# PEER-TO-PEER FILE SHARING SYSTEM IN CAMPUS USING MULTI-AGENT

**THIDA WIN**

**M.C.Sc.** **JUNE 2022**

# PEER-TO-PEER FILE SHARING SYSTEM IN CAMPUS USING MULTI-AGENT

**By**

**THIDA WIN**

**B.C.Sc. (Hons.)**

**A Dissertation Submitted in Partial Fulfillment of the Requirements for the Degree of Master of Computer Science**

**(M.C.Sc.)**

**University of Computer Studies, Yangon**

**JUNE 2022**

# ACKNOWLEDGEMENTS

I would like to express my deepest gratitude and appreciation to the following persons who supported and helped towards the success of the thesis.

Above all else, I would like to express my appreciation and sincere thanks to **Dr. Mie Mie Khin**, Rector, University of Computer Studies, Yangon, for giving the opportunity to develop this thesis.

I greatly appreciate and acknowledge to **Dr. Tun Myat Aung,** Principal and Pro-rector of the University of Computer Studies, Hinthada, for his kind permission and administrative support.

I would like to thank course coordinators, **Dr. Tin Zar Thaw and Dr. Si Si Mar Win,** Professors, Faculty of Computer Science, University of Computer Studies, Yangon, for their guidance, management and encouragement in the progress of the thesis.

My greatest pleasure and the deepest appreciation go to my supervisor **Dr. Aye Mya Hlaing**, Associate Professor, Faculty of Computer Science, University of Computer Studies, Yangon, for her sympathetically support, valuable guidance, detailed supervision, patience, enthusiasm, encouragement and good advice throughout the thesis work.

My heartfelt thanks and respect go to **Dr. Soe Lai Phyue,** Professor, Head of Faculty of Computer Science, University of Computer Studies, Hinthada, for her general guidance and suggestions.

I wish to express my special appreciation to **Daw Hnin Yee Aung**, Lecturer, the Department of English, University of Computer Studies, Yangon, for modification of my thesis in checking grammatical errors, choice of words from the language point of view.

My sincere gratitude also goes to all my respectful teachers for teaching valuable lectures, guiding suggestions and sharing knowledge during the master course work and thesis work.

Moreover, I also extend my sincere gratitude to all my teachers from childhood to the present time.

Last but not least, I am deeply grateful to my beloved parents and my family for their mental and financial support, continuous encouragement, care and kindness, and endless love over the years.

I especially thank to my friends and colleagues for their true friendship, encouragement, support and assistance throughout my studies.

## STATEMENT OF ORIGINALITY

I hereby certify that the work embodied in this thesis is the result of original research and has not been submitted for a higher degree to any other University or Institution.

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

Date Thida Win

# ABSTRACT

Peer-to-Peer (P2P) file sharing is a technique of network file sharing. Clients can direct access and share the files to other clients in P2P network. The P2P file sharing system does not require any central server. P2P is more effective than client-server method because the computers have shared responsibilities to communicate with one another. Every computer on a P2P network can operate a server as well as a client. The integration of P2P network architecture and multi-agent technology are able to be an optimal solution for the real environmental problems. Multi-agent innovation has demonstrated better performance results about the further development, adequacy and accuracy in dynamic and distributed environments. This system depends on the multi-agent architecture and presents how to manage and share the files with the concept of load balancing mechanism on the P2P local area network. The proposed system is developed by utilizing the Microsoft .Net platform and SQL Server.

# CONENTS

# LIST OF FIGURES

# LIST OF TABLES

**Table** **Page**

# CHAPTER 1
# INTRODUCTION

Nowadays, Peer-to-Peer (P2P) network is uncommonly notable in file sharing. P2P network is used to download and share files like recordings, music, videos, images, digital books, games, programs, and so on. P2P frameworks are described by decentralized control, enormous scope and outrageous dynamism of their workspace. This system is implemented on P2P local area network and designed for file sharing on campus. Because of the way that there is no central server, there are no expenses charged by the server that is facilitating the application. Each peer is responsible for storing and sending the requested information.

Developing intelligent and autonomous agents is a central goal of Artificial Intelligence (AI). Agents are well known research objects in all fields. Agents derived from the AI field. Agents are extremely delicate to the environment and give a few elements like to follow up for the benefit of others, independent, intelligent, reactive, proactive, receptive, capacity, collaborate and negotiate. A Multi-Agent System is made out of independent various agents. Agents can relate, team up and arrange to deal with problems that are past the singular limits each agent.

The proposed system depends on the multi-agent framework. It is a distributed P2P file sharing system and composed of independent different multiple agents. In this system, each peer has four agents: Manager Agent, File Send Agent, Download Agent and Load Balancing Agent. File Send Agent can send multiple files to multiple peers. The system can save memory storage space and keep away from copy files due to Download Agent. Agents can reduce their workload and allocate their tasks to available peers within the same LAN by the Load Balancing Agent.

## 1.1 Objectives of the Thesis

The main objectives of the thesis are as follows:

- To study how to manage and share the files in a peer-to-peer file sharing system
- To save the memory storage by using the Download Agent instead of duplicate file downloading
- To reduce the huge workload and to allocate the download tasks to be available peers within the same LAN by using the Load Balancing agent
- To understand how to work the agents with the aid of the multi-agent technology

## 1.2 Related Works

Ozalp Babaoglu, Hein Meling and Alberto Montresor developed "Anthill: A Framework for the Development of Agent-Based Peer-to-Peer Systems" [19]. Anthill is a system to help the plan, execution and assessment of P2P applications. It comprises of a powerful organization of friend hubs; social orders of versatile agents go through this organization, connecting with hubs and helping out different agents to tackle complex issues. It can be utilized to build various classes of P2P administrations and display strength, transformation and self-association properties. It can be also utilized for the acknowledgment of distributed applications by giving a JXTA based network foundation.

M. Grivas and S. J. Turner proposed "Agent Technology in Load Balancing for Network Applications" [15], the agent advancement in load adjusting to arrange the network applications. This paper consolidates gathering and executions that using programming experts to accomplish load changing in the Web programs. Conveyed enrolling described the issues and one more field of assessment for load changing. Many kinds of assessment proposed different techniques to stack changing in the scattered system yet researcher uses agent's base response for load changes because of agents have new properties for handling the problem in this workspace.

Budditha Hettige, Asoka Karunananda, Kathriarachchi and Weerasinghe described "ITray, Multi-agent solution for LAN based file sharing" [7]. ITray is designed as a multi-agent system using the MaSMT, a Java-based framework. Agents

communicate with each other via Java socket.

By utilizing load balancing component, agent reduces the own load and use tasks scheduling to allocate the task to available resources in the network. Utilizing these assets can be shared more effectively in the dynamic and distributed environment. The core system consists of an XML-based ontology. The MaSMT is a fully tested under the actual environment and it demonstrates the way that resources can share perfectly within the network. Furthermore, the research can be further developed by sharing handling power within the network as per the requirements of the network.

Chongjie Zhang, Victor Lesser and Prashant Shenoy proposed "A multi-agent learning approach to resource sharing across computing clusters" [8]. This system is a mediator-free multi-agent information retrieval system and use it for improving the online resource allocation in cluster networks. It provides some context-sensitive searching algorithms based on the various topologies of peer-to-peer networks. To develop a shared computing structure, the traditional model is to establish a set of shared clusters into a network and allow resource sharing through shared clusters. The resource sharing consequence is now spreading to each shared cluster. Individual cluster still uses a cluster-wide technique for managing its local resources. Task allocation requirements vary across clusters, a cluster may need to dynamically choice what tasks are allocated locally and wherever to direct unallocated tasks to compassionately improve the over-all effectiveness of the whole system.

## 1.3 Overview of the Thesis

In this system, how to work file sharing, downloading and load balancing on peer-to-peer network is implemented. One or more multiple files can send simultaneously via File Send Agent without interruption. By the Download Agent, save storage space of the system and avoid duplicate files. Downloading tasks can allocate by Load Balancing Agent. Moreover, how to cooperate, coordinate and negotiate in each agent with the aid of multi-agent technology.

## 1.4 Organization of the Thesis

This thesis is organized with five chapters. Chapter 1 includes the introduction, the objectives of the thesis, related works, system overview and thesis

organization. Chapter 2 expresses the background theory of the proposed system. The details of the Multi-Agent System are presented in Chapter 3. Chapter 4 briefly describes the design and implementation of the proposed system for file sharing and downloading. The last chapter, Chapter 5 concludes the conclusion, benefits of the proposed system, limitations and further extensions.

# CHAPTER 2

# THEORETICAL BACKGROUND

This chapter explains a few important topics to provide some background knowledge in peer-to-peer file sharing system. Some topic is originated the field of file sharing system and multi-agent system. Section 2.1 provides an overview of the network file sharing. Peer-to-Peer (P2P) file sharing are presented in Section 2.2 and the advantages of P2P network are presented in Section 2.3.

## 2.1 Network File Sharing

Network file sharing is the strategy associated with duplicating data, information, documents starting with one PC, then onto the next utilizing a live network association. There are many types of file sharing such as File Transfer Protocol (FTP) programs, removable storage devices, online file sharing services, file sharing tools and websites, etc.

## 2.2 Peer-to-Peer (P2P) File Sharing

Peer-to-peer is a methodology for network file sharing that permits clients to straightforwardly admittance to different clients to share documents. A client-server network is an association between a client PC and a server PC to furnish the client with the server's assets. Figure 2.1 describes the client-server and peer-to-peer network architecture. P2P network is different from a client-server network.

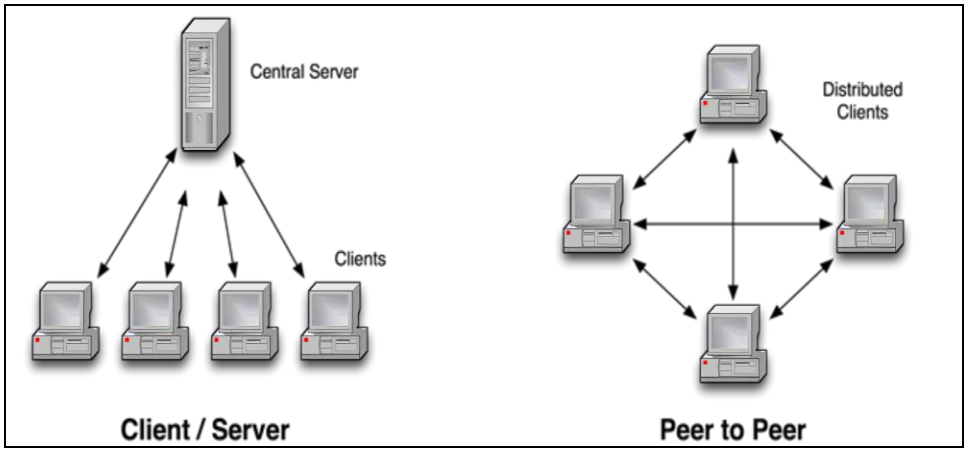



**Figure 2.1 Client/Server and Peer-to-Peer Network Architecture**

In a P2P network, files can be shared directly without the need of a central server. P2P technique is more powerful than client-server strategy in light of the fact that the PCs have shared liabilities to communicate with one another. P2P file sharing system utilize no focal servers with the exception of rather permit all PCs on the network to work both as a client and a server.

Every PC in a P2P network gives assets to the organization and consumes assets that the organization gives. Assets, for example, records, capacity, data transmission and handling power can be divided among different PCs in the organization.

A P2P network is not difficult to design. Whenever it's set up, access is constrained by setting sharing consents on every PC. Stricter access can be constrained by relegating passwords to explicit assets.

Some P2P networks are framed by overlaying a virtual organization on an actual organization. The organization utilizes the actual association with move information while the virtual overlay permits the PCs on the organization to speak with one another.

## 2.3 The Advantages of Peer-to-Peer Network

There are several advantages of peer-to-peer networks.

- Low latency – With low dormancy come better reaction times and more limited holding up times between demands. The length of the association way that is between peers is decreased, making the organization more effective by wiping out excess strides on the way towards the last objective.
- High bandwidth – A peer-to-peer network is given high transmission capacity, so there is no requirement for focal servers to arrangement assets. This makes it conceivable to give data to countless clients simultaneously without influencing the presentation of the organization.
- Low cost – Because of the way that there is no focal server in a shared organization, each friend is liable for putting away and sending the mentioned data. There are no expenses charged by the server that is facilitating the application.
- High security – A peer-to-peer network has no weak link. In the event that one

hub goes disconnected or becomes inaccessible, the organization is as yet ready to work.

- Decentralization – In a client-server design, the organization that possesses the server controls its clients. It can screen client action and erase data. That isn't the situation with regards to distributed networks, which let clients control their own information.
- Fault tolerance – P2P networks are exceptionally versatile to deficiencies. At the point when one hub goes down, different hubs will cover for this hub and keep the organization running.

# CHAPTER 3

# MULTI-AGENT SYSTEM

This chapter presents some general background and history on Artificial Intelligence (AI), artificial intelligent agents, multi-agent systems and distributed systems. Then, this section will discuss more specifically the details and history of distributed intelligent agents and of virtual environments. Section 3.1 presents agent in artificial intelligence. The nature of task environment describes in Section 3.2. Section 3.3 explains about the intelligent agents and their properties. Section 3.4 briefly describes the Multi-Agent System.

## 3.1 Agent in Artificial Intelligence

There is no generally acknowledged the meaning of the idea of the agent. Be that as it may, the accompanying four properties are broadly acknowledged to characterize agents: independence, reactivity, social capacity and favorable to liveliness. However, the four following properties are widely accepted to characterize agents: autonomy, reactivity, social capacity and pro-activeness.

Agents are independent computational substances (independence), which communicate with their current circumstance (reactivity) and different agents (social capacity) to accomplish their own objectives (supportive of movement). Agents are equipped for connection with different agents by correspondence, exchange and coordination. Agent is whatever can be seen as seeing its current circumstance through sensors and following up on its environment through actuators.

