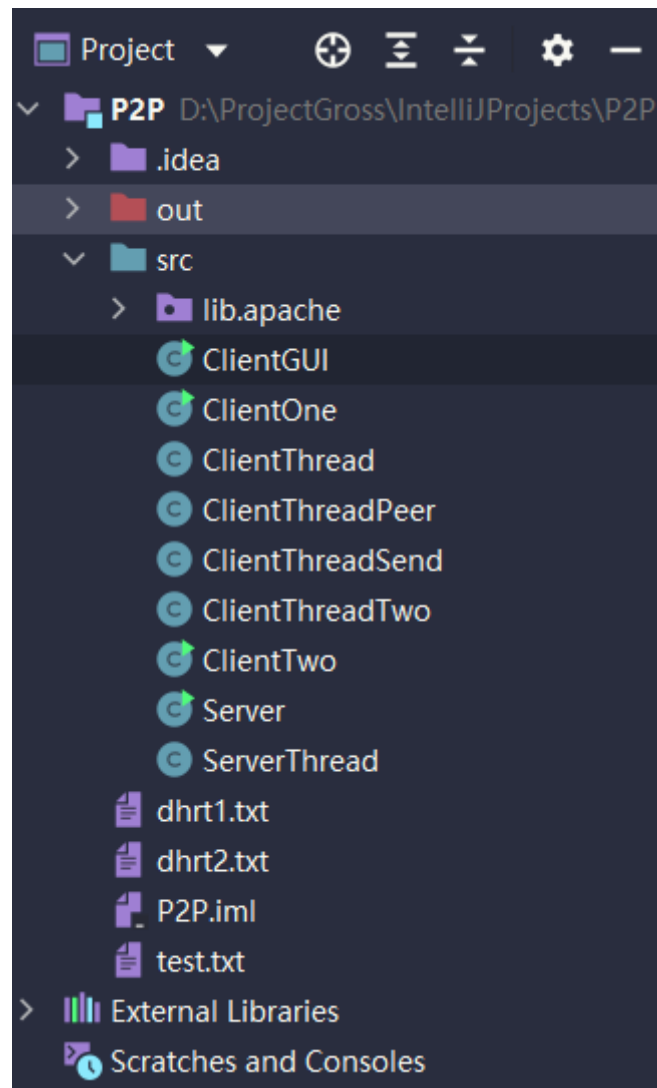


COMP3008J Distributed System Peer-to-Peer

Introduction

19206214 Tianli Ren

Project Structure:

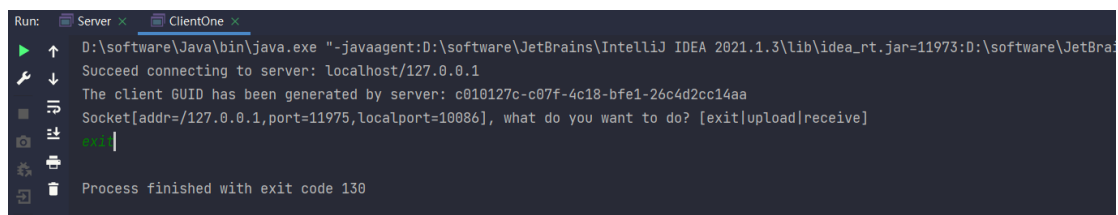


In this project, I implemented the backend of peer-to-peer file sharing system without GUI(The ClientGUI fails at last). The ClientOne and ClientTwo are two testing clients used by me. Each of them owns a ClientThread (One & Two) to handle detailed operations connected with

