FTP 프로젝트 설명

개발환경

Ubuntu Linux Terminal 창에서 동작하도록 구현

Command Line을 사용하여 명령이 실행되도록 할 것

개요

- 목적: 파일 업로드 & 다운로드 프로그램 작성.
- 명령어 종류는 문서 4쪽에 설명되어 있음.
- 각 쪽의 요구사항들을 자세히 읽고 모두 만족시켜야 함.
- Client는 한번 구현하면 모든 단계에서 변경 없이 사용 가능.
- Server는 요구사항에 맞도록 각 단계별로 구현.

FTP-client

- 연결 성공 시, “ftp>” 프롬프트가 뜨면 명령 입력.
- 아래 명령들이 수행되도록 구현해야 함.

사용자 명령	역할
help	FTP 사용 도움말 출력(명령어 목록)
cd [옵션]	서버의 pwd(Present Working Directory) 변경
ls [옵션] lcd(local change directory) cls [옵션]	서버 pwd의 파일 목록 디스플레이 클라이언트의 pwd 변경 클라이언트 pwd의 파일 목록 디스플레이
ccd [옵션]	클라이언트의 pwd 변경
get 파일이름	지정 파일 다운로드(서버 -> 클라이언트)
put 파일이름	지정 파일 업로드(클라이언트 -> 서버)
quit	서버 연결 해제 후 프로그램 종료

FTP-client

- 클라이언트는 사용자 입력 명령을 서버로 보내 해당 명령을 처리하게 함.
  1. 사용자 입력 명령을 받아 자체 처리하거나 필요 시 서버로 전송.
  2. 서버는 전송 받은 명령의 처리 결과를 사용자(클라이언트)에게 전송.
  3. 전송이 끝나면 서버는 클라이언트에게 '처리 완료'를 알림.

FTP-server

단계별 서버 구현:

1단계 – 서버와 클라이언트 1대1 통신 방식으로 구현.
2단계 – ‘Select’를 사용하여 다중 클라이언트 처리(멀티플렉싱)
3단계 – 멀티 프로세스 방식 서버 구현.
4단계 – 멀티 스레드 방식 서버 구현.

사용 포트는 제어 포트와 데이터 포트의 구분 없이 1 포트로 통신.
(Well-known 포트 제외)

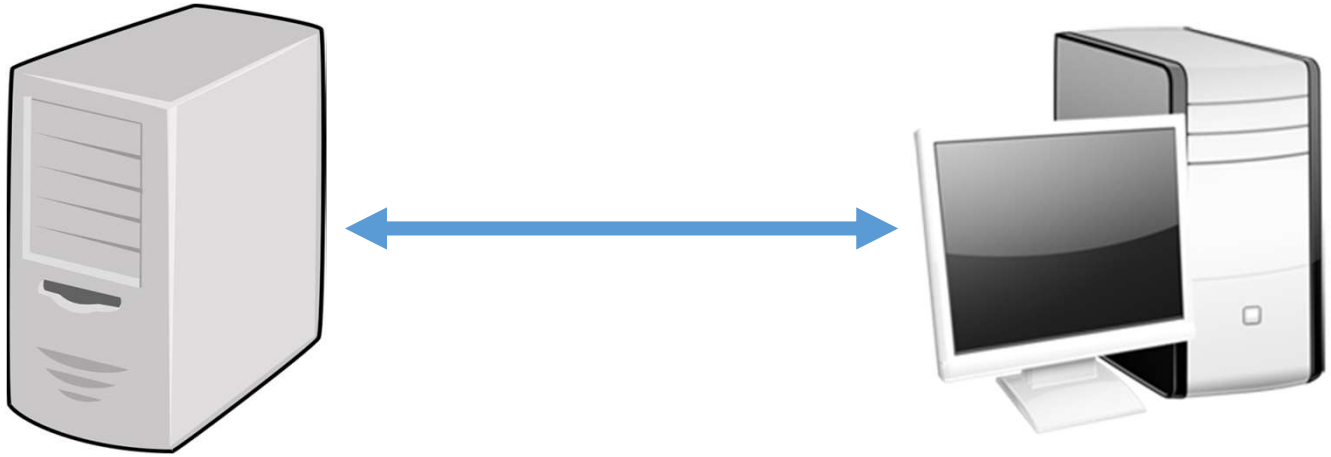
점수 배점:

1단계 8점. 2단계 3점. 3단계 5점. 4단계 4점.

