Programming Project
A WWW Client Server Application
COE442.03 Term 021

Kamran Arshad
ID # 210261
Email: akamran@kfupm.edu.sa
Date: 14-01-2003

1. Overview of the Application

Basic purpose of this project, is to develop a customized web server and a simple client using stream sockets communication. myserver.java process executes on one machine which we call web server and mybrowser.java runs on a separate machine which acts as a client. Client can interact with the server by using stream TCP based sockets, server is always listening on a port (first four digits of my ID i.e. 2102) and client interact with the server by using same port.

2. Design and Implementation

In this project, I supposed to writ two main programs, myserver.java and mybrowser.java,

myserver.java

myserver.java first run parent process in an infinite loop and look for clients asking for establishment of connections. Since here it is required that more than one client can be connected to the server simultaneously, that's why it runs in an infinite loop. When a client tries to communicate, this myserver makes a thread of it, which extends the main program. So as soon as any client requests for an establishment of connection, run( ) in the thread starts. BufferedReader( ) and PrintWriter( ) are used to read and write from the socket. BufferedReader reads text from a character input stream, buffering characters as to provide for the efficient reading of characters, arrays and lines. Buffer size used here is default size, because default size is large enough for many purposes. Here, in WWW client server program, client can communicate with the WWW server by two different ways:

```
get http://hostname:port/subdir/filename.ext
ims http://hostname:port/subdir/filename.ext UTC-time
```

either via a get command, which is used to get a html (or in general any) document or by using IMS (IF-Modified Since) command. Now if clients sends a GET command, than this command first establish a connection with the server on the port specified (first four digit of ID) and also mention the exact location of file on server machine. IMS also have same syntax except that it also indicates UTC time for comparison. Now at server, after establishing the connection, it tokenize the command by using StringTokenizer( ) and separates host-name, port number, directory name and file name.

Now once server knows the file name and its exact location, it checks whether it has file or not, in case if file is not found, it returns only HTTP header with return code, indicating file is nor found. Similarly, with IMS if file is not modified it returns HTTP header only with return code that file is not modified, but in case if file was modified it sends the whole file. So HTTP header is send in all cases, but if it found the file or it found that file was modified than it sends the whole file along with HTTP header. The end of the file is indicated by "END_" and clients looks for "END_".

mybrowser.java

Browser (client) starts communication with the server, by sending GET and IMS commands, and finally when it receives the required document(s) it terminates the connection by issuing a QUIT command. As soon as we starts mybrowser.java on client machine it displays a menu, and now user can enter any command, GET or IMS. Mybrowser.java also tokenize the input string reads from the user and than write it on socket. These commands first establishes connection with the

server on specified port and than reads whole file or just header of file. Header includes all information about the file.

3. Performance Measurements

File Size	Average Server Latency (msec)	Average Server Data Rate (Kbits/sec)	Average Client Latency (msec)	Average Client Data Rate (Kbits/sec)
50	3	226	131	13
500	4	1433	118	47
5,000	14	2987	166	265
50,000	122	3737	27078	15
500,000				
5,000,000				

Date of Experiment: 14 – January - 2003
Time: 02:00
Number of repetitions of each experiment (at least 3): 03
Client machine name: tiger.ccse.kfupm.edu.sa
Server machine name: vlsi.ccse.kfupm.edu.sa

QUESTIONS

1) **Is there a difference between the sender average time and the receiver average time? Why?**
Yes there is a clear difference between the server average time and receiver average time. Server average time is very less as compare to the receiver average time. This is very clear because server immediately puts all the data in its output buffer, while receiver average time is so large, because of the delays encountered in transmission.

2) **Which bandwidth is correct, the sender's or receivers?**
Obviously, receivers bandwidth is correct, because receiver receives the data after passing through the network.

3) **Did the file size change the results?**
Yes as the file size increases, data rate increases, but after a certain sie, this increase gets slower.

4. Conclusions

Here in this program, I use the default value of buffers, which can be increased, but normally these buffers are sufficient to handle large files. Programs are working fine, and this project, as I guess is a nice implementation of a very simple web browser with a IMS feature. From learning point of view, its really a good homework, I didn't know programming in java as well as network programming, but now after finishing this project, I feel java very handy and even network programming but with a little extent.

