Adding Peer-to-Peer Functionality to a Commercial Media Publishing Software

Yiping Huang
Abstract

Adding Peer-to-Peer Functionality to a Commercial Media Publishing Software

Yiping Huang

Abstract

Media distribution and publishing has been an emerging market in the IT industry for several years. There are a number of media distribution software programs in current marketplaces. However, few developers have developed this type of software with peer-to-peer file sharing functionality. The overall performance of these types of software may be significantly improved by peer-to-peer functionality compared to traditional client-server architectures. Thus, the aim of this project was to add peer-to-peer functionality to an existing media distribution application to solve the bottleneck problems and to make the application more efficient without much hardware cost.

The system’s database was achieved by using serialisation and de-serialisation technology in .NET with XML files, and some simple interfaces were implemented successfully with Microsoft Visual C#.

The project was functional, with the exception of two functions that were not implemented. All important issues related to research, requirements, problem analysis, design, solution proposal, and implementation and testing are discussed in the project.
Acknowledgements

I would like to take this chance to thank my supervisor, Björn Victor, for his great support. I would like also thank Justin Pearson for his timely feedback on my thesis report. I want to also thank my wife, Yuan Le, for all the love and support she gave. Finally, I would like to say, without your support, I cannot finish this interesting project.
Abstract ........................................................................................................................................ - 2 -
Acknowledgements .......................................................................................................................... - 3 -
Section 1 Introduction ...................................................................................................................... - 8 -
1.1 What is a Media Distribution Network (MDN)? ................................................................. - 8 -
1.2 The company’s current MDN ................................................................................................. - 9 -
   1.2.1 How the MDN works ........................................................................................................ - 9 -
   1.2.2 The company’s content delivery network ..................................................................... - 10 -
1.3 Thesis Motivation ..................................................................................................................... - 11 -
1.4 Scope & Rationale .................................................................................................................. - 12 -
1.5 Aims & Objectives .................................................................................................................. - 12 -
1.6 Report Structure & Time Management ................................................................................. - 12 -
1.7 Risks ..................................................................................................................................... - 13 -
1.8 Time ..................................................................................................................................... - 13 -
1.9 Economic Factors .................................................................................................................. - 14 -
Section 2 Research .......................................................................................................................... - 15 -
2.1 Current Research on Peer-to-Peer File Sharing Systems ...................................................... - 15 -
   2.1.1 What is Peer-to-Peer File Sharing? .................................................................................. - 15 -
   2.1.2 A Brief Review of Peer-to-Peer File Sharing .................................................................. - 16 -
2.2 Existing Research Achievements in Peer-to-Peer File Sharing Systems ...................... - 17 -
   2.2.1 Peer-to-Peer Algorithms ................................................................................................. - 17 -
      2.2.1.1 The Central Index Algorithm .................................................................................. - 17 -
      2.2.1.2 The Flooded Requests Algorithm .......................................................................... - 18 -
      2.2.1.3 The Distributed Hash Table (DHT) Algorithm ...................................................... - 18 -
   2.2.2 Peer-to-Peer Characteristics ............................................................................................ - 19 -
      2.2.2.1 Decentralisation ....................................................................................................... - 19 -
      2.2.2.2 Scalability ................................................................................................................ - 19 -
      2.2.2.3 Performance ............................................................................................................ - 20 -
   2.2.3 Network management ....................................................................................................... - 20 -
      2.2.3.1 NAT Traversal ......................................................................................................... - 20 -
      2.2.3.2 Location Finding ..................................................................................................... - 21 -
      2.2.3.3 Traffic Control ......................................................................................................... - 22 -
      2.2.3.4 Virtual Private Network .......................................................................................... - 22 -
2.3 Design Technologies in Peer-to-Peer File Sharing Systems .............................................. - 23 -
   2.3.1 Structured System Analysis and Design ....................................................................... - 24 -
   2.3.2 Prototype ....................................................................................................................... - 25 -
   2.3.3 Object Orientation .......................................................................................................... - 25 -
2.3.4 Development methodologies ......................................... - 25 -
2.4 The Steps of Developing a Peer-to-Peer File Sharing System .... - 26 -
2.5 The Future of Peer-to-Peer File Sharing Systems ................... - 27 -
2.6 Summary ........................................................................... - 27 -

Section 3 Design ....................................................................... - 28 -
3.1 System Requirements ................................................................ - 28 -
   3.1.1 Organisation-level Information Requirements .................. - 28 -
   3.1.1.1 Tracker Server ......................................................... - 29 -
   3.1.1.2 Supernode .............................................................. - 29 -
   3.1.1.3 Normal client node ............................................... - 29 -
   3.1.2 Database Requirements .............................................. - 30 -
   3.1.3 Application-level Information Requirements ................ - 30 -
3.2 System Constraints .................................................................. - 30 -
   3.2.1 Interface Requirements ............................................... - 31 -
   3.2.2 Performance Requirements ........................................... - 31 -
   3.2.3 Security Requirements ............................................... - 32 -
   3.2.4 Political & Legal Requirements .................................... - 32 -
3.3 System Architecture Design .................................................. - 32 -
3.4 Detailed Design ..................................................................... - 33 -
   3.4.1 User Interface Design ................................................. - 33 -
   3.4.2 Database Design ......................................................... - 34 -
   3.4.3 Product Entity Model .................................................. - 35 -
   3.5.3.1 Entity Required ....................................................... - 35 -
   Table 3.1 Entity Required .................................................... - 36 -
   3.4.3.2 Entity Relationship Types ....................................... - 36 -
   Table 3.2 Entity Relationship Types ..................................... - 37 -
   Table 3.3 Entity Relationship Diagram ................................ - 37 -
   3.4.4 Create Attribute List for Each Entity Type ..................... - 37 -
   3.4.5 Attribute Domains ..................................................... - 38 -
   3.4.6 Entity Key Identification ............................................. - 39 -
   3.4.7 Logical Database Model as DBDL (Database Design Language) - 39 -

