Carleton University

SYSC 3303 TEAM 8

Iteration 5

Trivial File Transfer Protocol

Jeremy Sawh	100940268
Jonathan Chan	100936881
Obinna Elobi	100953254
Yan Liao	100940287
Yash Patel	100943654

April 4, 2016

Contents

1 Description of Included Files

1.1 Client.java

Client issues a read request or write request for file transfers to the server.

1.2 ErrorSim.java

ErrorSim generates different error cases for each data packet that passes through it. These errors are sent back to the client or server. Simulated errors are sent via a new port created by ErroSim.java

1.3 ErroSimHolder.java

Creates a new thread in order to send duplicate packets.

1.4 ErrorType.java

This class contains holds the errors the user selected to be queue's up. It contains methods that can be called from the Errorsim.java class to gather more information about the packet. For example, packet number and which type of packet the error must be simulated on.

1.5 Server.java

Starts a listener threads that polls for a 'Shutdown' Command

1.6 InvalidRequestException.java

Inherits java exception message. Not used.

1.7 printByteArray.java

This class prints the packet information in Hex

1.8 ReadRequestHandler.java

This class handles all read requests(RRQ). When the request packet arrives its fields are verified (mode and filename). If vaild the filename specified is read from and packaged into a DATA packet. When an ACK is received the address, port, block number and op-code of the ACK are verified and if valid the transfer continues, until there is less than 512 bytes left to be read.

1.9 ReceivedPacketHandler.java

This file opens a new socket to receive following packets. It verifies the packets opcode and passes the packets to the appropriate handler. WriteRequestHandler or ReadRequestHandler.

1.10 WriteRequestHandler.java

This class handles all write requests(WRQ). When the request packet arrives its fields are verified (mode and filename). If valid a file is created with the specified file name. When a DATA packet arrives the address, port, block number, and op-code are verified. If valid the data is then written to created file until the DATA packet is less that 516 bytes.

1.11 ServerListener.java

Listens on port 69 for a packet. Once it receives a packet its starts a ReceivedPacketHander thread to process the packet.

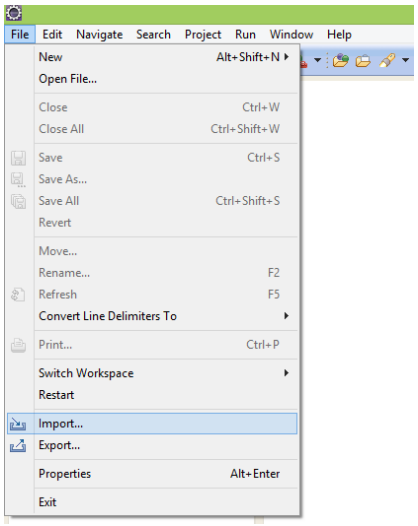
1.12 ErrorMessagesHandler

Collection of Errors used in created error packets to be sent when and error is discovered

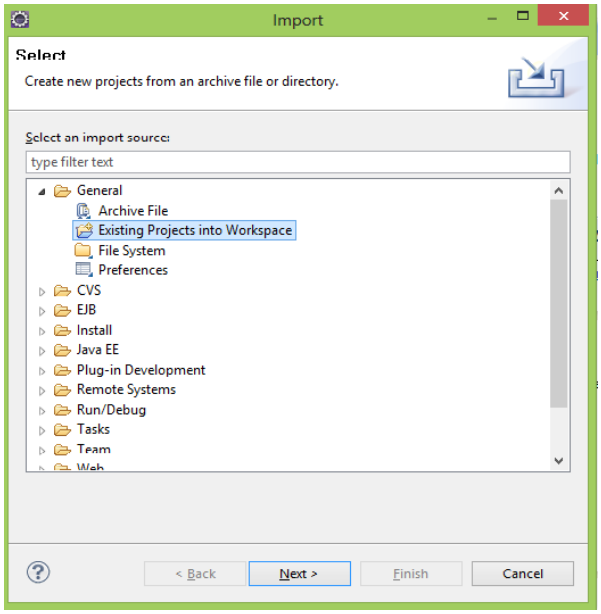
2 Setup Instructions

These are the following instructions to adding this existing project to your Eclipse IDE.

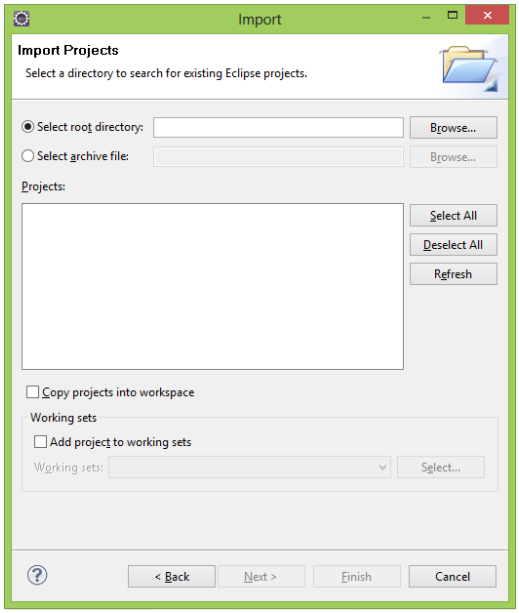
1. From the main menu select FILE¿IMPORT the wizard will open



2. Collapse the general tab and select add "Existing Project into workspace"



3. Select either root directory or archive file and browse to the directory containing those the project. Make sure to check copy projects into workspace



4. Click finish to import the files. You should see the following files in your workspace:

    - Server.java
    - ErrorSim.java
    - Client.java
    - PrintByteArray.java
    - ErrorMessagesHandler.java
    - WriteRequestHandler.java
    - ReadRequestHandler.java
    - ReceivedPacketHandler.java
    - InvalidRequestException.java
    - ErrorType.java
    - ServerListener.java

2.1 Run Instructions

Run the following files through eclipse Respectively:

1. Run Server.java
2. Run ErrorSim.java
3. Run Client.java

3 Operating the Client

The client prompts the user to select a mode. There are two different modes: Mode 1 is normal mode sending packets directly to the server. Mode 2 is the Testing mode that directs all packets to the error simulator for testing To operate the client follow the instructions below.

1. Select a mode for operation

   1– Normal Mode

   2– Test Mode

   3– Shutdown

2. Enter the server's IP address or type "local" to run all files on a local machine. The client will prompt the user with the option to change the IP address once a file transfer completes.

3. If "normal mode" is selected:

   A) Choose either RRQ or WRQ

   B) Type in file-name to read from

   C) Type in file name to write to

   D) Transfer files

4. If "Test mode" is selected:

   A) Choose errors from menu in error simulator

   B) then go back to client and preform steps in 3.

4 Operating the Error Simulator

The error simulator provides a number of options for generating errors. The error's are used to test how the client and server will respond in the event one of these error's occur in normal file transfer. The options available to error testing are as follows.

0 Do nothing

1 Invalid Operation (Changes the first 2 bytes to 99)

2 Wrong block number (Changes the 3rd and 4th byte)

3 Remove Zero (removes the last byte)

4 Change Mode (Change ASCII or octet)

5 Missing file name

6 Invalid Packet size (Change the file size to 1024)

7 Invalid TID

8 Duplicate packet

9 Packet lost

10 Delay Packet

Once you have selected your desired operation type 999 to finish. Multiple errors can be selected before the file transfer begins. They will be queued up.

NOTE: If Error Simulator fails to respond re-run the error simulator.

5 TroubleShooting

5.1 Errorcode 1 - File Not Found

WRQ: The Client does not make a request packet if the file does not exist. It displays the error and asks the user for the file-name again.

RRQ: The Client sends a read request packet to Server and if the Server cannot find the file, it sends an error packet (error code 1) and the transfer stops.

5.2 Errorcode 2 - Access Violation

WRQ: The Client sends a write request and if the Server cannot access the file, it will send an error packet (error code 2) back and the transfer will stop.

RRQ: The Client sends a read request and then if it cannot access the file, it displays "ACCESS VIOLATION" and quits the transfer. The Server times out since it does not receive any packets.

One way to simulate this is when the client asks for the file to write to (the second filename), enter a path to a read-only folder. When this is done the Server(for WRQ) or the Client(for RRQ) won't be able to write to it, causing an ACCESS VIOLATION error. For a visual representation of this error see figure 16

5.3 Errorcode 3 - Disk Full

WRQ: The transfer goes on as normal until the Server cannot write because the disk got full. It sends an error packet (error code 3) to the Client and the transfer stops.

RRQ: The transfer goes on as normal until the Client cannot write because the disk got full. The Client prints an error message saying the disk is full and quits the transfer. The Server times out and quits on its end too.
One way to simulate this is - to write to a USB that has less space than the file being written. This will cause an error when the disk gets full mid-transfer. For a visual representation of this error see figure 6.

