1 Aims

In this project we are using Python 2.7 language.We have develop a client server program. In this project there are two program one is Client and other is Server. Client will run on one computer e.g Computer A and server will run on another computer B. We will send a Hello message from Client to Server. When the Server receive Hello message , after listening to client it will send HI to client. client and server will be recognized by their IP address.There can be multiple clients which can send messages at a one. A server explains a socket and binds an IP address and port to it. If binding is successful, then it starts to listen to that socket for incoming connections. A client sends a connection request to related IP and port and the socket server listening to that port accepts it. After the server accepts a client request, a connection is established.

You can send and receive data through that connection while the connection is active. If the client connection is closed, the Server should accept it again. accept step is done once for each new connection and that connection do not needed to be accepted again while it was active.

A server should have a specific IP and port so the client can connect it. A server could not connect to a client since client does not have any socket bound to a specific IP and port for listening incoming connections.

Client 1 sends a message, the message should be received by the server since only servers can accept connections (thus, a client can not establish a connection to another client since another client can not accept any incoming connection). The server receives the message and sent it to client 2 Since a server could not establish a connection to a client, Client 2 should be already connected. Otherwise, the server should wait for the client 2 to establish a connection so it can send data.

From that point of view, all clients should be connected to the server and keep those connections alive. A Client may have an active connection while it was communicating, and close that connection a while after it was idle for a specific time. But the client should establish connections to server and checks if it has any data waiting to be transferred to himself. So keeping the relationship alive looks like a better approach.

There are protocols like TCP and UDP, and you define the socket as a TCP socket or a UDP socket. But these protocols have many details, and for simplicity, I can mention that TCP protocol is error-checked and controls if sent data is received by the receiver application (Instant messaging services

told you whether your message sent is successful or not, so this is TCP). UDP have no error-check mechanism (So you just send the package but do not be confirmed whether it is delivered or not). Client/server describes the relationship between two computer programs in which one program, the client, makes a service request from another program, the server, which fulfils the request. Although programs within a single computer can use the client/server idea, it is a more important idea in a network. In a network, the client/server model provides a convenient way to interconnect programs that are distributed efficiently across different locations.