A PROJECT REPORT ON

EFFICIENT DATA COMMUNICATION AND STORAGE USING SYNCHRONIZED SERVERS WITH LOAD DISTRIBUTION FACILITY

SUBMITTED TO THE SAVITRIBAI PHULE PUNE UNIVERSITY, PUNE
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE AWARD OF THE DEGREE

BACHELOR OF ENGINEERING

In

COMPUTER ENGINEERING

Of

SAVITRIBAI PHULE PUNE UNIVERSITY

By

MOHIT PATIL	**405C024**	**EXAM SEAT NO: B150234339**
AMIT RAJ	**405C002**	**EXAM SEAT NO: B150234215**
SMIT MAHAJAN	**405C017**	**EXAM SEAT NO: B150234309**
NEHA GOUNDADKAR	**405C009**	**EXAM SEAT NO: B150234263**

PROJECT GROUP NO. 36

Under the guidance of

PROF. APARNA KALASKAR

Sinhgad Institutes

DEPARTMENT OF COMPUTER ENGINEERING
SINHGAD COLLEGE OF ENGINEERING, PUNE-41
Accredited by NAAC
2020-21

Sinhgad Technical Education Society,
Sinhgad College of Engineering , Pune-41
Department of Computer Engineering

Date:

CERTIFICATE

This is to certify that the project report entitled

"Efficient data communication and storage using synchronized servers with load distribution facility"

Submitted by

MOHIT PATIL	**405C024**	**EXAM SEAT NO: B150234339**
AMIT RAJ	**405C002**	**EXAM SEAT NO: B150234215**
SMIT MAHAJAN	**405C017**	**EXAM SEAT NO: B150234309**
NEHA GOUNDADKAR	**405C009**	**EXAM SEAT NO: B150234263**

is a bonafide work carried out by him/her under the supervision of Prof. Aparna Kalaskar and it is approved for the partial fulfillment of the requirements of Savitribai Phule Pune University, Pune for the award of the degree of Bachelor of Engineering (Computer Engineering) during the year 2020-21.

Prof. Aparna Kalaskar
Guide
Department of Computer Engineering

Prof. M. P. Wankhade
Head
Department of Computer Engineering

Dr. S. D. Lokhande
Principal Sinhgad College of Engineering

ACKNOWLEDGEMENT

We would like to express our deepest appreciation to all those who provided us the possibility to complete this report. A special gratitude I give to the project guide, Prof. Aparna Kalaskar, whose contribution in stimulating suggestions and encouragement, helped us to coordinate our project. Furthermore, we would also like to acknowledge with much appreciation the crucial role of the respected HOD of Computer Department Prof. M. P. Wankhade, who gave the permission to use all required and necessary materials to complete the study. Thanks to respected Principal Dr. S. D. Lokhande for help. Special thanks to our friends for the comments and guidance during the completion of our project study and report.

MOHIT PATIL
SMIT MAHAJAN
AMIT RAJ
NEHA GOUNDADKAR

ABSTRACT

In the early 2000's, when data used to be of simple structure and small size, the client-server architecture handled the sharing and storage of data pretty efficiently. But as the technology progressed, the data became more and more complex and huge in size which led to load on servers for storing and sharing the data. Here, the P2P network architecture comes into picture where the peers could communicate and share the data without any centralized server.

The basic idea behind the proposal of this system is the combined use of segmented file transfer on P2P network. This makes it possible to download a single file from multiple peers at the same time. Moreover, it also includes a synchronized server module for efficient storage and updating data more efficiently on the servers.

This paper presents a system in which data transfer and storage system uses file downloading based on peer-to-peer network architecture. It's the software developed for sharing/downloading huge data files and storing important data efficiently. It's very convenient to share data using this system as the data is spread across the peer-to-peer network. The synchronized servers provide reliable data storage for any file format. All the peers present in the network communicate to other peers using the information present in the tracker. Tracker is responsible for maintaining the peers in the whole network.

List of Figures

List of Tables

Abbreviation

ML	Machine Learning
P2P	Peer-to-Peer
SHA	Secure Hash Algorithm
SRS	Software Requirement Specification
UML	Unified Modeling Language

TABLE OF CONTENTS

Chapter 1

INTRODUCTION

1.1 Background and Basics

The past few decades have experienced a huge paradigm shift from traditional client-server systems to peer to peer (P2P) network. In a client-server-based system, greater bandwidth is consumed as the demand for a particular file increase. Whereas in a P2P system, as the demands for a file increases, the nodes seeding that file also rise thus decreasing the delivery cost per file distributed. A P2P network is faster, robust and reduces the load on a single server.

Segmented Data Transfer is a technique of breaking files into chunks, downloading these chunks of the file simultaneously from a source and aggregating them on the client who requested the download. A file downloaded by using the concept of segmented file transfer is observed to have much lesser download time. The major advantage of using the concept of segmented file transfer (both upload and download) files over the internet is that the user will be able to utilize its internet connection to the maximum as the amount of data that is wasted will be reduced as well as the amount of time needed to transfer the file will also reduce without compromising on the security of the data.

BitTorrent, one of the most popular P2P technology, makes the distribution of files much easier by consuming less bandwidth from the publisher. A meta info file (called the torrent) file is created which contains information like file name, file size, hashing details along with the URL of the tracker. The download is initiated by opening this torrent file in the BitTorrent client (a free application). The system consists of a tracker which maintains the details of all the peers involved and helps to find peers. The seed (a user who initially has the file to be downloaded) uploads the entire file. Once an entire copy is distributed amongst the other downloading users, the seeder can stop uploading the file while allowing other people to download. As discussed previously, while making the torrent file, the original file is divided into smaller pieces and the hash codes of these pieces are included in the torrent file.