5.4 Errorcode 4 - Illegal TFTP Operation

Invalid TFTP Operation

WRQ and RRQ: When the server receives a packet, either a request packet, data packet or ACK (Acknowledge) in which the op-code does not correspond to the current TFTP operation, the server will send an error packet containing details about the operation and the transfer will end.

Incorrect Block Number

WRQ and RRQ: When the client receives an ACK or DATA packet with a block number less than what is expected the packet is ignored and the previous packet is re-transmitted. When the server receives a DATA packet with a block number less than what is expected the server sends a error packet to the client and the transfer is shut-down. When the server receives an ACK packet with a block number less that what is expected the packet is ignored and the server re-transmits the previous packet. For a visual representation of this error see figure 10.

If the client or server receives an ACK or DATA packet with a block number greater than what is expected the server sends an error packet containing details about the operation and the client then shuts down the transfer.

Missing File Name

WRQ and RRQ: When the server receives a request packet with no file-name it sends and error packet to the client and the transfer stops. Below is a Timing diagram representing what happens in the event the filename is missing. For a visual representation of this error see figure 14.

Incorrect Request Packet Format

WRQ and RRQ: When the server receives a request packet with missing zero's, invalid packet size, or incorrect mode the server will send an error packet to the client and the transfer will stop. For a visual representation of this error see figure 1 in section 7 diagrams.

5.5 Errorcode 5 - Unknown Transfer ID

WRQ and RRQ: When a an unknown TID is received the recipient will send an error packet and the transfer will stop. The sender's socket will then close. See figure 11.

5.6 Errorcode 6 - File Already Exist

WRQ: The Client sends a write request and once the Server sees that the file it is supposed to write to already exists, it sends an error packet (error code 6) and the the transfer stops. See figure 9.

RRQ: If the file the client writes to exist, it will be overwritten.

5.7 Additional Notes

Note: Duplicate request packets do not have desired results for outputs. The transfers stop due to other errors not because of the duplicate packet.

6 Member Responsibilities

ITERATION 1	
Jeremy	Implementation of the client
Yan	ErrorSim, Diagrams and test cases
Obinna	Implementation of the client
Jonathan	Implementation of the Server
Yash	ErrorSim, Diagrams and test cases

ITERATION 2	
Jeremy	Client, Debugging, Readme
Yan	ErrorSim, Client, Diagrams cases
Obinna	Client, Debugging, Diagrams
Jonathan	Server, Client, Diagrams
Yash	ErrorSim, Diagrams, Testing cases

ITERATION 3	
Jeremy	Diagrams, Readme, testing
Yan	Error simulator, debugging cases
Obinna	Client, debugging, testing
Jonathan	Server, debugging, testing
Yash	ErrorSim, debugging, Testing

ITERATION 4	
Jeremy	Readme, Client
Yan	Client, debugging cases
Obinna	Diagrams, debugging
Jonathan	Server, debugging
Yash	ErrorSim, debugging

ITERATION 5	
Jeremy	Readme, debugging
Yan	Client, debugging
Obinna	Client, debugging
Jonathan	Server, debugging
Yash	Diagrams, Errorsim