Appendix I (a): myserver.java

```
import java.net.*;
import java.io.*;

public class myserver {

  public static void main(String[] args) throws IOException {
  ServerSocket serverSocket = null;
  boolean listen = true;

  try {
     serverSocket = new ServerSocket(2102);
     } catch (IOException e) {
             System.err.println ("Could not listen on port: 2102");
             System.exit(-1);
                     }
   while (listen)
     new myserverThread (serverSocket.accept()).start();

    serverSocket.close();
  }
}
```

Appendix I (b): myserverThread.java

```
import java.net.*;
import java.io.*;
import java.util.StringTokenizer;
import java.util.Date;

public class myserverThread extends Thread {


    private Socket socket = null;

    public myserverThread(Socket socket) {
          super("daemonThread");
          this.socket = socket;
          System.out.println("Client arrived from "+socket.getInetAddress());
    }

    public void run() {
       String directory = "/a/general/export/homedir/gradics/sadnan/kamran/";
       String filename = "";
       BufferedReader inputfile = null;
```

```
String filename1 = null;

try
{
PrintWriter out = new PrintWriter(socket.getOutputStream(), true);
BufferedReader in = new BufferedReader(new
InputStreamReader(socket.getInputStream()));

String inputLine, outputLine;

loop1: while ((inputLine = in.readLine()) != null) {

if(inputLine.startsWith("get"))
{
Date d=new Date();
Date d1=new Date();
Date d2=new Date();
long time1=0,time2=0,time3=0;
int counter=0;

StringTokenizer st = new StringTokenizer(inputLine,"/");
String token = "";
int i=0;

try{
   while(true){
            i++;
            token = st.nextToken();
            if(i==3)
            filename=token;
                  if (i>3){
                     filename=filename+"/"+token;
                        }
           }
  } catch(Exception ex){
               System.out.println(filename);
                 }

String whole_date="";
Date dnew = new Date();
StringTokenizer st1 = new StringTokenizer(dnew.toString());
whole_date = st1.nextToken()+", "+dnew.toGMTString()+"\r\n";

String H1 ="HTTP/1.1 ";
String H2 ="Date: "+whole_date;
String H3 ="Server: KamranServer/1.0 (Unix)\r\n";
```

```
String H4 = "Last-Modified: ";
String H5 = "Content-Length: ";
String H6 = "Content-Type: ";
String returncode ="";
long size=0;

try{
   File f = new File(directory+"/"+filename);
   size = f.length();
   inputfile = new BufferedReader(new FileReader(f));
   returncode=H1+"200 OK"+"\r\n";
   d.setTime(f.lastModified());
   StringTokenizer sttok = new StringTokenizer(d.toString());
   String LM = ""+sttok.nextToken()+", "+d.toGMTString()+"\r\n";
   String CL = ""+f.length()+"\r\n";
   String CT = "";
   String name_of_file=f.getName();
   StringTokenizer stfilename = new StringTokenizer(name_of_file,".");
   stfilename.nextToken();
   String ext = stfilename.nextToken();

   if(ext.compareToIgnoreCase("htm")==0 ||ext.compareToIgnoreCase("html")==0)
                        CT = "text/html\r\n";
   if(ext.compareToIgnoreCase("JPEG")==0 ||ext.compareToIgnoreCase("JPG")==0)
                CT = "image/jpeg\r\n";
   if(ext.compareTo("gif")==0)
                        CT = "image/gif\r\n";
   if(ext.compareTo("txt")==0)
                        CT = "text/txt\r\n";
                        CT += "\r\n";
   String Header =
returncode+H2+H3+H4+LM+H5+CL+H6+CT+"_HEADER_ENDS";
                out.println(Header);
   String transfer= "";

   while(inputfile.ready())
          {
          if (counter==0)
                {
                 d  = new Date();
                 time1 = d.getTime();
                }
           counter++;
           transfer = inputfile.readLine();
           out.println(transfer);
           }
```

```
   } catch (Exception ex){
                returncode=H1+"404 FileNotFound"+"\r\n";
                String Header = H1+returncode+H2+H3+"\r\n_HEADER_ENDS";
                out.println(Header);
                     }
          out.println("END_");    // end of the file

          d1 = new Date();
          time2 = d1.getTime();


          System.out.println("Size of the file is " + size);
          System.out.println("Latency: " + (time2-time1));
          long rate =0;

          if((time2-time1)>0) {
                rate=(size*8)/(time2-time1);
                 }
          System.out.println("Data Rate: " + rate);

         } // get ends


////////////////////////////////// IMS ////////////////////////////////

         if(inputLine.startsWith("ims")||inputLine.startsWith("IMS"))
          {
          Date d=new Date();
          Date d1=new Date();
          Date d2=new Date();
          long time1=0,time2=0,time3=0;
          int counter=0;
          StringTokenizer storig = new StringTokenizer(inputLine);
