그림3 Server측 UML Diagram]

- HoolaServer

클라이언트의 연결 설정을 기다리는 클래스이다. (ServerSocket의 포트에서 accept() 메소드를 이용하여 클라이언트가 접속하기를 기다린다) 클라이언트가 접속을 시도하면, 클라이언트와 네트워킹하기 위한 소켓을 생성하고, 이 소켓을 HoolaServerThread의 객체에 매개변수로 전달하여 클라이언트와 직접 네트워킹할 수 있도록 한다.

- RoomInfo

방을 관리하기 위한 클래스이다. 클라이언트의 방 개설 요청이 있으면, 방 번호, 방 아이디, 방 타이틀, 방의 사용자 아이디, 각 사용자의 사이버 머니, 사용자의 소켓을 관리한다.

- HoolaServerThread

클라이언트와 네트워킹을 전담하기 위한 클래스이다. 각 클라이언트와 연결된 소켓을 가지고 있다. 클라이언트가 방 개설을 신청하면 RoomInfo객체를 사용하여 방의 정보를 저장한다.

- gsDBManager

클라이언트의 접속 시 클라이언트에 사이버머니를 가져오고, 게임이 끝난 후, 클라이언트의 사이버 머니를 갱신하는 역할을 담당한다.

2.3 GUI Design

GUI 디자인에 다음 개념을 적용하였다.

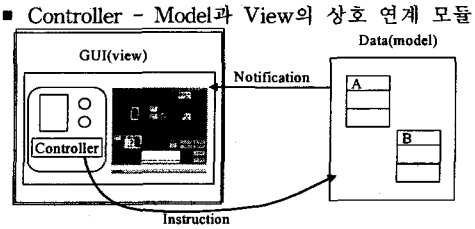
- MVC 디자인 패턴을 적용한다.
- Web Browser을 고려하여, JDK1.1 API를 사용한다.
- 객체 지향적인 GUI를 구성하는 클래스 구조를 생성한다.

웹에서의 구현과는 달리, Applet의 경우 너무 많은 이미지를 읽어 들이도록 구현을 하면, 전체 이미지를 읽는데, 다소 시간이 걸릴 수 있다.

2.3.1 Model-View-Controller Patton

본 Application은 GUI와 Protocol을 독립적으로 개발할 수 있는 전략인 Model View ontroller(MVC) 디자인 패턴을 적용하였고, 다음을 고려하였다.

- Model - 데이터의 내부적 표현
- View - GUI을 나타낸다.
- Controller - Model과 View의 상호 연계 모듈



[그림4 Model-View-Controller Design Patten]

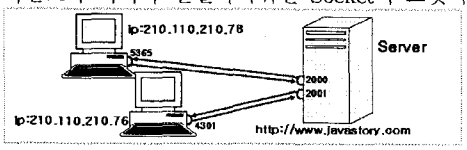
2.3.2 MVC 패턴의 장점

- 수행부의 변화가 다른 것에 영향을 미치지 않게 한다.
- 적은 클래스에 너무 많은 코드를 넣는 것을 피하도록 한다.
- 데이터의 View는 GUI에서 보여 지는 데이터의 내부적인 표현인데 어떤 변화가 일어나면 Model은 View에게 알리고, Model과 View간의 상호작용은 Controller가 한다.

2.4 TCP/IP와 프로토콜 정의

소켓(Socket)은 프로그래밍 모델에서 프로세스간 소통의 끝점을 가리키기 위해 명명되었다. 게임의 특성상 Web에서 실시간 자료의 전송이 가능해야 하므로, TCP/IP(Transmission Control Protocol/Internet Protocol)기반의 프로토콜을 채택하였다. java에서 TCP/IP프로토콜로 네트워크 통신을 위해 API를 제공해

주는데, 서버측의 주요 클래스로 ServerSocket클래스와 서버측에서 클라이언트와 통신을 연결하기위한 Socket, 클라이언트가 서버와 연결하기위한 Socket이 그것이다.



[그림5 네트워크 연결의 개요]

Java에서는 ServerSocket과 Socket에 기반으로 하고, 스트림을 사용하여, 패킷을 주고받는다. 다양한 스트림 모델을 제공하는데, 본 논문에서는 객체를 주고받을 수 있도록 ObjectInputStream과 Object-OutputStream을 사용하였다. 다음은 클라이언트와 서버간의 프로토콜을 정의하였다.

