Exp No: 7

Date: 11/9/24

Aim: Write a program to implement flow control at data link layer using sliding window Protocol. Simulate the flow of frames from one node to another.

Program should achieve at least below given requirements. You can make it a bidirectional program where in receiver is sending its data frames with acknowledgement (Piggybacking)

Create a sender program with following features:-

1. Input window size from the user.
2. Input a Text message from the user.
3. Consider 1 character per frame.
4. Create a frame with following fields [Frame no. DATA]
5. Send the frames. (Print the output on screen and save it in a file called Sender_Buffer].
6. Wait for the acknowledge from the Receiver. (Induce delay in the program].
7. Reader a file called Receiver_Buffer.
8. Check ACK field for the Acknowledgement number.
9. If the acknowledgement number is as expected, send new set of frames accordingly, [over write the Sender_Buffer with new frames] Else if NACK is received, resend the frames accordingly. [Overwrite the Sender_Buffer with old frame].

Create a receiver file with following features

1. Reader a file called Sender_Buffer.
2. Check the frame no.
3. If the frame no. are as expected, write the appropriate ACK no. in the Receiver_Buffer file. Else write NACK no. in the receiver_Buffer file.

Note: Induce error and verify the behaviour of the program. Manually change the frame no. and ACK no in the files].

Student Observation

i) Sender

```
import time
import os

def sender(window_size, message):
    sender_buffer = "Sender_Buffer.txt"
    receiver_buffer = "Receiver_Buffer.txt"
    frame_no = 0

    frames = [[i, message[i]] for i in
              range(len(message))]

    while frame_no < len(frames):
        for i in range(window_size):
            if frame_no + i < len(frames):
                print(f"Sending frame: {frames[frame_no
                    +i]}")
```

```
with open(sender_buffer, 'a') as f:
f.write(f"{frames[frame_no + i][0]}
        {frames[frame_no + i][1]}\n")
time.sleep(1)  #simulate delay.

while True:
    if os.path.exists(receiver_buffer):
    with open(receiver_buffer, 'r') as f:
        ack_no = int(f.read().strip())
    os.remove(receiver_buffer)
        break

if ack_no >= frame_no:
    print(f"Ack received for frame:
                {ack_no}")
        frame_no = ack_no+1
else:
    print(f"NACK received for frame: {frame_no}
                resending...")

if __name__ == "__main__":
    window_size = int(input("Enter window size:"))
    message = input("Enter message:")
    sender(window_size, message)
```

ii) Receiver

```
import time
import os

def receiver():
    sender_buffer = "Sender_Buffer.txt"
    receiver_buffer = "Receiver_Buffer.txt"
    expected_frame_no = 0
```

```
while True:
    if os.path.exists(sender_buffer):
        with open(sender_buffer, 'r') as f:
            lines = f.readlines()
        os.remove(sender_buffer)

    for line in lines:
        frame = line.strip().split()
        frame_no = int(frame[0])
        data = frame[1]

    if frame_no == expected_frame_no:
        print(f"Received frame: {frame_no}, data:
                {data}")
        with open(receiver_buffer, 'w') as f:
            f.write(str(frame_no))
        expected_frame_no += 1
    else:
        print(f"Unexpected frame: {frame_no},
                expected: {expected_frame_no}")
        with open(receiver_buffer, 'w') as f:
            f.write(str(expected_frame_no - 1))

if __name__ == "__main__":
    receiver()
```

o/p:-

```
Enter window size: 5
Enter message: hello
Sending frame: [0, 'h']
Sending frame: [1, 'e']
Sending frame: [2, 'l']
Sending frame: [3, 'l']
Sending frame [4, 'o']
```

NACK received for frame : 0, resending....
Sending frame: [0, 'h']
Sending frame: [1, 'e']
Sending frame: [2, 'l']
Sending frame: [3, 'l']
Sending frame: [4, 'o']
NACK received for frame: 0, resending....
Sending frame : [0, 'h']
Sending frame : [1, 'e']
Sending frame : [2, 'l']
Sending frame : [3, 'l']
Sending frame : [4, 'o']

receiver.py

Unexpected frame: 2 , expected: 0
Unexpected frame: 2 , expected: 0
Unexpected frame: 3 , expected: 0
Received frame: 0 , data: h
Received frame: 1 , data: e
Received frame: 2 , data: l
Received frame: 3 , data: l
Received frame: 4 , data: o

Result:

Thus the sliding window for Sender and Receiver file is executed successfully and output is verified.

18/9/24