인터넷에서 동시 파일 전송을 효과적으로 지원하는 NFTP의 설계 및 구현

정회원 최 재 남*

Next-Generation File Transfer Protocol which support concurrent file transmissions effectively in Internet

Jae-Nam Choi* *Regular Member*

요 약

현재 인터넷 FTP(File Transfer Protocol)는 그 범용성과 안정성에도 불구하고 구조적인 문제로 인하여, 동시 다중 파일 송수신이 불가능하여 파일을 전송하려면 현재 진행중인 전송이 종료될 때까지 대기해야 하는 불편함이 있었다. 그리고 n(n>1)개의 파일을 동시에 송수신 하려면 사용자 측 PC와 서비스 제공자 호스트(HOST)에 n개의 클라이언트와 서버를 각각 띄워야 한다. 따라서, 메모리 및 네트워트 자원의 낭비와 허용 할 수 없는 시스템 부하를 초래하는 위험성까지 내포하고 있다. 앞서 기술한 이와 같은 문제를 해결하기 위하여 본 논문에서는 시스템부하를 최소화하고 효과적으로 동시에 다중 파일 송수신이 가능하도록 멀티쓰레드를 기반으로 한 NFTP(Next-Generation FTP)구조를 설계하였고, NFTP 프로토콜을 구현하였다. 본 논문에서 설계한 NFTP를 구현하여 실제 서비스에 적용한 결과, 이용자의 파일 전송 종료 대기 시간을 대폭 줄일 수 있었을 뿐만 아니라 시스템 부하도 적게 주어서 안정적으로 서비스를 제공할 수 있었다.

ABSTRACT

Though the FTP(File Transfer Protocol) has been used widely and stable, It has the structural weakness that can't support current file transmissions so that we have to wait the completion of previous file transmission when try to transmit another file. If try to transmit multiple files concurrently using this FTP, It has to forking multiple FTP servers and clients in each user's PC and ISP's host machine it would result in the waist of memory , resource of network and the high workload of system. In order to solving previous problem, in this paper I have designed the new model of FTP which based on multi-thread and created NFTP(Next-Generation FTP)protocol so that may reduce the workload of system and support current file transmission effectively. I have implemented NFTP and also applied to real service, as a result It have provided reliable service by reducing the workload of system and saved the waiting time which would happened

Ⅰ. 서 론

인터넷에서 표준으로 자리잡은 대표적인 파일 송수신 프로토콜인 FTP(File Transfer Protocol)는 그 보편성과 안정성에도 불구하고 파일을 송수신을 하고 있는 경우, 사용자는 다른 파일들을 송수신을 하려면 이전의 파일 송수신이 완료될 때까지 기다려야만 했다. 다시 말하면 동시에 파일 송수신을 수행할 수 없는 불편함을 가지고 있다. 이는 FTP프로토콜의 설계 및 구조에서 기인한다. 즉 기존 FTP에서는 클라이언트(Client)와 서버(Server)간의 파일 송수신 과정에서 파일 송수신 관련 명령어와 이 명

* 한국통신 하이텔 연구소(unclee@hitel.net),
논문번호 : 010132-0530, 접수일자 : 2001년 5월 30일
※ 본 연구는 2000년 4월 21일에 서비스를 개시한 HiTEL2000에 구현 및 적용되었습니다.

령에 대한 응답을 주고받는 제어세션(control-connection) 한 개와 실제 파일 데이터가 송수신 되는 데이터 세션(data-connection) 한 개를 이용한다. 그리고 제어 세션은 FTP 클라이언트가 로그인하여 종료할 때까지 유지되고, 데이터 세션은 파일 송신 또는 수신할 때마다 생성 및 소멸되며 오직 한 개로 유지된다[3]. 그러므로 기존 FTP에서는 한 개로 유지되는 데이터 세션방식과 서버와 클라이언트가 단일 쓰레드(single-thread)로 구현된 구조로 인하여 동시에 파일 송신 및 수신을 수행할 수 없다.