1.2 Literature Survey

After doing a research on the papers on peer-to-peer file sharing efficient data communication and storage using synchronized with load distribution. One paper[7] from year 2020, in this paper of IEEE methodology use to increase the content availability, accelerating the download process. An improved applicable network coding scheme is referred as Super Generation Network Coding (SGNC). This is designed for file sizes ranging from 320MB to 5GB, and utilize three-piece scheduling policies namely: random, local RF, and global RF. Due to this paper pros we get are SGNC maximizes the generation size so that it is as close as possible to the optimal size without adding computational overhead and P2P content distribution systems is efficient in terms of content availability, download time, overhead, and decidability. We also get cons from this paper are Dense Network Coding (DNC) has been proposed theoretically considered as an optimal solution & have huge computational overhead, it is not viable for real-world systems. This is not for the overhead of very large files, i.e. more than 10GB's files.

In year 2020 paper[8], methodology use are an efficient direct and indirect file transfer protocol (C2CFTP) that transfers files between clients in a client-server system. Here the file transfer performance is be improved by selecting and using one of the proposed direct and indirect C2CFTPs depending on the situation when transferring a bundle of files. Pros of this paper are this can omit unnecessary file I/O overhead by relaying the file to the receiving client instead of storing the file at the server for indirect transmission. For direct transmission, instead of connecting a data channel every time a file is transmitted, the channel connection overhead is reduced. Cons is the file transfer is required within a predetermined time.

In year 2019 get two paper for literature survey, one[3] and [5], methodology used in paper[3] are Snapshot Replication, Merge Replication and Transactional Replication Pros and Cons are transactional Replication gives efficient data synchronization system and no load distribution is performed on synchronized servers resp. in the paper[5],methodology are architecture peer to peer communication network is used for distributed data transfer. It involves download different portion of file from multiple client. Pros for this paper are use of distributed data transfer results in faster data sharing as compared to centralized system. High bandwidth utilization. Load balancing is performed by using custom scoring logic. Cons for this paper is less secure as there is no server to check the data over the network.

1.3 Project Undertaken

1.3.1 Problem Definition

To design a system for distributed file sharing on P2P network using SHA256 algorithm and segmented file transfer technique. The system also consists of a separate module of synchronized servers for efficient data update and storage. The file transfer module is accompanied by a load distribution facility to avoid load on a single peer.

1.3.2 Scope Statement

1. The sharing of data on peer-peer network should be independent of the file name.
2. Server synchronization should assure that the data is not lost even if it is accidently deleted from one server.
3. The system should be able to balance the load while sharing a file which is in huge demand.

1.4 Organization of the Project Report

The project report consists of 4 chapters to represent the project idea and its functioning. Chapter 1 is the introduction to the project idea that includes background and basics, literature review. It clarifies the project idea, problem statement and scope.

Chapter 2 is project planning and management with software requirement specification (SRS). It covers functional, non-functional and external interface requirements. Details about required environment setup is also mentioned.

Chapter 3 is about analysis and design of project and its implementation. IDEA matrix, mathematical model, feasibility analysis is included along with all the necessary UML diagrams to well understand the control system

Chapter 4 focuses on testing to be performed on the modules, it includes test cases for the Unit testing, Integration testing and Acceptance testing.

Chapter 2

SYSTEM REQUIREMENTS AND SPECIFICATIONS

2.1 Detail System Requirement Specification (SRS)

2.1.1 System Overview

This system would be built for sharing/download huge data files and storing important data efficiency. It’s very convenient to share data using this system as the data is spread across the peer-to-peer network. The synchronized servers provide reliable data storage for any file format.

In this system tracker is responsible for maintain the peers in whole network. This classification model would then be implemented to share data files and storing using synchronized servers.

2.1.2 Functional Requirements

- Tracker should be active for any data transfer over P2P network.
- Hosts must know tracker IP and server IP address.
- To transfer a file, at least one host should have the complete file

2.1.3 Non-Functional Requirements

- Load balancing should be done at tracker.
- Any changes in one server should be replicated in another server within 10 seconds.

2.1.4 Deployment Environment

Software Required

- Operating System : Ubuntu 16.04 and above
- Editor: Terminal
- Language: python latest version and cpp
- RAM : 2 GB

Hardware Required

- Minimum intel core i3 processor
- Minimum 5GB hard disk space.
- Minimum 2GB RAM.

