

1st approach execution manual

I. Introduction :

In this document we'll present the execution of the Gnutella Peer-2-Peer application (which is our first approach for this project work).

First of all, you need to open "n" number of command line. With "n" being the number of peers that you choose in your own topology.

The topology can be defined easily by modifying the "Topology.txt" file which you can find under the source code folder.

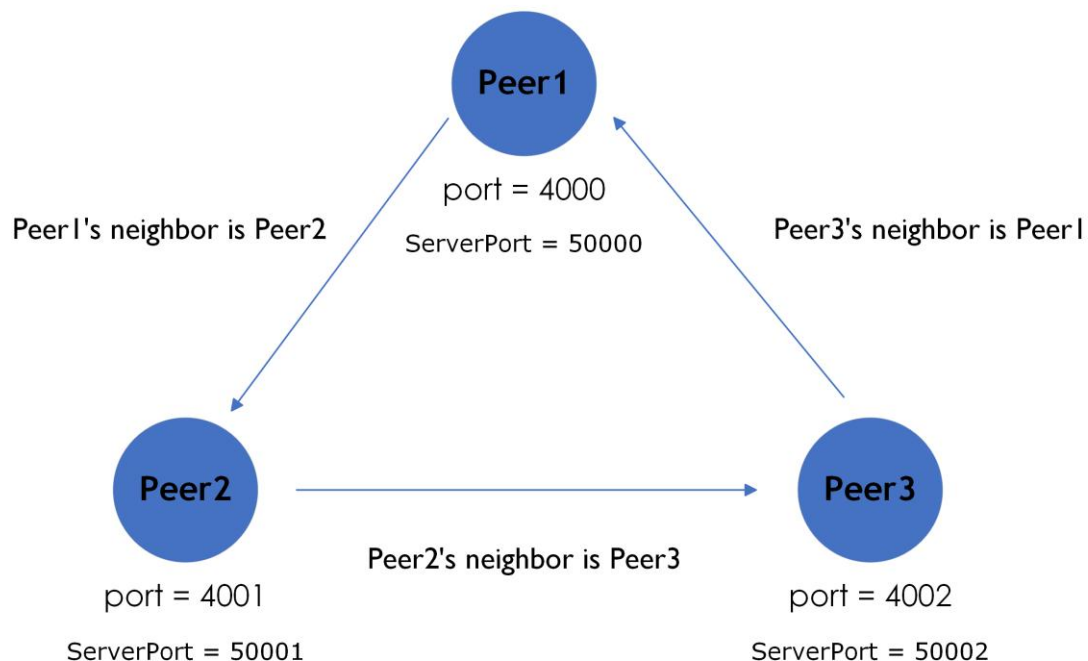
```
topology - Bloc-notes
Fichier Edition Format Affichage Aide
peer1.port = 4000
peer2.port = 4001
peer3.port = 4002

peer1.next=2
peer2.next=3
peer3.next=1

peer1.serverport=50000
peer2.serverport=50001
peer3.serverport=50002
```

« Topology.txt » file that we'll be working with for this example.

