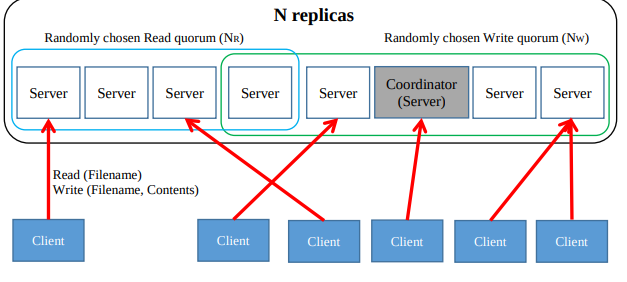
Design Document Anubhav Panda (panda047) ,Sanjit Dash(dash0030)  
Design of The PA3



OverView  
In this programming assignment, we have implemented a simple distributed file system in which multiple clients can share files together. In this file system, the files will be replicated to several servers for increased performance and availability.  
The Design of the PA3 can be divided into following sections.

Client .java

SeparteClient.java

coordinatorNode.thrift (Thrift server responsible for coordinator node communication)

coordinatorNodeServerHandler.Java

coordinatorNodeServer.Java

serverNode.thrift(Thrift server responsible for Server node communication)

serverNodeServerHandler.Java

serverNodeServer.Java

**Client:**  The client reads takes the input from the user and based on the input it decides to generate the number of SeparteClients.

**SeparteClient :**  SeparteClient reads takes the input from the user and based on the input it connects to the corresponding machine and the operation that thread will perform(read/write) and the number of time that corresponding thread will perform it.

**Thrift files used for CoordinatorNode communication**

The following are the thrift files that are used for client CoordinatorNode communication .It contains the following methods

coordinatorNode.thrift

bool Ping() check your calls are reaching to the coordinator or not.

String Join(String Ip,String Port) :This method is called from the Server nodes who are interested in joining the Quorum it takes two parameters ie

Ip:we are using the host name of the interested Server Node.

Port:The Port in which will be used for its communication.

string assembleQuorom(string ip,string port,i32 task,string FileName):This method is used to return the quorum based on the number of requests in the read /write operations. It uses the following varriables

ConcurrentHashMap<String,Integer>readRequests :it is used to maintain the no of read requests per file.

ConcurrentMap<String, ConcurrentLinkedQueue<Integer>> map :it is used to maintain a queue per file.

Here the assumption is the filenames are unique.It supports parallel write operations of Different files and parallel read operations as well.But if you are planning to write in the file in which read operations are getting performed it waits for all the read operations to complete.

public void finished(String ip,String port,int task,int id,String FileName):

This method is called from the server once it has finished the read /write operation.If it is write operation it will delete the entry(Task id ) from the queue.

If it is a read operation it decrements its counter in the corresponding hash table.

public void synch(): This method is called for the synchronization purpose (Eventual Consistency)in the file system. It fetches all files details from the servers and gets the maximum version of the files and it calls each server to update the the version number of the corresponding file. It is a background method and it is called in each 10 seconds(Configurable).It waits for a file if it finds that the corresponding file is in the writing queue .It starts the sync process once the write has been done .

**coordinatorNodeServerHandler.Java**

This file implements the the Above mentioned Thrift Files and apart from that it has some of its local methods. The below functions are implemented for local use.

private List<String> getUniqueFiles(List<Map<String, Integer>> Maps) :This method is used to find uniquefile names across all the systems and return it.  
  
**coordinatorNodeServer.Java**

It contains The following methods .

public static void main(String [] args) :The main method where Implementations Starts.It initiates the handler

public static void simple(SuperNode.Processor processor) :Its a simple server which is running as tt threaded server

**Thrift files used for Server Node communication**

The following are the thrift files that are used for client ServerNode communication .It contains the following methods

bool ping():check your calls are reaching to the server or not.

string write(string fileName,string contents): This method is used to get the quoram nodes ip and and the request id for the corresponding process ( write)and based on that it calls the write method of corresponding nodes for write operations.  
Parameters :  
filename : The file that needs to written /updated in the system  
contents: The contents of the file that needs to be written   
  
string writeAll(string filename,string contents):This method is used to write the contents of a file from the local system.

Parameters :  
filename : The file that needs to written /updated in the system  
contents: The contents of the file that needs to be written   
  
string read(string filename):This method is used to get the quoram nodes ip and and the request id for the corresponding process (ie read )and based on that it calls the read method of corresponding nodes(servers) for read operations.  
  
Parameters:

Filename :name of the file that needs to be read.

string readAll(string filename):This method is used to read the contents of a file from the local system based on the file value.  
String fileName: The name of file that needs to be read   
  
i32 getversion(String filename):This method is used to return the version number of a file of the local system.  
  
String fileName: The name of file whose version needs to be returned .  
  
map<string, i32> getMap(): This method is used to return the version map (which contains all the filenames and its corresponding versions)  
  
**serverNodeServerHandler:**This file implements the the Above mentioned Thrift Files and apart from that it has some of its local methods. The below functions are implemented for local use.  
  
  
ReadConfig():This file is used to read the config files and set the value of coordinator ip ,port ,filepath based on the config section.  
  
public void join(): This method is used to call the Coordinatorip to join the coordinator so that the coordinator can return its ip address while returning the quoram.  
  
  
**Features of the Implementations:  
Concurrent read operations can be performed  
Concurrent Write operations for Different Files can be performed.  
Synch is performed at a certain config Driven interval.**