네트워크프로그래밍-7주
화상강의 자료

정인환교수

공지사항

- 8주 중간고사 (Quiz 50분)
  - 이론 : TCP/IP 이론, 소켓 기본 프로그래밍
    - 온라인 Quiz 방식
  - 10/20(수) 화상강의 시간
    - A반 11:00 ~ 11:50
    - 7반 15:00 ~ 15:50
    - N반 18:00 ~ 18:50
  - 10/21(목) 화상강의 시간
    - B반 11:00 ~ 11:50
    - 8반 16:00 ~ 16:50
    - O반 18:00 ~ 18:50
- 7주 과제 → 9주 10/26까지
- 8주 동영상강의, 화상강의 없음

7주 화상강의 내용

- 과제 Review
  - chat_client/server 1, 2, 3, 4
  - 도전과제
    - chat_client/server5, 6

- 7주 강의
  - Java 소켓 프로그래밍
  - Java Echo client/server
  - Java Chat client/server

6주 과제1: 강의내용 복습

- 1단계 : chat_client1/server1
  - 기본적인 chatting
- 2단계 : chat_client2/server2
  - username 입력, [username] message .. 형식 사용
  - exit 종료 처리
- 3단계 : chat_client3/server3
  - /list 로 사용자 목록 보기
  - /login, /list, /exit 처리

6주 과제2: chat_client4/server4 채팅 기능 추가하기

- 새로운 사용자가 입장/퇴장하면 다른 사용자에게 알려주기
  - [user1]님이 입장하였습니다.
  - [user1]님이 퇴장하였습니다.
- /to 명령어 – 귓속말 기능
  - /list로 사용자 목록 확인후 사용하는 식
  - char userlist[MAXCLIENTS][BUF_LEN] 를 이용
  - atom 화면
    - atom> /to hansung Hello there ..
  - hansung 화면
    - [귓속말] [atom] Hello there ...
- /sleep 명령어 – 부재중 기능
  - 채팅 메시지 수신 거부
    - atom> /sleep
  - /wakeup 또는 아무 message나 전송하면 깨어난다.
  - userlist[][] 외에 usersleep[] 변수 추가, 서버에서 관리
- /list 에 user 상태 표시 추가

chat_server4.c /list, /to

```c
// /list 처리
if (strcmp(buf2, CHAT_CMD_LIST) == 0) { // "/list"
    printf("Sending user list to client[%d] %s\n", i, userlist[i]);
    memset(buf, 0, BUF_LEN);
    sprintf(buf, "User List\nNo\tname\tstatus\n----------------------\n");
    if (send(client_fds[i], buf, BUF_LEN, 0) < 0) {
        printf("client[%d] send error.", i);
        client_error[i] = 1;
        continue;
    }
    for (j = 0; j < num_chat; j++) {
        memset(buf, 0, BUF_LEN);
        sprintf(buf, "%02d\t%s\t%s\n", j, userlist[j],
            usersleep[j] ? "S" : "O" );
        if (send(client_fds[i], buf, BUF_LEN, 0) < 0) {
            printf("client[%d] send error.", i);
            client_error[i] = 1;
            break;
        }
    }
    memset(buf, 0, BUF_LEN);
    sprintf(buf, "----------------------\n");
    if (send(client_fds[i], buf, BUF_LEN, 0) < 0) {
        printf("client[%d] send error.", i);
        client_error[i] = 1;
        continue;
    }
    continue;
}
```

```c
// /to 처리 귓속말 처리
// [user1] /to user2 message ...
if (strcmp(buf2, CHAT_CMD_TO) == 0) { // "/to"
    char username[BUF_LEN], to[BUF_LEN], to_user[BUF_LEN], msg[BUF_LEN];
    sscanf(buf, "%s %s %s", username, to, to_user);
    strcpy(msg, buf + strlen(username) + strlen(to) + strlen(to_user) + 3);
    printf("[귓속말] from %s to %s : %s", username, to_user, msg);
    // 귓속말 전송
    for (j = 0; j < num_chat; j++) {
        // user가 sleep 이면 어떤 메시지도 수신하지 않는다.
        if (usersleep[j]!=1 && strcmp(userlist[j], to_user) == 0) {
            memset(buf2, 0, BUF_LEN);
            sprintf(buf2, "[귓속말] %s %s", username, msg);
            if (send(client_fds[j], buf2, BUF_LEN, 0) < 0) {
                printf("client[%d] send error.", j);
                client_error[j] = 1;
                break;
            }
        }
    }
    continue;
}
```