이러한 방식에서 파일 송신 및 수신 n(n>1)개를 동시에 해야 한다면, 사용자 PC에서 클라이언트 프로그램 n개를 더 띄워야 하며, 해당 서버가 있는 호스트에도 FTP서버 n개를 더 띄워야 하므로 n번의 재접속이 일어나고 n개의 제어 세션, n개의 데이터 세션이 생성된다. 이에 따라 상기 종래의 FTP에서 파일 다중 송수신은 시스템의 메모리 및 네트워크 자원의 낭비 초래와 허용할 수 없는 시스템 부하의 발생을 유발하는 비현실적인 문제점을 가지고 있다.

Ⅱ. 본 론

서론에서 제기되었던 문제점을 해결하기 위해 본 논문에서는 , 인터넷(Internet) FTP에서 시스템 부하를 최소화하면서 동시에 파일 송신 및 수신을 할 수 있는 차세대 FTP 시스템의 설계 및 구현을 목표로 삼았다. 상기 목표를 이루기 위하여 본 논문에서 설계한 NFTP(Next-Generation FTP)의 특징은 아래와 같다.

- 멀티쓰레드(multi-thread)를 기반으로 한 FTP서버와 클라이언트의 구조 설계.
- 다중 데이터 세션(multiple data connection)과 단일 제어세션(single control connection)을 이용한 동시 다중 파일 송수신.
- 동시 다중 파일 송수신 상태에서 선택적 파일 전송 중지.

시스템의 부하를 최소화하면서 동시 파일 송수신을 지원하기 위하여 멀티쓰레드를 기반으로 하였다. 그리고 동시 파일 전송 환경 하에 각 쓰레드가 사용할 다중 데이터 세션(multi-data connection)과 단일 제어 세션(single-control connection)을 효과적으로 관리 및 이용하기 위하여 NFTP프로토콜을 만들었다[1].

1. 멀티쓰레드를 기반으로 한 FTP서버와 클라이언트의 구조 설계

그림 1은 본 논문에서 설계한 FTP 서버와 클라이언트 구조도로서, FTP 클라이언트와 FTP서버는 파일송신 및 수신을 동시에 여러 개 처리하기 위하여 멀티쓰레드로 구현하였으며, 아래와 같이 관리 및 운영된다.

- 명령/응답의 송수신은 단일 제어 세션 이용.
- 서버와 클라이언트에서 동시 처리를 위한 쓰레드 할당 및 관리.
- 서버와 클라이언트에서 할당된 쓰레드 간의 개별 데이터 세션 생성 및 파일 전송.

FTP 클라이언트가 FTP서버에 접속하여 있는 동안 제어 세션은 한 개로 유지되며, 클라이언트는 사용자의 파일 송수신 명령을 받아서 이를 동시에 처리하기 위하여 멀티쓰레드를 생성하고, 생성된 멀티쓰레드는 서버의 송수신 쓰레드에 접속하여 데이터 세션을 생성하고, 이 데이터 세션을 통하여 파일을 송수신 하게 된다.

예를 들어, n(n > 1)개 파일을 동시에 송수신하고 있다면 서버와 클라이언트의 n개의 멀티쓰레드가 1개의 제어세션과 n개의 데이터세션을 이용하여 파일 송수신을 하고 있는 것이다. 동시에 n개의 다중 파일 전송을 수행하였을 시 기존 FTP와 비교해서, 본 논문에서 설계한 NFTP 는 다음과 같은 장점이 있다.

1) 메모리 사용 절감 및 운영체제 관리 부담 감소
2) 네트워크 자원의 절약
3) 인증 및 프로세스 생성 시간 단축으로 인한 속도증대

기존 FTP는 서버 및 클라이언트 프로세스를 각각 n개 생성하였지만, 본 논문에서 설계한 NFTP에서는 각각 1개의 프로세스만 생성하면 됨으로 그만큼 메모리 절감 및 운영체제(OS)관리 부담을 경감시킨다. 기존 FTP는 제어세션을 n개 사용했지만, 본 논문에서 설계한 NFTP에서는 1개만 사용하기 때문에 그 만큼 네트워크 자원을 절약한다.

