Split – Load

System

Submitted by

ABHISHEK DAS 13000216135

ALKA PRASAD 13000116138

AVNISH KUMAR 13000116121

MOHIT KUMAR 13000116094

Submitted for the partial fulfillment for the degree of Bachelor of Technology in Computer Science and Engineering



Techno Main Salt Lake

EM 4/1, Salt Lake, Sector – V, Kolkata – 700 091.

**CERTIFICATE**

This is to certify that the project “Split-Load System” prepared by Alka Prasad (13000116138), Mohit Kumar (13000116094), Avnish Kumar (13000116121), Abhishek Das (13000216135) of B.Tech (Computer Science & Engineering), Final Year, has been done according to the regulations of the Degree of Bachelor of Technology in Computer Science & Engineering. The candidates have fulfilled the requirements for the submission of the project report.

It is to be understood that, the undersigned does not necessarily endorse any statement made, opinion expressed or conclusion drawn thereof, but approves the report only for the purpose for which it has been submitted.

Mr. Srimanta Kundu Mrs. Poulami Dutta

----------------------------------------------- -----------------------------------------------

(Signature of the Internal Guide) (Signature of the HOD)

--------------------------------------------------------------------------------------------

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

--------------------------------------------------------------------------------------------

**ACKNOWLEDGEMENT**

We would like to express our sincere gratitude to our mentor of the department of Computer Science and Engineering, whose role as project guide was invaluable for the project. We are extremely thankful for the keen interest he / she took in advising us, for the books and reference materials provided for the moral support extended to us.

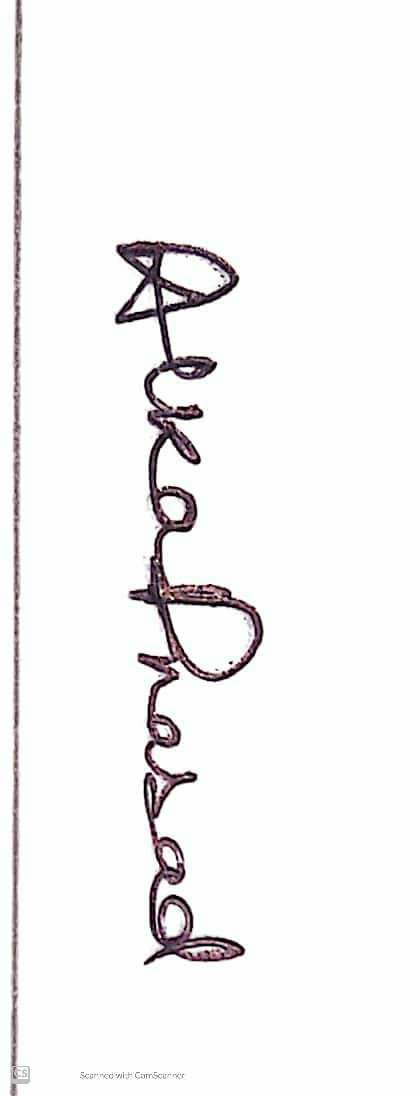
Last but not the least we convey our gratitude to all the teachers for providing us the technical skill that will always remain as our asset and to all non-teaching staffs for the gracious hospitality they offered us.

Place: Techno Main, Salt Lake

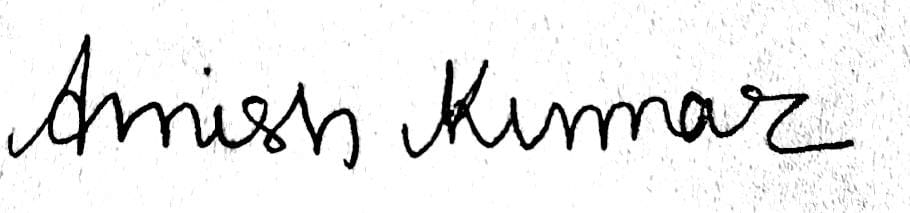
Date: 15/06/2019



………………………………



………………………………



………………………………



………………………………

Contents

[1 Introduction 1](#_Toc41936468)

[1.1 Abstract 1](#_Toc41936469)

[1.2 Problem Domain 1](#_Toc41936470)

[1.3 Related Studies 1](#_Toc41936471)

[1.4 Glossary 2](#_Toc41936472)

[2 Problem Definition 3](#_Toc41936473)

[2.1 Scope 3](#_Toc41936474)

[2.2 Exclusions 4](#_Toc41936475)

[2.3 Assumptions 4](#_Toc41936476)

[3 Project Planning 4](#_Toc41936477)

[3.1 Software Life Cycle Model 4](#_Toc41936478)

[3.2 Scheduling 5](#_Toc41936479)

[3.3 Cost Analysis 8](#_Toc41936480)

[4 Requirement Analysis 9](#_Toc41936481)

[4.1 Requirement Matrix 9](#_Toc41936482)

[4.2 Requirement Elaboration 9](#_Toc41936483)

[5 Design 10](#_Toc41936484)

[5.1 Technical Environment 10](#_Toc41936485)

[5.2 Detailed Design 11](#_Toc41936486)

[5.2.1 Hierarchy of Modules 11](#_Toc41936487)

[5.2.2 Data Flow Diagram 12](#_Toc41936488)

[6 Implementation 13](#_Toc41936489)

[6.1 Implementation Details 13](#_Toc41936490)

[6.2 System Installation Steps 14](#_Toc41936491)

[6.3 System Usage Instructions 15](#_Toc41936492)

[7 Test Results and Analysis 18](#_Toc41936493)

[7.1 Test Plan 18](#_Toc41936494)

[8 Conclusion 19](#_Toc41936495)

[8.1 Project Benefits 19](#_Toc41936496)

[8.2 Future Scope for improvements 19](#_Toc41936497)

[9 References / Bibliography 20](#_Toc41936498)

# Introduction

## Abstract

Our model Split-Load System revolves around the domain of Computer Networks and works on enhancing the download speed.

The report demonstrates the existent file downloading techniques along with their pros and cons.

Their inefficiencies and requirement of human intervention motivated us to undertake this project. We have incorporated the concepts of 2 highly successful and well-known services- Internet Download Manager and Torrent. While IDM works on a single computer, we have extended parallel segment downloading on multiple computers. Our proposed solution:

File is downloaded in segments on each Computer. These segments are shared among all of them which are then merged to get a complete file on each computer. The respective client nodes download the chunks of file and subsequently the merging process takes place which lets the complete file get downloaded in each client. The process of uploading segments while downloading has been borrowed from Torrent[7].

These features make our product efficient, robust and obviously, fast. The report also explains the assumptions taken and the various advantages provided by the product.

## Problem Domain

All the computers on the LAN that require the same file send a request to the server independently. All of them share the same bandwidth. As all clients work independently, even if one node goes down, none of the nodes are affected. There is no need of any communication among the Computers on the LAN. Huge traffic gets generated due to so many downloads and many times thereby creating redundancy. This also leads to a lot of data wastage.