chat_server4.c /exit, /sleep, /wakeup, 방송

```c
// /exit 처리
// exit 대신 /exit 로 변경.
if (strcmp(buf2, CHAT_CMD_EXIT) == 0) { // "/exit"
    RemoveClient(i);
    continue;
}
// /sleep 처리
if (strcmp(buf2, CHAT_CMD_SLEEP) == 0) { // "/sleep"
    usersleep[i] = 1;
    continue;
}
// /wakeup 처리
if (strcmp(buf2, CHAT_CMD_WAKEUP) == 0 || strlen(buf2) == 0) {
    usersleep[i] = 0;
    continue;
}
```

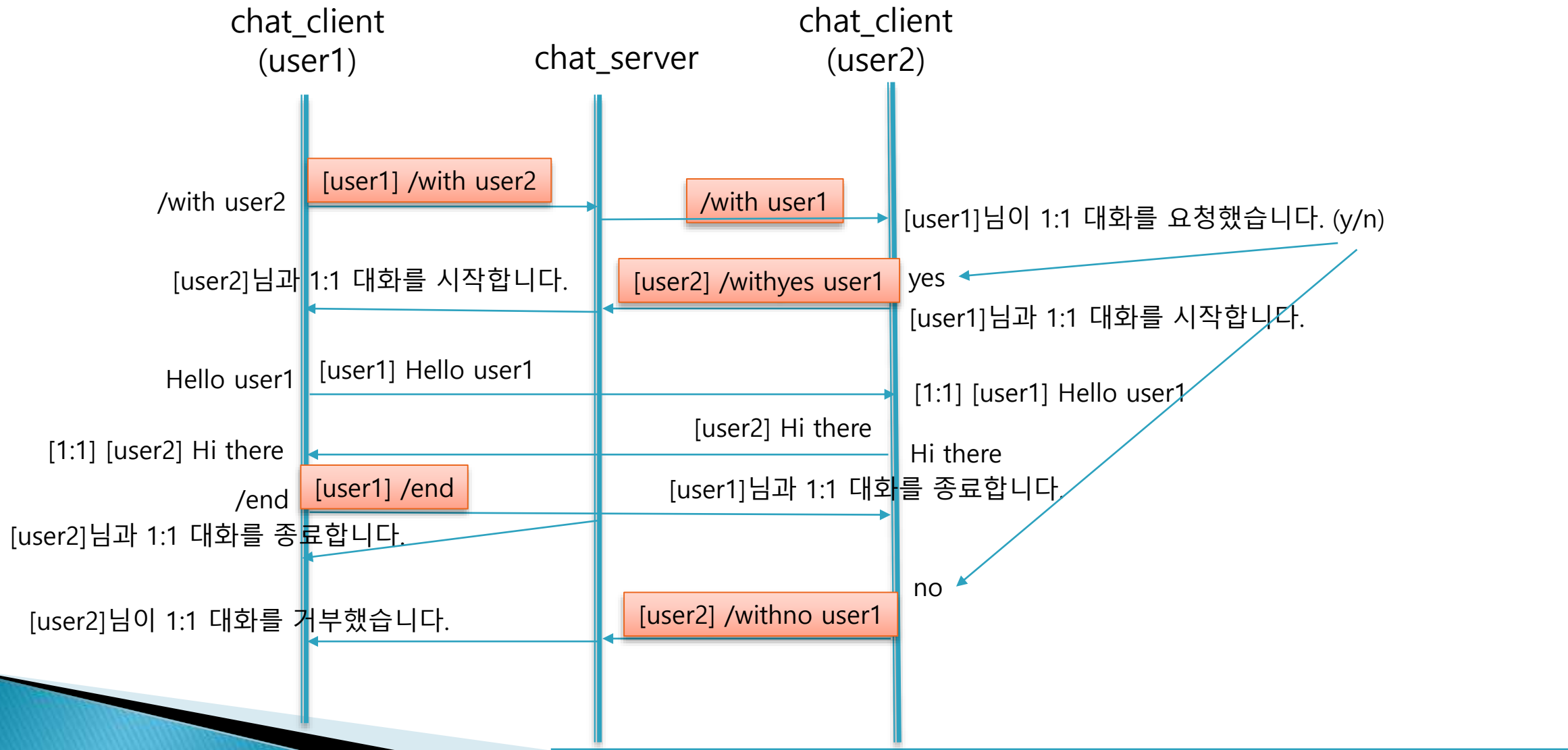
```c
// 모든 채팅 참가자에게 메시지 방송
//printf("%s", buf);
// Wrie All
usersleep[i] = 0; // message 가 있으면 무조건 깨어난다.
for (j = 0; j < num_chat; j++) {
    if (usersleep[j] != 1) { // user가 sleep 이면 어떤 메시지도 수신하지 않는다.
        ret = send(client_fds[j], buf, BUF_LEN, 0);
        if (ret <= 0) {
            printf("send error for client[%d]\n", j);
            client_error[j] = 1;
        }
    }
}
```