2.2 Project Process Modelling

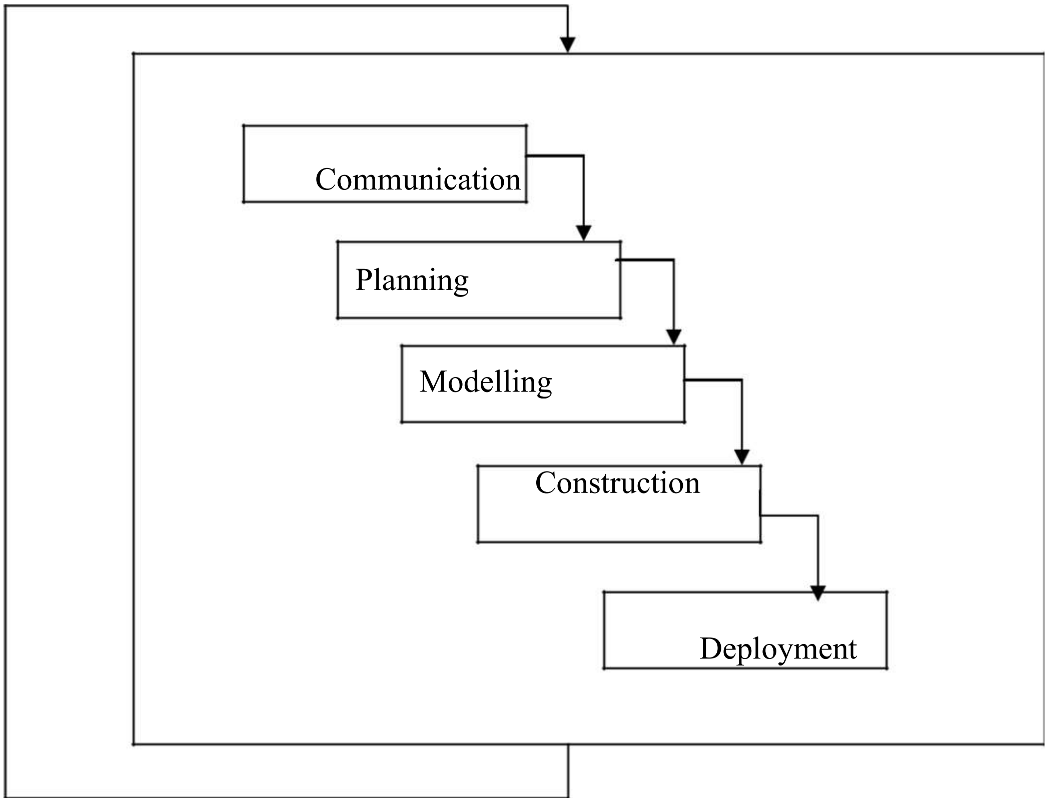


Figure 2.2.1: Project Process Modelling

To develop this project, an evolutionary process model for software development is adopted. Incremental Model is a process of software development where requirements are broken down into multiple standalone modules of software development cycle. Incremental development is done in steps from analysis design, implementation, testing/verification, maintenance.

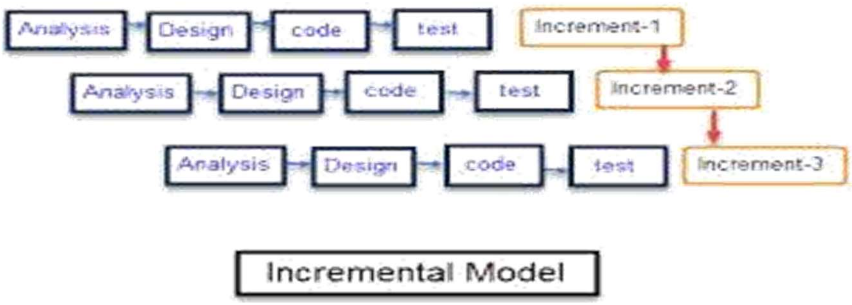


Figure 2.2.2: Incremental model

When an incremental model is used, the first increment is often a core product. That is, basic requirements are addressed, but many supplementary features (some known, others unknown) remain undelivered. The core product is used by the customer (or undergoes detailed review). As a result of use and/or evaluation, a plan is developed for the next increment. The plan addresses the medication of the core product to better meet the needs of the customer and the delivery of additional features and functionality.

This process is repeated following the delivery of each increment, until the complete product is produced. The incremental process model, like prototyping and other evolutionary approaches, is iterative in nature. But unlike prototyping, the incremental model focuses on the delivery of an operational product with each increment. Early increments are stripped down versions of the functional product, but they do provide capability that serves the user and also provide a platform for evaluation by the user. Incremental development is particularly useful when staffing is unavailable for a complete implementation by the business deadline that has been established for the project. Early increments can be implemented with fewer people. If the core product is well received, then additional staff (if required) can be added to implement the next increment. In addition, increments can be planned to manage technical risks.

In this project, we have broken down the entire project into three modules having their own functionalities. In the very first iteration, the extraction and classification module has been built which is the core product of this project. In second iteration generating sensitivity of tweet has been added to the classification module. The third major increment of the project was to develop web API to communicate between model and mobile app. Every module was built using all the steps of waterfall model

2.3 External Interface Requirements

User Interfaces

//Details of UI (ss of terminal n all)

Hardware Interfaces

It is not that hardware dependent as it is a small sized and almost no graphics application so it won’t push the system’s hardware on which it is being run. It is solely developed for desktops and not for mobile platforms.

Software Interfaces

The system is to be developed using C++ and python programming language and socket interfaces to establish connection with other devices/peers.

2.4 Other Nonfunction Requirement

2.4.1 Performance Requirement

The data sharing module in the system doesn't have much performance requirements as it is a light application but considering the download speed it depend on bandwidth of the network and the load balancing at the tracker (if any). The average download speed for a particular file increases as the number of downloads increase as they are become uploaders at later point of time. Also, the storage module consisting of synchronized servers don't have performance requirements as it just mirrors the complete data reliability.

2.4.2 Safety Requirement

The system doesn't take into account what type of data is present on the network. Therefore, firewall should be enabled at the peer end to ensure security as the security is not one of the core requirements of this application. Considering storage module, there are no security issues as the connection establishment is one to one TCP connection.

2.4.2 Security Requirement

The user needs administrative permissions to the system he or she is using is the only security requirement for this application.

