Batch: A1 Roll No.: 16010120015

Experiment / assignment / tutorial No.__7_

Grade: AA / AB / BB / BC / CC / CD /DD

Signature of the Staff In-charge with date

Experiment No.:7

TITLE: Flow control Mechanism: Go-Back- N ARQ Sliding Window Protocol using Socket programming

---

AIM: Implementation of Flow Control Mechanism: Stop and Wait ARQ and Go-Back- N Sliding Window Protocol ARQ using sockets.

Expected Outcome of Experiment:

CO: Demonstrate Data Link Layer, MAC layer technologies & protocols and implement the functionalities like error control, flow control

---

Books/ Journals/ Websites referred:

1. A. S. Tanenbaum, "Computer Networks", Pearson Education, Fourth Edition
2. B. A. Forouzan, "Data Communications and Networking", TMH, Fourth Edition

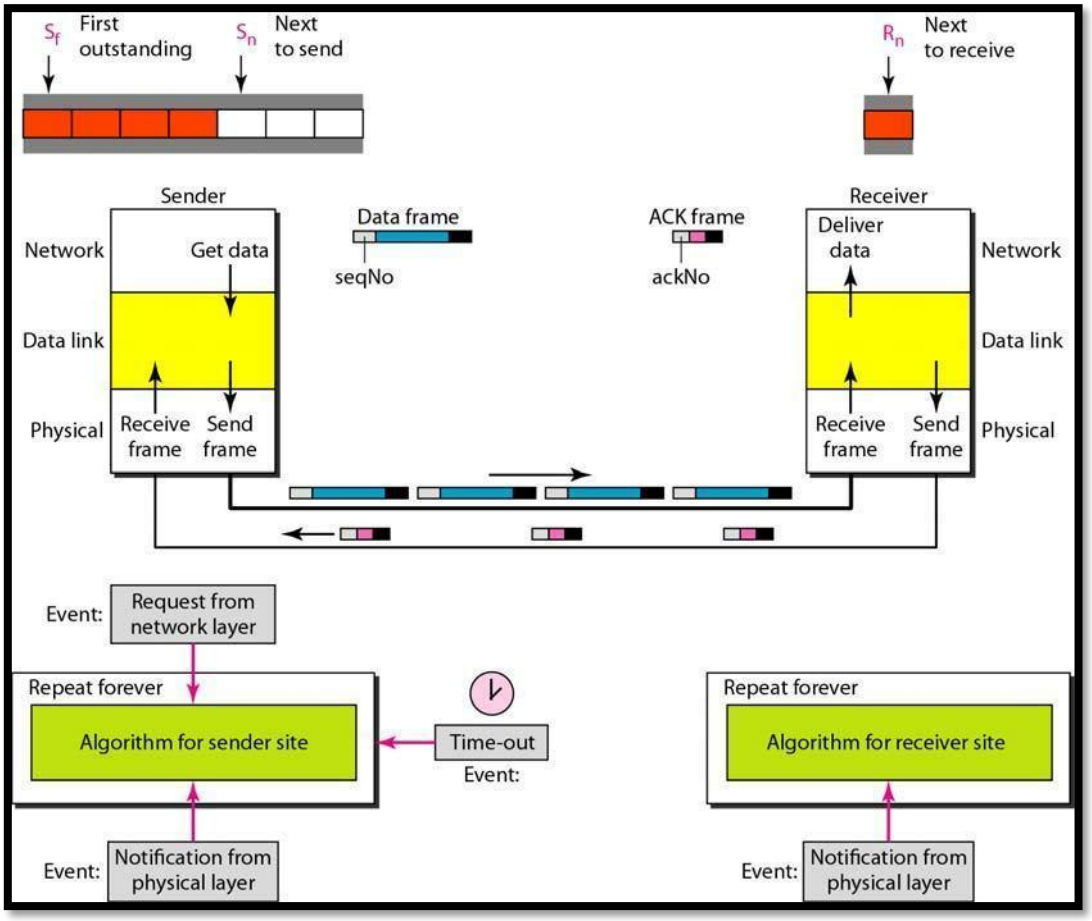
---

Pre-Lab/ Prior Concepts:

Java Socket Programming, Flow Control, Go-Back-Stop and Wait

New Concepts to be learned: Window Flow Control

Design of Go-Back-N ARQ



1. Take data from user about how many bit windows is case of go back n and selective repeat.

2. Generate frames randomly and show the transmission

3. Generate the random number for the frame to be lost.

4. For Go – Back – N transmit all the frames after that number till max number

5. For Selective repeat transmit the selected frame which is not received by the receiver.

IMPLEMENTATION: (printout of code)