6주 도전과제1: 1:1 채팅모드 전환 chat_client5/server5

- 명령어 추가
  - /with user2 – user2 와 1:1 채팅 시작
  - /end – 1:1 채팅 종료
- 1:1 채팅을 user2가 허락해야 가능하도록 프로토콜 구현

  예) user1 화면 – user1> /with user2
  user2 화면 – [user1]님이 1:1 대화를 요청했습니다. (y/n)? y
  user1 화면 – [user2]님과 1:1 대화 시작
  user2 화면 – [user1]님과 1:1 대화 시작
  user1 화면 – user1> /end (또는 user2 화면 – user2> /end)
  user1 화면 – user1> Hello user2
  user2 화면 – [1:1] [user1] Hello user2, user2> Hi user1
  user1 화면 – [1:1] [user2] Hi user1
  user1 화면 – [user2]님과 1:1 대화 종료
  user2 화면 – [user1]님과 1:1 대화 종료
  user2 화면 – [user1]님이 1:1 대화를 요청했습니다. (y/n)? n
  user1 화면 – [user2]님이 1:1 대화를 거부했습니다.
- Hint : Sever에서 1:1 제어 int userwith[] 배열에 정보 저장
  - /with 를 요청한 user1 번호 = i, user2 번호 = k 라면
  - userwith[i] = k, userwith[k] = i 로 저장하고
  - user1이 message를 보내면 1:1 대상자에게만 전송