2.5 Project Scheduling

2.5.1 Time Line Chart

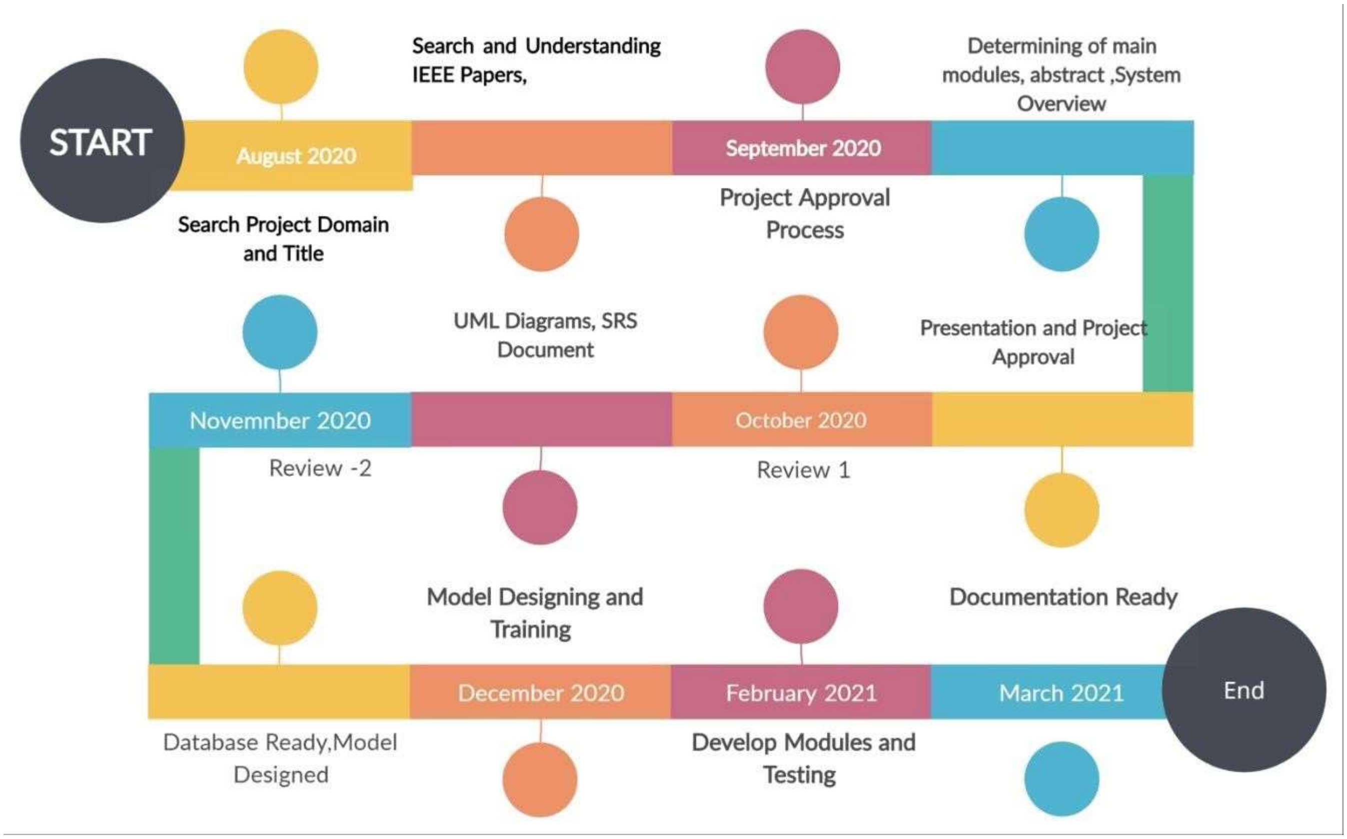

START
August 2020
Search Project Domain and Title
Search and Understanding IEEE Papers,
UML Diagrams, SRS Document
September 2020
Project Approval Process
Determining of main modules, abstract ,System Overview
Presentation and Project Approval
October 2020
Review 1
Model Designing and Training
Novemnber 2020
Review -2
Database Ready,Model Designed
December 2020
February 2021
Develop Modules and Testing
Documentation Ready
March 2021
End

Figure 2.5.1: Time Line Chart – 1

Chapter 3

ANALYSIS & DESIGN

3.1 IDEA Matrix

Table 3.1.1 : I

Idea	Deliverables	Parameters affected
Increase	Data transfer speed.	File sharing and user convenience
Increase	Number of data copies of data at different servers	Data reliability
Ignore	Type of file to be shared	File type

Table 3.1.2 : D

Idea	Deliverables	Parameters affected
Decrease	Excess load on a machine	Load distribution
Decrease	Transfer latency	Rate of transfer
Define	Meta data file	Communication
Deliver	Fast transfer of file and reliable storage	Efficient storage and transfer

Table 3.1.3 : E

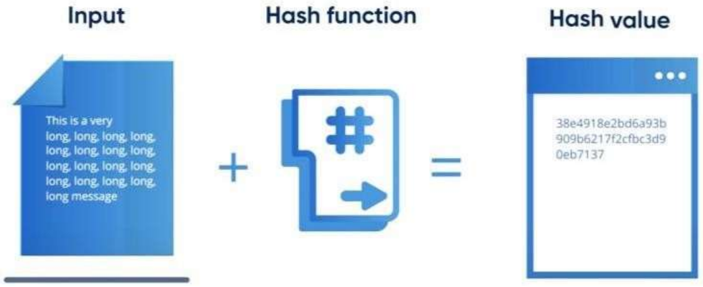
Idea	Deliverables	Parameters affected
Eliminate	Complete load on single machine	Data availability
Eliminate	Reliability on a single server	Data reliability
Educate	Educate project members and users	Project member and user

Table 3.1.3 : E

Idea	Deliverables	Parameters affected
Advance	Advancement File sharing and storage	Transfer and storage of data
Avoid	Web-based user interface	UI
Advantage	Provide faster data transfer with efficient and reliable data storage module.	User convenience

3.2 Mathematical Model

Hashing:- It is the process of converting an input of any length into an array of fixed size string text or number using a mathematical function


Input
Hash function
Hash value
This is a very long, long, long, long, long, long, long, long, long, long, long, long, long, long, long, long, long message
+
#
=
38e4918e2bd6a93b
909b6217f2cfbc3d9
0eb7137

SHA-256:- It is one of the successor hash functions to SHA-1 (collectively referred to as SHA-2), and is one of the strongest hash functions available.

