

# 디자인 및 구현 문서 - TCP

2020036371 조재혁

## I. 디자인 설계

본 문서는 구현을 시작하기 전 설계하면서 생각했던 과정을 기술한다.

TCP의 기초적인 개념은 Server 하나와 Client가 존재하며, Server는 모든 Client에 1대1 대응한다는 것이다. 따라서 최소 2개의 클래스가 필요하다고 느꼈으며, message의 흐름이 Client->Server->Client로 진행되기 때문에 Client에서 Send와 Receive Thread, Server에 모든 기능을 종합하여 수행하는 Thread까지 해서 총 3개의 스레드가 필요하다고 판단했다.

추가로, 이번 과제에서는 다수의 채팅방을 만들 수 있어야 하고, 각 방끼리만 대화가 돼야 한다는 조건이 있기 때문에 채팅방의 리스트가 필요할 것이라고 확신하였고, 각 방마다 유저들의 이름이 겹치지 않게 유저 이름 리스트까지 구현하면 좋을 것이라는 생각이 들었다.

마지막으로, message를 parsing하고 수행하는 역할은 전부 Server에 할당하였다. Client와 Server가 각각의 기능을 하고 있으면 논리적으로 불편하다고 판단하여 Client는 오직 형식에 맞게 Sending하고 Receiving만 하는 것으로 정했고, 그 외의 모든 기능은 Server에서 수행하도록 구상하였다.

## II. 구현 과정

본 문서는 기본적인 구현 과정은 생략하고 큰 틀에서의 구현 과정을 말하며, 추가로 어려움을 느꼈던 부분을 해결한 과정과, 명세 외로 개인적으로 추가한 기능들에 대한 내용을 다룬다.

### 1. Other Classes

구상하는 과정에서 Room List가 있으면 좋을 것 같아서 'ChattingRoom'이라는 클래스는 먼저 만들었다. 모든 message와 file은 한 채팅방 안에서만 접근가능 해

야 하므로 file들을 저장하는 FileList와 message를 방 내의 모든 Client에게 보내기 위한 socket들의 list를 ArrayList로 선언했고, 방 이름을 string로 선언해서 마무리했다.

그러나 이후에 socket의 user name을 가져오는 절차가 불편하다고 느꼈고, socket list보다는 user list라고 쓰는게 코드의 가독성을 높이기 좋을 것이라 판단하여 socket과 user name을 pair처럼 쓸 수 있게 만드는 User class를 새로 만들어서 socket list를 UserList로 대체하였다.

## 2. Client

Client의 Thread 2개는 UDP때와 굉장히 유사한 구조로 구성된다. Main 함수에서 실행인자를 parsing한 후 client socket과 file socket을 생성해주고 2개의 socket을 Send Thread, Receive Thread 2개에 각각 삽입해주었다.

Send Thread에서는 기본적으로 system에서 input을 받고 그대로 server에 전송만 해주는 역할을 수행하며, Receive Thread는 reader로 서버에서 보내주는 메시지만 화면에 출력해주는 역할을 수행한다. 단, PUT 명령어를 수행할 때에는 따로 상황에 맞게 file stream을 열어주어야 하기 때문에 Receive가 아닌 Send Thread에서 stream open 작업을 처리하게 했다.

File 전송의 구조를 Client 측에서 자세히 살펴보면 PUT의 명령어가 들어오게 되면 server에서 파일을 보내줄 것을 준비하는 작업을 한다. 먼저 system로부터 파일을 받아온 뒤, 65536byte(64Kbyte)로 쪼개서 차례대로 보내는 식이다. GET의 경우는 FILE을 받기만 하면 되므로 #SENDING이라는 서버의 메시지가 도착하면 file stream을 열어 파일을 받을 준비를 한다.

