P2P기반 인터넷 부하 분산 시스템 구축

이정환*
*배재대학교

Network load balancing and system construction of the Internet system which is based on P2P.

Jung-hwan Lee*
*PAI CHAI University
E-mail : p0214605@backsan.pcu.ac.kr

요 약

P2P (Peer to Peer)는 최근 인터넷에서 많이 논의 되고 있고 관심을 가지는 부분이다. P2P는 기존의 클라이언트-서버 중심의 구조를 서버가 존재하지 않는 클라이언트간의 정보공유 구조 즉 P2P의 구조로 변화 시켰다. 하지만 완벽한 P2P 구조는 아직 넘어야할 문제가 많이 존재하고 있다. 본 논문에서는 서버와 클라이언트간의 통신 사이에 발생하는 네트워크 부하를 여러 대의 피씨로 부하를 분산하고 클라이언트 간에 자원의 공유와 CPU의 공유를 통하여 분산된 컴퓨터들을 하나로 묶고 여러 대의 클라이언트를 하나로 묶어서 커다란 하나의 가상서버와 커다란 하나의 가상클라이언트를 구성하여 네트워크통신을 가능하게 한다. 순수한 P2P의 구조를 구현하는 것이 주된 목적이 아니라 인터넷 시스템에서 P2P를 기반으로 하는 분산된 컴퓨팅을 구현하여 P2P에서의 네트워크의 분산, 유도로 차세대의 핵심인 P2P 인터넷 시스템을 설계 및 구현하고자 합입니다.

ABSTRACT

P2P (Peer to Peer) is part that is discussed much in the latest Internet and is interested. P2P changes the structure of Client-Server into Information sharing between client that is the structure of p2p. But perfect P2P system has many problems awaiting solution. In this treatise we can realize network communication to compose one virtual server and one virtual client which is intergrated several client and distributed computer that decentralize network traffic into the traffic of several pc by sharing resource and cpu between client. Our main purpose is not real P2P system but realization,organization of the P2P internet system which is composed by distributed computing and network load balancing in P2P internet system environment.

키워드
P2P, 분산컴퓨팅, 가상서버, 가상클라이언트

Ⅰ. 서 론

지금까지 네트워크상에서 자료를 주고받거나, 정보를 주고받을 때에는 전형적인 방식인 클라이언트-서버 중심의 방식으로 수행되어 왔다. 이러한 방식은 클라이언트의 수가 증가하면 할수록, 서버에 대한 부하가 증가하여 나중에는 클라이언트가 서버에 접근을 할 수 없게 되는 상황에 이른다. 인터넷에서 접속되어 있는 사용자들이 네트워크에서 자료나 정보를 검색하기 위해서는 기존에 서비스가 되고 있는 웹 사이트의 서버를 통해서만 가능했다. 그렇게 하다보니 서버의 부하는 증가하고, 사용자들은 정보 검색에 대한 만족도는 떨어지게 되었다. 이런 단점을 극복하기 위한 방법으로 여러 가지 대안이 나오게 되었는데, P2P는 이러한 대안들 중에 하나이다.

P2P의 특징은 하나의 서버에 의존하는 것이 아니라 네트워크상에 존재하는 여러 대의 서버가 아

닌 PC 즉 피어(Peer) 에게 분산되어, 네트워크에 접속된 모든 피어를 사용하여 자원의 사용을 최대한으로 이끌어내고 높은 자원의 가용성을 제공한다.

II. 시스템 설계

2.1 전체적 구성

본 논문에서 P2P형 인터넷 시스템은 여러 곳에 분산된 PC기반의 자원들과 이 자원들을 통하여 연산기능을 처리할 수 있도록 문제를 제공하는 Provider, 이모든 자원들을 관리할 수 있는 Server로 구성되어 있다. 즉 시스템 전체적, 대략적으로 보면 Server, Provider 및 Client로 구성되어 있다. Server는 메인운용자, Provider는 응용프로그램 개발자이고 Client는 일반사용자로서 CPU의 시간을 Provider에게 제공 및 기타 자료를 공유시켜준다.

표1. 시스템 구성의 설명

구분	내용
Server	◆ 시스템 전체를 관리하는 모듈 ◆ 작업의 스펙을 제공 : provider에게 작업 스펙을 제공 ◆ 메시지 처리 : provider/client와의 통신에서 메시지 처리
Client	◆ 서버에 자신의 응용프로그램을 적용하는 응용분야의 고객들이 사용하는 서버 프로그램 ◆ 데이터 리소스 관리 : 지정된 작업용 데이터에 대한 인덱싱을 수행하고 내용을 찾을 수 있도록 관리 ◆ 데이터 제공 : provider가 작업을 요청할 경우 작업 데이터를 제공 ◆ 결과저장 : provider가 작업한 작업결과를 업로드 할 수 있도록 포트를 열어주고 작업 결과를 저장
Provider	◆ 일반 사용자들의 PC에 설치되어 서버로부터 작업을 할당 받아 수행하고 작업결과를 보고할 수 있는 프로그램 ◆ PC사양 분석 ◆ 작업 다운로드 및 업로드 : 서버와 통신하여 작업을 다운로드하고 결과를 업로드 함