7 Diagrams

7.1 Timing Diagrams

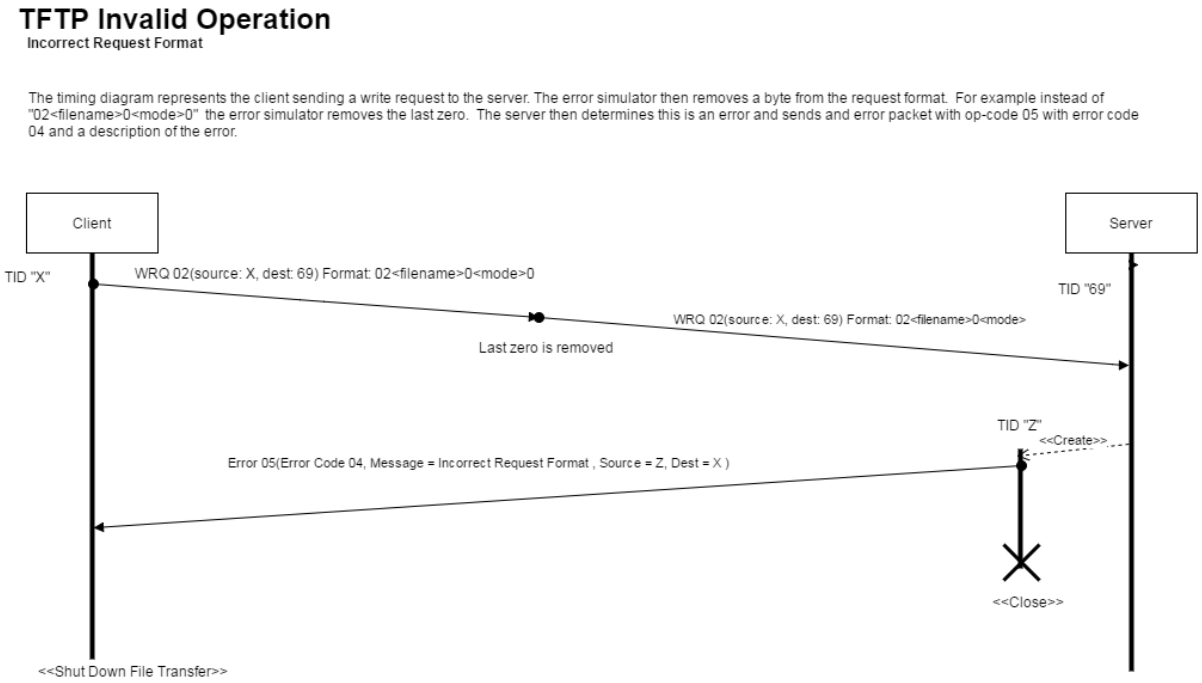


Figure 1: Incorrect Request Format

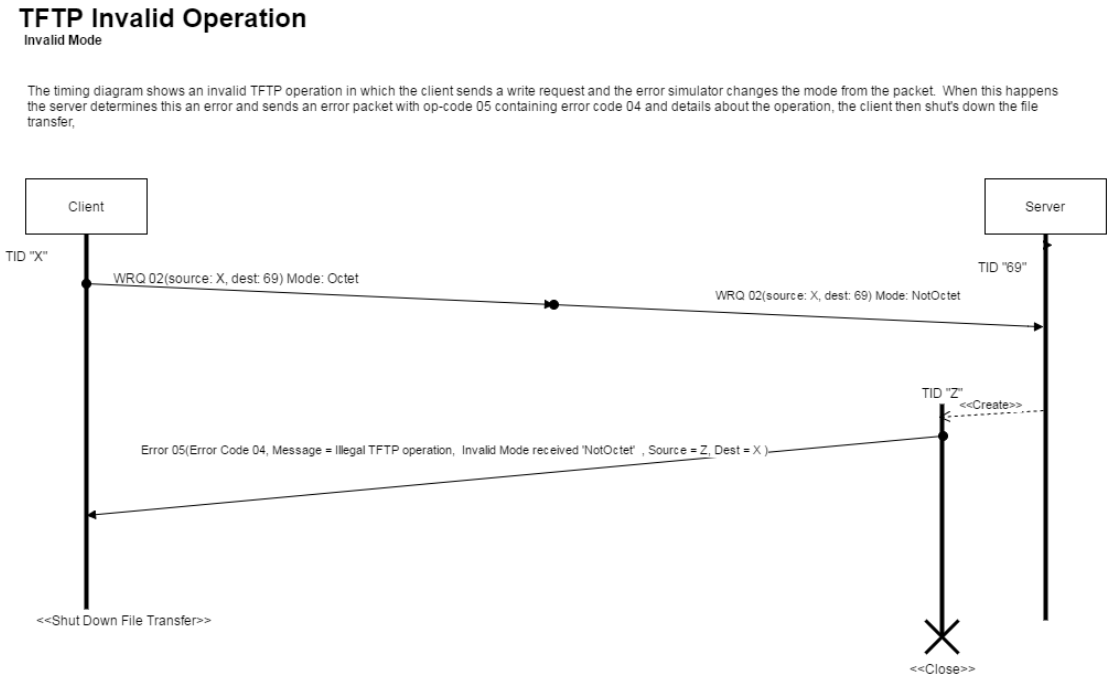


Figure 2: Invalid Mode

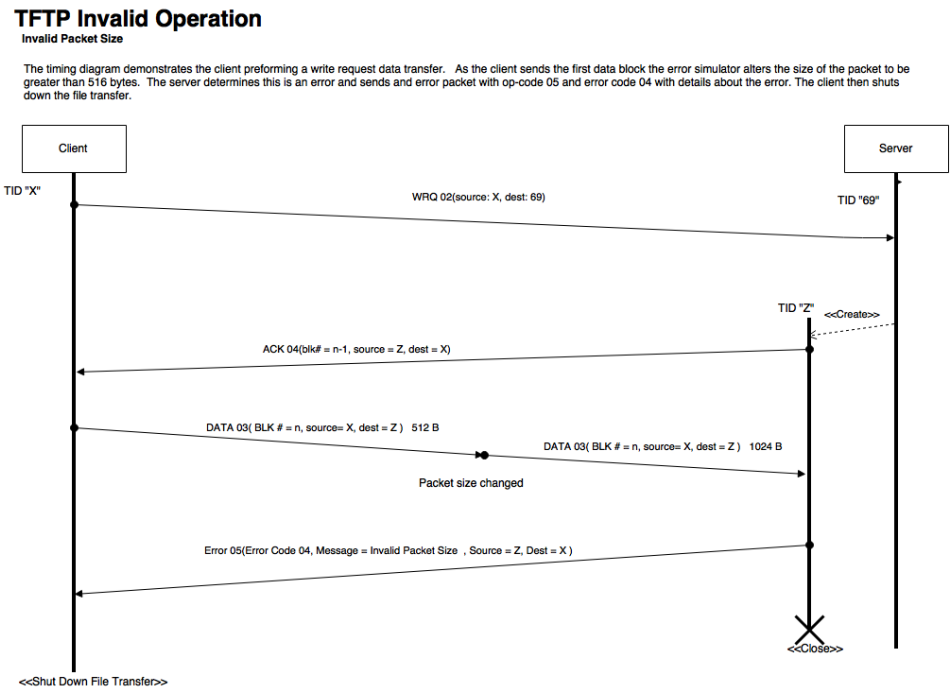