Our solution provides much better result than the existing method. One computer on the LAN downloads the file from the server and then shares it on the local network, so that all the systems that require that file can download it locally. Since transfer of file on LAN is generally faster than that from the ISP, it proves to be quite efficient. As data is downloaded once from the server, it uses less data and hence consumes less time[3].

## Related Studies

• **Naive Solution:** All the clients work independently, and the file is downloaded separately on each computer. Due to this huge traffic takes. Huge amount of data is wasted, and a lot of manual effort takes place.

• **Better Approach**: One system downloads the file from the server. After the download, the system shares the file to all other systems via LAN.

• **IDM:** Most of the servers allow more than one connection to a SINGLE FILE STREAM from any client IP address requesting the file. IDM first checks whether multiple connections to the file can be made or not. Now that IDM knows that the server allows multiple connections, it makes requests to the server for the file that it wants to download, and the server returns a "BYTE STREAM". Now, we can specify from which byte, the byte stream to be returned from the server should start. IDM also supports `Dynamic Segmentation` [[1] i.e., the total file-byte-segments being downloaded may](https://www.quora.com/How-does-IDM-internet-download-manager-work-in-layman%E2%80%99s-terms#NXrbj) increase over the download process. It basically breaks an existing file-byte-segment into two for even more network utilization. Although the concurrent active network calls are fixed (what you set inside the application), but the calls can be divided over the file-byte-segments being downloaded [10].

* **Bit Torrent: Torrent[4],** peer to peer file sharing, works like this. First you have the trackers. When you add a torrent, your torrent client checks through the list of trackers it has, and contacts all the trackers part of the torrent to say "hey, I'm here!". Trackers also point you in the direction of other anonymous torrent seeders with the same torrent. Seeders are people who keep uploading a torrent to strengthen the network. When you download a torrent, you download it from the random seeders. It's ensured that each part is intact via the torrent's metadata via a checksum, it will ensure that the contents haven't been tampered by any of the peers [2].

## Glossary

* [Multicast] - Group communication where data transmission is addressed to a group of destination computers simultaneously. Multicast[11] can be one-to-many or many-to- many distribution.
* [LAN]: A Local area network (LAN)is a computer network that spans a relatively small area. Most often, a LAN is confined to a single room, building or group of buildings, however, one LAN can be connected to other LANs over any distance via telephone lines and radio waves.
* [Bandwidth]: Bandwidth is the capacity of a wired or wireless network communications link to transmit the maximum amount of data from one point to another over a computer network or internet connection in a given amount of time- usually one second.
* [Broadcast]- Broadcasting is the simultaneous transmission of the same message to multiple recipients. In networking, broadcasting[12] occurs when a transmitted data packet is received by all network devices.
* [Client]: A client is a piece of computer hardware or software that accesses a service made available by a server. The server is often (but not always) on another computer system, in which case the client accesses the service by way of a network.

• [Server]: a computer or computer program which manages access to a centralized resource or service in a network.

* [HTTP]: The Hypertext Transfer Protocol (HTTP) [13] is an application protocol for distributed, collaborative, hypermedia information systems. HTTP is the foundation of data communication for the World Wide Web, where hypertext documents include hyperlinks to other resources that the user can easily access, for example by a mouse click or by tapping the screen in a web browser.
* [TCP]: TCP stands for Transmission Control Protocol[14]. It is a fundamental protocol within the Internet protocol suite — a collection of standards that allow systems to communicate over the Internet. It is categorized as a "transport layer" protocol since it creates and maintains connections between hosts.
* [UDP]: User Datagram Protocol (UDP) [15] is part of the Internet Protocol suite used by programs running on different computers on a network. UDP is used to send short messages called datagrams but overall, it is an unreliable, connectionless protocol.

• [IP]: IP stands for Internet Protocol. It provides a standard set of rules for sending and receiving data over the Internet.

# Problem Definition

## Scope

The project relies on TCP as well as the concepts of torrent to get the job done. Downloads can be segmented because HTTP protocol [8] allows the client to specify the start position of the stream.

• Improvement in the speed of downloading a file from a server and distributing it to the other Computers on the Local Area Network. Currently, two models for downloading exist:

* Download the same file from the server simultaneously on all the computers.
* Download the file in a single computer and host it on the local network to be downloaded by the connected Computers. This method generally performs much better than the first one as the Bandwidth of a LAN is more than that of a connection from the ISP Download the file in a single Computer and host it on the local network
* to be downloaded by the connected Computers. This method generally performs much better than the first one as the Bandwidth of a LAN is more than that of a connection from the ISP.

## Exclusions

While creating our project, to increase the efficiency and to decrease the complexity, we have made certain exclusion. We have not yet incorporated an algorithm to distribute the load if a node or a series of nodes crashes. Also, our project excludes the algorithm to support simultaneous multiple file downloads.

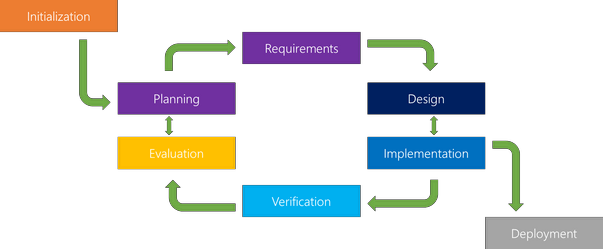
## Assumptions

• There should be multiple connections for a single IP.