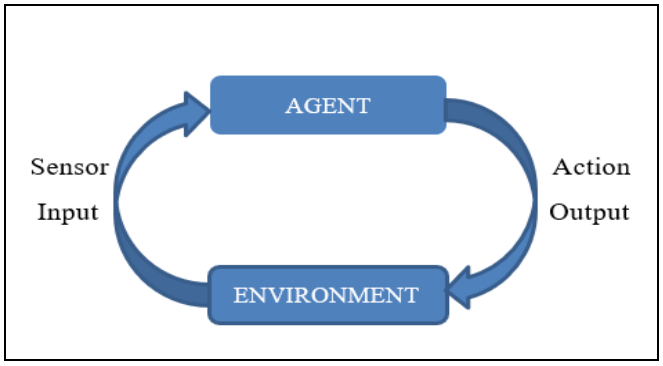



**Figure 3.1 An agent in its environment**

Figure 3.1 gives an agent in its environment. The agent takes sensory input from the environment and produces as output actions that affect it. The interaction is usually an ongoing, non-terminating one.

A human agent has eyes, ears, and other organs for sensors and hands, legs, mouth and other body parts for actuators. A robotic agent might have cameras and infrared range finders for sensors and various motors for actuators. A software agent receives keystrokes, file contents and network packets as sensory inputs and acts on the environment by displaying on the screen, writing files and sending network packets.

The agent program carries out the specialist capability planning percepts to activity. The program will run on a processing gadget with actual sensors and actuators of some kind. The agent function is a theoretical numerical depiction and the agent program is a substantial execution, running on the specialist engineering. The agent function describes the agent's behavior maps any given percept grouping to an activity. Agent program plans rely upon the idea of the climate and differ in effectiveness, minimization and adaptability.

## 3.2 The Nature of Environments

Task environment is the environment in which agent performs its task. Its detail incorporates Performance Measure, Environment, Actuators and Sensor (PEAS). In planning an agent, the initial step should constantly be to determine the undertaking environment as completely as could really be expected.

| Agent Type | Performance Measure | Environment | Actuators | Sensors |
|---|---|---|---|---|
| Taxi driver | Safe, fast, legal, comfortable trip, maximize profits | Roads, pedestrians, other traffic, customers | Steering, accelerator, brake, signal, horn, display | Cameras, sonar, speedometer, GPS, odometer, accelerometer, engine sensor |
| Vacuum Cleaner | Cleanliness, Efficiency, Battery Life, Security | Hall, Chairs, Tables, Carpets, Other Obstacles | Wheels, Vacuum Extractor, Brushes | Camera, Dirt Detector, Bump Sensor, Cliff Sensor |
| Interactive English Tutor | Maximize student's score on test | Set of students, testing agency | Display exercises, suggestions, corrections | Keyboard entry |
| Medical diagnosis system | Healthy patient, minimize costs, lawsuits | Patient, Hospital, Staff | Display questions, tests, diagnoses, treatments, referrals | Keyboard entry of symptoms, findings, patient's answers |

**Table 3.1 Examples of agent types and their PEAS descriptions**

## 3.2.1 Properties of Task Environments

Task environment shifts along a few critical aspects. They can be fully or partially observable, deterministic or stochastic, episodic or sequential, static or dynamic, discrete or continuous and single-agent or multi-agent.

**Fully observable vs. Partially observable**

In the event that an agent sensor can detect or get to the total condition of a climate at each mark of time, it is a completely noticeable climate, else it is somewhat perceptible. A completely perceptible climate is simple as there is compelling reason need to keep up with the inside state to keep track history of the world. An agent without any sensors in all conditions then such a climate is called as undetectable.

**Deterministic vs. Stochastic**

Assuming the following condition of the climate not entirely set in stone by the present status and the activity executed by the agent, then the climate is deterministic, in any case it is stochastic. Assuming the climate is deterministic aside from the activities of different specialists, that climate is vital.

**Episodic vs. Sequential**

Agent's experience is separated into nuclear episodes. Every episode comprises of the specialist seeing and afterward playing out a solitary activity. In verbose conditions, just the ongoing discernment is expected for the activity. Each episode is free of one another. In consecutive conditions, the ongoing choice could influence every single future choice. An agent requires memory of past activities to decide the following best activities.

**Static vs. Dynamic**

Static environment doesn't change while a specialist is acting. Dynamic environment is continuously changing over the long haul. In the event that the actual environment is not changed with the progression of time however the agent's presentation score does, then the environment is semi-dynamic.

**Discrete vs. Continuous**

On the off chance that there are a limited number of particular states, obviously characterized percepts and activities, the climate is discrete. The climate wherein the activities performed can't be numbered is supposed to be nonstop.

**Single Agent vs. Multi Agent**

If by some stroke of good luck one specialist is associated with a climate, and working without anyone else then such a climate is called single specialist climate. On the off chance that numerous specialists are working in a climate, such a climate is called multi specialist climate.

| Task Environment | Observable | Deterministic | Episodic | Static | Discrete | Agent |
|---|---|---|---|---|---|---|
| Crossword Puzzle | Fully | Deterministic | Sequential | Static | Discrete | Single |
| Chess | Fully | Strategic | Sequential | Semi | Discrete | Multi |
| Poker | Partially | Strategic | Sequential | Static | Discrete | Multi |
| Taxi driver | Partially | Stochastic | Sequential | Dynamic | Continuous | Multi |
| Part-picking robot | Partially | Stochastic | Episodic | Dynamic | Continuous | Single |
| Interactive English Tutor | Partially | Stochastic | Sequential | Dynamic | Discrete | Multi |

**Table 3.2 Examples of task environments and their characteristics**

## 3.3 Intelligent Agent (IA)

Developing intelligent and autonomous agents is a central goal of Artificial Intelligence. An intelligent agent (IA) is an independent element which act coordinating its action towards achieving goals, upon an environment utilizing perception through sensors and resulting actuators. They may moreover learn or use data to achieve their goals.

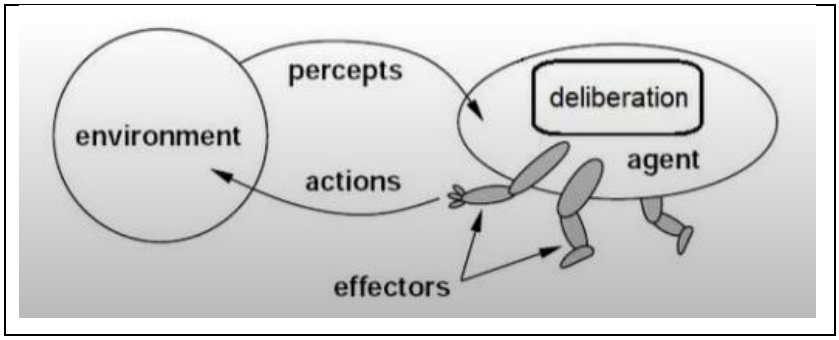



**Figure 3.2 An Intelligent Agent (IA)**

A simple intelligent agent (IA) is shown in Figure 3.2. Intelligent agents have properties such as reactivity, pro-activity and social capacity.

Reactivity: Agents can see their current circumstance, and answer in an ideal style to changes that happen in it to fulfill their plan targets.

Pro-activity: Agents can show objective coordinated conduct by stepping up to fulfill their plan targets.

Social capacity: Agents are equipped for connecting with different specialists to fulfill their plan targets.

### 3.3.1 Types of Intelligent Agents

Simple reflex agents, model-based reflex agents, goal-based agents and utility-based agents are four basic types of intelligent agents [4].

**Simple reflex agent**

These agents select activities based on the ongoing percept, overlooking the remainder of the percept history.

**Model-based reflex agent**

These agents keep up with some kind of inner express that relies upon the percept history, monitors the present status of the world involving an inward model and picks an activity similarly as the reflex specialist.

**Goal-based agent**

These agents need kind of objective data that portrays circumstances that are attractive and consolidate with data about the aftereffects of potential activities in the specialist program. Now and again objective based activity choice is direct and it will be interesting.

**Utility-based agent**

In the event that one world state is liked to another, it has higher utility for the specialist. A utility capability maps a state (or a succession of states) onto a genuine number, which depicts the related level of bliss. A total determination of the utility capability permits reasonable choices.

Simple reflex agents respond straightforwardly to percepts, though model-based reflex specialists keep up with inward state to follow parts of the world that are not obvious in the ongoing percept. Objective based specialists act to accomplish their objectives and utility-based specialists attempt to amplify their own normal satisfaction.

All agents can work on their exhibition through learning. A learning specialist can be partitioned into four theoretical parts: learning component, execution component, issue generator and prize.

## 3.4 Multi-Agent System (MAS)

Multi-Agent System (MAS) is an inexactly coupled organization of programming agents that collaborates to take care of issues that are past the singular limits or information on every issue solver. Multi-agent framework comprises of different specialists, which collaborate with each other by trading messages through PC network foundation.

A multi-agent system (MAS) is shown in Figure 3.3. The attributes of Multi-Agent System are that every agent has fragmented data or capacities for taking care of the issue, there is no framework worldwide control, information is decentralized and calculation is non concurrent. The characteristics of Multi-Agent System are shown in Figure 3.4. Each agent can collaborate, communicate, and negotiate with other agents. There is no framework worldwide control in MAS.

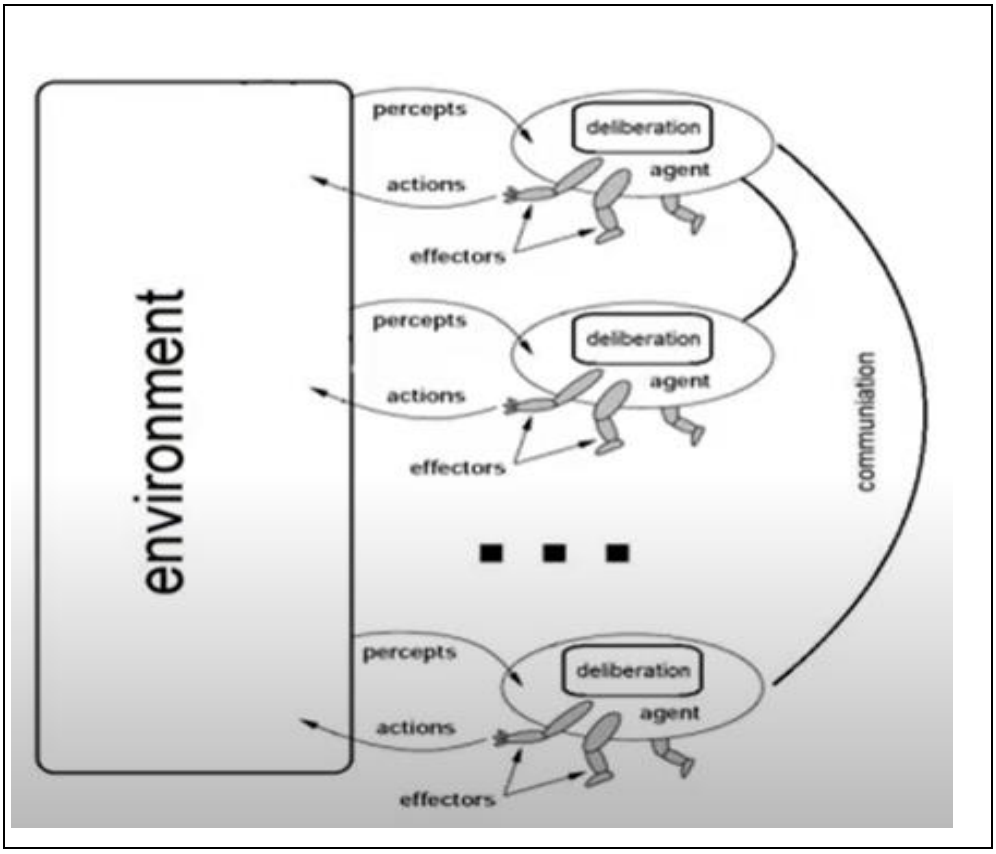



**Figure 3.3 A Multi-Agent System (MAS)**

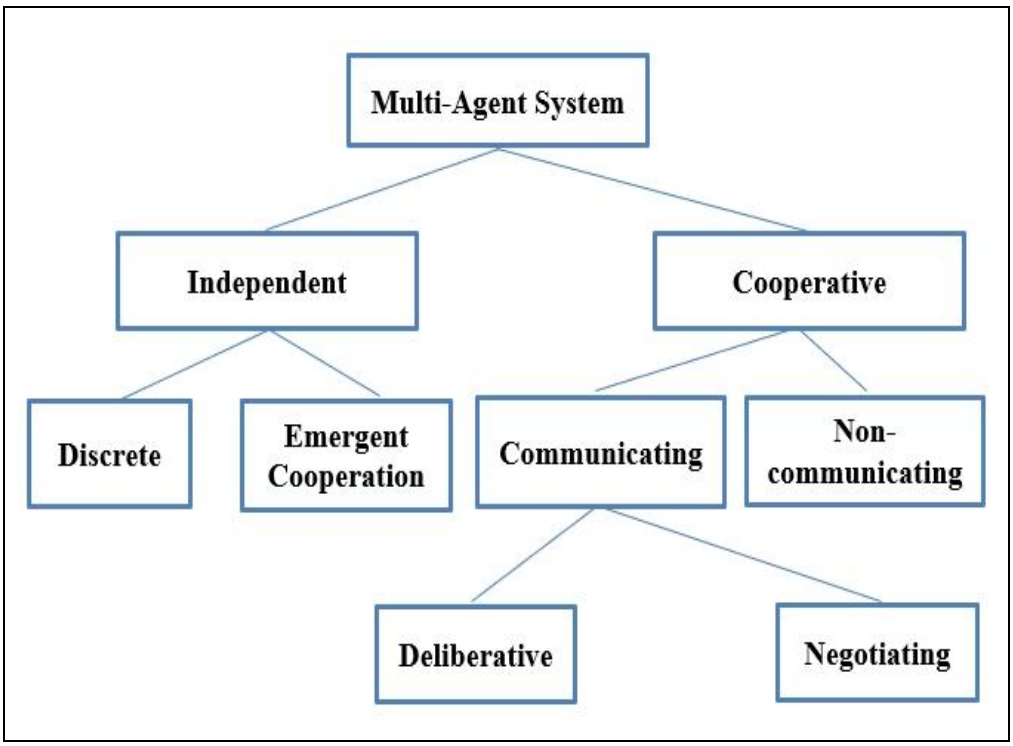



**Figure 3.4 The characteristics of Multi-Agent System**

### 3.4.1 Multi-Agent Systems Classification Dimensions

**Mobility**

Mobile agents can be resident in the source machine or temporarily in another one.

