A20354704 Ashwin Chati

Design Document

Simple Decentralized Peer to Peer File Sharing System

Problem Statement:

To develop a decentralized file sharing system through the integration of centralized file sharing system and distributed hash table.

The system should have two components:

- **A decentralized indexing server:** The indexing server should be able to perform below operations.
  - **Registry** : This function is used to register the information about files and the peers on which those files exists. To implement this functionality a hashmap is created. Hashmap holds key-value pairs . Here key is the file name and value is the address of the file that is the peer information on which the file exists. When peer is connected and wants to register a new file then entry map is created.
  - **Search** : This functionality is used to process search request given by the peer. When server received request to search a file, it searches the hashmap with key as a filename. If the file entry is present in the hashmap then it sends the peer details to the peer.

- **Peer:** A peer is both a client and a server. As a client, then user specifies a file name with the indexing server using "lookup". The indexing server returns a list of all other peers that hold the file. The user can pick one such peer and the client then connects to this peer and downloads the file. As a server, the peer waits for requests from other peers and sends the requested file when receiving a request. Minimally, the peer server should provide the following interface to the peer client:
  - **obtain(filename):** invoked by a peer to download a file from another peer.

System Architecture:

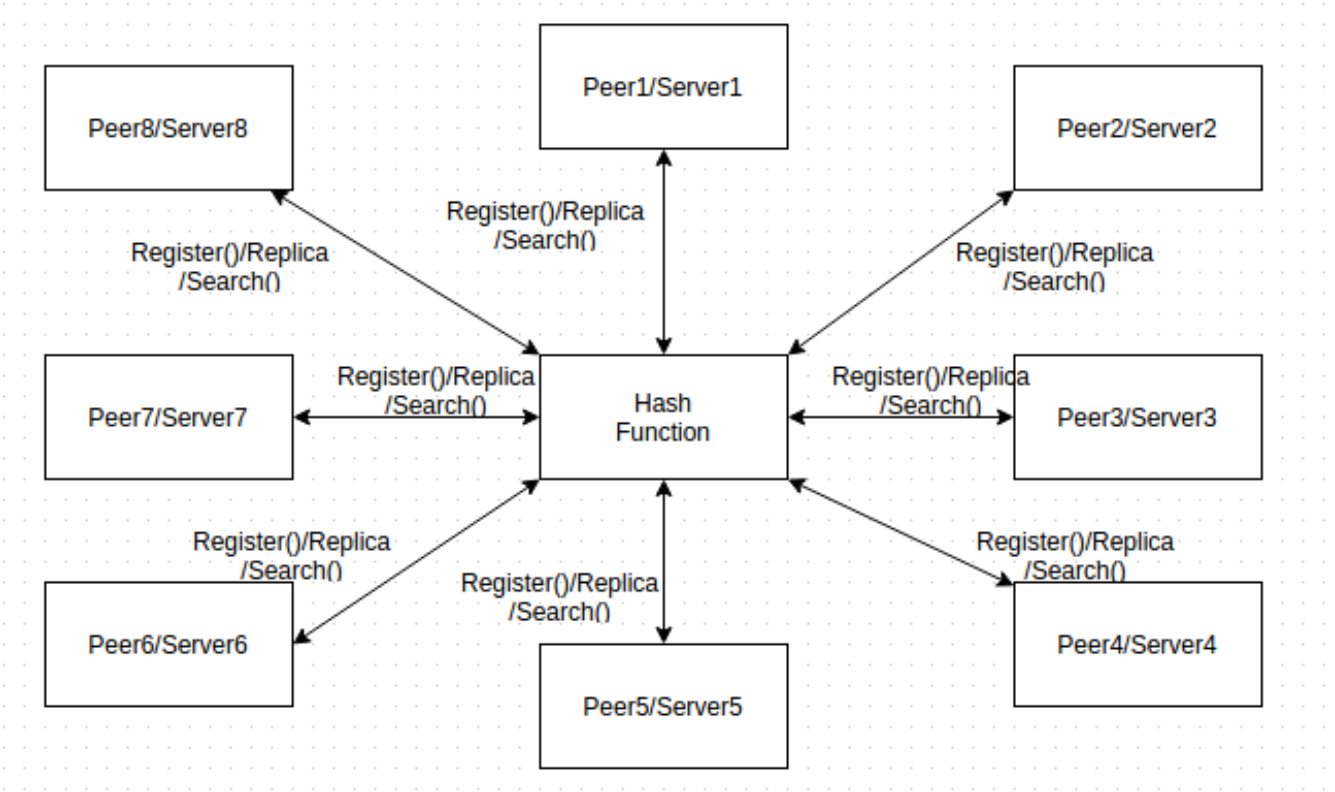
The high level architecture is shown in the given figure.

