서버 부하 감소를 위한 P2P 기반 데이터 서비스 시스템의 설계 및 구현

이 광 현[†] · 전 형 수[††] · 유 철 중[†††] · 장 옥 배[††††]

요 약

최근 홈 네트워킹에 대한 관심이 증가하면서 멀티미디어 클립서비스에 관한 연구가 진행되고 있다. 가정내 네트워크의 대역폭이 가지는 한계로 인해 클라이언트-서버 방식을 통한 대용량 멀티미디어 자료들을 장시간 동안 받아야하고 서버에 트래픽이 증가하는 문제점들이 나타나고 있다. 본 논문에서는 서버를 거치지 않고 가정과 가정을 직접 연결하여 자료를 교환할 수 있는 Peer-to-Peer(P2P) 기술이 적용된 자료공유 네트워크 시스템을 설계하였고 서버의 트래픽을 줄이도록 하였으며 Java 언어로 플랫폼 독립적인 시스템을 구현하였다.

Design and Implementation of P2P-Based Data Service System to Reduce Server Load

Kwang Hyon Lee[†] · Hyung Su Jeon[††] · Cheol Jung Yoo[†††] · Ok Bae Chang[††††]

ABSTRACT

Recently, as interest in Home networking is increased, research about multimedia clip service is gone. Because of the limit to bandwidth of home network, the problem of server traffic is increased by transferring many multimedia data through Client-Server Way for long. The data share network system applied is designed and implemented to solve this problem. P2P is connected directly between clients and exchanges data without going through a server. We implemented platform-independent System using Java and applied P2P skill to reduce server traffic.

키워드 : Peer-to-Peer, 클라이언트/서버(Client/Server), 부하분산(load valance)

1. 서 론

최근 초고속 정보통신망이 일반 가정에 보급됨에 따라 가전제품들을 홈 네트워크에 연결하여 인터넷을 통한 정보서비스가 가능하도록 인터넷 정보가전이 발전하고 있다. 또한, MP3 파일을 공유하는 Napster라는 새로운 킬러 애플리케이션이 인터넷 비즈니스에서 Peer-to-Peer(이하 P2P)라는 용어로 자주 쓰이고 있다. 이는 곧 기존 인터넷 비즈니스의 지배적 구조였던 클라이언트-서버 중심의 비즈니스 모델이 P2P 구조로 변화하고 있다는 것을 의미한다[1-3].

인터넷 비즈니스 모델로서 P2P가 등장하게 된 기술-환경적 요인 중 가장 중요한 것은 인터넷 대역폭의 광대역 및 고속화라고 할 수 있다. 그러나, 대부분의 인터넷 사용자가 저속 모뎀 환경에 있는 상황에서는 Napster와 같은 모델이 성립하기 힘들다. 최근 DSL, ADSL, 케이블 모뎀 등의 급속한 보급은 P2P 비즈니스 모델 탄생의 가장 중요한 인프라적 배경이 되고 있다. P2P 비즈니스 모델은 파일이나 CPU를 공유할 뿐만 아니라 공동작업, 커뮤니케이션, 검색과 상거래 또는 호스팅 기능을 수행하고 있고 모바일 환경에서도 P2P 네트워크는 성장하고 있다[4]. 전자상거래[5] 분야의 경우 한국의 open4u가 세계 최초이며 현재까지 유일한 P2P 전자상거래 네트워크라는 점에서 특히 주목할 만하다. P2P 비즈니스 모델은 인터넷의 비즈니스 모델이 인터넷의 하부구조를 닮아가게 된다는 일종의 가설을 뒷받침한다. 즉 인터넷 그 자체의 본래 설계 구조인 P2P 구조가 이제는 비즈니스 모델 차원에서 실현되고 있다고 해석할 수 있는 것이다. 또한 P2P 비즈니스 모델[6]은 인터넷 비즈니스 모델이 중개에서 네트워킹으로 변화한다는 중요한 가설을 뒷받침하고 있는 것이다.

P2P 통신 방식은 특정한 컴퓨터에 의존하지 않을 수 있으며, 어느 한 지역과 지역의 통신회선이 두절되더라도 또 다른 회선을 통해 상호연락이 문제없이 사용할 수 있는 통신체계이다. 따라서 인터넷상에 존재하는 모든 웹사이트는 논리적으로 동등한 기회를 지니고 있으며, 네트워크들이 그 물처럼 연결되어 어느 한 선로가 작동하지 않더라도 목표로 하는 웹사이트까지 아무 어려움 없이 접근할 수 있다.

† 준 회 원 : 전북대학교 대학원 컴퓨터정보학과
†† 준 회 원 : 전북대학교 대학원 전산통계학과
††† 종신회원 : 전북대학교 컴퓨터과학과 교수
†††† 정 회 원 : 전북대학교 전자 · 정보공학부 교수
논문접수 : 2002년 5월 4일, 심사완료 : 2002년 7월 23일

이러한 특성에 의하여 인터넷에서는 일방적이 아닌 상호작용을 통하여 모든 일이 전개되고 있으며, 웹에 접속한 모든 사람들은 직접 얼굴을 보지 않고 서로간에 주고받는 정보에 의하여 모든 일을 처리할 수 있도록 한다[7].

그러나, P2P 기술을 이용한 시스템의 여러 변형 형태가 있음에도 자료를 다운 받는 환경의 한계와 간편한 클라이언트 프로그램의 부재로 많은 어려움에 놓여 있고 상용화가 늦어지고 있다. 이것은 클라이언트-서버 방식에 익숙해진 사용자들이 서버 트래픽으로 인해 더 이상 사용자들의 요구를 충족 시켜줄 수 없다는 사실을 알면서도 이에 대체할 마땅한 서비스 방식을 찾지 못하고 있기 때문이다.

따라서, 자료의 목록만을 중앙 서버에 올려서 중앙 서버의 역할을 최소화하고 클라이언트간의 자료교환이 이루어지게 하여 서버의 부하를 분산시키는 방법에 대한 연구가 필요하다. 그리고 클라이언트가 접속하지 않은 상태에서는 검색이 되지 않는 단점을 보완하는 방법으로 서버에 한번 접속한 클라이언트의 자료 목록을 가지고 있어서 자료를 가진 클라이언트가 접속하고 있지 않은 상태에서도 검색이 가능하도록 하는 기능이 필요하다. 이것은 접속했을 때만 검색이 가능한 타 프로그램과는 구별되는 기능으로 원하는 자료의 빠른 검색을 도와줄 뿐만 아니라 서버의 부하를 미리 분산시키는 역할도 하게 된다. 본 논문에서는 이러한 일련의 연구와 함께, Java 언어를 사용하여 플랫폼에 독립적인 장점을 지닌 시스템을 설계 및 구현하는 방법을 제시한다. 또한 실험을 통해 P2P 자료교환의 시간 기준에 따른 성능을 비교·분석한다.

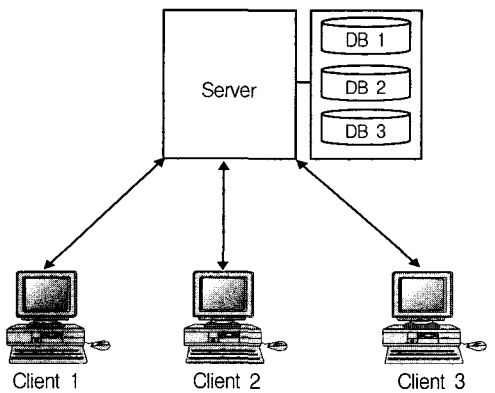
본 논문의 구성은 다음과 같다. 우선 2장에서 P2P 기술을 이용한 시스템을 살펴보고, 3장에서는 시스템들의 단점을 해결하기 위한 데이터 서비스 시스템인 자료공유 네트워크를 설계하고, 4장에서 자료공유 네트워크 시스템의 구현과 실험 결과를 보여주고, 5장에서 결론 및 향후 연구 방향을 서술한다.