제출기한:
단계별로 기한 별도 공지

FTP-server

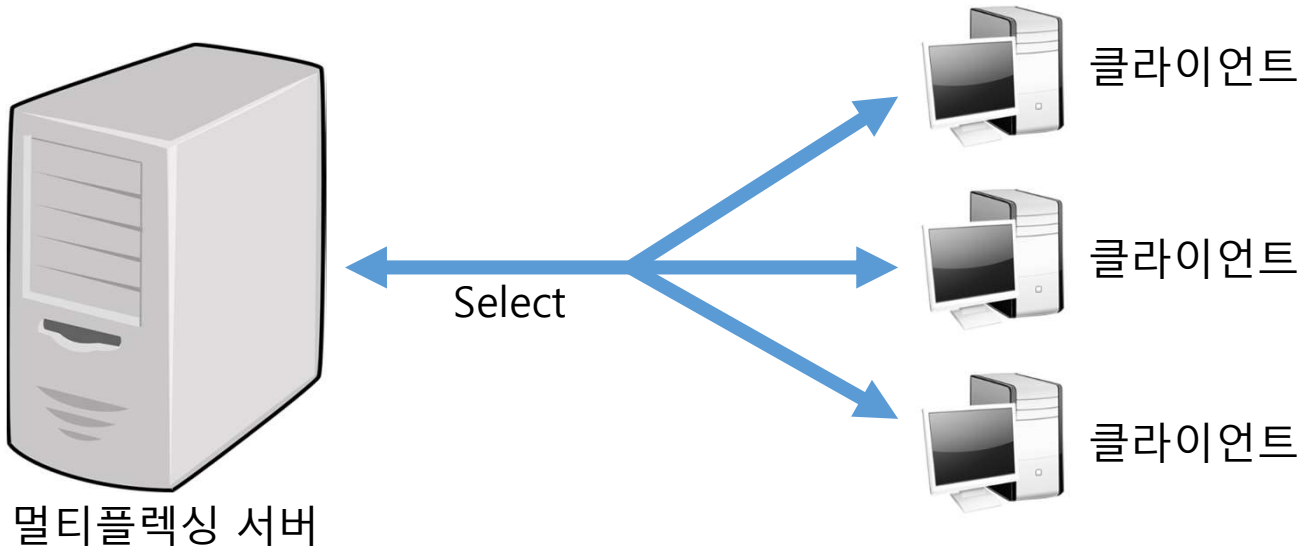
- 1단계



- 1대1 통신
- 명령들(help와 quit을 제외한 cls, ccd, ls, cd, get, put)을 모두 구현.
- (Well-known 포트 제외)

FTP-server

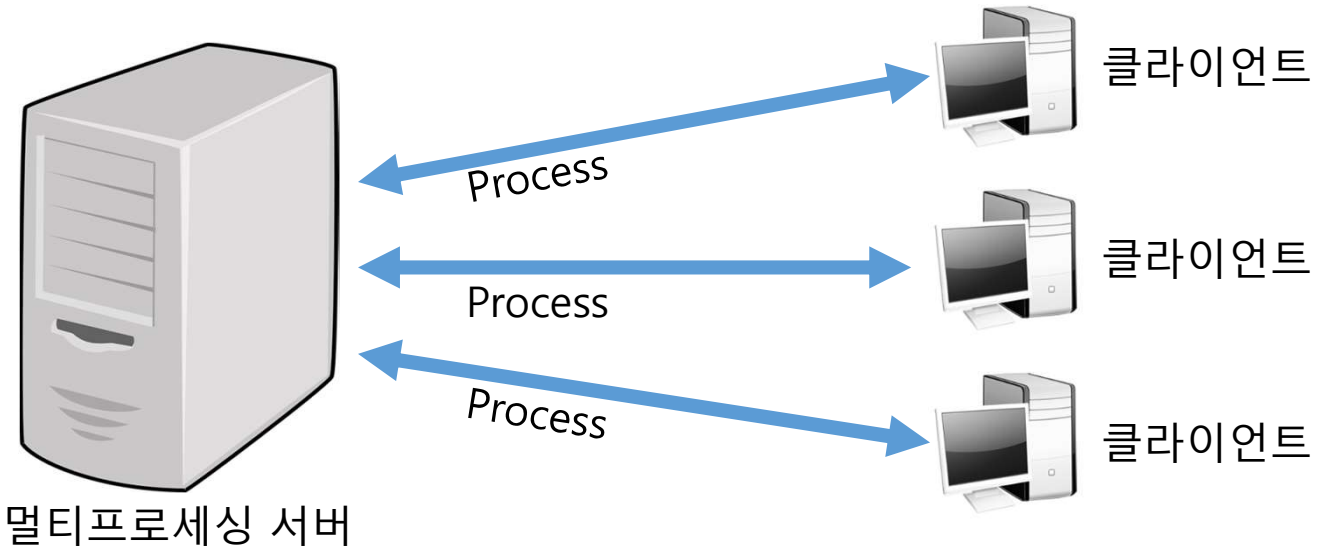
- 2단계



- 1대 N 통신.
- select문을 사용하여(멀티플렉싱 방식) 구현
- Select를 사용 시 최대 클라이언트 수는 4개로 고정.

