Operating System

Multiple Access Chatting Program using Multithread

Han Jun Bae

Korea University

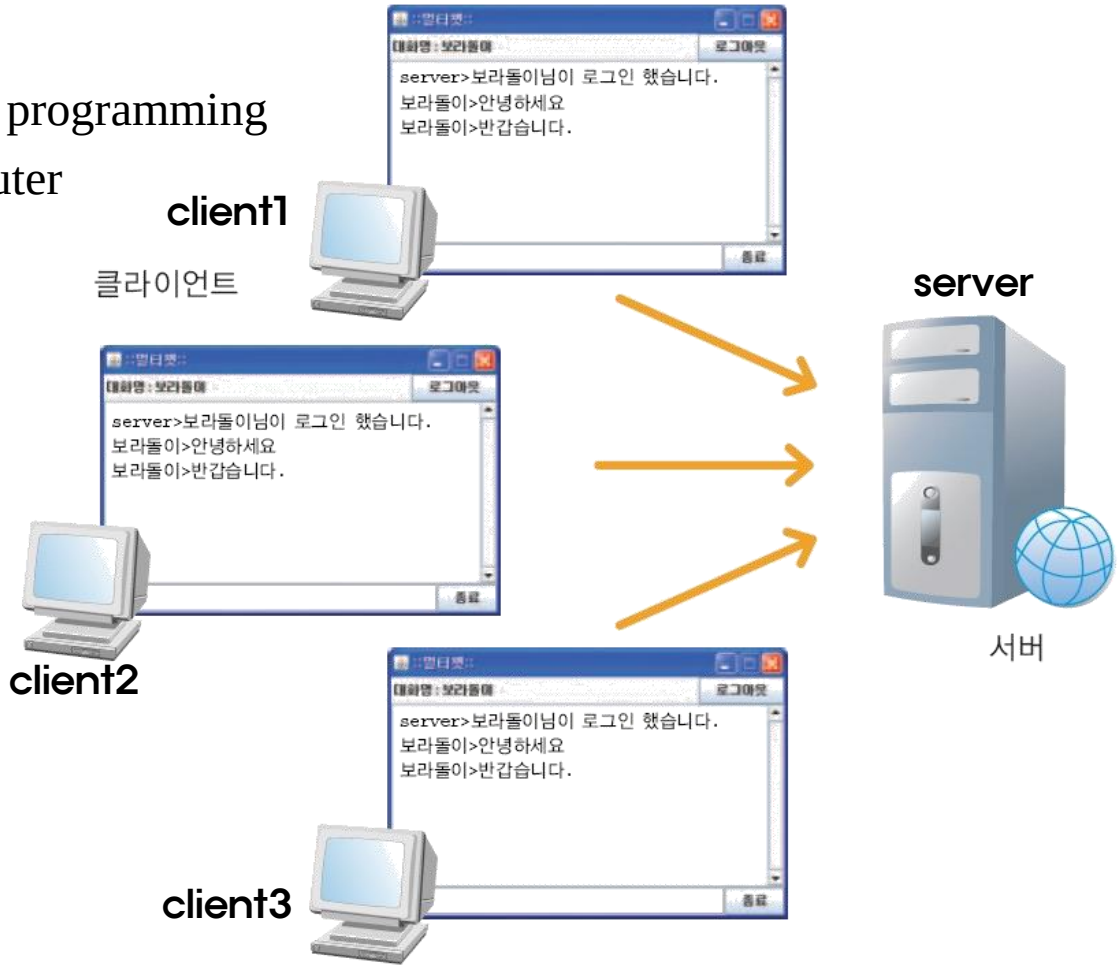
高麗大學校

1905

Computer System Laboratory

Project purpose

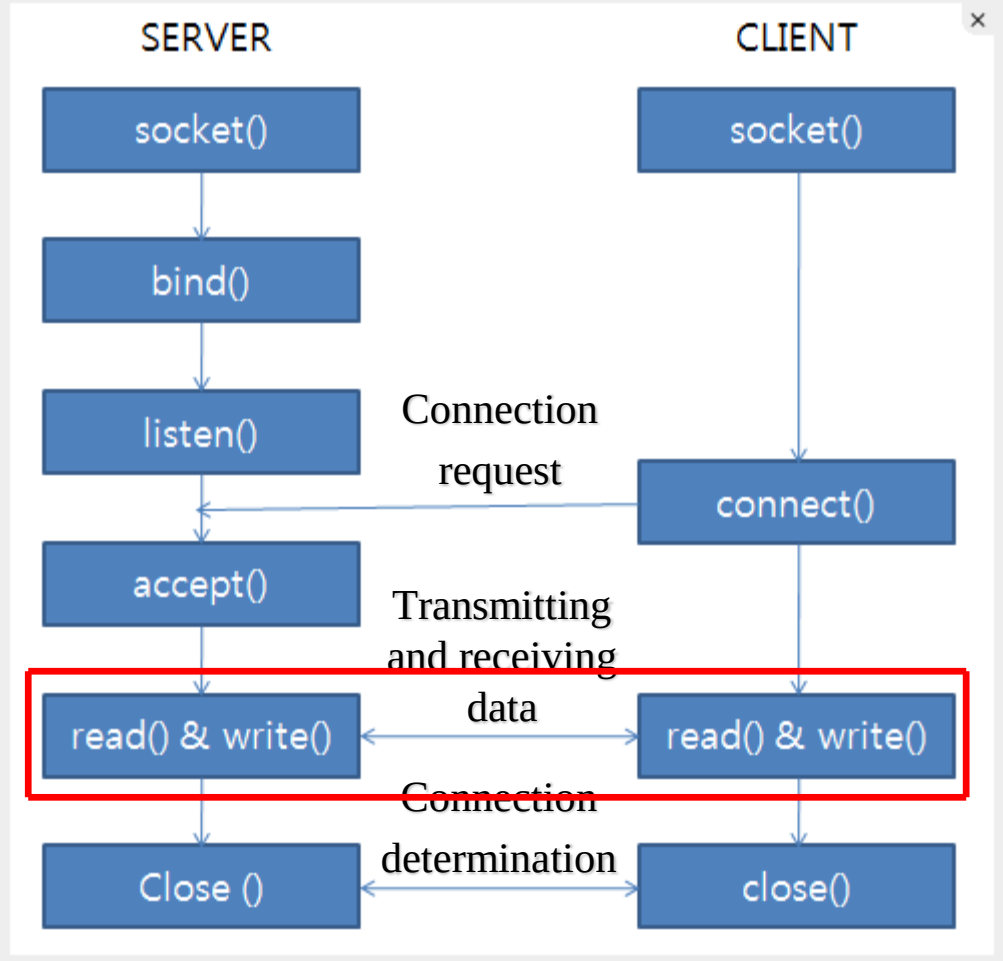
- **Main purpose of this Project**
  - Understand Thread
  - Develop to linux through C programming
  - Enable to test on one computer (IP – 127.0.0.1)



Function of socket program

- Function of program
  - Server program
    - Relay to chatting message between client and client.
    - Enable to network connect many clients at the same time using multithread.
    - Every message should be delivered to every connected client.
  - Cilent program
    - Should be able to connect to the server when you input the server IP and PORT.