1:1 채팅 프로토콜

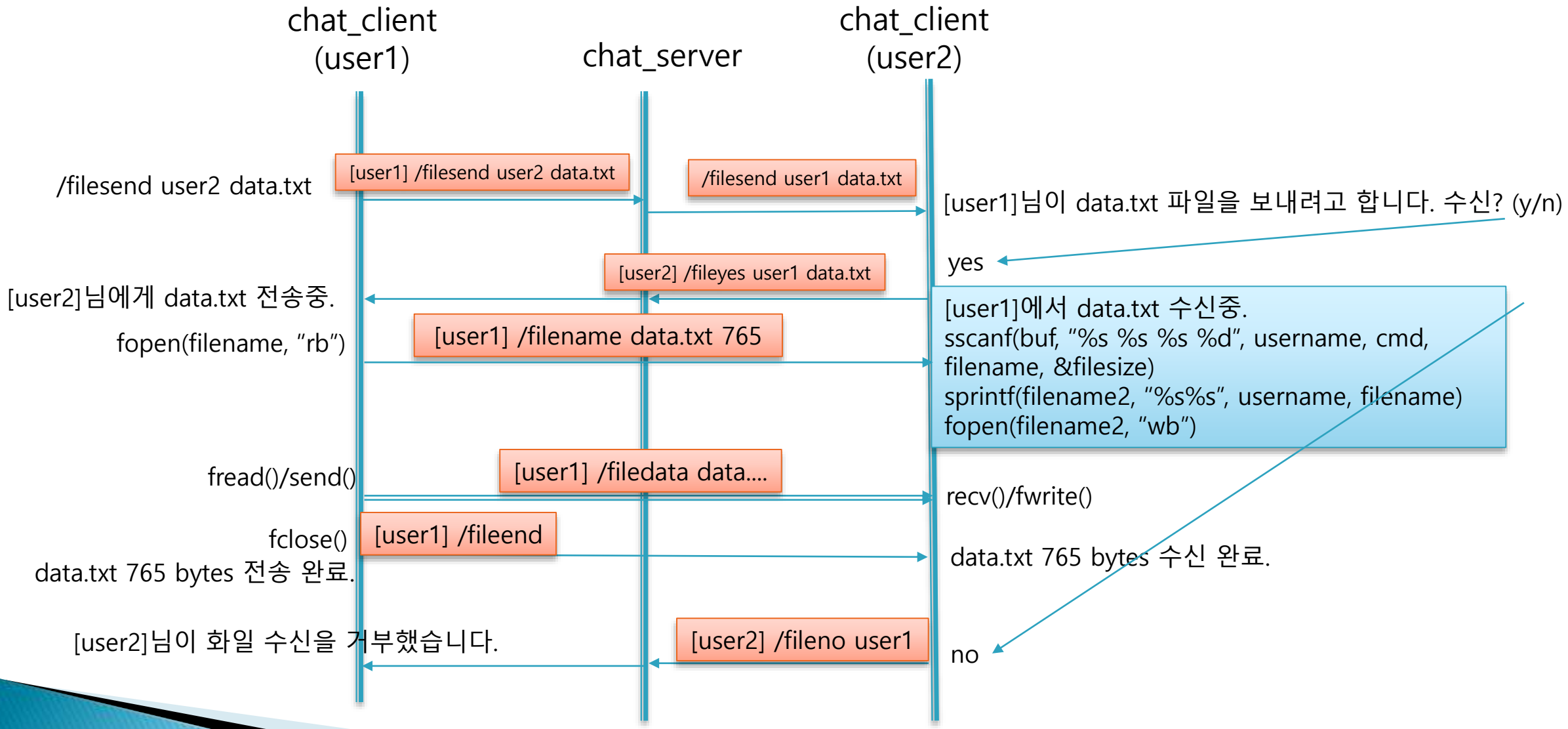

chat_client
(user1)
chat_server
chat_client
(user2)
/with user2
[user1] /with user2
/with user1
[user1]님이 1:1 대화를 요청했습니다. (y/n)
[user2]님과 1:1 대화를 시작합니다.
[user2] /withyes user1
yes
[user1]님과 1:1 대화를 시작합니다.
Hello user1
[user1] Hello user1
[1:1] [user1] Hello user1
[user2] Hi there
[1:1] [user2] Hi there
Hi there
/end
[user1] /end
[user1]님과 1:1 대화를 종료합니다.
[user2]님과 1:1 대화를 종료합니다.
no
[user2]님이 1:1 대화를 거부했습니다.
[user2] /withno user1

6주 도전과제2: 파일 전송 기능 chat_client6/chat_server6

- 명령어 추가
  - /filesend user2 filename
- 파일 전송을 user2가 허락해야 가능하도록 구현

  예) user1 화면 – user1> /filesend user2 data.txt
  user2 화면 – [user1]님이 파일을 보내려고 합니다. 수신 (y/n)? y
  user1 화면 – [user2]님에게 data.txt 전송중
  user1 화면 – data.txt 765 bytes 전송 완료.
  user2 화면 – [user1]에서 data.txt 수신중.
  user2 화면 – data.txt 765 bytes 수신 완료.
  user2 화면 – [user1]님이 파일을 보내려고 합니다. 수신 (y/n)? n
  user1 화면 – [user2]님이 파일 수신을 거부했습니다.
- 파일전송 프로토콜 고려사항
  - Server가 file_client/server 프로토콜을 중계해야 된다.
  - 1:1 대화모드와 같은 방식으로 키입력/화면 대신 file을 사용한다.
- 주의사항
  - 파일전송 테스트할 때 chat_client6를 두 번 실행하면 파일 전송시 같은 폴더에서 같은 폴더로 전송을 시동하기 때문에 오류가 발생할 수 있음.
  - 따라서 수신하는 clien에서 data를 저장할 때 파일 명을 [user1]data.txt 식으로 새로운 파일명을 사용하여야 함.

파일 전송 프로토콜


chat_client
(user1)
chat_server
chat_client
(user2)
/filesend user2 data.txt
[user1] /filesend user2 data.txt
/filesend user1 data.txt
[user1]님이 data.txt 파일을 보내려고 합니다. 수신? (y/n)
yes
[user2] /fileyes user1 data.txt
[user2]님에게 data.txt 전송중.
[user1]에서 data.txt 수신중.
sscanf(buf, "%s %s %s %d", username, cmd, filename, &filesize)
sprintf(filename2, "%s%s", username, filename)
fopen(filename2, "wb")
fopen(filename, "rb")
[user1] /filename data.txt 765
fread()/send()
[user1] /filedata data....
recv()/fwrite()
fclose()
[user1] /fileend
data.txt 765 bytes 수신 완료.
data.txt 765 bytes 전송 완료.
[user2]님이 화일 수신을 거부했습니다.
[user2] /fileno user1
no