Each peer should be both server and a client.

As a server: It should accept requests from other peers to register files, or search file and send the response to the peer.

As a client: It should provide interfaces through which users can request to register, search and download file.

Hash Function: A hash function maps keys to small integers (buckets). Hash function is implemented to decide which peer to connect to for performing the operations.



A20354704 Ashwin Chati

Functionality:

Peer can perform below operations:
1. Register Files
2. Search Files

1. Register Files:

As a client: It takes all files from a folder specified by the user in the config file and calculates hash value for all files to determine on which server the files to be stored. It then sends the request to server to register files. Here key is filename and value is IP and port of the peer.

As a server: It takes key and value pair sent by the other peer and inserts the values into the hashmap. If the key and value is already present in the hashmap then it does not get stored. If key is present but value is different then the the value is appended into previous value.

2. Search File:

As a client: It takes input in the form of key for which the value needs to be find. It then decides which peer to connect using hash function based on the key and sends the key to that particular peer. If file is found then the option to download is given to the user and if yes then file is downloaded form the peer.

As a Server: It takes key sent by other peer for which the value needs to be searched. Then the value for that particular key value is searched and if it is present then the response is sent back to the requesting peer.

Other Functionalities:

1. Replication: To avoid the data loss in case the peer goes down, we have replicated the meta data and files from the server into other peers. The number of replicas to be made is decided by the replication factor specified in the configuration file.

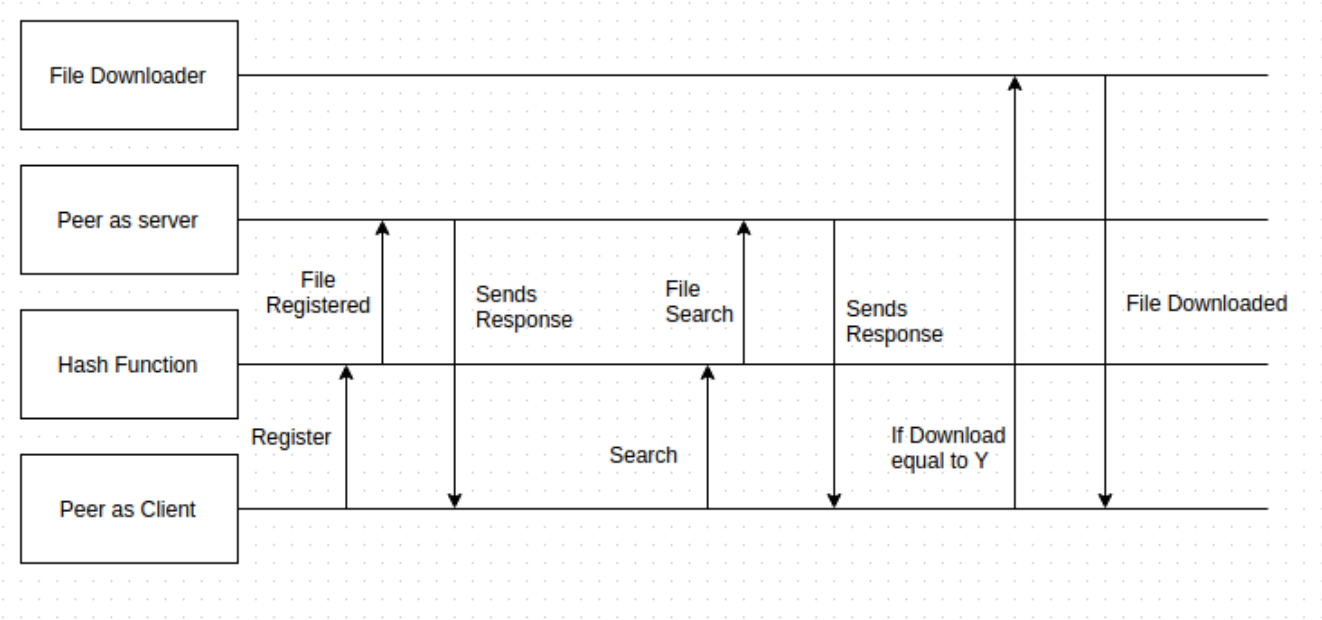
Register: If replication factor is 0 then no replicas are made. If replication factor is grater than 1 then peer as a client will add number of replica of Hash Table and file replicas on other peers.

