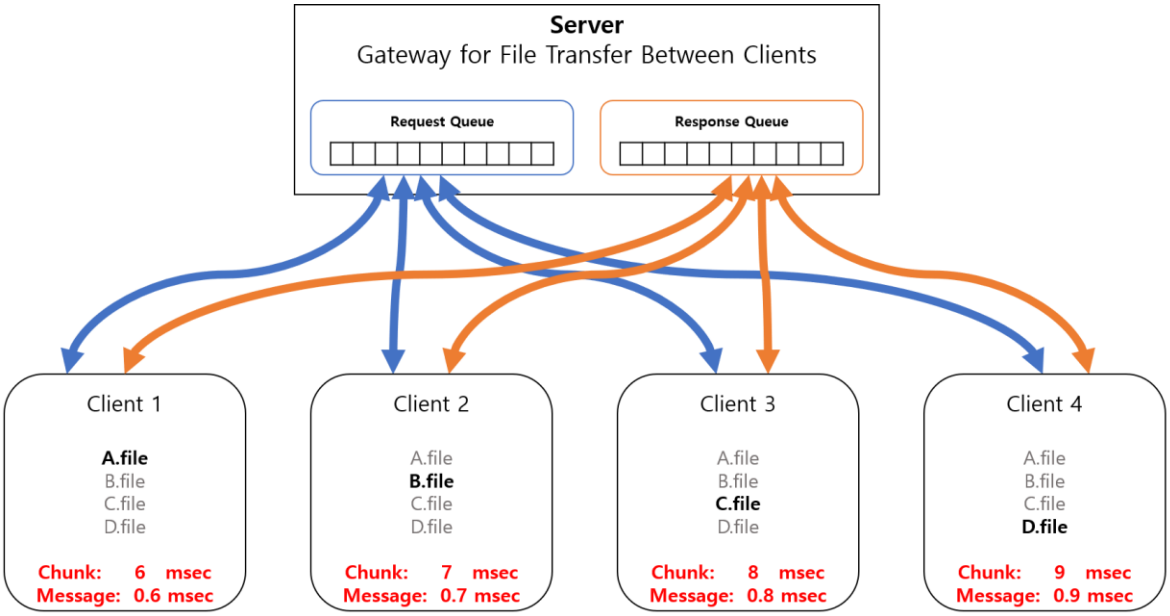
HW#4 데이터통신 프로그래밍 과제

서버-클라이언트 구조의 파일 전송 시스템 구현

○ DUE: 2024.11.18(월). 23:59	Assigned: 2024.11.4. 15:00
○ How: 1개화일 제출 – G조이름HW4.zip (조별로 한명만 제출하면 됨)	
◆ 모든 소스화일 및 결과화일 전체를 압축한 G조이름HW4.zip으로 제출 ex) G1HW4.zip	

1. (시뮬레이션 환경 및 구성요소) 다음과 같은 실험환경을 구성한다. 1개의 Server와 4개의 Client는 thread간 Socket 통신으로 동작하며 연결 구조의 제약은 없다. 각각의 객체는 다음의 기능을 수행한다.



① 파일: 모든 파일은 약 500MB이며, 최소단위는 chunk로 구성되어 있다. 각 chunk는 128KB 크기를 가진다. 최초로 서버에 접속한 클라이언트는 chunk가 없지만 시간이 지남에 따라 더 많은 chunk를 쌓을 수 있다. 각 파일의 chunk는 순차적으로 송수신 않아도 되며, 모든 chunk를 보유하였을 때 최종적으로 완벽하게 구성된 파일로 이루져야 한다.

② 서버: 전역변수 혹은 임의의 객체인 System Clock을 유지, 관리, 업데이트 하고 프로그램이 수행되면 가상의 System Clock(msec 단위로 기록)을 수행시킨다. 서버는 시작 시 모든 클라이언트의 각 파일 정보(파일 종류, chunk 목록)를 받아 관리하고, 계속하여 송수신되는 상태를 업데이트한다. 클라이언트의 특정 파일 요청을 받게 되면 request queue에 요청을 저장하여, 해당 파일을 가지고 있는 다른 클라이언트에게서 파일을 요청하여 response queue를 통해 전달해준다(queue의 사이즈에 제약은 없다). 이때 서버는 파일을

따로 저장할 수 없으며 클라이언트 간 통신의 게이트웨이 및 릴레이 역할로 사용된다. 어떤 클라이언트에게 요청할지에 대한 제약은 없으며 통신 상태 및 전송 속도를 고려하여 최적의 알고리즘을 제안하여 구현해야 한다. 서버는 모든 송수신 event 및 각 chunk의 평균 전송소요 시간을 Log로 기록한다.

③ 클라이언트: 각각의 클라이언트는 사전에 제공된 500MB 크기의 각기 다른 파일을 보유하고 있다. 클라이언트는 최종적으로 기존에 보유하고 있던 파일을 포함하여 총 4개의 파일을 보유해야한다. 이를 위해서 서버에게 보유하지 않은 임의의 chunk를 요청하는 메세지를 전송하여 진행된다. 마찬가지로 서버에서 클라이언트가 보유하고 있는 파일을 요청한다면 해당 파일을 서버에게 전달하여야 하며, 서버의 파일 정보 업데이트를 위해 chunk 전송이 완료되었다는 메세지를 보낸다. 서버와 각 클라이언트간의 연결은 Half-Duplex 통신으로 구성되어야 하며, 파일 및 메세지의 전송속도는 그림과 같이 구성 되어 있다. 클라이언트가 모든 파일을 다운로드 받았을 때 서버와의 connection을 종료한다. 각각의 클라이언트는 클라이언트에서 발생하는 모든 event 및 서버와의 송수신, 파일 별 다운로드 받은 chunk의 순서와 다운로드 소요 시간을 모두 클라이언트 Log에 기록한다.

