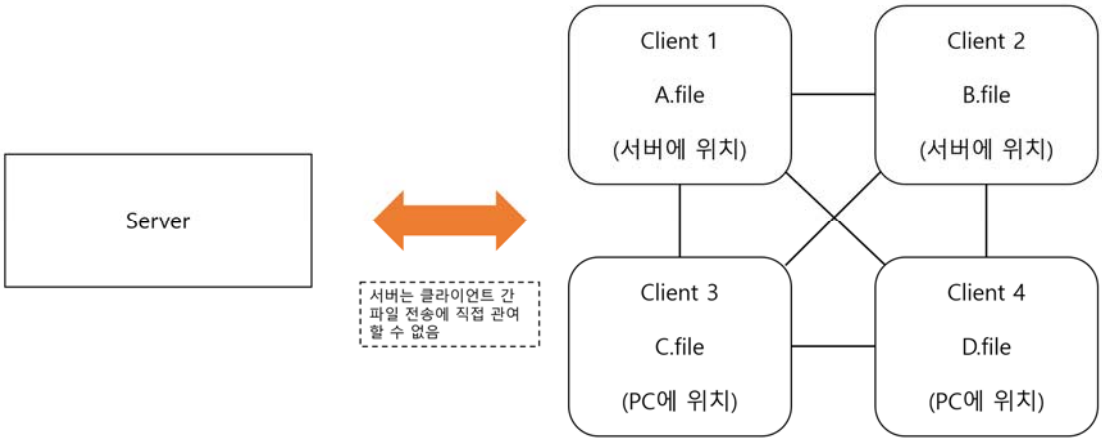
HW#3 데이터통신 프로그래밍 과제

P2P 방식 파일공유 시스템 구현

○ DUE: 2023. 12. 7(목) 23:59 (NO Late Submission Allowed)
○ How: 1개화일 제출 – G조이름HW3.zip (조별로 한명만 제출하면 됨)
◆ 모든 소스화일 및 결과화일 전체를 압축한 G조이름HW3.zip으로 제출 ex) G1HW3.zip

1. (시뮬레이션 환경 및 구성요소) 다음과 같은 실험환경을 구성한다. 4개의 클라이언트와 1개의 서버는 thread/process 간 Socket 통신으로 동작하며 연결 구조의 제약은 없다. 각각의 객체는 다음의 기능을 수행한다.



① 파일: 모든 파일은 약 500MB이며, chunk로 구성되어 있다. 각 chunk는 256KB 크기를 가진다. 최초로 서버에 접속한 클라이언트는 자신의 file이외의 다른 chunk가 없지만 시간이 지남에 따라 더 많은 chunk를 받을 수 있다. 각 파일의 chunk는 순차적으로 받지 않아도 되며, 해당화일의 모든 chunk를 수신하게 되면 최종적으로 하나의 완벽한 파일을 구성해야 한다.

② 서버: 전역변수 혹은 임의의 객체인 System Clock을 유지, 관리, 업데이트 하고 프로그램이 수행되면 System Clock을 수행시킨다. (System Clock은 time.h등을 활용하여 **실제시간**과 동일하게 계산한다.) 서버는 최소한의 동작만 수행하며, 업로드 및 다운로드 되는 실제파일을 보유할 수 없다. 클라이언트가 서버에 접속하면 클라이언트로부터 보유중인 파일 정보(파일 식별자, 보유중인 chunk 목록)를 일정 주기마다 획득하며, 이 정보를 통해 해당 파일이 필요한 클라이언트들의 정보(IP주소, 필요한 파일 및 chunk 목록)를 제공한다. 서버는 모든 event 및 결과 정보를 서버 Log로 기록한다.

③ 클라이언트: 각각의 클라이언트는 사전에 제공된 서로다른 500MB 크기 파일을 각각 보유하고 있다. 서버에 접속하여 자신 이외의 모든 클라이언트가 보유한 파일을 요청한다. 만약 다른 클라이언트에서 자신이 보유한 파일을 요청한다면, 그 즉시 해당 클라이언트와의 소켓 통신을 구성한다. 전송 1회에 하나의 chunk(256KB)만 전송할 수 있으며, 어떤 chunk를 먼저 요청할 것인지, 어떤 클라이언트에게 요청할 것인지의 제약은 없지만 조별로 가장 최적의 알고리즘을 제안하여 구현해야 된다. 클라이언트가 모든 파일을 다운로드 받았을 때 서버와의 connection을 종료한다. 각각의 클라이언트는 수행된 event 및 파일 별 다운로드 받은 chunk의 순서 및 최종 완성된 화일의 다운로드 소요 시간을 Log로 기록한다.

※ 서버는 반드시 AWS나 구글클라우드 등 Physically 외부서버에 구현 되어야함!!