Figure 3: Invalid Packet Size

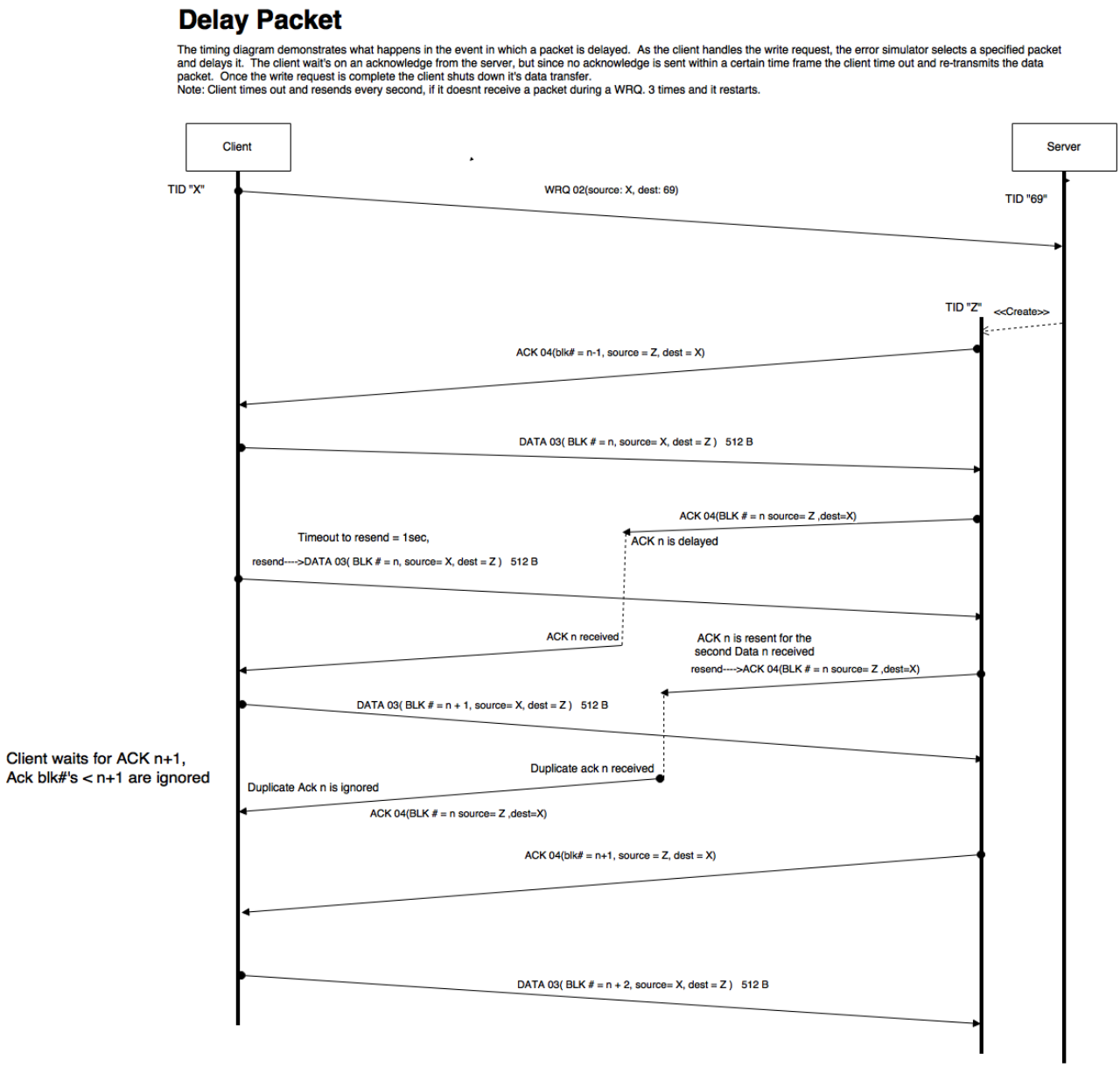


Figure 4: DelayPacket

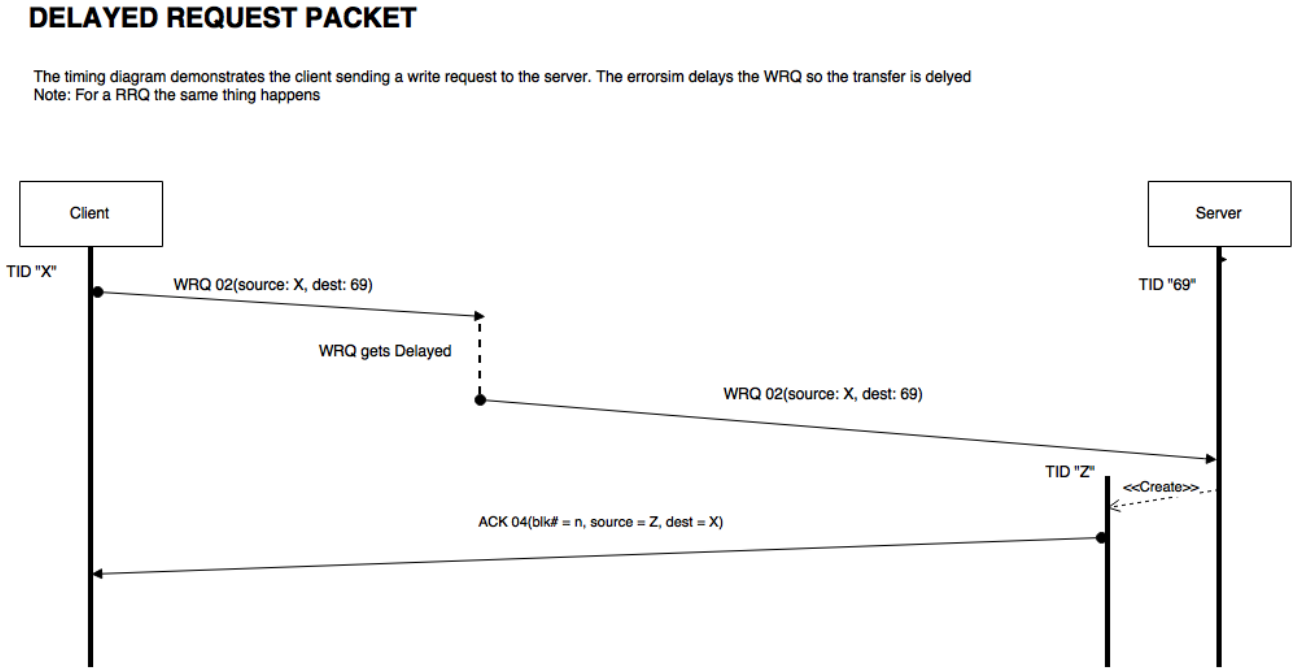


Figure 5: Delayed Request packet

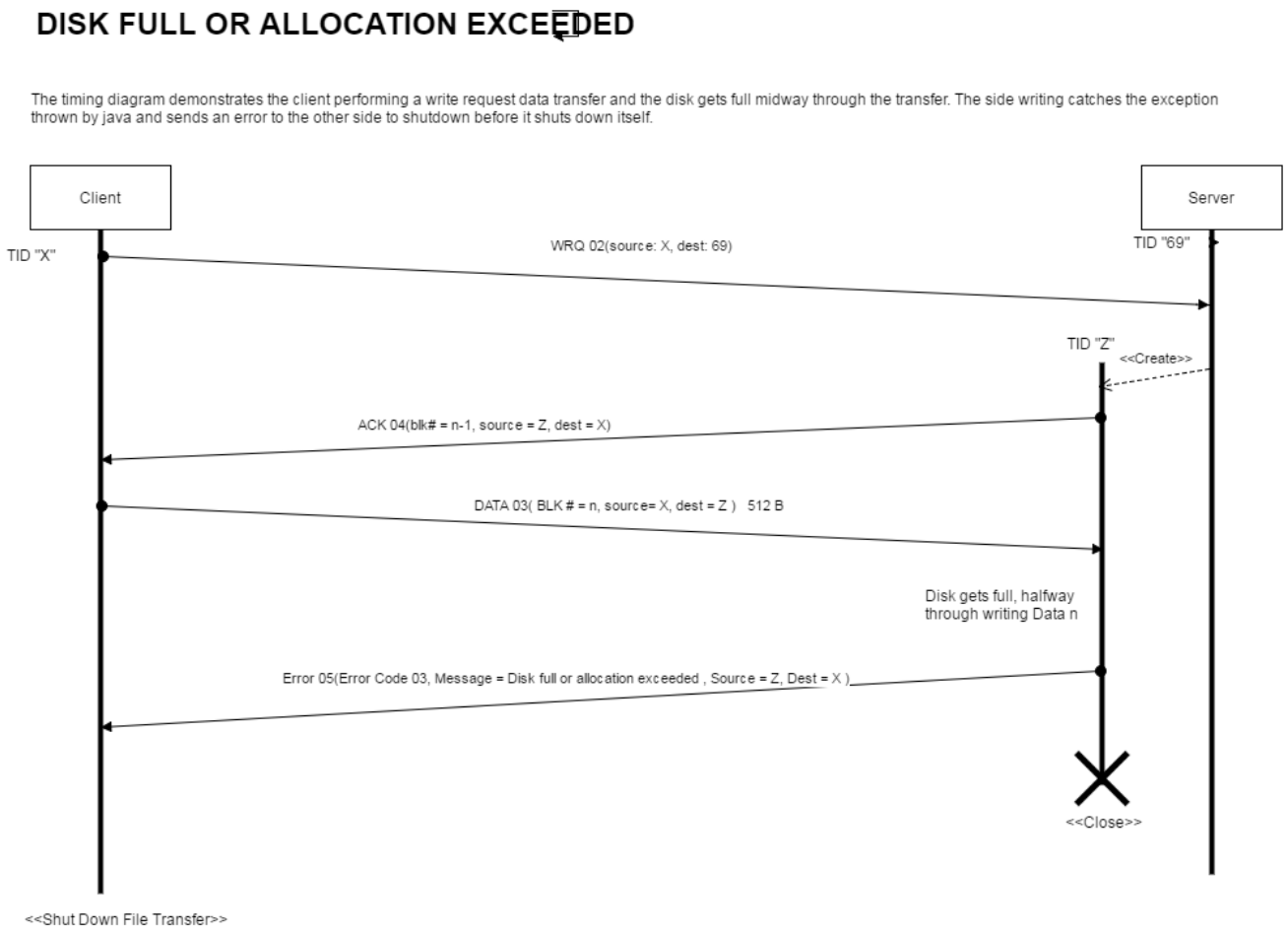


Figure 6: Disk Full