Section 4 Solution Proposal ...................................................... - 40 -
4.1 Database Comparisons and Justification ............................... - 40 -
   4.1.1 Oracle ..................................................................... - 40 -
   4.1.2 Microsoft Access .................................................... - 41 -
   4.1.3 MySQL ................................................................. - 41 -
   4.1.4 Sybase ................................................................... - 42 -
   4.1.5 Serialisation and De-serialisation in .Net ....................... - 42 -
4.1.6 Database Decision ........................................ - 43 -

4.2 Programming Language Comparison and Justification ........... - 43 -

4.2.1 C++ ......................................................... - 43 -
4.2.2 Java ......................................................... - 44 -
4.2.3 Microsoft Visual C# ......................................... - 44 -
4.2.4 Programming Language Decision ................................ - 44 -

4.3 Hardware & Software Setup ....................................... - 45 -

4.3.1 Hardware Requirements ........................................ - 45 -
4.3.2 Software Requirements ........................................ - 45 -

Section 5 Implementation & Testing ................................... - 46 -

5.1 Implementation .................................................. - 46 -

5.1.1 GUI Design .................................................. - 46 -
5.1.2 Database Schema .............................................. - 47 -
5.1.3 Required features ............................................. - 48 -
5.1.4 How the P2P system works .................................... - 48 -
5.1.5 Classes Specification .......................................... - 49 -
  5.1.5.1 Client side classes ....................................... - 49 -
  5.1.5.2 Server side classes ..................................... - 56 -
5.1.6 Candidate Delete IP .......................................... - 57 -
5.1.7 Candidate Supernode IP ...................................... - 58 -
5.1.8 Fastest Download Node Selection ............................... - 59 -
5.1.9 MD5 Hash Attribute of File ................................... - 59 -
5.1.10 Multithreading ............................................... - 60 -
5.1.11 Scheduled Traffic Schemes .................................. - 61 -
5.1.12 Secure User Authentication .................................. - 62 -
5.1.13 Virtual Network Topology ................................... - 63 -

5.2 Testing .......................................................... - 64 -

5.2.1 Test Plan ...................................................... - 64 -
5.2.2 Objectives of Testing ......................................... - 65 -
5.2.3 Software description ......................................... - 65 -
5.2.4 Testing tool selection ........................................ - 65 -
5.2.5 Testing method ............................................... - 65 -
5.2.6 List of functions to be tested ................................ - 66 -
5.2.7 Details of Testing ............................................ - 66 -

Section 6 Conclusion .................................................. - 68 -

6.1 Functional Achievements ........................................ - 68 -

6.2 Future Improvements ............................................. - 69 -

6.2.1 Functional Improvements ...................................... - 69 -
6.2.2 Security Considerations ...................................... - 69 -
6.3 Learning Outcomes ........................................................................ - 69 -

Appendix A: Table of Figures .............................................................. - 71 -
Appendix B: Index of Tables ................................................................. - 73 -
Appendix C: Bibliography ..................................................................... - 74 -
Section 1 Introduction

In this section a description about what is a media distribution network will be given firstly. And then discuss briefly about the company’s current MDN. The most important part in this section is the motivation for this master thesis. Besides that, other factors such as scope, objectives, and time etc for the master thesis will be gone through in this section.

1.1 What is a Media Distribution Network (MDN)?

The purpose of an MDN is to publish pre-defined content on desired terminals in a network. A terminal is usually a screen that can be placed at a rail station, a foreign exchange office, a reception area, etc. The common information shared will be train or bus timetables, exchange rates, promotions, etc. With a media distribution application, a customer can substantially increase the power of internal information in his/her organisation and gain more satisfied and informed personnel. The screens can placed anywhere, with electricity, that will catch the eyes of surrounding people. With this solution, companies can, for example, publish their advertisements, display product information or inform employees of news.

The available resources are becoming increasingly rich in today's MDNs. These resources include text, images, music, streaming, animation, etc. In recent years, Internet traffic has increased exponentially, which has resulted in a large demand for network bandwidth. Network access delays and waiting problems are always prevalent. These problems lead to network congestion, server overload and many other issues that decrease the entire content delivery performance.

Easing network congestion and improving end-user access speed has become major issue for many MDN providers. One solution to this problem is to increase Internet
bandwidth from Internet Service Providers (ISPs). However, this solution is unfeasible due to congestion and transmission latency, web server processing capacity and the last mile problem. Therefore, even if every ISP increased their network bandwidth, network delay would be inevitable.

1.2 The company’s current MDN

1.2.1 How the MDN works

The company’s MDN works in the following manner:

1. A task is defined by the publisher. The most important information includes time, content file and a player for display.
2. The client and publisher obtain copies of the files that they would like to display and transfer those files to the company’s server.
3. Those files are transferred to the player, which will display the content files.
4. The display job on the player is triggered at the pre-defined time.

![Figure 1.1 How the company’s MDN works](image)

The supported players in such a network could be a web player, a mobile player, a screen saver etc.

The supported files are listed below:
1.2.2 The company’s content delivery network

The company in which the master project will be completed uses a CDN. CDNs are value-added networks built on top of an existing IP network infrastructure and are deployed in the application layer of the network architecture. CDNs are built on distributed IP network content distribution networks. The basic principle of a CDN is to deploy more edge servers in the areas that have more end users. Contents in such networks are stored in a central server and distributed to the edge server through the backbone network. In this case, customer requests are handled by the edge server to ease the pressure on the backbone network bandwidth.

Adopting a CDN will give the company’s MDN the following advantages:

- Allow users connect to the closest server according to the user's location and connection bandwidth.
- Faster access speed.
- Global load balancing.
- Improve network performance and service quality.
- High reliability, availability, fault-tolerance and easy extendibility.
- Seamless integration into the existing network.
- Reduced network bandwidth consumption, latency and network access user response times.
- Encrypted contents and the prevention of modification of the contents by unauthorised users.
1.3 Thesis Motivation

Although the company’s MDN has so many advantages, it still has its disadvantages.

