Projet: Développement d'un système de partage de fichiers

Réalisé par:

Mouhib Ben Jemaa

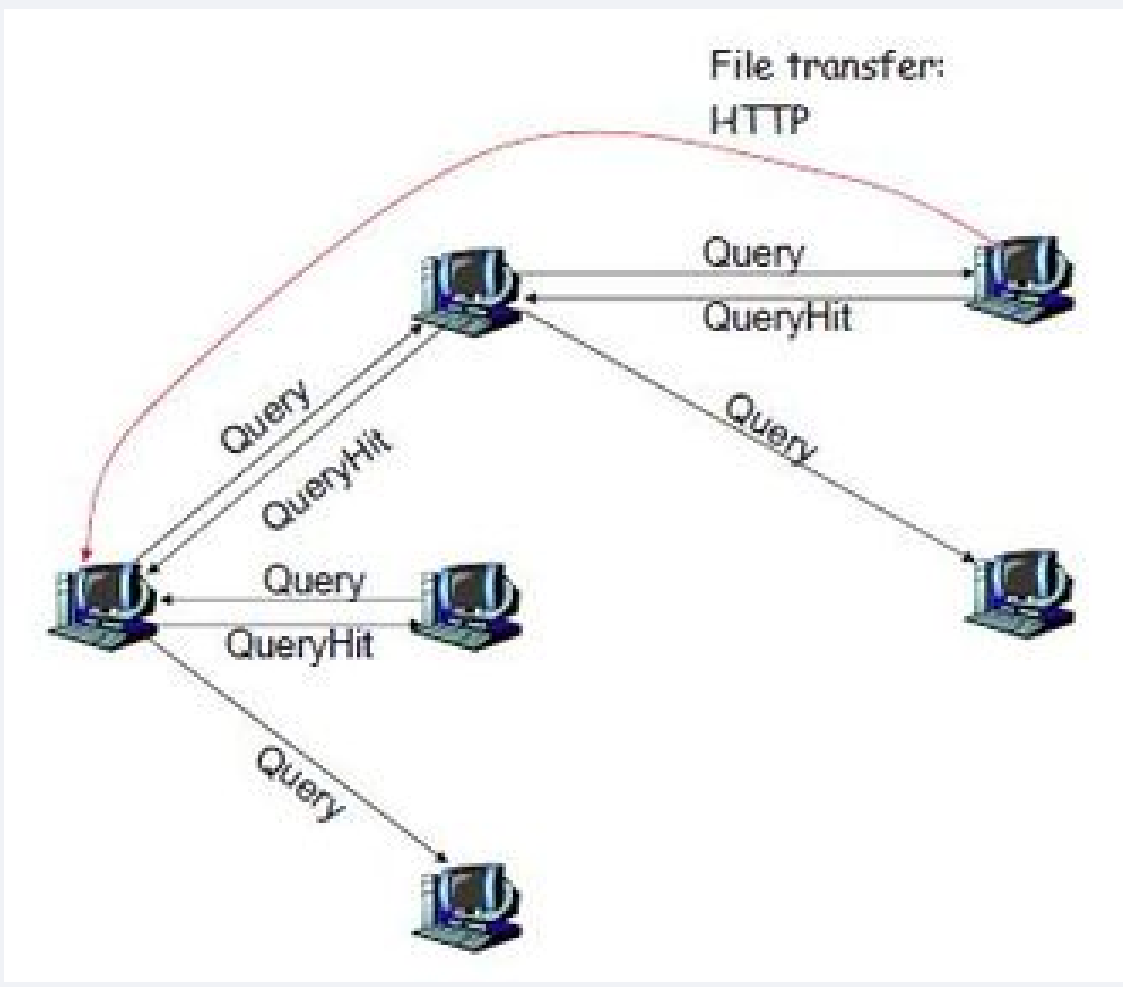
Souha Loulou

RT4 groupe 1

Gnutella Peer-To-Peer

In this project we chose to work with Gnutella Peer-To-Peer.

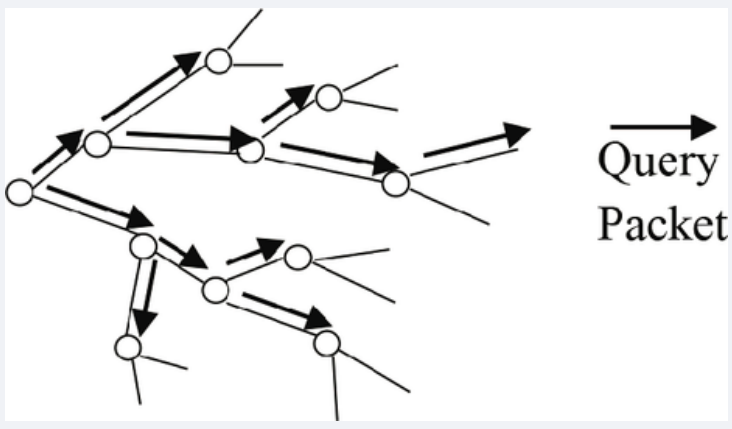
It a decentralized computer protocol for peer-to-peer file search and transfer.
It is composed mainly by 5 different messages :
*- "Ping" : query from a component in a network to another one, to determine whether there is a connection to it
*- "Pong" : the answer to a ping, much more like an "Echo response".
*- "Push" : a style of Internet-based communication where the request for a given transaction is initiated by the publisher or central server.
*- "query" : The primary mechanism for searching the distributed network.
*- "queryHit" : It is a reply to a query from a component in the Gnutella Network and it sent over the reverse path.



Gnutella general architecture

Gnutella uses the "Flooding Protocol" to speak to all its neighbors which consists of a method to search for a resource on a peer-to-peer network. Each Peer broadcasts queries for all the neighbours, and each neighbour does the same with its attached neighbours until a certain result is achieved.
That result is indeed when the queried peer has the desired object, it sends a message back to the querying peer.
We define the scalability here as a limited scope flooding.
We also note that it isn't necessary to have a hierarchy in the Gnutella network.

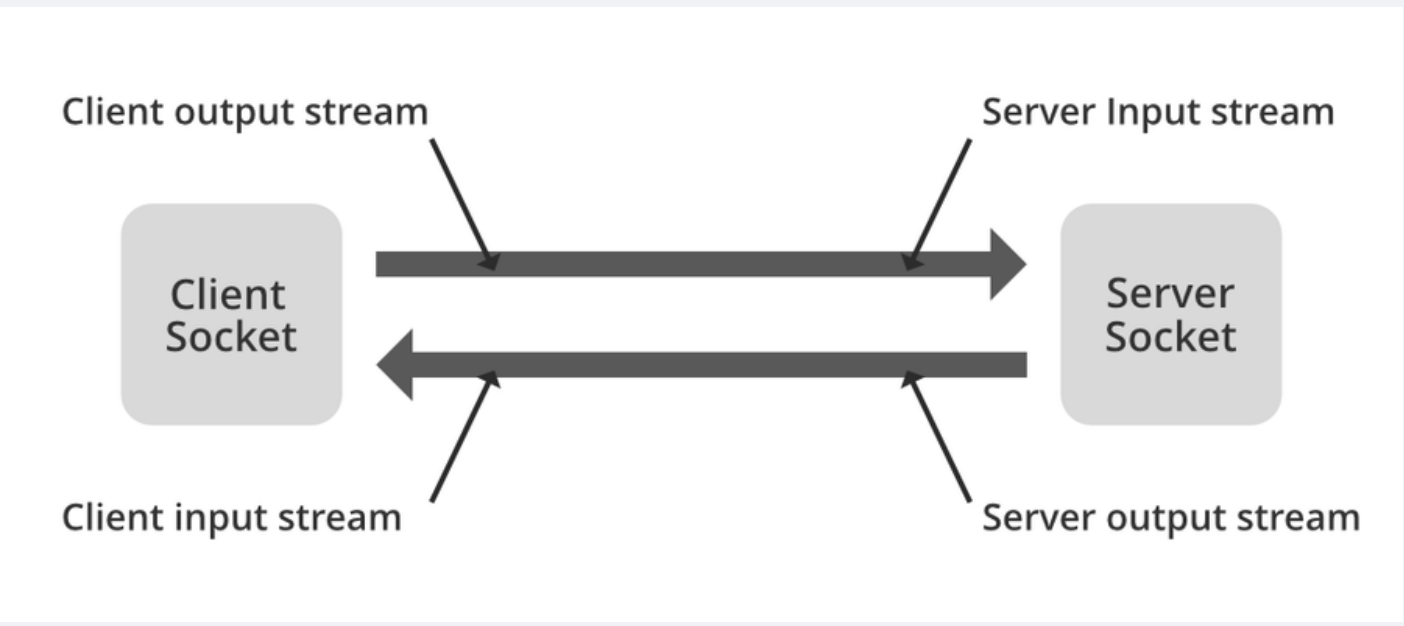
We define the scalability here as a limited scope flooding.
We also note that it isn't necessary to have a hierarchy in the Gnutella network.



Flooding

1st approach: The method using Sockets

For this method, we have used the Sockets of Java.
It is a communication tunnel used by a Thread (which will also be implemented through the extension of the Thread class).
This tunnel allows to send or receive a result between a client and a remote server interconnected by a port.



Socket general structure

The principle we used is that each Peer can act as a client and a server at the same time. This approach consists in the use of the principle of Multithreads and Sockets.