**Reasoning Model**

Presence or not of a type of symbolic reasoning.

**Agent Function**

The agent function assumed by the agent, such as information search agent or interface.

**Autonomy**

Agents can work with no immediate human or other agent's intercession. They have a command over their activities and inner condition of some sort or another, and they can trade data with different specialists.

**Cooperation**:

Acknowledgment of agreeable activities with different agents.

### 3.4.2 The Main Features of Multi-agent

- Autonomy: Act independently somewhat for the benefit of clients or different projects likewise by changing the manner by which they accomplish their targets.
- Pro-activity: Seek after their own singular put forth objectives, including by settling on choices as consequence of inner choices.
- Re-activity: React to external events and stimuli and consequently adapt their behavior and make decisions to carry out their tasks.
- Communication and Cooperation: Connect and speak with different agents and get guidelines and give reactions and coordinate to satisfy their own objectives.
- Negotiation: Carry out organized conversations to achieve a degree of cooperation with other agents.
- Learning: Improve performance and decision making over time when interacting with the external environment.

# CHAPTER 4
# SYSTEM DESIGN AND IMPLEMENTATION

This chapter briefly describes design and implementation of the multi-agent system. The proposed system is specially designed for file sharing, file downloading and load balancing within the peer-to-peer network.

## 4.1 Agent Architecture of the Proposed System

The agent architecture of the proposed system is shown in figure 4.1.This system has used three peers' network and share files among these three peers. The system is composed of a dynamic set of agent platforms connected through Internet. Each agent platform acts as a peer of the system and it is based on four kinds of agents: Manager Agent, File Send Agent, Download Agent and Load Balancing Agent. Peers can share the files using a live network connection within the same local area network.

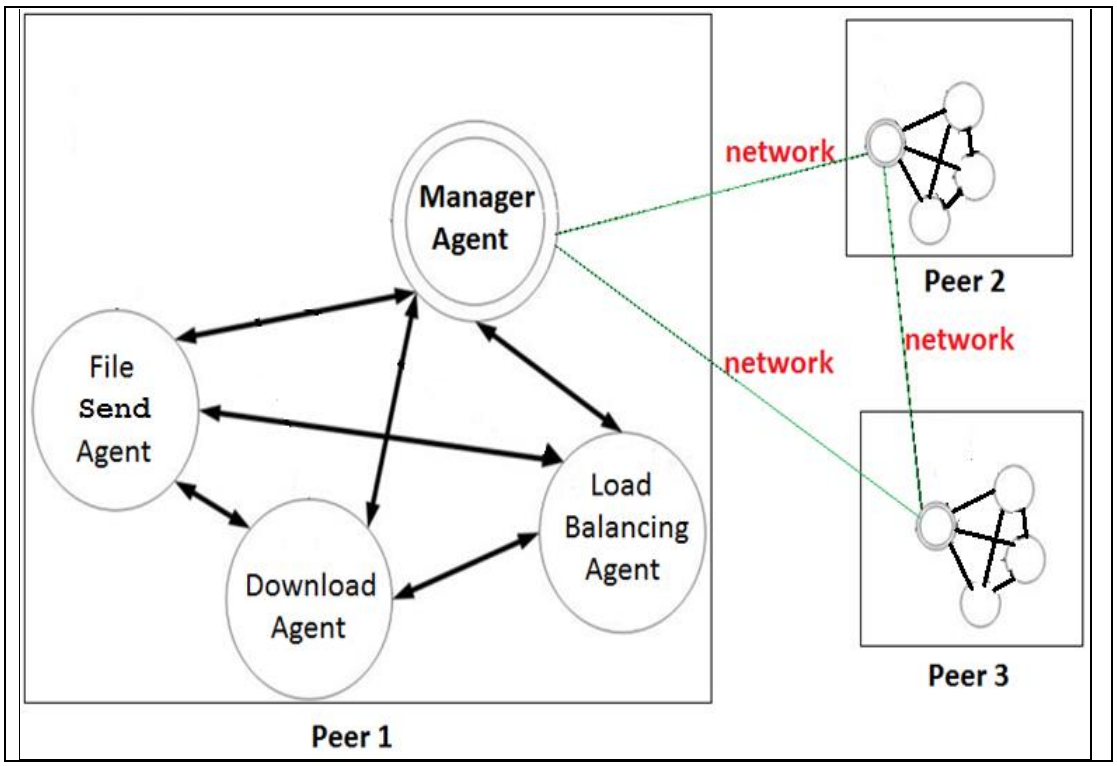



**Figure 4.1 Agent Architecture of the Proposed System**

### 4.1.1 Manager Agent

Manager Agent can contact with other Manager Agent in different computer devices inside the similar network. Manager Agent can invoke its ordinary agents to do the client's task by passing the message. Manager Agent manages the file inventory of the which can direct the incoming process to an appropriate agent.

### 4.1.2 File Sent Agent

Manager Agent summons File Send Agent by passing the directive to send records to the other peer or companions in a similar organization. That message contains the IP address of the beneficiary peer and file location which need to ship off the other machine by the client. As indicated by this message, File Send Agent gets the record area and send the file through an organization.

### 4.1.3 Download Agent

Manager Agent invokes Download Agent by passing the message when the client enters a URL (Uniform Resource Identifier) to the framework. That message contains the URL of the file to be downloaded which is entered by the client. Download Agent checks the URL whether or not that file is existed or not in the system and downloads in the event that file isn't existed. Subsequently, this structure can diminish memory wastage and can make an effort not to duplicate the copying file.

### 4.1.4 Load Balancing Agent

When the client enters a web URL, the framework sends that URL to any remaining peers in LAN. Other peer’s Manager Agent gets the URL and ensures that the file is accessible. Assuming that file is accessible, send that file by File Send Agent to the mentioned peer. In any case, the framework keeps a queue to store URL in a first in first out based mechanism. Manager Agent invokes Download Agent through communicating something specific to download that URL.

When the download queue is become increasing, Manager Agent invokes Load Balancing Agent to get shared file and to assign download tasks with other peers. Load Balancing Agent distribute the URL to any remaining accessible peers in the network

to download that URL and send back downloaded files to the mentioned peer. Other accessible peers' Manager Agent gets the URL and invokes their Load Balancing agent to really look at the responsibility of the framework.

At the point when the framework is free, that file is downloaded by Download Agent. Then Manager Agent invokes File Send Agent and sends that file to the mentioned peer which sends the URL. Subsequent to downloading the record or totally distribution undertaking to the next agent, the URL is consequently eliminated from download queue.

## 4.2 Downloading and Load Balancing

Load balancing is the mechanism that uniformly distributes data across servers on network resources, like disk drives, clusters, network links and more. The idea of load balancing is to use resources optimally, thereby reducing the response time, which in turn, improves performance and reliability.

A load balancer is essentially placed between the user and the web server so that the server is not overloaded with incoming requests. The load balancer is a product or equipment, gadget that holds any one server back from becoming over-burden.

### 4.2.1 Benefits of Load Balancing

- **High-availability**: While multiple servers are incorporated together, it supports accessibility. For instance, assuming one server becomes lethargic, the heap will be picked by other back-end servers that guarantees to answer the approaching traffic and keep the administrations unaffected.

- **Scalability**: Load balancing gives the ability to add more servers to the gathering to deal with the developing approaching solicitations. Rather than traveling to a totally new climate, you can basically build the quantity of burden balancers when required.

- **Flexibility**: Doing maintenance work is rather easy because administrators can direct all traffic to one server and place the other load balancer in active/passive mode. This allows them to do the maintenance without causing downtime issue. The approach can be used similarly for performing

maintenance work on the others load balancer, with at least one server up and running, which ensure to upkeep high-availability all the while.

### 4.2.2 Types of Load Balancing Algorithms

There are two types of load balancing algorithms: dynamic load balancing algorithms and static load balancing algorithms.

**Dynamic load balancing algorithms**

- **Least connection:** Checks which servers have the least associations open at that point and sends traffic to those servers. This expects all associations require generally equivalent handling power.

- **Weighted least connection:** Empowers heads to dole out various loads to every server, expecting that a few servers can deal with additional associations than others.

- **Weighted response time:** Midpoints the reaction season of every server, and joins that with the quantity of associations every server has open to figure out where to send traffic. By sending traffic to the servers with the speediest reaction time, the calculation guarantees quicker administration for clients.

- **Resource-based:** Disseminates load in light of what assets every server has accessible at that point. Specific programming (called an "specialist") running on every server estimates that server's accessible CPU and memory, and the heap balancer questions the specialist prior to disseminating traffic to that server.

**Static load balancing algorithms**

- **Round robin:** Cooperative burden adjusting disseminates traffic to a rundown of servers in pivot utilizing the Domain Name System (DNS). A definitive nameserver will have a rundown of various A records for a space and gives an alternate one in light of each DNS question.

- **Weighted round robin:** Permits a director to allocate various loads to every server. Servers considered ready to deal with more traffic will get somewhat more. Weighting can be designed inside DNS records.

- **IP hash:** Joins approaching traffic's source and objective IP locations and utilizations a numerical capability to change over it into a hash. In light of the hash, the association is relegated to a particular server.