It is like the fingerprints of the data. Even if only one symbol is changed the algorithm will produce different hash value. SHA256 algorithm generates an almost-unique, fixed size 256-bit (32-byte) hash.

Mathematical Function-

1. SHR 32- Shift right input bit by 32 bits.
2. ROTR- perform Circular right shift on input bit
3. XOR- Perform XOR on two binary number
4. ADD- Add two binary number
5. Lowercase Sigma zero- Perform ROTR (7 bits) on input -> ROTR (18 bits) -> SHR (3 bits)
6. Lowercase Sigma one- Perform ROTR (17 bits) on input -> ROTR (19 bits) -> SHR (10 bits)
7. Uppercase Sigma zero- Perform ROTR (2 bits) on input -> ROTR (13 bits) -> SHR (22 bits)

8. Uppercase Sigma zero- Perform ROTR (6 bits) on input -> ROTR (11 bits) -> SHR (25 bits)
9. Choice(ch)- Uses first input to decide whether to take input from first or second.
10. Majority ()- It takes three input binary string and consider majority of three bits as input.
11. Constant- calculate using cube root of prime number (3√prime number).

$$T1= \sum 1(e) + ch\ (e, f, g) + h + K+W$$

$$T2=\sum 0(a) + Majority\ (a, b, c)$$

Here a, b, c, d, e and f are working variables.