    - This program is able to input the message and output the message received from server.
  - You can programming it if you know how to use the network related socket function even if you don’t have the enough knowledge about network.

Configuration of server and client



Purchase phone

Assigned phone number

Listening

Pick up the phone

Conversation

End of conversation

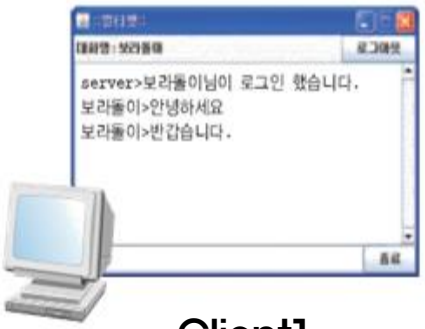
Purchase phone

Make a call

Conversation

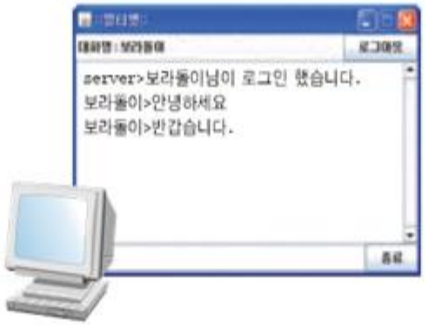
End of conversation

Multithread through chatting program

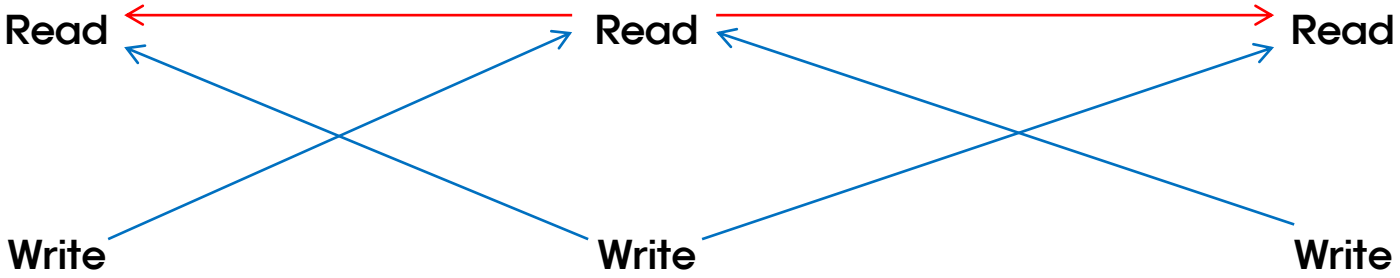


Client1

Server



Client2



Received messages from the client is sent to all clients.