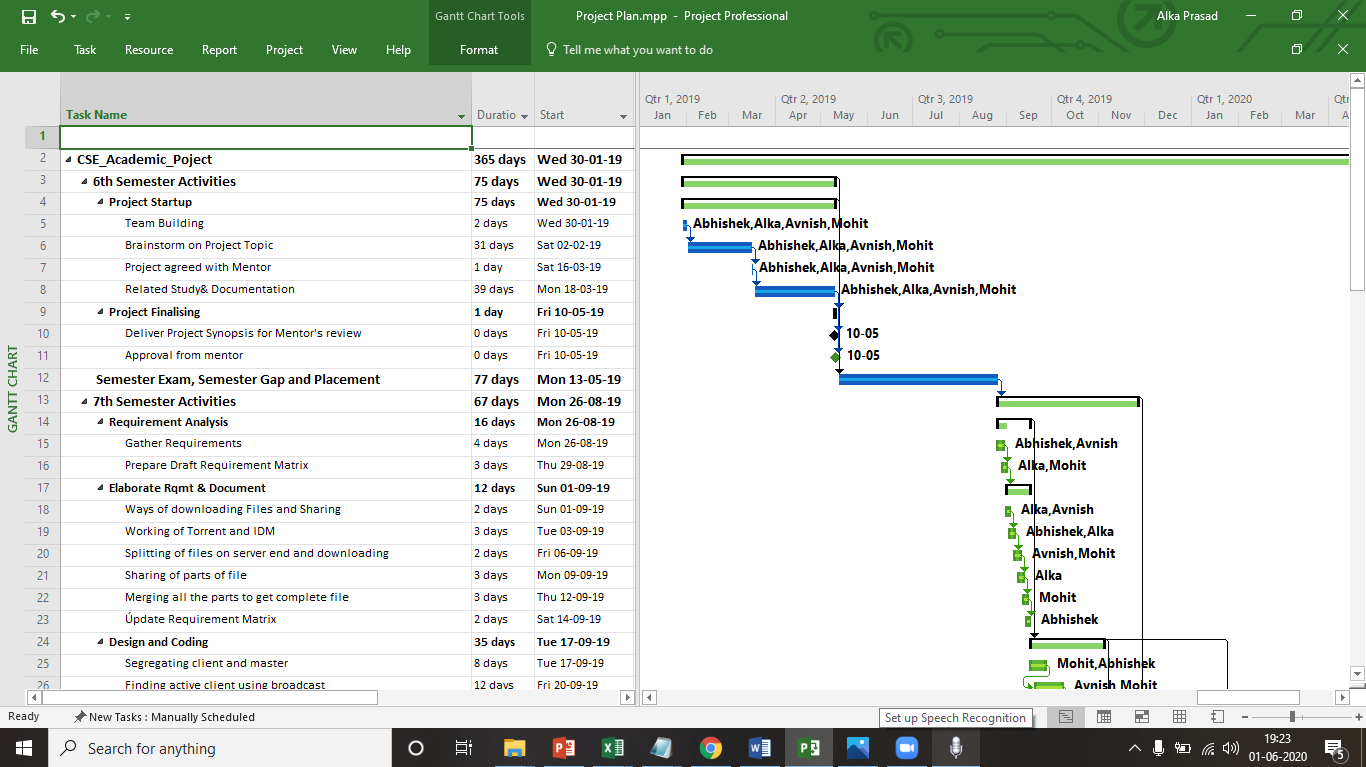
* There should not be any bandwidth limit per connection.
* As the sharing will be done over a LAN connection, therefore the bandwidth for the LAN must be much greater than the bandwidth of the ISP.
* There should be a stable internet connection for smooth functioning.

