Dakin Oliver

CPE 464 - 3pm

Packet Diagrams

Flag=1



Flag=2



Flag=3



Flag=4



Flag=5





Flag=6

Flag=7



Flag=8



Flag=9



Flag=10



Flag=11



Flag=12

1 per handle



Flag=13



Packet Flow Diagrams

Connection setup (sending handle to server)





Sending a message (%M) to a single client





Sending a message (%M) to multiple clients





Broadcasting a message (%B)



Sending messages if the entered message text is 450 characters.



Listing the handles (%L) (complete flow, including flags 10,11,12,13)



Ending a connection (%E)



Handle Table data structure on the server.

Hash table where the key is the handle name, and data stored is:

socket number

handle name

|  |  |  |
| --- | --- | --- |
| Socket/Handle State on the Server | How can this happen (what can cause this on the server) | If you looked at your socket/handle table, how do you know the socket/handle is currently in this state (what in your data structure would tell you this) |
| Client socket is opened but handle not valid | Client sends an invalid handle | Hashes to a used spot, and has a name already in that hash spot |
| Client Socket open and client handle is valid | Client sends valid handle | Hash spot open or name not already in that spot |
| Socket was opened and had a handle, but then the socket was closed | Client terminated or sent closing packet | Hash no longer in table |

lookUpHandle:

Input: Handle name

Action: search hash table for handle name

Output: True if handle is in the table, otherwise False

deleteHandle:

Input: handle

Action: remove handle and socket from the table

Output: <0 if error occurs

addHandle:

Input: handle

Action: and handle and socket to the table

Output:<0 if error occurs

Hierarchical drawing





Common Code

* createPDU
* sendPDU
* recvPDU
* createFlagPDU - creates a pdu that only contains a 2 byte size, 1 byte flag, and a null character