Send message.

Chatting program example

Server

```
user@user-System-Product-Name:~/zz/Project_example_file$ ./server.o
Server Port : 1234
open!! server
Chatting on
conneted to Client 1
conneted to Client 2
message from client 2 : hi
message from client 1 : hello
```

Client 1

```
user@user-System-Product-Name: ~/zz/Project_example_file
user@user-System-Product-Name:~/zz/Project_example_file$ ./client.o
Input Server IP Address : 127.0.0.1
Input Server Port Number : 1234
open!! client
Sucesss to connect to server!
Server : welcome to chatting server!!
 Chatting On...
input 'Q' to exit
Client 2 has been connected!
Client 2 : hi
hello
```

Client 2

```
user@user-System-Product-Name: ~/zz/Project_example_file
user@user-System-Product-Name:~/zz/Project_example_file$ ./client.o
Input Server IP Address : 127.0.0.1
Input Server Port Number : 1234
open!! client
Sucesss to connect to server!
Server : welcome to chatting server!!
 Chatting On...
input 'Q' to exit
hi
Client 1 : hello
```

pthread

- #include <pthread.h>
  - To use the pthread function, you should be declare the header file.
  - http://pubs.opengroup.org/onlinepubs/7908799/xsh/pthread.h.html

- Compile in Linux
  - gcc –c [compile target].c
  - gcc **–pthread** –o [output file name] [compile target].c
  - **-pthread** command connects library to use thread.
  - ./server.o
  - ./client.o

pthread

- **int pthread_create(pthread_t *th_id, const pthread_attr_t *attr, void* function_name, void *arg);**
  - Create pthread
  - First argument: If the thread is created successfully with the pthread identifier, a thread identification value is given.
  - Second argument: pthread attribute (optional), or NULL if using the default thread attribute
  - Third argument: The function to branch to pthread. Only functions whose return value is of type void * and whose parameters are declared void * are also possible.
    - ex) void * handler (void * arg) {...}
  - Fourth argument: The argument value to be passed to the branch function. Since you do not know which datatype to pass, you can pass it as void and cast it to the original datatype in a function that branches to your situation.
  - Return value: Return 0 if pthread is successfully created

pthread

- **int pthread_join(pthread_t th_id, void** thread_return);**
  - Wait until the specific pthread is terminated, and release the resource when the specific pthread exits.
  - First argument: an identifier that determines which pthread to wait for
  - Second argument : the return value of pthread, which receives the value as a pointer.

- **int pthread_detach(pthread_t th_id);**
  - The pthread with the th_id identifier is independent from the parent pthread.
  - That is, independent pthreads are automatically released at the end without pthread_join ().

pthread example

```c
#include <stdio.h>
#include <unistd.h>
#include <pthread.h>

void* do_loop(void *data)
{
    int i;

    int me = *((int *)data);
    for (i = 0; i < 10; i++)
    {
        printf("%d - Got %d\n", me, i);
        sleep(1);
    }
}

int main()
{
    int       thr_id;
    pthread_t p_thread[3];
    int status;
    int a = 1;
    int b = 2;
    int c = 3;

    thr_id = pthread_create(&p_thread[0], NULL, do_loop, (void *)&a);
    thr_id = pthread_create(&p_thread[1], NULL, do_loop, (void *)&b);
    thr_id = pthread_create(&p_thread[2], NULL, do_loop, (void *)&c);

    pthread_join(p_thread[0], (void **) &status);
    pthread_join(p_thread[1], (void **) &status);
    pthread_join(p_thread[2], (void **) &status);

    printf("programing is end\n");
    return 0;
}
```

```
1 - Got 0
3 - Got 0
2 - Got 0
1 - Got 1
2 - Got 1
3 - Got 1
1 - Got 2
2 - Got 2
3 - Got 2
1 - Got 3
2 - Got 3
3 - Got 3
1 - Got 4
2 - Got 4
3 - Got 4
1 - Got 5
2 - Got 5
3 - Got 5
1 - Got 6
2 - Got 6
3 - Got 6
1 - Got 7
2 - Got 7
3 - Got 7
1 - Got 8
2 - Got 8
3 - Got 8
1 - Got 9
2 - Got 9
3 - Got 9
programing is end
```

Project

- Due date : 11/11(Monday)

- Multiple access chatting program using multithread
  - Both client and server should be able to send and receive messages to each other.
  - Program should be contain concept of multithread.

- A code file [filename.c] : 40%
  - ex) server.c, client.c
  - Include detailed comments inside your code

- A word / hwp document that describes your program : 60%
  - File name is your student number. ex) 2015000001.doc
  - You can use any word program.
  - You should be explain your program in detail.
    (Include your program execution results.)

Project

- All the files are compressed into one file (ex: 2015000001.zip)
- Submit your zip file to kilingki@korea.ac.kr
- If you have any question, please mail me or visit to laboratory (engineering bldg #236)