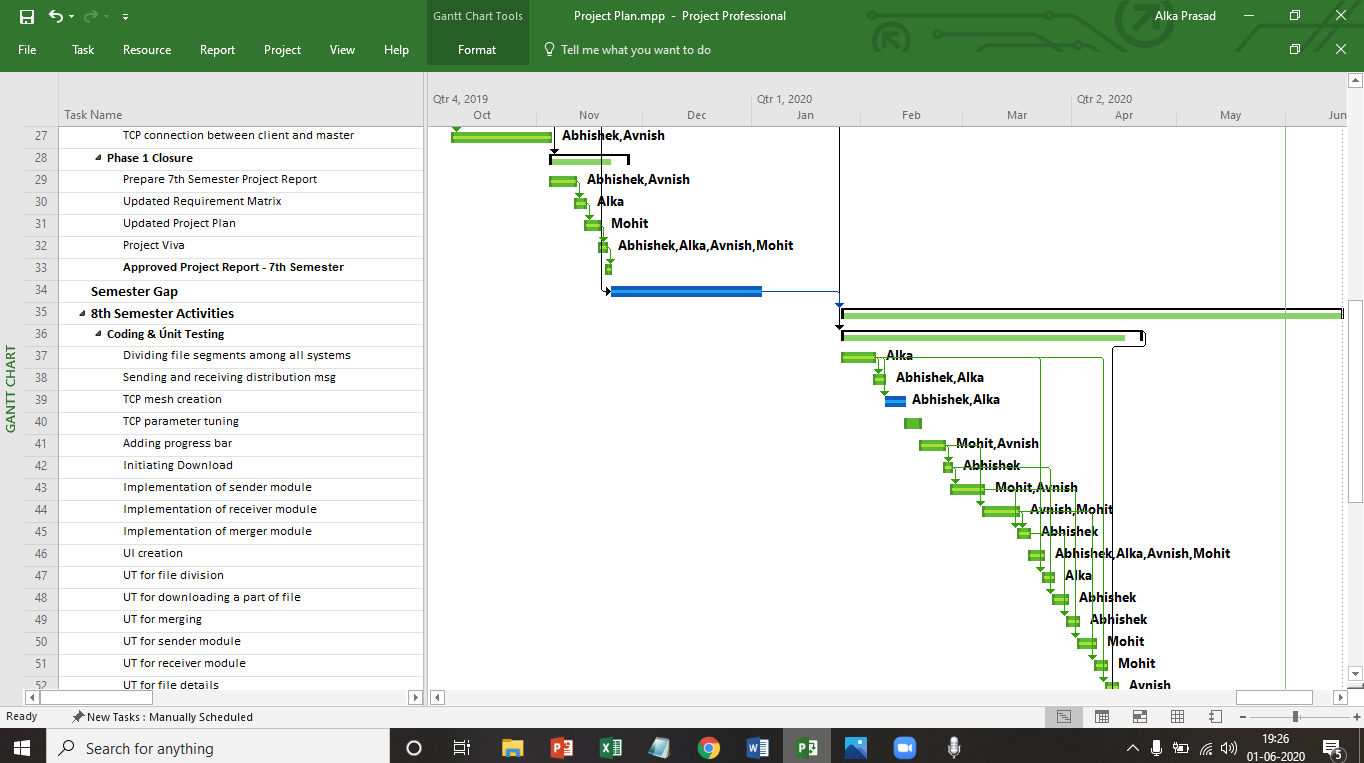
# Project Planning

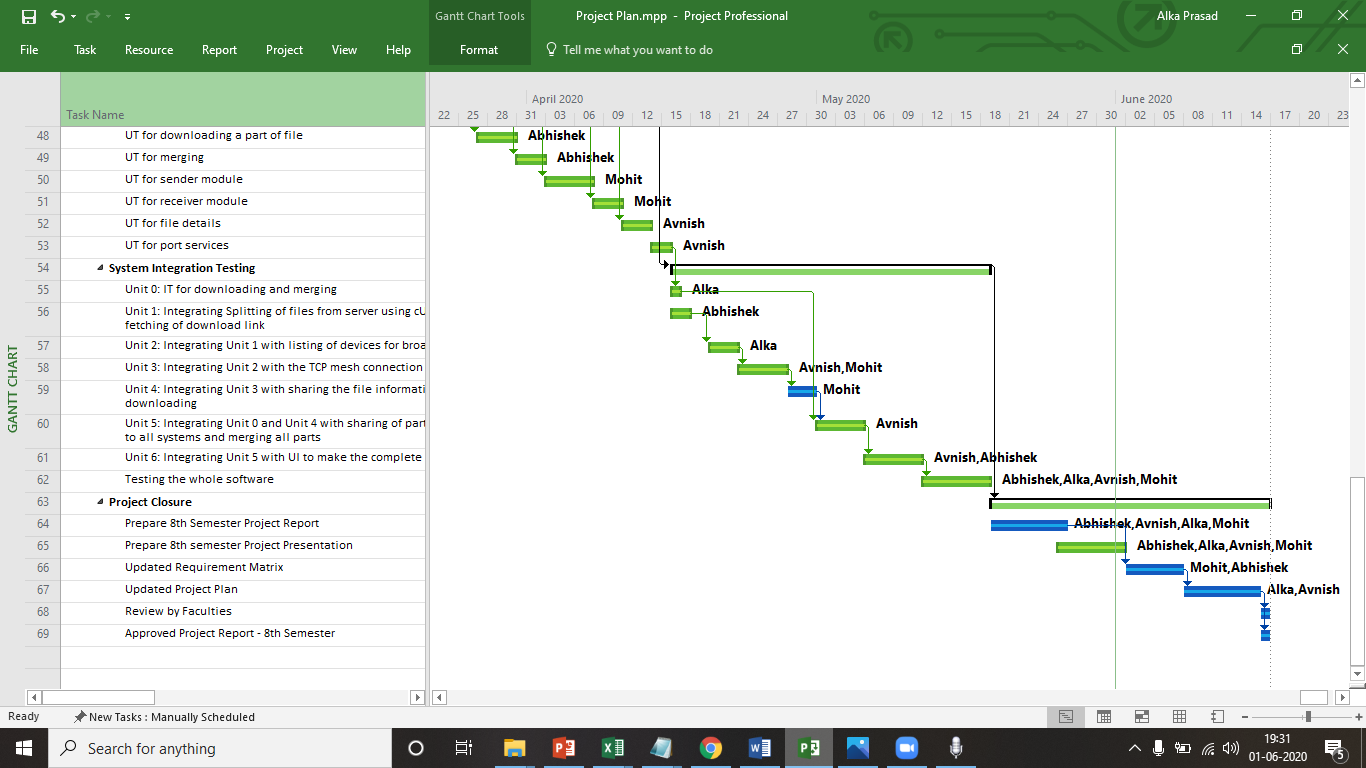
## Software Life Cycle Model

The classical waterfall model[16] is intuitively the most obvious way to develop software. However, it is not a practical model. This model considers the transition between two phases to be similar to a waterfall. That is, once a phase is complete, the various activities during the phase are assumed to be flawlessly done and there is no scope for rework later. In this project, we have used a modified version of the waterfall model, which is called the iterative waterfall model of development. In this model, there is a feedback path from every phase to its preceding phase. The feedback paths allow for correction of the errors committed during a phase, as and when these are detected in a later phase.

## Scheduling







## Cost Analysis

The basic COCOMO[5] model gives an approximate estimate of the project parameters. The project we are working on is an organic model.

Organic: A development project can be considered of organic type, if the project deals with developing a well understood application program, the size of the development team is reasonably small, and the team members are experienced in developing similar types of projects.



COST ESTIMATION

Estimation Formula:

Effort= 2.4\*(KLOC)1.05 PM

Tdev= 2.5\*(Effort) 0.38 Months

Where,

● KLOC= estimated size of the product expressed in Kilo Lines Of Code

● a,b,c,d are constants

● Tdev is the estimated time to develop the software, expressed in Months.

● Effort is the total effort required to develop the project estimated in Person Months.

Assume that the size of the organic type of software product has been estimated to be

1100 lines and the average salary of software engineers be Rs. 30,000/- pm. Determine the effort required and nominal development time of the project. (Use Basic COCOMO for this estimation).

Effort=2.4\*(1.1)1.05==2.652 PM.

Tdev= 2.5\*(2.652)0.38= 3.62 Months.

Cost required= 3.62\*30,000= Rs 1,08,656

# Requirement Analysis

## Requirement Matrix

## Requirement Elaboration

4.2.1 CUI-1.1:

Creation of a User Interface for the software.

4.2.2 MSD-1.2:

The machine that has been assigned as master starts the download.

4.2.3 BMS-1.3:

The broadcast message is sent to all the systems in the network.

4.2.4 CTM-1.4:

The supported systems create a TCP connection with Master

4.2.5 RCN-1.5:

Range of Bytes for division of the file is calculated.

4.2.6 SDL-1.6:

Sharing of download link with all other information to each client.

4.2.7 TCC-1.7:

All clients create a TCP connection amongst each other

4.2.8 TPT-1.8:

TCP connection is enhanced to improve the speed of download and share.

4.2.9 DFP-1.9:

Downloading of the file parts in all systems.

4.2.10 SFP-1.10:

Sharing of file parts while downloading.

4.2.11 MPF-1.11:

Merging of all parts of the file in all systems.

4.2.11 MPF-1.11:

Merging of all parts of the file in all systems.

4.2.12 FPS-1.12:

Get a Free TCP Port.

# Design

## Technical Environment

o **Hardware requirements:**

➢ **Minimum Hardware Requirements**

• Windows 8

• Processor: Pentium III

• Hard disk drive 40 GB

• RAM: 1 GB

• Cache: 512 kb

➢ **Preferred Hardware Requirements**

• Windows 10

• Processor: Intel Core i3

• Hard disk drive: 80 GB

• RAM: 2 GB

• Cache: 512 kb

o **Software requirements:**

* Our project is developed in Python 3.
* For developing the UI of our project, we have used pyQt5.
* cURL and copy command line tools are used in our software.
* In addition to this, we will include all our code in gitHub for better understanding and code sharing. Any Windows based operating system is primary requirements for software development.
* All system must be connected via LAN to internet, which is mandatory for the smooth execution of our software to download and share the file.

