Design Document

1. Introduction

Reason for this record is to give outline how this - A Simple Decentralized Peer to Peer File Sharing System - has been designed and implemented.

2. Architecture Design

System has been implemented in JAVA using Sockets and Threads thus, Architecture design is aligned to them.

Decentralized Indexing server: - Decentralized Indexing server has one main class which creates server socket to listen peers' requests on particular port. Server serves requests by creating a thread so it can serve multiple peers simultaneously in individual threads. Server keeps track of all registering peers' file information in a HashTable. Information which server gets from peers are their IP address, filename, file path and port on which they listen to other peers. When a peer searches for file, server checks into HashTable and gives list of other peers which have the file.

Peer as client: - Each peer registers its files on particular server based on computed hash value. When a client is initialized, it reads configuration and stores information of all other peers into the array. Server to which client will be connected is decided by hash function whose input is file name sent by client and output is index location of array where information of server is stored. Client then connects to server and performs register and search operations accordingly.

Peer as server: - When peer acts as a server it listens to other peers requests and serves their download requests. Server opens a socket and accepts other clients' connection requests in separate threads. By multithreading implementation server can handle multiple requests from clients.

Dual Client/Server behavior of peer: - A peer acts as server and client at the same time. This has been achieved by creating two threads. One thread enables peer act as client and client can connect to server and perform get, put and delete operation. Whereas another thread creates a server socket to listen other peers on particular port.

Configuration file:- Configuration file is one property file 'config.txt' whose copy is resided with each peer. Configuration file contains information of all servers such as IP address and port address on which they are listening to other clients. When client is initialized it reads configuration file and stores information of servers in array and uses it for later connection.

3. Tradeoffs/Limitations

- Configuration file must be kept up and refreshed in light of the machine on which program is running.
- Hash function is received from outside sources, accordingly crash relies upon how hash work is executed.

4. Improvements and Extensions

Apart from notable improvements GUI, Some other possible improvements are as follows.

1. Implementation of scheduler to update replica regularly.

   A scheduler can be created which updates replica regularly based on fault tolerance requirement. For example, a scheduler can be created which replicates server's data into its replica every 15 minutes. Replication time can be decided based on system's requirement.

2. Feature to allow peer to hide its specific files from registration and downloading.

   Example:-

   Peer may be given list of files and asked to select files which it wants to registers.

Select files which you want to register.	
	File10.txt
	File20.txt
	File30.txt

3. Feature to allow peer server to block particular peers from downloading files from it.

   Example:-Peer may be given list of peers to select peer which it wants to block.

Select peers which you want to block.	
	Peer1
	Peer2
	Peer3

4. Feature to receive notification when source file is updated, feature to auto update file from source file, feature to schedule download or auto download file whenever file becomes available.