***- The code is composed of 5 files:**

***) Main.java :**
It is the main class where all the operation is located, we get the useful information (file name, directories, peer ID, ports... etc).
We define the clients one by one as Threads, and these clients are in fact the neighbors of each peer according to a topology which will be defined in the file «topology.txt». And we wait for the end of all the threads.

```
//s'il existe un file à downloader, cette partie du code sera executée:
if (fileToDownload_exist){
//long startTime = System.currentTimeMillis();

//recuperer le fichier à downloader
String fileToDownload_Name=scan.nextLine();
System.out.println("The file to be downloaded is :  "+fileToDownload_Name);

System.out.println("~~[[ Start time : "+System.currentTimeMillis());
++nbMsgCount;
msgID= peerID +"."+nbMsgCount;

//Création d'un thread client pour chaque pair voisin
String[] neighbours=prop.getProperty("peer"+peerID+".next").split(regex: ",");
TTL = neighbours.length; //ttl selon la longueur de la liste des voisins


for(int i=0;i<neighbours.length;i++){
    int connectionPort = Integer.parseInt(prop.getProperty("peer"+neighbours[i]+".port")); //port de connection pour chaque voisin Peer
    int neighbourPeer = Integer.parseInt(neighbours[i]); //recuperation des voisins peers

    System.out.println("connection port : "+connectionPort+" || neighbourPeer : "+neighbourPeer);
    //Initialisation d'un client voisin selon les données dans topology.txt
    Client client = new Client(connectionPort, neighbourPeer, fileToDownload_Name, msgID, peerID, TTL);
    //execution du client voisin dans un nouveau thread
    Thread t=new Thread(client);
    t.start();
    thread.add(t);
    peers.add(client); //ajout du client dans la liste des clients

    }
```

*) Server.java : (Un Thread):

For this class we will establish the connection with the clients accepting the request of their Sockets.
We define another class called « Download » that is also a Thread. In this class we will search for the file to download locally.
We check if the message received by the server is duplicated or not (that is to say that the ID of the message has not already appeared) and we instantiate the neighboring clients to this Peer that acts as a server. The list of neighbors can be retrieved from the file « topology.txt ».
Lastly, we define the « ClientasServer » class that will establish the sending of the desired file to the client that requested it, that is, the Peer server will behave like a client.

```
public class Server extends Thread{
    int peerID;
    String directory;
    int port;
    ServerSocket serverSocket = null;
    Socket socket = null;
    String msg;
    static ArrayList<String> msgList;
    static boolean duplicateStatus = false;
    static int[] peersHaveTheFile = new int[10];
    public Server(int port, String directory, int peerID) {
        this.port = port;
        this.directory = directory;
        this.peerID = peerID;
        msgList=new ArrayList<String>();
    }

    public void run(){
        //Création d'un socket de serveur
        try{
            serverSocket = new ServerSocket(port);
        }
        catch(IOException io){
            io.printStackTrace();
        }
        //Accepter la création d'un socket serveur pour chaque demande
        while(true){
            try{
                socket = serverSocket.accept();
                System.out.println("Connected to client at "+socket.getRemoteSocketAddress()+" with peer "+peerID);
                //Initialiser la classe Download pour écouter les telechargements
                new Download(socket,directory,peerID,msgList).start();
```

```
public void ClientasServer(int ClientPeerID,int ServerPortNb,String filename,String sharedDir)
{
try{
Socket clientasserversocket=new Socket(host: "localhost",ServerPortNb); //creation du socket
// Les flux d'objets entrée et sortie
ObjectOutputStream ooos=new ObjectOutputStream(clientasserversocket.getOutputStream());
ooos.flush();
ObjectInputStream oois=new ObjectInputStream(clientasserversocket.getInputStream());
ooos.writeObject(filename);
//lecture du longueur fichier en tant qu'une variable de bytes
int readbytes=(int)oois.readObject();
System.out.println("bytes transferred: "+readbytes);

//lecture du fichier en tant qu'une variable de bytes
byte[] b=new byte[readbytes];
oois.readFully(b);


String currentDir = System.getProperty(key: "user.dir"); // Pour trouver le répertoire où le code est installé

Path fileName= Path.of(currentDir+"\\PeerFilesdirectory.txt"); //trouver le nom de fichier concerné pour le telechargement


String PeersFilesDirectory = Files.readString(fileName);// Pour lire le fichier PeerFilesdirectory.txt contenant le nom du repertoi
sharedDir = PeersFilesDirectory;


//fichier à recevoir
OutputStream fileos=new FileOutputStream(sharedDir+"//"+filename);
BufferedOutputStream bos=new BufferedOutputStream(fileos);
bos.write(b, off: 0,(int) readbytes); //envoi du fichier octet par octet
```

*) MessageSpecs.java :

This class defines the format of the message that will be exchanged in the network.

It is composed of

*- msgID: the unique id of the messages

*- senderPeerID: the ID of the Peer who sent the message

*-fileName: the name of the file to download

*- TTL: the lifetime of a message

```java
import java.io.Serializable;
//cette classe va definir la structure des messages echangees dans le reseau

public class MessageSpecs implements Serializable {
    String msgID; //l'id unique des messages
    int senderPeerID; //l'ID du peer qui a envoyé le message
    String fileName; //le nom de fichier à telecharger
    int TTL; //la durée de vie d'un message
}
```

*) DownloadFromServer.java: (Un Thread):

This class accepts a connection from a client and instantiate the class «Telechargement».

In this class, we perform the download and the reception of the file by the Server

```
wnloadFromServer.java > Telechargement > run()
import java.nio.file.Path;

public class DownloadFromServer extends Thread{
    //cette classe servira pour le telechargement depuis le serveur
    int nbPort;
    String directory;
    ServerSocket serverSocket;
    Socket socket;
    DownloadFromServer(int nbPort, String directory){
        this.nbPort = nbPort;
        this.directory = directory;
    }

    public void run(){
        try{
            serverSocket = new ServerSocket(nbPort); //creation d'un socket serveur
        }
        catch(IOException io){
            io.printStackTrace();
        }

        try{
            socket = serverSocket.accept();
        }
        catch(IOException io){
            io.printStackTrace();
        }
        //initialisation de la classe Telechargement pour lancer un nouveau telechargement de ce niveau
        new Telechargement(socket, nbPort, directory).start();
    }

}
```

```java
}

class Telechargement extends Thread{
    //cette classe est où se déroule le telechargement au niveau d'un Peer qui se comporte comme un serveur
    int nbPort;
    Socket socket;
    String directory;
    String fileName;
    Telechargement(Socket socket, int nbPort, String directory){
        this.socket = socket;
        this.nbPort = nbPort;
        this.directory = directory;
    }

    public void run(){
        try{
            // les flux d'entrée et sortie d'objets
            InputStream is=socket.getInputStream();         //Connecter le Client jouant le role de serveur au fichier Clie
            ObjectInputStream ois=new ObjectInputStream(is);
            OutputStream os=socket.getOutputStream();
            ObjectOutputStream oos=new ObjectOutputStream(os);

            fileName=(String)ois.readObject();                  //Nom du fichier à télécharger

            //while(true)
            //  {
                    File myFile = new File(directory+"//"+fileName);
                    long length = myFile.length();
                    System.out.println("File length: "+length+ " File name : "+fileName);
                    //******* */
```

*) Client.java: (Un Thread):

This class creates a connection and sends the message according to the parameters defined in the «MessageSpecs.java».

```
nt.java > Client > run()
import java.io.OutputStream;
import java.net.Socket;

public class Client extends Thread {

    int connectionPort;
    int peerToConnect;
    String fileToDownload;
    Socket socket=null;
    int[] peersArray;

    MessageSpecs message=new MessageSpecs();
    String msgID;
    int senderPeerID;
    int TTL;

    public Client(int connectionPort, int peerToConnect, String fileToDownload, String msgID, int senderPeerID, int TTL) {
        this.connectionPort=connectionPort;
        this.peerToConnect=peerToConnect;
        this.fileToDownload=fileToDownload;
        this.msgID=msgID;
        this.senderPeerID=senderPeerID;
        this.TTL=TTL;
    }
    public void run(){
        try{
            socket = new Socket(host: "localhost",connectionPort); //etablissement d'une connection avec le client
            //les flux d'entree et de sortie d'objets
            OutputStream os=socket.getOutputStream();
            ObjectOutputStream objOS=new ObjectOutputStream(os);
            InputStream is=socket.getInputStream();
            ObjectInputStream objIS=new ObjectInputStream(is);
```

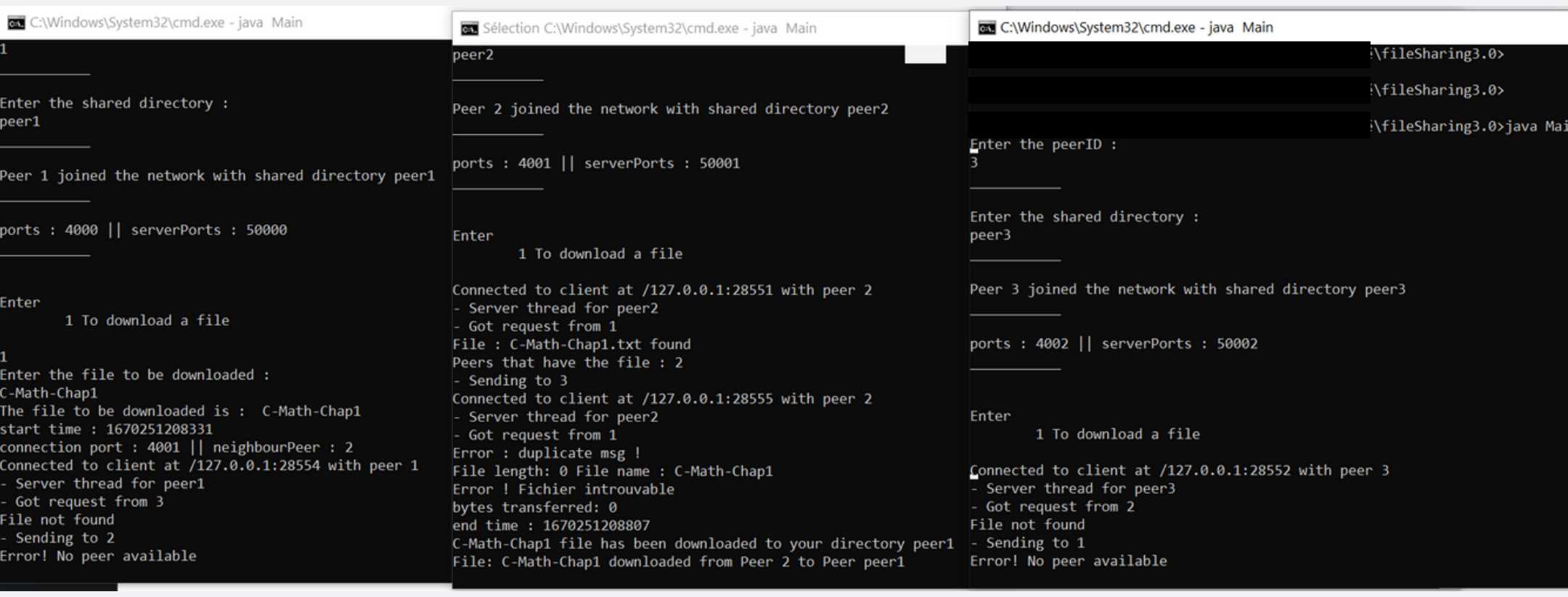
1st approach: Execution Results

In each instance of CMD, we run a peer. We'll have 3 peers, with the respective IDs , "1", "2" and "3".

