# Rough Implementation:

**Server parent thread**

Initialize variables

Continuous loop: {

Accept new socket. If it’s a new socket check by reading IP address and tracking) then spawn new server child thread pass argument with socket ID. Threads must be detached.

}

}

**Server child thread**

Initialize JSON tree, variables and client states

Continuous loop: {

Check if event on socket {

If directory exploration command, look up the JSON tree, determine the max amount of the JSON tree you can put into one message and send it to socket

Else if request for file: map JSON tree to local directory tree. Fetch file into buffer and send to socket supported by custom Selective repeat mechanism. Start timer.

Else if ACK for a sequence number, mark the respective packet as received, move window forward.

}

Check if event on timer {

If expired: Send latest unACK’d outstanding packet supported by custom Selective Repeat. Reset Timer.

}

}

**Objectives:**

* Custom Selective Repeat implementation right before call to socket send
* Integrating TFTP with the custom Selective Repeat
* JSON tree dynamic construction
* Command Line interface for directory tree traversal
* GUI (If time permits)
* Putting it all together

Useful resources:

TFTP implementations-

<https://github.com/labcoder/simple-tftp>

<https://github.com/Corvinus96/TFTP-Server>

<https://github.com/crossbowerbt/tftpserver>

Concurrent processes on server:

<https://stackoverflow.com/questions/15731924/udp-multi-client-server-basics>

JSON tree implementation:

<https://github.com/contiki-os/contiki/tree/master/apps/json>

Selective Repeat in C:

<https://github.com/viclai/Reliable_UDP>