First, the most important feature of the company’s MDN is that the content routing structure is mostly unchangeable. A request sent by a user can only be handled by certain servers within a limited range, which means that the MDN structure cannot be changed adaptively. If a sudden increase in the number of users in a specific area occurs, explosive data traffic and services will likely cause congestion within the range of data and may even cause a collapse.

Second, another important feature of the company’s MDN is its file-based approach to media files. These media files’ sizes are usually large, and the file-based approach consumes cache rapidly which is accompanied by rapid decreases in the ability to support concurrent user requests. Thus, this feature further stresses network traffic and services.

Additionally, the MDN is a client-server (C/S)-based architecture. The C/S architecture has advantages for content file management. All content files can be controlled via the content server in this architecture. However, this architecture has more disadvantages as described below:

- It is difficult to cope with large-scale concurrent clients with a multiple-client-single-server architecture.
- Resources on the server easily result in bottleneck problems in network.
- Clients’ utilizations of resources such as disk space and CPU time are low.
- Unicasting and multicasting is difficult to achieve between clients in a C/S architecture.

Last but not least, the company needs to think about the cost of adding to and maintaining its MDN if the number of users increases dramatically in future. As the company described in the thesis specification, MDNs are a very competitive area in which cost is one of the most important keys to success.
In conclusion, the motivation behind the master’s thesis is to enable the company’s MDN to handle thousands of concurrent requests in a manner that is both economical and efficient.

1.4 Scope & Rationale

In June of 2007, it has been decided that the master’s project will be in the field of computer network technology because this is my favourite area of the computer science discipline. After several days, an opportunity was given to do a master’s project on the topic of peer-to-peer platforms for media distribution in a company. This company explained the problems they were currently facing to me, and we also discussed what could be done to improve the performance and security of the media distribution software. Thus, an idea has been popped up that to develop a peer-to-peer functionality for media distribution software that can be widely used in the future. Although quite a few P2P systems already exist, the company wanted to create its own product because this P2P system will hopefully be used in the company’s current product, which is completely closed source.

1.5 Aims & Objectives

The aim of this project is to add peer-to-peer functionality to an existing media distribution application to solve the bottleneck problems caused by the traditional client-server architecture and increase the application’s efficiency with minimal hardware cost.

The objective of the project is to design, implement and test the system and database. During the time of this project, several different databases and programming languages will be compared and discussed, and ultimately, a decision will be made regarding the implementation of this system.

1.6 Report Structure & Time Management

The structure of the report is based on logic and a timeelapsed process. Logic indicates
the regularity of thinking, and meticulous logical thinking is key for the completion and success of the master’s project. The project consists of research, analysis, design, implementation, testing and conclusions. The first part, research on current peer-to-peer file sharing systems and analyses of problems in these systems, will lead to the second part, which will be design and implementation. After implementation, some testing will be performed, and some feelings and suggestions will be produced during this time.

Time-elapsed process indicates that the master’s project was carried out according to a pre-defined schedule that consisted of various types of tasks in different project phases. Making a reasonable schedule was a key factor in making the master’s thesis successful. Scheduled tasks are useful. With this timetable, my research, analysis and design are all done prior to September 2007. In the subsequent two and one-half months, implementation and testing were performed.

1.7 Risks
There are some risks in the project. The data in this project are not sensitive, and there are no dangers in the implementation of the project; however, this does not mean that security features should be implemented in the system because security feature can ensure the security of the system and improve information security. The risk of failure is important because I entered into a contract with a small company regarding the software. Additionally, the source code needs to be ensured is confidential.

1.8 Time
A contract has been entered regarding the piece of software. Thus, it is important to submit the software before the deadline. Additionally, the master’s thesis course has a deadline. The deadline needs to be strictly followed. Several factors may affect the plan to develop the system; for example, changing, or adding new requirements to, the system and other assignments or examinations that need to be completed during the period the system is under developing may affect my plan.
1.9 Economic Factors

The development software needed to create this system is free of charge for me. Microsoft Visual C# is accessible from Microsoft’s website and Microsoft Office is also accessible from Uppsala University. The completion of this project will incur expenses for the company; the company has agreed on a small salary, which has been written into this master’s project contract.
Section 2 Research

Research is a crucial part of developing a system. During the time of this master’s thesis project, both theoretically and practically related materials will be explored.

The project originally sought to develop a peer-to-peer file sharing function for an existing commercial media distribution software. However, it is found out that it almost impossible to do that because the source code of the current version of the software is not accessible, and the time for the project was restricted. Thus, after a discussion with the CTO of the company, it has been decided to develop the system as a prototype, and all future work regarding integration will be performed by someone else in the future if the company adopts my prototype. However, future integration needs to be considered because the prototype under developing will have the ability to add functions, and integration will be possible without major changes to the source code.

In this section, some key aspects of the project will be discussed, which includes current research achievements in peer-to-peer technology and problems emerged from its previous achievements.

2.1 Current Research on Peer-to-Peer File Sharing Systems

2.1.1 What is Peer-to-Peer File Sharing?

Peer-to-peer file sharing is not a new in computer network technology. Each user in a peer-to-peer a network is required to fully or partially share his/hers files with others while downloading files. In such a system, one node acts as both contributor and consumer. Regarding contribution, a node has server-like functions that include routing and sharing. Regarding consumption, a node has client-like functions that include file integration checks and obtaining files. All nodes in a P2P file sharing system have equal status. A node can join and quit a P2P network freely. Decentralised, centralised and
hybrid P2P systems exist. Although topologies differ between different P2P systems, the rationales behind the systems are the same; i.e., to establish direct file sharing between peers.

2.1.2 A Brief Review of Peer-to-Peer File Sharing

There have been four generations peer-to-peer file sharing models.