In this example, the peer "1" says that he's looking for a file. He enters the name of the file "C-Math-Chap1.txt". Then he forwards a message to his neighbor peer "2". Peer "2" does a local search in his files and he says that he found the file.
He establishes a "download" communication with peer "1" to send the wanted file to its directory "peer1".

Peer "2" forwards the message to peer "3" and he does a local research but he can't find the file so he sends "File not found". Peer "3" sends back to peer "1" to look for the same file, but "1" says that he doesn't have the file because the download starts after the end of the search messages in all the network.
In other words, the downloads starts in the first peer that finds the wanted file in his directory and also after all the search messages went through the whole network peers.

We also print the start time and end time of the transmission (since the moment we type the file to search for, untill the moment the download has ended), then we calculate the response time by using a simple substraction.

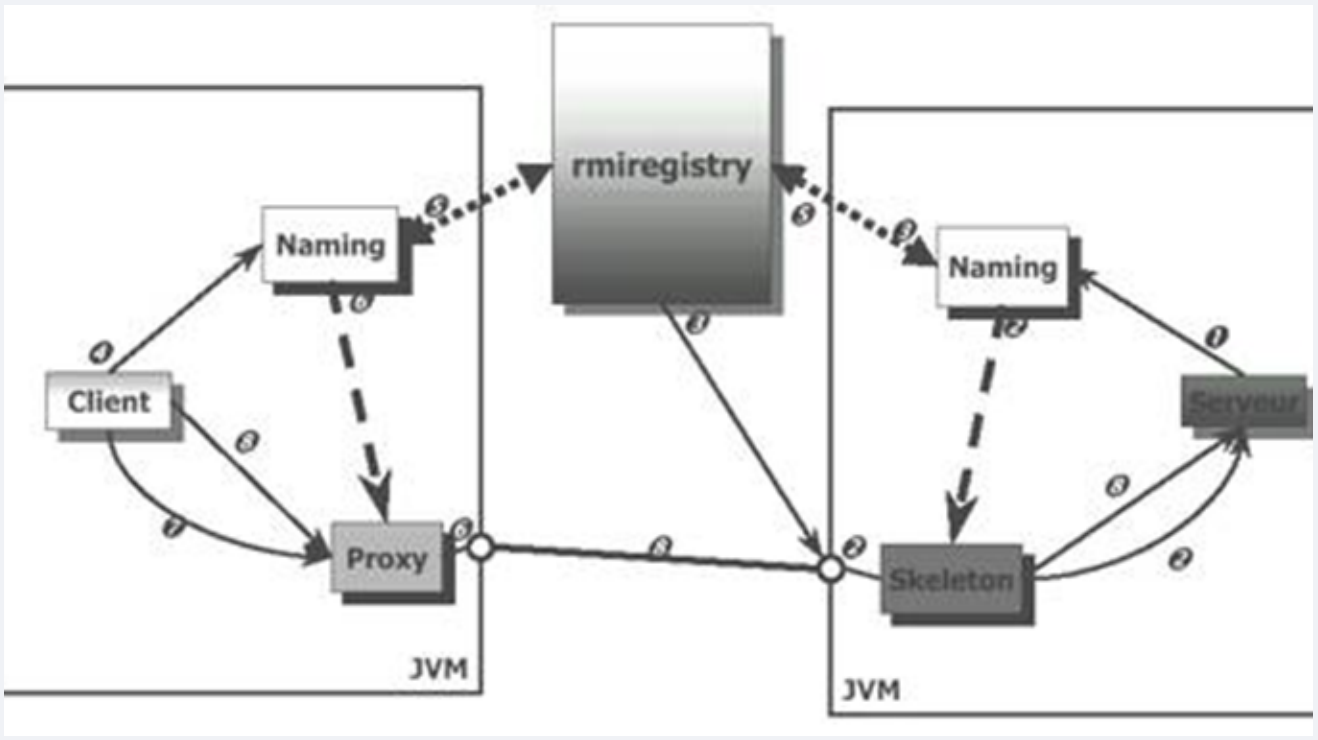


2nd approach: The method using RMI

For this method, we have used RMI:

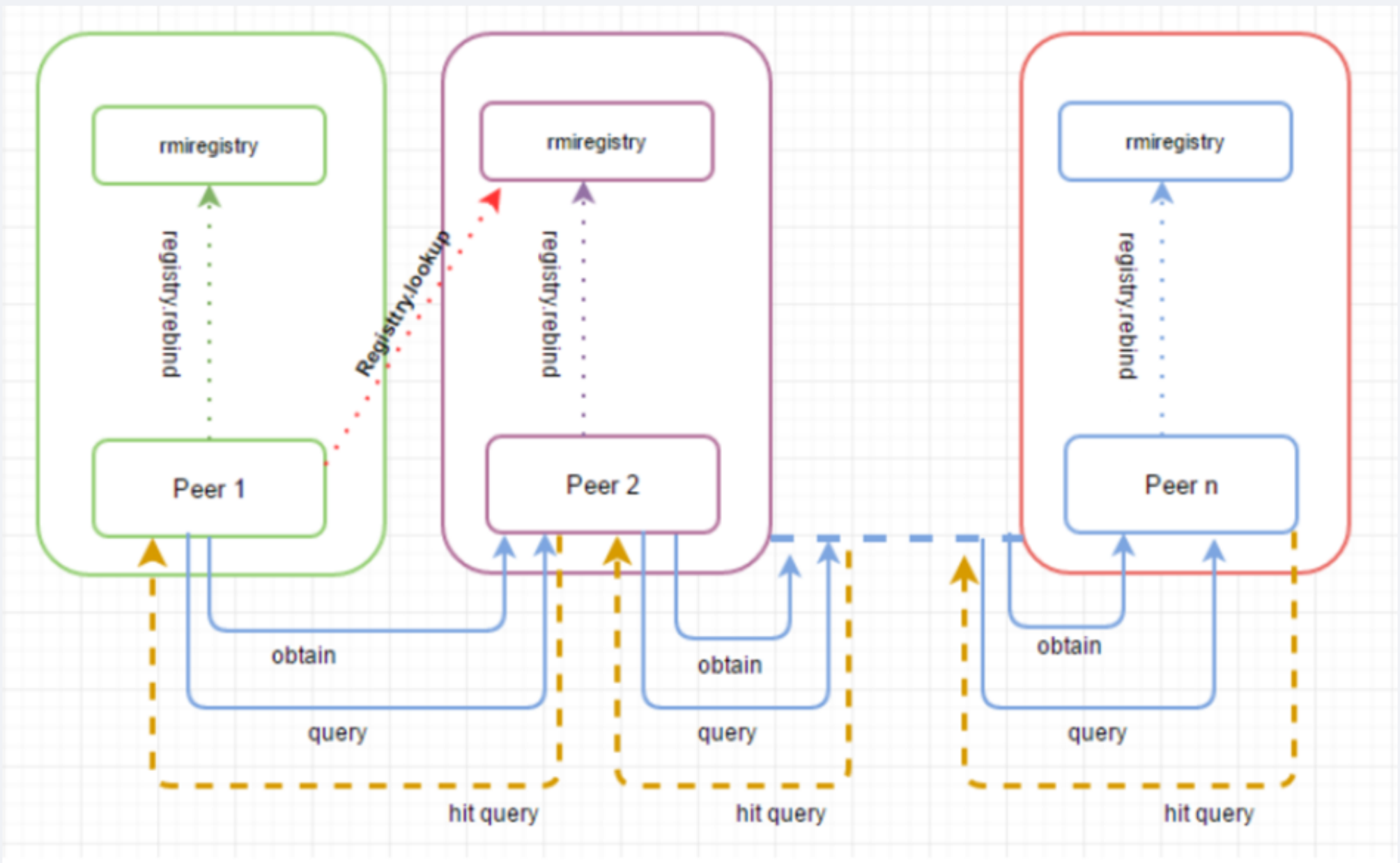
RMI:

- **The call can be made between java objects that run on separate JVMs.**
- **The call can be made on the same machine or on machines connected in network.**
- **The exchanges respect a protocol: Remote method protocol.**



In this example, we suppose that the topologie is static
We will run the program first locally on the same machine then we will proceed the execution of the program using two different machines connected to the same network.

