1. Client creates packet
2. Client prints info
3. Client sends from send/rec port to port 23
4. Intermediate Host receives packet
Port 23
5. Host prints info
6. Host creates new packet
7. Host sends data on server
8. Server receives packet Port 69
9. Server validates packet

10. Server prints out received data

Port "w"
Port "z"
Port "x"
Port "y"

repeat 11 times
one.
close socket

18. Client prints data
17 client receives packet
16. Host sends packet to client and closes socket
15 Host creates reply packet
14 Host prints reply data
13. Host receives reply
12 server sends reply and closes socket.
11. server creates response packet

Client

#SendPacket : DatagramPacket
#receivePacket : DatagramPacket
#sendReceiveSocket : DatagramSocket

+Client() : Client
+SendAndReceive (opcode: int) : void
+main (args: String[]) : void

Intermediate

#sendPacket : DatagramPacket
#receivePacket : DatagramPacket
#serverSideSocket : DatagramSocket
#receiveSocket : DatagramSocket
#sendSocket : DatagramSocket

+Intermediate() : Intermediate
+forward() : void
+main (args: String[]) : void

Server

#sendPacket : DatagramPacket
#receivePacket : DatagramPacket
#receiveSocket : DatagramSocket
#sendSocket : DatagramSocket
+readAck : byte[]
+writeAck : byte[]

+Server() : Server
+receiveAndReply() : void
+main (args: String[]) : void

Message

+formatRequest (filename: String, format: String, opcode: int) : byte
+printIncoming (p: DatagramPacket, name: String) : void
+printOutgoing (p: DatagramPacket, name: String) : void
+validate (data: String) : boolean

<<use>>
<<use>>
<<use>>
<<use>>

:Intermediate

1. <<create>>
5. send (send Packet)
8. receive (receive Packet)
14. close()

Server Side Sokt
Datagram Socket

2. <<create>>
4. receive (receive Packet)
17. close()

receive Socket
Datagram Socket
{port = 23}

3. <<create>>
7. <<create>>

receive Packet
Datagram Packet

5. <<create>>
9. <<create>>

send Packet
Datagram Packet

10. <<create>>
11. send (send Packet)
12. close()

Reply Socket
Datagram Socket

1. <<create>>
3. receive (receivePacket)
8. close()

:Server

receiveSocket:
DatagramSocket
{port = 69}

2. <<create>>

5. <<create>>

4. <<create>>
6. send(sendPacket)
7. close()

sendSocket:
DatagramSocket

receivePacket
DatagramPacket

sendPacket
DatagramPacket

:Client

1. <<create>>
3 send (send Packet)
5 receive (receive Packet)
6 close ()

Send Receive Socket :
Datagram Socket

2 <<create>>

Send Packet :
Datagram Packet

4 <<create>>

receive Packet :
Datagram Packet