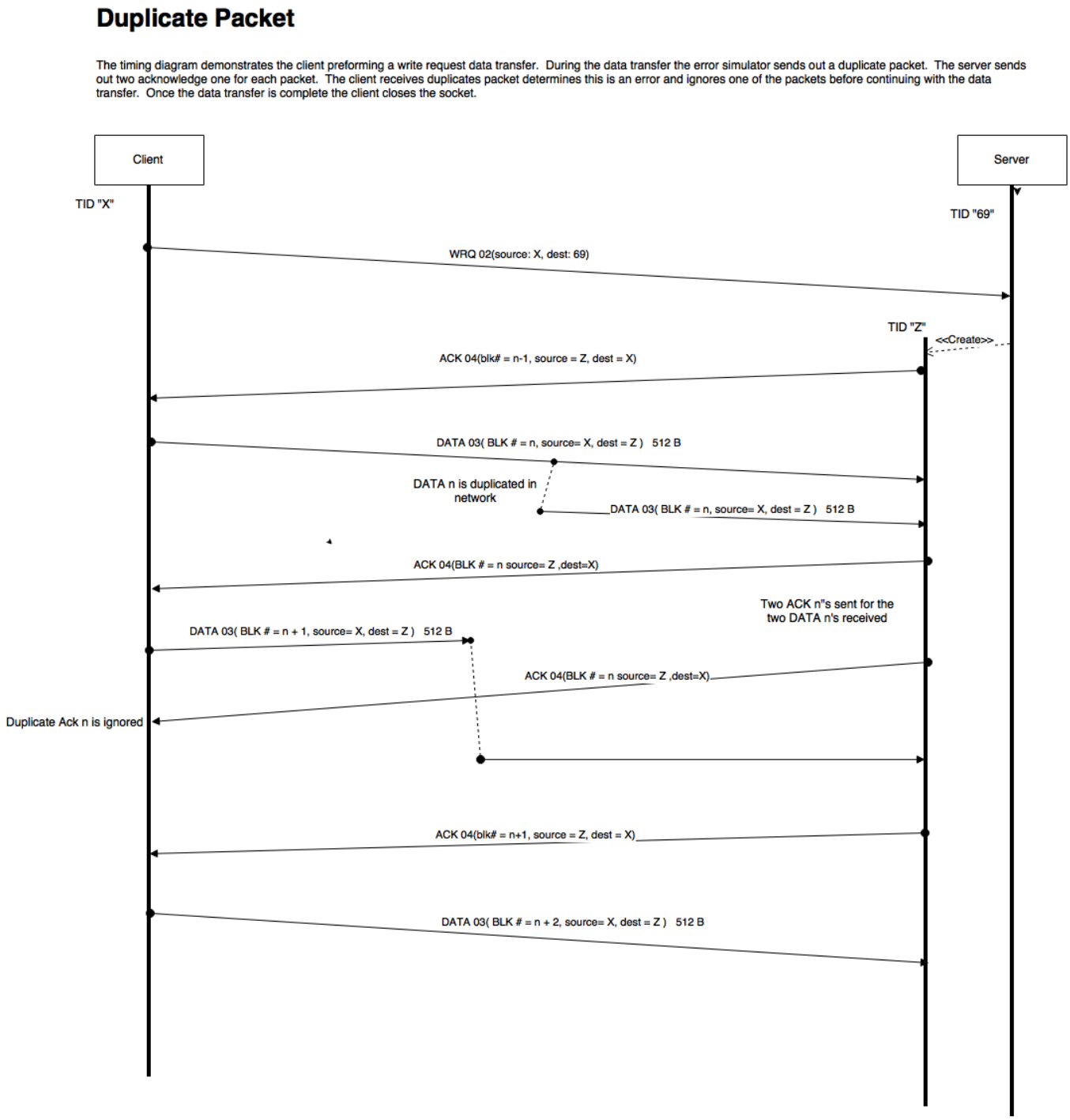


Figure 7: Duplicate packet

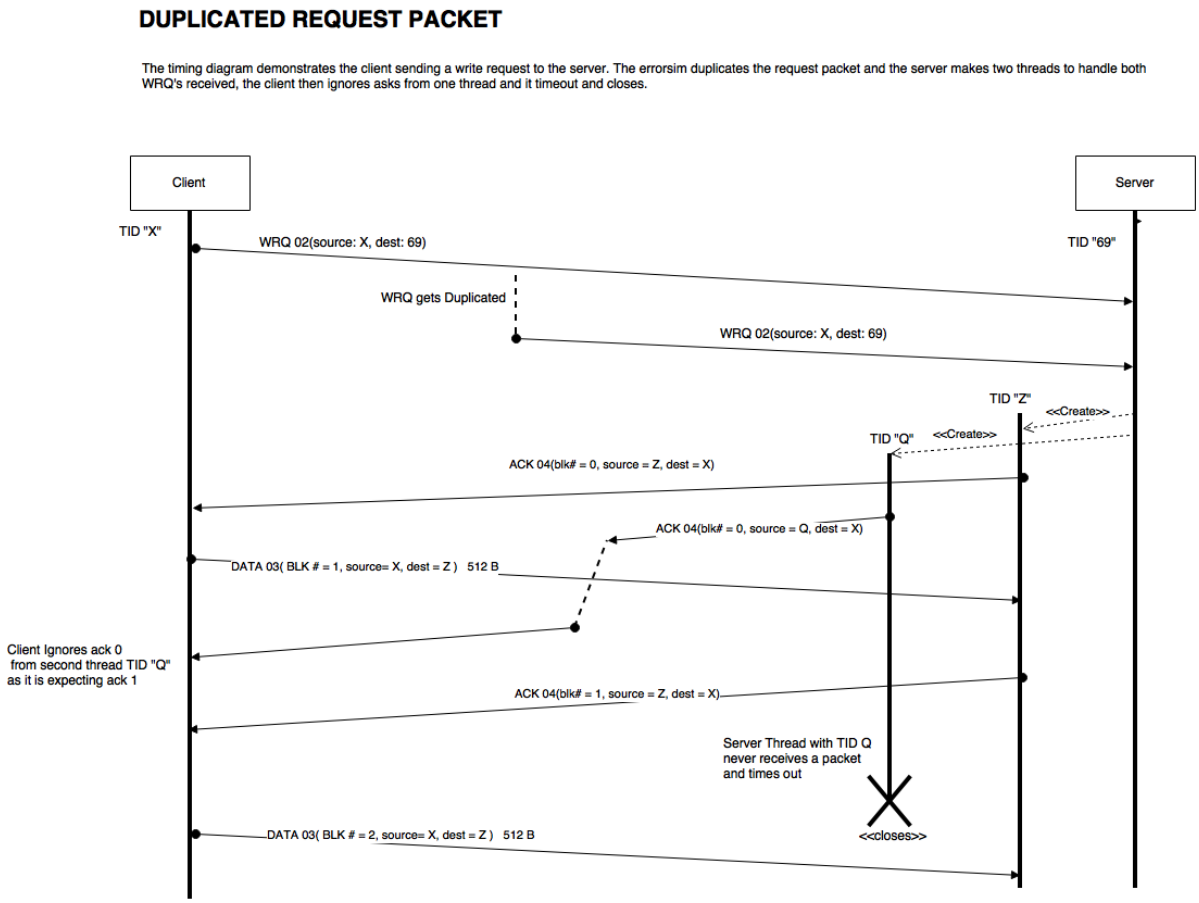


Figure 8: Duplicate Request packet

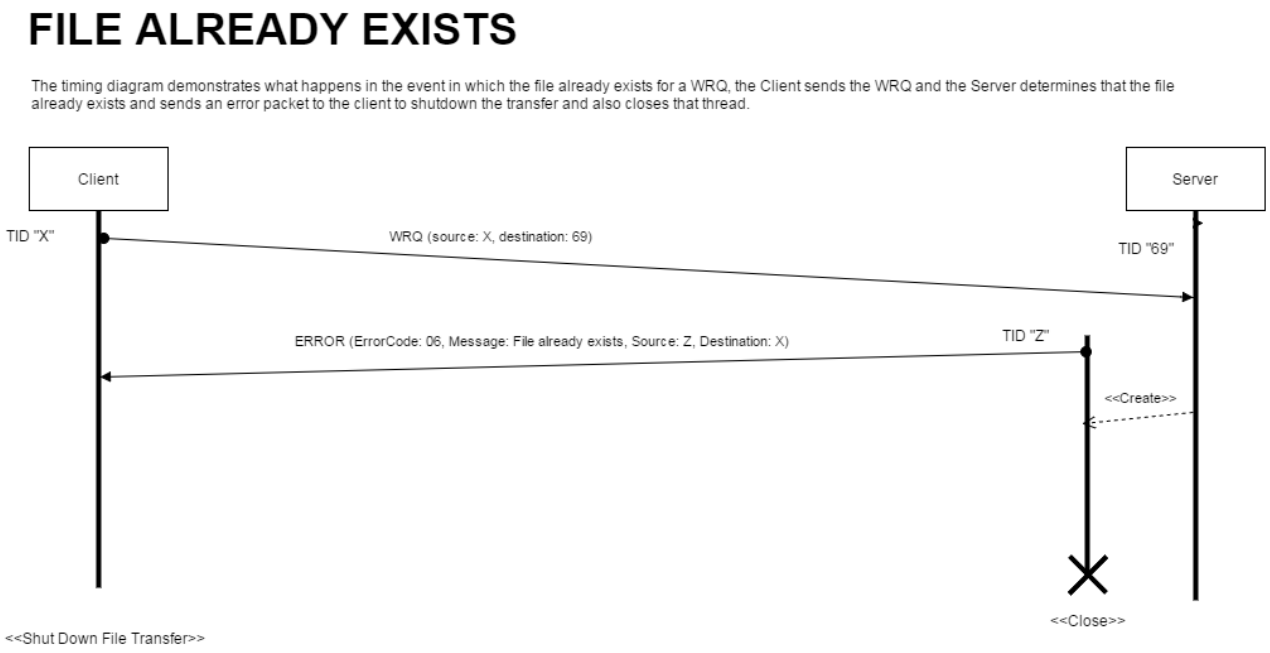


Figure 9: File Already Exists