2. 관련 연구

P2P는 상대적으로 클라이언트-서버 구조에서 서버가 없고 클라이언트의 역할이 강화된다는 가장 본질적인 특징을 가진다. Napster의 클라이언트 프로그램은 사용자의 지정된 디렉토리로부터 파일의 이름을 읽고 인덱스 서버에 보내며, 다운로드 요청이 올 때 이를 보내주는 일종의 서버의 역할까지 수행하게 되는 것이다. Napster 클라이언트 프로그램은 파일 검색 요청을 보내고 특정 파일의 다운로드의 역할을 요청하는 클라이언트로서의 역할도 하게 된다. 즉, 이 프로그램은 서버와 클라이언트라는 두 가지 역할을 다 하게 되는 것이다. 그래서 클라이언트 프로그램이라고 부르지 않고, 피어 프로그램(Peer Program)이라고 명명하고 있다. Napster의 경우 서버는 현재 접속된 클라이언트 프로그램의 유동 IP 주소를 유지하고, 주기적으로 이들 클라이언트 프로그램의 파일 이름의 인덱스를 유지하는 역할을 하고 있다. 소리바다의 경우 접속된 클라이언트 프로그램의 유동 IP 주소를 유지하는 것으로 서버의 역할을 축소하고 있으며, GnuTella와 Freenet의 경우 IP 주소를 유지하는 서버마저도 없다. 이 경우 각 클라이언트 프로그램은 연결하고자 하는 주위의 클라이언트 프로그램의 IP 주소를 지정해야 한다. 이 정도 수준까지 되면, 클라이언트 프로그램이라는 말 자체가 잘못된 용어가 된다. 왜냐하면, 연동하는 서버가 없기 때문이다. 이 경우 피어 프로그램이라는 용어가 더욱 적합해지는 것이다. P2P라는 개념은 정보통신이론이나 컴퓨팅 구조에서는 오래 전부터 사용되어온 개념이다. LAN 환경에서 프린터, 스캐너 등의 자원을 공유하는 방법으로서도 사용되고, 인터넷을 비롯한 많은 통신프로토콜 등이 P2P에 기반하여 설계되어 있다.

P2P는 크게 2가지로 나누어지며 순수 P2P와 혼합형 P2P가 이에 속한다. 순수(pure) P2P는 쉽게 얘기하면 비슷한 성능을 가진 PC 끼리만 연결된 형태이고, 하이브리드(hybrid) P2P는 PC 끼리의 인터액션을 원활하게 해주는 서버가 개입되는 형태이다. P2P는 중앙 서버의 존재 여부에 따라 크게 2가지 방식으로 구분할 수 있으며 이에 따라 검색이 이루어지는 방식에 차이가 생기며 대표적인 예로 Napster와 Gnutella가 있고 국내에는 소리바다가 있다. Napster는 중앙 서버에 의존하는 방식으로 대부분의 작업이 중앙 서버에서 처리되며, 직접적인 파일 전송만 개인 대 개인으로 이루어진다. Gnutella는 중앙 서버가 존재하지 않는 방식으로 사용자들의 컴퓨터가 중앙 서버를 통하지 않고 직접 연결되어 있다[8]. 소리바다는 중앙 서버가 존재하지만, 중앙 서버의 역할을 최소화하고 클라이언트가 대부분의 작업을 처리하게 되어 있다. 하지만, 소리바다는 운영체제가 바뀌면 실행되지 않는 치명적인 단점이 있다. 따라서, 자료공유 네트워크 시스템은 운영체제에 상관없이 P2P로 자료 교환이 가능하며 서버에서 클라이언트의 파일 목록만을 저장하고 있으므로 서버의 역할을 최소화하면서 트래픽을 줄였다.

2.1 Napster



(그림 1) Napster 구조

사용자가 서버에 접속한 후 자신이 가지고 있는 파일 리스트를 서버에 등록하면 서버는 현재 (서버에) 접속되어 있는 사용자들의 파일 리스트를 데이터베이스화하여 저장하고 사용자가 검색어를 서버로 보내면 서버는 데이터베이스에서 검색을 한 후 검색 결과를 사용자에게 전송하고 검색 결과를 가지고 다른 사용자에게 연결하여 파일을 다운로드 받는다[9].

2.1.1 Client-Server 프로토콜

Client-Server 프로토콜은 기본적으로 사용자가 자신이 공유하고 있는 파일 리스트를 서버에 알리고 원하는 파일을 검색하는 방식을 정의한다. 서버는 8888 또는 7777 TCP 포트를 사용하고 클라이언트와 서버 사이의 모든 메시지는 다음과 같은 포맷을 따른다.

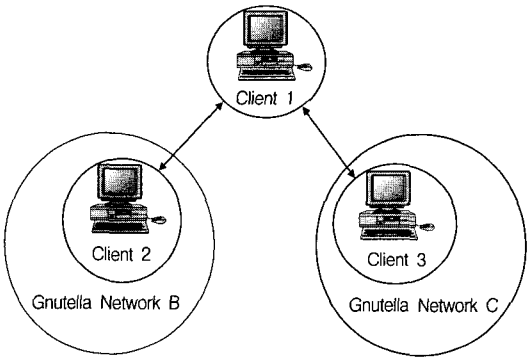
- <데이터 길이(2바이트)><명령어(2바이트)><데이터>
- 명령어 : 100(공유파일 리스트 올리기), 200(검색요청) 등

2.1.2 Client-Client 프로토콜

Client-Client 프로토콜은 사용자들이 서로 파일을 주고받고 하는 방식을 정의한다. 사용자들간의 파일 전송 방법으로는 upload, download, firewalled upload(방화벽 뒤에서의 업로드), firewalled download(방화벽 뒤에서의 다운로드) 등 4개의 전송 모드가 있다. Napster는 순수한 P2P 방식은 아니다. Napster를 다운받아 실행하면 먼저 자신이 서비스할 파일의 위치를 입력하게 한다. 그리고 Napster 서버에 로그인하는 과정을 거친다. 이때 Napster는 사용자의 MP3 파일을 분석해 그 리스트를 서버에 보고한다. 서버는 이 리스트를 저장하고 있다가 파일 찾기 요구가 들어올 때 저장된 리스트로부터 검색한 결과와 현재 해당 사용자의 온라인 여부, 회선 속도 등을 바탕으로 원하는 파일을 찾게 해준다. 여기까지의 과정에는 Napster의 중앙 서버가 개입하고 실제로 파일을 다운로드 하는 과정은 P2P 방식으로 이루어진다.

2.2 Gnutella

Gnutella는 네트워크에 연결되어 있는 1개 이상의 컴퓨터 주소를 입력하고 그 컴퓨터들을 통해 Gnutella 네트워크에 연결한다. 사용자가 검색어를 입력하면 서로 연결되어 있는 컴퓨터들끼리 검색어를 전송하면 다른 사용자들의 검색어도 내 컴퓨터를 통해 다른 컴퓨터로 전송된다[10, 11].



(그림 2) Gnutella 구조

2.2.1 Gnutella 프로토콜

Gnutella는 분산 검색 프로토콜로 네트워크에 연결된 후부터 <표 1>에 나오는 명령어를 사용하여 서로 정보를 교환한다.

2.2.2 명령어

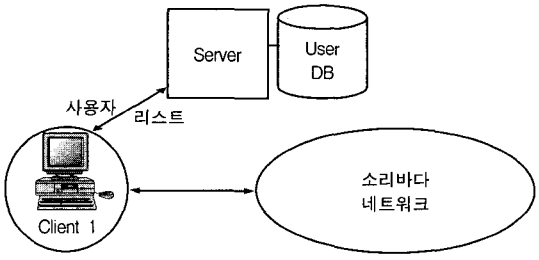
<표 1>은 Gnutella 명령어에 대한 설명을 나타내고 있다.

〈표 1〉 Gnutella 명령어

명령어	설 명
Ping	Gnutella 네트워크에서 호스트를 찾기 위해 사용되는 명령어로써, 이 명령어를 받은 peer는 Pong 명령어를 이용하여 답변한다.
Pong	*Ping* 명령어에 대한 답변으로써, *Gnutella* 네트워크에 연결되어 있는 사용자의 주소와 공유되고 있는 파일크기를 포함한다.
Query	검색 명령어를 받은 사용자는 검색어에 해당하는 공유 파일이 있을 경우 QueryHit 명령어를 이용하여 검색결과를 보낸다.
QueryHit	Query 명령어에 대한 답변으로써 검색을 시도한 사용자가 검색한 파 일을 다운로드 받아갈 수 있도록 충분한 정보를 제공한다.
Push	방화벽 뒤에 있는 사용자도 파일을 다른 사용자가 다운로드 받을 수 있도록 고안된 명령어

2.3 소리바다

Gnutella와 Napster의 중간 형태이며 중앙서버가 존재하기는 하나 사용자들이 연결 상태만 관리한다. 로그인 시 서버로부터 현재 소리바다에 연결된 사용자 리스트를 받고, 그 이후 모든 통신은 사용자들 사이에서 일어난다. Gnutella와 같이 서로의 검색어를 전송 해주는 방식이 아니라 병목현상이 발생하지 않는다. 검색어, 검색결과, 파일 전송 모두 사용자들 사이에서 발생한다[12].



(그림 3) 소리바다 구조

국내 솔루션 1호라 할 수 있는 소리바다는 Napster와 상당히 유사한 사용자 인터페이스를 가지고 있고, 동작 방식도 매우 유사하다. Napster의 사용자 인터페이스가 이미 검증을 받았기 때문인지 소리바다는 출시 후에 많은 사용자를 확보하고 있다.