※ 서버와 모든 클라이언트의 통신은 반드시 Socket으로 구현되어야 함!!

※ 서버 및 Client1,2는 서버와 같은 곳에 위치하며 Client3,4는 PC에 위치해야 함!!

※ 모든 클라이언트는 Thread나 Process를 통해 구현되어야 함!!

(해당 네가지 조건은 **과제수행의 전제조건**으로 만족되지 않으면 과제수행 인정불가)

2. (시뮬레이션 시나리오) 다음과 같은 실험 시나리오를 수행하여 각각의 구성요소 별 Log를 기록한다. 즉, 4개의 클라이언트와 서버는 각각 모든 이벤트와 연산 정보를 Client1.txt, Client2.txt, Client3.txt, Client4.txt, Server.txt에 기록한다.

① 시나리오: 프로그램 시작 후, 서버는 System Clock 0.0 msec에서 시작된다. 클라이언트 간 peer 쌍을 구성하는 역할을 수행하며, 전송되는 파일을 직접 보유할 수 없다. 각 클라이언트는 서버에 접속하여 자신이 보유한 파일의 정보를 제공하며, 그외의 나머지 모든 파일을 요청한다. 이때 서버는 각 클라이언트로부터 보유한 파일 정보를 주기적으로 요청하여 클라이언트 간의 연결을 manage한다. 이때 전송되는 파일은 chunk(256KB) 단위로 구성되어 있으며 전송 1회에 하나의 chunk만 전송이 가능하다. 클라이언트는 완전한 파일을 보유하지 않더라도 일부 chunk만 보유할 시 해당 chunk를 제공할 수 있어야 한다.

각 클라이언트는 서버에 제공하거나 제공받은 event 및 클라이언트간 화일들의 chunk의 송신 및 수신상황을 시간별로 기록한다. 하나의 파일의 모든 chunk가 수신되면 해당화일의 모든 chunk는 merge되어 하나의 완성된 화일이 구성되어야 하며, 완성된 파일의 md5해쉬값을 계산하여 original 파일과 동일한지 확인되어야 하며 해당화일의 다운로드 시간을 출력한다. 그리고, 모든 클라이언트가 모든 파일을 다운로드 받았을 때 서버는 전

체 파일의 전송에 소요된 시간을 총합하여 출력한 후 모든 connection을 종료한다 (서버 및 클라이언트는 Gracefully Termination 되어야 함). 또한 "모든 클라이언트가 서버접속 이후부터 모든 파일 다운로드 완료 후 모든 화일의 merge완성시 까지의 시간"으로 화일 전송시간을 계산하며, 해당시간이 최소가 되도록 조별로 최적의 알고리즘을 구현한다. 클라이언트의 파일 전송 시간 이외의 다른 delay는 없다고 가정하며, 구현하는 프로그래밍 언어의 제한은 없다. 프로그램 수행동안 각 요소들은 수행된 모든 event 및 결과 정보를 각각의 Log에 기록한다.

3. (과제제출) 다음화일을 조별로 기한안에 제출한다.

- G조이름HW3.zip (ex. G1HW3.zip)
  - 모든 소스 화일들
  - AllDefinedLogs.txt
    - 조별로 프로그램에서 정의한 모든 Log 메시지 명세 및 설명
  - 서버 및 클라이언트별 출력된 모든 Log 화일들
  - download.txt
    - G조이름HW3.mp4 (ex. G1HW3.mp4) : 5분이내의 설명 동영상을 제작하여, 해당동영상을 다운로드 할 수 있는 link를 포함하는 화일
    - 반드시 동영상이 다운되는 링크를 삽입할 것
    - 포함된 링크로 동영상 다운이 안되거나, 공유권한이 없거나, 다운후 동영상 실행이 안되거나 하는 등의 에러는 프로그램 수행이나 포함내용의 미포함처럼 모두 해당조의 과실로 감점대상임
  - Readme.txt
    - 조 이름, 모든 조원 학번&이름
      - 학생별 역할 명시 (불참여학생 표시할 것)
    - 프로그램 구성요소 설명
    - 소스코드 컴파일방법 명시
    - 프로그램 실행환경 및 실행방법 설명

- 구현한 최적의 알고리즘 제시 및 설명 (Pseudo Code 작성 및 설명)
- Error or Additional Message Handling에 대한 사항 설명
- Additional Comments: 추가로 과제제출관련 언급할 내용 작성

▶ 다운로드 링크 (file.zip – A.file, B.file, C.file, D.file 포함됨)

https://drive.google.com/file/d/1MJi2ejX7EaUfuBAiSRWBzGhWqz5NdczC/view?usp=drive_link

▶ A, B, C, D file MD5 해시값