The first-generation model, called the server-client model, maintains a central server. When a user wants to download a file, the user asks the central server where the file is and then a download task is executed between the node with the file and the current user. This process differs from traditional client-server architectures in which the download only occurs between the central server and the user.

The second-generation peer-to-peer model is called the decentralised model. This model differs from the first generation in that there is no need for a central server. All nodes in this type of network can voluntarily be a server.

The third-generation peer-to-peer model is called the indirect and encrypted model. This model incorporates the decentralised model and adds more sophisticated technology and anonymity. Due to this anonymity, it is very hard to determine who is downloading a file or from where. Additionally all data that are transferred on network can be encrypted. These two new features increase the security level of peer-to-peer sharing.

Due to increase in the demand for media file sharing, the fourth generation, peer-to-peer streaming, was developed. This model can use either centralised or decentralised peer-to-peer sharing and has some support for large file sharing. One of the most interesting technologies is called swarm technology. Swarm technology allows a group of users to form a swarm, and each member of a swarm can be both a contributor and a distributor simultaneously. This solution provides an alternative method of sharing data between nodes that is efficient in networks with low bandwidth.
2.2 Existing Research Achievements in Peer-to-Peer File Sharing Systems

In this section, the progress that has been made in peer-to-peer file sharing systems will be discussed. This section is divided into three main parts: peer-to-peer algorithms, peer-to-peer characteristics, and network management.

2.2.1 Peer-to-Peer Algorithms

Peer-to-peer algorithms refer to the algorithms that allow one node to find another node in a P2P network. Thus, these algorithms are mainly concerned with routing. There are three types of P2P algorithms: central index algorithms, flooded requests algorithms, and distributed hash table algorithms.

2.2.1.1 The Central Index Algorithm

This is the peer-to-peer algorithm that is most similar to traditional client/server architectures. This algorithm could be deployed on systems with central servers. Each newly joined node registers its contents with the server. In such a system, all requests come to the server, and the server is responsible for finding the best node to match the request. The most well-known peer-to-peer software that uses this algorithm is Napster[1], which has been shut down for illegally transferring copyrighted material.

![Figure 2.1 The central index algorithm (Androutsellis-Theotokis, 2004)[2]](image-url)
2.2.1.2 The Flooded Requests Algorithm

In this type of algorithm, there is no need for a central server. When a request is sent out from one node, it is first broadcast to all connected nodes. If there a node that matches the request exists, communication is established between these two nodes. Broadcasting can consume a lot of bandwidth, and this issue needs to be overcome. One typical solution is to set a pre-defined value. This value can help to limit the flooding. This method is similar to hop in networking. This algorithm is used by Gnutella[1], which is currently one of the most successful peer-to-peer file sharing software programs.

![Diagram of Flooded Requests Algorithm](image)

**Figure 2.2** The Flooded Requests Algorithm (Hölzl, 2000)[3]

2.2.1.3 The Distributed Hash Table (DHT) Algorithm

In the DHT, each node is assigned a random ID when registering to the network, and each new node knows some other nodes in the network. When a file is registered on a node, the file will also be assigned an ID by its content and name. DHT is the most advanced algorithm because it helps to reduce the latency of each peer-to-peer hop and also reduces the number of hops required to search a node. There are many peer-to-peer software programs that use the DHT algorithm including CAN[4], Chord[5], Pastry[6], etc.
2.2.2 Peer-to-Peer Characteristics

This section will discuss three main characteristics of peer-to-peer file sharing systems that differentiate these systems from other types of system.

2.2.2.1 Decentralisation

Decentralisation is considered as one of the most important features of peer-to-peer systems. In a pure peer-to-peer network, there is no central server. By eliminating the central server, the network can avoid several problems including inefficiency and bottlenecks. On the other hand, controlling the network is difficult because each node has the same level of ownership.

2.2.2.2 Scalability

A major drawback of the client-server architecture from scalability perspective is that this architecture needs to maintain a large index server. The greater the number of nodes in such a network, the heavier the burden is on the central server. This issue can be avoided with fully decentralised peer-to-peer networks because there is no need to
maintain a central server, and each node is smarter than the nodes of a centralised peer-to-peer network. Here, smarter indicates that each node has central server-like functions such as node join, quit, network scaling, etc.

2.2.2.3 Performance

A peer-to-peer network’s performance is determined by several factors that include processing, the network and storage. Processing performance mainly refers to how quickly a request can be handled. Processing performance relies on hardware parameters such as CPU speed and memory size. Network performance refers to how fast a file can be downloaded from other nodes. It is important to ensure that each node has a good upload speed because while a specific node is downloading a file, other nodes are downloading part of the same file from that specific node.

In these cases, partial file sharing can help to improve the download speed. Partial file sharing let a node obtain its requested file from several nodes and has the following advantages:

- Increased download speed – a node can create multiple sessions with different nodes to increase download speed compared to downloading from a single node.
- Ensured file integration – multiple copies of a file on a network can ensure file integration by comparing a downloaded file with several copies on different nodes.
- Bandwidth optimisation – idle network bandwidth is utilised to reduce the bandwidths of busy nodes.

Storage performance determines how many files can be stored on each node. When the downloading file(s) exceeds the disk quota, the downloading action will fail.

2.2.3 Network management

There are several new features have been introduced in peer-to-peer networks to improve the performance of the system including NAT traversal[8], location reference finding, traffic reference control and virtual private networks[9].

2.2.3.1 NAT Traversal
Due to the limit number of IP address in the IPV4 protocol, it is common that a node only has a private IP address in the local area network (LAN). Normally, this type of node is behind a firewall (private) and connects to the Internet through a router, which has a public IP address. The NAT traversal solution is suitable for creating a bridge between two nodes behind such a firewall. Hole punching is one NAT traversal technique. The hole punching technique is further split into UDP, TCP and ICMP hole punching. The basic idea of these three types of hole punching techniques is similar, and the only difference is how the connection is created between two nodes. UDP hole punching will be used as an example to explain the hole punching technique.