          String UTCTIME = storig.nextToken();
          UTCTIME = storig.nextToken();
          UTCTIME = storig.nextToken();

          StringTokenizer st = new StringTokenizer(inputLine,"/");
          String token = "";
          int i=0;

          try {
              while(true){
                        i++;
                        token = st.nextToken();
                        if(i==3)
```

```
                        filename=token;
                if (i>3){
                        filename=filename+"/"+token;
                    }
                }
            }catch(Exception ex) {
                    }

        String whole_date="";
        Date dnew = new Date();
        StringTokenizer st1 = new StringTokenizer(dnew.toString());
        whole_date = st1.nextToken()+", "+dnew.toGMTString()+"\r\n";

        String H1 ="HTTP/1.1 ";
        String H2 ="Date: "+whole_date;
        String H3 ="Server: KamranServer/1.0 (Unix)\r\n";
        String H4 = "Last-Modified: ";
        String H5 = "Content-Length: ";
        String H6 = "Content-Type: ";
        String returncode ="";

        StringTokenizer strFN = new StringTokenizer(filename);
        filename = strFN.nextToken();
        long size=0;

        try {
            File f = new File(directory+"/"+filename);
            inputfile = new BufferedReader(new FileReader(f));

/**************************/

            System.out.println(f.lastModified()/1000);
            if(f.lastModified()/1000 == Long.parseLong(UTCTIME)) // if both times are same
                    {
                     returncode=H1+"303 NotModified"+"\r\n";
                     String Header = returncode+H2+H3+"_HEADER_ENDS";
                     out.println(Header);
                    } else {    // if the file is a modified one.
                        returncode=H1+"200 OK"+"\r\n";
                        d.setTime(f.lastModified());
                        StringTokenizer sttok = new StringTokenizer(d.toString());
                        String LM = ""+sttok.nextToken()+", "+d.toGMTString()+"\r\n";
                        String CL = ""+f.length()+"\r\n";
                        String CT = "";
                        String name_of_file=f.getName();
                        StringTokenizer stfilename = new StringTokenizer(name_of_file,".");
```

```
                        stfilename.nextToken();

                        String ext = stfilename.nextToken();
                        if(ext.compareToIgnoreCase("htm")==0
||ext.compareToIgnoreCase("html")==0)
                                CT = "text/html\r\n";
                        if(ext.compareToIgnoreCase("JPEG")==0
||ext.compareToIgnoreCase("JPG")==0)
                                CT = "image/jpeg\r\n";
                        if(ext.compareTo("gif")==0)
                                CT = "image/gif\r\n";
                        if(ext.compareTo("txt")==0)
                                CT = "text/txt\r\n";
                         CT += "\r\n";
                         String Header =
returncode+H2+H3+H4+LM+H5+CL+H6+CT+"_HEADER_ENDS";
                         out.println(Header);
                         String transfer= "";

                        while(inputfile.ready())
                          {
                           if (counter==0)
                            {
                              d  = new Date();
                                      time1 = d.getTime();
                            }

                            counter++;
                            transfer = inputfile.readLine();
                            size=size+transfer.length();
                            out.println(transfer);
                          }
                       } // else file modified ends
/*********************/
              } catch (Exception ex){
                        returncode=H1+"404 FileNotFound"+"\r\n";
                        String Header = H1+returncode+H2+H3+"\r\n_HEADER_ENDS";
                        out.println(Header);
                        System.out.println(ex);
                          }


        out.println("END_");    // end of the file

        d1 = new Date();
        time2 = d1.getTime();
```

```
            System.out.println("Size of the file is " + size);
            System.out.println("Latency: " + (time2-time1));
            long rate =0;

            if((time2-time1)>0) {
                    rate=(size*8)/(time2-time1);
                            } //*(time2-time1));
             System.out.println("Data Rate: " + rate);

           }//ims ends


           if(inputLine.startsWith("quit"))
             {
                   break loop1;
             }

       }

     out.close();
     in.close();

   }
      catch (IOException e) {
            e.printStackTrace();
          }
   }

}
```