Peers Topology Diagram for this execution test

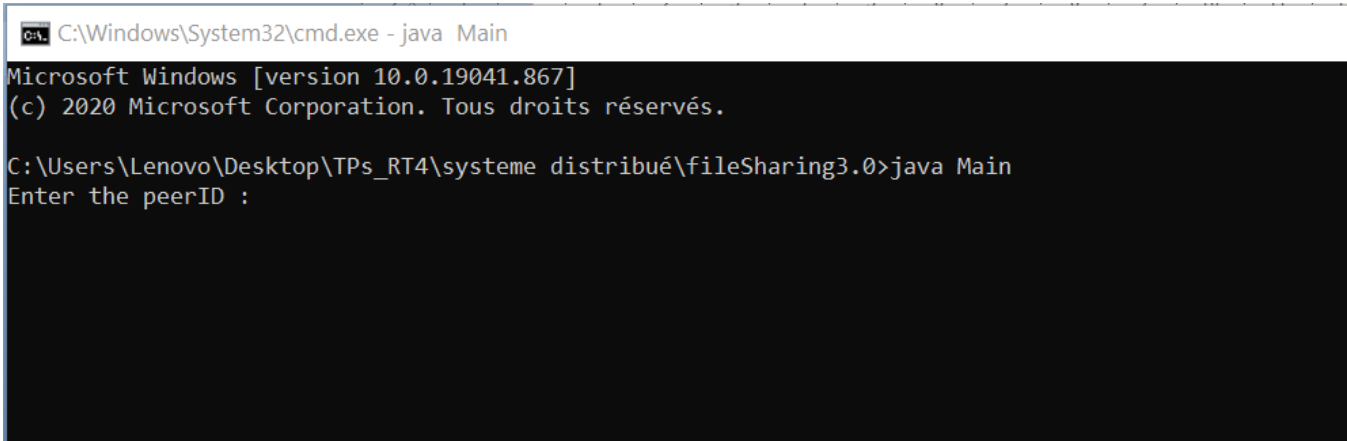


If you want to implement any topology, you can modify the parameters in the “Topology.txt” file according to the needs of your network.

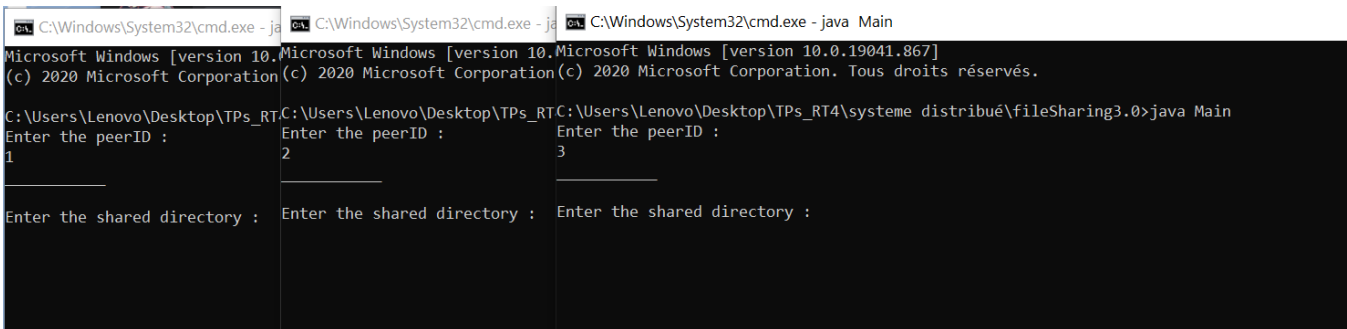
II. Execution process for this simple example :

First of all, since we have 3 peers, we’ll open 3 command line interfaces “CMD” from the path of the source code.

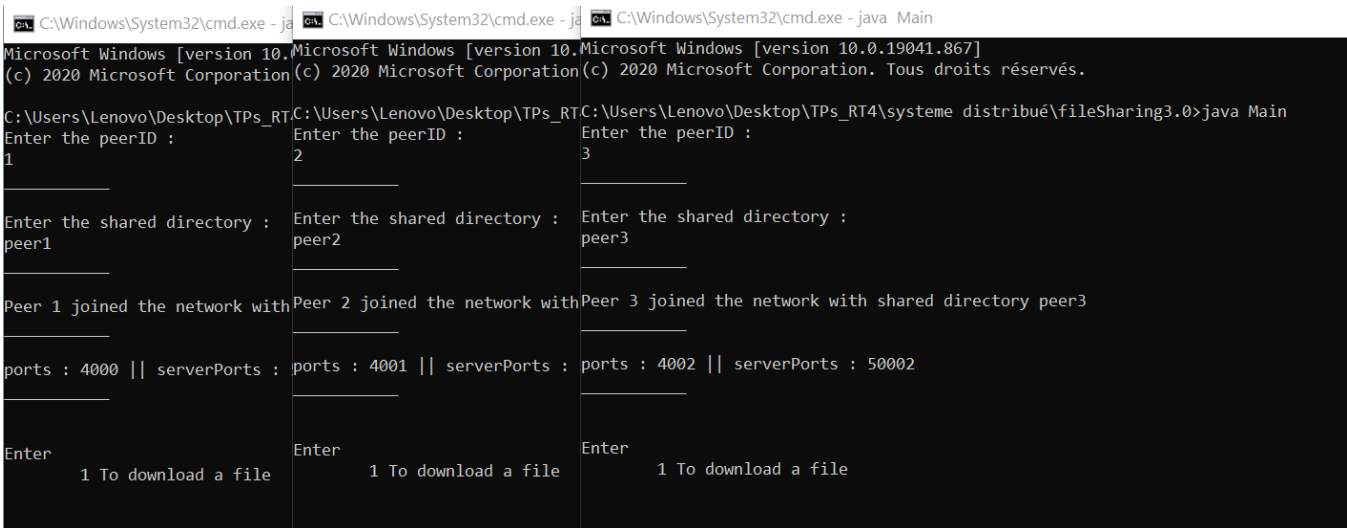
Then you run the code in each CMD interface using the command “java Main” which runs the “Main.java” file which contains the main class of the application.