Protocol	기 능
ICR	권한 확인
IDF	신상 명세 발송
SRR	방만들기 요청
SRC	방 리스트 접수
GMR	리스트 접수 완료
MRD	방만들기 불허
OVU	정원 초과 메시지
RET	없어진 방
SRR	방만들기 요청
RGD	초기 게임 데이터 접수
RRE	방 참가 요청
ERM	방 나가기 메시지 발송
SVM	사이버머니 저장
ACD	접속 승인 요청
CHT	대기실 채팅 메시지
SUL	대기실 유저 리스트 접수
STC	방/대기실 리스트 접수
MRA	방만들기 허용 접수
ARE	방 참가 허용 접수
RID	룸 아이디 다름
RLL	방 참가자 리스트 갱신
RUL	방 참가 리스트 후 초기화
RCM	방 채팅 메시지
RFS	새로 고침
GDT	게임 데이터 접수
END	애플릿 종료

[표1 클라이언트와 서버간의 프로토콜]

2.5 JAVA 컴퓨터 그래픽스

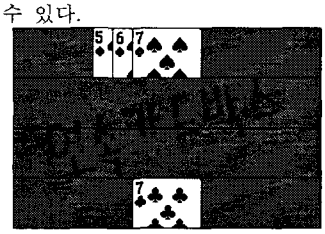
애니메이션 기법의 사용은 게임의 흥미를 더하기 위해 필수적이라 할 수 있다. 게임에 따라서, 2차원 애니메이션이 필요한 경우와 3차원 애니메이션이 필요한 경우가 있는데 각각을 빠른 시간에 구현하기 위해서는 여러 가지 최적화된 알고리즘이 필요하게 된다. 다음에 JAVA를 이용한 애니메이션 구현에 기본적으로 알아야 할 몇 가지를 설명하였다.

2.5.1 Double Buffering

Thread를 이용한 애니메이션 구현 시, 크기가 큰 이미지를 빠른 시간에 한 화면에서 그려야 하는데, 자바가 가진 Graphics객체를 사용한다. update()를 override하여 애니메이션을 구현하더라도 여전히 깜빡거리는 현상이 있다. 이 현상을 없애기 위해 가장 많이 사용하는 것은 Double Buffering 기법이다. Double Buffering을 위해서 기본적으로 2개 이상의 Graphics객체를 사용하고, 각각의 Graphics객체를 Swap하는 방식으로 화면의 깜박임을 없앤다.

2.5.2 Clipping

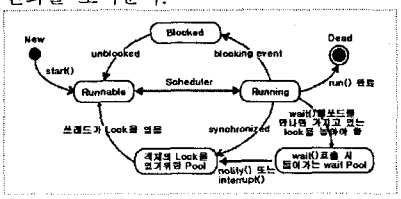
편집 등의 목적으로 그림의 일부를 잘라 내거나 화면의 특정 영역을 설정하고, 그 영역의 밖에서는 그려지지 않도록 한다. 이 기술을 사용하면, 큰 그림의 일부만 그리게 함으로써, 전체적으로 그림을 다시 그려야 할 때, 그림의 깜빡임이나, 그림을 그리는 속도에 향상을 가져올 수 있다.



[그림6 Clipping의 예]

2.6 Synchronization

JAVA를 이용한 네트워크 게임에서는 서버에 접속한 사용자 개개인을 각각의 Thread을 생성하여 관리한다. 여러 개의 Thread가 서버의 자원을 공유하게 되는데, 이때 반드시 고려되어야 하는 것이 정확한 정보를 클라이언트에게 알려주는 것인데, 여러 Thread가 사용하는 자원의 동기화와 관련 있다. 서버 측에서 발생되는 여러 개의 Thread가 하나의 자원을 공유하면, 공유된 자원에 대하여 잘못된 자료의 생성과 참조가 생길 수 있다. 이러한 문제는 동기화로 해결한다. 데이터를 동기화 했을 경우, 2개 이상의 Thread가 각각의 데이터에 대한 Lock을 가지고 서로 다른 Thread가 가진 데이터에 대한 Lock을 놓아 주기를 기다리는 경우가 발생하는 데, 이러한 경우 DeadLock이 발생하며, DeadLock은 시스템을 비정상 종료 시키는 원인이 될 수 있다. 이런 문제는 wait()와 notify()를 이용하여 해결할 수 있다. 아래의 그림이 wait()와 notify()을 호출했을 때, Thread의 상태변화를 보여준다.



[그림7 wait()와 notify()가 호출될 때 변화]

2.7 JDBC-ODBC Bridge

회원의 점수를 관리하기 위한 방법으로 서버측에서 JDBC를 이용하거나, CGI를 통해서 데이터베이스를 조작하는 방법이 있다. 본 논문에서는 JDBC-ODBC Bridge를 사용하여 회원이 획득한 점수를 관리 하였다.

Application
JDBC Driver Manager
JDBC-ODBC Bridge
ODBC Driver Manage
ODBC Driver Libraries

[그림8 JDBC-ODBC Bridge]

JDBC-ODBC Bridge는 JDBC드라이브가 JDBC의 호출이 있으면 ODBC를 조작한다.

