**Experiment 7**

**Name: Khan Mohd Aaquib Sameer**

**Roll No. 22CO21**

**Batch : 2**

**Aim : Stop-and-Wait ARQ: A Simple Error Control Protocol**

Stop-and-Wait Automatic Repeat Request (ARQ) is a basic error control protocol used in data communication.It ensures reliable data transmission by sending a single data frame at a time and waiting for an acknowledgment (ACK) before sending the next frame.

**How it Works:**

1. **Transmission:**The sender transmits a data frame to the receiver.
2. **Acknowledgment:** The receiver processes the received data frame and sends an ACK back to the sender if the frame is received correctly. If there's an error, the receiver doesn't send an ACK.
3. **Timeout:** If the sender doesn't receive an ACK within a specified timeout period, it assumes the frame was lost or corrupted. The sender then retransmits the same frame.
4. **Repeat:** This process continues until the ACK is received, indicating successful transmission.

**Java Implementation:**

1. **Sender :**

import java.io.\*;

import java.net.\*;

public class Sender {

private Socket sender;

private ObjectOutputStream out;

private ObjectInputStream in;

private String packet, ack, str, msg;

private int n, i = 0, sequence = 0;

public Sender() {

// Empty constructor

}

public void run() {

try ( // Use try-with-resources for automatic closing

BufferedReader br = new BufferedReader(new InputStreamReader(System.in));

Socket sender = new Socket("localhost", 2004);

ObjectOutputStream out = new ObjectOutputStream(sender.getOutputStream()); {

out.flush();

in = new ObjectInputStream(sender.getInputStream());

str = (String) in.readObject();

System.out.println("Receiver > " + str);

System.out.println("Enter the data to send....");

packet = br.readLine();

n = packet.length();

do {

try {

if (i < n) {

// Build message with sequence number and character

msg = String.valueOf(sequence) + packet.substring(i, i + 1);

} else if (i == n) {

// Send "end" message to signal completion

msg = "end";

out.writeObject(msg);

break;

}

out.writeObject(msg);

sequence = (sequence == 0) ? 1 : 0;

out.flush();

System.out.println("Data sent > " + msg);

ack = (String) in.readObject();

System.out.println("Waiting for ack.....\n\n");

if (ack.equals(String.valueOf(sequence))) {

i++;

System.out.println("Receiver > packet received\n\n");

} else {

System.out.println("Timeout, resending data....\n\n");

sequence = (sequence == 0) ? 1 : 0;

}

} catch (Exception e) {

// Handle exceptions gracefully (optional)

}

} while (i < n + 1);

System.out.println("All data sent. Exiting.");

} catch (Exception e) {

// Handle exceptions gracefully (optional)

} finally {

try {

if (in != null) in.close();

if (out != null) out.close();

if (sender != null) sender.close();

} catch (Exception e) {

// Handle exceptions gracefully (optional)

}

}

}

public static void main(String[] args) {

Sender s = new Sender();

s.run();

}

}

1. **Receiver :**

New! Keyboard shortcuts …

Drive keyboard shortcuts have been updated to give you first-letters navigation

import java.io.\*;

import java.net.\*;

public class Reciever{

ServerSocket reciever;

Socket connection=null;

ObjectOutputStream out;

ObjectInputStream in;

String packet,ack,data="";

int i=0,sequence=0;

Reciever(){}

public void run(){

try{

BufferedReader br=new BufferedReader(new InputStreamReader(System.in));

reciever = new ServerSocket(2004,10);

System.out.println("waiting for connection...");

connection=reciever.accept();

sequence=0;

System.out.println("Connection established :");

out=new ObjectOutputStream(connection.getOutputStream());

out.flush();

in=new ObjectInputStream(connection.getInputStream());

out.writeObject("connected .");

do{

try{

packet=(String)in.readObject();

if(Integer.valueOf(packet.substring(0,1))==sequence){

data+=packet.substring(1);

sequence=(sequence==0)?1:0;

System.out.println("\n\nreceiver>"+packet);

}

else

{

System.out.println("\n\nreceiver>"+packet +" duplicate data");

}

if(i<3){

out.writeObject(String.valueOf(sequence));i++;

}

else{

out.writeObject(String.valueOf((sequence+1)%2));

i=0;

}

}

catch(Exception e){}

}while(!packet.equals("end"));

System.out.println("Data recived="+data);

out.writeObject("connection ended .");

}

catch(Exception e){}

finally{

try{

in.close();

out.close();

reciever.close();

}

catch(Exception e){}

}

}

public static void main(String args[]){

Reciever s=new Reciever();

while(true){

s.run();

}

}

}