기존 FTP는 n번의 프로세스 생성 및 인증 과정을 거치지만 본 논문에서 설계한 NFTP에서는 1번만 거치기 때문에 그만큼 시간 단축으로 인한 속도 증대 효과가 있다.

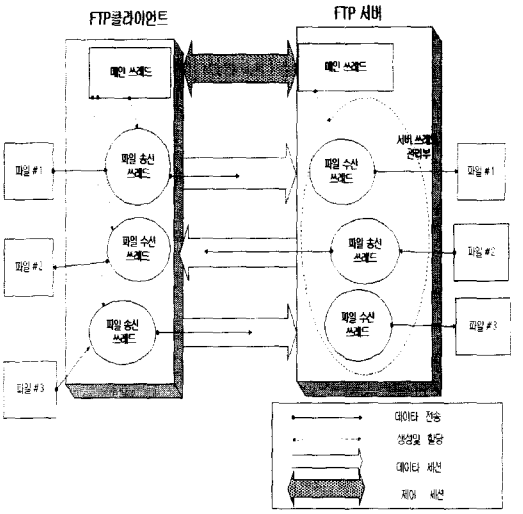


그림 1. 본문에서 설계한 NFTP의 구조

2. NFTP 프로토콜의 설계

본 논문에서 사용하는 NFTP프로토콜은 기존 FTP 프로토콜과 호환성을 지니며, 다음과 같은 점에서 구별된다. 실제 파일 송수신은 다중 데이터 세션을 이용하지만 명령을 보내고 이에 대한 응답을 보내는 제어 세션은 하나를 사용하기 때문에 명령을 발생시킨 쓰레드 ID가 각 명령 및 응답에 추가된다[2]. NFTP프로토콜에서 대표적인 명령 및 그 응답을 ABNF(Augmented Backus Naur Form)[4]으로 나타내면 그림 2와 같다.

```
size_cmd="SIZE" SP "|" thread_id ("|" file _name )*
CRLF ; 수신할 파일 크기 질의
response_size_cmd = "213" SP thread_id SP (file_size
SP)* CRLF ;파일 크기 응답
retrival_cmd = "RETR" SP "|" thread_id "|" file _name
"|" CRLF; 파일 수신 명령
store_cmd = "STORE" SP "|" thread_id "|" file_name"|"
CRLF ; 파일 송신 명령
thread_id = <10>DIGIT ; 쓰레드 ID
file_name = VCHAR* ; 파일이름
file_size = DIGIT* ; 파일크기
```

그림 2. ABNF로 표기한 NFTP 프로토콜

3. 다중 데이터 세션을 이용한 동시 다중 파일 송수신

본 논문에 따른 FTP의 파일 수신 흐름도를 그림 3에서 나타내었으며 그 동작은 다음과 같다.

① 사용자가 파일 수신 명령을 내린다.

② 이 수신 명령은 FTP클라이언트의 메인 쓰레드

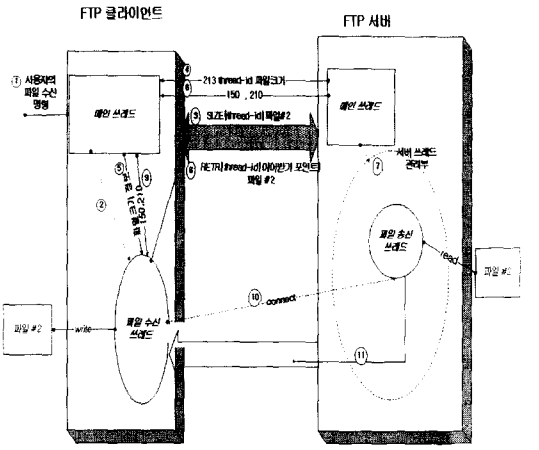


그림 3. NFTP에서 파일 수신 처리 과정

에게 전달되며, 메인 쓰레드는 이 명령을 동시에 처리하기 위하여 파일 수신 쓰레드를 생성한다.