## 4.3 Flow Diagram of the System

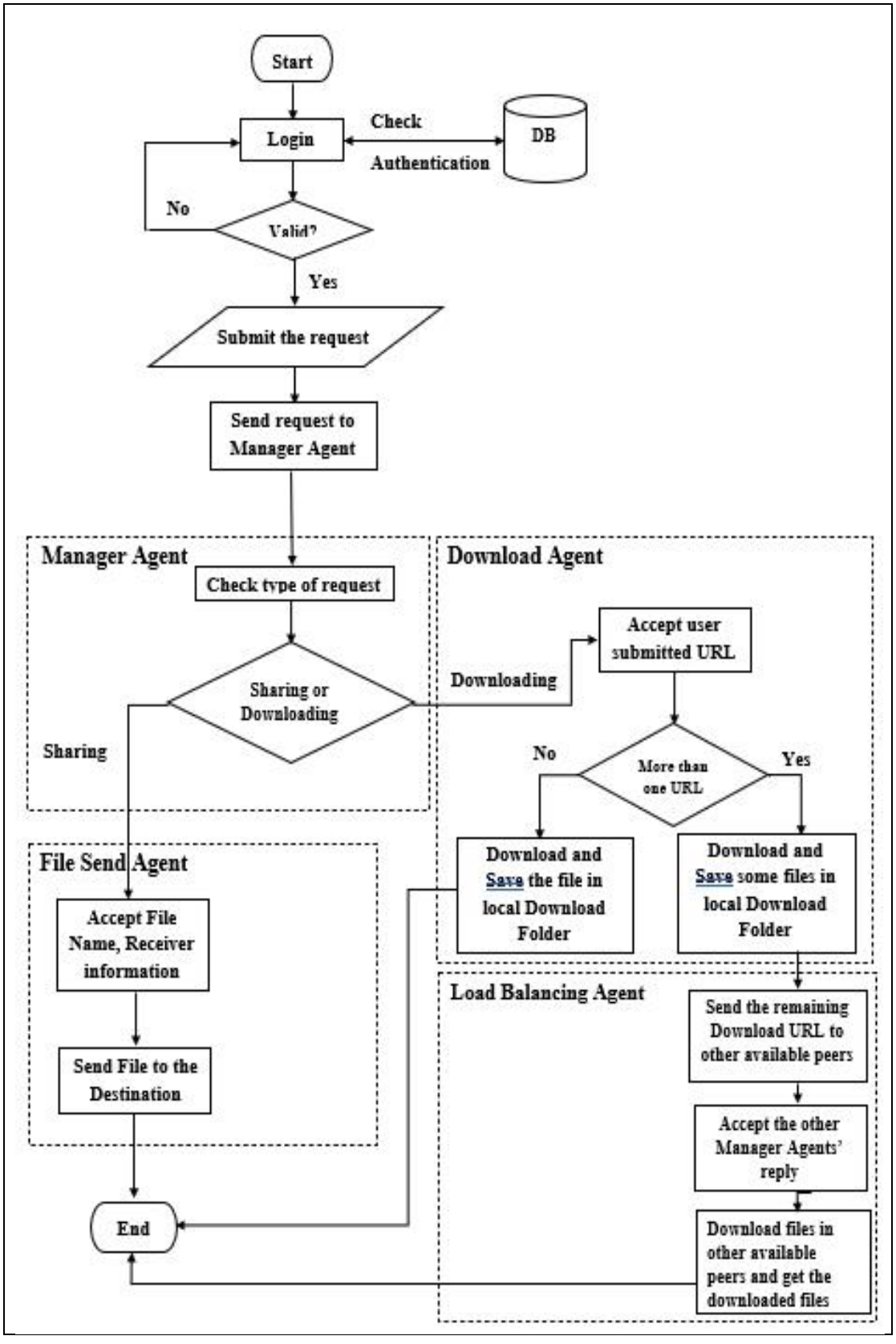



**Figure 4.2 Flow Diagram of the System**

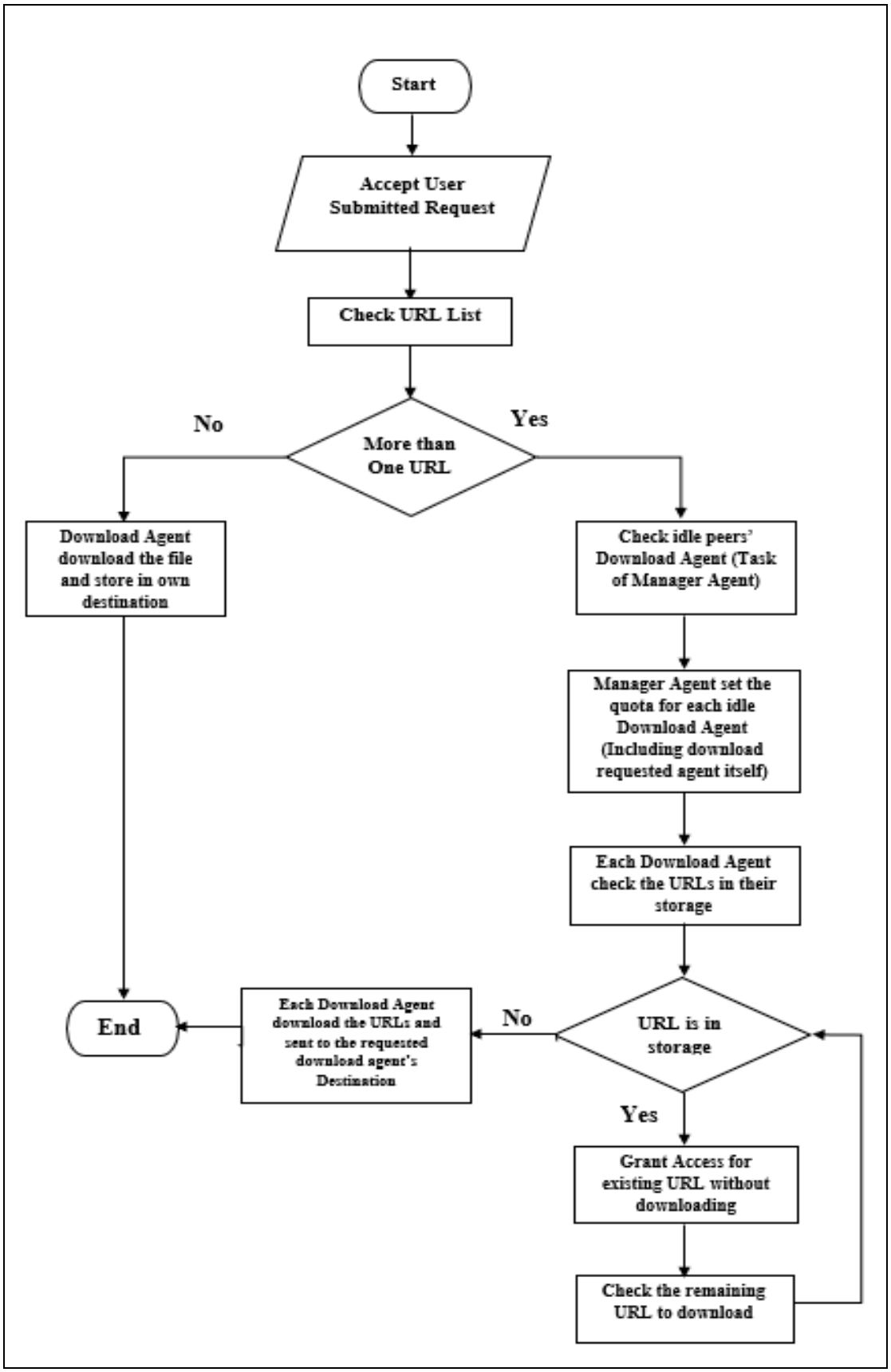



**Figure 4.3 Flow Diagram of the Load Balancing Agent**

## 4.4 Implementation of the System

The proposed system developed for the ease of file sharing in Wireless Local Area Network and load balancing for downloading process. This system is implemented by C#. Net Language on Microsoft Visual IDE and Microsoft SQL Server is used as the system database engine. The system designs and detail implementation are explained with three sections. Section 4.4.1 is the user authentication. Section 4.4.2 is the file sharing and section 4.4.3 is the file downloading.

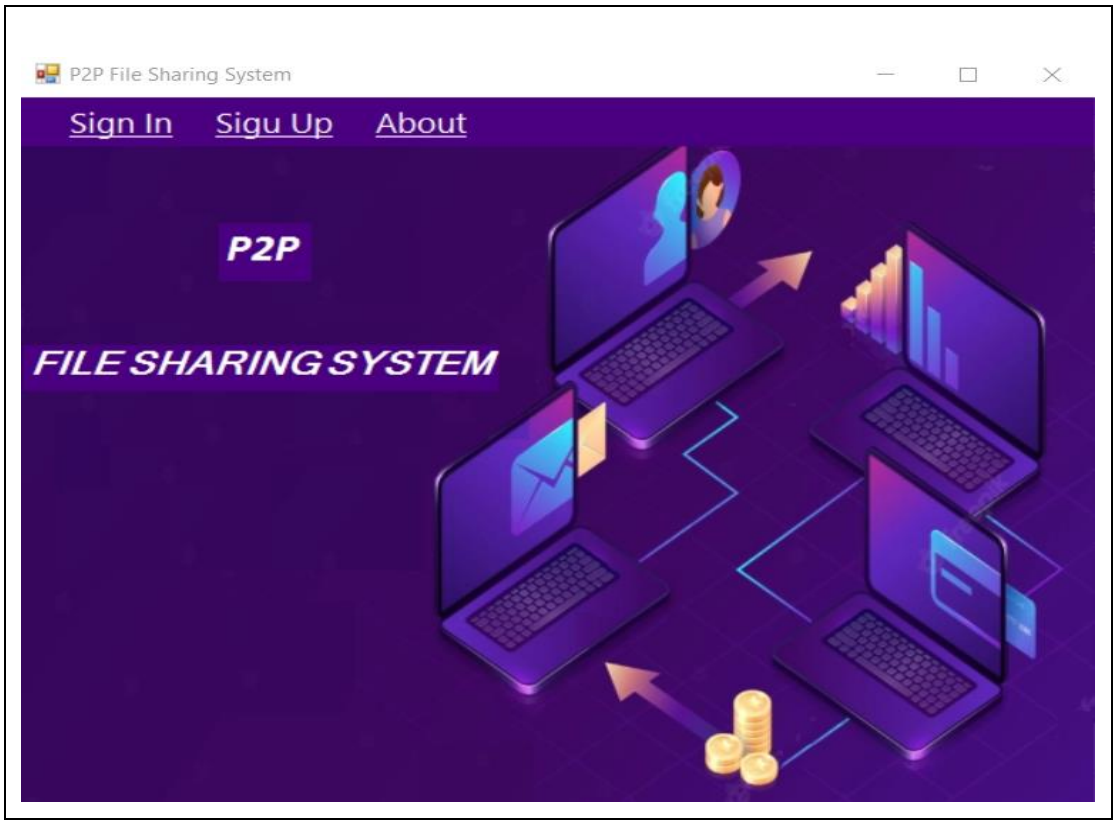



**Figure 4.4 Main Page of the Proposed System**

This system will start with the main page as shown in figure 4.4. The main page has three menus: “Sign In” menu, “Sign Up” menu and “About” menu.

### 4.4.1 User Authentication

Firstly, the user must register for accessing the files in the system. The registered user can be only used the system. The registered user can enter into the system via the login page. So, if the new user has not registered, the new user must be signed up.

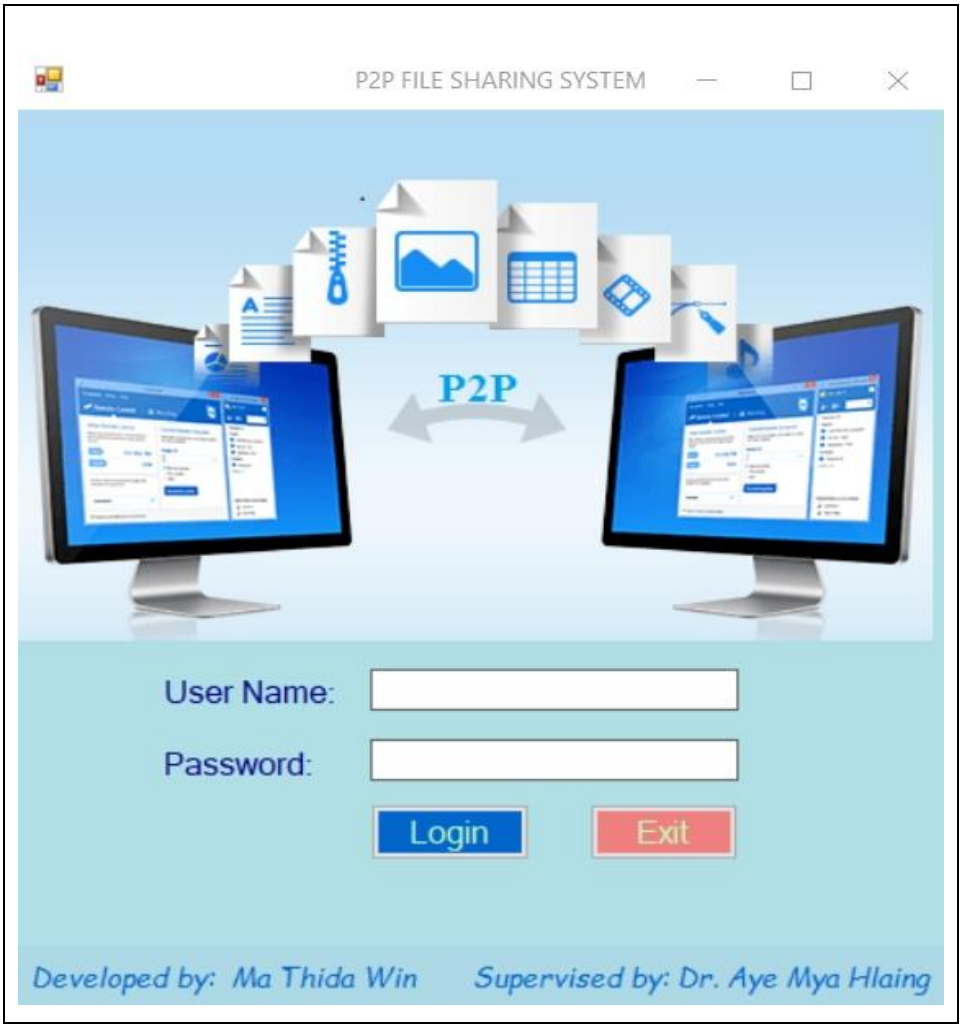



**Figure 4.5 The Login Page of the System**

The Login Page of the system is shown in figure 4.5 and Sign Up Page of the system are shown in figure 4.6. As shown in figure 4.5, this system allows the registered user. If the user is not registered in the system, the user must register in the Sign Up page. New user must be registered as requested personal information (such as User Name, Password, Re Enter Password, and Peer) as shown in figure 4.6.

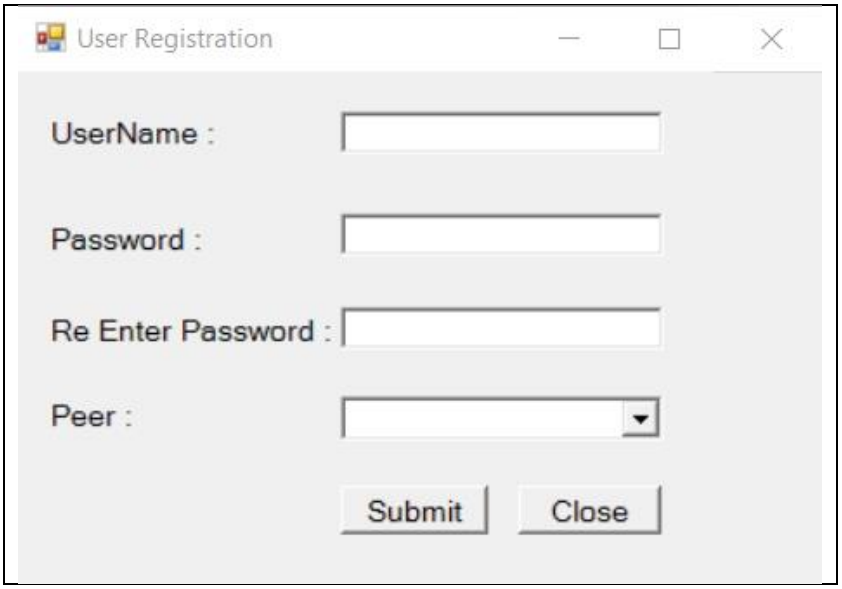



**Figure 4.6 Sign Up Page of the System**

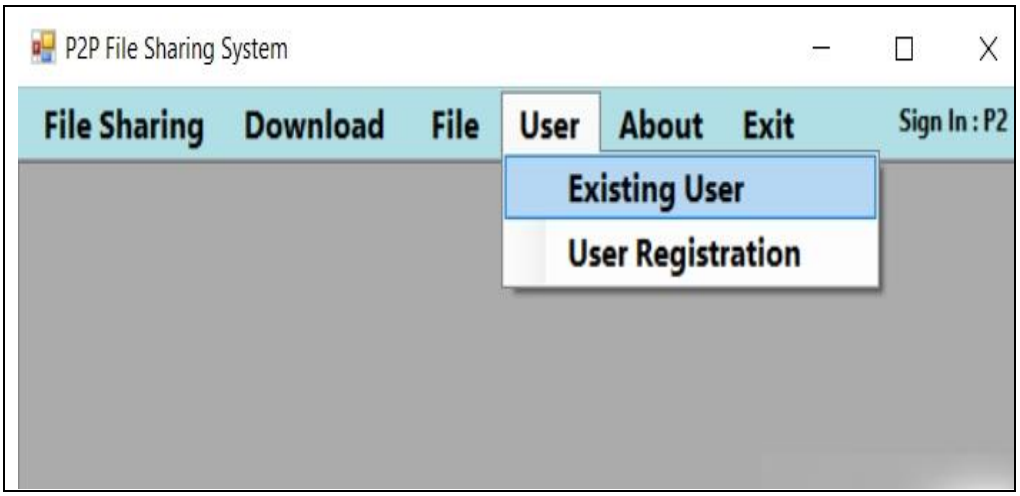



**Figure 4.7 Contents of User Menu**

The contents of "User" menu can allow the registered user as shown in figure 4.7. It contains two sub menus: "Existing User" sub menu and "User Registration" sub menu. The registered user can see the existing user lists and register the new user. "Existing User" sub menu can be seen as shown in figure 4.8. "User Registration" sub menu can be seen as shown in figure 4.6.