3. 자료공유 네트워크 시스템 설계

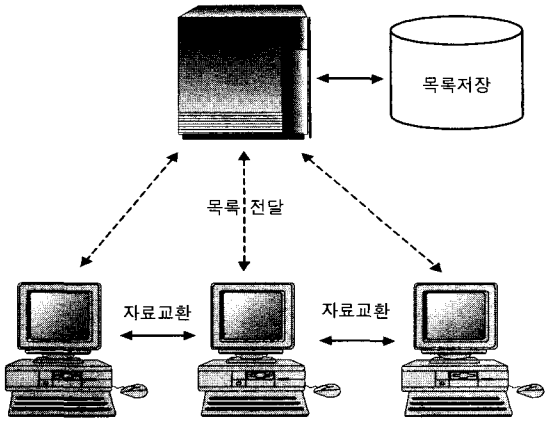
자료공유 네트워크 시스템은 인터넷을 활용하여 일반 사용자가 필요한 데이터 정보를 각각의 개인용 PC에 저장하면 서버는 사용자의 데이터 정보를 확인하여 목록을 공유함으로써 일반 가입자가 공동으로 DB에 저장되어 있는 정보를 파악하도록 운영하고 있다. 서버는 일반 가입자가 필요한 데이터를 찾고자 할 때 가입자 PC에 멀티미디어 데이터 목록을 보내주어 데이터를 검색할 수 있도록 하고 가입자와 가입자간의 PC를 연결하여 1 : 1로 정보 데이터를 교환할 수 있도록 해준다. 가입자는 서버에 있는 다른 가입자의 데이터 접근 권한을 설정할 수 있어 보안에 필요한 사용제한을 적용할 수 있도록 하고 있다.

클라이언트 프로그램은 웹 서비스를 통한 자료 검색 및 서버를 이용 MS-SQL DB에 저장되어진 자료의 목록을 검색할 수 있는 프로그램이다. 또한 인터넷을 활용한 교육용 정보 서비스를 제공함으로써 일반 사용자가 인터넷 또는 실생활에서 필요로 하는 교육용 비디오나 오디오등의 멀티미디어 데이터를 여러 사용자가 가정에서 다운 받아 사용할 수 있도록 하는 서비스를 제공한다. P2P 방식의 데이터 교환 방식을 적용한 클라이언트 프로그램은 가입자가 쉽게 멀티미디어 데이터를 공유할 수 있고 다른 사용자가 검색할 수 있도록 하여, 정보 데이터를 1 : 1로 교환할 수 있게 한다. 이러한 방식은 서버에 클라이언트의 인덱스 정보만을 남겨두고 사용자들간의 자료의 송·수신이 이루어질 때 서버를 경유하지 않아 서버의 과부하를 줄일 수 있을 뿐 아니라, 더 많은 정보를 공유할 수 있도록 한다.

자료공유 네트워크 시스템은 데이터를 다운로드 하다가 서버가 정지되는 경우에 생기는 문제점을 해결하기 위해 P2P 고유의 특성을 활용하여 상호간에 데이터를 교환할 경우 서버가 정지되어도 전송 데이터에는 아무런 지장이 없다. 특히 일반 사용자 누구나 손쉽게 사용할 수 있고, 모든 운영체제 환경에서 실행이 가능하도록 자바언어로 구현하였으며, 가정에서 교육용 데이터 정보 이외에도 상호간에 필요로 하는 교육정보를 교환할 수 있도록 하는 P2P 교환방식에 초점을 맞추어 개발하였다.

3.1 시스템 구성도

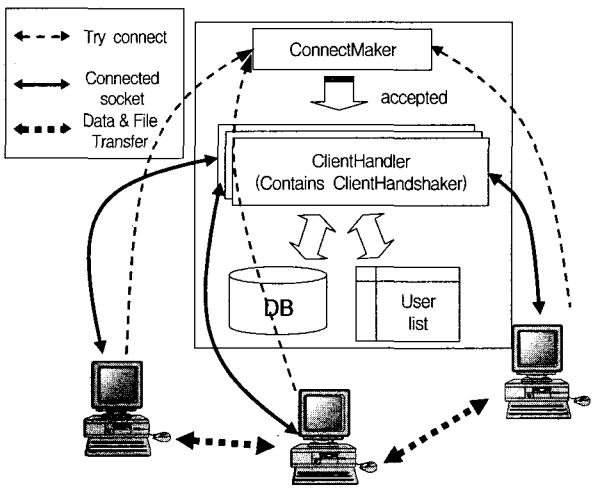
(그림 4)는 자료공유 네트워크 구조를 나타내는 것으로 서버와 클라이언트의 관계 및 자료의 송·수신 흐름을 나타내는 것이다. 서버에 접속한 일반 사용자들이 서버에 등록된 목록을 검색한 후, 원하는 데이터가 있는 다른 사용자에 접속하여 데이터를 받아 가는 흐름을 보여주고 있다. 자료공유 네트워크는 크게 서버와 클라이언트 부분으로 구성된다. 서버는 각 클라이언트를 인증하며 클라이언트 사이를 연결시켜주고 목록 검색요청이 들어 왔을때 DB에 저장된 자료를 검색하여 보여준다. 클라이언트에서는 하이브리드 P2P 방식을 이용하여 사용자가 원하는 자료목록을 서버에 검색요청하고, 검색된 목록을 받아와 파일이 있는 클라이언트에 접속하여 실제파일을 받아온다. 또한 클라이언트는 파일 요청을 받아들이거나 파일을 보내주는 기능을 포함한다.



(그림 4) 자료공유 네트워크 구조

3.1.1 서 버

서버 구성은 (그림 5)와 같이 클라이언트의 접속을 받아들이는 부분, 클라이언트를 핸들링 해주는 핸들러 부분으로 나눌 수 있다. 클라이언트 핸들러는 클라이언트 핸드 쉐이커를 포함하고 있다.



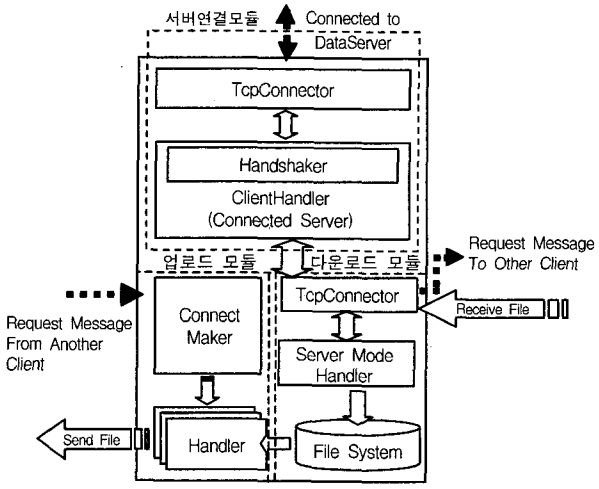
(그림 5) 서버 구성도

클라이언트가 서버에 접속 요청을 하면 커넥트 메이커(Connect Maker)가 접속을 받아서 핸들러에게 넘긴다. 접속을 받은 핸들러는 우선 서버와 통신을 할 수 있는지 접속을 다시 핸드쉐이커에게 넘긴다. 접속을 받은 핸드쉐이커는 메시지 통신을 하면서 인증을 거치고 인증과정을 통해 자신과 통신할 수 있는지 검사하게 된다. 인증이 끝난 접속은 다시 핸들러로 돌아와 클라이언트와 대화할 준비를 하게된다. 인증된 클라이언트의 정보를 유저리스트라는 하나의 맵(map)

에 클라이언트 사용자 이름, IP, 클라이언트에 파일을 받아 갈 때 접속할 수 있는 포트번호 등을 저장해 현재 클라이언트가 서버에 접속해 있다는 것을 알 수 있게 한다. 연결된 클라이언트가 자신의 공유파일 목록을 서버로 보내게 되면 서버는 목록을 디코드 해서 DB에 저장하며, 새로운 데이터 목록을 저장하기 전에 사용 중이던 클라이언트에 대한 목록을 모두 지우게 된다. 이때, 검색 요청이 들어오면 DB에 저장된 모든 클라이언트가 올려놓은 공유 목록에서 검색하여 검색 요청이 들어온 클라이언트로 검색된 목록을 보내준다.

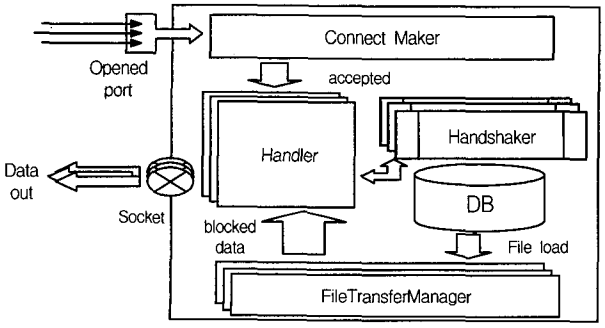
3.1.2 클라이언트

클라이언트 구성은 (그림 6)과 같다.