③ 2번 과정에서 생성된 수신 쓰레드는 파일 수신에 앞서 수신할 파일들의 크기를 알기 위하여 FTP서버로 파일크기 질의 명령을 그림2의 size명령 형식으로 제어 세션을 통하여 보낸다.

④ FTP서버에서 전송한 그림 2의 size 명령응답을 FTP 클라이언트 메인 쓰레드에서 수신한다.

⑤ 클라이언트의 메인 쓰레드는 파일 크기 응답에 포함되어 있는 쓰레드 식별 번호를 추출하여 해당 파일 수신 쓰레드에게 파일 크기 정보를 전달한다.

⑥ 파일 수신 쓰레드는 수신할 파일 개수만큼 그림2의 파일 수신명령을 서버로 송신한다.

⑦ 서버의 메인 쓰레드는 이 수신 명령을 처리할 파일 송신 쓰레드를 서버 쓰레드 관리부에서 할당받는다.

⑧ 이 할당된 파일 송신 쓰레드는 파일 수신 시작 코드인 150과 NFTP프로토콜의 데이터 세션 생성 준비 응답코드인 210을 제어세션에 송신한다.

⑨ 8번 과정에서 수신한 데이터 세션 생성준비 응답을 클라이언트의 메인 쓰레드가 제어 세션을 통하여 수신하여 파일 수신 쓰레드에게 전달한다.

⑩ 파일 수신 쓰레드는 서버 접속과정을 통하여 이미 알고 있던 IP주소 및 포트번호를 이용하여 서버에 데이터 세션 생성을 위한 접속을 한다.

⑪ 10번 과정에서 생성된 데이터 세션을 통하여 서버 측 파일 송신 쓰레드가 송신한 파일을 클라이언트 측의 파일 수신 쓰레드에서 수신하게 된다.

수신 종료는 서버 측의 파일 송신 쓰레드가 데이터 세션을 닫음과 상기 파일 송신 쓰레드를 서버 쓰레드 관리부에 반납함으로써 이루어진다. 클라이언트 파일 수신 쓰레드는 파일크기 응답과 실제 수신한 파일크기를 비교하여 파일 수신의 성공여부를 판단함으로써 파일 수신을 종료한다.

그림 4는 본 논문에 설계한 NFTP에서 파일 송신 흐름도이며, 다음과 같은 순서로 동작한다.

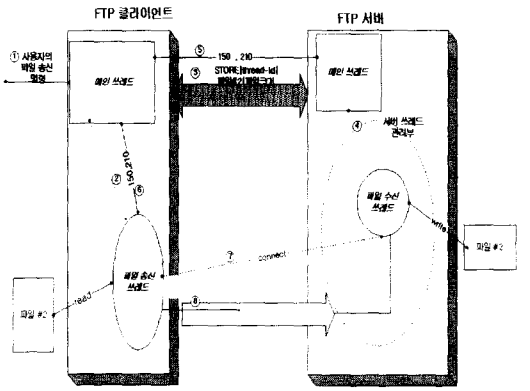


그림 4. NFTP에서 파일 송신 처리 과정

① 사용자 송신 명령어가 FTP클라이언트의 메인 쓰레드에게 전달된다.

② 메인 쓰레드는 동시에 파일 송신을 처리하기 위하여 파일 송신 쓰레드를 생성한다.

③ 2번 과정에서 생성된 파일 송신 쓰레드는 그림2의 파일 송신명령을 제어 세션을 통하여 전송한다.

④ 서버의 메인쓰레드는 이 송신 명령어를 제어 세션으로부터 수신하여 파일 수신 쓰레드를 쓰레드 관리부에서 할당받는다.

⑤ 할당된 파일 수신 쓰레드는 파일 송신 시작 코드인 150과 NFTP 프로토콜의 데이터 세션 생성 준비 응답코드인 210을 제어세션을 통하여 전송한다.

⑥ 클라이언트의 메인 쓰레드는 제어 세션을 통하여 데이터 세션 생성준비 응답을 수신하고, 쓰레드 식별번호를 추출하여 해당 파일 송신 쓰레드에게 전달한다.