3.3 System Architecture

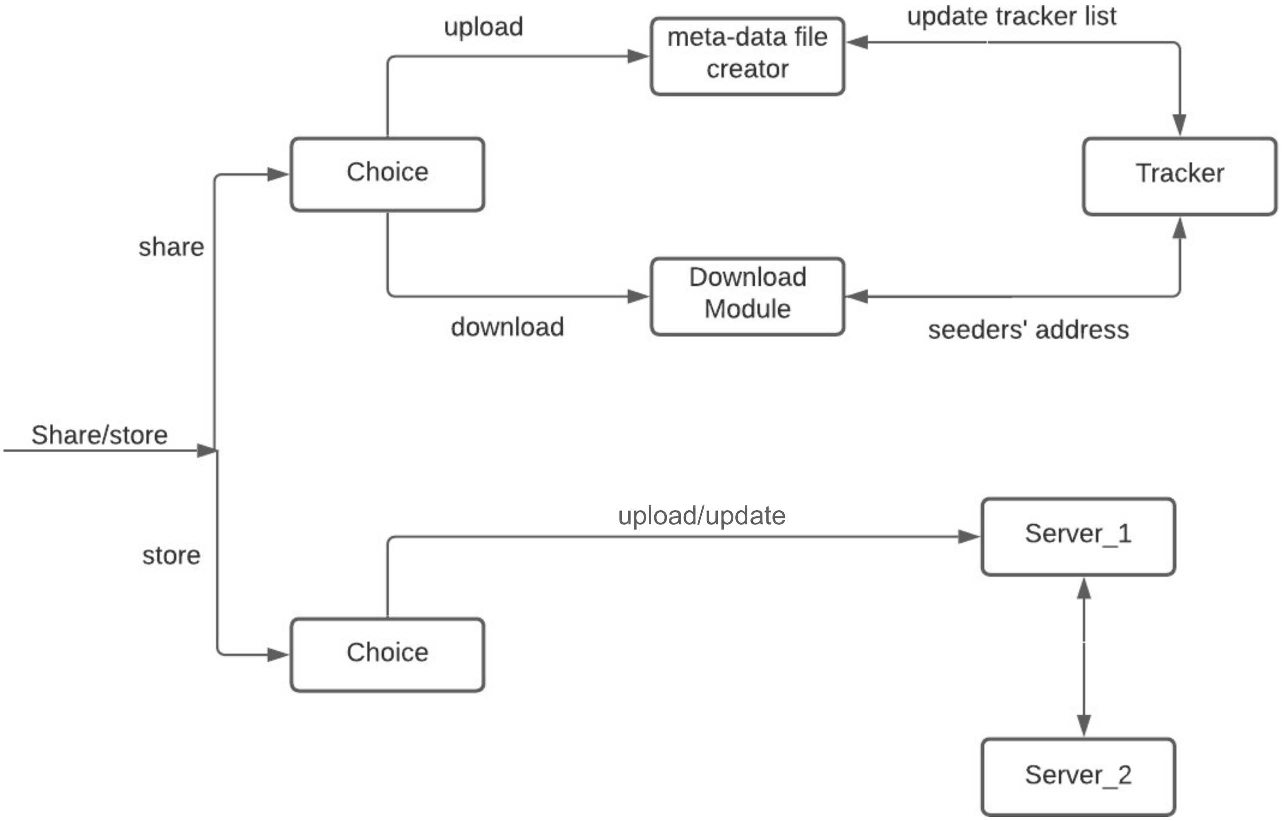


Figure 3.3: System Architecture

3.4 Use Case Diagram

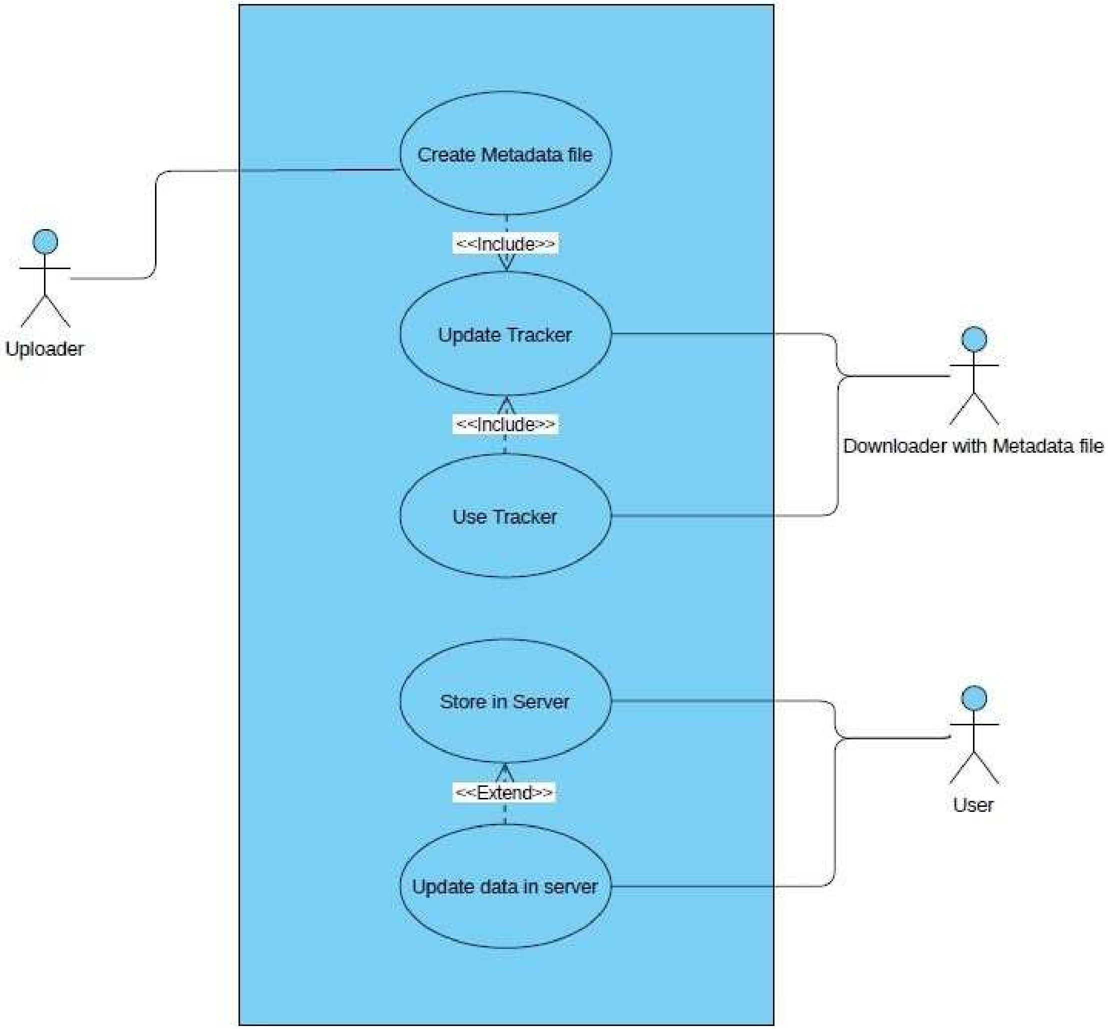


Figure 3.2: Use Case Diagram

3.5 Class Diagram

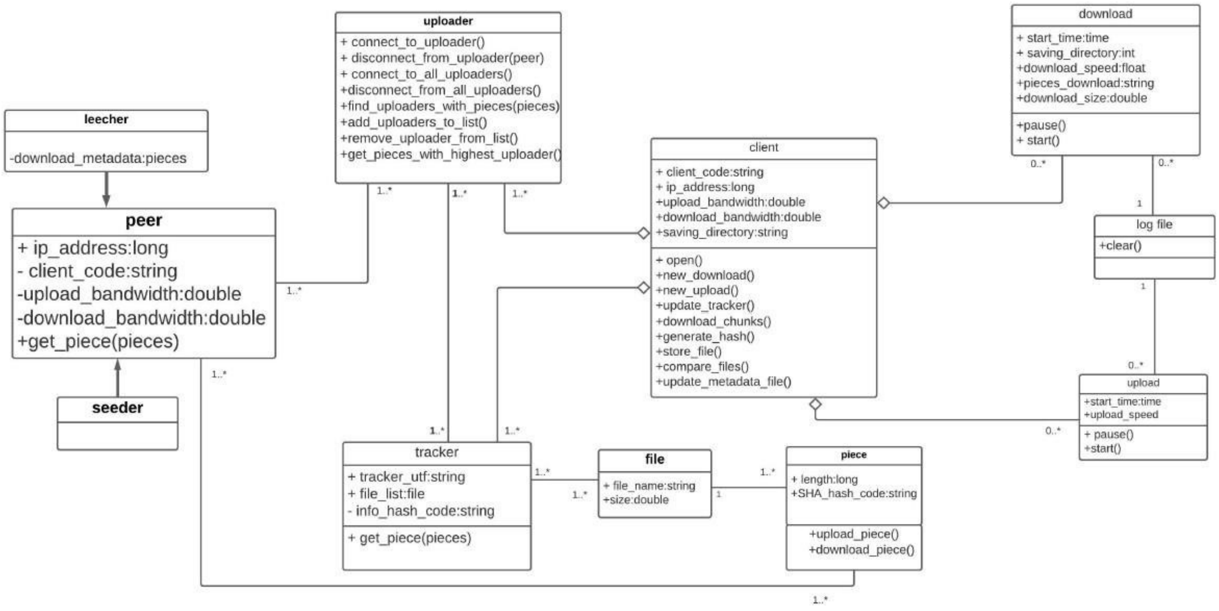


Figure 3.3: Class Diagram

3.6 Activity Diagram

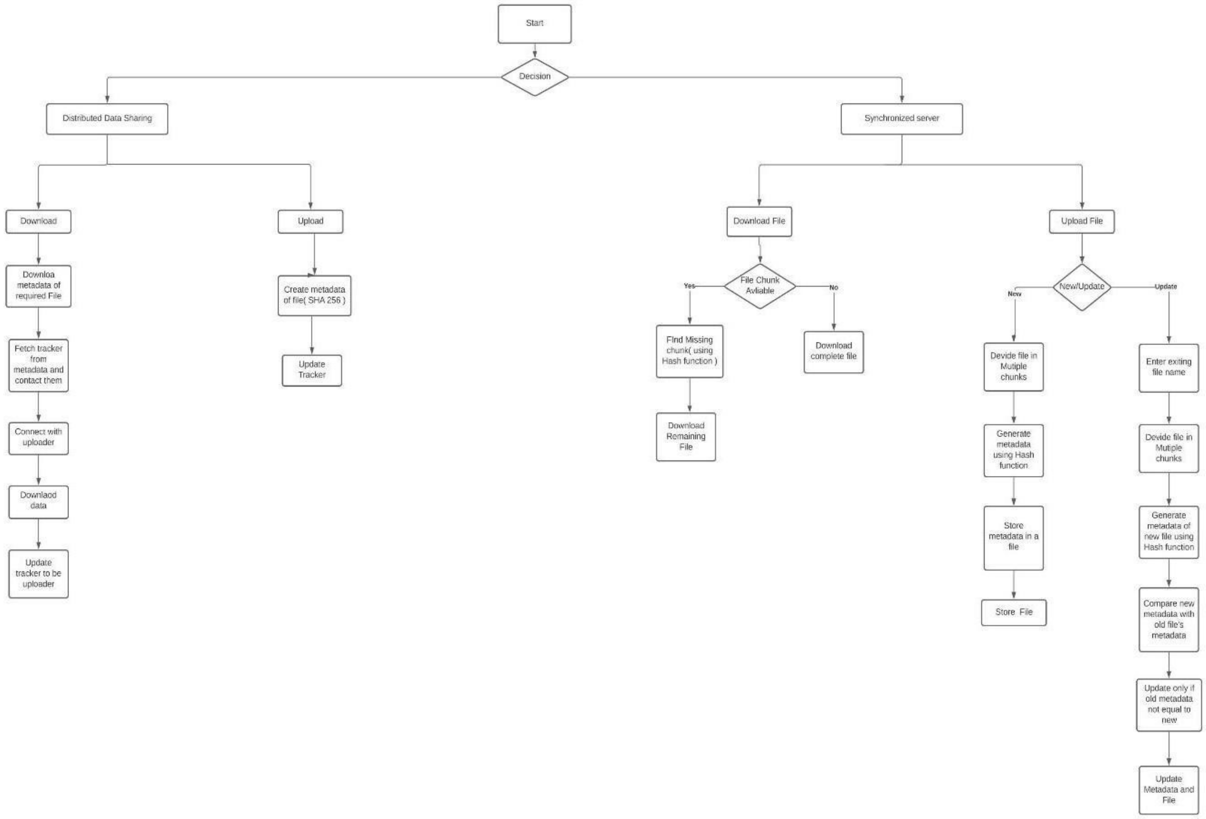

Start
Decision
Distributed Data Sharing
Synchronized server
Download
Upload
Download File
Upload File
Downloa metadata of required File
Create metadata of file(SHA 256)
Update Tracker
Fetch tracker from metadata and contact them
Connect with uploader
Downlaod data
Update tracker to be uploader
File Chunk Avliable
Yes
No
FInd Missing chunk(using Hash function)
Download complete file
Download Remaining File
New/Update
New
Update
Devide file in Mutiple chunks
Generate metadata using Hash function
Store metadata in a file
Store File
Enter exiting file name
Devide file in Mutiple chunks
Generate metadata of new file using Hash function
Compare new metadata with old file's metadata
Update only if old metadata not equal to new
Update Metadata and File