(그림 6) 클라이언트 구성도

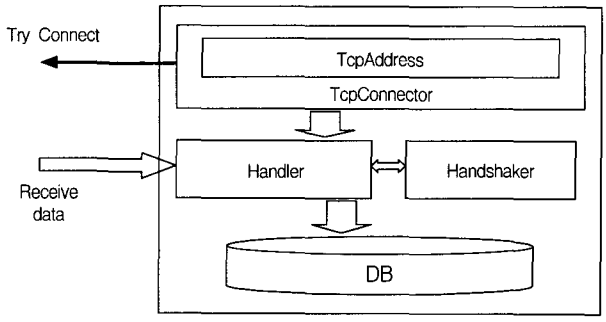
클라이언트는 크게 세 부분으로 구성되어 있다. 첫째, 서버연결 모듈은 서버와 클라이언트를 연결해 주고 연결관계를 유지하는 부분이며, 둘째, 업로드 모듈은 클라이언트측의 파일 업로드 모듈로 다른 클라이언트로부터 파일 전송 요청을 받아들이거나 요청된 파일을 시스템에서 읽어 들여 클라이언트로 보내주는 부분이다. 세째, 다운로드 모듈은 클라이언트측의 파일 다운로드 모듈로 자신이 다른 클라이언트에 정보를 요청하였을 때 그 요청이 받아들여지면 파일을 받아와 파일 시스템에 저장하는 역할을 한다. 서버와 접속은 클라이언트 프로그램에서 로그인 할 때 자신의 접속을 알리고, 자신이 공유해놓은 목록에 대한 새로운 정보를 서버에 보내고, 올려진 목록을 검색어를 이용해서 검색할 수 있도록 한다. 클라이언트들간의 연결 상태는 (그림 7), (그림 8)과 같다.



(그림 7) 클라이언트 파일 업로드 모듈

클라이언트 파일 업로드 모듈에서도 커넥트 메이커가 있어 다른 클라이언트가 접속을 할 때 하나의 열려진 포트를 찾아서 그 부분을 열어두어 클라이언트가 접속할 수 있게 한다.

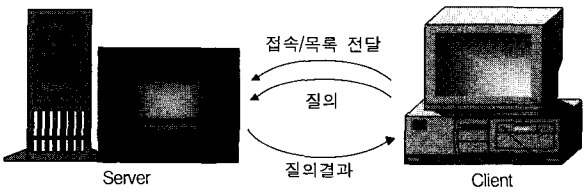
접속된 다른 클라이언트의 접속은 핸들러로 넘기게 되고, 핸들러는 다시 핸드쉐이커에 넘겨서 클라이언트와 통신할 수 있는지 검증하게 된다. 만약 통신할 수 없는 상황이거나, 요청된 파일이 자신의 것이 아닐 때에 접속을 거부하게 된다. 인증과정을 통해 접속이 되면, 핸들러는 파일 트랜스퍼 매니저를 호출해 파일 시스템에 존재하는 파일을 읽어 블록별로 전송하게 된다.



(그림 8) 클라이언트 파일 다운로드 모듈

클라이언트 파일 다운로드 모듈은 클라이언트가 다른 클라이언트로 접속을 하여 파일을 받아오는 기능을 한다. 일단 사용자가 어떤 파일을 받아 오겠다고 하면, 검색하고자 하는 파일이 있는 클라이언트로 접속하게 된다. 이 때 접속은 TcpConnect를 이용한다. TcpConnector가 접속 여부를 확인하여 핸들러에게 넘기고 올바른 통신과정을 거치기 위해서 핸들러는 또 다시 핸드쉐이커에 접속을 넘긴다. 핸드쉐이커는 자신의 ID나 클라이언트로부터 정보가 될만한 사항을 보내주고 인증을 마친다. 사용 인증이 끝나면 파일을 요청한 클라이언트로부터 파일이 전송되기 시작하는데, 핸들러는 파일 블록을 받아와 자신의 로컬 파일 시스템에 저장한 후 파일을 다 받았으면 접속을 끝낸다.

클라이언트 프로그램 구성은 클라이언트와 서버 연결 부분과 클라이언트간의 연결부분으로 나뉘어 있다.



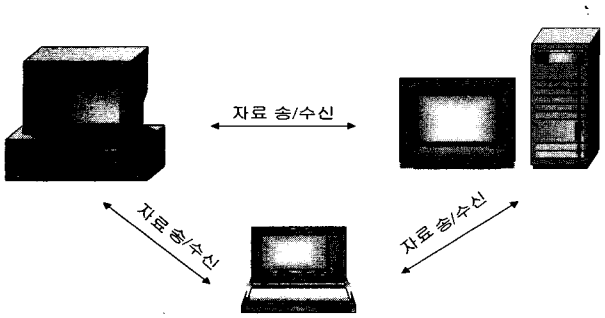
(그림 9) 인증 및 목록 전달

(1) 클라이언트와 서버 연결

클라이언트는 서버에 접속하여 공유목록을 전달하고 서버의 정보를 통해 데이터 목록을 검색하여 다른 클라이언트의 데이터 목록을 대한 정보를 전달받는다.

(2) 클라이언트간의 연결 구성

클라이언트가 데이터 정보를 서버로부터 전달받아 다른 클라이언트와 서버를 거치지 않고 직접 자료 송·수신이 일어나는 과정을 나타내고 있다.



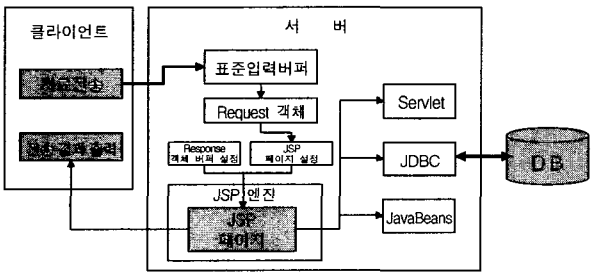
(그림 10) 클라이언트들간의 자료 송·수신

3.2 시스템 설계

클라이언트 프로그램을 이용하기 위해서는 서버 웹사이트에 들어가서 사용자 가입 및 인증을 받는 과정이 필요하다.

3.2.1 사용자 가입 및 인증

클라이언트 프로그램을 이용하기 위하여 자료공유 네트워크 시스템을 제공하는 웹사이트에 접속하여 사용자로 등록한 후, 시스템을 사용할 수 있도록 한다. 웹사이트의 응용 프로그램은 JSP로 구현되어 있으며, 가입을 하지 않으면 클라이언트 프로그램을 사용할 수 없다. 서버에 가입되었으면 웹사이트의 방문 없이 클라이언트 프로그램을 실행시켜서 사용자 인증을 거친 후, 다른 사용자와 여러 데이터 정보를 공유 및 다운받아 이용할 수 있다.

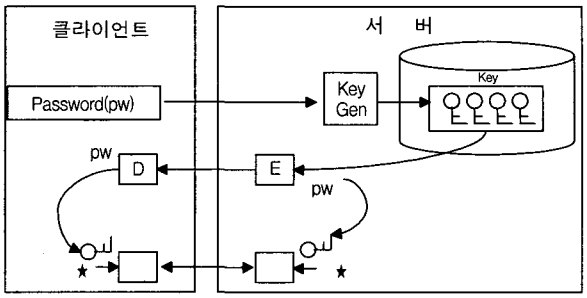


(그림 11) 사용자 가입 및 인증 구성도

(그림 11)는 클라이언트와 서버간의 전송관계를 나타내고 있는 것으로 클라이언트가 브라우저를 통해서 사용자 자료 입력 및 전송을 요구 시 서버에서는 표준입력버퍼 요청자료 임시저장을 통하여 요구객체에 자료를 보낸다. 보내진 요구객체는 JSP 페이지 설정과 응답객체 버퍼설정과 함께 표준출력버퍼 JSP 페이지 실행 결과에 보내지게 된다. 전송된 내용이 클라이언트의 브라우저에 보내지게 되고 클라이언트가 전송된 결과를 볼 수 있게 된다.

3.2.2 암호화 패스워드 인증과 DB 테이블 구성

(그림 12)은 클라이언트에서 입력한 암호가 서버에서 Key_Gen을 통하여 암호화[13]되고 암호화된 코드가 DB에 저장되어 있다가 클라이언트의 인증 요청이 있을시 클라이언트에서 입력한 패스워드와 서버에 저장된 코드를 서로 비교하여 인증을 통한 접속 결과를 확인할 수 있는 전달과정을 나타내고 있다.



(그림 12) 암호화키의 전달과정

서버에서 사용되는 DB 테이블은 사용자가 공유하고 있는 파일의 정보를 보관하고 있는 테이블과 사용자의 인증과정에 필요한 정보들의 테이블로 구성되어 있다.