o **Other requirements:**

Other requirements include-

• Security

• Portability

• Correctness

• Efficiency

• Flexibility

• Testability

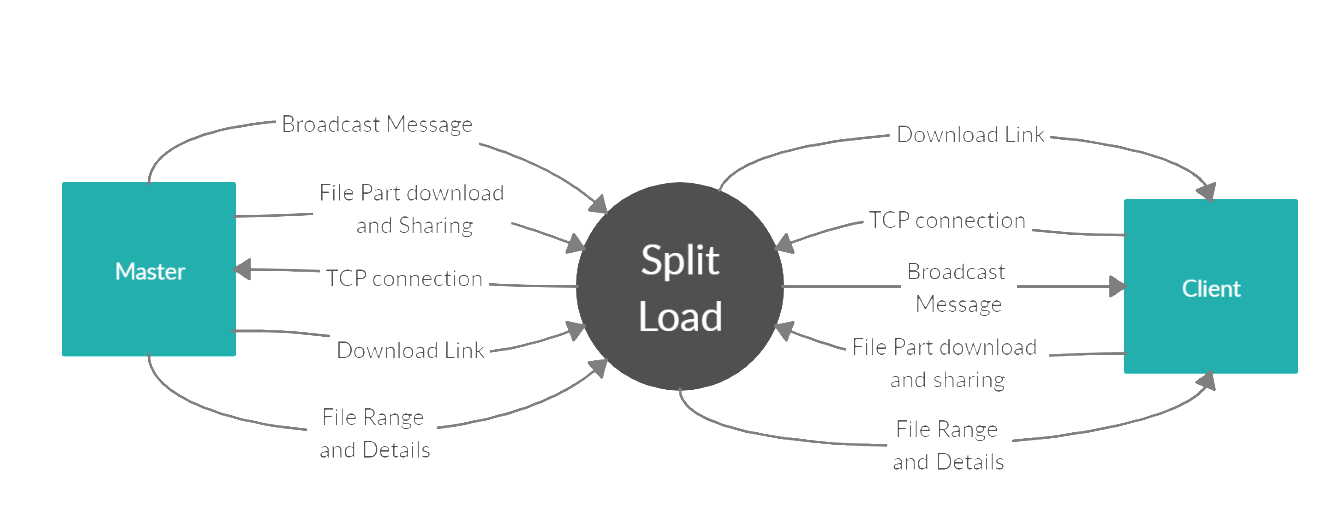
• Reusability

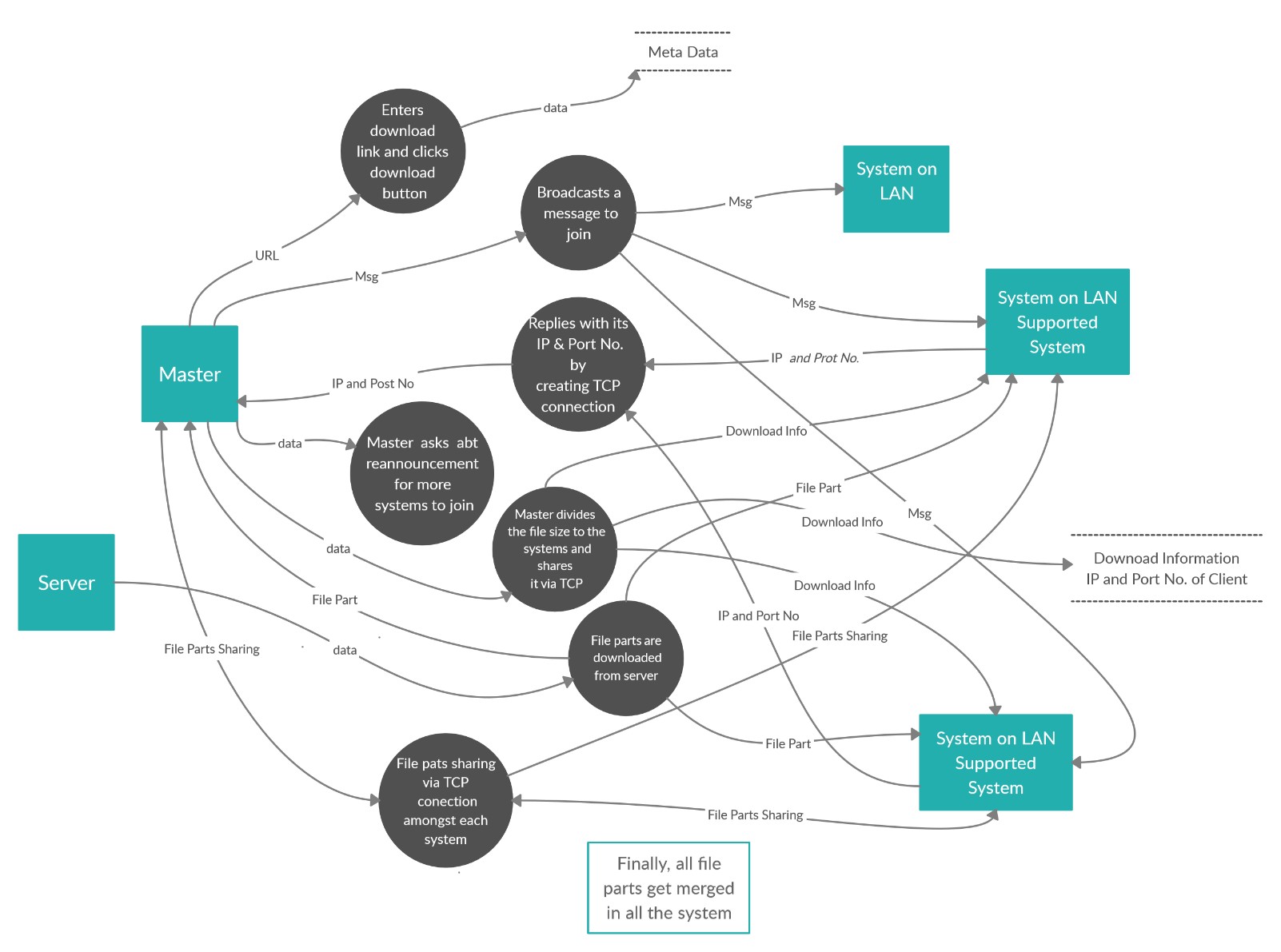
## Detailed Design

## Hierarchy of Modules

## Data Flow Diagram

Level 0:



Level 1: 

# Implementation

## Implementation Details

Our project focuses on download a file in parts in different computers and then sharing the parts over LAN and merging the parts to get the complete file on each and every system. This project can be broken into several modules:

* + 1. CUI: Creating UI

We create a user interface for the software to make it user friendly such that non-technical people can also use the software. Creating and integrating the UI with the program will prevent the user from using command prompt.

* + 1. MSD: Master Starts Download

The master system starts the whole process of downloading by clicking on “start” button of the software and enter the URL from which the file must be downloaded.

* + 1. BMS: Broadcast message sent

A broadcast message is sent to all the systems by the master system which are connected on the same network via LAN.

* + 1. CTM: Client creates a TCP connection with Master

The systems which have the software and are ready for download on the same network, requests a TCP connection [6] with the master. Master accepts the TCP connection request and saves the IP address of the system requesting the connection.

* + 1. RCN: Range Calculation

The master divides the file using cURL command and calculates the range i.e. it tells all the systems which system must download which part and shares the download link of the file.

* + 1. SDL: Sharing of Download Links

After the connection is setup, the master shares the information of download link and range of the file.

* + 1. TCC: TCP Mesh Connection amongst the Client

After sharing of link, all clients create a TCP connection amongst each other involved so that it can share its downloaded part with other systems.

* + 1. TCT: TCP connection Tuning

TCP connection is enhanced by incorporating features of UDP connection to enhance the speed of download and share.