- Clients A and B reside on different private networks and would like to create a UDP communication via server S.
- NAT A and NAT B help assign clients A and B, respectively, to UDP ports.
- S checks whether the UDP packet has been sent and received correctly.
- S forwards both A and B’s endpoint information to the other side.
- Clients A and B create a direct connection between each other with their NAT devices.

![Figure 2.4 UDP hole punching (Ford 2005)](image)

### 2.2.3.2 Location Finding

Location finding is a technology that improves the scalability of peer-to-peer systems. Suppose a node finds several nodes that match its request, that node must then
determine which of the other nodes is best. Here, ‘best’ can be defined in terms of speed, reliability, etc. Using location finding, a node can find the logically closest node among a list of nodes. Logically close indicates the smallest number of network hops between two nodes.

2.2.3.3 Traffic Control
Most of us who have used peer-to-peer software such as BitTorrent[10] and Emule[11] have had the experience of peer-to-peer uploading and downloading consuming all available bandwidth. In the past, downlink traffic composed the majority of network traffic. However, due to P2P technology, all clients need to upload their resources to the entire network, which consumes enormous amounts of uplink traffic. Usually the uplink speed is lower than the downlink speed, and continuous uploading will result in extreme network congestion, especially during the working hours from 9 am to 5 pm. This process adversely affects other uses of the network. The problem becomes worse if peer-to-peer software is used in the office during peak times. The function called traffic control solves this problem. The traffic control function can limit both upload and download speeds and reserve some bandwidth for other uses. The function has can be enabled and configured according to users’ needs.

2.2.3.4 Virtual Private Network
A virtual private network (VPN) is a network inside a network and is an internal communication channel for an organisation over a public network such as the Internet. A VPN establishes a dedicated communication channel between two different places with the help of a special encrypted communication protocol such as the Internet Protocol Security (IPsec), Transport Layer Security (SSL/TLS), etc. Virtual Private Networks are an extension of the enterprise intranet. Virtual Private Networks can help remote users establish reliable communication channels and ensure the security of data transmission.

Network administrators may want to control all resources in a VPN. For example, a network administrator can use some advanced security features of a VPN to ensure that only management staff can access those nodes that contain a company’s financial
information and customer list. A common method of performs task such as this is to have each node maintain an access control list (ACL). The ACL can determine whether a node can access another specific node.

![Diagram showing how a VPN crosses the Internet](image)

**Figure 2.5** How a VPN crosses the Internet (Gregory, 2013)[12]

### 2.3 Design Technologies in Peer-to-Peer File Sharing Systems

Many types of technologies are involved in system development. However, problem analysis is very important to any type of development strategy. Problem analysis includes the following aspects:

- What problem or problems is the system supposed to resolve? For example, how should problems in information processing be solved? Additionally, how should new requirements be satisfied?

- Research regarding the system’s feasibility [13] should confirm the aims and objectives of system. The objectives of a system’s feasibility include the consideration of technology and financial support.

- The principle of the system’s development; this principle can include end-users’ participation, innovation and practicality.

- The hardware and software preparations for system development.

Based the above considerations, a suitable system development strategy and plan is established. Then, according to the chosen strategy and plan, a development method is selected. There are three main types of development methods: structured system
analysis and design[14], choosing prototypes[15], and object-oriented methods[16].

2.3.1 Structured System Analysis and Design

Currently, structured system analysis and design is much more popular in system development. The basic concept of structured system analysis and design is to use a systems’ thought and systems engineering to analyse and design systems, from top to bottom, according to their lifecycles while following the rule that end-users are sovereign in systems.

Because the master’s project description only defines what should be delivered and there is no strict requirement about how to realise peer-to-peer functionality, it has much freedom to define the development strategies, database or application-level requirements, development methodologies, etc.

Based on structured system analysis and design, the whole development process should be split into the following five phases:

- **System layout phase**: the primary task is to ensure the system’s requirements. Some investigation should be performed during this phase, and developers should ensure what to do next via feasibility research. SST (strategy set transformation), CSF (critical success factors) and BSP (business system planning) are frequently used in system layout.

- **System analysis phase**: the main task of this phase is to analyses the system’s structure and function and thereby organise a logical scenario.

- **System design phase**: the chief goal of this phase is to confirm the whole system's scheme, split it into several sub-systems and then design each part of the system in detail; these sub-systems could include, for example, database design, interface design and programming design.

- **System implementation phase**: the principal tasks of the phase are to discuss design methods, test sub-systems, input data and train operators.

- **System running phase**: the primary tasks of the phase are the maintenance of the system and the evaluation of the system’s performance.
2.3.2 Prototype
The basic concept of this method is that developers construct a concrete system prototype with powerful support of the software environment by writing our their understandings of users’ requirements and then modifying the prototype depending on users' suggestions and developers’ opinions until the users are satisfied with the prototype. Prototypes gradually increase the understanding of both the developers and users and then ensure measure for measure system development. Additionally, the use of prototypes provides many functional facilities that ensure development runs quickly and efficiently.

2.3.3 Object Orientation
The concept of object orientation brought computer programming into a new era. This concept has recently become more popular. The basic concept of object orientation involves thinking of the world as many objects that are related to each other and then developing a suite of software based on objects and methods. The main features of object orientation are encapsulation, inheritance and polymorphism.