〈표 2〉 저장될 목록 DB 테이블

열 이름	데이터 형식	길 이
filename	varchar	128
directory	varchar	128
filesize	varchar	25
title	varchar	128
user_serial_id	int	4
userid	varchar	20
keyword	varchar	256

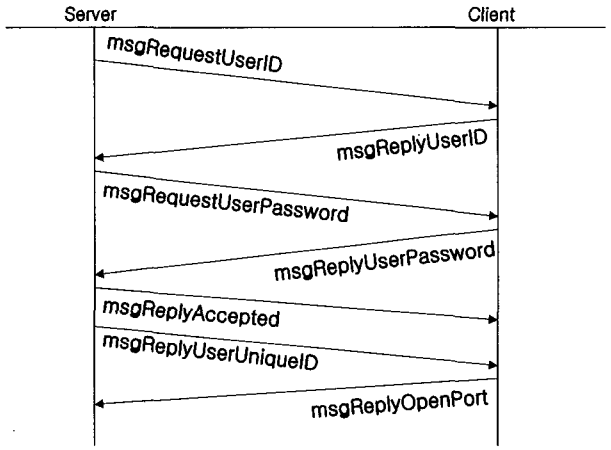
〈표 3〉 홈페이지 가입 목록 DB 테이블

열 이름	데이터 형식	길 이
UniqueID	int	4
userid	varchar	10
userpass	varchar	25
username	varchar	10
socialid1	char	6
socialid2	char	7
birth	datetime	8
address	varchar	150
zip	char	7
email	varchar	50
job	varchar	50

클라이언트가 서버의 다른 사용자 정보 목록을 볼 수 있도록 DB 테이블 구성을 <표 2>와 같이 구성하였다. 이와 같은 테이블 구성은 사용자가 서버에 접속하여 다른 사용자가 가지고 있는 파일의 정보 리스트를 볼 수 있게 한다. <표 3>은 자료공유 네트워크 시스템을 사용할 수 있도록 홈페이지에 가입할 때 필요한 사용자 정보를 저장하는 기능을 가진 테이블이다.

3.2.3 자료공유 네트워크 프로토콜

서버와 클라이언트간의 원활한 데이터 전송이 이루어질 수 있도록 하는 프로토콜은 개인 PC의 사양이 각각 다르기 때문에 나타날 수 있는 여러 환경에 유연하게 대처할 수 있도록 서버와 클라이언트간의 통신 규약을 정해 놓았다. (그림 13)은 서버와 클라이언트의 메시지 전달 과정을 나타내는 것으로 클라이언트가 서버에 접속을 시도하면 서버에서 클라이언트에 UserID를 요청하고 클라이언트는 UserID를 응답한다. 그러면 서버에서 패스워드 요청이 들어오고 클라이언트는 이 요청에 응답한 후 서버에 접속이 이루어지고 포트가 열리게 되는 과정을 나타내고 있다.



(그림 13) 서버와 클라이언트의 연결 과정

<표 4>는 서버의 DB에 저장될 정보 목록을 클라이언트가 유니코드로 시작되는 인코딩 된 파일정보를 서버에 보내고 서버에서 디코딩하여 DB에 입력하는 과정을 나타낸다.

〈표 4〉 클라이언트가 서버에 보내는 정보 목록

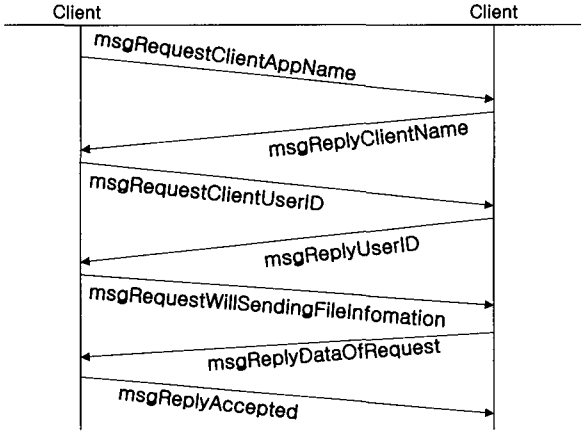
클라이언트	서 버
msgRequestClearMyShareFileList	DB에 있는 해당되는 클라이언트의 목록을 지운다.
유니코드(\u0002\u0001\u0003)로 시작되는 인코딩 파일정보를 보낸다.	인코딩된 파일 정보를 디코딩해서 DB에 입력한다.

<표 5>는 클라이언트가 서버에서 목록을 검색할 때 서버에 넘겨주는 메시지의 종류를 나타내는 것으로 파일이름, 제목, 키워드 등의 검색어를 사용할 수 있다.

〈표 5〉 클라이언트가 검색하는 목록

클라이언트	서 버
msgRequestFindByFilename	메시지와 같이 넘어온 검색어를 파일이름에서 찾는다
msgReqeustFindByTitle	메시지와 같이 넘어온 검색어를 제목에서 찾는다
msgReqeustFindByKeyword	메시지와 같이 넘어온 검색어를 키워드에서 찾는다
msgRequestFindByAll	메시지와 같이 넘어온 검색어를 파일이름, 제목, 키워드 모두에서 찾는다

(그림 14)는 클라이언트간의 메시지 교환 흐름을 나타내는 것으로 데이터를 가진 클라이언트가 상대 클라이언트에 이름을 요청하면 상대 클라이언트가 이에 응답하고 차례로 UserID의 확인이 이루어진 후 파일의 정보를 요청한 클라이언트에게 목록이 보내진다. 그러면 목록을 보고 상대 클라이언트는 원하는 데이터를 받게 된다.



(그림 14) 클라이언트간의 메시지 전달 과정

3.3 클라이언트/서버 설계 알고리즘

3.3.1 클라이언트

클라이언트가 자료의 목록을 서버에 등록하고 자료를 검색하며 자료를 송·수신할 수 있도록하는 알고리즘은 <표 6>의 Queus 방식을 이용한 목록 처리 알고리즘이 핵심적인 역할을 한다. FIFO(First-In First-Out) 스케쥴링으로 먼저 요구하는 순서대로 CPU를 할당하고 각 목록을 처리하도록 하였다.

〈표 6〉 Queue 방식의 목록 처리 알고리즘

```
public class Queue {
  protected Vector queue ;
/* Queue 생성자 */
  public Queue ( ) {
    queue = new Vector ( ) ;
  }
/* 리스트로부터 노드 제거연산 */
  public Object remove ( ) throws InterruptedException {
    synchronized ( queue ) {
      while ( queue.isEmpty ( )) { // Queue가 빈 리스트이면
```

```
        queue.wait ( ) ;  // 대기
      }
      Object item = queue.firstElement ( ) ;
      queue.removeElement(item) ;  // 제거
      return item ;
    }
  }
/* 리스트에 노드 추가연산 */
  public void add ( Object item ) {
    synchronized ( queue ) {
      queue.addElement ( item ) ;
      queue.notify ( ) ;
    }
  }
/* 빈 Queue 리스트 체크 */
  public boolean isEmpty ( ) {
    return queue.isEmpty ( ) ;  // Queue가 비어 있으면 TRUE(1),
                                아니면 FALSE(0)을 반환
  }
}
```

3.3.2 서 버

서버는 클라이언트의 접속과 자료 목록을 가지는 역할을 하며 서버에 접속한 클라이언트를 처리하는 알고리즘의 구성은 <표 7>과 같다. 서버와의 연결 시간을 30초로 두고 이 시간안에 연결이 되면 인터럽트를 발생하고 연결이 되지 않았을 경우에는 TimeOut을 발생시키고 예외처리를 한다.

〈표 7〉 서버 접속 처리 알고리즘

```
public class TcpConnection
{     public static final int defaultConnectTimeout = 30000 ; // 30
        seconds
      public static int connectTimeout = defaultConnectTimeout ;
                ...
      /* 생성자 정의 */
      public TcpConnection(TcpAddress addr) throws IOException,
            ConnectException {
                ... // 예외처리 정의
          for (long r = 0 ; r < connectTimeout ; r += 100)
          {     ...
          catch ( InterruptedException e ){ // 인터럽트는 접속을 의미
          System.out.println ( "Connection established : "
                              + peerAddr.toString ( )) ;
          break ;       }     }
          if (socket == null )
          {
            connectThread.timeOut ( ) ;
          /* TimeOut을 발생시키고, exception을 일으킨다 */
            if (connectThread.exception == null)
            {
              throw new transport.ConnectTimeoutException ( ) ;
            } else
                throw new transport.ConnectException (
                        connectThread.exception.toString ( )) ;
          }
                ...
      }
 /* 접속 부분을 쓰레드로 구현. 오랫동안 접속이 이뤄지지 않으면 접속
    을 취소 */
      private class TcpConnectThread extends Thread{ ... } ;
                ...
      public void close ( ) {
                ... // 예외처리 정의
          try{ socket.close ( ) ; }
          catch ( IOException e ){
      /* 외부에서 접속을 끊은 경우만 발생 */
                socket = null ;
            }
      }
} .
```

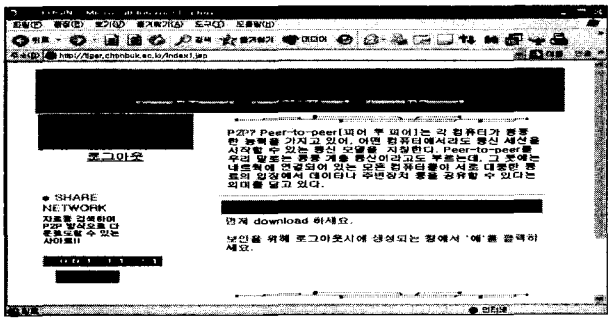
4. 자료공유 네트워크 시스템 구현

본 장에서는 구현한 자료공유 네트워크 시스템의 실행 순서에 따라서 서버와 클라이언트의 순으로 살펴보도록 하겠다.