⑦ 파일 송신 쓰레드는 서버 접속과정을 통하여 이미 알고 있던 IP주소 및 포트번호를 이용하여 서버에 데이터 세션 생성을 위한 접속을 한다.

⑧ 7번 과정에서 생성된 데이터 세션을 통하여 클라이언트의 송신 쓰레드가 송신한 파일을 서버의 파일 수신 쓰레드가 수신하게 된다.

파일 송신 종료는 FTP 클라이언트의 송신 쓰레드가 데이터 세션을 닫음으로써 이루어지며, 서버측 파일 수신 쓰레드는 상기 파일 송신 명령어에 포함되어 있는 송신 파일크기와 실제 파일크기를 비교하여 파일 송신의 성공여부를 판단한다.

그림 5는 실제 NFTP가 구현된 시스템에서 동시 다중 파일 송수신을 보여주고 있다.

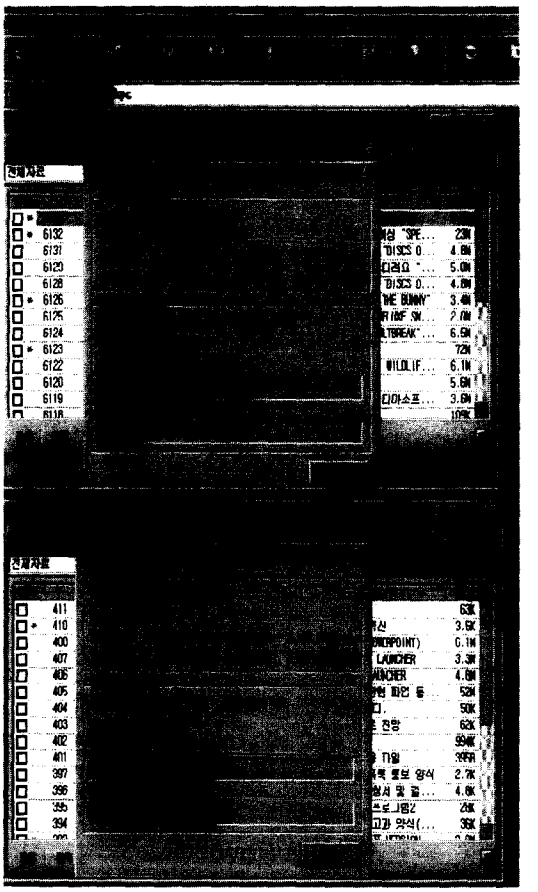


그림 5. NFTP에서 동시 파일 전송 장면

4. 동시 다중 파일 전송 상태에서 선택적 파일 전송 중지

기존 FTP는 중지명령어(abort)를 이용하여 파일 전송 중지를 하였는데, 이는 파일 송수신이 동시에 하나밖에 없는 경우만 고려하여 만들어진 것으로 동시 다중 파일 송수신에 적합하지 않았다.

그래서 본 논문에서는 설계한 NFTP는 다중 파일 송수신중 선택적 전송중지를 해당 송수신중인 쓰레드가 데이터 세션을 닫음으로써 구현하였다.

서버 측에서는 클라이언트의 파일 송수신의 중단

을 아래의 원리에 의해 알 수 있다.

송신하고 있는 파일을 중지하고 있는 경우, 서버는 그림2의 파일송신 명령어에 포함되어 있는 송신 파일크기와 실제 파일 크기가 다르기 때문에 송신의 중단을 감지하여 해당 서버의 수신 쓰레드를 쓰레드 관리부에 반납한다.

또한 수신하고 있는 파일을 중지한 경우에는, 클라이언트의 수신 쓰레드는 이전에 수신한 그림2의 파일크기 응답과 실제 수신한 파일크기가 다르기 때문에 파일 수신 중단을 감지할 수 있다.

5. 구현

NFTP 서버 및 클라이언트의 구현 환경은 아래와 같다.