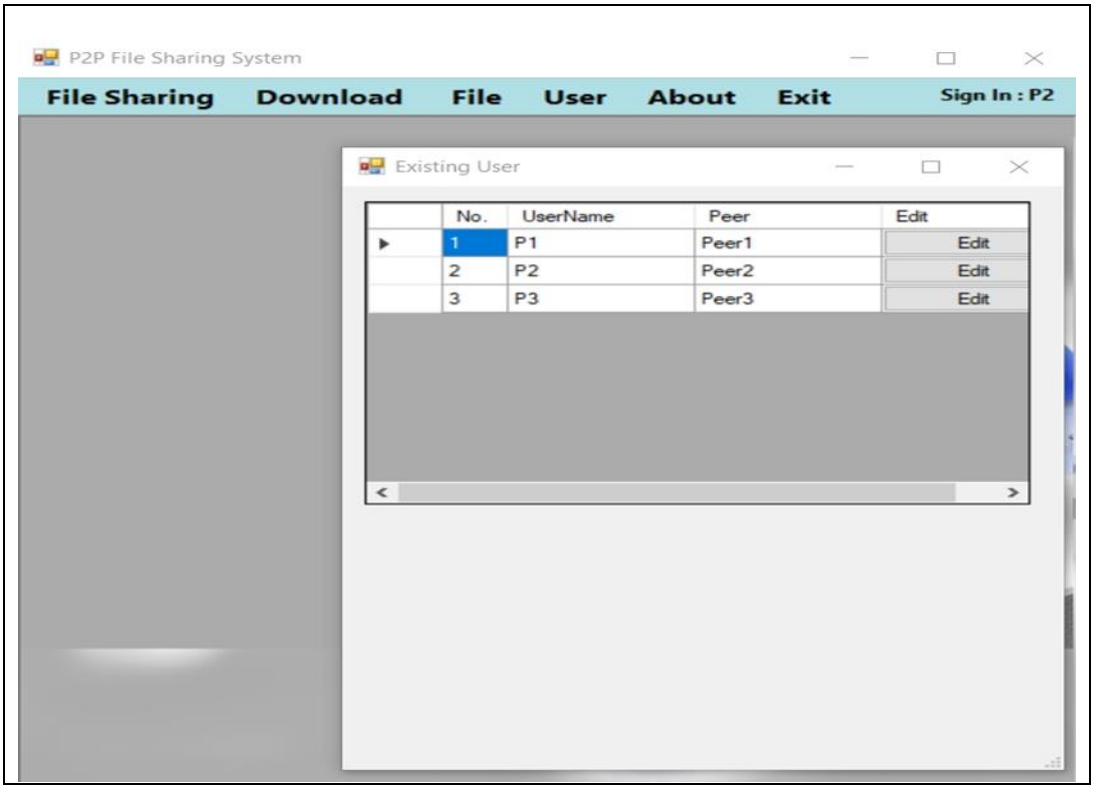



**Figure 4.8 Existing User List**

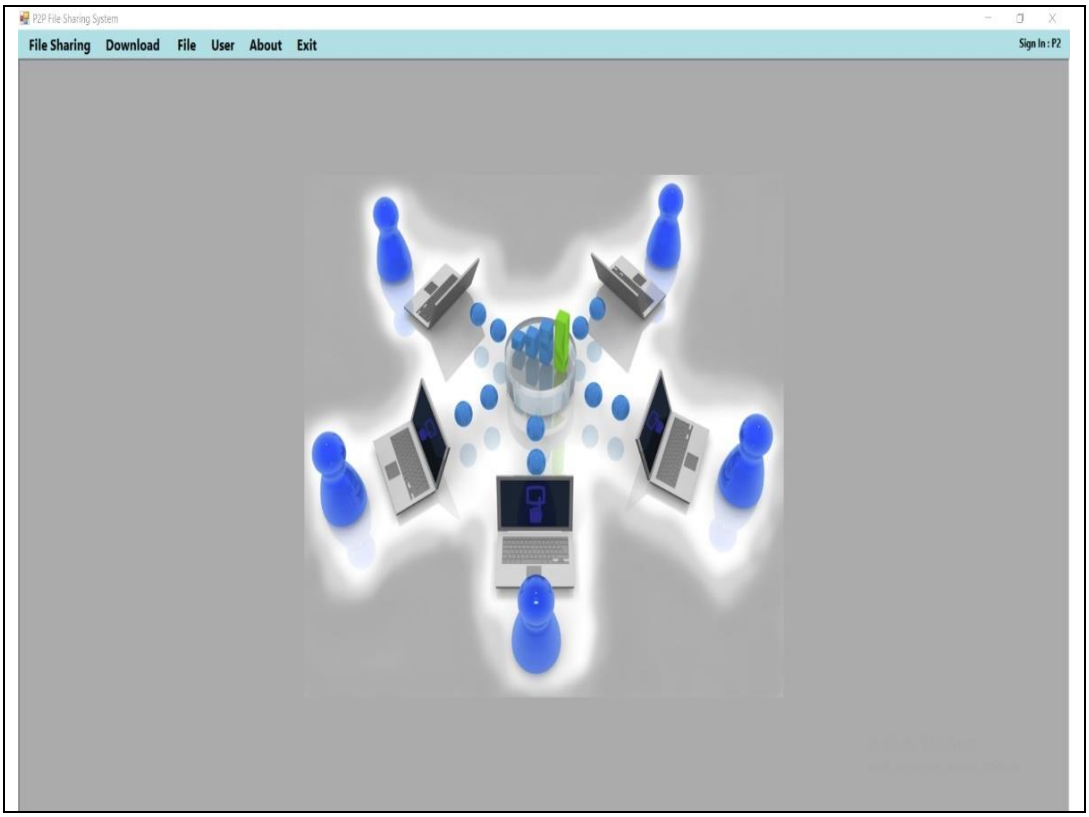



**Figure 4.9 The Process Page of System**

After the user login process is successful, the registered user can be reached the Process page of the system as shown in figure 4.9. The Process page has six menus: “File Sharing” menu, “Download” menu, “File” menu, “User” menu, “About” menu and “Exit” menu. The user can choose the file sharing or file downloading.

### 4.4.2 File Sharing

In this file sharing section, the user can send the files in the following processes: one file to one peer, one file to multiple peers, multiple files to one peer and multiple files to multiple peers. “File Sharing” menu can be seen in figure 4.10.

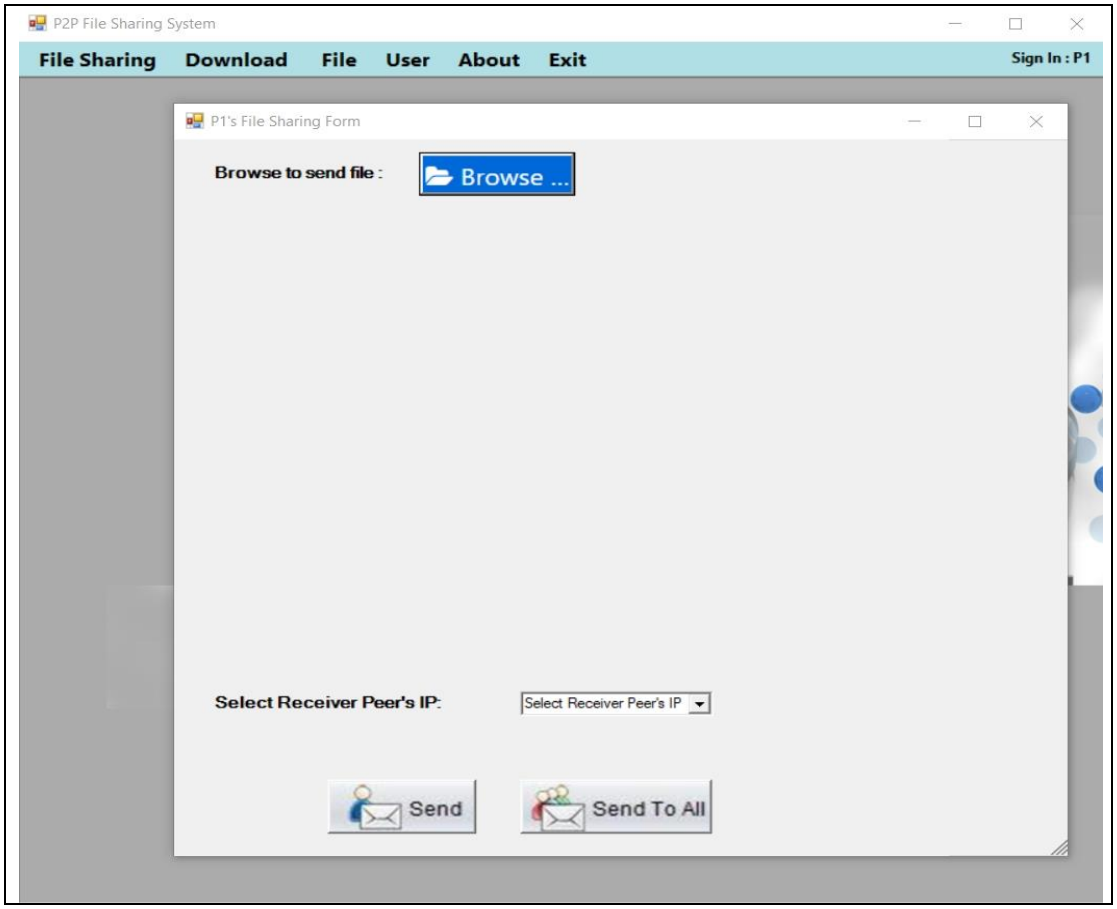



**Figure 4.10 File Sharing Page of System**

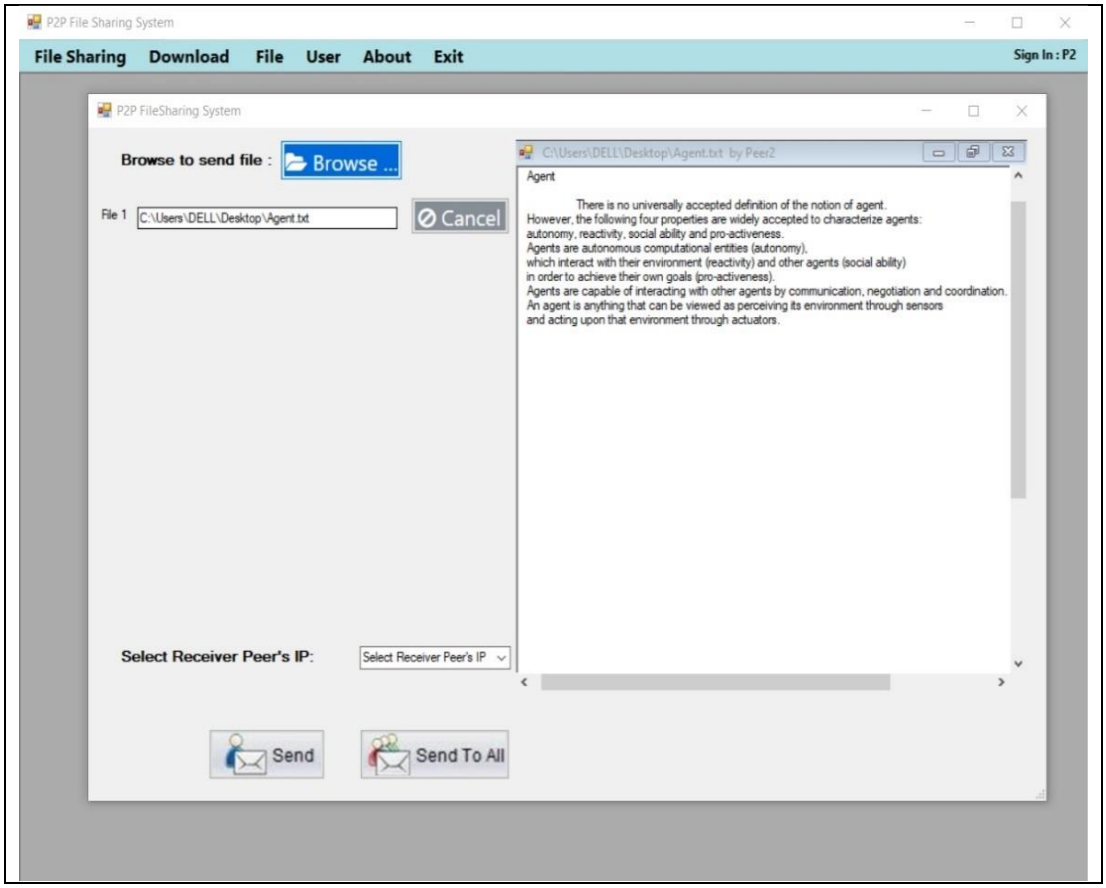



**Figure 4.11 Sending One File to One Peer**

Figure 4.11 shows file sending that the user send one file to one peer. Sending multiple files to one peer is described in figure 4.12. Sending multiple files to multiple peers is shown in figure 4.13.

