

06/09/24

## EXPERIMENT NO: 07

SLIDING WINDOW PROTOCOL (FLOW CONTROL AT  
DATA LINK LAYER)

## Aim:

To write a program to implement flow control at datalink layer using sliding window protocol. Simulate flow of frame from one node to other.

## Code:

sender1.py

```
import time
```

```
import os
```

```
def create_frames(text_message):
```

```
    frames = [i, char) for i, char in enumerate(text_message)]
```

```
    frames.append((len(text_message), 'END'))
```

```
    return frames
```

```
def write_to_file(filename, data):
```

```
    with open(filename, 'w') as file:
```

```
        for frame in data:
```

```
            file.write(f"{frame[0]} {frame[1]} \n")
```

```
def read_file(filename):
```

```
    if not os.path.exists(filename):
```

```
        return []
```

```
    with open(filename, 'r') as file:
```

```
        return [line.strip().split(' ') for line in file.readlines()]
```

```
def send_frames(frames, window_size):
```

```
    i = 0
```

```
    while i < len(frames):
```

```
        window = frames[i : i + window_size]
```

```
        print(f"sending frames: {window}")
```

```
        write_to_file('sender_Buffer_text', window)
```

```
        time.sleep(3)
```

```
    receiver_buffer = read_from_file('Receiver_Buffer_text')
```



```
if not receiver.Buffer:
```

```
    print ("No acknowledgement received yet")
```

```
    continue
```

```
ack-frame = receiver.buffer[0]
```

```
ack-number, ack-type = int(ack-frame[0]) ack-frame[0]
```

```
if ack-type == 'ACK':
```

```
    print(f"ACK received for frame {ack-number},  
          sender next set of frame.")
```

```
    i += window-size
```

```
else ack-type == 'NACK':
```

```
    print(f"Nack {ack-number} resending {ack-number}")
```

```
    i = ack-number
```

```
def main_sender():
```

```
    window-size = int(input("Enter window size"))
```

```
    text-message = input("Enter text")
```

```
    frames = create-frames(text-message)
```

```
    send_frames(frames, window-size)
```

```
if __name__ == '__main__':
```

```
    main_sender()
```

receiver.py

```
import random
```

```
import time
```

```
import os
```

```
def write_to_file(filename, data):
```

```
    with open(filename, 'w') as file:
```

```
        file.write(data)
```

```
def read_from_file(filename):
```

```
    if not os.path.exists(filename):
```

```
        return []
```

```
    with open(filename, 'r') as file:
```

```
        return [line.strip().split(',') for line in file.readlines()]
```



```
def process_frames(frames):
```

```
    acks = []
```

```
    frame_seen = set()
```

```
    for frame in frames:
```

```
        frame_number = int(frame[0])
```

```
        data = frame[1]
```

```
        if frame_number in frame_seen:
```

```
            continue
```

```
        print(f"Received frame {frame_number}: {data}")
```

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

```
            print(f"Sending Ack for {frame_number}")
```

```
            acks.append(f"{frame_number}, Ack\n")
```

```
            frame_seen.add(frame_number)
```

```
        else:
```

```
            print(f"Sending NACK {frame_number}")
```

```
            ack.append(f"{frame_number}, NACK\n")
```

```
            break
```

```
    return '\n'.join(acks)
```

```
def main_receiver():
```

```
    while True:
```

```
        time.sleep(3)
```

```
        frames = read_from_file('Sender-Buffer-text')
```

```
        if not frames:
```

```
            print("No frames to process, waiting")
```

```
            continue
```

```
        acks = process_frames(frames)
```

```
        write_to_file('receiver-buffer-text', acks)
```

```
        if (acks[frame[0]] == 'END' for frame in frames):
```

```
            print("End of transmission received")
```

```
            break
```

```
if __name__ == "__main__":
```

```
    main_receiver()
```



### Output:

Enter window size: 2

Enter text message: bell

Sending frame: [(0, 'b'), (1, 'e')]

Ack received for frame, sending next frame

Sending frame: [(2, 'l'), (3, 'l')]

Ack received for frame, sending next frame

Sending frame: [(4, 'END')]

Ack received for frame 12, sending next frame

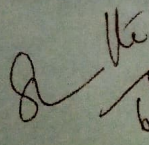
Received frame 4: END

Sending NACK for frame 4

End of transmission received.

### Result:

Thus flow control using sliding window has been successfully implemented and output is verified.

  
6/9/24