- 개발 언어: C++
- 컴파일러 : SUNWorkPro 5.0,
  MS VC++5.0
- 실험장비 : SUN/E3500, 100MHZ* 4CPU,
  Solaris2.6, 4G memory
- 라이브러리 : Posix thread,
  Roguewave C++ class lib,
  MFC class lib

NFTP 구현 내용은 아래와 같다.

- 사용자 인증
- 함수객체(Functor)를 이용한 프로토콜 처리
- POSIX 쓰레드 관리 기능(thread-pool)구현
- 데이터 세션 모듈에서 임계 영역
  (critical section)관리
- 다중 쓰레드에서 신호(signal)처리

사용자 인증은 서비스 가입자에 한하여 NFTP서비스를 제공하기 위하여 , 가입자 데이터베이스에 질의를 통하여 이루어졌다.

NFTP 프로토콜의 명령어 처리 모듈은 함수객체(functor)[5]를 이용하여 서버의 프로토콜 처리속도를 개선하고, 프로그램의 가독성(readability)을 높였다.

NFTP 서버 및 클라이언트에서 멀티쓰레드 관리는 메모리 및 시스템 부하를 줄이기 위하여, 동시 파일 전송을 위한 쓰레드를 매번 생성/소멸하는 방식을 택하지 않았다. 대신 thread.h++[6]에서 제공하는 쓰레드 풀(thread pool) 관리자를 이용하여 아래와 같은 방식으로 관리하였다.

1) 메인쓰레드는 프로토콜을 해석한 후 해당 프로토콜을 처리할 함수객체 생성을 한다. 그리고 이 함수 객체를 쓰레드 풀 관리자의 큐에 넣는다.

2) 쓰레드 풀 관리자는 큐에서 함수 객체를 빼내어서, 함수객체 안에 들어 있는 함수(function)을 처리할 쓰레드를 할당(dipatch)한다.

3) 할당된 쓰레드는 함수객체 안에 있는 함수를 수행함으로써 파일 전송이 이루어진다.

데이터 세션의 생성 함수는 lock을 두어서, 각 쓰레드 간에 동기를 맞추었다.

멀티쓰레드 환경 하에서 데이터세션이 닫힐 때 발생하는 신호와 타임아웃신호는 어느 쓰레드에게 전달될지 모른다. 그래서 NFTP서버에서는 상기 신호에 대한 처리를 막는 대신 시스템 함수의 결과 값과 멀티 쓰레드에서 안정(multithread-safe)한 함수를 사용하여 해결하였다.

6. 결과 분석

본 논문에서는 NFTP와 FTP의 성능 비교를 위하여 아래와 같은 항목에 대하여 실험을 하였다.

1. 10M byte파일을 n개 수신할 경우 전송 완료 대기 시간 (0 < n< 6)
2. 4시간동안 송신 할 수 있는 파일 개수(파일크기는 10M byte)

실험 시 네트워크 트랙픽 상태는 50~70 (KB/sec) 이었으며, 전송 대기 시간 실험 결과를 가지로 비교 그래프로 나타내었다.

그림 6에서 볼 수 있듯이 전송 파일 개수가 증가할 수록 FTP는 전송 완료 대기 시간은 비례에서 증가하는 반면 , NFTP는 전송 파일 개수의 증가에 무관하게 일정한 전송 완료 대기 시간을 보여 주고 있다.

그림 7은 4시간 동안 전송할 수 있는 파일 개수에 대한 실험에 대한 비교 그래프이다.
시간이 경과됨에 따라 전송 파일 개수의 차이가 현격하게 벌어짐을 볼 수 있다.

