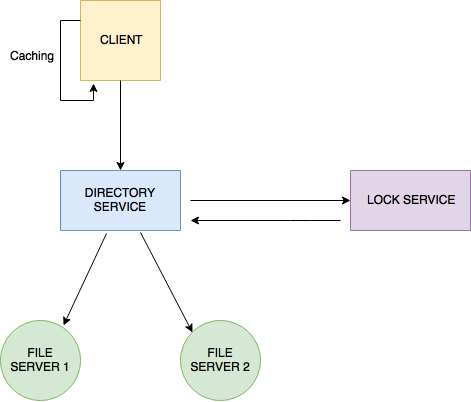
Name: Ujjwal Arora

Student id :17311879

Email id : arorau@tcd.ie

Dependencies: Python 2.7

**Distributed Systems**

****

**Introduction:**

A distributed System is one in which independent, self-sufficient – often autonomous or heterogeneous – spatially-separated components must use a common interconnect to exchange information and coordinate actions, and allow the whole to appear to the user as one single coherent system.

In order for separate and independent processes to make sense of the communication that they are engaging in, then they must follow a set of rules (**protocols**) of engagement and exchange.

**Sockets:**

Low level, logical level IPC.

In unix they can be viewed as input/output streams.

Client-side sockets are often short-lived, e.g. in a web browser one is created to send a request and receive a response, and then discarded:

⁃  Create a socket for the given transport (e.g. TCP) and internet protocol (e.g. IPv4).

⁃  Connect to the server using the port linked to the protocol (e.g. HTTP is 80).

⁃  Then the client can send the request. Through the same socket a response is waited for and then received in chunks.

⁃  Socket is discarded.

⁃  Gopher is a great example.

Server-Side Sockets:

⁃  Acts more like a dispatcher.

⁃  It listens for connections on the server socket and ports them.

⁃  When it gets a new connection it gets a new socket in response.

Server-Side Socket set up:

⁃  Create a server socket for the given transport (e.g. TCP) and internet protocol (e.g. IPv4).

⁃  Set the socket options.

⁃  Bind the server socket to a host address and a port.

⁃  Start listening to connections on the server socket and set the max number that could be left waiting.

I have implemented :

1. Directory Service

2. Lock service

3. Replication

4. Caching

5. TCP server

Brief description of the services implemented is below.

**Directory service:** Directory Server provides a central repository for storing and managing information. Almost any kind of information can be stored, from identity profiles and access privileges to information about application and network resources, printers, network devices and manufactured parts. Information stored in Directory Server can be used for the authentication and authorization of users to enable secure access to enterprise and Internet services and applications. Directory Server is extensible, can be integrated with existing systems, and enables the consolidation of employee, customer, supplier, and partner information.

It loops through all file servers and returns/updates/adds files to each of the file servers which have been specified by the client.

Steps included in the directory service implementation:

1. overriding request processing function

2. requesting file details from directory

3. adding the folder to directory listing if writing

4. checking if folder exists in directory listing

5. if not writing then assigning folder to random server

6. returning the directory id and location

7. checking if the file is in directory

8. returning the file not found if file\_id key not in files directory.

**Lock Server** : When a client needs to access any file or anything from a server it sends a request to the server and ask if that file is available, if the file is available then the server gives the client the permission to work on that file. But during that situation if any other client comes asking for the same file then it will get the message “the file is locked”. So, when a file is being used by the client, it is locked till the time client finishes the work on the client.

Steps included in the implementation:

1.overriding the request processing function

2.lock request

3.acquiring locks mutex

4.returning failure if file is locked and lock owner is different client

5.otherwise okay to lock file for client and returning success

6.unlock request

7.acquiring locks mutex

8.unlocking and return success if file is locked and owned by client

9.otherwise returning failure if file is not in array

10.otherwise returning file locked by another client

**Replication:** Replication is the method of creating several copies or replicas of a specific resource which has to be used by the client again and again. The client can choose the copy from the nearest server and work on it. Replication is done in order to reduce the load on the server, access latency and the network .congestion.

**Caching Service:** Caching in the distributed file systems are employed to avoid the network accessesat different clients and also to compensate the differential speed between main memory and disks at file servers. It also contributes to the scalability and reliability of the distributed systems as data can be remotely cached on the client node , an effective caching strategy will seek to maintain consistent copies of the file server data. There are three Locations where caching solution implemented :

·         Maintaining a copy in memory on client

·         Maintaining a copy in disk on client

·         Maintaining in memory of server

Steps followed for the implementation:-

1.importing thread

2. Setting the socket.

3. Waiting for connection.

4. Accepting connection

5. Starting thread for connection.

6. Closing the connection.

7. todo parse to return file object

8. returning file TODO

9. New socket.

10. Getting a responceof a file name and path

11. saving the file into my database

12. saving to cachedfiles list.