Figure 3.4: Activity Diagram

3.7 Sequence Diagram

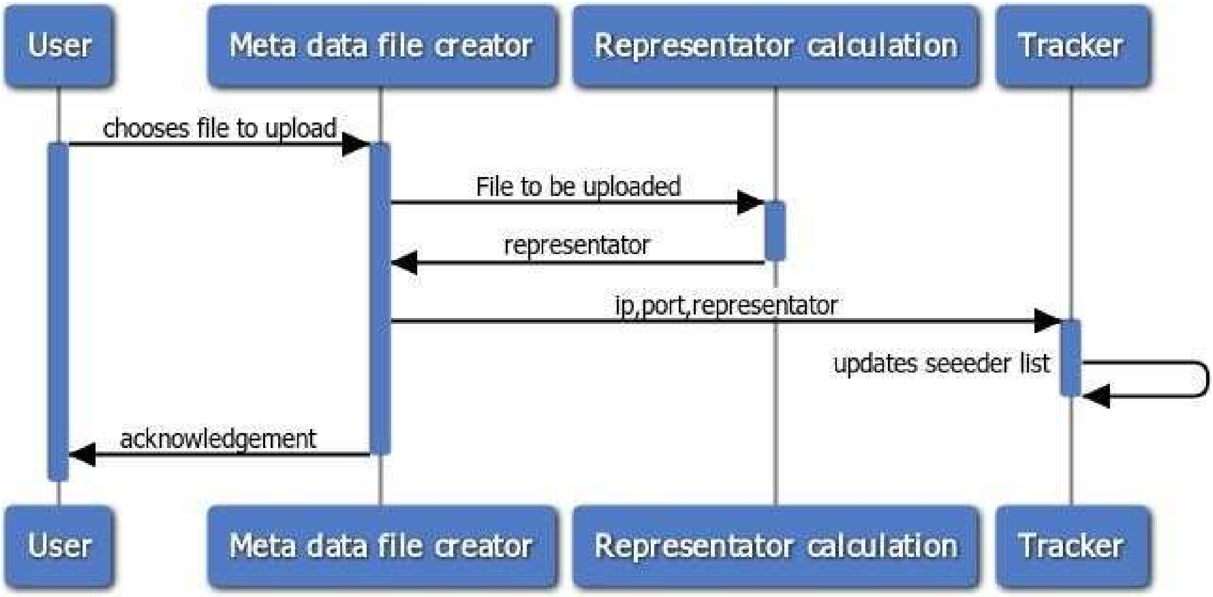


Figure 3.7.1: Sequence Diagram(Upload)

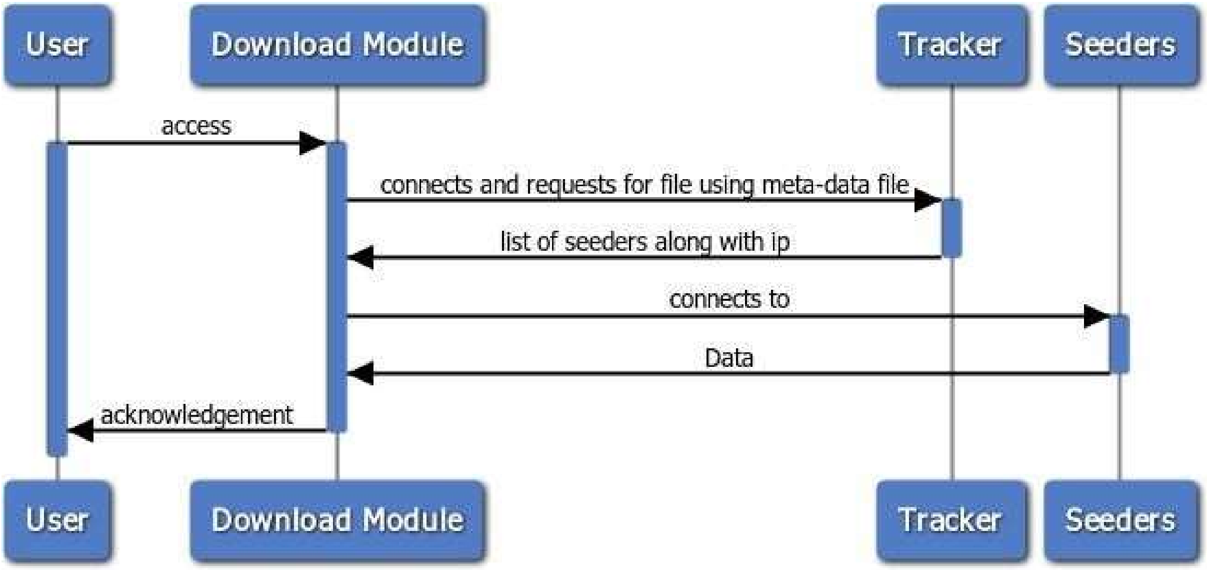


Figure 3.7.2: Sequence Diagram(Download)

Chapter
TESTING

4.1 Unit Testing

Unit testing concentrates verification on the smallest element of the program module. Using the detailed design description important control paths are tested to establish errors within the boundaries of the module. In this system each sub modules tested individually.

Table 4.1.1: Update Module

Test to be conducted	Input	Expected output	Actual output	Test Result
Update file	File	File Updated Successfully	-	-
Update file	Invalid File	File Not Updated Successfully		

4.2 Integration Testing

Once all the individual unit have been tested there is need to test how they were put together to ensure no data is lost across interface, one module does not have an adverse impact on another and a function is not performed correctly. After unit testing each and every sub module is tested with integrating each other

4.3 Acceptance Testing

Acceptance testing, a testing technique performed to determine whether or not the software system has met the requirement specifications. The main purpose of this test is to evaluate the system’s compliance with the business requirements and verify if it has met the required criteria for delivery to end users. Acceptance test are black box system test. Each acceptance test represents an accepted result from the system. Customers are responsible for verifying the correctness of the acceptance test and reviewing test scores to decide which fails test are of highest priority

Chapter
REFERENCE

[1] Peer Assisted Parallel Downloading System International Conference on Communication and Signal Processing, April 4-6, 2019, India.

[2] Minimizing One-to-Many File Transfer Times using Multipath-Multicast with Reed-Solomon Coding 2019 31st International Tele traffic Congress (ITC 31).

[3] Research on Data Synchronization of Cement Intelligent Control Cloud Platform

[4] Generalized Secure Hash Algorithm: SHA-X

[5] Peer to Peer file sharing

[6] A High-Performance Parallel Computation Hardware Architecture in ASIC of SHA-256 Hash International Conference on Advanced Communications Technology(ICACT)

[7] Super Generation Network Coding for Peer-to-Peer Content Distribution Networks IEEE-2020

[8] Direct and Indirect File Transfer Protocols Between Clients in Client-Server Architecture