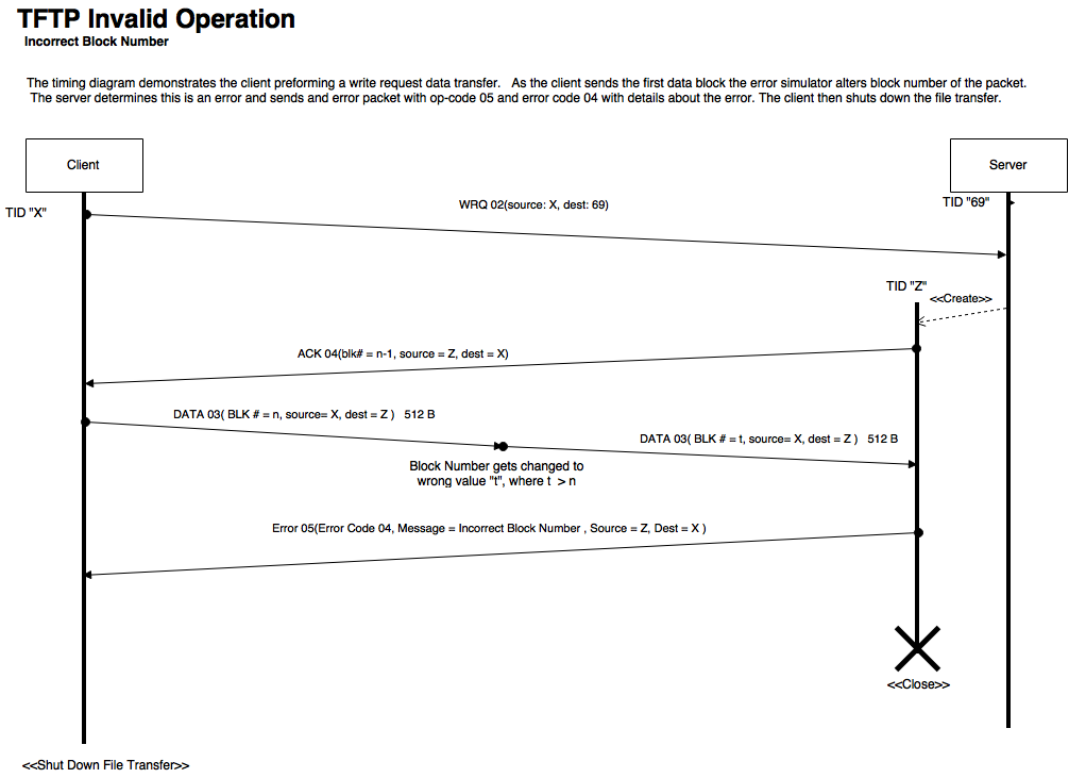


Figure 10: Incorrect Block Number

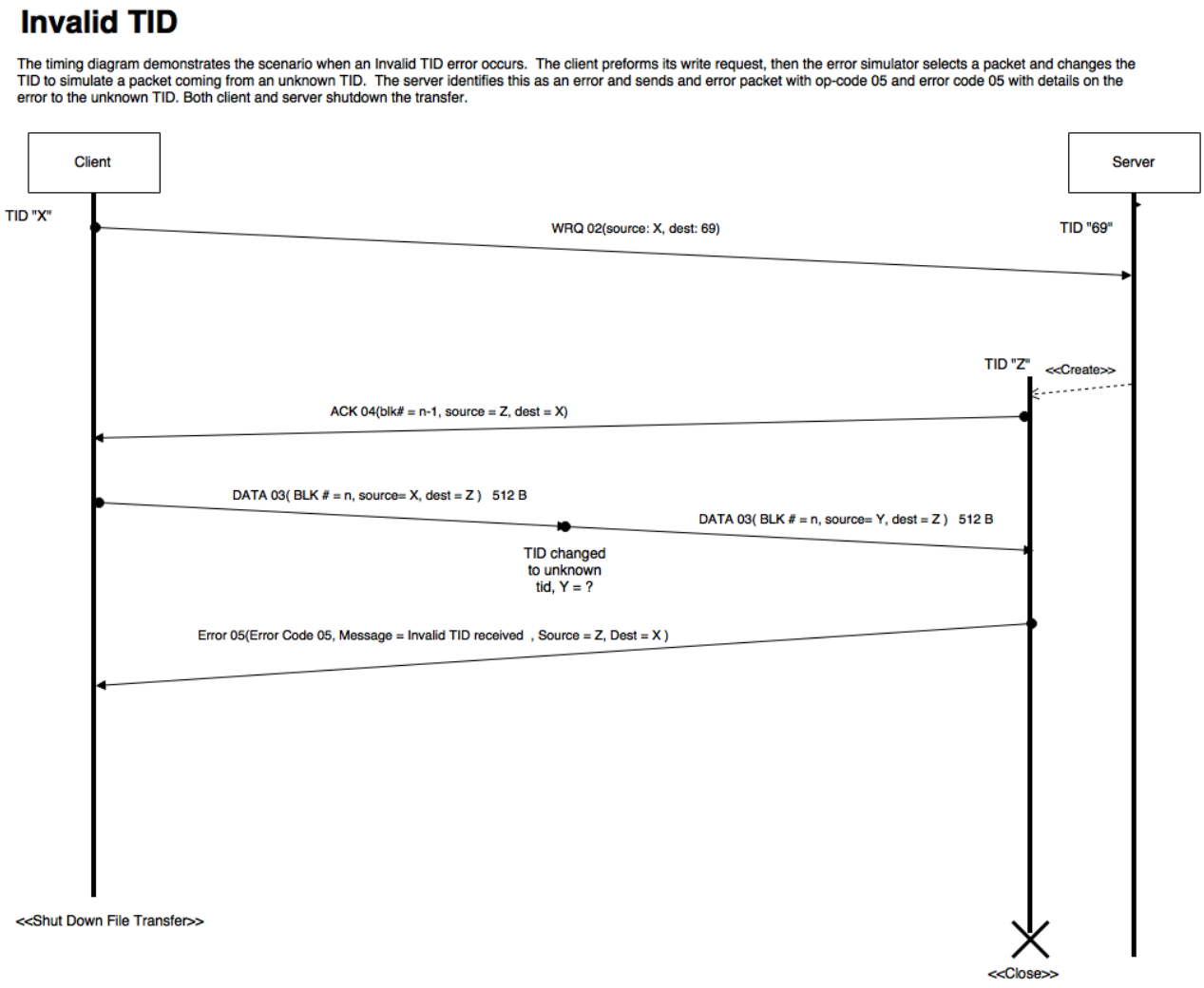


Figure 11: Invalid TID

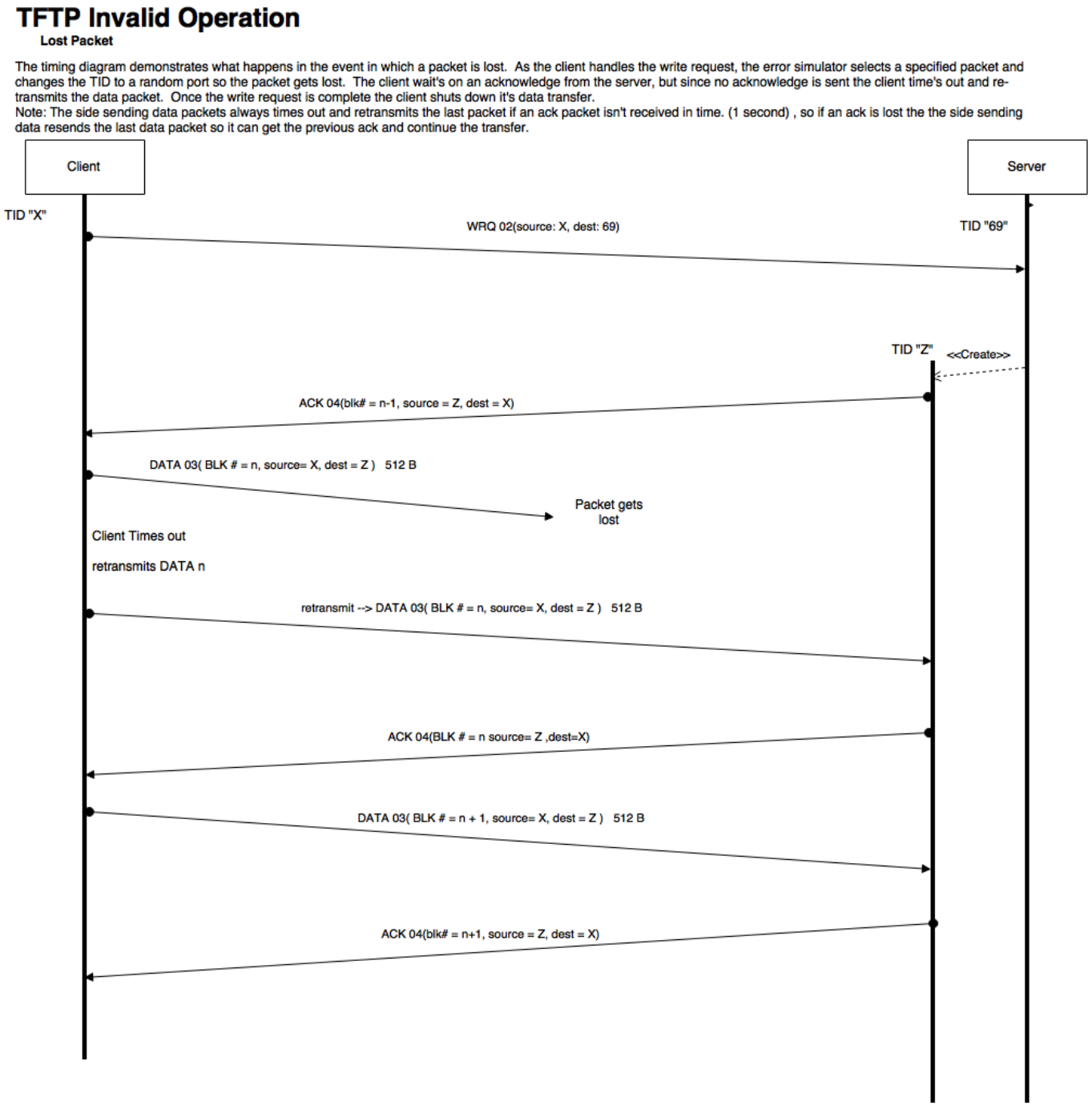


Figure 12: Lost packet

Figure 13: Lost Request packet

