CS 550 Programming Assignment2
P2P File Sharing System
Yue Wu A20359521
Xin Liu A20353208
Group# 17

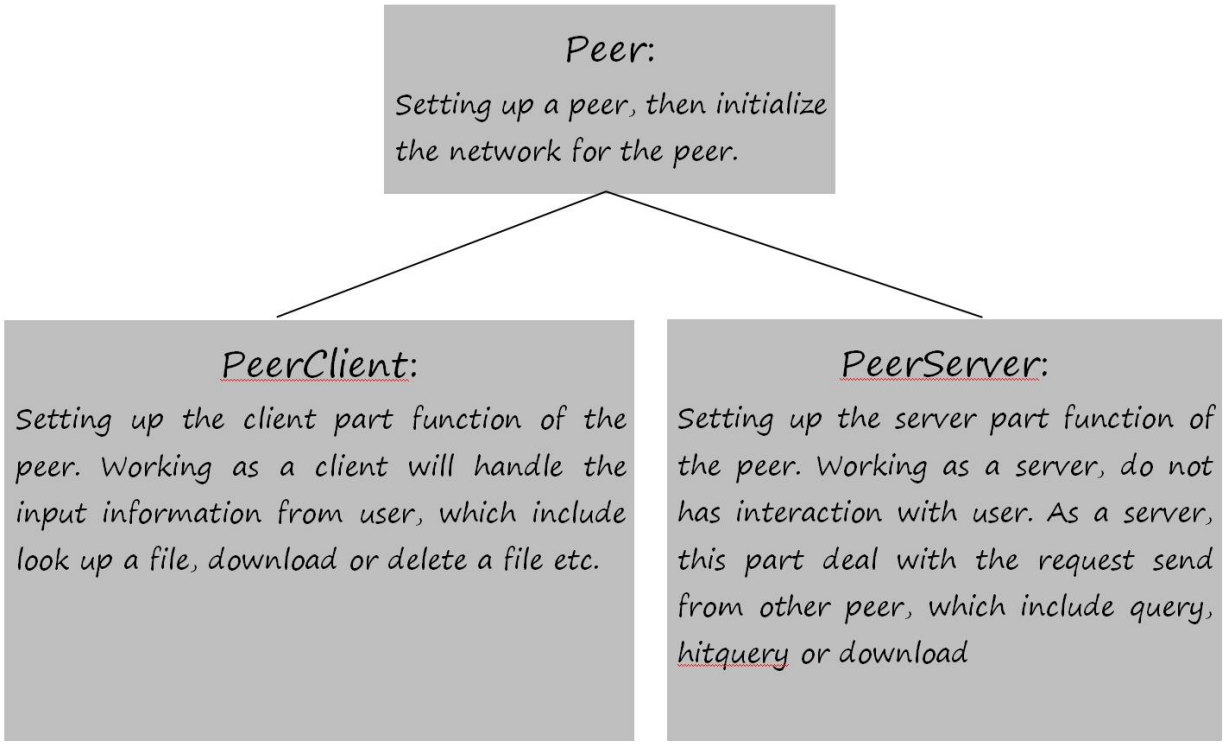
Design Document

I.Introduction

In this project, we need design a Gnutella-style P2P file sharing system, which is a famous distributed P2P system. So, as the same as the last project, we use java for programming language and socket api to implement multiply-clients transfer files.