* + 1. DFP: Downloading of the File Parts

After the TCP connection has been built, all the systems start downloading the file parts.

* + 1. SFP: Sharing of the File Parts

All the systems start sharing and receiving the part of the file while the file parts are getting downloaded.

* + 1. MPF: Merging of all the parts of the file

After receiving all the parts from all the systems i.e., after sharing and downloading of the files is over, merging of the files starts with the help of copy command line tool. It merges the sperate parts of the same files in each system.

* + 1. FPS: Free Port Service

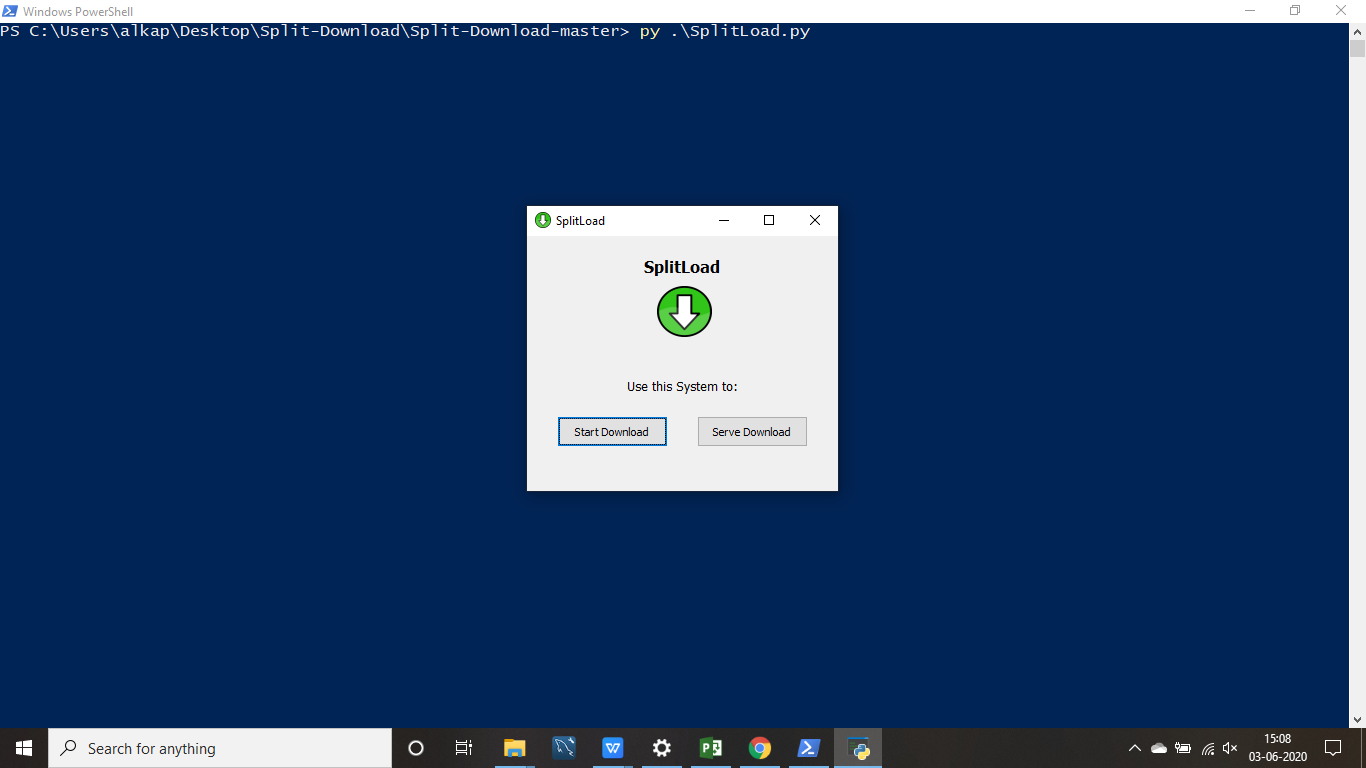
This can be used to get a free TCP port from the system.

## System Installation Steps

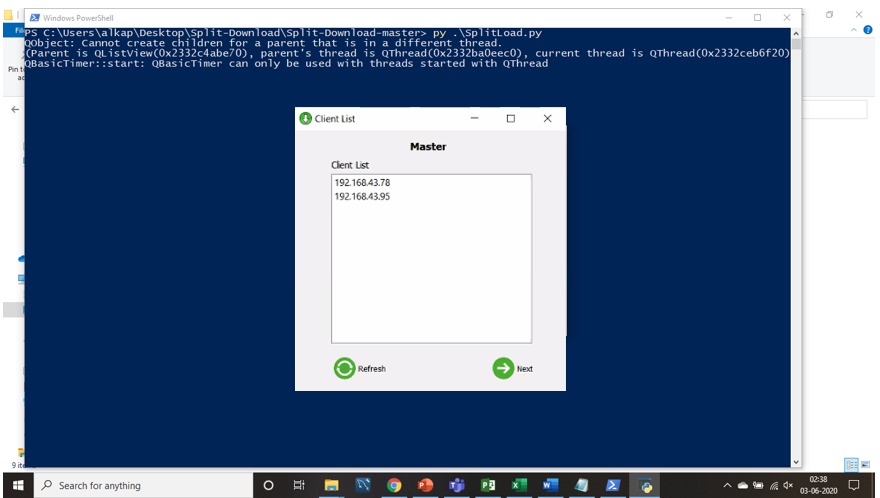
* Download and install Python 3.x on the system and ensure to add the path in the environment variable.
* Open command prompt and navigate to the project folder.
* Install “requests” from command prompt using the command “pip install requests”.
* Install “pyQt5” from command prompt using command “pip install pyQt5”.
* Install “enlighten” from command prompt using command “pip install enlighten”.
* Now start the project using the command “py SplitLoad.py”.
* The project is now up and running.