4.1 서 버

4.1.1 사용자 가입 및 인증 구현

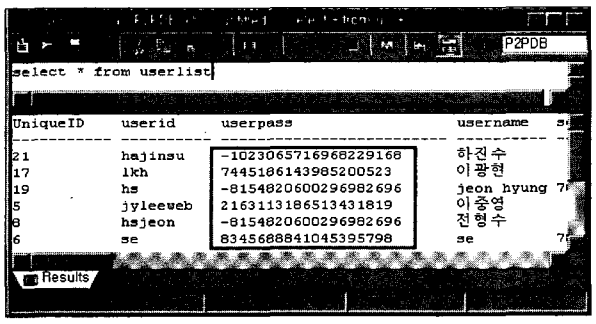
(그림 15)는 사용자 가입 및 인증을 할 수 있도록 JSP로 구현한 홈페이지로 사용자의 정보입력을 하고 서버의 DB에 사용자의 ID 및 패스워드가 전송되어 로그인된 화면을 보여준다.



(그림 15) 로그인 성공

4.1.2 암호화 알고리즘 구현

입력된 비밀번호를 암호화해서 DB에 저장할 때는 암호화된 데이터가 저장되기 때문에 다른 사용자들이 DB에 접근하여 사용자의 ID를 알아내어도 패스워드를 알아낼 수 없는 강력한 기능을 가진 클래스를 구현하였다. (그림 16)은 암호화된 패스워드가 DB에 저장된 내용을 보여주고 있다.



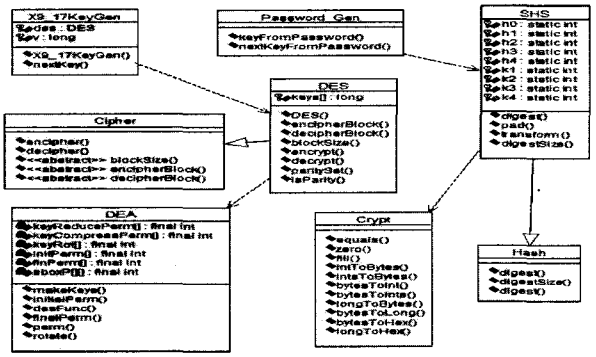
(그림 16) 암호화되어 DB에 저장

<표 8>은 인터페이스와 각 클래스와 슈퍼 클래스를 나타낸 것으로 어떤 기능을 담당하고 있는지를 나타내고 있다.

〈표 8〉 암호화 클래스부 구성

클래스 이름	기 능
DES	상위 수준의 인터페이스
Cipher	모든 암호화 클래스의 수퍼 클래스
X9_17KeyGen	X9.17 키 생성기 표준안을 구현한 클래스
Password_Gen	패스워드 생성기 클래스를 구현
SHS	보안 해쉬 암고리즘을 구현한 클래스
Hash	모든 해쉬 함수용 클래스의 수퍼 클래스
Crypt	암호화에 사용되는 기본적인 함수들의 집합
DEA	하위 수준의 인터페이스

(그림 17)은 <표 8>에서 언급된 암호화 클래스의 연관관계를 클래스 다이어그램을 나타낸 그림이다.



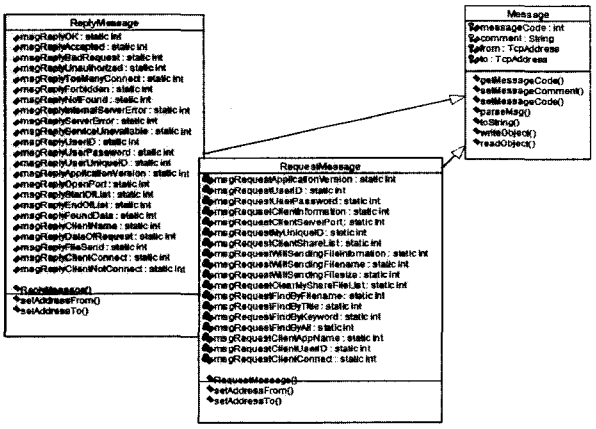
(그림 17) 암호화 클래스 다이어 그램

4.1.3 자료공유 네트워크 프로토콜

요구메시지와 응답메시지를 처리해 주는 클래스를 구성한 표이다.

〈표 9〉 메시지 클래스 세부 구성

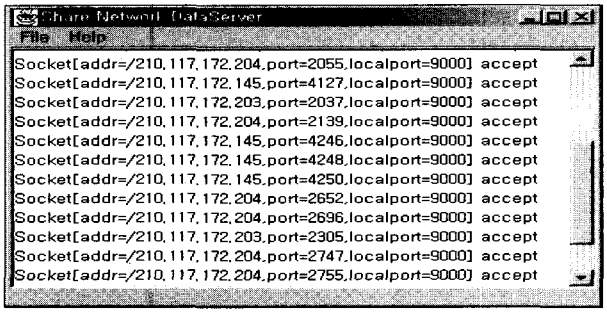
클래스 이름	기 능
Message	요구와 응답을 구현하는 클래스의 상위 클래스
RequestMessage	요구를 담당하는 클래스
ReplyMessage	응답을 담당하는 클래스



(그림 18) 메시지 클래스 다이어그램

<표 9>의 메시지 클래스 세부 구성을 보면 최상위에는 메시지가 있고 하위에 요구를 담당하는 요구메시지와 응답을 담당하는 응답메시지가 있음을 볼 수 있다. 이들의 관계를 표현한 (그림 18)의 메시지 클래스 다이어그램을 보면 더욱 명확해진다.

(그림 19)는 서버에 접속한 클라이언트의 IP 주소와 포트, 그리고 접속 상태를 확인할 수 있도록 하는 서버 관리자용 프로그램이다.



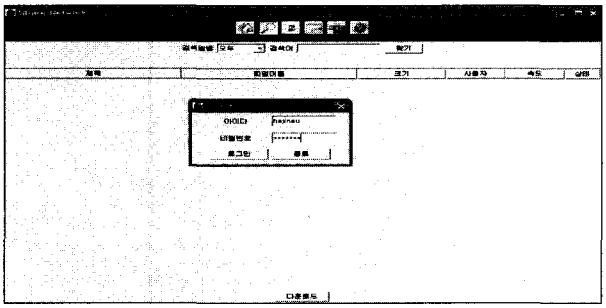
(그림 19) 서버에 접속한 클라이언트 리스트

4.2 클라이언트 프로그램 구현

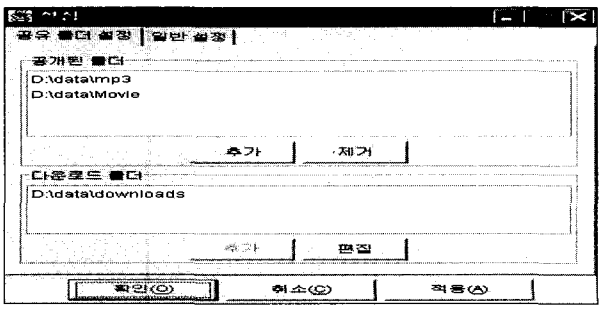
4.2.1 프로토콜 동기화 및 접속 구현

프로토콜의 동기화는 중요한 문제중의 하나이다. 서버와 클라이언트간에 자료의 원활한 전송이 이루어지게 하는 것으로 다수의 클라이언트가 접속하였을 경우 상황에 맞게 대처하도록 하는 프로토콜의 동기화 및 접속 제한 기능을 추가하였다.

(그림 20)은 클라이언트 프로그램을 구현한 클라이언트 프로그램의 초기화면이다.



(그림 20) 클라이언트 프로그램 초기 화면

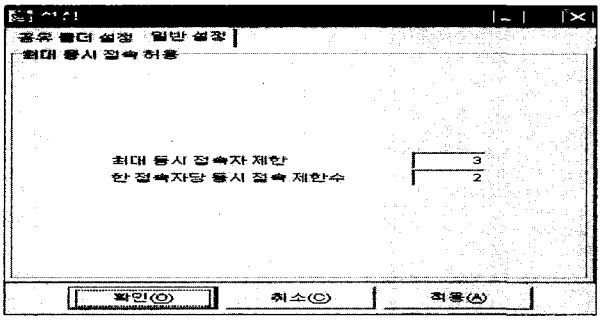


(그림 21) 클라이언트 설정화면 1

(그림 21)은 클라이언트가 다른 클라이언트와의 데이터 교환을 위한 공유 폴더와 다운로드 폴더를 설정하는 부분을 구현한 화면이다.