Appendix II: mybrowser.java

```
import java.io.*;
import java.lang.*;
import java.net.*;
import java.util.StringTokenizer;
import java.util.Date;

public class mybrowser {
   public static void main(String[] args) throws IOException {

       Socket kkSocket = null;
       PrintWriter out = null;
       BufferedReader in = null;
```

```
PrintWriter outputfile = null;

BufferedReader stdIn = new BufferedReader(new InputStreamReader(System.in));
String fromServer;
Date d=null,d1=null,d2=null;
long time1=0,time2=0,time3=0;
String fromUser="";

loop1: while (true/*(fromServer = in.readLine()) != null*/) {

 showmenu();


   fromUser = stdIn.readLine();


 if (fromUser.equals("quit"))
       break loop1;

        if ((fromUser.startsWith("ims"))||(fromUser.startsWith("IMS")))
          {

                        String filename="";
                StringTokenizer storig = new StringTokenizer(fromUser);
                        String host_file_name = storig.nextToken();
                        host_file_name = storig.nextToken();
                        String UTCTIME = storig.nextToken();

                        StringTokenizer st = new StringTokenizer(host_file_name,"/");
                        String host;
                        String token = "";
                        st.nextToken();
                        host=st.nextToken();

                        StringTokenizer stport = new StringTokenizer(host,":");
                        host= stport.nextToken();
                        int port= Integer.parseInt(stport.nextToken());


                        host_file_name.trim();
                        StringTokenizer st2 = new StringTokenizer(host_file_name,"/");

                        int i=0;
                        filename="";
                        try{
                              while(true){
```

```
                                        i++;
                                        token = st2.nextToken();
                                         if(i==3)
                                           filename=token;

                                          if (i>3){
                                            filename=filename+"/"+token;
                                         }
                                        }
                                    }catch(Exception ex){
                                            }


                            try {
                        kkSocket = new Socket(host, port);
                            out = new PrintWriter(kkSocket.getOutputStream(), true);
                               in = new BufferedReader(new
InputStreamReader(kkSocket.getInputStream()));
                               } catch (UnknownHostException e) {
                                        System.err.println("Don't know about host:
taranis.");
                                        System.exit(1);
                                } catch (IOException e) {
                                        System.err.println("Couldn't get I/O for the
connection to: "+ host);
                                        System.exit(1);
                                }
                                    System.out.println("Connection Successfully
Established:");



                                    d = new Date();
                                time1 = d.getTime();
                                    out.println(fromUser);
                                    String data;


                                    int counter = 0;        // If counter ==0 then it is a first line
                                    String Header="";


while(!((data=in.readLine()).equalsIgnoreCase("_HEADER_ENDS")))
                                        {
                                            if(counter==0)
```

```
                                        {
                                        d1 = new Date();
                                    time2 = d1.getTime();
                                    }
                                counter+=data.length();

                                Header+=data;
                                System.out.println(data);

                            }

                                if
(!(data=in.readLine()).equalsIgnoreCase("END_"))
                                    {
                                        outputfile = new PrintWriter(new
FileWriter(filename));
                                        outputfile.println(data);
                                    System.out.println(data);


        while(!((data=in.readLine()).equalsIgnoreCase("END_")))
                                        {

                            outputfile.println(data);
                                    System.out.println(data);

                                        }
                                        outputfile.close();
                                    }else{

                                        }
                                        d2 = new Date();      // Second timer
starts
                                        time3 = d2.getTime();

                                    File f = new File(filename);
                                    long size = f.length();
                        long total_size = size+(long)counter; // Total size of file including
header