## System Usage Instructions

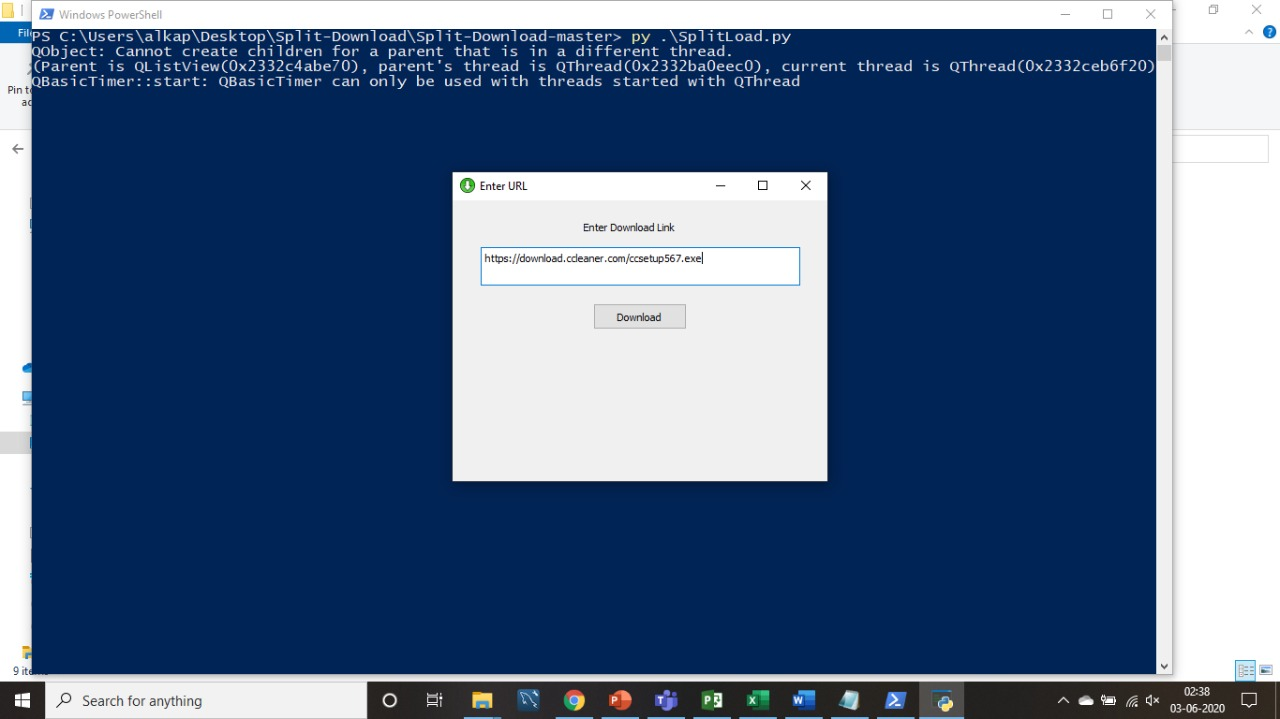
* A UI pops up with options of Start Download and Serve Download. All the respective clients press the Serve Download button.



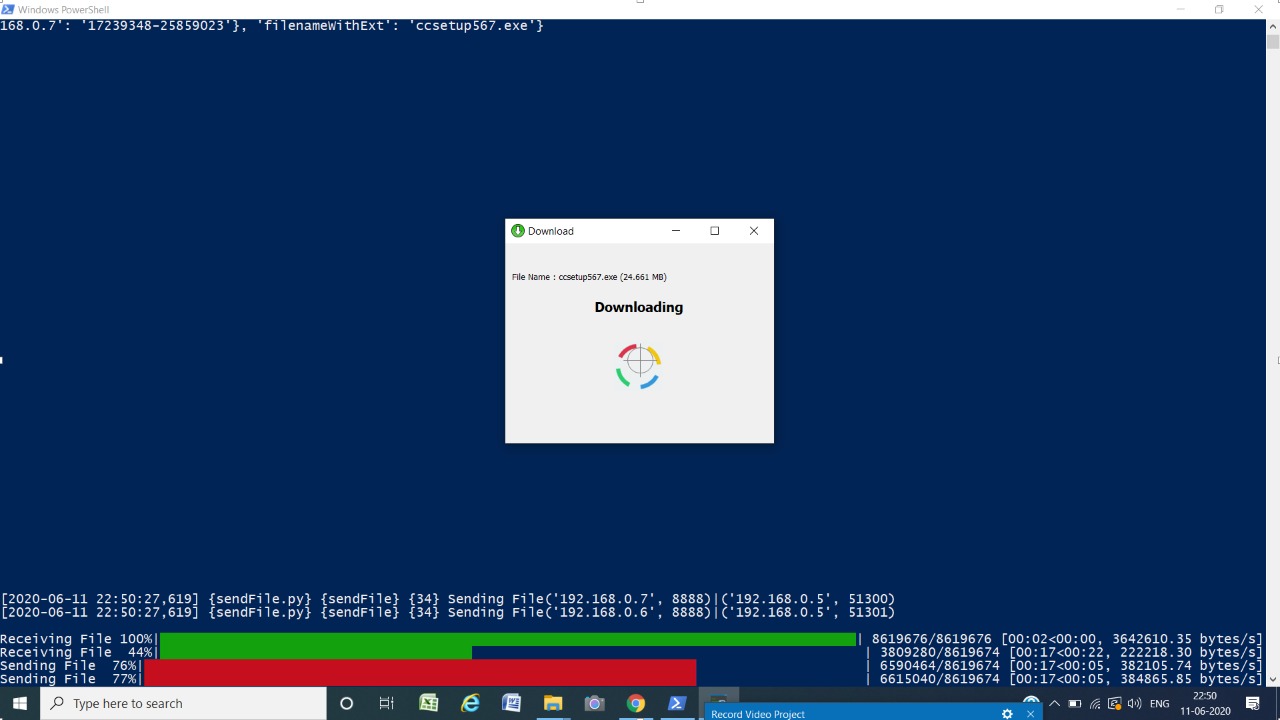
* The master on the other hand presses the Start Download option and obtains the list of the IPs of the respective clients.



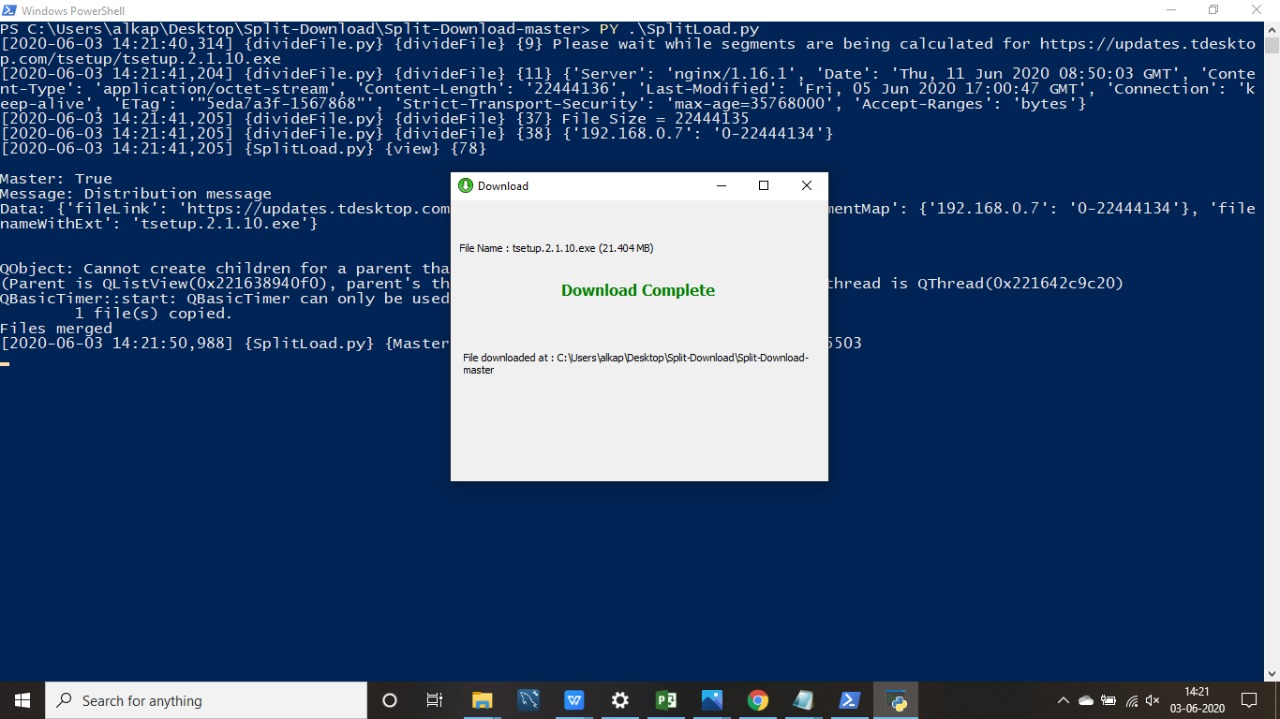
* Upon clicking the Next button in the previous pop up a new UI is presented with an option to enter the download link of the file that needs to be downloaded in all the systems.



