Computer Architecture and Operating Systems
MONSOON SEMESTER of 2019

INSTRUCTOR: DR. Sambuddho Chakravarty

Multi-User Chat System

A simple implementation of a multi-client chat system has been developed using Sockets in Inter-Process Communication(IPC).

It consists of a Chatroom Server that can serve multiple clients.

Setup:

1. **The server is started by specifying an available port number.**
2. **The clients connect to the server by specifying the same port number.**
3. **Once clients are connected to the server, they can communicate with each other.**

Usage:

1. **To message all other connected users, type directly.**
2. **To message a specific user, start with `@<NAME OF USER TO CHAT>`.**
3. **To exit the chatroom, type `bye` or `exit`.**

Implementation:

1. **Using Sockets, the server and clients communicate with each other.**
2. **Initially, the input expects an available port number to launch the chatroom on.**
3. **In the server file, a socket is created. This socket then binds to the specified address and port number. It then listens for incoming connections.**
4. **Socket() -> Bind() -> Listen()**
5. **On any incoming connection from client, the connection request is checked with the Max Allowed Connections in server, after which it is Accepted.**
6. **In the server, each client is allocated a new thread to operate and communicate on.**
7. **In the client, a similar process of creation of Socket takes place after which the Socket tries to Connect to the server.**
8. **Socket() -> Connect()**
9. **The client has two threads, one for sending messages and another for receiving messages. These work simultaneously, hence solving the issue of synchronization.**
10. **The client sends messages using the `write()` or `send()` call. The client receives messages using the `recv` command.**

Errors:

Any error in the creation of sockets or connecting to the server will be displayed on the screen along with the error message.