The next step is to enter the peer ID in each CMD.




Then you enter the “shared directory” which means the directory that contains the files to be shared, or the locations where the downloaded files from other peers will be saved into.




After entering the directory, the application will print the respective port and server port for each peer.

Now each peer is a Server listening to the network, waiting for a “search” request to happen in the network. If you want to download a file you just need press “1”.


*** Peer 1's directory :**

TPs_RT4 > systeme distribué > fileSharing3.0 > peer1				
	Nom	Modifié le	Type	Taille
	DS-Ang-2022	06/12/2022 20:59	Document texte	20 Ko

*** Peer 2's directory :**

TPs_RT4 > systeme distribué > fileSharing3.0 > peer2				
	Nom	Modifié le	Type	Taille
	C-Math-Chap1	06/12/2022 20:59	Document texte	22 Ko

*** Peer 3's directory :**

TPs_RT4 > systeme distribué > fileSharing3.0 > peer3				
	Nom	Modifié le	Type	Taille
	TP-Program-Chap1	06/12/2022 21:00	Document texte	34 Ko

Now let's try to look for a file from peer1.

```
C:\Windows\System32\cmd.exe - java Main
Microsoft Windows [version 10.0.19041.867]
(c) 2020 Microsoft Corporation. Tous droits réservés.

C:\Users\Lenovo\Desktop\TPs_RT4\systeme distribué\fileSharing3.0>java Main
Enter the peerID :
1
_____

Enter the shared directory :
peer1
_____

Peer 1 joined the network with shared directory peer1
_____

ports : 4000 || serverPorts : 50000
_____

Enter
1 To download a file
1
Enter the file to be downloaded :
```

Now we need to enter the file name that we want to download. Let's try to look for the peer2's file “C-Math-Chap1”.

```
C:\Windows\System32\cmd.exe - java Main
Microsoft Windows [version 10.0.19041.867]
(c) 2020 Microsoft Corporation. Tous droits réservés.

C:\Users\Lenovo\Desktop\TPs_RT4\systeme distribué\fileSharing3.0>java Main
Enter the peerID :
1
_____
Enter the shared directory :
peer1
_____
Peer 1 joined the network with shared directory peer1
_____
ports : 4000 || serverPorts : 50000
_____
Enter
    1 To download a file
1
Enter the file to be downloaded :
C-Math-Chap1
```

Now we run the code.

***- The execution result in Peer1's CMD :**

```
C:\Windows\System32\cmd.exe - java Main

Peer 1 joined the network with shared directory peer1
_____
ports : 4000 || serverPorts : 50000
_____
Enter
    1 To download a file
1
Enter the file to be downloaded :
C-Math-Chap1.txt
The file to be downloaded is : C-Math-Chap1.txt
start time : 1670356871136
connection port : 4001 || neighbourPeer : 2
Connected to client at /127.0.0.1:1091 with peer 1
- Server thread for peer1
- Got request from 3
File : C-Math-Chap1.txt found
Peers that have the file : 1
- Sending to 2
File length: 22132 File name : C-Math-Chap1.txt
bytes transferred: 22132
- Sending file of 22132 bytes
end time : 1670356871313
C-Math-Chap1.txt file has been downloaded to your directory peer1
File: C-Math-Chap1.txt downloaded from Peer 1 to Peer peer1
```

In this execution, we print that the file to be downloaded is “C-Math-Chap1”, we print the start time of the process. Then we print the connection port and the neighbor peer of the client that will send the file. We also print the request that came from Peer3, since the search message is broadcasted to all the neighbors and we already defined that Peer3’s neighbor is Peer1 so he also sends a search message for Peer1, Peer1 then prints that the file isn’t found. Peer1 sends the same request that he received from Peer3 to Peer2 to continue the search, but then he prints an error, “No peer available”. This is due to the fact that we prevented the flooding of duplicated messages.