Search: If replication factor is greater then 0 then in case of search requests if the main peer is down then the next available peer who is having the replica of that file is contacted and search operation is performed. If search is successful then the file is downloaded.

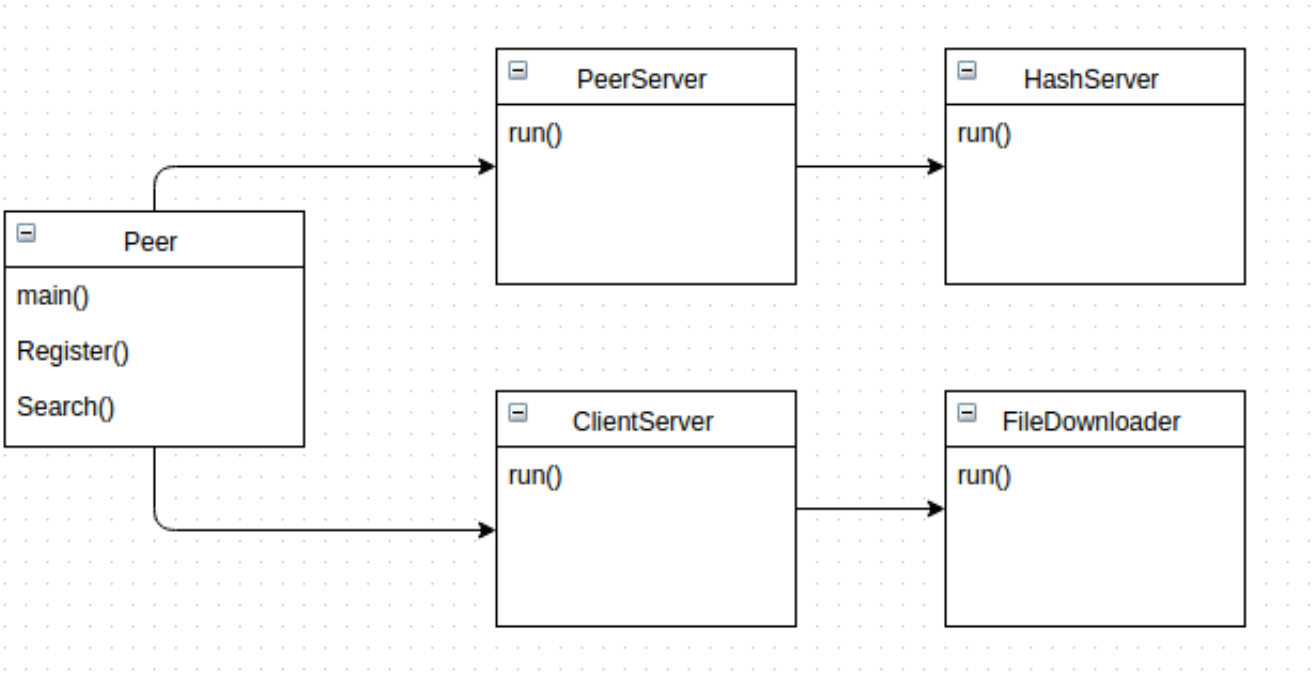
2. Fault tolerance:

The system supports fault tolerance. To test this first establish the connection between all the

peers and then kill any peer. To establish the connection you should perform random register and search requests.



Class Diagram:



Peer: This is the main peer class. When it up and running it keeps listening for the peer request to connect. When the peer gets connected it generates a new thread and keeps listening for other peers.

PeerServer: This class is used to process the threads initiated by the main Peer class. Register and search requests are processed by this class.

HashServer: This class is used to process the thread form the peers for register and search.

ClientServer: This class is used to process the threads from the peers to download file.

FileDownloader: This class processes the requests for downloading the file.

System Requirement:

1. Operating System: Windows/Linux
2. Softwares: a. java
b. Ant

Future Scope:

The future scope is to implement the system at a larger scale.