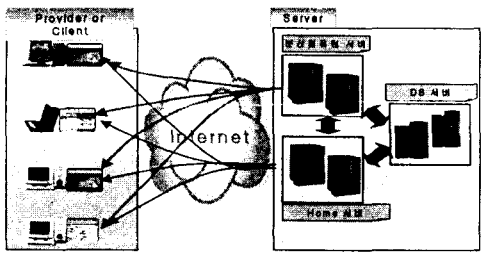

그림 1. 전체적 구성도

2.2 논리적 시스템 구성

전체적 구성을 좀더 논리적으로 구성해서 본다면 Project Work Manager, Database, Data Server, Scheduling Server 및 Agent로 구성된다. Project Work Manager에서 모든 Data에 대한 상태를 파악하고 Data Server에서는 Data에 대한 처리를 하며 Scheduling Server 에서는 Agent에 대한 모든 일들을 순서적으로 관리한다. 이 모든 구성 요소들이 유기적으로 관계를 맺으면서 시스템을 구성한다.

표2. 모듈의 기능

구분	내용
Project Work Manager	◆ application을 제공한다. ◆ Work units(작업 할당량)들을 제공한다. ◆ 컴퓨터 사용 결과들을 처리한다. ◆ 클라이언트를 관리한다. ◆ 프로바이더를 관리한다.
Data Server	◆ input & output file을 분배한다. ◆ Data File을 Download한다. ◆ Data File을 Upload한다.
Scheduling Server	◆ 에이전트에게 Data를 분배한다. ◆ 에이전트에게 있는 Data를 수집한다.

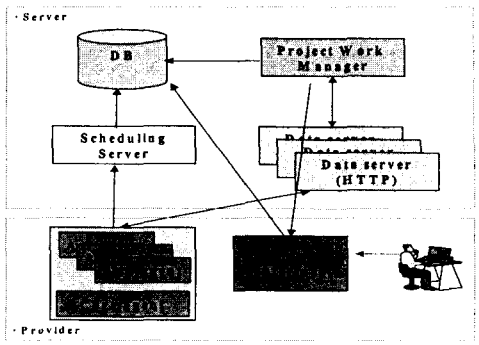

그림 2. 논리적 시스템 구성도

III. 시스템의 상세구조

3.1 서버구조

가. 자원제공 PC관리

(1) 등록

효율적인 어플리케이션 관리와 유효한 사용자 인증을 위해 관리자는 사용자에게 고유한 아이디

어를 제공하여, 인증 과정 시에 사용 한다.

절차로서는
▶ 웹 페이지를 통한 사용자 정보를 등록
▶ 에이전트 다운로드 및 설치
▶ 서버에서 사용자 고유 아이디어 생성 및 메일을 발송
▶ 서버에서 사용자 등록 정보와 고유 아이디어를 DB 에 저장
▶ 사용자 고유 아이디어와 어플리케이션 아이디어를 링크한다.

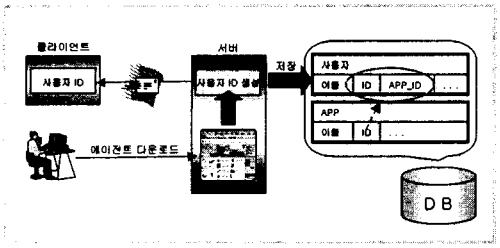

그림 3. 등록

(2) 삭제

클라이언트의 원활한 관리를 위하여 관리자는 유효하지 않은 클라이언트의 고유 아이디어 및 사용자 정보를 삭제할 수 있다. 또한, 자원 제공을 원하지 않는 사용자는 자신의 정보를 삭제할 수 있다.

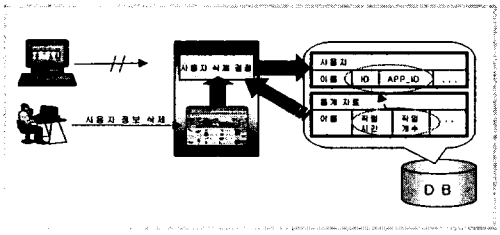

그림 4. 삭제

(3) 정보수집

관리자는 작업 할당 및 통계 자료 산출을 위하여 사용자 PC의 정보가 필요로 한다. 사용자 PC 리소스 정보에는 CPU속도, OS종류, 메모리, 디스크 공간, 네트워크 종류 및 속도 등이 있다.

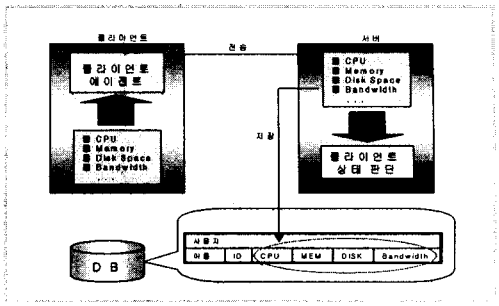

그림 5. 정보수집

(4) 상태관리

관리자는 클라이언트의 작업 상태를 모니터링을 할 수 있어야 한다. 왜냐하면 클라이언트의 작업 수행 정도에 따라서 새로운 작업의 분배 및 결과파일의 수거가 필요하기 때문이다.