*) Code explication :



Gnutella style Peer to Peer file management design with RMI

*) PeerInterface.java: interface :

*- obtain(filename)

*- query(fromPeerId,msgId,fielname)

=> remote methods

```java
gnutellaP2P > PeerInterface.java > PeerInterface
package gnutellaP2P;
import java.rmi.*;
//
// Interface pour les méthodes remote
public interface PeerInterface extends Remote{
    public byte[] obtain(String filename)throws RemoteException;
    public HitQuery query(int fromPeerId,String msgId,String fileName)throws RemoteException;
}
```

*) HitQuery.java: class:

This class is used to handle the details of the result of every query. It contains properties of the peer details of the requested file and the path through which search performed.

```java
1   package gnutellaP2P;
2
3   import java.io.Serializable;
4   import java.util.ArrayList;
5
6   //class pour HitQuery : réponse des voisins
7   public class HitQuery implements Serializable {
8
9       private static final long serialVersionUID = 1L;
10      public ArrayList<PeerDetails> foundPeers = new ArrayList<PeerDetails>();
11      public ArrayList<String> paths = new ArrayList<String>();
12  }
```

*) PeerInterfaceRemote.java: class :

implement l'interface PeerInterface.java

*- ***Attributes:*** serialVersionUID = 1L; sharedDirectory;

ArrayList<String> localFiles=new ArrayList<String>();
ArrayList<String> processdMsgIds;
peerId;currentPeerPort;

*- Define remote methods:

query(fromPeerId,msgId,fielname)

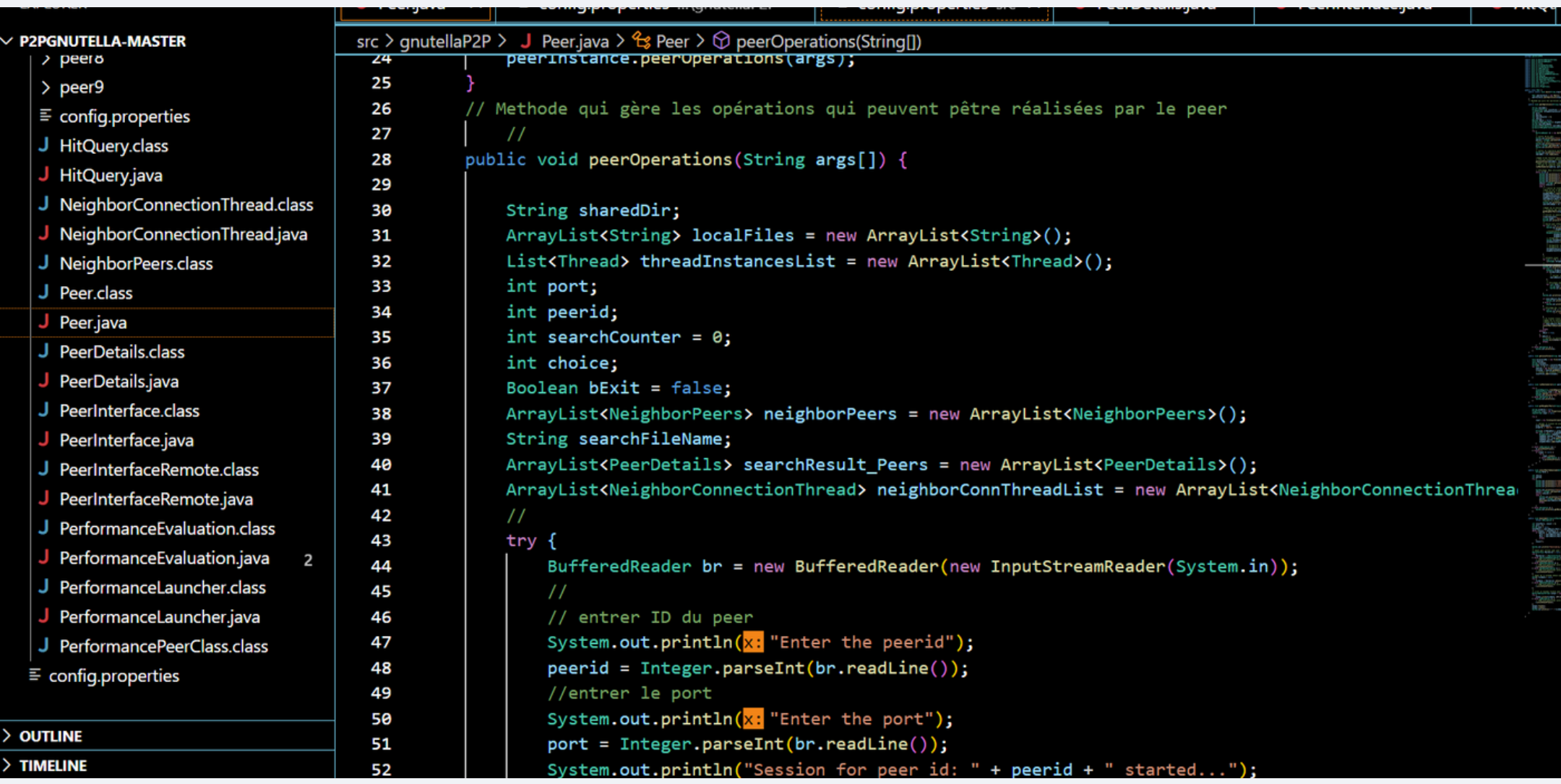
=> This remote method takes the filename, message id as arguments. It checks whether the peer already processed that msgId or not. If it is already processed, it will skip the process and return to the called client. If message id is not there with the peer then it will perform the search locally. It will establish the connection with its neighbor peers and will get the results. All these results will be sent back to the called client.

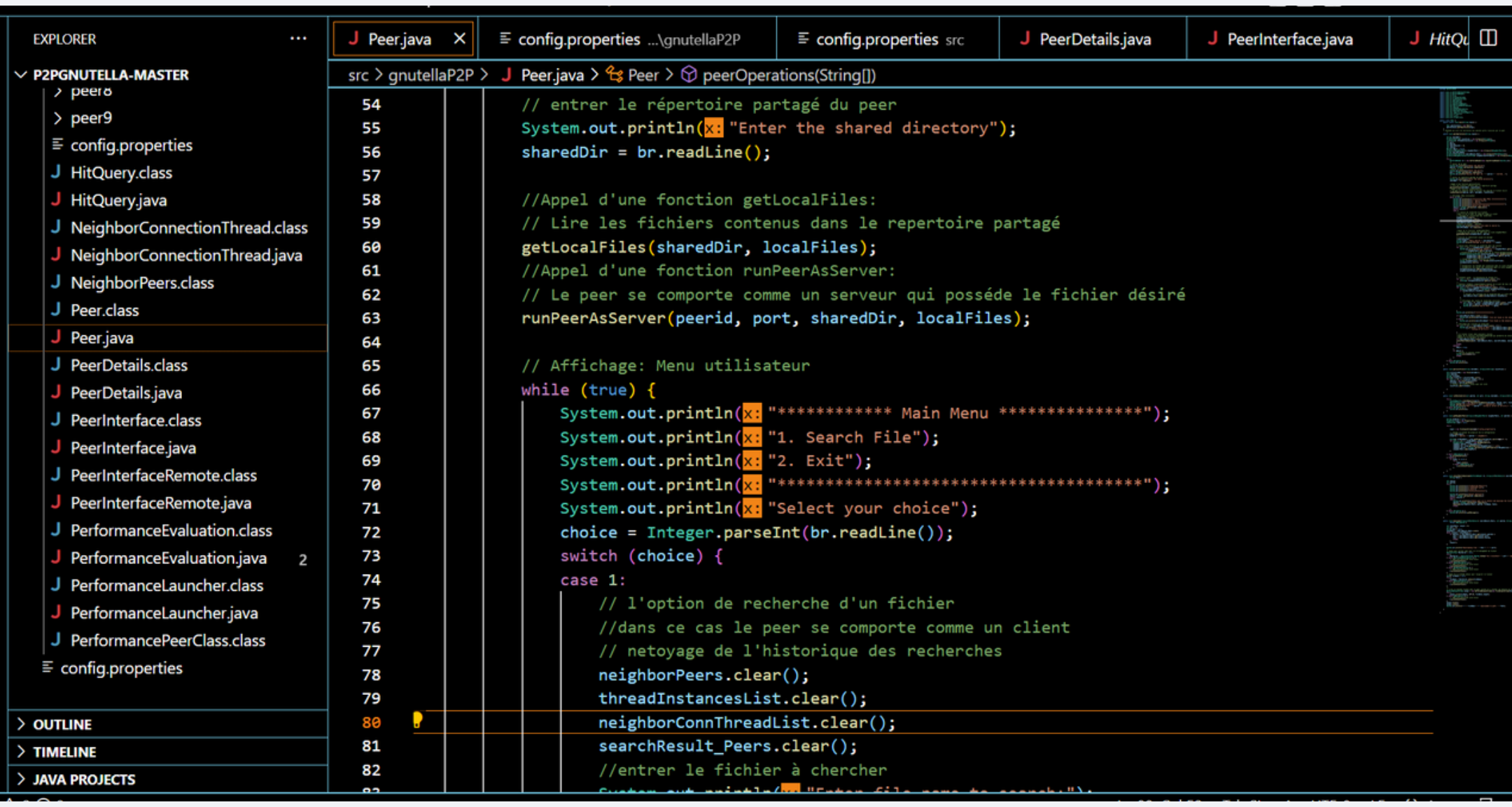
Obtain(filename)

=> This remote method takes the file name as an argument. It will send the file contents as bytes and client will create a new file with these bytes of data.

*) Peer.java: class:

*- Enter the directory path that is going to be shared with other peers
Prompts for file name to search
*- Get the neighbor peers from the config file and get the result from those peers.
*- Display the list of peers containing the requested file.
*- Prompt to enter the peer id from where the file should be downloaded
*- Establish connection with peer and download the file.