참고: 과제를 위한 프로토콜 명령어 정의

```
#define CHAT_CMD_LOGIN   "/login"// connect하면 user name 전송 "/login user1"
#define CHAT_CMD_LIST    "/list"// userlist 요청
#define CHAT_CMD_EXIT    "/exit"// 종료
#define CHAT_CMD_TO      "/to"// 귓속말 "/to user2 Hi there.."
#define CHAT_CMD_SLEEP   "/sleep"// 대기모드(부재중) 설정
#define CHAT_CMD_WAKEUP  "/wakeup"// wakeup 또는 message 전송하면 자동 wakeup
#define CHAT_CMD_WITH    "/with"// [user1] /with user2, user1이 user2와 1:1 대화 요청
#define CHAT_CMD_WITH_YES        "/withyes"// 1:1 대화 허락 [user2] /withyes user1
#define CHAT_CMD_WITH_NO         "/withno"// 1:1 대화 거부 [user2] /withno user1
#define CHAT_CMD_WITH_END        "/end"// 1:1 채팅 종료 [user1] /end or [user2] /end
#define CHAT_CMD_FILESEND        "/filesend"// [user1] /filesend user2 data.txt 파일 전송 요청
#define CHAT_CMD_FILE_YES        "/fileyes"// [user2] /fileyes user1 파일 전송 허락
#define CHAT_CMD_FILE_NO         "/fileno"// [user2] /fileno user1 파일 수신 거부
#define CHAT_CMD_FILE_NAME       "/filename"// [user1] /filename data.txt 765 파일 정보 전달
#define CHAT_CMD_FILE_DATA       "/filedata"// [user1] /filedata data...
#define CHAT_CMD_FILE_END        "/fileend"// [user1] /fileend 파일전송 끝
```

1:1 채팅 + 파일전송 chat_client5.c

- 프로그램 구조 변경
  - do_recv() 함수 추가
    - server message 처리 함수

chat_client3.c
기존 방식

```
while (1) {
    memset(buf1, 0, BUF_LEN);
    memset(buf2, 0, BUF_LEN);
    // Non-blocking read이므로 데이터가
    n = recv(s, buf2, BUF_LEN, 0);
    if (n > 0) { // non-blocking read
    // network에서 읽어서
    // 화면에 출력
        printf("%s", buf2);
    }
```

```
while (1) {
    // Non-blocking read이므로 데이터가 없으면
    memset(buf, BUF_LEN, 0);
    n = recv(s, buf, BUF_LEN, 0);
    if (n > 0) { // non-blocking read
        do_recv(s, buf);

    }
```

변경된 방식

```
char username[BUF_LEN]; // user name
void read_key_send(int s); // key입력후 보내는 code (Linux/Windows공용)
void do_recv(int s, char* buf); // recv() 처리 공통 함수
void do_with(int s, char *swith, char* user1); //
void do_filesend(int s, char* user1, char* filename);
void do_fileyes(int s,  char* user2, char* sfileyes, char *user1, char *filename);
void do_filename(int s, char* user1, char* sfilename, char* filename, char* sfilesize);
void do_filedata(int s, char* user1, char* sfiledata, char* buf);
void do_fileend(int s, char* user1, char *sfileend);
FILE* fp_send, * fp_recv; // 파일 전송, 수신 용
int file_recv_filesize; // 수신할 파일 크기
int file_recv_readsum;
int file_recv_toread; // filesize - readmsum
```