II.Program Design

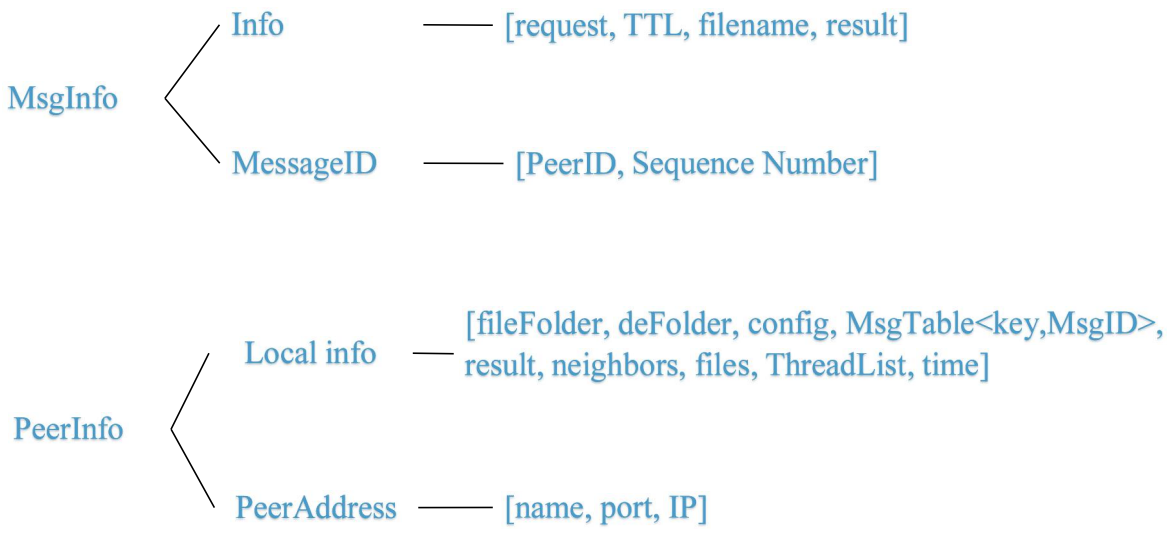
As a Gnutella-style P2P File Sharing System, since there are no index server in this system, all functions should be implemented on the peer. As a peer, it can send request, also it can deal with request sent by other peer. Then we divided a peer into two part, peer as a client and peer as a server. Following is the design structure of this program.



III.Overall Program

Data structure:

In our design, we use 4 data structures to store our data, they are **MessageID**, **MsgInfo**, **PeerInfo**, **PeerAddress**. The relationship between them is shown in the following figure.



In this program, each peer works independently and has both client and server function.

Peer: the first thing a peer should do is read configure file which contains all peer IP, port and topologies information.

PeerClient: As a client, the peer should share its own file for other peers to search and download. Then, it should able to send query messages to the neighbors and transmit hitquery messages back. In the end, when it choose to download file, it need set up a connection to object peer and locate object file to download.

Mainly functions:

1) Lookup files: This part is just like the last project, we have file_folder to save local files and dl_folder to save download files. The files will automatic register in file_folder and store all file's name in an arraryList. We put lookup function in Connection.java, we will explain later.

2) Delete files:The delete function also we called unregister will judge file exist or not, if exist, remove file from register list.

3) Download files: We use the same download function as the last project to achieve download files from other peers.

PeerServer: As a server, it can listen requests from other peer and provide download files for other peers. In the function we called 'handler', server will judge the request type which include query, hitquery and download. Then respond to other peers with corresponding action.

Connection: This part we used for saving all function what we could use in this project. The most important functions are Query and HitQuery.

Mainly functions:

1) Query: there are two part of query: sendQuerymessage and Queryhandler. The first part is responsible for sending message to other peer; and the second part is responsible for judge statement to see if the upstream peer is in the neighbor list of current peer, if it is, current peer will not send query message.

2) HitQuery: just like query, it also be divided to two parts: sendHitQuerymessage and HitQueryHandler. The first part is responsible for sending message back to the peer who send query to it. The second part is responsible for lookup object files and send a hitquery message back to original peer.

IV. Improvements and Extensions

1. We can make client update the files periodically to the server by using a Watch mechanism which means create a loop to keep watching local client files, once the client files have some changes (like modify, delete, add) in object file folder, the clientServer can receive the request.
2. In our program, we can transfer .txt files only. As for improvement, we can add a function which can convert different types of file to byte data stream, then saving these data in indexing server, . The client and server transfer byte data stream, and convert byte data stream back to original file at received port(client). In this way, we can using this system transfer multiple file types such as .jpg, .pdf, .zip, etc.
3. We can implement a multithread mechanism by using thread pool for efficient scheduling of multithread, whcih may able to improve the efficiency to connect to different peerServer at the same time.