2.3.4 Development methodologies
There are quite a few development methodologies including structured systems analysis and design method (SSADM) and rapid application development (RAD). SSADM was introduced by Learmonth and Burchett[17] and become a popular version of the traditional information systems development approach. This methodology defines rules and guidelines in detail meaning that SSADM is highly structured and restricted. RAD caters to the need to develop information system quickly. RAD is not based on the traditional life cycle but adopts suitable approaches from the traditional life cycle, and the early enrolment of end users is important in RAD; without this early enrolment, RAD can’t be achieved. Because a weekly meeting is set with my supervisor in the company, the requirements for the system can be changed every week; thus, RAD is a better development methodology for this project, and RAD is adopted the development methodology of the master’s project.
2.4 The Steps of Developing a Peer-to-Peer File Sharing System

When concerned with database technology, developing a system usually involves the following eight steps:

1) System development preparation
2) System investigation
3) System analysis
4) System design
5) System implementation
6) System conversion
7) System maintenance
8) System evaluation

Database technology is an unavoidable part of developing a system because data is the key factor for such systems, and all functions of such systems are useful only when manipulating data. A database is a software system for handling data in an efficient, safe and easy way. The development of a database system should be viewed from the perspective of the wider requirements of the organisation.

The design of a suitable database should be split into the following six steps.

- Clarifying the requirements of systems based on the required functions.
- Design of the conceptual schema, which is an invaluable and stable description of the contents of the database.
- Choosing a suitable DBMS. Designers should consider technical, economic, and organisational factors.
- Data model mapping. The result of this phase should be DDL statements in the language of the chosen DBMS.
- Design of the physical database based on system performance requirements
- Implementation and testing of the systems.

During the system development period, these steps have been followed, which it is proved to be helpful.
2.5 The Future of Peer-to-Peer File Sharing Systems

Currently peer-to-peer is a mature computer network technology with many successful software programs because it has been corrected and improved for decades. The discussion of the future of peer-to-peer systems will be more about legal issues rather than technology. One advantage of P2P file sharing is that everyone can contribute to the entire network, which makes tracking copyrighted content, such as music and video, difficult. For example, the Pirate Bay is blocked in many countries due to copyright issues[18]. However, P2P file sharing in itself is legal as long as the content shared is legal. Copyright issue also exists in both http and ftp based file transferring. The only difference between them with P2P is downloading speed.

2.6 Summary

In summary, in this section, several aspects of peer-to-peer systems have been discussed including peer-to-peer technology, development methods, development steps and the future of P2P.
Section 3 Design

After understanding the current problems with the company’s MDN and performed some research in several fields, it was the time for the design phase. In this section, the requirements and constraints of the system will be illustrated. Subsequently, a description about how to design the system will be given.

3.1 System Requirements

The analysis of a system’s requirements is an early stage of developing a system. However, this does not mean that system requirement analysis is unimportant in system design. The purpose of requirement analysis is to provide a narrative definition of functional and other requirements, including social, communication, ethical issues, etc., that must be met in the implemented and deployed system.

The system’s requirements will be discussed from three levels: organisation-level information requirements, database requirements, and application-level information requirements.

3.1.1 Organisation-level Information Requirements

This is the highest level of system requirements. This level is a key element in planning a system because the general structure of the system is defined at this level. Now, the overall structure of the system will be detailed.

The objective of this system is to add peer-to-peer technology for downloading and uploading files. Because the system is developed as a prototype, many add-on functions, such as add-on on web browsers, multi-protocol support and anonymous download,
have not been implemented in the prototype. However, the functions that have been achieved, which include peer-to-peer file transferring, file integration check, multithreading, traffic control, user authentication and closest IP finding, are adequate to make the system as working prototype. The system is organised into two parts. There are only two types of node in the system, and they act as tracker servers or client nodes. All clients the network are either normal client nodes or supernodes. Any normal client node has the potential to become a supernode when it has contributed to other clients in the system. A client node’s database contains a property, ‘supernode_option’, that determines whether that node is a supernode. A supernode is a client the network that can be both a contributor and a beneficiary. A normal node is a client that can only request files in the network and does not share anything.

3.1.1 Tracker Server
The tracker server is responsible for helping any pair of nodes communicate with each other and the transfer of essential meta-data to the node that requested the information. Additionally the tracker server holds all copies of files in the entire network. Thus, the tracker server is more important than the supernodes in the system. To make the system work, at least one tracker server has to be introduced.

3.1.1.2 Supernode
A supernode is a normal node in the system. Each node knows its supernode and child nodes. Each node is actually an LCD screen plus a PDA or other embedded PC. The node will obtain files in some way and then display the files according to the schedule made by management staff.

3.1.1.3 Normal client node
A normal client node is a supernode, and it can be displayed on any client such as LCD screen, PDA or other embedded system. The normal client nodes will display the file at the requested time.
3.1.2 Database Requirements

End users in the system may include foreign currency exchange counters, hotels or bus stations. An end user will create its own job based on what information it would like to display and when and how it would like to display that information. Database requirements can be divided into the users’ requirements and the requirements for physical design.

End-users’ requirements are referred to as conceptual or logical requirements because the users’ view of database is separated from the physical location of the database. There are two ways to meet users’ requirements, research of existing systems and data modelling. In the peer-to-peer system, end user information can be tracked on the tracker server, and file information will be stored on each client.

The requirements for physical design are concerned with the how. These requirements describe the base relations, file organisations, and indexes used to achieve efficient access to the data and any associated integrity constraints and security measures. In the peer-to-peer system, a lightweight XML-based database will be created for each node to store file-related information.

3.1.3 Application-level Information Requirements

An application is a subsystem of the overall information system structure. Applications provide information processing for an organisational unit or organisational activity. At this level, there are two types of requirements: social and technical.

In the peer-to-peer system, the most important aspect of this stage is the choice of suitable attributes. The attributes in the system include the file and the user. When attributes are defined, the major part of system design has been accomplished.

3.2 System Constraints

System requirements define what the system must accomplish. System constraints describe how the system is constrained in performing those services. The system’s
constraints in terms of four aspects will be discussed.