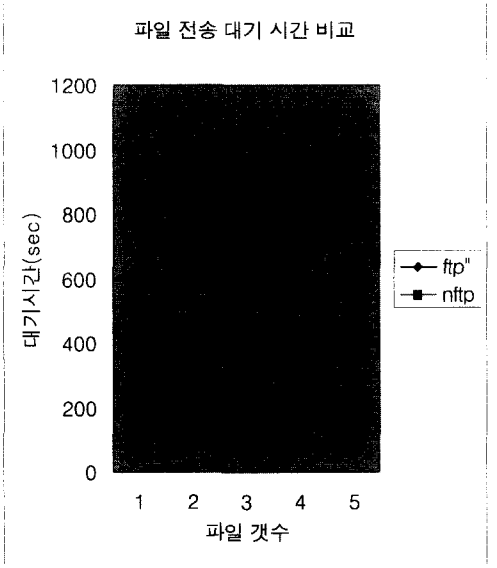


그림 6. FTP와 NFTP 전송 완료 대기 시간

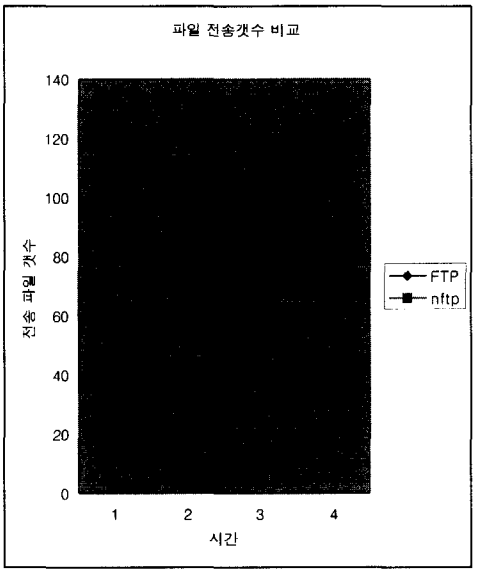


그림 7. 전송 파일 개수 비교

실험 결과를 요약하면 NFTP는 FTP에 대하여 파일 전송 완료 대기 시간 및 시간 당 전송 파일 량에 있어서 모두 성능 상 우위를 보였다.

Ⅲ. 결 론

본 논문에서는 설계한 NFTP는 동시 파일 전송을 위하여, 해당 파일 전송 쓰레드 할당과 다중 데이터 세션과 제어세션을 효과적으로 관리 및 이용할 수 있도록 프로토콜을 구현하였다.

NFTP는 기존 FTP보다 시스템 부하를 최소화하면서 파일 송수신을 안정적으로 동시 다중으로 수행할 수 있도록 설계 및 구현되었다.

실험 결과에서 볼 수 있듯이 NFTP는 기존 FTP에 비하여 파일 전송 완료 대기 및 시간 당 전송 파일 량에 있어서 모두 성능 상 우위를 보였다.

NFTP는 인터넷 파일 전송 서비스를 보다 편리하게 이용가능 하게 하는 프로토콜 및 새로운 FTP 모델이라 판단된다.

참 고 문 헌

[1] 최재남 "인터넷 파일 전송 프로토콜에서 파일 동시 송신 및 수신방법", 특허출원명세서, pp. 2-11, 2000.

[2] 최재남, 정지은, "Hitel FTP extension", H2K개발문서, pp. 10 ~24, 1999.

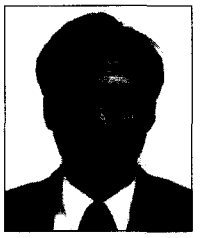
[3] J. Postel, J.Reynolds, "File Transter Protocol," RFC959, pp. 3-10, Oct 1985.

[4] D. Crocker, P. Overell, "ABNF", RFC2234, pp. 1-14, Nov 1997.

[5] Hichkey,R "Callbacks in C++ Using Template Functors", C++Report, pp. 43-50, Feb,1995.

[6] Patrick Thompson, Greg Bumgardner, "Thraed. h++ A Portable C++ Library for multithreaded Programming", C++Report, pp. 25-37, March, 1997.

최 재 남(Jae-nam Choi) 정회원



1994년 2월 : 인하대학교 전자계산공학과 졸업
1996년 2월 : 인하대학교 전자계산공학과 석사
1996년 3월~현재 : 한국통신 하이텔연구소, 선임연구원

<주관심 분야> 객체지향 개발 방법론, 분산처리, 인터넷 게임