chat_client5.c do_recv()

```c
// recv() 처리
void do_recv(int s, char* buf) {
    char buf1[BUF_LEN] = { 0 }, buf2[BUF_LEN] = { 0 },
        buf3[BUF_LEN] = { 0 }, buf4[BUF_LEN] = { 0 };
    // [user1] Hello there...  정상적인 message buf1=[user1]
    // /with user1 ==> 1:1 대화 요청 (Server가 중계한 명령어) buf1=/with
    // /filesend user1 data.txt ==> 파일 전송 요청
    // [user2] /fileyes user1 data.txt
    // [user1] /filename data.txt 765
    // [user1] /filedata data....
    // [user1] /fileend
    (void)sscanf(buf, "%s %s %s %s", buf1, buf2, buf3, buf4);

    // 1:1 대화 요청
    // /with user1 buf1 = /with // buf2 = user1
    if (strcmp(buf1, CHAT_CMD_WITH) == 0) {
        do_with(s, buf1, buf2);
    }
    else if (strcmp(buf1, CHAT_CMD_FILESEND)==0) {
        // 파일 전송 요청 들어오면
        // /filesend user1 data.txt 이렇게 서버가 보내준다.
        do_filesend(s, buf2, buf3);
    }
```

```c
    else if (strcmp(buf2, CHAT_CMD_FILE_YES) == 0) {
        // 파일 전송 수락
        // [user2] /fileyes user1 data.txt
        do_fileyes(s, buf1, buf2, buf3, buf4);
    }
    else if (strcmp(buf2, CHAT_CMD_FILE_NAME) == 0) {
        // 파일 정보 수신
        // [user1] /filename data.txt 765
        do_filename(s, buf1, buf2, buf3, buf4);
    }
    else if (strcmp(buf2, CHAT_CMD_FILE_DATA) == 0) {
        // 파일 data 수신
        // [user1] /filedata data....
        do_filedata(s, buf1, buf2, buf);
    }
    else if (strcmp(buf2, CHAT_CMD_FILE_END) == 0) {
        // 파일 수신 종료 처리
        // [user1] /fileend
        do_fileend(s, buf1, buf2);
    }
    else {
        // 일반 채팅 메시지
        // 화면에 출력
        printf("%s", buf);
    }
}
```

chat_client5.c 1:1 대화 처리 do_with()

```c
// 1:1 대화 처리
void do_with(int s, char* buf1, char* user1) {
    char buf[BUF_LEN];
    char answer;
    do {
        printf("[%s]님이 1:1 대화 요청을 했습니다. (y/n) : ", user1);
        fgets(buf, BUF_LEN, stdin);
        answer = toupper(buf[0]);
    } while (answer != 'Y' && answer != 'N');
    if (answer == 'Y') { // 대화 허락
        sprintf(buf, "[%s] %s %s\n", username, CHAT_CMD_WITH_YES, user1);
    }
    else { // 대화 거부
        sprintf(buf, "[%s] %s %s\n", username, CHAT_CMD_WITH_NO, user1);
    }
    if (send(s, buf, BUF_LEN, 0) < 0) {
        printf("send error.\n");
        exit(0);
    }
}
```

chat_client5.c 파일전송 처리 do_filesend()

```c
// 파일 전송 요청 처리
// user1이 보낸 [user1] /filesend user2 data.txt 를
// 서버가 /filesend user1 data.txt 로 변경한다.
void do_filesend(int s, char* user1, char *filename) {
    char buf[BUF_LEN + 1] = { 0 };
    char answer;
    do {
        printf("[%s]님이 %s 파일을 보내려고 합니다. 수신 (y/n) : ", user1, filename);
        fgets(buf, BUF_LEN, stdin);
        answer = toupper(buf[0]);
    } while (answer != 'Y' && answer != 'N');
    if (answer == 'Y') { // 대화 허락
        sprintf(buf, "[%s] %s %s %s\n", username, CHAT_CMD_FILE_YES, user1, filename);
    }
    else { // 파일 수신 거부
        sprintf(buf, "[%s] %s %s\n", username, CHAT_CMD_FILE_NO, user1);
    }
    if (send(s, buf, BUF_LEN, 0) < 0) {
        printf("send error.\n");
        exit(0);
    }
}
```

