**MINI PROJECT: Network Compiler**

**Group Members:**

1. Sarthak Dadhakar (15)
2. Umesh Ramchandani (57)
3. Labhesh Valechha (78)

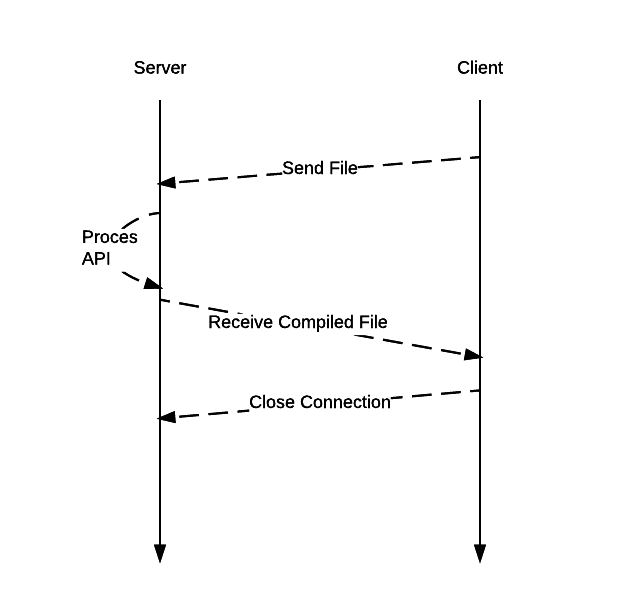
**Aim:**  
Mini project Network Compiler

**Theory**:  
 Network compiler are required because there are many use cases where the classes and the program increase in size and complexity which can lead to heavy load in a single machine which will lead to increase in compile time.  
 Network Complier is basically a system that communicates using Socket and file to byte array transfer. The server receives the file and send the compiled file back to client again.

**Methodology**:

* A connection between the server and client is made using Socket API.
* The client sends the file to server by converting the file to byte array.
* This byte array is sent to the server.
* The server receives the byte array and convert it to a file.
* The server uses Process API to compile the file using gcc or javac command.
* The server converts the compiled file to byte array and sent it client back.
* The client receives it and closes the connection or send another file.

**Diagram**:

****

**Code:**

**Server:**

package osserver;  
import java.io.\*;  
import java.net.\*;  
import java.util.\*;  
public class OsServer {  
 public final static int SOCKET\_PORT = 3000;  
 private static Socket socket;  
 public final static String FILE\_TO\_RECEIVED = "C:\\Users\\Umesh Ramchandani\\Documents\\NetBeansProjects\\OsServer\\";   
   
 public static void main(String[] args) throws IOException {  
 try{  
 ServerSocket serverSocket = new ServerSocket(SOCKET\_PORT);  
 System.out.println("Server Started and listening to the port "+SOCKET\_PORT);  
 while(true)  
 {  
 socket = serverSocket.accept();  
 InputStream is = socket.getInputStream();  
 InputStreamReader isr = new InputStreamReader(is);  
 Scanner sc = new Scanner(isr);  
 String fileName = sc.next();  
 int size = sc.nextInt();  
 System.out.println("Message received from client is "+fileName+size);  
 String returnMessage = 200 + "\n";  
 OutputStream os = socket.getOutputStream();  
 OutputStreamWriter osw = new OutputStreamWriter(os);  
 BufferedWriter bw = new BufferedWriter(osw);  
 bw.write(returnMessage);  
 System.out.println("Message sent to the client is "+returnMessage);  
 bw.flush();  
 while(true){  
 int bytesRead;  
 int current = 0;  
 FileOutputStream fos = null;  
 BufferedOutputStream bos = null;  
 System.out.println("Deadlock");  
 System.out.println(socket.isClosed());  
 try{  
 byte [] mybytearray = new byte [size];  
 fos = new FileOutputStream(FILE\_TO\_RECEIVED+fileName);  
 bos = new BufferedOutputStream(fos);  
 bytesRead = is.read(mybytearray,0,mybytearray.length);  
 current = bytesRead;  
 do {  
 bytesRead =  
 is.read(mybytearray, current, (mybytearray.length-current));  
 if(bytesRead >= 0) current += bytesRead;  
 } while(bytesRead > 0);  
   
 bos.write(mybytearray, 0 , current);  
 bos.flush();  
 System.out.println("File " + FILE\_TO\_RECEIVED+fileName  
 + " downloaded (" + current + " bytes read)");  
 Process p=Runtime.getRuntime().exec("cmd /c javac "+fileName);   
 p.waitFor();   
 BufferedReader reader=new BufferedReader(new InputStreamReader(p.getInputStream()));   
 String line=reader.readLine();   
 while(line!=null)   
 {   
 System.out.println(line);   
 line=reader.readLine();   
 }

while(true){  
 System.out.println(socket.isClosed());  
 FileInputStream fis = null;  
 BufferedInputStream bis = null;  
 OutputStream osf = null;  
 try{  
 File myFile = new File (FILE\_TO\_RECEIVED+"Test.class");  
 fis = new FileInputStream(myFile);  
 byte mybytearray1[] = new byte [(int)myFile.length()];  
 bis = new BufferedInputStream(fis);  
 bis.read(mybytearray1,0,mybytearray1.length);  
 osf = socket.getOutputStream();  
 System.out.println("Sending " + FILE\_TO\_RECEIVED + "(" + mybytearray1.length + " bytes)");  
 osf.write(mybytearray1,0,mybytearray1.length);  
 osf.flush();  
 System.out.println("Done.");  
 }  
 finally{  
 if (bis != null) bis.close();  
 if (osf != null) osf.close();  
 }  
 break;  
 }  
 }  
 catch(Exception e){  
 System.out.println(e);  
 }  
 finally{  
 if (fos != null) fos.close();  
 if (bos != null) bos.close();  
 }  
 break;  
 }  
   