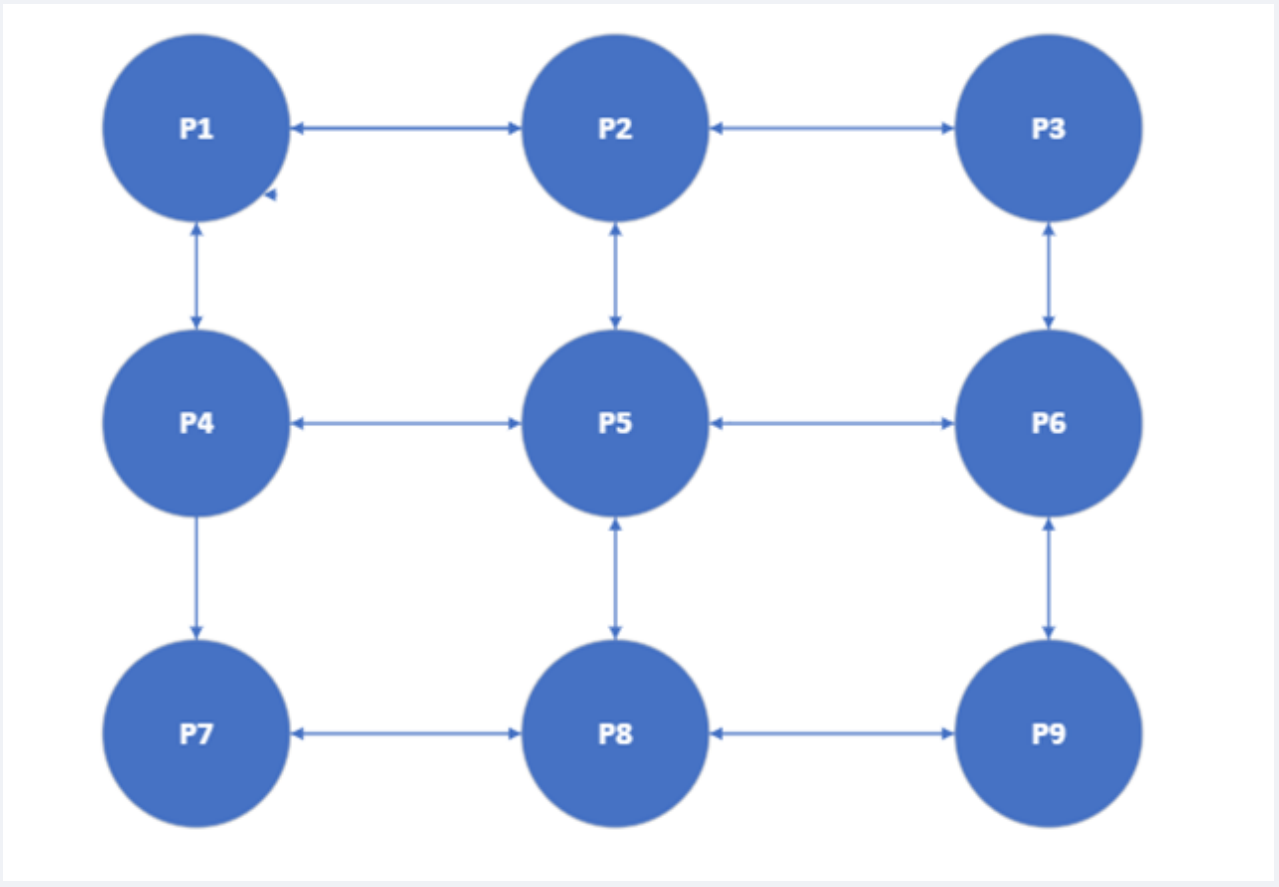




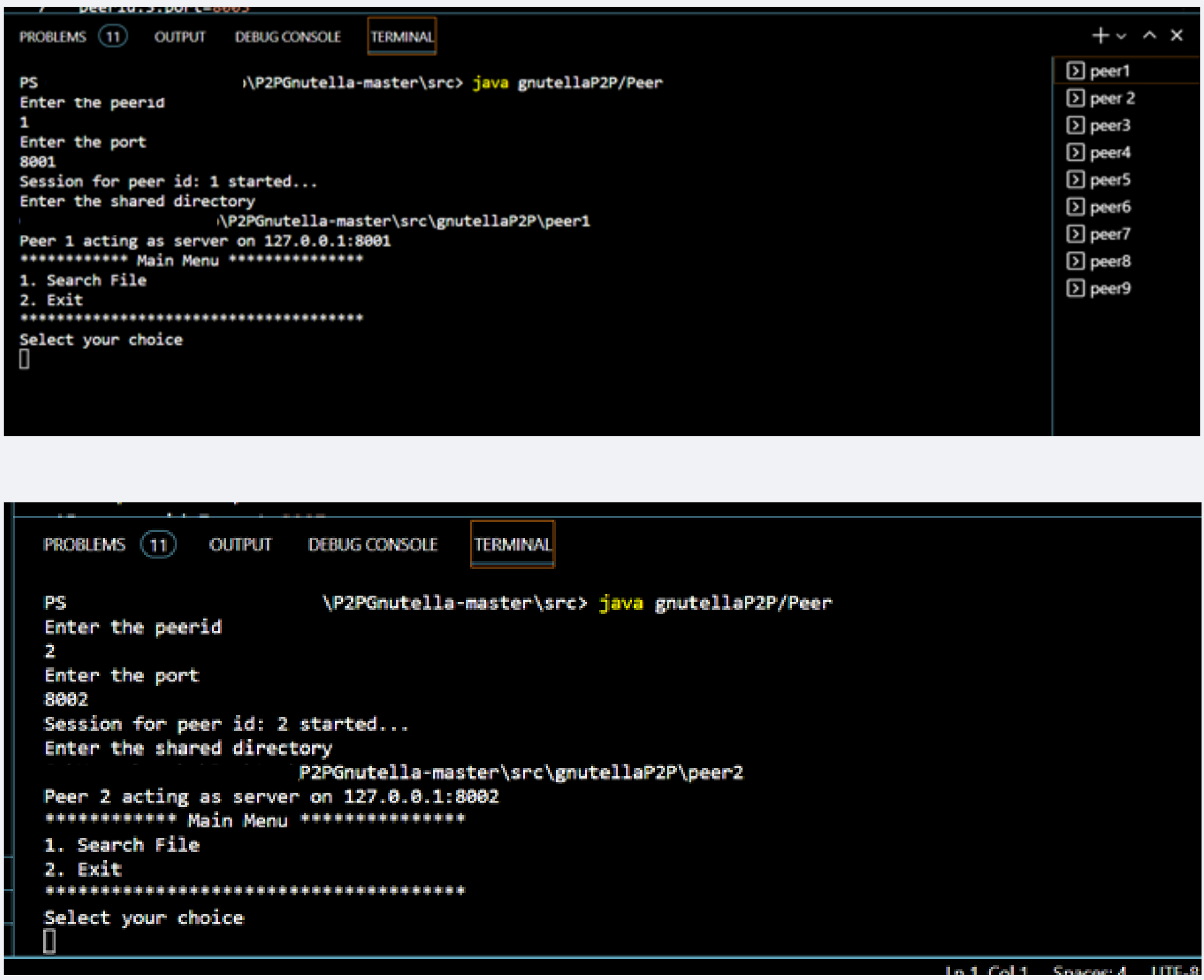
2nd approach: Execution Results

*) Execution of the code :

Topologie Mesh 3*3:



- Execution of Peer.java class for 9 peers:



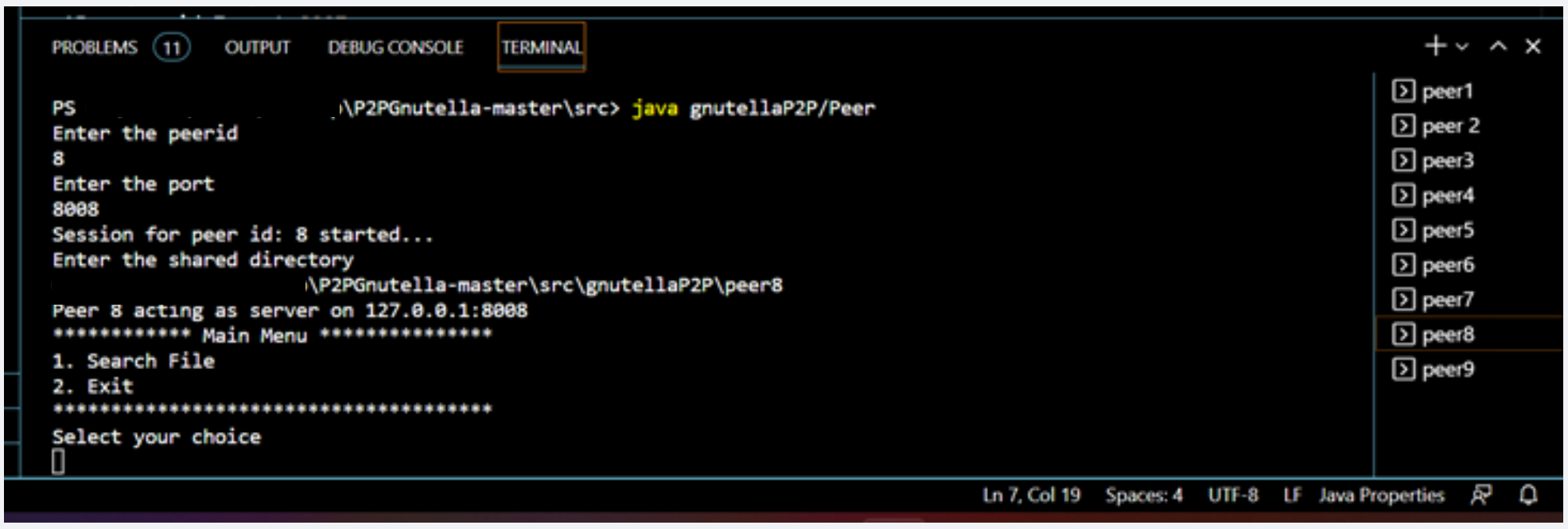
```
PS                      \P2PGnutella-master\src> java gnutellaP2P/Peer
Enter the peerid
3
Enter the port
8003
Session for peer id: 3 started...
Enter the shared directory
                    \P2PGnutella-master\src\gnutellaP2P\peer3
Peer 3 acting as server on 127.0.0.1:8003
************ Main Menu ***************
1. Search File
2. Exit
**************************************
Select your choice
```