## 3. Server

Server는 socket을 열고 thread를 시작하기전, 전역변수를 선언해줄 필요가 있다. 구상 단계에서 설명했듯이 room list를 선언해주었고, 추가로 채팅 방과 상관없이 서버 전체에서 user목록을 확인할 수 있게 user list를 선언했다. 이는 Client 하나가 특정 이름을 쓰다가 다른 방으로 넘어가서 같은 이름을 쓰고 싶은데 다른 Client에게 그 이름을 뺏기지 않게 하기 위한 조치이다. Main 함수에서는 welcome socket을 2개 선언하고 (chat과 file) while문을 돌면서 client가 접속할 때마다 thread에 socket을 탑재하고 시작한다.

Thread의 첫 부분에는 기본적인 stream선언을 해준 뒤, join flag라는 boolean 변수를 하나 선언해준다. 이는 메시지를 전달하거나, file을 주고받고자 할 때 채팅방에 속해 있는지 쉽게 확인하기 위함이다. 반대로 이미 참가한 상태인데 실수로 CREATE이나 JOIN을 시도할 때 방지하는 역할을 해주기도 한다.

이후로는 메시지가 #으로 시작하는 지 구분한 후 #으로 시작하면 각종 명령어를 수행, 일반 메시지인 경우에는 room list에 있는 user list를 참고하여 본인을 제외한 모든 socket으로 send한다. 또한, 새로운 client가 자신이 속한 채팅방에 입장했을 경우나 나가는 경우에는 이를 알려주는 메시지를 출력함으로써 일일이 status를 보지 않고도 현재 상태를 인지할 수 있게 해주었다.

Client가 보낸 file을 받는 경우는 64Kbyte chunk로 넘어온 바이트를 한 줄로 이어주고 'files'라는 폴더로 저장을 해주는 형태를 취한다. 그렇게 해서 원래 실행파일과 같은 경로에 있던 file을 삭제하더라도 files 폴더에서 그 파일을 찾아 다시 download할 수 있게 실제 통신과 가깝게 구현해보았다. 반대로 client에게 send해줄 때에는 client가 server에 보낼 때의 방식과 같다. 단, 찾고자 하는 file이 존재하지 않는 경우 #CANNOTSEND 명령어를 client에 보내서 client의 receive thread가 직접 예외처리 할 수 있게 만들었다.

#### 4. More Options

명령어 QUIT과 HELP, welcome message, 다양한 예외처리를 추가했다. QUIT의 경우 client가 채팅방을 나가는 것이 아니라 program 자체를 종료하고 싶을 때 server에서는 client의 모든 socket을 닫고 client에서는 system을 종료하도록 만들었다. HELP는 흔히 보이는 명령어로, command에 대한 정보가 없는 유저를 위하여 현재 지원되는 모든 명령어와 각 기능들을 간단히 보여준다. 이는 client가 처음 접속할 때 보여주는 welcome message에 help command의 존재를 알려주었으며, 잘못된 command를 입력했을 경우에도 다시 한번 help를 추천하는 식으로 활용해보았다.

Join되어있는 상태에서 join을 하거나, 이미 있는 이름으로 또 다른 방을 만들거나, 방에 가입되어 있지 않은데 exit을 입력하는 등의 잘못된 명령어가 들어오는 등의 모든 예외 상황에 오류 메시지를 출력하도록 노력했다. 알림 메시지나 오류 메시지와 같이 server에서 보내는 모든 메시지에는 [Server]을 앞부분에 위치시켜

서 client가 그 메시지가 server에서 온 메시지임을 알 수 있게 만들어 편의성을 높였다.

### III. 실행 방법

본 문서는 class 파일을 컴파일 및 실행하는 방법과 예시 화면을 제시한다. 본인은 mac 환경에서 작업하여 terminal로 컴파일 하는 과정을 거쳤으며, windows 환경에서도 cmd로 작동하는지 확인한 결과 문제는 없었다.