FTP-server

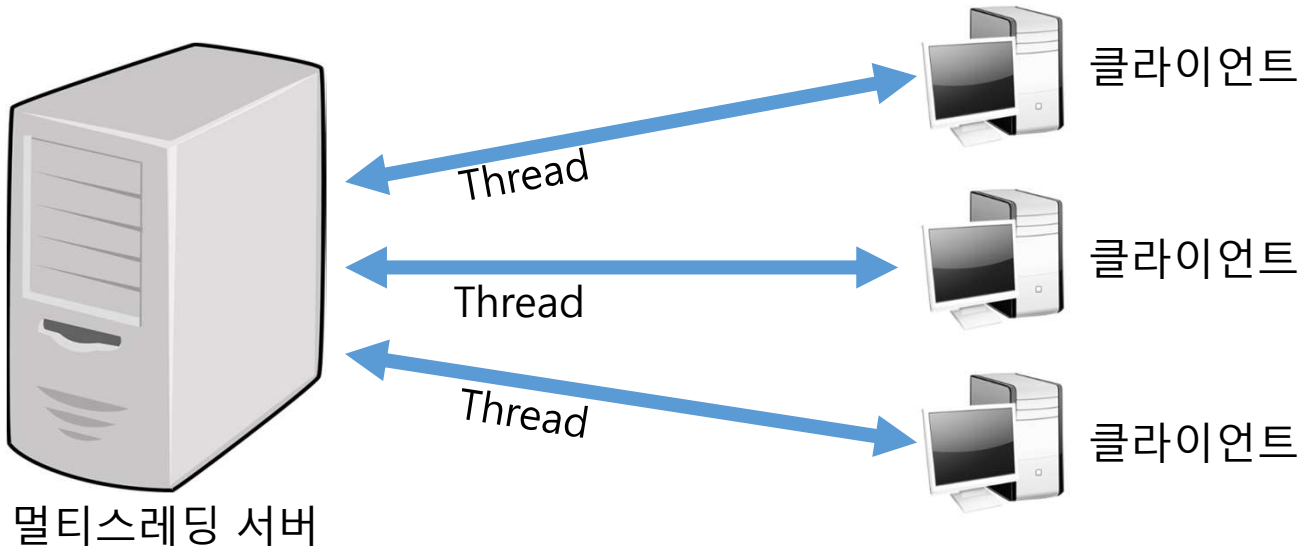
- 3단계



- 1대 N 통신
- 제어 포트로 부터 클라이언트 접속 요청을 받으면
  해당 클라이언트를 처리할 프로세스를 새로 생성하여 명령어 처리..

FTP-server

- 4단계



- 1대 N통신
- 멀티 프로세싱 서버의 경우에는 한 클라이언트와의 연결을 자식 프로세스를 생성하여 처리했으나 이 단계에서는 한 프로세스 내에 스레드를 생성하여 처리.
- 실행결과는 3단계와 동일하지만 내부 구현은 다중 스레드 방식으로 구현.

FTP-server

- 단계 4 추가 설명

  - 클라이언트로부터 연결 요청이 오면 해당 요청을 맡아 처리할 새 스레드 생성

-> 로그인 처리 완료 후 클라이언트로부터 명령 수신

-> 이후 처리는 3단계와 동일.

  - 스레드로 구현할 경우, 스레드 간 문맥교환은 OS가 해주지만
    문맥교환에 따른 스레드 상태정보는 별도로 저장해 둬야만
    해당 스레드가 다시 실행상태가 되었을 때
    자신이 저장해 놓은 상태정보를 가져와서 다시 처리를 진행할 수 있음.
  - 그렇게 하지 않으면 다른 스레드들이 실행될 경우
    자신의 상태정보가 깨져 처리가 어려울 수 있음.
    따라서 스레드(클라이언트)마다 자신의 클라이언트 ID 및
    현재 자신이 동작하는 디렉토리 정보(CWD) 등과 같은 상태정보를
    저장해 놓을 자료구조가 필요함.

FTP 프로젝트에서의 파일 송수신

- 제어(Command:명령) 채널과 데이터 송수신 채널 분리

- 데이터 채널의 설정, 데이터 송수신, 해제
  Passive Mode를 단순화 하여

1. 클라이언트 측에서 데이터 채널 생성.
   서버와 미리 약속한 데이터 채널 포트(서버 포트번호+1)로
   클라이언트 측에서 데이터 채널 연결 요청.
3. 데이터 채널 연결 설정이 완료되면 파일 데이터 송수신 진행.
4. 데이터 송수신 완료 후 데이터 채널 연결 종료.

gcc 컴파일레이션

옵션:

-o [원하는 출력파일 이름]: 출력 파일 지정.

사용자가 지정한 이름으로 실행파일 생성.

-l[라이브러리 이름]: 해당 라이브러리(아카이브)를 참조하여 컴파일.
4단계 스레드를 활용 과제에서 pthread라이브러리를 사용할 경우,

-lpthread 옵션을 명령줄에 꼭 포함시켜야 함.

gcc 사용 예:

$ gcc –o server ftpserver.c –lpthread

실행

- 터미널창 하나를 활성화하여 명령을 입력하여 실행.

- 활성화된 터미널에서 cd 명령어로
  서버와 클라이언트 실행파일이 존재하는 디렉토리로 이동.
  예: cd network_homework/2018_FTP/

- 서버 및 클라이언트 실행
  예) ./ftp_server 50000
  ./ftp_client