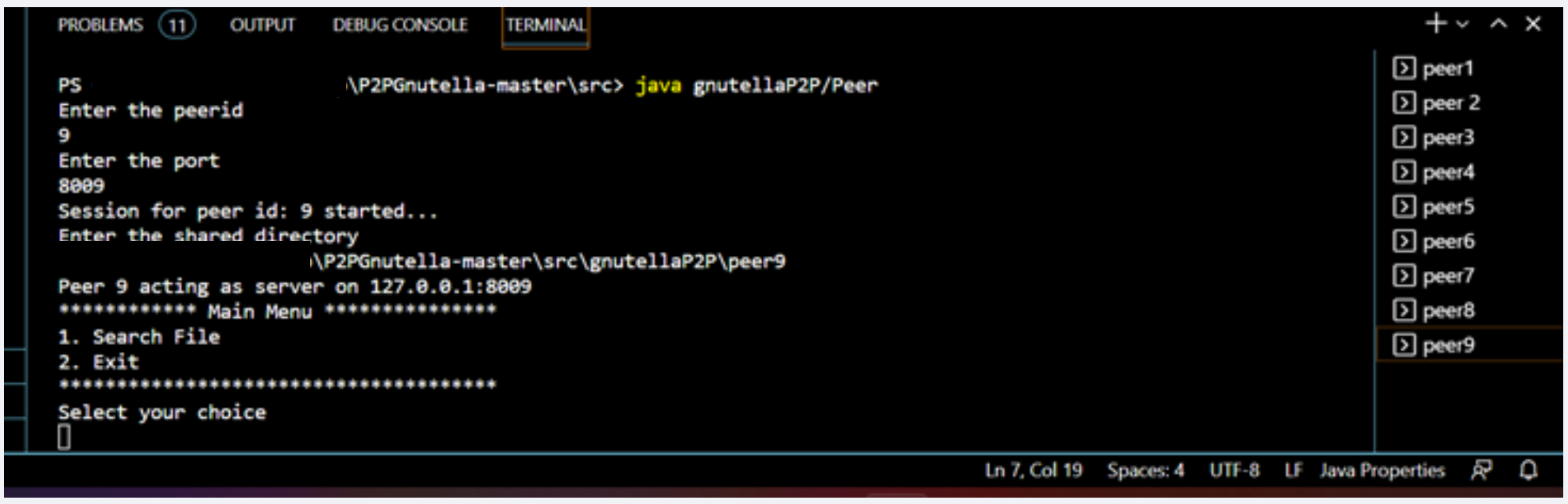
```
PS                      \P2PGnutella-master\src> java gnutellaP2P/Peer
Enter the peerid
4
Enter the port
8004
Session for peer id: 4 started...
Enter the shared directory
                    \P2PGnutella-master\src\gnutellaP2P\peer4
Peer 4 acting as server on 127.0.0.1:8004
************ Main Menu ***************
1. Search File
2. Exit
**************************************
Select your choice
```

```
PS                      \P2PGnutella-master\src> java gnutellaP2P/Peer
Enter the peerid
5
Enter the port
8005
Session for peer id: 5 started...
Enter the shared directory
                    \P2PGnutella-master\src\gnutellaP2P\peer5
Peer 5 acting as server on 127.0.0.1:8005
************ Main Menu ***************
1. Search File
2. Exit
**************************************
Select your choice
```

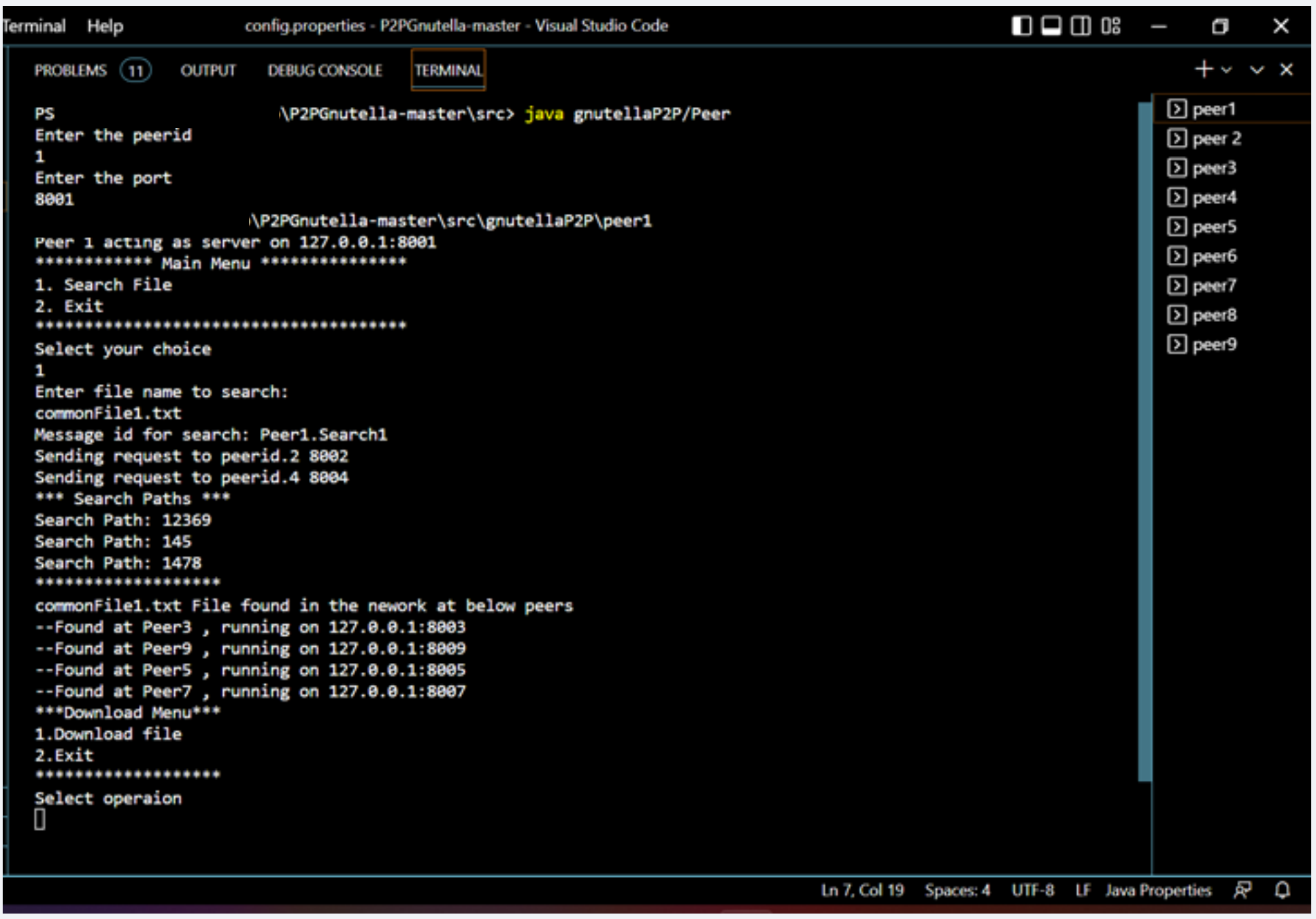
```
PS                      \P2PGnutella-master\src> java gnutellaP2P/Peer
Enter the peerid
6
Enter the port
8006
Session for peer id: 6 started...
Enter the shared directory
                    \P2PGnutella-master\src\gnutellaP2P\peer6
Peer 6 acting as server on 127.0.0.1:8006
************ Main Menu ***************
1. Search File
2. Exit
**************************************
Select your choice
```

```
PS                      \P2PGnutella-master\src> java gnutellaP2P/Peer
Enter the peerid
7
Enter the port
8007
Session for peer id: 7 started...
Enter the shared directory
                    \P2PGnutella-master\src\gnutellaP2P\peer7
Peer 7 acting as server on 127.0.0.1:8007
************ Main Menu ***************
1. Search File
2. Exit
**************************************
Select your choice
```





- peer 1 search fo commonFile1:



```
Terminal  Help                config.properties - P2PGnutella-master - Visual Studio Code

PROBLEMS 11   OUTPUT   DEBUG CONSOLE   TERMINAL

PS                      p\P2PGnutella-master\src> java gnutellaP2P/Peer
Enter the peerid
2
Enter the port
8002
Session for peer id: 2 started...
Enter the shared directory
                    p\P2PGnutella-master\src\gnutellaP2P\peer2
Peer 2 acting as server on 127.0.0.1:8002
************ Main Menu ***************
1. Search File
2. Exit
**************************************
Select your choice
Incoming Request to peer 2: From - 1 Search locally and send request to neighbours for msg id- Peer1.Search1
Local Search: File not found in the current peer
Outgoing Request from peer 2: Sending request to peerid.3 8003
Outgoing Request from peer 2: Sending request to peerid.5 8005
Incoming Request to peer 2: From - 5 Duplicate Request - Already searched in this peer- with message id - Peer1
.Search1
HitQuery: Send following result back to 1
--Found at Peer3 on localhost:8003
--Found at Peer9 on localhost:8009
```

peer1
peer 2
peer3
peer4
peer5
peer6
peer7
peer8
peer9

```
Terminal  Help                config.properties - P2PGnutella-master - Visual Studio Code

PROBLEMS 11   OUTPUT   DEBUG CONSOLE   TERMINAL

PS                      \P2PGnutella-master\src> java gnutellaP2P/Peer
Enter the peerid
3
Enter the port
8003
Session for peer id: 3 started...
Enter the shared directory
                    \P2PGnutella-master\src\gnutellaP2P\peer3
Peer 3 acting as server on 127.0.0.1:8003
************ Main Menu ***************
1. Search File
2. Exit
**************************************
Select your choice
 ncoming  equest to peer 3: From - 2 Search locally and send request to neighbours for msg id- Peer1.Search1
 ocal Search: File Found in the current peer
 utgoing  equest from peer 3: Sending request to peerid.6 8006
 it uery: Send following result back to 2
--Found at Peer3 on localhost:8003
--Found at Peer9 on localhost:8009
```

peer1
peer 2
peer3
peer4
peer5
peer6
peer7
peer8
peer9

```
Terminal  Help                config.properties - P2PGnutella-master - Visual Studio Code

PROBLEMS 11   OUTPUT   DEBUG CONSOLE   TERMINAL

PS                      \P2PGnutella-master\src> java gnutellaP2P/Peer
Enter the peerid
4
Enter the port
8004
Session for peer id: 4 started...
Enter the shared directory
                    \P2PGnutella-master\src\gnutellaP2P\peer4
Peer 4 acting as server on 127.0.0.1:8004
************ Main Menu ***************
1. Search File
2. Exit
**************************************
Select your choice
 ncoming  equest to peer 4: From - 1 Search locally and send request to neighbours for msg id- Peer1.Search1
 ocal Search: File not found in the current peer
 utgoing  equest from peer 4: Sending request to peerid.5 8005
 utgoing  equest from peer 4: Sending request to peerid.7 8007
 it uery: Send following result back to 1
--Found at Peer5 on localhost:8005
--Found at Peer7 on localhost:8007
```