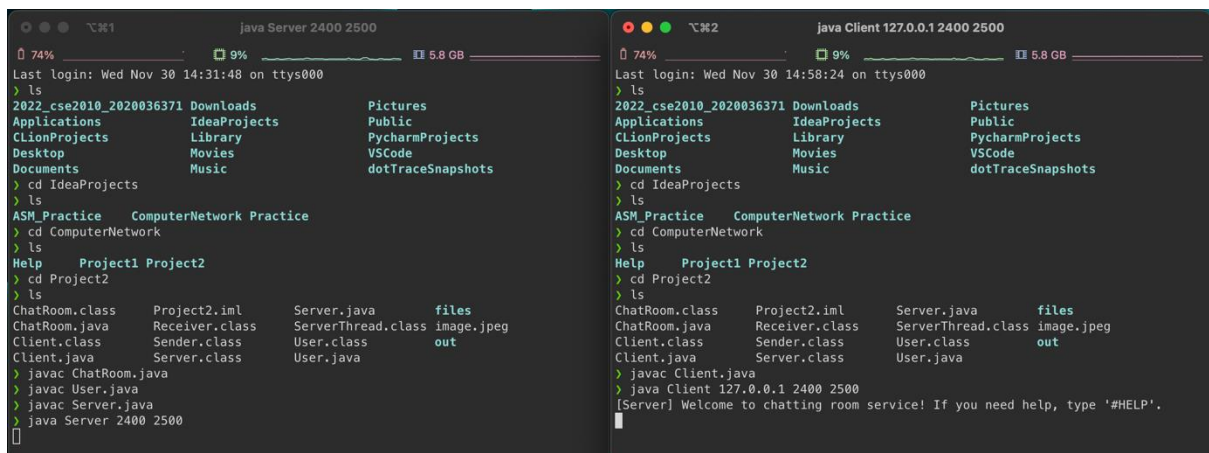
UDP와 달리 Server는 단 한 개만 실행되어야 하며, 첫 Client를 실행하기 전 반드시 Server를 먼저 실행할 것을 유의한다.

- 다음 단계에 따라 실행 -

1. terminal (cmd) 실행 후 java 파일이 있는 경로로 이동
2. 컴파일-1: javac ChatRoom.java, javac User.java
3. 컴파일-2: javac Client.java, javac Server.java
4. 실행-1: java Server port1 port2 (임의의 port 번호 입력)
5. 실행-2: java Client IP port1 port2 (local IP 주소와 임의의 port 번호 입력)
6. 추가로 client을 생성하고 싶은 경우 1단계와 5단계 반복

아래에 예시 컴파일 화면과 실행 화면을 첨부 하였다.

예시 컴파일 화면 (최초 실행)



```
java Server 2400 2500
Last login: Wed Nov 30 14:31:48 on ttys000
> ls
2022_cse2010_2020036371 Downloads Pictures
Applications IdeaProjects Public
CLionProjects Library PycharmProjects
Desktop Movies VSCode
Documents Music dotTraceSnapshots
> cd IdeaProjects
> ls
ASM_Practice ComputerNetwork Practice
> cd ComputerNetwork
> ls
Help Project1 Project2
> cd Project2
> ls
ChatRoom.class Project2.iml Server.java files
ChatRoom.java Receiver.class ServerThread.class image.jpeg
Client.class Sender.class User.class out
Client.java Server.class User.java
> javac ChatRoom.java
> javac User.java
> javac Server.java
> java Server 2400 2500

java Client 127.0.0.1 2400 2500
Last login: Wed Nov 30 14:58:24 on ttys000
> ls
2022_cse2010_2020036371 Downloads Pictures
Applications IdeaProjects Public
CLionProjects Library PycharmProjects
Desktop Movies VSCode
Documents Music dotTraceSnapshots
> cd IdeaProjects
> ls
ASM_Practice ComputerNetwork Practice
> cd ComputerNetwork
> ls
Help Project1 Project2
> cd Project2
> ls
ChatRoom.class Project2.iml Server.java files
ChatRoom.java Receiver.class ServerThread.class image.jpeg
Client.class Sender.class User.class out
Client.java Server.class User.java
> javac Client.java
> java Client 127.0.0.1 2400 2500
[Server] Welcome to chatting room service! If you need help, type '#HELP'.
```

