

18/10/2024

## Practical - 7

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 wherein receiver is sending its data frames with acknowledgement.

Create a sender program with following features:

1. Input Window Size from 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.
6. Wait for acknowledgement from receiver
7. Reader a file called receiver - buffer
8. Check Ack field to Acknowledgement number.
9. If acknowledgement number is as expected, send next set of frames accordingly.



Create a receiver file with following features

1. Reads 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.

Student Observation:

```
import time
```

```
import random
```

```
def send-frames (Window-size, message):
```

```
    frame-no = 0
```

```
    frame = []
```

```
    for char
```

```
        sent-frames += 1
```

```
    sent for char in message:
```

```
        frames.append (frame-no, char)
```

```
        frame-no += 1
```

```
    sent-frames = 0
```

```
    while sent-frames < len (frames):
```

```
        Sender-buffer = Open ('sender-Buffer.
```

```
                                Ext', 'w')
        for i in range (Window-size):
```

```
            if sent-frames + i < len (frames):
```

```
                Sender-buffer.write (f"frame {frames[sent-
```

```
frames + i ]LoJ}: {frames[sent-frames + i]
```

```
[i]}\n"]
```



```
Print ("for sending frame {frames[Sent-frames + i][0]}")
i += 1
Sender-buffer.close()
```

```
time.sleep(2)
```

```
check_ack (Sent-frames, frames, Window-Size)
```

```
Sent-frames += Window-Size
```

```
def check_ack (Sent-frames, frames, Window-Size):
    try:
```

```
    receiver-buffer-open ("Receiver-Buffer.txt", "r")
    ack_lines = receiver-buffer.readlines()
```

```
    receiver-buffer.close()
```

```
    acks = [int(line.strip().split()[1]) for line in
```

```
    for i in range(Window-Size): ack_lines]
```

```
    if Sent-frames + i < len(frames):
```

```
        expected_ack = frames[Sent-frames + i][0]
```

```
        if expected_ack in acks:
```

```
            Print ("for ACK {expected_ack} received.")
```

```
        else:
```

```
            Print ("for NACK {expected_ack} received.
                    Resending...")
```

```
            resend-frame (Sent-frames + i, frames)
```

```
            break
```

```
except FileNotFoundError:
```

```
    Print ("Receiver-Buffer.txt not found,
            waiting for acknowledgement...")
```



```
def resend-frames(index, frames):
```

```
Sender-Buffer = Open('Sender-Buffer.txt', 'w')
```

```
Sender-Buffer.Write(f"Frame {frames [index]}  
{frames [index] [1] } \n")
```

```
Sender-Buffer.close()
```

```
def read-frames():
```

```
try:
```

```
Sender-Buffer = Open('Sender-Buffer.txt', 'r')
```

```
frames = Sender-Buffer.readlines()
```

```
Sender-Buffer.close()
```

```
Receiver-Buffer.close()
```

```
Receiver-Buffer = Open('Receiver-Buffer.txt', 'w')
```

```
for frame in frames:
```

```
frame-no = int (frame.split(' ').strip(':'))
```

```
if random.choice([True, False]):
```

```
Print(f"Frame {frame-no} received successfully")
```

```
Receiver-Buffer.Write(f"ACK {frame-no} \n")
```

```
else:
```

```
Print(f"Error detected in Frame {frame-no}")
```

```
Receiver-Buffer.Write(f"Sending NACK")
```

```
time.sleep(1)
```

```
Receiver-Buffer.close()
```

```
except FileNotFoundError:
```

```
Print("Sender-Buffer.txt for sender or
```

```
if mode == 's' : 'r' for receiver : ?-locally
```

```
Window-Size = int(input("Enter Window
```

```
message = input("Enter Message")
```



Send-frames (window-size, message)

elif mode == (r):

While True:

receive-frames()

time.sleep(5)

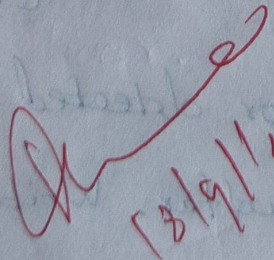
else:

Print("Invalid mode selected. Please choose

's' for sender or 'r' for Receiver.")

Result:

Thus an program to implement  
flow control of data link layer using  
SLIDING WINDOW Protocol has been  
successfully executed.

  
13/9/24