* As the download button gets clicked the file starts downloading and simultaneous sharing begins in all the systems.

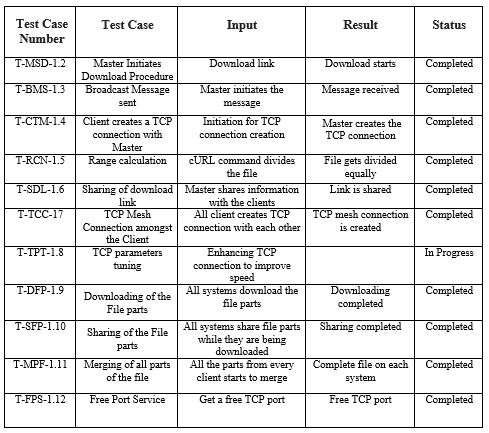


* The download gets completed and shared in all the systems. The filename with its size and the location where the file is downloaded in the respective systems are presented.



# Test Results and Analysis

## Test Plan



# Conclusion

Most of the time, the bandwidth provided by the ISP is under-utilized. While the CPU processes each packet, the next packets wait to be processed. The Project removes this bottleneck providing the processing power of several computers at a time. It also implements the concept of torrent to merge the files. This technique has already been proven to be reliable and successful. Hence, the software can prove to be quite useful in many scenarios.

## Project Benefits

Most of the organizations have several computers on their local network. Downloading and transferring a file to all those computers is a slow and tedious task. Even if we download a file somewhere and share it on the LAN, each computer must manually download it.

The Project improves the downloading speed by utilizing the full bandwidth of the network. The task of downloading a file is divided amongst many computers. Each one downloads its assigned part and efficiently merges it to provide an easy to use, one stop solution for downloading as well as sharing the file wherever required on the LAN. The USP of the software is that it’s one of a kind. It can have a huge impact in the corporate as well as educational institutions.

## Future Scope for improvements

While creating our project, to increase the efficiency and to decrease the complexity, we have made certain exclusion. We have not yet incorporated an algorithm to distribute the load if a node or a series of nodes crashes. Also, our project excludes the algorithm to support simultaneous multiple file downloads.

# References / Bibliography

1. Cardona, O. and Cunningham, J.B., International Business Machines Corp, 2008. System Load Based Dynamic Segmentation for Network Interface Cards. U.S. Patent Application 12/188,838
2. John Borland. BitTorrent file-swapping networks face crisis. 20.12.2004. <http://news.com.com/BitTorrent+file-swapping+networks+face+crisis/2100->

[1025\_35498326.html?tag=nl](http://news.com.com/BitTorrent+file-swapping+networks+face+crisis/2100-1025_35498326.html?tag=nl)

1. Bhide, C.W., Singh, J. and Oestreicher, D., Shiva (US) Network Corp, 1998. Performance

optimizations for computer networks utilizing HTTP. U.S. Patent 5,852,717.

1. Bram Cohen. Incentives Build Robustness in BitTorrent. May 22, 2003.

<http://www.bittorrent.com/bittorrentecon.pdf>

1. Khalifelu, Zeynab Abbasi, and Farhad Soleimanian Gharehchopogh. "Comparison and evaluation of data mining techniques with algorithmic models in software cost estimation."Procedia Technology 1 (2012): 65-71.
2. Hamad, Ashraf, Tao Wu, Ahmed E. Kamal, and Arun K. Somani. "On multicasting in

wavelength-routing mesh networks." Computer Networks 50, no. 16 (2006): 3105-3164.

1. Andrew Kantor. Despite reports, Grokster decision is a win for file sharing. http://www.usatoday.com/tech/columnist/andrewkantor/2005-07-01-[groksterdecision\_x.htm](http://www.usatoday.com/tech/columnist/andrewkantor/2005-07-01-groksterdecision_x.htm)
2. David Karger, Eric Lehman, Tom Leighton, Matthew Levine, Daniel Lewin and Rina Panigrahy. Consistent Hashing and Random Trees: Distributed Caching Protocols for Relieving Hot Spots on the World Wide Web.

<http://portal.acm.org/citation.cfm?doid=258533.258660>

1. LegalTorrents.com. <http://www.legaltorrents.com/>

[10] Jacques, R.S., Xerox Corp, 2011. Peer-to-peer file sharing system and method using downloadable data segments. U.S. Patent 7,970,835.

[11] Chawathe, Y., McCanne, S. and Brewer, E.A., 2000, March. RMX: Reliable multicast

for heterogeneous networks. In Proceedings IEEE INFOCOM 2000. Conference on

Computer Communications. Nineteenth Annual Joint Conference of the IEEE Computer

and Communications Societies (Cat. No. 00CH37064) (Vol. 2, pp. 795-804). IEEE.

[12] <https://en.wikipedia.org/wiki/Broadcasting_(networking)>

[13] Zhao, Yan. "Client/server two-way communication system framewor under HTTP protocol." U.S. Patent Application No. 09/776,478.

[14] Forouzan, B.A., 2002. TCP/IP protocol suite. McGraw-Hill, Inc..

[15] Liu, Peter Xiaoping, Max Meng, Xiufen Ye, and Jason Gu. "An UDP-based

protocol for internet robots." In Proceedings of the 4th World Congress on

Intelligent Control and Automation (Cat. No. 02EX527), vol. 1, pp. 59-65.

IEEE, 2002.

[16] Bassil, Y. (2012). A simulation model for the waterfall software development dddddlife cycle. arXiv preprint arXiv:1205.6904.