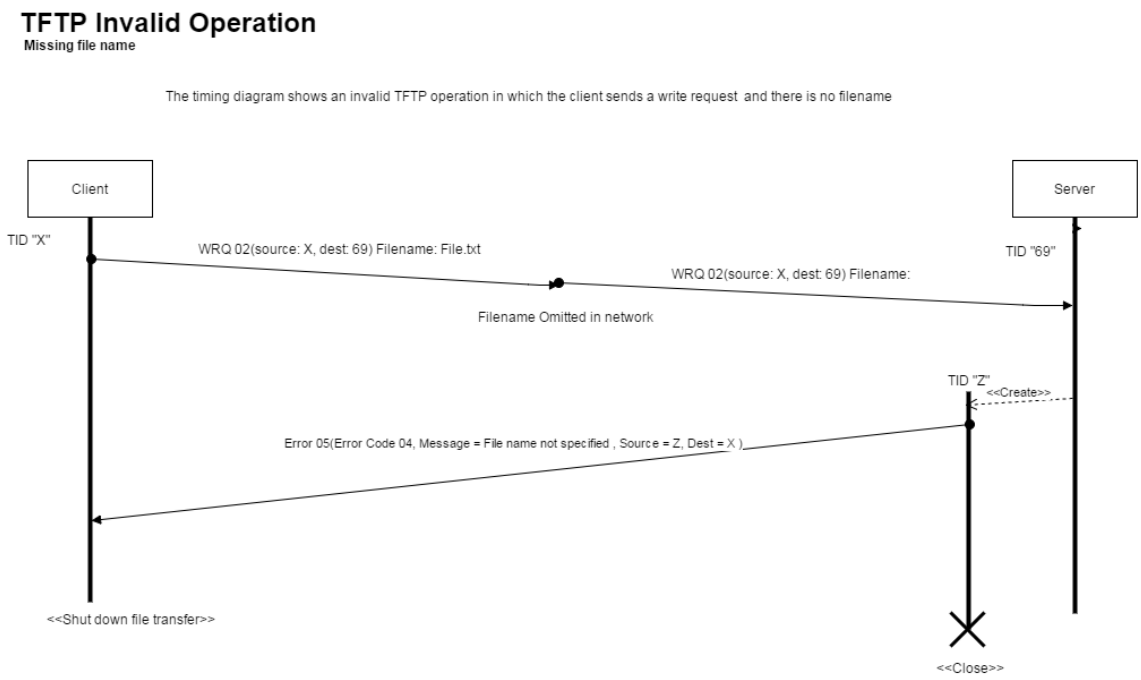


Figure 14: Missing File Name

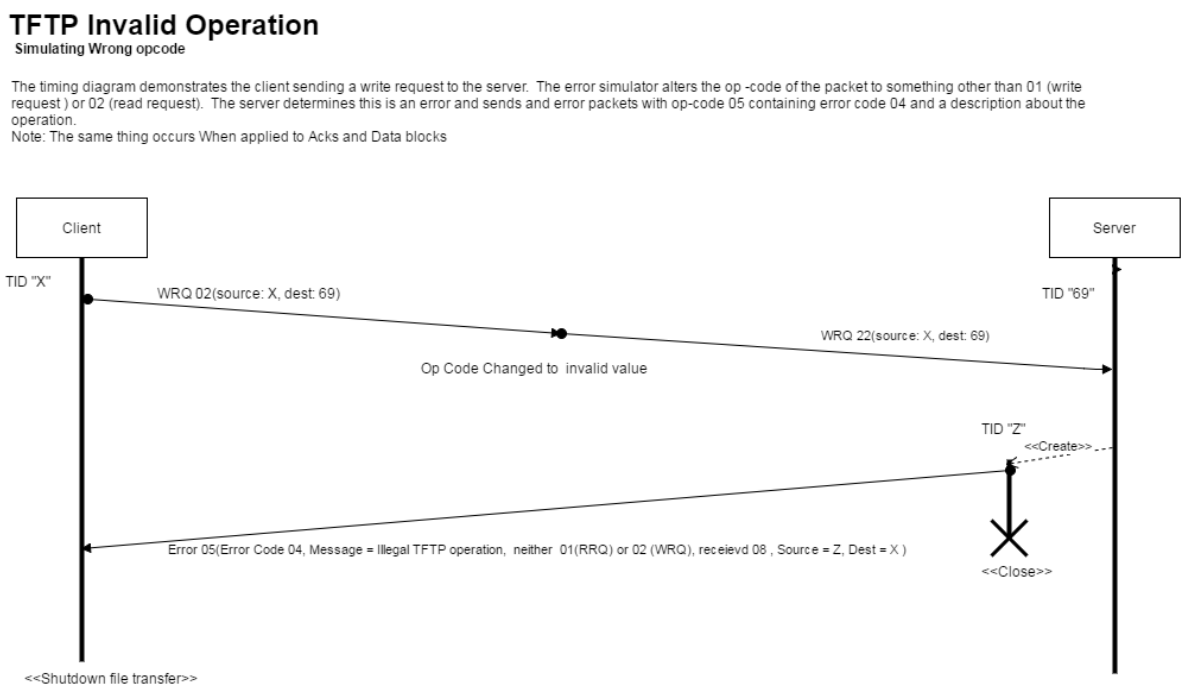


Figure 15: Incorrect OpCode

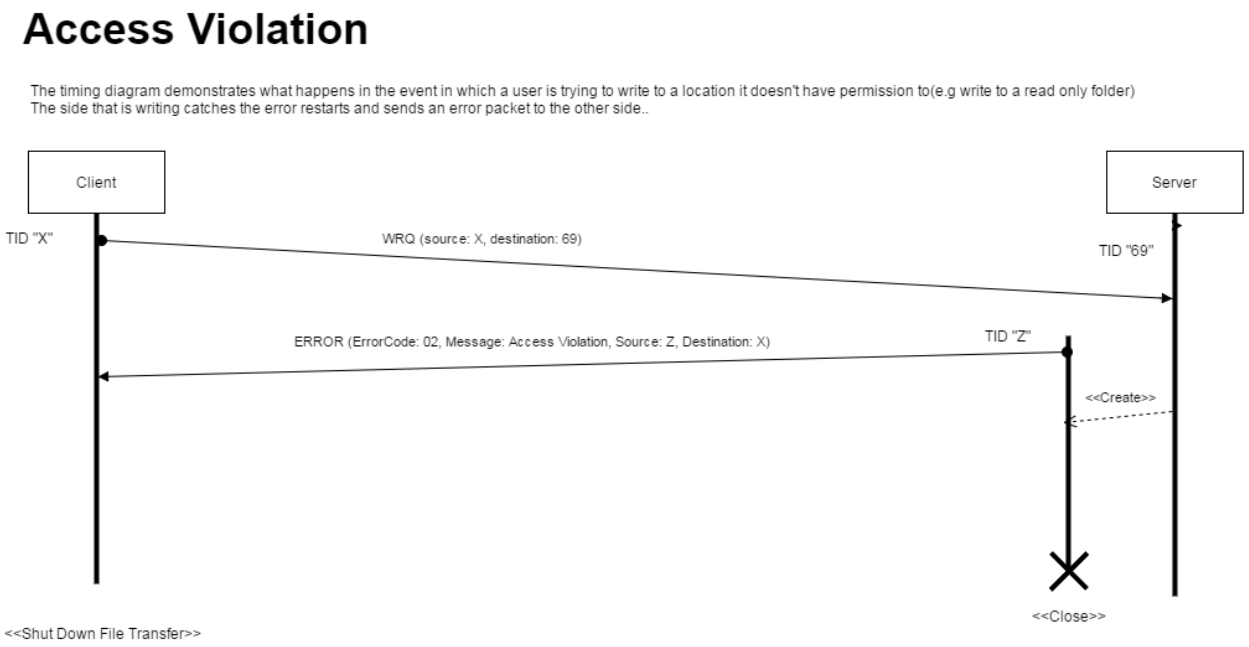


Figure 16: Access Violation

7.2 UML Diagram