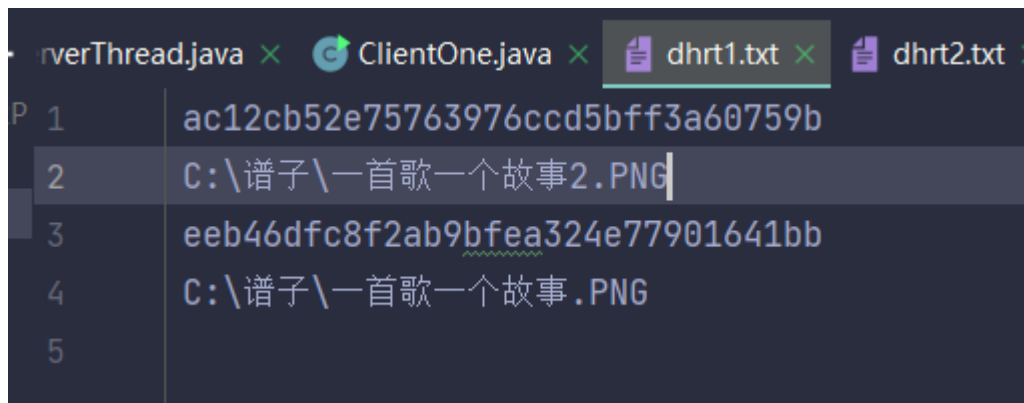
servers. The ClientThreadPeer is shared by two clients to deal with the operations with other peers or clients. The ClientThreadSend is used for sending files to different clients. The Server and ServerThread work together to maintain the system, as servers.

Design:

Following the guide of practical assignment, I implement the Universal Hashed Peer Table, Universal Hashed Resource Table, and Distributed Hashed Resource Table. Besides these 3 tables, I consider that the port number of each client is important as well so that different clients/peers can connect each other more quickly. To reach that, I design another table called portMap which applies String of GUID as key, and port integer as value. Each time when a client requests for a file, the server could transfer the client the port number to increase convenience. Furthermore, after exiting from the system, the DHRT of that client will be stored as an offline “txt” file so that next time when entering the system, previous uploading records can be again recorded in the servers.



```
Run: Server x ClientOne x
D:\software\Java\bin\java.exe "-javaagent:D:\software\JetBrains\IntelliJ IDEA 2021.1.3\lib\idea_rt.jar=11973:D:\software\JetBrains\IntelliJ IDEA 2021.1.3\bin" -Dfile.encoding=UTF-8
Succeed connecting to server: localhost/127.0.0.1
The client GUID has been generated by server: c010127c-c07f-4c18-bfe1-26c4d2cc14aa
Socket[addr=/127.0.0.1,port=11975,localport=10086], what do you want to do? [exit|upload|receive]
Process finished with exit code 130
```

A screenshot of a code editor with four tabs: 'ServerThread.java', 'ClientOne.java', 'dhrt1.txt', and 'dhrt2.txt'. The 'dhrt1.txt' tab is active and shows a list of peers. The list has five rows, each with a peer ID and a file path. The file paths are 'ac12cb52e75763976ccd5bff3a60759b', 'C:\谱子\一首歌一个故事2.PNG', 'eeb46dfc8f2ab9bfea324e77901641bb', 'C:\谱子\一首歌一个故事.PNG', and an empty field.

P	ID	File Path
1	ac12cb52e75763976ccd5bff3a60759b	
2	C:\谱子\一首歌一个故事2.PNG	
3	eeb46dfc8f2ab9bfea324e77901641bb	
4	C:\谱子\一首歌一个故事.PNG	
5		

Performance Analysis:

The performance is incredible. Once I transfer a file with around 2GB and that only costs a few seconds. Due to the reason of transferring within the localhost, the speed and performance are very great.

Pros:

This system is very light and convenient. It follows the concepts of peer-to-peer and distributed hash table. Different clients can simultaneously share and download files on the server. Each peer can act as a client or a server. If you transfer files in localhost, sometimes the speed is even faster than replication in the computer. The system supports clients to upload files, download files, and exit the system.

Cons:

There are two main disadvantages in my system. My system finally did not own a Graphical User Interface, which decreases the usability of my

system and users have to operate the system with commands. In additional, small bugs still exist. If we directly shut down the running of a client, the server would be unable to deal with that and exception happens.

Summary:

This project is not perfect, but I implement the backend of p2p system and all related theories are included. From this trip, I have a closer relationship with socket, java, thread and some other techniques. I will continue studying more knowledge on Distributed System in the future.