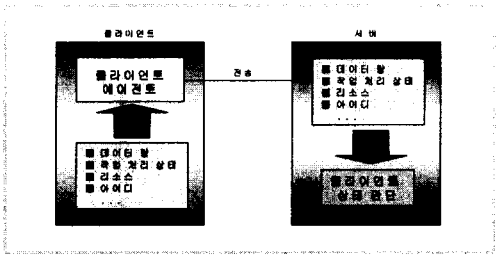

그림 6. 상태관리

나. 작업의 관리

(1) 작업의 생성

단위 작업 및 결과물에 관한 정보를 작업 스펙/결과 스펙에 기재하여 응용에서는 이러한 스펙을 가지고 다른 작업을 생성 및 제공을 하게 된다.

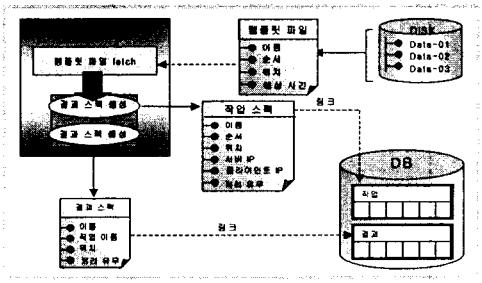

그림 7. 작업의 생성

(2) 작업의 분배

작업을 분배하는 목적은 서버에서 작업 가능한 자원 제공PC (즉, 클라이언트)에게 작업을 할당하고, 완료된 결과물을 수거하는 것이다. 에이전트가 적절한 작업을 수행하기 위한 필요한 작업 데이터의 분배는 에이전트의 현재상태를 고려한 후 이루어져야 한다.

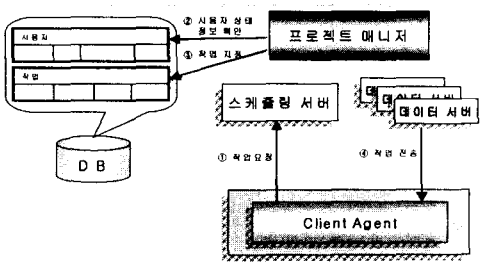

그림 8. 작업의 분배

작업의 분배는 스케줄링 서버에서 이루어진다. 스케줄링 서버는 요청 작업 우선순위를 부여하여 모든 작업을 처리한다.

작업조건은

- 먼저 들어온 요청을 먼저 처리하고, 나중에 들어온 요청을 나중에 처리(default)한다.
- 특정 레벨 이상의 클라이언트의 동시 요청 메시지 처리를 위해 서버에서 각 자원 제공 PC 리소스 정보와 작업 처리 히스토리정보 등을 토대로 우선순위를 부여한 후 요청된 작업을 처리한다.
- 다수 메시지의 올바른 처리를 위해서 FSM (Finite State Machine)기법을 사용한다.

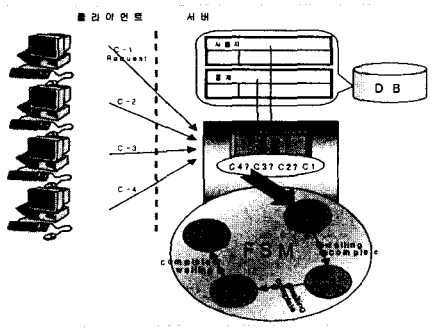

그림 9. 스케줄링 기법

(3) 작업의 결과 수거

전체 작업 결과를 만들기 위해 서버는 단위 작업을 자원 제공 PC에 제공한다. 자원 제공 PC의 에이전트로 전송된 단위 작업을 클라이언트에서 처리한 후, 일정 레벨의 작업이 완료되면 서버는 완료된 결과물을 수거한다.

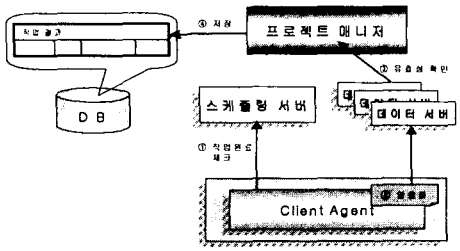

그림 10. 작업의 수거

Ⅳ. 결 론

본 논문에서는 P2P를 이용하여 피어에게 CPU의 공유를 통한 작업을 수행하는 과정을 살펴보았다. 에이전트의 인스톨을 시작으로 스케줄링을 사용하여 작업의 생성, 분배, 수거를 처리했다. 아직은 미흡하지만 좀더 연구한다면 더 낫은 결과를 얻을 수 있을 것이다.

참고 문헌

[1] JXTA, Bredon J. Wilson, LIFE & POWER PRESS
[2] http://conferences.oreillynet.com/p2p/
[3] http://www.jxta.org/
[4] http://jxtakorea.net/