***- The execution result in Peer2's CMD :**

```
C:\Windows\System32\cmd.exe - java Main
peer2

Peer 2 joined the network with shared directory peer2

ports : 4001 || serverPorts : 50001

Enter
1 To download a file

Connected to client at /127.0.0.1:1088 with peer 2
- Server thread for peer2
- Got request from 1
File : C-Math-Chap1.txt found
Peers that have the file : 2
- Sending to 3
File length: 22132 File name : C-Math-Chap1.txt
bytes transferred: 22132
- Sending file of 22132 bytes
end time : 1670356871263
C-Math-Chap1.txt file has been downloaded to your directory peer1
Connected to client at /127.0.0.1:1092 with peer 2
- Server thread for peer2
- Got request from 1
Error : duplicate msg !
File: C-Math-Chap1.txt downloaded from Peer 2 to Peer peer1
```

In Peer2's CMD, we print that this peer got a request from Peer1. Peer2 does a local research for the requested file and then prints "C-Math-Chap1.txt found". We print that Peers that have the file are the Peer2. Then we send the request to peer3 and we reinstate the peer2 as a server. Peer2 gets another request from peer1, which is a request that peer1 got from peer3 (it is the request of the message that already reached all the peers in the network which means that the message was received by all the messages). We then print that it's an error! "Duplicate message", the message has already been sent so we stop all further flooding.

At that moment, the process of sending the file is starting. The file length is printed with its name. We print the result of local search of the file for the 2nd duplicate request, and then we send the file and print how many bytes were transferred.

At the end of the download, we print the ending time of the whole download process from the moment of the beginning of the search for the file until the final moment of the download.

Then we do another connection for the Peer2 as a server to continue listening for requests. At that moment we print that the Peer2 got a request from Peer1 which is the duplicated request, then we print the error.

Finally we print that the file "C-Math-Chap1.txt" has been downloaded from Peer2 to Peer1.

***- The execution result in Peer3's CMD :**

```
C:\Windows\System32\cmd.exe - java Main
Microsoft Windows [version 10.0.19041.867]
(c) 2020 Microsoft Corporation. Tous droits réservés.

C:\Users\Lenovo\Desktop\TPs_RT4\systeme distribué\fileSharing3.0>java Main
Enter the peerID :
3

Enter the shared directory :
peer3

Peer 3 joined the network with shared directory peer3

ports : 4002 || serverPorts : 50002

Enter
1 To download a file

Connected to client at /127.0.0.1:1089 with peer 3
- Server thread for peer3
- Got request from 2
File not found
- Sending to 1
Error! No peer available
```

Peer3 will connect as a server and keeps listening for requests. He'll get the request from Peer2, he does a local search and then prints that the file isn't found. Peer3 forwards the same request having the same message ID for its neighbor Peer1. And says that no peer is available because the message is duplicated.

***- Result of the download :**

TPs_RT4 > systeme distribué > fileSharing3.0 > peer1				
	Nom	Modifié le	Type	Taille
	C-Math-Chap1	06/12/2022 21:29	Document texte	22 Ko
its	DS-Ang-2022	06/12/2022 20:59	Document texte	20 Ko

***- Future perspectives :**

For this particular test, the Peer-2-Peer Gnutella network has been implemented locally on a single machine. The communication between distant peers on different machines but in the same local network is indeed possible. We just need to introduce some simple modifications on the code by replacing the "localhost" statements in each definition when creating sockets by the real IP addresses of each machine in its local network.

This can be done by adding another part in the "Topology.txt" file which will contain a predefined set of IP addresses for the machines that exist in the new local network having a certain topology. And then in the code, we just need to define the Property java object to extract all the information related to the IP addresses of these peers.

*- Performance evaluation :

The performance can be improved if we do restrict the flooding effect through the network which will reduce the response time between the peer that searches the file and the one that sends it.