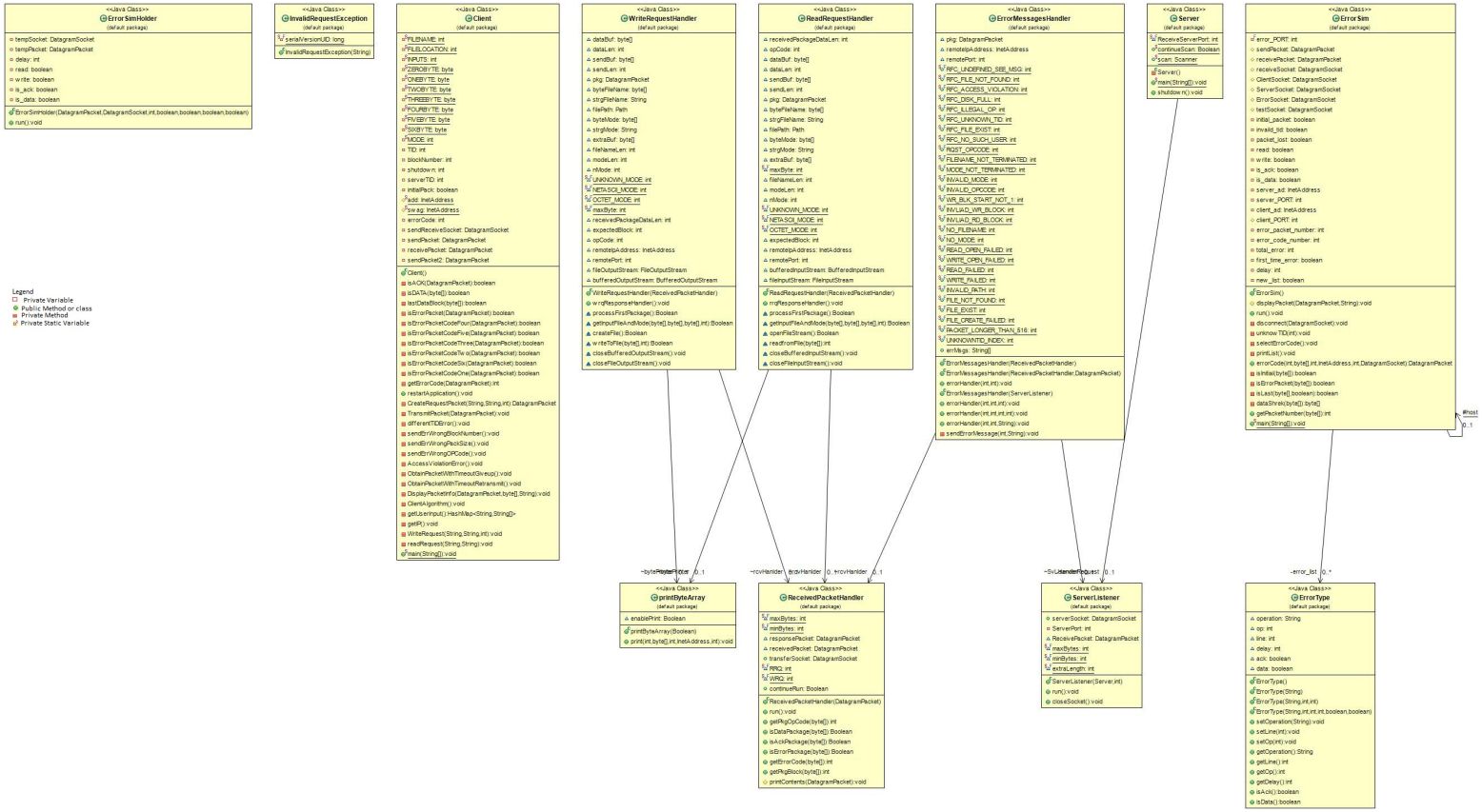


Figure 17: UML Diagram

7.3 UCM Diagrams

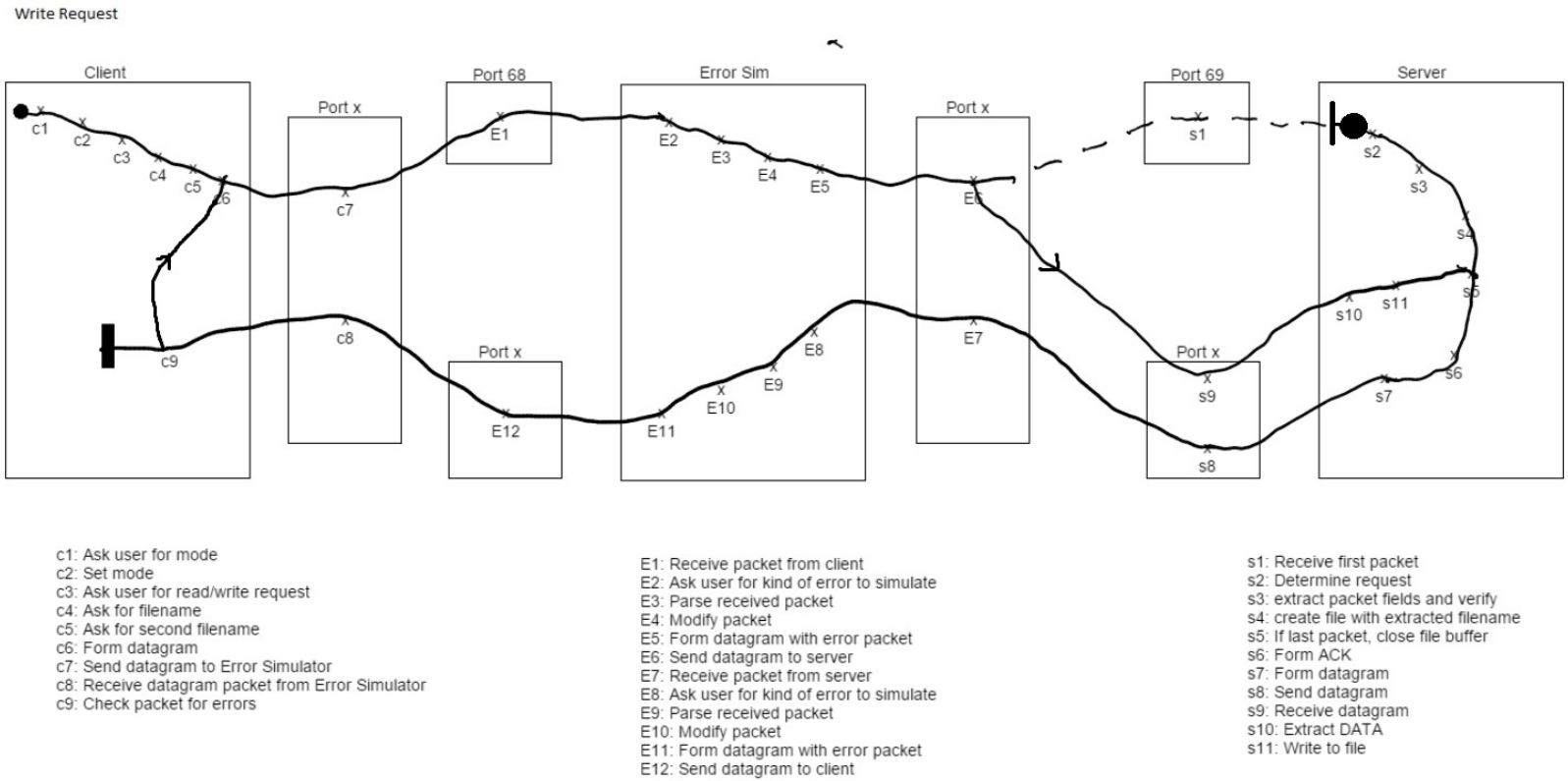


Figure 18: Write Request UCM Diagram

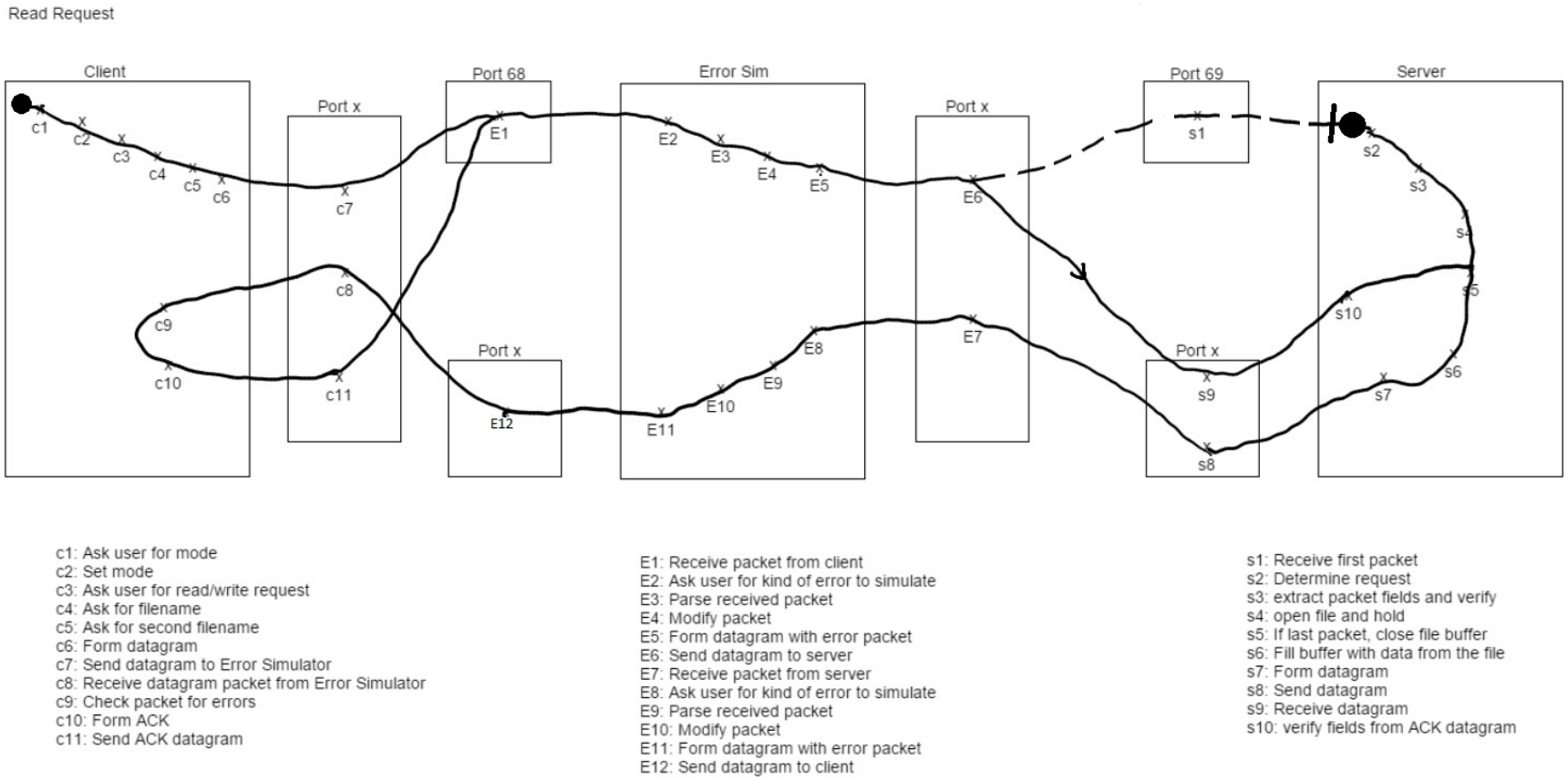


Figure 19: Read Request UCM Diagram