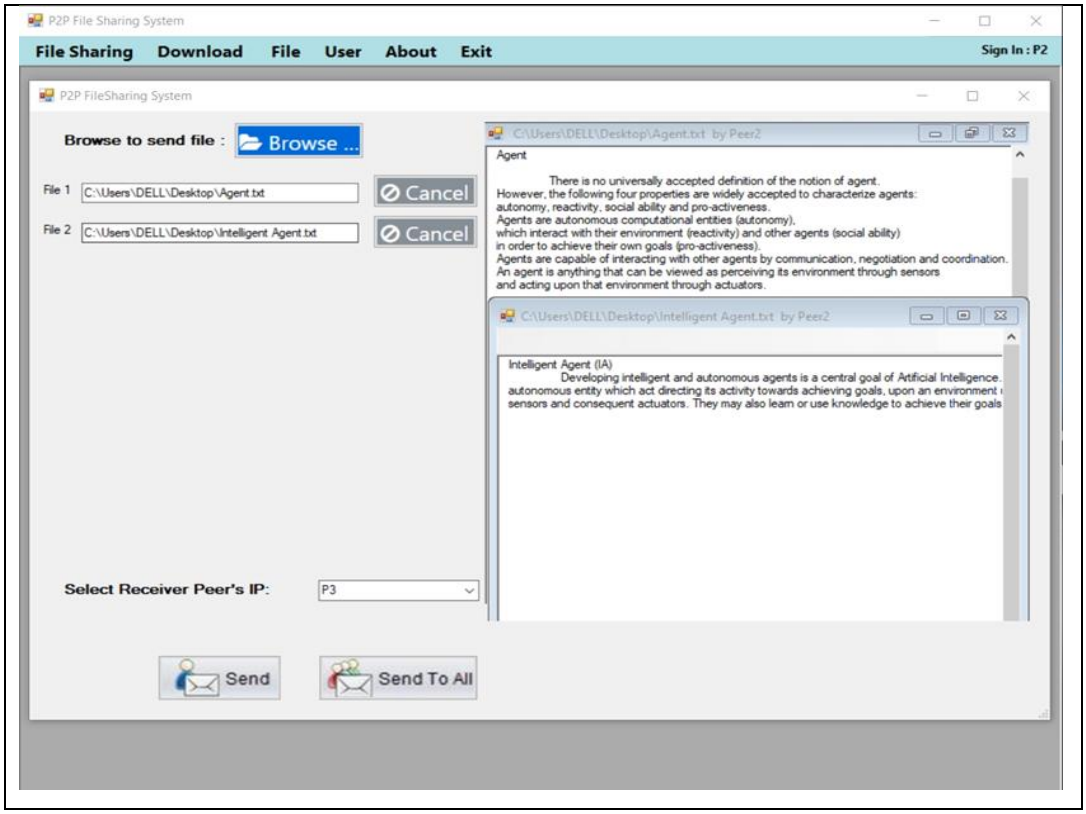



**Figure 4.12 Sending Multiple Files to One Peer**

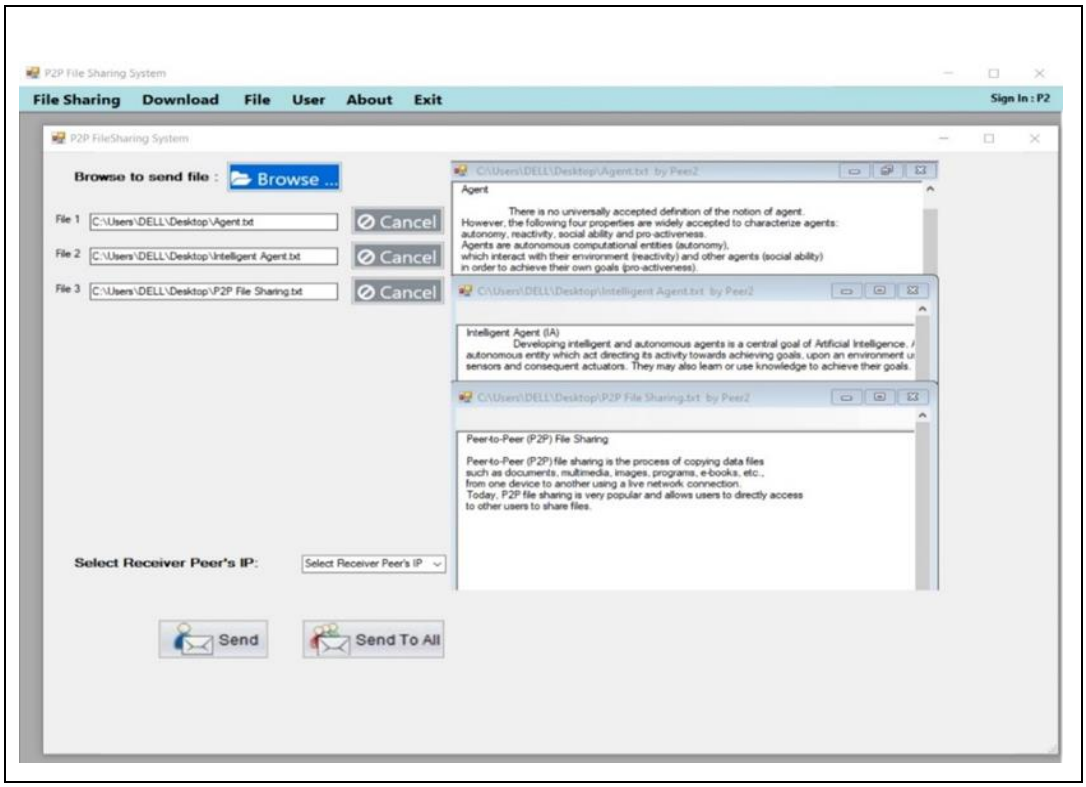



**Figure 4.13 Sending Multiple Files to Multiple Peers**

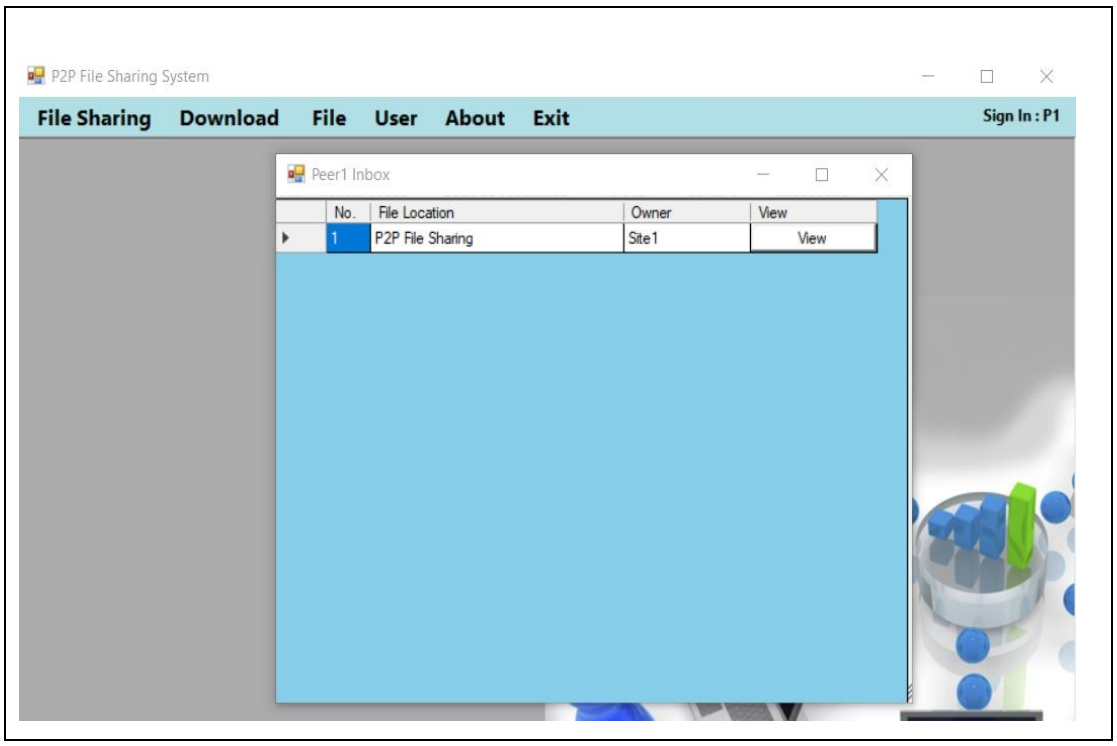



**Figure 4.14 Inbox / File Lists of Peer**

"File" menu can be seen in figure 4.14. This page can view the file lists which sent from other peers. "About" menu can be seen in figure 4.15.

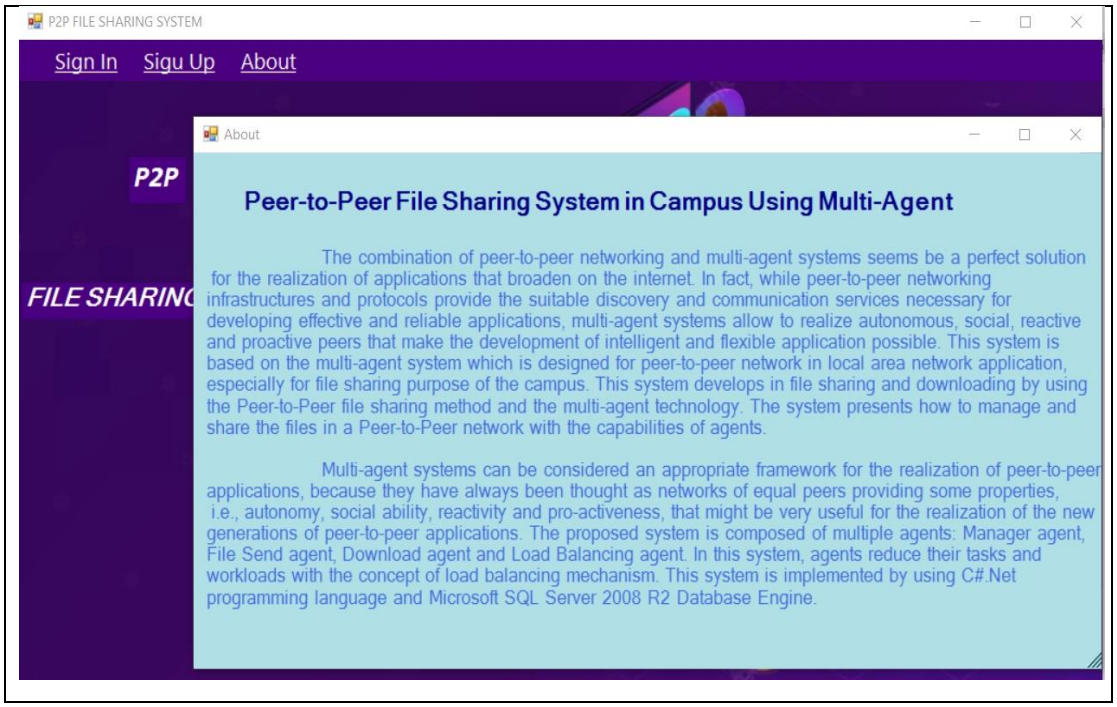



**Figure 4.15 About Page of the System**

### 4.4.3 File Downloading

In this file downloading section, the user can download single file or multiple files. If the user clicks the "Download" menu, the figure 4.16 will appear. The download section of the proposed system consists of three main menus: "Home", "Server's File Lists" and "Downloaded File Lists".

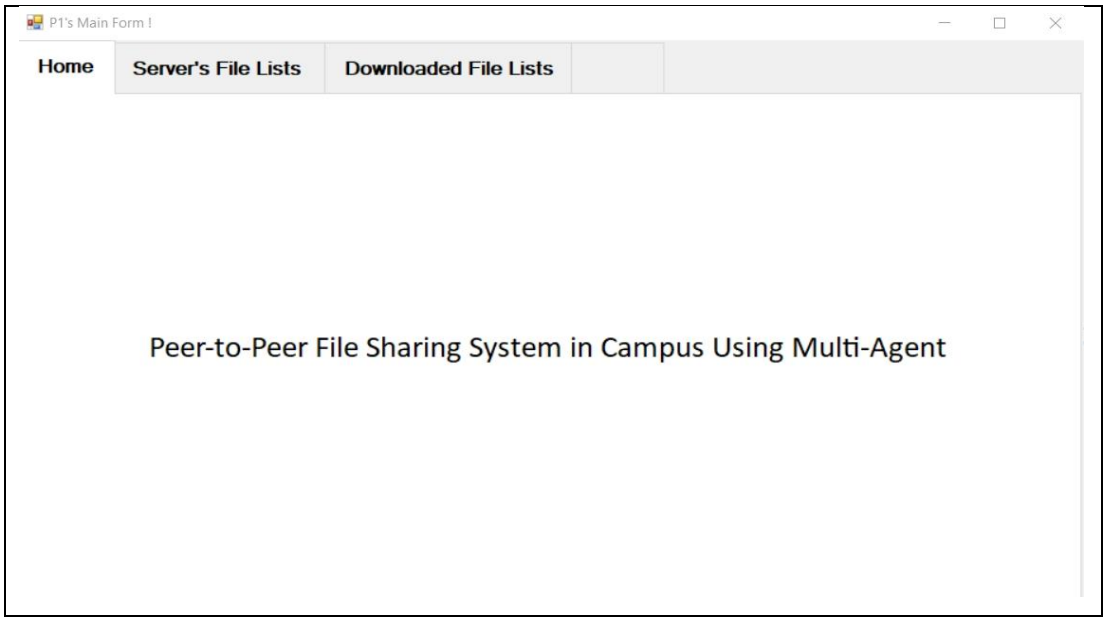



**Figure 4.16 File Downloading Page of the System**

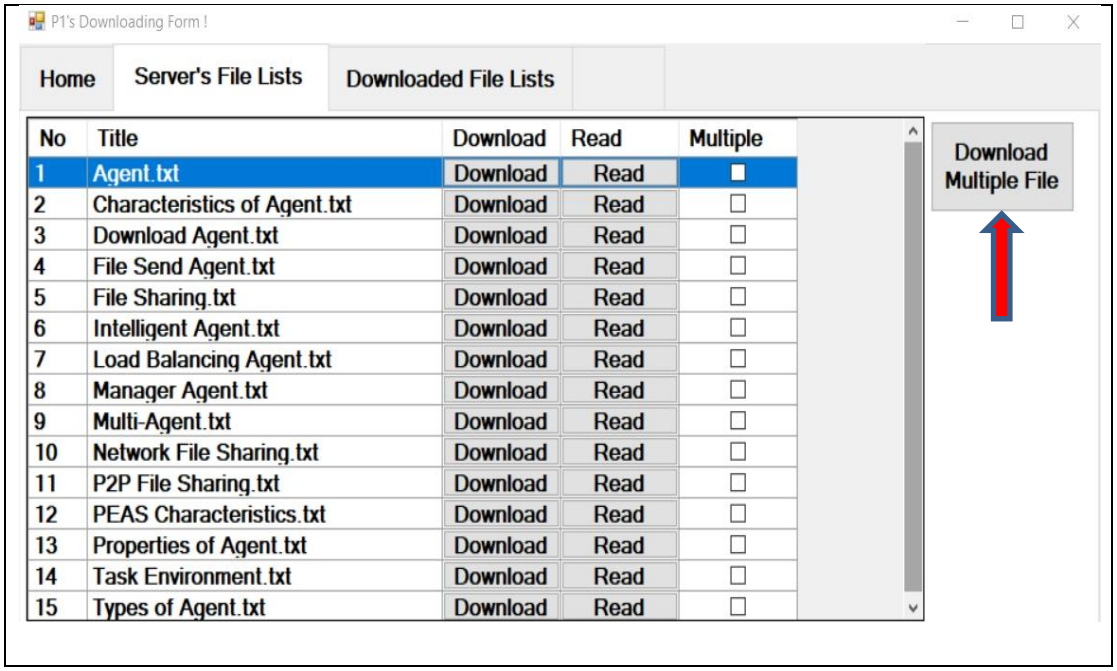



**Figure 4.17 File Lists for Downloading**

The storage file lists of the system for the file downloading are shown in above figure 4.17. This page also contains the property buttons: “Download”, “Read” and “Download Multiple File” for multiple files selection to download. Each “Download” button is used to down the respective file to the user storage cache. “Read” is to read the respective data. “Download Multiple File” button (navigate by red arrow) is to download the multiple files after the user selected in the “Multiple” check box column.