peer1
peer 2
peer3
peer4
peer5
peer6
peer7
peer8
peer9

erminal Help config.properties - P2PGnutella-master - Visual Studio Code

PROBLEMS 11 OUTPUT DEBUG CONSOLE TERMINAL

```
PS                    \P2PGnutella-master\src> java gnutellaP2P/Peer
Enter the peerid
5
Enter the port
8005
Session for peer id: 5 started...
Enter the shared directory
C:\Users\souha\Desktop\P2PGnutella-master\src\gnutellaP2P\peer5
Peer 5 acting as server on 127.0.0.1:8005
************ Main Menu ***************
1. Search File
2. Exit
**************************************
Select your choice
 ncoming  equest to peer 5: From - 4 Search locally and send request to neighbours for msg id- Peer1.Search1
 ocal Search: File Found in the current peer
 ncoming  equest to peer 5: From - 2 Duplicate  equest -  lready searched in this peer- with message id - Peer1
.Search1
 utgoing  equest from peer 5: Sending request to peerid.2 8002
 utgoing  equest from peer 5: Sending request to peerid.6 8006
 utgoing  equest from peer 5: Sending request to peerid.8 8008
 it uery: Send following result back to 4
--Found at Peer5 on localhost:8005
 ncoming  equest to peer 5: From - 6 Duplicate  equest -  lready searched in this peer- with message id - Peer1
.Search1
 ncoming  equest to peer 5: From - 8 Duplicate  equest -  lready searched in this peer- with message id - Peer1
.Search1
```

peer1
peer 2
peer3
peer4
peer5
peer6
peer7
peer8
peer9

erminal Help config.properties - P2PGnutella-master - Visual Studio Code

PROBLEMS 11 OUTPUT DEBUG CONSOLE TERMINAL

```
PS                    \P2PGnutella-master\src> java gnutellaP2P/Peer
Enter the peerid
6
Enter the port
8006
Session for peer id: 6 started...
Enter the shared directory
                      \P2PGnutella-master\src\gnutellaP2P\peer6
Peer 6 acting as server on 127.0.0.1:8006
************ Main Menu ***************
1. Search File
2. Exit
**************************************
Select your choice
 ncoming  equest to peer 6: From - 3 Search locally and send request to neighbours for msg id- Peer1.Search1
 ocal Search: File not found in the current peer
 ncoming  equest to peer 6: From - 5 Duplicate  equest -  lready searched in this peer- with message id - Peer1
.Search1
 utgoing  equest from peer 6: Sending request to peerid.5 8005
 utgoing  equest from peer 6: Sending request to peerid.9 8009
 it uery: Send following result back to 3
--Found at Peer9 on localhost:8009
```

peer1
peer 2
peer3
peer4
peer5
peer6
peer7
peer8
peer9

Terminal Help config.properties - P2PGnutella-master - Visual Studio Code

PROBLEMS 11 OUTPUT DEBUG CONSOLE TERMINAL

```
PS                    \P2PGnutella-master\src> java gnutellaP2P/Peer
Enter the peerid
7
Enter the port
8007
Session for peer id: 7 started...
Enter the shared directory
                    \P2PGnutella-master\src\gnutellaP2P\peer7
Peer 7 acting as server on 127.0.0.1:8007
************ Main Menu ***************
1. Search File
2. Exit
**************************************
Select your choice
 ncoming  equest to peer 7: From - 4 Search locally and send request to neighbours for msg id- Peer1.Search1
 ocal Search: File Found in the current peer
 utgoing  equest from peer 7: Sending request to peerid.8 8008
 it uery: Send following result back to 4
--Found at Peer7 on localhost:8007
```

peer1
peer 2
peer3
peer4
peer5
peer6
peer7
peer8
peer9

Terminal Help config.properties - P2PGnutella-master - Visual Studio Code

PROBLEMS 11 OUTPUT DEBUG CONSOLE TERMINAL

```
P:                          \P2PGnutella-master\src> java gnutellaP2P/Peer
Enter the peerid
8
Enter the port
8008
Session for peer id: 8 started...
Enter the shared directory
                        \P2PGnutella-master\src\gnutellaP2P\peer8
Peer 8 acting as server on 127.0.0.1:8008
************ Main Menu ***************
1. Search File
2. Exit
*************************************
Select your choice
 ncoming  equest to peer 8: From - 7 Search locally and send request to neighbours for msg id- Peer1.Search1
 ocal Search: File not found in the current peer
 ncoming  equest to peer 8: From - 5 Duplicate  equest -  lready searched in this peer- with message id - Peer1
.Search1
 utgoing  equest from peer 8: Sending request to peerid.5 8005
 utgoing  equest from peer 8: Sending request to peerid.9 8009
 it uery: Send following result back to 7
 ncoming  equest to peer 8: From - 9 Duplicate  equest -  lready searched in this peer- with message id - Peer1
.Search1
```

peer1
peer 2
peer3
peer4
peer5
peer6
peer7
peer8
peer9

Terminal Help config.properties - P2PGnutella-master - Visual Studio Code

PROBLEMS 11 OUTPUT DEBUG CONSOLE TERMINAL

```
PS                          ,P2PGnutella-master\src> java gnutellaP2P/Peer
Enter the peerid
9
Enter the port
8009
Session for peer id: 9 started...
Enter the shared directory
                       \P2PGnutella-master\src\gnutellaP2P\peer9
Peer 9 acting as server on 127.0.0.1:8009
************ Main Menu ***************
1. Search File
2. Exit
*************************************
Select your choice
 ncoming  equest to peer 9: From - 6 Search locally and send request to neighbours for msg id- Peer1.Search1
 ocal Search: File Found in the current peer
 ncoming  equest to peer 9: From - 8 Duplicate  equest -  lready searched in this peer- with message id - Peer1
.Search1
 utgoing  equest from peer 9: Sending request to peerid.8 8008
 it uery: Send following result back to 6
--Found at Peer9 on localhost:8009
```

peer1
peer 2
peer3
peer4
peer5
peer6
peer7
peer8
peer9

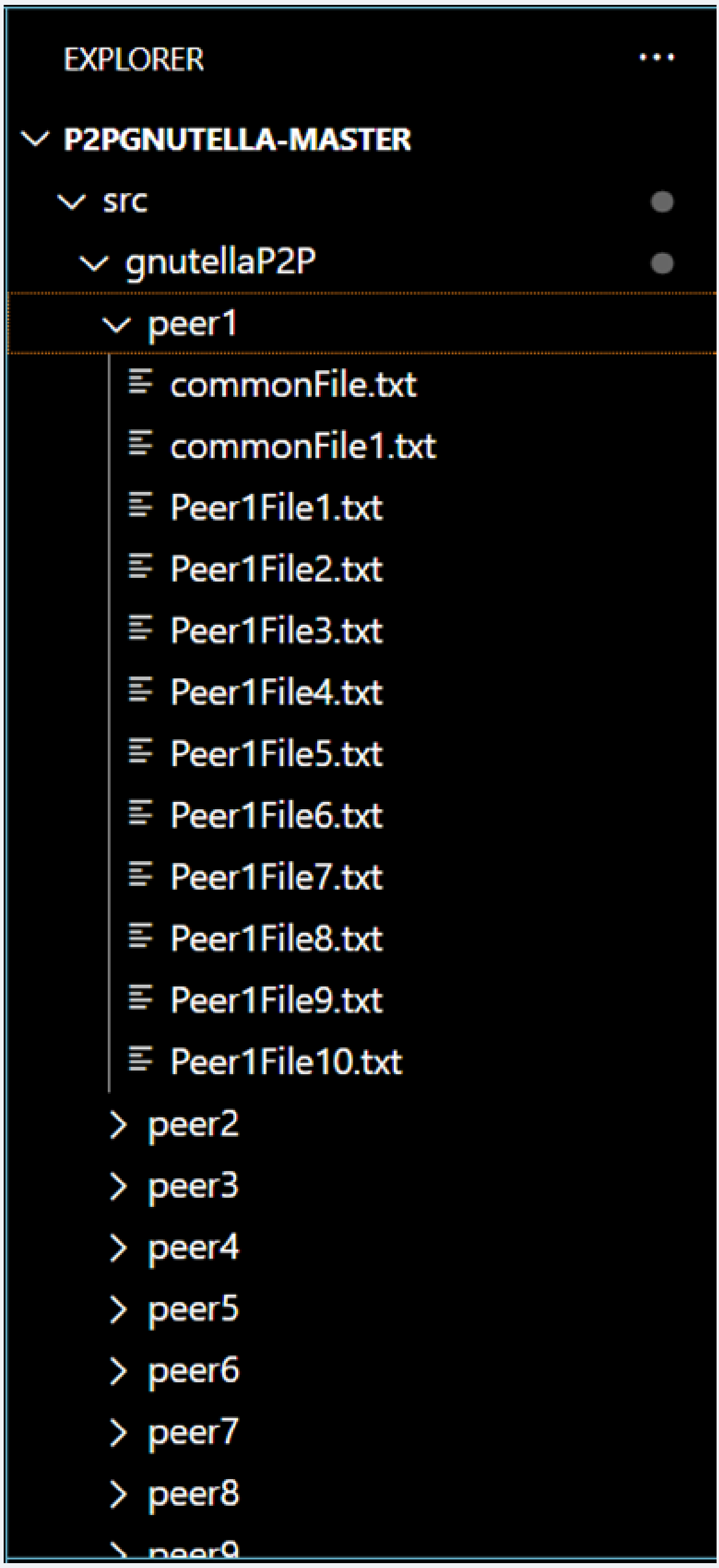
Terminal Help config.properties - P2PGnutella-master - Visual Studio Code