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

※ Server와 모든 Client의 통신은 반드시 Socket으로 구현되어야 함!!

※ 모든 구성요소는 Thread를 통해 구현되어야 함!!

※ 제출된 소스는 안전하게 컴파일되어야 함!!

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

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

① 시나리오: 프로그램 시작 후, 서버는 모든 클라이언트와의 통신 환경을 초기화 후 가상의 system clock을 생성한다. System clock은 서버에서 관리하며, 실제 시간이 아닌 operation에서 소요되는 가상의 연산 시간과 가상의 network delay를 측정한다. 서버는 클라이언트 간 통신에서 필요한 전달 역할 이외에 별도로 파일을 저장할 수 없다. system clock의 시작과 함께 서버는 모든 클라이언트의 파일 정보를 확인하여 계속 관리를 진행한다. 각 클라이언트는 서버에게 필요한 임의의 chunk를 요청한다. 이때 서버에서는 thread를 사용하여 각 클라이언트의 통신을 관리하며, Half-Duplex 통신으로 구성된다. 클라이언트는

완전한 파일이 구성되지 않고 일부 chunk만 보유하고 있더라도 해당 chunk를 서버에게 제공할 수 있으며, chunk를 순차적으로 구성하지 않아도 된다. 서버는 요청된 chunk를 어느 클라이언트에게 받아 전달할 것인지에 대한 제약조건이 없지만 최적의 알고리즘을 제출 시 구현된 규칙이나 알고리즘에 대해 명시하여야 한다. 예를 들어 클라이언트 1에서 C.file의 6번 chunk를 요청하여 chunk를 받는 과정은 (1) 클라이언트 1에서 0.6 msec 동안 C.file의 6번 chunk 요청을 서버로 전송한 후, (2) 서버는 각 클라이언트의 통신상태 및 보유 파일을 확인하여 클라이언트 3에게 0.8 msec동안 chunk 요청 메세지를 전송한다. (3) 그 후 클라이언트 3은 8 msec 동안 서버로 해당 chunk를 전송한다. (4) 서버는 전송받은 chunk를 6 msec 동안 클라이언트 1에게 전송한다. (5) 전송이 끝난 후 클라이언트 1은 0.6 msec 동안 전송이 완료되었다는 메세지를 서버에게 전달한다. (6) 전송 완료 메세지를 받는 서버에서는 파일 관리를 위해 정보를 업데이트 한다. 모든 클라이언트가 4개의 파일을 모두 보유한다면 서버는 모든 클라이언트와의 연결을 종료한다(모든 node는 Gracefully Termination 되어야 함). 프로그램 수행동안 서버는 모든 이벤트(클라이언트의 chunk 요청정보, 송수신 정보)를 System Clock과 함께 log로 기록 하며, 모든 클라이언트는 모든 이벤트(chunk 요청정보, 송수신 정보)를 System Clock과 함께 log로 기록한다. 모든 클라이언트는 log의 마지막에 각각의 파일 다운로드에 소요된 시간과 자신의 파일 및 수신된 모든 chunk를 각각의 하나의 파일로 합친후 정확한 MD5 해시값을 출력한다. (각화일의 모든 MD5 값은 동일해야함)

- 각 파일들의 MD5 Hash 예시

```
(base) phs@super:~/Desktop/file$ md5sum A.file
db386d262ce1e7c3152f273f42819f9f  A.file
(base) phs@super:~/Desktop/file$ md5sum B.file
9b536092ed3164f3e276124aa19caa09  B.file
(base) phs@super:~/Desktop/file$ md5sum C.file
103e97e3e82d5d2701e487f60175070e  C.file
(base) phs@super:~/Desktop/file$ md5sum D.file
386cbd00333b04efc2d60202df709ade  D.file
```

3. (과제제출) 다음 화일을 조별로 기한안에 제출한다.

   - G조이름HW4.zip (ex. G1HW4.zip)
     - 모든 소스 화일들
     - AllDefinedLogs.txt

- 조별로 프로그램에서 정의한 모든 Log 메시지 명세 및 설명

◆ 서버 및 클라이언트별 출력된 모든 Log 화일들

◆ download.txt

- G조이름HW4.mp4 (ex. G1HW4.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, 다운 후 동영상 실행이 안되거나 하는 등의 에러는 프로그램 수행이나 포함내용의 미포함처럼 모두 해당조의 과실로 감점대상임

◆ Readme.txt

· 조 이름, 모든 조원 학번&이름

- 학생 별 역할 명시

· 프로그램 구성요소 설명

· 소스코드 컴파일방법 명시

· 프로그램 실행환경 및 실행방법 설명

· 구현한 최적의 알고리즘 제시 및 설명 작성

· Error or Additional Message Handling에 대한 사항 설명

· Additional Comments: 추가로 과제제출관련 언급할 내용 작성