SYSC 3303A Group 5 TFTP Project - Use Cases

Scenario: Client makes a read request to a Server Instance

Summary: User starts up a new client thread and enters a request. The request is sent to the intermediate host (error handler), copied and sent to a server instance, checked and validated and a response is sent back to the client. If the system is in verbose mode, these steps are printed on arrival and exit to/from each component.

Precondition: The Client and Host (error handler) are all running on the same machine as an instance of the server, alongside the server connection handler.

Main Sequence:

1. A new client is created, and the user is asked for various input settings such as which server to send to, which mode they would like and the directory that the files they wish to transfer are stored
2. The Error Simulator also requires similar input, if the user wishes to test irregular behavior
3. While the client is waiting, the user gives the system a read request and provides the file name as additional input
4. Client creates a request Packet, prints all relevant information about that Packet and sends that request to the Host
5. Host prints upon receiving, then creates an identical Packet to be sent to the Server
6. Server prints upon receiving and checks validity
7. Server sends over the file in similar 512 MB Packets, prints an acknowledgement response and sends back to Host
8. Host prints upon receiving, then reprints and sends back to Client
9. Client prints response and waits for more Packets
10. Steps 3 to 8 are repeated until the Client receives a Packet with a size smaller than 512 MB, signaling the transfer is finished, and the Client closes all Sockets.

Alternative Sequences:

Step 2: User may give the client instance a shutdown command, and the system will quit
Step 6: User may have given an invalid request, in which case the system quits here
Step X: At any step, if the mode is changed from the default ‘verbose’ to ‘quiet’, no printing will occur

Non Functional Requirements:

a) Timeout will occur if the system is inactive for too long
b) The user may enter an invalid request, in which case the system will exit

Postcondition: The data has been sent and a response has been received, and the system is ready to repeat the process until the user quits manually

Scenario: Client makes a write request to a Server Instance

Summary: User starts up a new client thread and enters a request. The request is sent to the intermediate host (error handler), copied and sent to a server instance, checked and validated and a response is sent back to the client. If the system is in verbose mode, these steps are printed on arrival and exit to/from each component.

Precondition: The Client and Host (error handler) are all running on the same machine as an instance of the server, alongside the server connection handler.

Main Sequence:

1. A new client is created, and the user is asked for various input settings such as which server to send to, which mode they would like and the directory that the files they wish to transfer are stored
2. The Error Simulator also requires similar input, if the user wishes to test irregular behavior
3. While the client is waiting, the user gives the system a write request and provides the file name as additional input
4. Client creates a request Packet, prints all relevant information about that Packet and sends that request to the Host
5. Host prints upon receiving, then creates an identical Packet to be sent to the Server
6. Server prints upon receiving and checks validity
7. Server prints an acknowledgement response and sends back to Host
8. Host prints upon receiving, then reprints and sends back to Client
9. Client prints response and begins sending the file in 512 MB sized Packets
10. Steps 3 to 8 are repeated until the Client receives a Packet with a size smaller than 512 MB, signaling the transfer is finished, and the Client closes all Sockets.

Alternative Sequences:

Step 2: User may give the client instance a shutdown command, and the system will quit
Step 6: User may have given an invalid request, in which case the system quits here
Step X: At any step, if the mode is changed from the default ‘verbose’ to ‘quiet’, no printing will occur

Non Functional Requirements:

a) Timeout will occur if the system is inactive for too long
b) The user may enter an invalid request, in which case the system will exit

Postcondition: The data has been sent and a response has been received, and the system is ready to repeat the process until the user quits manually

Scenario: Client makes a request but packets are interfered with

Summary: User starts up a new client thread and enters a request. The request is sent to the intermediate host (error handler) where it may be subject to loss, delay, corruption or duplication. It is then copied and sent to a server instance, checked and validated and a response is sent back to the client. If the system is in verbose mode, these steps are printed on arrival and exit to/from each component.

Precondition: The Client and Host (error handler) are all running on the same machine as an instance of the server, alongside the server connection handler.

Main Sequence:

1. A new client is created, and the user is asked for various input settings such as which server to send to, which mode they would like and the directory that the files they wish to transfer are stored
2. The Error Simulator also requires similar input, if the user wishes to test irregular behavior
3. While the client is waiting, the user gives the system a read request and provides the file name as additional input
4. Client creates a packet with a max size of 512 MB, prints all relevant information about that Packet and sends that request to the Host
5. Host prints upon receiving, then creates two identical Packets to be sent to the Server
6. Server prints upon receiving and checks validity
7. Server sends over the file in similar 512 MB Packets, prints an acknowledgement response and sends back to Host
8. Server receives duplicate Packet and prints information
9. Server recognizes that the packet is a duplicate and does not respond to the extra information (block number mismatch) and terminates sending back an error code
10. Host prints upon receiving the original Packet, then reprints and sends back to Client
11. Client prints response and terminates upon receiving error code

Alternative Sequences:

Step 2: User may give the client instance a shutdown command, and the system will quit

Step 3: User may opt to lose, or delay a Packet instead of duplicating, in which case the appropriate behavior will occur at this step, and the server will refuse to acknowledge erroneous data

Step 3: User may have given a write request, in which case steps are similar to the prior use case outlined above.

Step 5: User may have interfered with the RRQ or WRQ itself, in which case the Host delays that packet rather than the DATA or ACK Packets

Step 6: User may have given an invalid request, in which case the system quits here

Step 9: If testing other errors, such as corruption or loss, the server will respond with the appropriate error code, and only continues as normal if the Error simulator modified the packet to result in an unknown transfer ID (error code 5).

Step X: At any step, if the mode is changed from the default ‘verbose’ to ‘quiet’, no printing will occur

Non Functional Requirements:

a) Timeout will occur if the system is inactive for too long
b) The user may enter an invalid request, in which case the system will exit

Postcondition: The data has been sent and a response has been received, all irregular behavior has been resolved and the system is ready to repeat the process until the user quits manually