chat_server5.c 1:1 모드 + 파일 전송

```c
char userlist[MAXCLIENTS][BUF_LEN]; // user name 보관용
int userwith[MAXCLIENTS]; // 1:1 대화 상대 보관
int userstatus[MAXCLIENTS]; // sleep 상태인지 상태 보관
// Online, Sleep, Private(1:1) Online, Private Sleep 상태 정보
char* str_userstatus[] = { "O", "S", "PO", "PS" };
#define BIT_SLEEP   0x1 // Online 이면 0 Sleep 이면 1
#define BIT_PRIVATE 0x2 // 1:1 이면 0x2
#define IsOnline(n) (!(userstatus[n]&BIT_SLEEP))
#define IsPrivate(n) (userstatus[n]&BIT_PRIVATE)
#define SetSleep(n) (userstatus[n]|=BIT_SLEEP)
#define SetOnline(n) (userstatus[n]&=(~BIT_SLEEP))
#define UnsetSleep(n) (userstatus[n]&=(~BIT_SLEEP))
#define SetPrivate(n) (userstatus[n]|=BIT_PRIVATE)
#define UnsetPrivate(n) (userstatus[n]&=(~BIT_PRIVATE))
int userfile[MAXCLIENTS]; // 파일 송/수신 상대 보관
```

```c
if ((n = recv(client_fds[i], buf, BUF_LEN, 0)) <= 0) {
    // client 가 비 정상 종료한 경우
    printf("recv error for client[%d]\n", i);
    client_error[i] = 1;
    continue;
}
printf("received %d from client[%d] : %s", n, i, buf);
// /login username --> buf1=/login, buf2=username
// [user1] /list --> buf1=[user1], buf2=/list
// [user1] /sleep --> buf1=[user1], buf2=/sleep
// [user1] /exit --> buf1=[user1], buf2=/exit
// [user1] /to user2 --> buf1=[user1] buf2=/to, bu3=user3
// [user1] /with user2 --> buf1=[user1], buf2=/with, buf3=user2
// [user1] /filesend user2 data.txt --> buf1=[user1], buf2=/file
sscanf(buf, "%s %s %s %s", buf1, buf2, buf3, buf4);
// /login 처리
if (strcmp(buf1, CHAT_CMD_LOGIN) == 0) { ... }
// /list 처리
if (strcmp(buf2, CHAT_CMD_LIST) == 0) { ... }
// /to 처리 귓속말 처리
// [atom] /to username2
if (strcmp(buf2, CHAT_CMD_TO) == 0) { ... }
// /with 1:1 대화 요청 처리
// [user1] /with user2
if (strcmp(buf2, CHAT_CMD_WITH) == 0) { ... }
// /withyes 1:1 대화 허락 처리
// [user2] /withyes user1
if (strcmp(buf2, CHAT_CMD_WITH_YES) == 0) { ... }
// /withyno 1:1 대화 거부 처리
// [user2] /withyno user1
if (strcmp(buf2, CHAT_CMD_WITH_NO) == 0) { ... }
// /end 1:1 대화 종료
// [user2] /end
if (strcmp(buf2, CHAT_CMD_WITH_END) == 0) { ... }
```

```c
// 파일 전송 요청 처리
// [user1] /filesend user2 data.txt
if (strcmp(buf2, CHAT_CMD_FILESEND) == 0) { ... }
// /fileyes 파일 전송 허락 처리
// [user2] /fileyes user1 data.txt
if (strcmp(buf2, CHAT_CMD_FILE_YES) == 0) { ... }
// /fileno 파일 수신 거부 처리
// [user2] /fileno user1
if (strcmp(buf2, CHAT_CMD_FILE_NO) == 0) { ... }
// 파일 정보 (filename + filesize) 전달 처리
// [user1] /filename data.txt 765
if (strcmp(buf2, CHAT_CMD_FILE_NAME) == 0) { ... }
// 파일 data 전달 처리
// [user1] /filedata data....\0
if (strcmp(buf2, CHAT_CMD_FILE_DATA) == 0) { ... }
// 파일 전송 종료 처리
// [user1] /filend
if (strcmp(buf2, CHAT_CMD_FILE_END) == 0) { ... }
// /exit 처리
// exit 대신 /exit 로 변경.
if (strcmp(buf2, CHAT_CMD_EXIT) == 0) { ... }
// /sleep 처리
if (strcmp(buf2, CHAT_CMD_SLEEP) == 0) { ... }
```