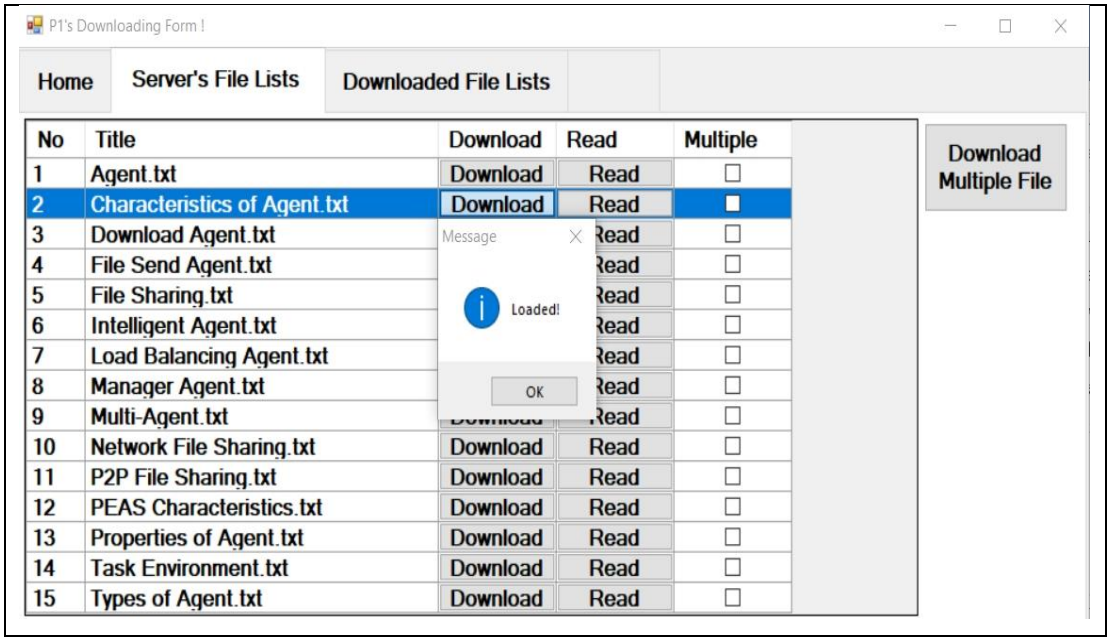



**Figure 4.18 Direct Downloading from Server**

When the user downloads the document from the original server, the system informs the download status to the user as shown in figure 4.18. The downloaded file lists are shown in 4.19.

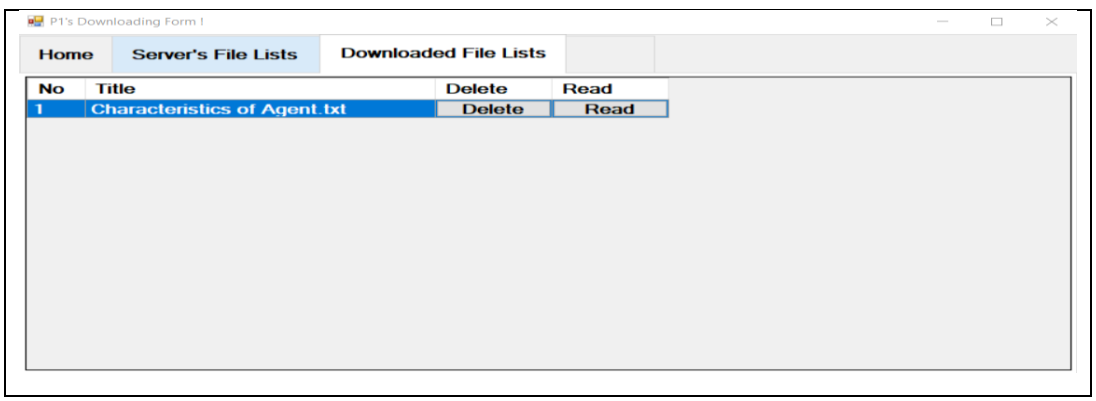



**Figure 4.19 The Downloaded File Lists of the user**

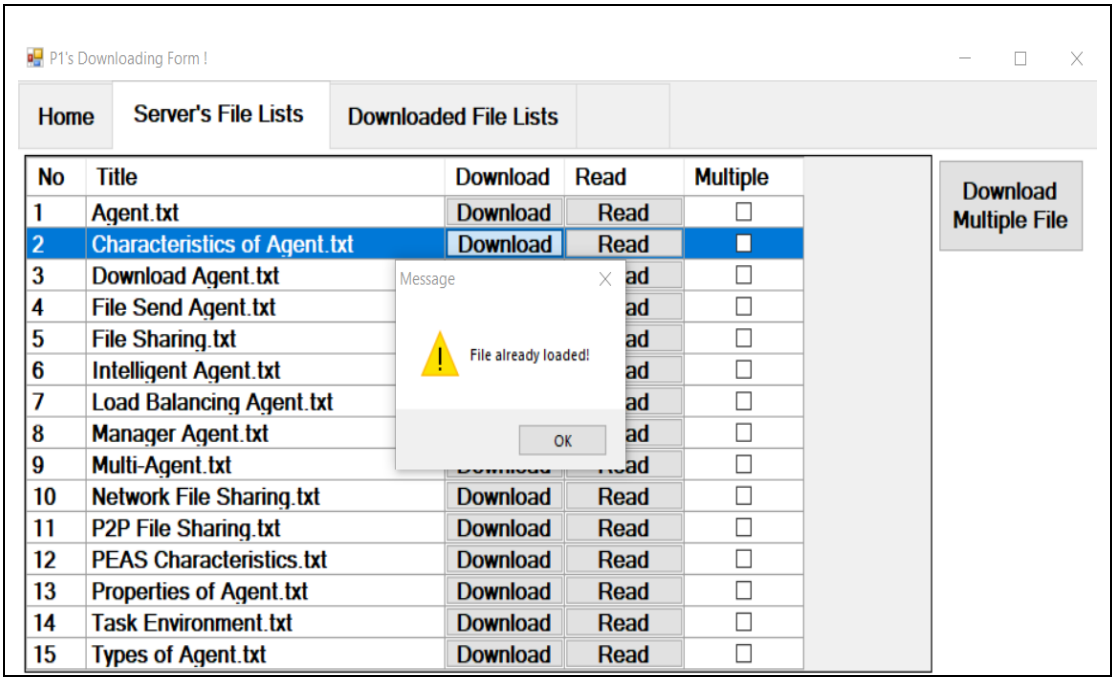



**Figure 4.20 Warning Message for Duplicate File Downloading**

If the user downloads the existing file in the user cache, warning message will appear. This proposed system can control the duplicate file downloading and can save the memory storage space.

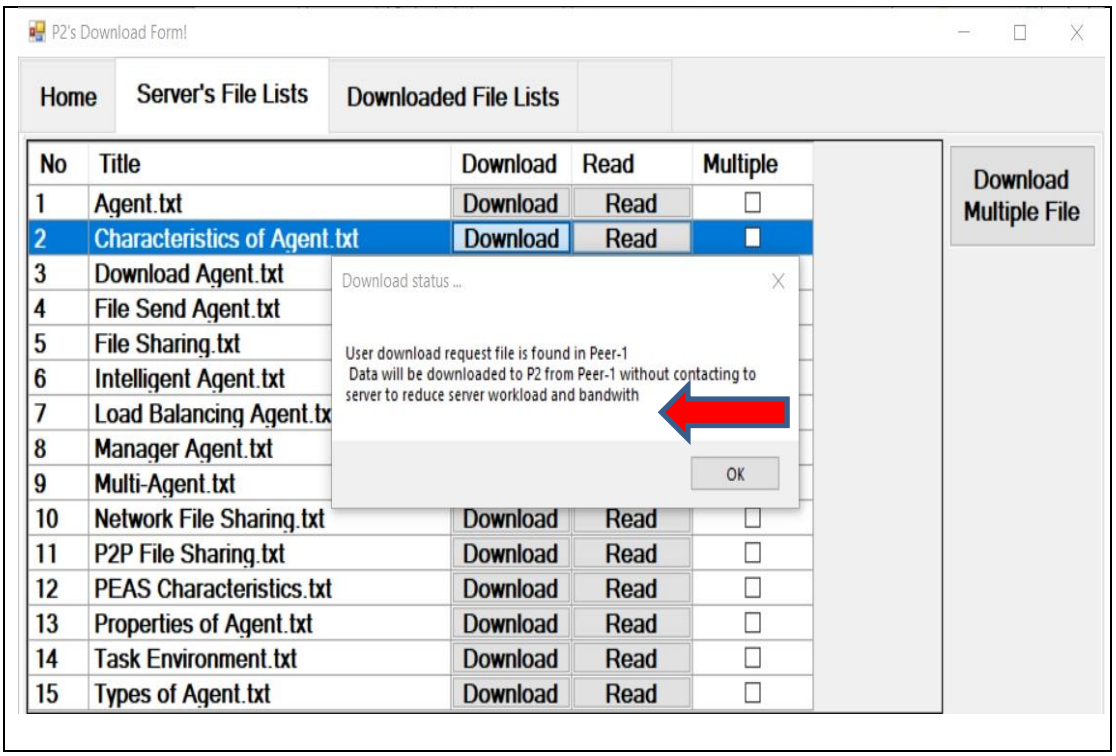



**Figure 4.21 Requested File is Found in Local Peer File Lists**

If the requested file is found in other peers' storage file lists, the information message will be sent to the user as shown in figure 4.21.

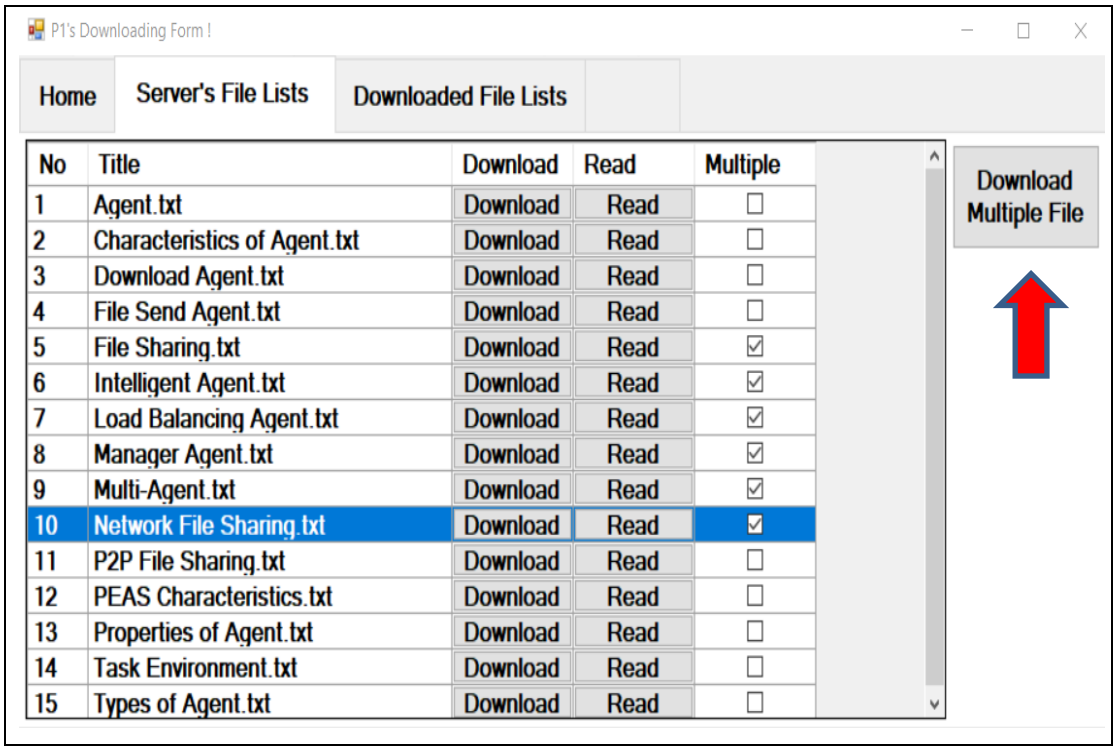



**Figure 4.22 Downloading Multiple Files with Load Balancing**

When the user wants to download the multiple files, the user must select the files in "Multiple" check box column and click "Download Multiple File" button. The process of downloading multiple files with the load balancing agent is shown in figure 4.22. The system will check the workload of the other peer. If the other peers are available, the download tasks will be allocated to other peers. Some files will be downloaded by the Download Agent of the available peers and sent back to the mentioned user. So, the system can reduce the huge workload of the peer and to allocate the download tasks to be available peers within the same LAN by using the Load Balancing agent

## 4.5. System Performance and Evaluation

As the performance of load balancing, the downloading of the text files is tested and results are shown in the below tables and figure. Downloading a single file without load balancing is shown in table 4.1 and downloading multiple files is shown in table 4.2. Their analysis of downloading a single file and multiple files is shown in figure 4.23 and 4.24.

| No. | Downloaded File | Size | Download Time Without Load Balancing (in second 's') |
|---|---|---|---|
| 1 | Test1.txt | 15 KB | 3 sec |
| 2 | Test2.txt | 17 KB | 3 sec |
| 3 | Test3.txt | 19 KB | 4 sec |
| 4 | Test4.txt | 14 KB | 2 sec |
| 5 | Test5.txt | 18 KB | 3 sec |
| 6 | Test6.txt | 15 KB | 3 sec |
| 7 | Test7.txt | 16 KB | 3 sec |
| 8 | Test8.txt | 14 KB | 2 sec |
| 9 | Total | 128 KB | 23 sec |

