Design Document for Task #1

Central Index Server

Central Index Server indexes the contents of the entire peer that register with it. It keeps list of file names present in the directory of the registered peer and provides the functionality to search for file name present to others registered peer of server. If the file name is found in any connected peer it will response peer with the peer IP address and port number that have those files.

Central Index Server API:

Central index server is written by keeping in mind of scalability that we can increase the number of peers connected to server. To make server more scalable I have created an API for the server:

1. This API can be exposed to any application for accessing required functionality of server without worry of internal functionality.

2. We can change our server implementation any time on need basis without changing the API that is for client side.

3. As API is written for generic use so any developer can change the internal functionality without worry.

Managing Peer Connections

Central index server application accepts connection at any given port given there is no process running on that port by default server accepts connection at port no 12346. Every time any application makes connection to server listening port. Connection Handler gives that connection to Connection Manager instance. Connection Manager instance takes a Runnable instance of Peer Connection object for every connection made to the server. Every connection is handled by a new instance of Peer Connection, which is Runnable.

Adding a Peer Connection to Server

Once the connection is established between the server and the peer. Peer connected to server by providing Peer port number, Peer File transfer port and Peer File Names.

Peer Application

Peer Application is designed in such a way that it act as client for getting file names from central index server as well as work as server to other peer for downloading data. It's a multithreaded application that has one thread acting as client for central index server, another thread working as server for accepting the connection from other peers to facilitate file download functionality.

Peer Application acting as client

1. Making Connection with Server: When we start the peer application it creates a connection to central index server using the default IP address 127.0.0.1(localhost) and port no 12346. It can be configured to other IP Address and port no

2. Registering to Application to Server: Once Peer Application gets response form central index server it ask the client to register the peer to Server by giving the peer a name and providing a directory from where peer application can share it's available file list as well download the available file form other peers.

3. File Look up in server: After the peer is registered then application ask to search any file in server and server give response with matching filename as well as location of peer port no.

4. If the user choose to terminate the peer . Then the peer and its files which have been registered will be deleted from the indexed server so the other peers cannot download any file from this peer .

Peer Application working as File download Server

Every peer client has an additional thread working as server hat accepts request from other peer and allows those connected peer to download the file from it directory. Port no of every peer application can be configured using properties files. Peer file download server instance can handle multiple peers those acting as client to the file download server.

IMPROVMENTS ON THE CODE

My implementation has some errors :

1. If two peers has the same name of file the errors will be fired so the improvement is that user will choose which port to download the file from it
2. My second error is when download file it will download the text file only so I improve it by download binary file so it can download images and text files etc

Language and Data structure Used

Programming Language: python

Network Connection: Remote Procedure Call

Connection Management: Using RPYC package

Data Structure:

1. Dictionary: For keeping connection successful connection of peer. Where key is combination of port number of Peer and File transfer port.