- **Server (Receiver)**

```
import socket,random
from time import sleep

s = socket.socket()
host = socket.gethostname()
port = 8080
s.bind((host, port))
s.listen(5)
print("Server Running...")

c, addr = s.accept()
receivedMsg = []
index = 0
count = 0
l = 0
n = int(c.recv(1024).decode('utf-8'))
msgLength = int(c.recv(1024).decode('utf-8'))
print(f">>The size of message is {msgLength} and sliding window size is
{n}\n")
flag = True

while True:
    try:
        if index < msgLength:
            msg = c.recv(1024).decode('utf-8')
            print(f"Received Message: {msg}")
            receivedMsg.append(msg)
        count += 1
        if count >= n:
            randomAckLoss = random.randint(0,3)
            if randomAckLoss == 0 and flag:
                flag = False
                count = 0
                index -= n
                print(f"\n>>Simulating either loss of message or
acknowledgement<<\n")
                print(f"Discarding elements {receivedMsg[-n:]}\n")
                del receivedMsg[-n:]
                c.send(bytes("AckLost",'utf-8'))
            else:
                print(f"\tSending Ack for {receivedMsg[l]}...")
                c.send(bytes("Ack: '"+receivedMsg[l]+"' received", 'utf-8'))
                l += 1
            sleep(1)
        index += 1
```

```
    except:
        print(f"\n>>Final Received Message: {receivedMsg}")
        print("Connection Closed!")
        c.close()
        break
```

- **Client (Sender)**

```
import socket
from time import sleep

s = socket.socket()
host = socket.gethostname()
port = 8080
s.connect((host, port))
print("Connected to",str(host)+":"+str(port))

msg = input("Enter list of messages to be sent separated by ',': ").split(',')
n = int(input(f"Enter size of sliding window (size<{len(msg)}): "))
print(msg," ",n)
print()
s.send(bytes(str(n),'utf-8'))
s.send(bytes(str(len(msg)),'utf-8'))
index = 0
count = 0
ackCount = 0
windowBuffer = [msg[i] for i in range(n)]

while True:
    if index < len(msg):
        print(f"\tSending {msg[index]}...")
        s.send(bytes(msg[index], 'utf-8'))
        sleep(1)
        count += 1
    if count >= n:
        ack = s.recv(1024).decode('utf-8')
        if ack == 'AckLost':
            print(f"\n\tDiscarding messages {windowBuffer}...\n\tResending
frames again...\n\n")
            index -= n
            count = 0
            ackCount -= n
            if ackCount < 0:
                ackCount = 0
        else:
            print(f"{ack}", end="\t")
```

```
            ackCount += 1
            if index+1 < len(msg):
                windowBuffer.pop(0)
                windowBuffer.append(msg[index+1])
    print(f"Sliding Window: {windowBuffer}")
    index += 1
    if ackCount == len(msg):
        print("\nMessage Sent Successfully!")
        break

s.close()
```

Output:

```
$ python gbnS.py
Server Running...
>>The size of message is 10 and sliding window size is 4

Received Message: 1
Received Message: 2
Received Message: 3
Received Message: 4
        Sending Ack for 1...
Received Message: 5
        Sending Ack for 2...
Received Message: 6
        Sending Ack for 3...
Received Message: 7
        Sending Ack for 4...
Received Message: 8
        Sending Ack for 5...
Received Message: 9
        Sending Ack for 6...
Received Message: 10

>>Simulating either loss of message or acknowledgement<<

Discarding elements ['7', '8', '9', '10']

Received Message: 7
Received Message: 8
Received Message: 9
Received Message: 10
        Sending Ack for 7...
        Sending Ack for 8...
        Sending Ack for 9...
        Sending Ack for 10...

>>Final Received Message: ['1', '2', '3', '4', '5', '6', '7', '8', '9', '10']
Connection Closed!
(python38)
```

```
$ python gbnC.py
Connected to   YASH : 8080
Enter list of messages to be sent separated by ',': 1,2,3,4,5,6,7,8,9,10
Enter size of sliding window (size<10): 4
['1', '2', '3', '4', '5', '6', '7', '8', '9', '10']   4

        Sending 1...
Sliding Window: ['1', '2', '3', '4']
        Sending 2...
Sliding Window: ['1', '2', '3', '4']
        Sending 3...
Sliding Window: ['1', '2', '3', '4']
        Sending 4...
Ack: '1' received       Sliding Window: ['2', '3', '4', '5']
        Sending 5...
Ack: '2' received       Sliding Window: ['3', '4', '5', '6']
        Sending 6...
Ack: '3' received       Sliding Window: ['4', '5', '6', '7']
        Sending 7...
Ack: '4' received       Sliding Window: ['5', '6', '7', '8']
        Sending 8...
Ack: '5' received       Sliding Window: ['6', '7', '8', '9']
        Sending 9...
Ack: '6' received       Sliding Window: ['7', '8', '9', '10']
        Sending 10...

        Discarding messages ['7', '8', '9', '10']...
        Resending frames again...


Sliding Window: ['7', '8', '9', '10']
        Sending 7...
Sliding Window: ['7', '8', '9', '10']
        Sending 8...
Sliding Window: ['7', '8', '9', '10']
        Sending 9...
Sliding Window: ['7', '8', '9', '10']
        Sending 10...
Ack: '7' received       Sliding Window: ['7', '8', '9', '10']
Ack: '8' received       Sliding Window: ['7', '8', '9', '10']
Ack: '9' received       Sliding Window: ['7', '8', '9', '10']
Ack: '10' received      Sliding Window: ['7', '8', '9', '10']
        Sliding Window: ['7', '8', '9', '10']
        Sliding Window: ['7', '8', '9', '10']
        Sliding Window: ['7', '8', '9', '10']
        Sliding Window: ['7', '8', '9', '10']

Message Sent Successfully!
```

CONCLUSION:

Thus, in this experiment the concept of flow control protocol was understood using which different scenarios of go back n protocol was implemented successfully in the above code.

Post Lab Questions