**Table 4.1 Downloading Single File without Load Balancing**

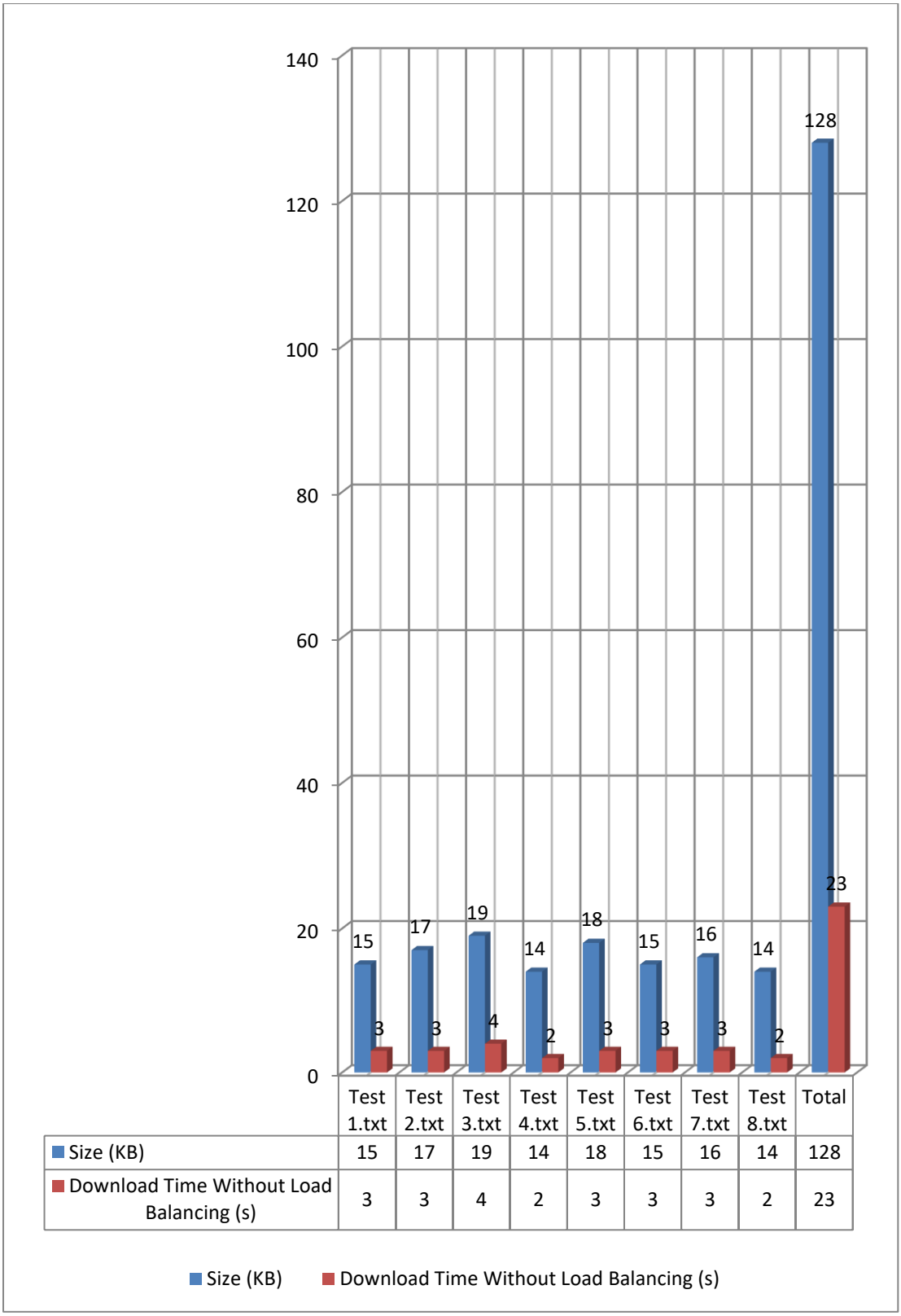
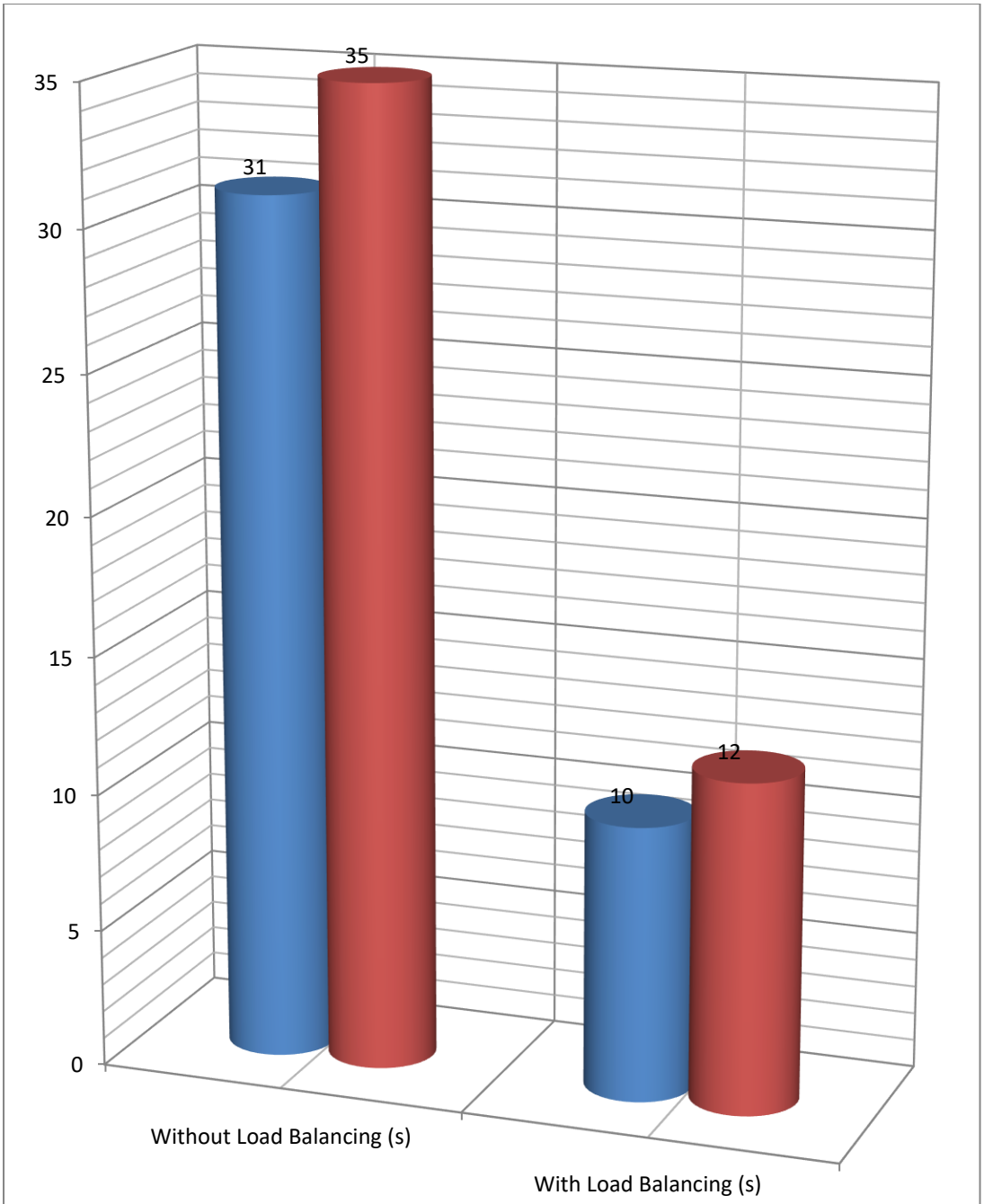



**Figure 4.23 Analysis for Downloading Single File without Load Balancing (in second 's')**

| No. | Download Multiple File | Without Load Balancing | With Load Balancing |
|---|---|---|---|
| 1 | TestingFile1.txt<br>TestingFile2.txt<br>TestingFile3.txt | 31 sec | 10 sec |
| 2 | TestingFileA.txt<br>TestingFileB.txt<br>TestingFileC.txt<br>TestingFileD.txt<br>TestingFileE.txt | 35 sec | 12 sec |

**Table 4.2 Downloading Multiple Files**




| | Without Load Balancing (s) | With Load Balancing (s) |
|---|---|---|
| TestingFile1.txt<br>TestingFile2.txt<br>TestingFile3.txt | 31 | 10 |
| TestingFileA.txt<br>TestingFileB.txt<br>TestingFileC.txt<br>TestingFileD.txt<br>TestingFileE.txt | 35 | 12 |

**Figure 4.24 Analysis for Downloading Multiple Files (in second 's')**

# CHAPTER 5

# CONCLUSION

Peer-to-Peer (P2P) file sharing application has well known a hot conversation on P2P file sharing between all businesses. The proposed system is being revolved around comfort and cost adequacy by utilizing P2P file sharing method on the LAN and capacities of multi-agent technology. This framework is executed considering P2P file sharing system and introduced file sharing component and load balancing component. Utilizing load balancing component agent diminishes the own load and use tasks, planning to designate the task to accessible assets in the network.

Multi-Agent Systems are equipped for dealing with the intricacy of this present reality issues through its arising highlights including coordination, correspondence and talks. Consequently, it can be applied to deal with the ecological intricacy of a PC organization to accomplish better execution and lessen asset wastage.

## 5.1 Benefits of the Proposed System

To decrease the inactivity for recognizing information clashes and to ease the server bottleneck, an efficient load balancing agent, has been proposed for file sharing architectures. By permitting clients to know about the worldwide condition of their stored data, the clients are effectively engaged in keeping up with the downloaded file sharing and load balancing. The client downloaded file sharing with other peers has been intended to decrease the message overhead for maintaining load balancing, network traffic reduction and reduce server workload.

## 5.2 Limitations and Further Extensions

There are some limitations in this system. The system is implemented with three peers and the user must register for accessing the files. Currently, the system is tested to share and download the text files. So, the system can be extended the files with the following extensions: pdf, doc, docx, jpg, jpeg, png, xlsm, .mp3, log, ppt and exe.

This system developed for file sharing and downloading based on peer-to-peer local area network and multi-agent system. The system can be extended in wide area network more than three peers and for accessing the files via the mobile phone tablet.

# AUTHOR'S PUBLICATION

[1] Thida Win, Aye Mya Hlaing, "Peer-to-Peer File Sharing System in Campus Using Multi-Agent", the Proceedings of the Conference on Parallel & Soft Computing (PSC 2022), University of Computer Studies, Yangon, Myanmar.

# REFERENCES


[1] Agostino Poggi and Michele Tomaiuolo, “Integrating Peer-to-Peer and Multi-agent Technologies for the Realization of Content Sharing Applications”, Italy.

[2] Alberto Grosso, “An Agent Programming Framework Based on the C# Language and the CLI”, University of Genova, DIST, Italy, 2003

[3] “An Introduction to Multi-Agent Systems” by Michael Wooldridge

[4] “Artificial Intelligence A Modern Approach Second Edition” by Stuart J.Russel and Peter Norvig

[5] “Autonomous Agents Modelling Other Agents: A Comprehensive Survey and Open Problems” by Stefano V. Albrecht and Peter Stone

[6] Bertolini, D., Busetta, P., Nori, M., Perini, “Peer-to-peer multi-agent systems technology for knowledge management applications”. An agent-oriented analysis. In: WOA 2002, Milano, Italy, pp. 1–6 (2002)

[7] Budditha Hettige, Asoka Karunananda, Kathriarachchi and Weerasinghe, “ITray: Multi-agent solution for LAN based file sharing”. Article in International Journal of Computer Applications, June 2017 Chongjie Zhang, Victor Lesser and Prashant Shenoy, “A multi-agent learning approach to resource sharing across computing clusters”, UMass Computer Science UM-CS-2008-035,2008.

[8] Chongjie Zhang, Victor Lesser and Prashant Shenoy, “A multi-agent learning approach to resource sharing across computing clusters”, UMass Computer Science UM-CS-2008-035,2008.

[9] “Computation Aspects of Cooperative Game Theory” by Georgios Chalkiadakis, Edith Elkind, Michael Wooldridge

[10] “Intelligent Agents: Theory and Practice” by Michael Wooldridge and Nicholas R. Jennings

[11] J. Ferber and O. Gutknecht, “A meta-model for the analysis and design of organizations in multi-agent systems,” presented at the Multi Agent Systems, 1998. Proceedings. International Conference on, 1998, pp. 128–135

[12] K.P. Sycara, "Multiagent Systems", AI Magazine, American Association for Artificial Intelligence, 1998.

[13] Koubarakis, “Multi-agent Systems and Peer-to-Peer Computing: Methods, Systems, and Challenges.” In: CIA 2003. LNCS (LNAI), vol. 2782, pp. 46–61. Springer, Heidelberg (2003)

[14] Lopes, A.L., Botelho, L.M.: “Improving Multi-Agent Based Resource Coordination in Peer-to-Peer Networks”, Journal of Networks 3(2), 38–47 (2008)

[15] M. Grivas and S. J. Turner, “Agent Technology in Load Balancing for Network Applications”, in International Workshop on Intelligent Agents on the Internet and Web, Mexico, 1998.

[16] M. Wooldridge, N. R. Jennings, and D. Kinny, “The Gaia methodology for agent-oriented analysis and design”, journal of Autonomous Agents and Multi-Agent Systems, 15, 2000

[17] Martin Beer, Mark d’Inverno, Michael Luck, Nick Jennings, Chris Preist, Michael Schroeder,” Negotiation in Multi-Agent Systems”

[18] N. Tony and W. Shawn, “Microsoft .Net Framework 2.0 Application Development Foundation”, Microsoft Press, America, 2006.

[19] Ozalp Babaoglu, “Anthill: A Framework for the Development of Agent-Based Peer-to-Peer Systems”, in Distributed Computing System, 2012, Proceedings. 22nd International Conference on. IEEE.

[20] Soumya Banerjee, Joshua P. Hecker, “A Multi-Agent System Approach to Load-Balancing and Resource Allocation for Distributed Computing”, USA, 2015