chat_server5.c 1:1 대화 처리 /with, /withyes

```c
// /with 1:1 대화 요청 처리
// [user1] /with user2
if (strcmp(buf2, CHAT_CMD_WITH) == 0) {
    SetOnline(i); // user 입력이 있으면 무조건 Online으로 변경한다.
    //char username[BUF_LEN], with[BUF_LEN], with_user[BUF_LEN];
    //sscanf(buf, "%s %s %s", username, with, with_user);
    char* username, * with, * with_user;
    username = buf1;
    with = buf2;
    with_user = buf3;

    printf("[1:1 대화 요청] from %s to %s\n", username, with_user);
    // 1:1 대화 수락 전송
    for (j = 0; j < num_chat; j++) {
        // user가 sleep 이면 어떤 메시지도 수신하지 않는다.
        // 이미 1:1 모드이면 요청을 보내지 않는다.
        if (strcmp(userlist[j], with_user) == 0) {
            if (!IsPrivate(j) && IsOnline(j)) {
                // "/with user1"
                memset(buf2, 0, BUF_LEN);
                sprintf(buf2, "%s %s\n", CHAT_CMD_WITH, userlist[i]);
                if (send(client_fds[j], buf2, BUF_LEN, 0) < 0) {
                    printf("client[%d] send error.", j);
                    client_error[j] = 1;
                }
            }
            break;
        }
    }
    continue;
}
```

```c
// /withyes 1:1 대화 허락 처리
// [user2] /withyes user1
if (strcmp(buf2, CHAT_CMD_WITH_YES) == 0) {
    //char username[BUF_LEN], withyes[BUF_LEN], with_user[BUF_LEN];
    //sscanf(buf, "%s %s %s", username, withyes, with_user);
    char* username, * withyes, * with_user;
    username = buf1;
    withyes = buf2;
    with_user = buf3;
    printf("[1:1 대화 허락] from %s to %s\n", username, with_user);
    // 1:1 대화 허락 전송
    for (j = 0; j < num_chat; j++) {
        // user가 sleep 이면 어떤 메시지도 수신하지 않는다.
        if (IsOnline(j) && strcmp(userlist[j], with_user) == 0) {
            memset(buf2, 0, BUF_LEN);
            sprintf(buf2, "%s님과 1:1 대화를 시작합니다.\n", username);
            if (send(client_fds[j], buf2, BUF_LEN, 0) < 0) {
                printf("client[%d] send error.", j);
                client_error[j] = 1;
                break;
            }
            memset(buf2, 0, BUF_LEN);
            sprintf(buf2, "[%s]님과 1:1 대화를 시작합니다.\n", with_user);
            if (send(client_fds[i], buf2, BUF_LEN, 0) < 0) {
                printf("client[%d] send error.", j);
                client_error[i] = 1;
                break;
            }
            SetPrivate(i); // Private 1:1 mode 설정
            SetPrivate(j);
            userwith[i] = j; // i = user2
            userwith[j] = i; // j = user1
            break;
        }
    }
    continue;
}
```

chat_server5.c 1:1 대화 처리 /withno, /end

```c
// /withyno 1:1 대화 거부 처리
// [user2] /withyno user1
if (strcmp(buf2, CHAT_CMD_WITH_NO) == 0) {
    //char username[BUF_LEN], withno[BUF_LEN], with_user[BUF_LEN];
    //sscanf(buf, "%s %s %s", username, withno, with_user);
    char* username, * withno, * with_user;
    username = buf1;
    withno = buf2;
    with_user = buf3;
    printf("[1:1 대화 거부] from %s to %s\n", username, with_user);
    // 1:1 대화 거부 전송, userwith[] 에는 변화가 없다.
    for (j = 0; j < num_chat; j++) {
        // user가 sleep 이면 어떤 메시지도 수신하지 않는다.
        if (IsOnline(j) && strcmp(userlist[j], with_user) == 0) {
            memset(buf2, 0, BUF_LEN);
            sprintf(buf2, "%s님이 1:1 대화를 거부했습니다.\n", username);
            if (send(client_fds[j], buf2, BUF_LEN, 0) < 0) {
                printf("client[%d] send error.", j);
                client_error[j] = 1;
                break;
            }
            break;
        }
    }
    continue;
}
```

```c
// /end 1:1 대화 종료
// [user2] /end
if (strcmp(buf2, CHAT_CMD_WITH_END) == 0) {
    char *username, *withend;
    if (userwith[i] == -1) // 1:1 모드가 아니다.
        continue;
    //sscanf(buf, "%s %s", username, withend);
    username = buf1;
    withend = buf2;
    printf("[1:1 대화 종료] from %s with %s\n", username, userlist[userwith[i]]);
    j = userwith[i]; // 대화 상대
    // 1:1 모드 종료
    UnsetPrivate(i);
    UnsetPrivate(j);
    userwith[i] = -1; // i = /end 요청 user1 이라면
    userwith[j] = -1; // j = 대화 상대 user2
    memset(buf2, 0, BUF_LEN);
    sprintf(buf2, "%s님과 1:1 대화를 종료합니다.\n", username);
    if (IsOnline(j) && send(client_fds[j], buf2, BUF_LEN, 0) < 0) {
        printf("client[%d] send error.", j);
        client_error[j] = 1;
        break;
    }
    memset(buf2, 0, BUF_LEN);
    sprintf(buf2, "[%s]님과 1:1 대화를 종료합니다.\n", userlist[j]);
    if (IsOnline(i) && send(client_fds[i], buf2, BUF_LEN, 0) < 0) {
        printf("client[%d] send error.", j);
        client_error[i] = 1;
        break;
    }
    continue;
}
```

학생 PC chat_client <WiFi> <ISP> 교수 Windows chat_server

14.38.71.116 30000