PROBLEMS 11 OUTPUT DEBUG CONSOLE TERMINAL

```
1. Search File
2. Exit
*************************************
Select your choice
1
Enter file name to search:
commonFile1.txt
Message id for search: Peer1.Search1
Sending request to peerid.2 8002
Sending request to peerid.4 8004
*** Search Paths ***
Search Path: 12369
Search Path: 145
Search Path: 1478
********************
commonFile1.txt File found in the nework at below peers
--Found at Peer3 , running on 127.0.0.1:8003
--Found at Peer9 , running on 127.0.0.1:8009
--Found at Peer5 , running on 127.0.0.1:8005
--Found at Peer7 , running on 127.0.0.1:8007
***Download Menu***
1.Download file
2.Exit
********************
Select operaion
1
Enter peer id to connect and download the file
3
Downloading from localhost:8003
"commonFile1.txt" downloaded to path                    P2PGnutella-master\src\gnutellaP2P\peer1
************ Main Menu ***************
1. Search File
2. Exit
*************************************
Select your choice
```

peer1
peer 2
peer3
peer4
peer5
peer6
peer7
peer8
peer9

Ln 7, Col 19 Spaces: 4 UTF-8 LF Java Properties




```
# Neighbor details
#3X3 Mesh
peerid.1.neighbors=peerid.2,peerid.4
peerid.2.neighbors=peerid.1,peerid.3,peerid.5
peerid.3.neighbors=peerid.2,peerid.6
peerid.4.neighbors=peerid.1,peerid.5,peerid.7
peerid.5.neighbors=peerid.2,peerid.4,peerid.6,peerid.8
peerid.6.neighbors=peerid.3,peerid.5,peerid.9
peerid.7.neighbors=peerid.4,peerid.8
peerid.8.neighbors=peerid.5,peerid.7,peerid.9
peerid.9.neighbors=peerid.6,peerid.8
```

*) RMI execution in two differents PCs :

*- Peer1:

C:\Windows\System32\cmd.exe - java gnutellaP2P/Peer

```
[redacted]\P2PGnutella-master\P2PGnutella-master\src>java gnutellaP2P/Peer
Enter the peerid
1
Enter the port
8001
Session for peer id: 1 started...
Enter the shared directory
[redacted]\shared
Peer 1 acting as server on 127.0.0.1:8001
************ Main Menu ***************
1. Search File
2. Exit
****************************************
Select your choice
1
Enter file name to search:
Peer2File2.txt
Message id for search: Peer1.Search1
Sending request to peerid.2 8002
*** Search Paths ***
Search Path: 12
******************
Peer2File2.txt File found in the nework at below peers
--Found at Peer2 , running on 127.0.0.1:8002
***Download Menu***
1.Download file
2.Exit
******************
Select operaion
1
Enter peer id to connect and download the file
2
Downloading from localhost:8002
```

*- Peer2:

PROBLEMS 11 OUTPUT DEBUG CONSOLE TERMINAL

```
PS [redacted]\P2PGnutella-master\src> java gnutellaP2P/Peer
Enter the peerid
2
Enter the port
8002
Session for peer id: 2 started...
Enter the shared directory
                    \P2PGnutella-master\src\gnutellaP2P\peer2
--Found at Peer2 on localhost:8002
```

*- config file:

```
1   # Peer details
2   peerid.1.ip=192.168.43.140
3   peerid.1.port=8001
4   peerid.2.ip=192.168.43.215
5   peerid.2.port=8002
6
7
8
9   peerid.1.neighbors=peerid.2
```

Performance Evaluation

***)System Performance:**

In this section we evaluate the performances of the system that is running the applications, which means we evaluate the performances of the computer that runs the application.
We measure this performance using a command line in CMD "Winsat formal"

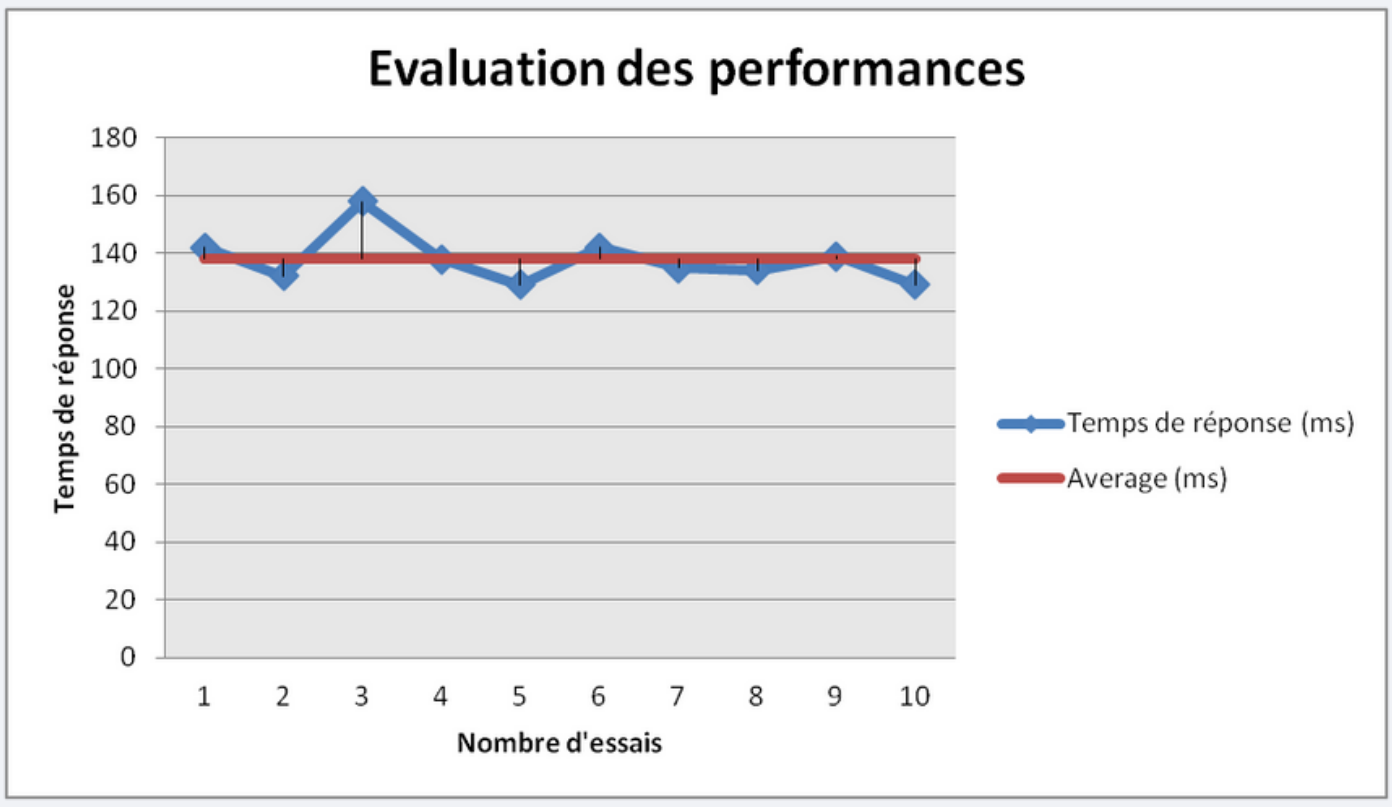
```
C:\Windows\system32>winsat formal
Outil d?évaluation du système Windows
> Exécution de l?évaluation formelle
> En cours d?exécution : Énumération de fonctions ''
> Heure d?exécution 00:00:00.00
> En cours d?exécution : Évaluation WinSAT Direct3D '-aname DWM -time 10 -fbc 10 -disp off -normalw 1 -alphaw 2 -width 1280 -height 1024 -winwidth
rtdelta 3 -nolock'
> Évaluation des performances des graphismes du bureau
> Heure d?exécution 00:00:20.78
> En cours d?exécution : Évaluation WinSAT Direct3D '-aname Batch -time 5 -fbc 10 -disp off -animate 10 -width 1280 -height 1024 -totalobj 300 -bat
delta 3 -texnobj 6(1)'
```

=> The final result of the performances :

```
> Compression LZW d?UC                          457.14 MB/s
> Chiffrement AES256 d?UC                       5376.39 MB/s
> Compression Vista d?UC                        1017.43 MB/s
> Hachage SHA1 d?UC                             2054.97 MB/s
> Compression LZW d?UC monoprocesseur           89.37 MB/s
> Chiffrement AES256 d?UC monoprocesseur        852.43 MB/s
> Compression Vista d?UC monoprocesseur         226.19 MB/s
> Hachage SHA1d?UC monoprocesseur               627.69 MB/s
> Performances de la mémoire                    25152.26 MB/s
> Performances de Direct3D Batch                42.00 F/s
> Performances de Direct3D Alpha Blend          42.00 F/s
> Performances Direct3D ALU                     42.00 F/s
> Performances de Direct3D Texture Load         42.00 F/s
> Performances de Direct3D Batch                42.00 F/s
> Performances de Direct3D Alpha Blend          42.00 F/s
> Performances Direct3D ALU                     42.00 F/s
> Performances de Direct3D Texture Load         42.00 F/s
> Performances géométriques Direct3D            42.00 F/s
> Performances géométriques Direct3D            42.00 F/s
> Performances de tampon constant Direct3D      42.00 F/s
> Débit de la mémoire vidéo                     5886.96 MB/s
> Temps de codage vidéo Dshow                   0.00000 s
> Temps de décodage vidéo Dshow                 0.00000 s
> Temps de décodage de Media Foundation         0.00000 s
> Disk  Sequential 64.0 Read                    303.06 MB/s          7.7
> Disk  Random 16.0 Read                        7.87 MB/s          5.2
> Durée d?exécution totale 00:00:30.55
```

***)Application Performance:**

In this section we evaluate the performances of the application where we do many trials and we measure the response time for each one and we calculate the average of it then we plot everything on a plot to have a visualisation about the performance of the application (of the Gnutella Peer-2-Peer network).



Try	Temps de réponse (ms)
1	142
2	132
3	158
4	138
5	129
6	142
7	135
8	134
9	139
10	129
Average (ms)	137,8

COLLAB with a colleague : Comparison of the Performance between Gnutella and Bittorrent P2P network