3.2.1 Interface Requirements

Interface requirements define how the product interacts with the users. Because the output information for defined task is implemented in the main application, the prototype will only have a basic input window for verifying the end user by checking the user’s name and password. Although the system’s interface is quite simple, several rules should be followed:

- The interface is for communication between a human and a computer. We should try to make it simple because it will only be used for information input.
- Uniform layout means that each human-computer interface in such a system should have the same style. The benefit for this is it will help end users avoid having to change to other types of interface only for aesthetic reasons.
- The interface must help end users understand what system is processing at any given time; for example, the status of the running software should be accessible, like in the “Windows task manager” of the Windows operating system. This feature will help me to check the current system status and find any potential problems such as network outages and power failures.
- A help function describing how to use the system will be useful.
- Input validation and help will be necessary. For example, when end users input letters in the “Date of birth” blank, the system should have a hint such as, “please input numbers in the following format “DD/MM/YYYY”.

3.2.2 Performance Requirements

Performance requirements can be an important element of a successful system. Performance requirements are concerned with the system’s reliability, throughput, etc. The tracker server should be able to handle incoming requests in an accurate and timely manner. The distributing node should be able to download files efficiently.

Because the system is a prototype, the company only defined the functions need to be achieved, and most of the time is spent on these function-level requirements first. A
performance test may be carried out when the system is done. The key indexes of such a system could be download speed, response time, etc.

3.2.3 Security Requirements

Security requirements describe the users’ access privileges in the system and file integrity checks of the files transferred over the network. These requirements are usually a part of network security. Several mechanisms, including authentication, encryption and access control, are discussed in this stage.

The security requirements include user authentication, access control lists and file integrity checks. Because the system will only allow specific users, a login dialogue will be made to check whether the current user is allowed to enter the network. Furthermore, different levels of privileges will be awarded depending on the users’ login names and passwords. A successfully logged-in user can be an administrator or an end user. Then, access control lists will be used to control the accessibility of each node to enhance the network’s security. Moreover, the file integrity check mechanism will be adopted to ensure that the user can obtain the correct file without being changed over the network. These security mechanisms may not be sufficient for a business network solution; thus, a VPN may be adopted to achieve sufficient network safety.

3.2.4 Political & Legal Requirements

These two requirements are possibly not stated in the system constraints, which could be a very serious mistake. Code need to be protected unless they are open. In the system, all code need to be maintained confidentially. Thus, the system on the code-level will not be discussed in the following sections.

3.3 System Architecture Design

The traditional two-tier architecture[20] has been used to design the system’s architecture. The user interface is integrated with business logic, thus, the client architecture is thick. The business logic, in this context, is responsible for parsing the commands sent from the user interface layer, handling data transfer and data integrity
checks. Business logic is also called middleware, which, among others, can be web server or a transaction processor.

![Diagram: Two-tier system architecture](image)

**Figure 3.1** Two-tier system architecture

Currently, the more advanced three-tier architecture is much more popular for solving the bottleneck of narrow network bandwidth. In the traditional two-tier architecture, a centralised database system holds a full copy of the system. A client needs to communicate with that database server directly to ship data from the database server to the local client, which generates vast amounts of data traffic. In the three-tier architecture, the client will interact with the application logic instead of the database server. This extra layer offloads work onto the database server, which can improve the system's performance significantly. Although several problems with the two-tier architecture have emerged in recent years, such as performance and portability, this architecture is considered more suitable for the system design because the amount of information that will be processed by the system will not be very large, so no problems will arise for this thick client architecture.

### 3.4 Detailed Design

After the architecture of the system has been defined, it is time to begin detailed design.

#### 3.4.1 User Interface Design

Designing a user interface is not difficult; however, designing a set of good user
interfaces for a system is not easy. Many skills are involved in user interface design including arts, requirement analysis, etc. For the prototype, only some simple required interfaces are provided to make the system work. The system is developed as a ‘console application’ in Microsoft Visual C#. All basic information, such as login, configuration, downloading information, etc., is displayed in the command window either on the client side or on the server side. User authentication, file transfer and bandwidth control are the most important functions in the system.

After launching the application, the client first needs to login to the system using a visiting password. The password has been predefined, and the client will be forced to exit if the wrong password is entered more than three times. When the client types in the correct password, he/she will enter the network. After logging in, the client can define his/her own password with the command "livefile --user 'mypassword'", which will be saved in an xml file. After that, the user can ask whether there is a file called "fileName" is on the network with the command "livefile--file fileName". If the file can be found in the network, the file will be transferred to the client. "--file" is the compulsory option. Additionally, the application also supports the following options:

--address: force retrieval of a file from that IP address, return fail if not found.
--bandwidth: set download speed in KB.
--help: list the available options for the command livefile.

If an end user types an invalid command into the command prompt, which cannot be parsed by the application, an error message will be thrown, and the end user will have to re-type the command.

3.4.2 Database Design

Database design involves two main stages: producing an entity model and creating an attribute list for each entity type[21].

The database is designed following these two stages. The database design mentioned here is the conceptual and logical database design.

A light-weight xml based database is used on both the tracker server and client node to
store data. On a client node, this database holds the access control list, file information and user information. Additionally, a tracker server also knows which IP addresses have been unreachable in the network. The available IP’s database will be updated regularly on the tracker server. When a client requests a file, the tracker server will tell the request side to obtain the file and specify the IP address from which the file should be retrieved. If the requested file transfers successfully over the network, the contributor's IP will be saved on the client, together with the file name and the MD5 checksum information, in File.xml. After a client contributes a file successfully to any other node in the network, that client will become a supernode. If the tracker server gives an IP address that is unreachable at the moment, the client will send a signal to the tracker server, and the tracker server will increase the counter for that IP in deleteIP.xml by one. Then, the tracker server will look in the .xml file to find another IP and send it back to client. If there is no other IP that holds a copy of that file, the tracker server will send its backup copy to the requesting client.

There is no need to differentiate supernodes and normal client nodes on the tracker server because a client node will automatically become supernode once a file on that node is requested by another client node. Regarding privileges, the system has only one administrator, and the information is stored on application itself. Thus, there is no need for an extra table for privileges that which link to different nodes.