## 예시 실행 화면 (Server 1개, Client 6개, Room 2개)

```

Client 1:
> java Client 127.0.0.1 2400 2500
[Server] Welcome to chatting room service! If you need help, type '#HELP'.
#CREATE cnet James
[Server] You successfully made room <cnet>.
[Server] Stella has joined the room.
[Server] Henry has joined the room.
Hello
Stella: Hello :)
Henry: Nice to meet yo
[Server] James has left the room.
[Server] Stella has left the room.
#EXIT
[Server] You have left the room <cnet>.

Client 2:
> java Client 127.0.0.1 2400 2500
[Server] Welcome to chatting room service! If you need help, type '#HELP'.
#JOIN cnet Stella
[Server] You successfully joined room <cnet>.
[Server] Henry has joined the room.
James: Hello
Hello :)
Henry: Nice to meet yo
[Server] James has left the room.
[Server] Stella has left the room.
#EXIT
[Server] You have left the room <cnet>.

Client 3:
> java Client 127.0.0.1 2400 2500
[Server] Welcome to chatting room service! If you need help, type '#HELP'.
#JOIN cnet2 Daniel
[Server] You successfully made room <cnet2>.
[Server] Lea has joined the room.
[Server] Amy has joined the room.
Hello guys
Whats up
Amy: doing great!
Lea: me too!
Amy: happy to see you
Amy: must go now :(
[Server] Amy has left the room.
bye
#EXIT
[Server] You have left the room <cnet2>.

Client 4:
> java Client 127.0.0.1 2400 2500
[Server] Welcome to chatting room service! If you need help, type '#HELP'.
#JOIN cnet2 Lea
[Server] You successfully joined room <cnet2>.
[Server] Amy has joined the room.
Daniel: Hello guys
Daniel: Whats up
Amy: doing great!
me too!
Amy: happy to see you
Amy: must go now :(
[Server] Amy has left the room.
Daniel: bye
[Server] Daniel has left the room.
#EXIT
[Server] You have left the room <cnet2>.

Client 5:
> java Client 127.0.0.1 2400 2500
[Server] Welcome to chatting room service! If you need help, type '#HELP'.
#JOIN cnet2 Amy
[Server] You successfully joined room <cnet2>.
Daniel: Hello guys
Daniel: Whats up
doing great!
Lea: me too!
happy to see you
must go now :(
#EXIT
[Server] You have left the room <cnet2>.

Client 6:
> java Client 127.0.0.1 2400 2500
[Server] Welcome to chatting room service! If you need help, type '#HELP'.
#JOIN cnet Stella
[Server] You successfully joined room <cnet>.
[GET image.jpeg]
[Server] Upload process: #####
[Server] Successfully uploaded the file.

```

## 예시 실행화면 (Server 1개, Client 2개, 파일 upload and download)

```

Client 1:
> cd IdeaProjects
> ls
ASM_Practice ComputerNetwork Practice
> cd ComputerNetwork
> ls
Help Project1 Project2
> cd Project2
> ls
ChatRoom.class Project2.iml Server.java files
ChatRoom.java Receiver.class ServerThread.class image.jpeg
Client.class Sender.class User.class out
Client.java Server.class User.java
> java Client 127.0.0.1 2400 2500
[Server] Welcome to chatting room service! If you need help, type '#HELP'.
#CREATE cnet James
[Server] You successfully made room <cnet>.
[Server] Stella has joined the room.
#PUT image.jpeg
[Server] Upload process: #####
[Server] Successfully uploaded the file.

Client 2:
> cd IdeaProjects
> ls
ASM_Practice ComputerNetwork Practice
> cd ComputerNetwork
> ls
Help Project1 Project2
> cd Project2
> ls
ChatRoom.class Project2.iml Server.java files
ChatRoom.java Receiver.class ServerThread.class image.jpeg
Client.class Sender.class User.class out
Client.java Server.class User.java
> java Client 127.0.0.1 2400 2500
[Server] Welcome to chatting room service! If you need help, type '#HELP'.
#JOIN cnet Stella
[Server] You successfully joined room <cnet>.
#GET image.jpeg
[Server] Download process: #####
[Server] Successfully downloaded the file.

```