ECE 355

Rdt1.0

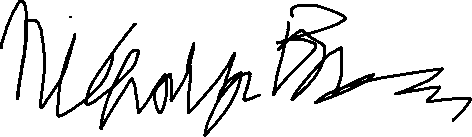
Course Project

Nicholas Barnes

01067990

Nicholas Barnes

Due: 12/13/2020



Honor Code: X\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# Introduction

The course project for ECE355 is to implement the transfer protocol rdt1.0. Using socket programming, the sender must send a message “This is a test” to the receiver. The state diagram of the program/protocol is outlined below.

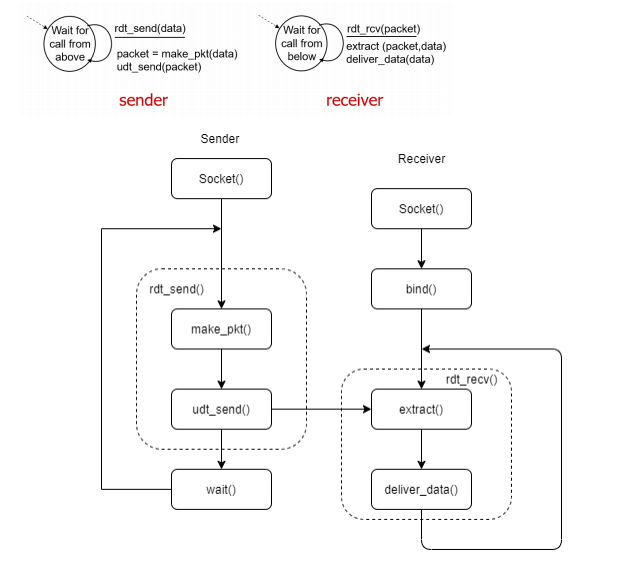


Figure 1. State Diagram for rdt1.0

# Implementation

Programming for this project was done in Python 3.8.6 on a Windows 10 machine. Two files were created, sender.py and receiver.py, which are run in tandem to communicate with each other on the same device.

* 1. Sender

The sender acts like a client in socket programming.

* + 1. Rdt\_send()

This function is located inside a while loop that breaks if the packet has finished sending. It calls make\_pkt() and udt\_send().

* + 1. Make\_pkt()

This function creates a packet to be sent from the phrase “This is a test”. It iterates through the phrase with a packetctr.

* + 1. Udt\_send()

This function sends the packet to receiver.py, the server, at local IP 127.0.0.1 and at port 20001. If the phrase has finished sending, it sends an end code ‘0’.

* + 1. Wait()

This function calls a sleep() function to stall the sending by 1 second. It’s called after rtd\_send() in the while loop.

* + 1. Sooket()

This function applies properties to the socket to make it suitable for UDP transmission. “Sooket” is used instead of socket to prevent confusion.

* 1. Receiver

The receiver acts like a server in socket programming.

* + 1. Rdt\_recv()

This function is located inside a while loop that breaks if the endcode ‘0’ has been received. It calls extract() and deliver\_data().

* + 1. Extract()

Extract receives one character from the socket and adds it to the received phrase. If it receives the endcode ‘0’, an endflag is set to ‘1’ to break the while loop.

* + 1. Deliver\_data()

This function prints the phrase to the screen. Since the receiver gets one character at a time, this function should output: “T”, “Th”, “Thi”, “This”, etc.

* + 1. Softbind()

This function binds the server socket to the local ip 127.0.0.1 and port 20001. Once connected, the function prints connected.

* + 1. Sooket()

This function applies properties to the socket to make it suitable for UDP transmission. “Sooket” is used instead of socket to prevent confusion.

# Compilation

Ensure Python 3.8.6 is installed on a Windows 10 machine. Open two command prompts and navigate both to the ~/355 Course Project directory where sender.py and receiver.py are located. On one prompt, run the command “python receiver.py” to start listening for the sender. On the other prompt, run the command “python sender.py” to connect to the receiver.

# Execution

* 1. Sender.py

Sender.py will output a message “wait one second” each time a message is sent, and will end the program once the entire message is sent.

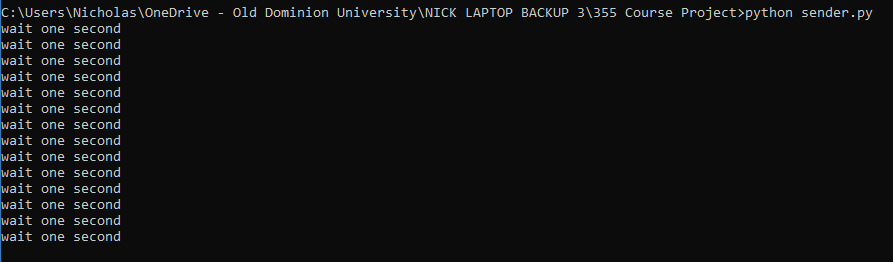


Figure 2. sender.py cmd output

* 1. Receiver.py

Receiver.py will output a ready message once its connected to the socket and will wait for a packet from the sender. Once it has received a packet, it will output the message’s current progress. Once it has received the end code, it will output “END” and end the program.

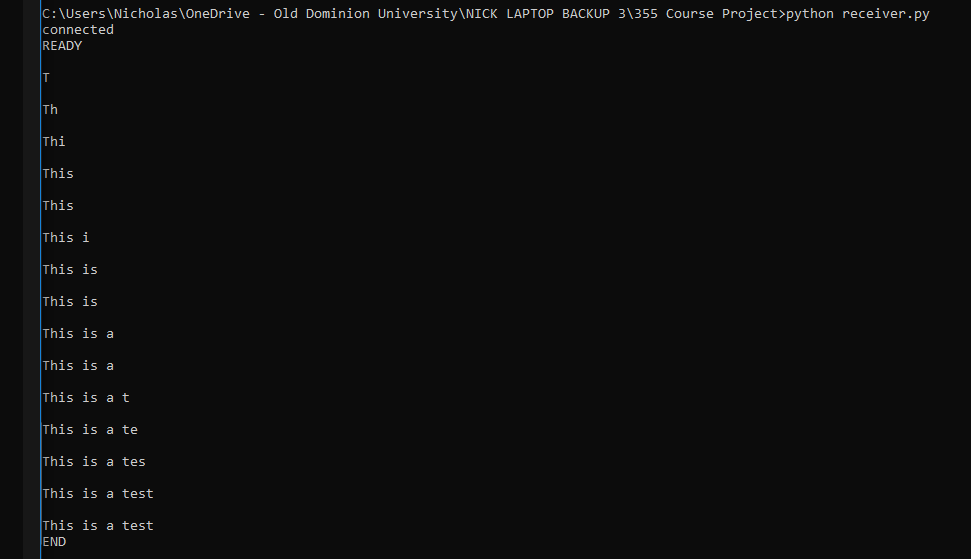


Figure 3. receiver.py cmd output

# Developer’s Notes

This is after about 3 weeks of learning python for developing UDP/TCP clients and servers in ECE484W Computer Engineering Design I. As such there were a few things I could do better that I am not yet familiar with.

* 1. Passing by reference

From what I read about python, all variables are passed by reference. This made it awkward for me to use methods in python, especially in the case of sooket() in both programs. My solution was to use global variables, but really, this program looked a lot neater and probably more effective when nothing was embedded in methods.

* 1. No clear screen function

I could not find a clear screen function! With how print() works it prints a new line every single time it is called. Although it looks better on paper, in execution, it would be nice to clear the screen every time a new character is received and print the message.