                        System.out.println("Size of the file including header is "+total_size+"
bytes.");
                        System.out.println("First Byte Latency Time : " + (time2-time1)+"
msec.");
```

```
                                                    System.out.println("File Transfer Latency
Time: " + (time3-time2)+" msec.");
                                                    System.out.println("Latency: " + ((time2-
time1)+(time3-time2))+" msec.");
                            long  rate = (total_size*8);  // *(time3-time1));
                                                    System.out.println("Data Rate: " +
(rate/(time3-time1))+" Kbps.");
                                                    counter=0;
                                        /*      }//else ends*/


        }  // ims ends


   if((fromUser.startsWith("get"))||(fromUser.startsWith("GET"))) {



        String filename="";
        StringTokenizer storig = new StringTokenizer(fromUser);
        String host_file_name = storig.nextToken();
    host_file_name = storig.nextToken();


    StringTokenizer st = new StringTokenizer(host_file_name,"/");
    String host;
    String token = "";
    st.nextToken();
    host=st.nextToken();

    StringTokenizer stport = new StringTokenizer(host,":");
      host= stport.nextToken();
      int port= Integer.parseInt(stport.nextToken());


 host_file_name.trim();
        StringTokenizer st2 = new StringTokenizer(host_file_name,"/");

      int i=0;
      filename="";
      try{
        while(true){
          i++;
          token = st2.nextToken();
        if(i==3)
```

```
   filename=token;

   if (i>3){
    filename=filename+"/"+token;
     }
   }
   }catch(Exception ex){
  }


try {
   kkSocket = new Socket(host, port);
   out = new PrintWriter(kkSocket.getOutputStream(), true);
   in = new BufferedReader(new InputStreamReader(kkSocket.getInputStream()));
} catch (UnknownHostException e) {
   System.err.println("Don't know about host: taranis.");
   System.exit(1);
} catch (IOException e) {
   System.err.println("Couldn't get I/O for the connection to: "+ host);
   System.exit(1);
}
            System.out.println("Connection Successfully Established:");




                  d = new Date();
            time1 = d.getTime();
                  out.println(fromUser);
                  String data;

                  int counter = 0;          // If counter ==0 then it is a first line
                  String Header="";

         while(!((data=in.readLine()).equalsIgnoreCase("_HEADER_ENDS")))
                  {
                        if(counter==0)
                    {
                  d1 = new Date();
                  time2 = d1.getTime();
                  }
            counter+=data.length();
                        Header+=data;
                        System.out.println(data);
            System.out.println(Header.length());
```

```
                }

                if (!(data=in.readLine()).equalsIgnoreCase("END_"))
                {
                                outputfile = new PrintWriter(new FileWriter(filename));
                                outputfile.println(data);
                   System.out.println(data);

                        while(!((data=in.readLine()).equalsIgnoreCase("END_")))
                        {
                                   outputfile.println(data);
                          System.out.println(data);

                        }
                        outputfile.close();
                }else{

                        }
                        d2 = new Date();     // Second timer starts
                        time3 = d2.getTime();

                        File f = new File(filename);
                        long size = f.length();
                        long total_size=size+(long)counter;
                System.out.println("Size of the file including header is "+total_size+" bytes.");
                        System.out.println("First Byte Latency Time : " + (time2-time1)+"
msec.");
                        System.out.println("File Transfer Latency Time: " + (time3-time2)+"
msec.");
                        System.out.println("Latency:     "     +     ((time2-time1)+(time3-time2))+"
msec.");
                long  rate = (total_size*8);  // *(time3-time1));
                        System.out.println("Data Rate: " + (rate/(time3-time1))+" Kbps.");
                        counter=0;
/*        }//else ends*/
         } // get ends

            }

         try{
      out.close();
      in.close();
      stdIn.close();
      kkSocket.close();
         }catch(Exception ex){
```

```
                }
    }

    static public void showmenu()
    {
                System.out.println();
                System.out.println();
            System.out.println();
            System.out.println("*******************************");
                System.out.println("    get http://hostname:port/subdir/filename.ext          ");
                System.out.println("    ims http://hostname:port/subdir/filename.ext UTC-time ");
                System.out.println("    quit                          ");
                System.out.println("*******************************");
                System.out.println();
                System.out.println();
                System.out.println();

        }
}
```