4.2.2 접속자 제한 설정 구현

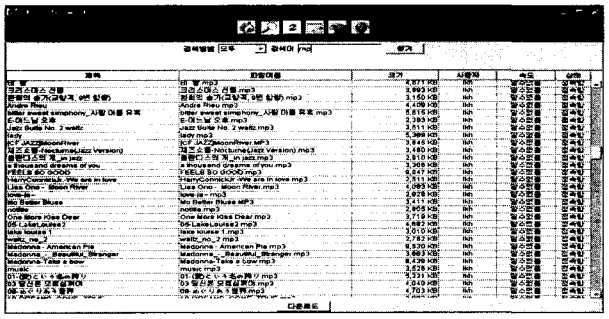
(그림 22)는 최대 동시 접속자 제한을 두고 있으며 한 접속자당 동시 접속 제한 수를 두어서 자신의 컴퓨터가 과부하에 걸리지 않고 다른 사용자에게 서비스할 수 있도록 한다.



(그림 22) 클라이언트 설정화면 2

4.2.3 미디어 자동 검색 기능 구현

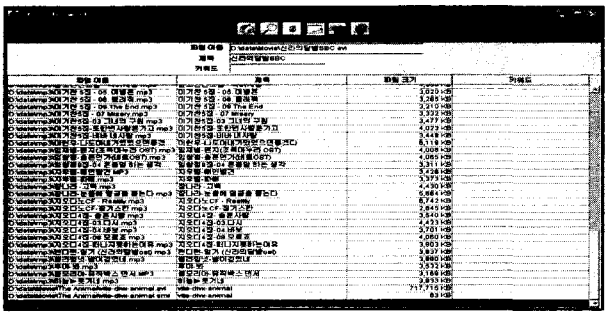
(그림 23)는 미디어 자동 검색 기능으로 질의가 들어 왔을 때, 복합적인 검색 기능을 구현하는 것으로 키워드, 제목, 내용별, 파일 이름 등의 정보를 이용하여 검색을 한다.



(그림 23) 미디어 자동 검색

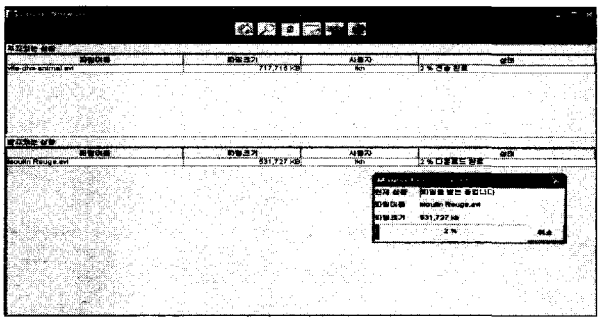
4.2.4 P2P 방식을 이용한 1 : 1 자료 송·수신 기능 구현

P2P 방식을 통한 1 : 1 자료 송·수신을 나타내는 기능을 적용하여, 서버에서 인덱스 정보를 찾은 후 원하는 데이터 정보를 상호간에 주고받는 상태를 나타낸다. (그림 24)는 자신의 공유된 데이터 목록이 나타난 화면으로, 파일이름, 제목, 파일크기 등의 목록이 나타난다.



(그림 24) 공유 목록

(그림 25)는 파일을 주고있는 상황을 나타내는 전송부분과 받고 있는 상황을 나타내는 다운로드 부분으로 전송 상태와 다운로드 상태를 확인할 수 있다.



(그림 25) 전송 및 다운로드

4.3 실험 결과

서로 다른 시스템에서의 호환 여부를 판단하기 위하여 테스트 한 결과를 나타낸 것으로 컴퓨터의 시스템 성능이 낮은 가운데에서도 클라이언트 프로그램이 사용될 수 있다.

4.3.1 시스템 실험 환경

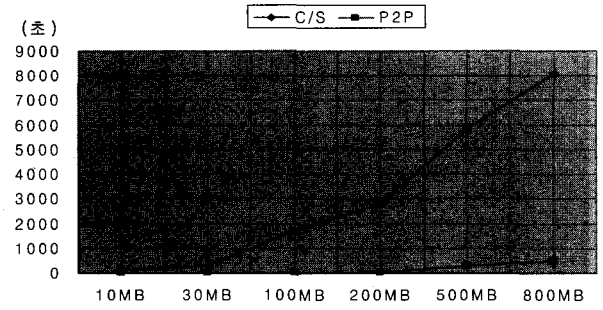
서버 프로그램을 구축한 환경은 다음과 같다. 시스템 사양을 보면 Pentium Ⅲ 800MHz, 256 RAM, 운영체제는 Windows 2000 Professional, XP에서 사용하였고, DB는 MS-SQL 7.0을 사용하였다. 구현 언어는 Java, 개발 툴로 JBuilder 4.0을 사용하였고 웹페이지 구현을 JSP를 사용하였다. JSP의 실행을 위해 웹서버 엔진은 경량 서버를 돌리기에 알맞은 resin 1.1.3을 선택하였다.

클라이언트 프로그램을 실행한 환경은 다음과 같다. 시스템은 Pentium Ⅰ, Ⅱ, Ⅲ 사양을 가진 컴퓨터를 모두 이용하였고, 운영체제는 Windows 95, Windows 98, Windows 2000, Windows 2000server, Windows XP, Linux에서 실험을 하였다.

4.3.2 실험결과 및 분석

본 논문에서 실험환경은 시스템 운영체제 Windows 계열과 Linux에서 클라이언트 프로그램을 실행하였다. 그 결과 운영체제와 무관하게 클라이언트 프로그램이 수행되었다. 또한 사용자로 하여금 사용자 시스템 환경에 맞추어 시스템에 접근할 수 있는 사용자 수를 지정하여 시스템의 부하를 최소화하면서 서로의 정보를 1 : 1로 교환할 수 있도록 하였다. 본 실험에서는 사용자 환경의 기본 값인 최대 동시 접속자 수 3, 한 접속자당 동시 접속 수 2로 하여 실험하였다. 한 클라이언트에 최대 동시 접속자 수를 3명까지 해서 실험한 결과 클라이언트 접속 대역폭에 따라 차이가 나는

것을 볼 수 있었다. 접속하는 환경에 따라 클라이언트 환경을 적절히 설정해야 데이터의 교환이 원활하게 이루어지는 것을 확인하였다. 그리고 A사의 클라이언트/서버(이하 C/S) 방식을 이용한 다운로드 보다 P2P 방식의 다운로드가 빠르다는 간단한 실험을 통해 결과를 얻었다. 즉, 같은 용량의 데이터를 C/S 방식과 P2P 방식을 사용하여 다운로드한 실험이다. (그림 26)을 보면 C/S 방식과 P2P 방식을 사용했을 때, 시간지연의 차이가 크게 나타나는 것을 확인할 수 있다.



(그림 26) C/S 와 P2P 방식의 다운로드 시간 비교 그래프

C/S 방식은 대용량의 데이터를 받을수록 시간의 지연이 심해지는 것을 그래프와 <표 10>을 보면 알 수 있다. 이에 비해, 같은 용량의 데이터를 다운로드하는 P2P 방식은 훨씬 적은 시간이 사용된다는 것을 볼 수 있다. 이것은 C/S 방식의 약점인 서버의 과부하를 P2P 방식을 사용하여 검색시에는 서버를 사용하지만 실제 다운로드시에는 서버를 사용하지 않으므로 서버의 부하는 분산하는 단적인 예라고 할 수 있다.

〈표 10〉 C/S와 P2P 방식의 다운로드 시간

방식 / 용량(MB)	C/S	P2P
10M	126초	5초
30M	288초	10초
100M	1683초	15초
200M	2822초	60초
500M	5802초	300초
800M	8075초	480초

실험한 결과 가장 큰 특징으로는 개발한 클라이언트 프로그램이 Java로 설계한 프로그램이기 때문에 Linux 환경에서도 충분히 구동된다는 특징과 함께 Windows 환경에서만 구동되는 소리바다에 비해 운영체제에 상관없이 작동하는 자료공유 네트워크 시스템만의 특징을 볼 수 있었다. 소리바다의 경우 검색 시 접속이 안 된 클라이언트의 목록은 검색되지 않으며 모든 클라이언트들에게 일방적으로 목록을 보내 주기 때문에 시간이 많이 걸린다는 단점이 있다. 하지만, 자료공유 네트워크 시스템은 검색 시 접속이 안 된 클라이언트의 목록까지 저장하고 있어서, 차후에 접속이 이루어졌을 때 빠르게 데이터를 전송 받을 수 있는 이점이 있다. 그리고, 전체 클라이언트에게 일방적으로 목록을 보내 주지 않고 목록을 필요로 하는 클라이언트에게만 보내주므로 빠른 검색이 이루어진다는 장점이 있다.