3. 네트워크 게임 구현

JAVA로 작성된 Web 기반 네트워크 게임을 구현하

기 위해서 기본적으로 Server측에서는 Server측 OS, Database, WebServer가 있어야 하고, JDK(자바개발도구)가 있어야 한다. Client측에서는 마이크로소프트사의 Internet Explorer로 테스트 하였다.

3.1 게임 구현 환경

JAVA언어의 특성상 JVM(Java Virtual machine)이 있는 플랫폼에서는 실행이 가능하다. 클라이언트와 서버 공히 JDK1.3.1을 사용하였지만, Applet의 구현은 JDK1.1 이상의 API을 사용하는 것을 피했다. 왜냐하면 Web Browser가 지원하지 않기 때문이다.

3.1.1 Server측 환경

- Server : Windows 2000
- 웹서버 Server : IIS
- RDBMS : MS_SQL Server 2000

3.1.2 Client측 환경

- 웹브라우저 : Internet Explorer

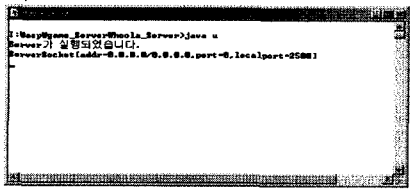
3.1.3 Server측 웹 서버 실행

회원 등록자에 한하여 네트워크 게임을 할 수 있도록 하기 위하여 웹 서버를 실행한다.

[그림9 웹 서버 실행 후 클라이언트 측 로그인]

3.1.4 Server측 게임 서버 실행

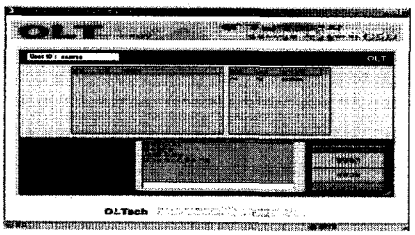
웹 브라우저를 이용한 사용자들이 애플릿(Applet)을 통하여 서버에 소켓연결 할 수 있도록 게임 서버를 실행 시킨다.



[그림10 Server측 게임 서버 실행]

3.1.5 Client측 사용자 로그인

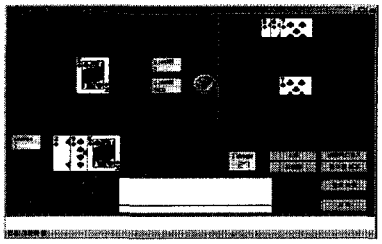
로긴한 회원은 "Hoola"게임을 위한 방을 만들 수 있다.



[그림11 Client측 사용자 로그인]

3.1.6 Client측 게임 진행

게임의 진행은 방에 참여한 2~5명의 회원이 순서를 기다리며 게임할 수 있다.



[그림12 Client측 사용자 게임 진행]

4. 결론 및 연구과제

네트워크 게임은 직접적인 수익 모델이 되는 경우도 많고, 인터넷에서 커뮤니티를 형성하는데 아주 좋은 방법 중의 하나이고, 그로인한 부차적인 수익을 발생시킬 여지가 많다. 이번의 솔루션 구현은 네트워크게임의 기본적인 기능을 중심으로 구현이 되었고, 향후 지속적인 유지보수를 통하여 좀더 사용자 편의적인 게임으로 발전시킬 것이고, Mobile 솔루션으로 만들 계획이다. 특히 게임의 인공지능 분야를 추가하여 클라이언트에게 흥미로운 게임을 할 수 있도록 유도할 것이다. 본 게임 솔루션이 다자간 네트워크 게임을 하는 기본 기능을 제공하나, 다음과 같은 개선의 여지가 있다.

- 편리한 유지 보수를 위한 MVC 디자인 패턴의 강화
- 다이나믹한 게임을 위한 애니메이션 보강
- 1인용 플레이어를 위한 인공지능 개발
- PDA와 모바일 솔루션으로의 전이

참고문헌

[1] Patrick Naughton, Herbert Schildt, *The Complete Reference JAVA 1.1*, Osborne Mc Graw Hill, 1998(1028 pages)

[2] Patrick Chan, Rosanna Lee, doublas Kramer, *The Java Class Libraries Second Edition Volume1*, Addison Wesley, 1988(2050 pages)

[3] James Gosling, Frank Yellin, Java Team, *The Java Application Programming Interface Volume1*, Addison Wesley, 1996(494 pages)

[4] John Lewis, William Loftus, *Java Software Solutions Foundations of Program Design*, Addison Wesley, 1998(857 pages)

[5] Sun microsystems, *sun educational serviecs Java Programming SL-275, SunSoft Press, 2000*(720 pages)

[6] Sun microsystems, *sun educational serviecs Advanced Java Programming SL-300, SunSoft Press, 2000*(400 pages)