 }  
 }  
 catch(IOException e){  
 System.out.println(e);  
 }  
 finally{  
 try  
 {  
 socket.close();  
 }  
 catch(IOException e){}  
 }  
 }  
}

**Client:**

package osclient;  
import java.io.\*;  
import java.net.\*;  
import java.util.Scanner;  
public class OsClient {  
   
 public final static int SOCKET\_PORT = 3000; public final static String SERVER = "127.0.0.1";  
 public final static String FILE\_TO\_SEND = "C:\\Users\\Umesh Ramchandani\\Desktop\\OS\\";   
 private static Socket socket;  
 public final static int FILE\_SIZE = 6022386;  
  
 public static void main(String[] args) throws IOException {  
 try{  
 String host = "localhost";  
 int port = 3000;  
 InetAddress address = InetAddress.getByName(host);  
 socket = new Socket(address, port);  
 System.out.println("Enter the filename");  
 Scanner sc = new Scanner(System.in);  
 String fileName = sc.next();  
 File myFile = new File (FILE\_TO\_SEND+fileName);  
 byte mybytearray[] = new byte [(int)myFile.length()];  
 String sendMessage = fileName+" "+myFile.length()+"\n";  
 OutputStream os = socket.getOutputStream();  
 OutputStreamWriter osw = new OutputStreamWriter(os);  
 BufferedWriter bw = new BufferedWriter(osw);  
 bw.write(sendMessage);  
 bw.flush();  
 InputStream is = socket.getInputStream();  
 InputStreamReader isr = new InputStreamReader(is);  
 Scanner br = new Scanner(isr);  
 int status = br.nextInt();  
 if( status == 200){  
 System.out.println("Successfull");  
 FileInputStream fis = null;  
 BufferedInputStream bis = null;  
 OutputStream osf = null;  
 try{  
 fis = new FileInputStream(myFile);  
 bis = new BufferedInputStream(fis);  
 bis.read(mybytearray,0,mybytearray.length);  
 osf = socket.getOutputStream();  
 System.out.println("Sending " + FILE\_TO\_SEND + "(" + mybytearray.length + " bytes)");  
 osf.write(mybytearray,0,mybytearray.length);  
 osf.flush();  
 System.out.println("Done.");  
 System.out.println(socket.isClosed());  
 System.out.println(socket.isClosed());  
 int bytesRead;  
 int current = 0;  
 FileOutputStream fos = null;  
 BufferedOutputStream bos = null;  
 byte [] classarray = new byte [FILE\_SIZE];  
 fos = new FileOutputStream(FILE\_TO\_SEND+"Test.class");  
 bos = new BufferedOutputStream(fos);  
 bytesRead = is.read(classarray,0,classarray.length);  
 current = bytesRead;  
 System.out.println(bytesRead);  
 do {  
 bytesRead =  
 is.read(classarray, current, (classarray.length-current));  
 if(bytesRead >= 0) current += bytesRead;  
 System.out.println(bytesRead);  
 } while(bytesRead > -1);  
  
 bos.write(classarray, 0 , current);  
 bos.flush();  
 System.out.println("File " + FILE\_TO\_SEND+fileName  
 + " downloaded (" + current + " bytes read)");  
 }  
 finally{  
   
 if (bis != null) bis.close();  
 if (osf != null) osf.close();  
 }  
   
 }  
 }  
 catch(IOException e)  
 {  
 System.out.println(e);  
 }  
 finally{  
 try  
 {  
 System.out.println("Here");  
 socket.close();  
 }  
 catch(IOException e)  
 {  
 System.out.println(e);  
 }  
 }  
 }  
}

**Output:**

**Client:**

run:

Enter the filename

Test.java

Successfull

Sending C:\Users\Umesh Ramchandani\Desktop\OS\(280 bytes)

Done.

File C:\Users\Umesh Ramchandani\Desktop\OS\Test.java downloaded (399 bytes read)

Here

BUILD SUCCESSFUL (total time: 6 seconds)

**Server:**

run:

Server Started and listening to the port 3000

Message received from client is Test.java280

Message sent to the client is 200

File C:\Users\Umesh Ramchandani\Documents\NetBeansProjects\OsServer\Test.java downloaded (280 bytes read)

Sending C:\Users\Umesh Ramchandani\Documents\NetBeansProjects\OsServer\(399 bytes)

Done.

**Conclusion**:

Hence we learnt how to design and develop a Network Compiler.