<표 11>은 P2P 기술로 구현된 대표적인 시스템(A, B, C, 자료공유 네트워크)들을 비교 분석한 것으로 각 시스템들을 검색 및 검색 방식, 서버 역할, 검색 대상 수에 따른 관점에서 장·단점을 기술하였다.

5. 결론 및 향후 연구 방향

최근 인터넷 서비스가 급증하면서 가정용 홈 네트워킹에 대한 관심이 커지고 있어 정보가전 환경에서의 인터넷 홈서버 활용에 많은 관심이 집중되고 있다. 그러나 아직은 인터넷 활용이 정보가전의 일부분까지 활용할 수 있는 수준은 아니다. 따라서 본 논문은 네트워크에 홈서버와 가정용 PC를 1 : 1로 연결할 수 있도록 하여 상호 데이터를 웹상에서 공유하게 하고 각종 정보를 교환하는 교육용 웹서비스 시스템을 구축하였다. 구현된 결과물은 크게 세 가지로 나눌 수 있다.

첫째, 서버 측의 웹사이트 구축과 각종 웹 기반의 기능을 수행할 수 있도록 사용자 DB와 교육용 DB 스키마를 구축하였다. 둘째, 서버 관리자용 프로그램을 구현하여 실시간으로 클라이언트의 접속 상황을 모니터링 할 수 있도록 하였다. 셋째, 클라이언트 프로그램 구현으로 프로토콜 동기화 및 접속, 검색 기능 구현 그리고 교육용 미디어 자동 검색 기능 구현을 통해 클라이언트간 1 : 1 자료 송·수신 기

〈표 11〉 A, B, C, 자료공유 네트워크 비교

종류 / 구분	A	B	C	자료공유 네트워크
검 색	TCP(connection)	TCP(connection)	UDP(connectionless)	TCP(connection)
검색 방식	서버에서 검색	Search word forwarding	Search word broadcasting	서버에서 검색
서버 역할	파일 리스트 관리 및 검색	N/A	사용자 리스트 관리	파일 리스트 관리 및 검색
검색 대상 수	서버의 용량과 직결 (7~8천명)	현재 네트워크 및 TTL (Time to live)과 직결	서버에서 그룹핑 (현재 7000 여명)	서버에서 그룹핑
장 점	• 안정적인 서비스 • 빠른 검색 속도 • 서버 확장	• 완전분산형(Decentralized) • 익명성 보장	• 대부분의 사용자가 초고속 인터넷 사용자 • 서버 확장	• 빠른 검색 속도 • 서버 확장
단 점	• 많은 서버를 요구함 • 인덱싱에 대한 법적 책임	• Not user-friendly • 느린 검색 속도 • 병목 현상	• 방화벽 뒤에서 사용 불가	• 방화벽 뒤에서 사용 불가

능을 구현하였다. 본 논문에서는 P2P 방식을 이용하여 서버의 과부하를 줄이며 사용자의 질적 서비스 향상에 노력하였다. 그리고 각 클라이언트들에 접속된 커넥션 수와 대역을 고려해 클라이언트에 접속자 수를 제한할 수 있는 설정부분을 두어 부하를 조절하도록 하였다. 또한, 기존의 P2P 응용 프로그램에서는 볼 수 없는 기능으로 클라이언트가 접속중인 상태에서만 검색이 가능했던 점을 보완하여 클라이언트가 접속중이지 않더라도 클라이언트의 자료목록을 서버에 저장하여 상대 클라이언트가 검색이 가능하게 하여 자료를 가진 클라이언트가 접속하였을 때 또 다른 검색없이 자료를 다운로드 할 수 있도록 하였다. 자료공유 네트워크 시스템은 Java를 이용하여 개발하였기 때문에 클라이언트의 플랫폼에 상관없이 독립적으로 실행할 수 있는 커다란 장점을 가지고 있다. 그리고, 클라이언트 프로그램을 구성하는 각 모듈은 다른 프로그램에서 재사용이 가능하다. 실험을 위해 대역폭의 환경이 좋은 곳에서 가상 실험을 하였기 때문에 무리 없는 실험이 진행되었지만, 사용자 수가 증가하였을 경우에도 파일의 송·수신이 진행되는 실험이 향후 연구되어질 것이다. 아직은 파일공유의 용도로 이용되는 P2P 기술이 인터넷 서비스의 주류로 부상할 것이 예상되지만 거미줄 형태로 얽혀 있는 수천·수만의 개인 PC에 서비스가 집중되어 있고, 서버는 단순 작업만을 수행하기 때문에 보안 대책 준비가 필요하다.

참 고 문 헌

[1] Andy Oram, "PEER-TO-PEER," O' Relly, 2001.

[2] 이재규, "신세대 네트워크의 키워드 'P2P'의 힘", 마이크로 소프트웨어, 2000.

[3] Endeavor Technology, "Introducing Peer-to-Peer," 2000, http://www.peer-to-peerwg.org/.

[4] R. Gold, C. Mascolo, "Use of context-awareness in mobile peer-to-peer networks," Future Trends of Distributed Computing Systems, 2001. FTDCS 2001. Proceedings. The Eighth IEEE Workshop on, pp.142-147, 2001.

[5] T. Iwao, Y. wada, S. Yamasaki, M. Shiouchi, M. Okada, M. Amamiya, "Framework for the next generation of e-commerce by peer-to-peer contact : Virtual Private Community," Enabling Technologies : Infrastructure for Collaborative Enterprises, 2001. WET ICE 2001. Proceedings, Tenth IEEE International Workshops on, pp.340-341, 2001.

[6] G. Reif, E. Kirda, H. Gall, G. P. Picco, G. Cugola, P. Fenkam, "A Web-based peer-to-peer architecture for collaborative nomadic working," Enabling Technologies : Infrastructure for Collaborative Enterprises, 2001. WET ICE 2001. Proceedings, Tenth IEEE International Workshops on, pp.334-339, 2001.

[7] A. Bakker, M. van Steen, A. S. Tanenbaum, "A law-abiding peer-to-peer network for free-software distribution," Network Computing and Applications, 2001. NCA 2001. IEEE International Symposium on, pp.60-67, 2001.

[8] S. Yamaguchi, K. Maruyama, "Autonomous load balance system for distributed servers using active objects," Database and Expert Systems Applications, 2001. Proceedings. 12th International Workshop on, pp.167-171, 2001.

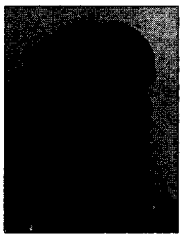
[9] http://www.napster.com/.

[10] http://gnutella.wego.com/.

[11] M. Portmann, P. Sookavatana, S. Ardon, A. Seneviratne, "The cost of peer discovery and searching in the gnutella peer-to-peer file sharing protocol," Networks, 2001. Proceedings. Ninth IEEE International Conference on, pp.263-268, 2001.

[12] http://www.soribada.com/.

[13] Merlin Hughes, "JAVA Network Programming," Michael Shoffner, 2000.



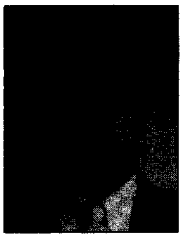
이 광 현

e-mail : khylee@cs.chonbuk.ac.kr

2001년 서남대학교 전자공학과 졸업(학사)

2001년~현재 전북대학교 대학원 컴퓨터정보학과(석사과정)

관심분야 : 소프트웨어공학, 멀티미디어, 이동 컴퓨팅 등



전 형 수

e-mail : hsjeon@cs.chonbuk.ac.kr

1992년 전북대학교 수학과 졸업(이학사)

1997년 전북대학교 대학원 전산통계학과 졸업(이학석사)

1997년~현재 전북대학교 전산통계학과 (박사과정)

관심분야 : 멀티미디어, 실시간시스템, 네트워크, 정보보호 등

유 철 중

e-mail : cjyoo@moak.chonbuk.ac.kr

1982년 전북대학교 전산통계학과 졸업 (이학사)

1985년 전남대학교 대학원 계산통계학과 졸업(이학석사)

1994년 전북대학교 대학원 전자계산학과 졸업(이학박사)

1982년~1985년 전북대학교 전자계산소 조교

1985년~1996년 전주기전여자대학 전자계산과 부교수

1997년~현재 전북대학교 자연과학대학 컴퓨터과학과 조교수

관심분야 : 소프트웨어공학, 에이전트공학, 컴포넌트기술, 분산객체기술, 지리정보시스템, 멀티미디어, 인지과학 등

장 옥 배

e-mail : okjang@moak.chonbuk.ac.kr

1966년 고려대학교 수학과 졸업(이학사)

1973년 고려대학교 교육대학원(교육학석사)

1980년 죠지아 주립대(박사과정수료)

1987년 산타바바라대학교 졸업(Ph.D)

1990년~1991년 영국에딘버러대학교 객원교수

1980년~현재 전북대학교 공과대학 전자·정보공학부 교수

관심분야 : 소프트웨어공학, 전산교육, 수치해석, 인공지능 등