TCP Protocol Design

The server will check for an incoming argument instructing the server port to be set, otherwise the server port will default to 1234.

The data transfer port will be set to the initial port +1.

TCP Sockets will be made for both content and data with a buffer handled by the TCP protocol in its congestion control which will manage the buffer automatically.

The sockets will be bound to the port.

After these events have been completed, the server will listen for incoming connections and print a statement “Ready to accept a connection..”.

Once a connection is established with a client, the server will accept the client’s socket.

A function on the server side will receive the number of bytes being transmitted through a given socket and will add the received bytes to a temporary buffer. The transmitted string will be encoded into bytes, then sent, then decoded on the client side depending on which command was sent.

A while loop from the client will execute until a “quit” command is sent upon which the control socket will shut down the server. Within this while loop there will be a control socket receive command that will encode whether to expect data or not. If given a valid command, the server will execute the given request and transmit the data until the “quit” command is sent at which point the loop will break with a success message. After this point the socket will be closed.

The format of data sent will be in string format with the size being constrained by the receive buffer that will loop as many times as necessary to get the full data transfer.

The receiving side will know when to start/stop receiving the file when a send command is sent through the control socket from the client to the server and when the control socket closes on the server end. There will be a counter on the server and client side that keeps track of bytes sent that will act as a termination for data transmission or reception. This will necessitate the client/server connection exchanging the byte amount for proper handling.

A function will receive the specified number of bytes from the specified socket and wait to receive all bytes until either the client has closed the socket or the length of an incremented buffer storing the number of bytes received matches the number of bytes expected.