### 3.4.3 Product Entity Model

#### 3.5.3.1 Entity Required

<table>
<thead>
<tr>
<th>Entity Name</th>
<th>Description</th>
<th>Aliases</th>
<th>Occurrence</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACL</td>
<td>A list of IP which is used to built up virtual network topology.</td>
<td>Access</td>
<td>Each node in system maintains a table named ‘ACL’ to control access to other nodes when needs.</td>
</tr>
</tbody>
</table>

| | | | | |
File  A collection of related data or program records stored as a unit with a single name.

User  A combination of IP and password which is used to identify a unique node on network. All nodes are defined by IP because each IP will be assigned statically in such a network.

DeleteIp  A combination of IP and count which is used to count the failure of reach the node with this IP.

When a node needs to communicate with another node on network, the node is defined by IP address together with its password to check if it is an authorized user by checking password provided.

When a node is not reached, its count attribute will be increased by one. When count reach pre-defined number, the node with this IP will be deleted from network.

Table 3.1 Entity Required

<table>
<thead>
<tr>
<th>Entity Name</th>
<th>Multiplicity</th>
<th>Relationship</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

3.4.3.2 Entity Relationship Types

Entity relationship types is a high-level data model which representing data in a conceptual way in a system. The two key factors are entity and relationship. It helps database designer to have a clear picture about what entities and how they will link to other entities in a database system. Below is the ER-types in the system.

<table>
<thead>
<tr>
<th>Entity Name</th>
<th>Multiplicity</th>
<th>Relationship</th>
<th>Entity Name</th>
<th>Multiplicity</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
User 1..* transfers File 0..*
1..* configures ACL 1..1
1..1 counts DeleteIp 0..*

Table 3.2 Entity Relationship Types

![Entity Relationship Diagram]

Table 3.3 Entity Relationship Diagram

3.4.4 Create Attribute List for Each Entity Type

<table>
<thead>
<tr>
<th>Entity</th>
<th>Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td></td>
</tr>
<tr>
<td>ACL</td>
<td>allowed_ip</td>
</tr>
<tr>
<td>File</td>
<td>ip_address, file_name, md5</td>
</tr>
<tr>
<td>User</td>
<td>ip_address, user_name, password, supernode_ip, supernode_option</td>
</tr>
<tr>
<td>DeleteIp</td>
<td>ip_address, count</td>
</tr>
</tbody>
</table>

Table 3.4 Create Attribute List for Each Entity Type
### 3.4.5 Attribute Domains

<table>
<thead>
<tr>
<th>Entity Name</th>
<th>Attribute</th>
<th>Description</th>
<th>Data Type &amp; Length</th>
<th>Nulls</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACL</td>
<td>allowed_ip</td>
<td>Unique identifier for a node</td>
<td>15 variable</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>characters</td>
<td>No</td>
</tr>
<tr>
<td>File</td>
<td>ip_address</td>
<td>Unique identifier for a node</td>
<td>15 variable</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>characters</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td>file_name</td>
<td>Name of the file</td>
<td>80 variable</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>characters</td>
<td>No</td>
</tr>
<tr>
<td>md5</td>
<td></td>
<td>Md5 hash code of the file</td>
<td>128 variable</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>characters</td>
<td>No</td>
</tr>
<tr>
<td>User</td>
<td>ip_address</td>
<td>Unique identifier for a node</td>
<td>15 variable</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>characters</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td>user_name</td>
<td>Name of each user</td>
<td>20 variable</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>characters</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td>password</td>
<td>Password of each user</td>
<td>80 variable</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>characters</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td>supernode_ip</td>
<td>Supernode of current node</td>
<td>15 variable</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>characters</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>supernode_optio</td>
<td>If current node is a supernode or not</td>
<td>1 variable</td>
<td></td>
</tr>
<tr>
<td></td>
<td>n</td>
<td></td>
<td>character</td>
<td>No</td>
</tr>
<tr>
<td>DeleteIp</td>
<td>ip_address</td>
<td>Unique identifier for a node</td>
<td>15 variable</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>characters</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td>count</td>
<td>Count of current ip_address</td>
<td>1 short integer</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>No</td>
<td></td>
</tr>
</tbody>
</table>
3.4.6 Entity Key Identification

<table>
<thead>
<tr>
<th>Entity name</th>
<th>Primary key</th>
<th>Candidate key</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACL</td>
<td>Ip_address</td>
<td></td>
</tr>
<tr>
<td>File</td>
<td>Ip_address</td>
<td></td>
</tr>
<tr>
<td>User</td>
<td>Ip_address</td>
<td></td>
</tr>
<tr>
<td>DeleteIp</td>
<td>Ip_address</td>
<td></td>
</tr>
</tbody>
</table>

Table 3.6 Entity Key Identification

3.4.7 Logical Database Model as DBDL (Database Design Language)

<table>
<thead>
<tr>
<th>ACL</th>
<th>(ip_address)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Primary key</td>
<td>ip_address</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>File</th>
<th>(ip_address, file_name, md5)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Primary key</td>
<td>ip_address</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>User</th>
<th>(ip_address, user_name, password, supernode_ip, supernode_option)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Primary key</td>
<td>ip_address</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>DeleteIp</th>
<th>(ip_address, count)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Primary key</td>
<td>ip_address</td>
</tr>
</tbody>
</table>

Table 3.7 Logical Database Model as DBDL (Database Design Language)
Section 4 Solution Proposal

With the completion of the analysis of the project’s requirements and design, the project has entered the implementation stage. However, before implementation, a suitable programming language (for the interface and function realisation) and a database for the peer-to-peer system need to be chosen. Thus, in this section, several programming languages and database products will be compared and then choose a suitable solution for my projects. At the end of this section, an overall setup environment of the project will be provided that includes hardware setup and software setup.

4.1 Database Comparisons and Justification

4.1.1 Oracle

The Oracle Corporation is the leading information management software supplier and the world’s second largest independent software company (Microsoft is the largest.) Oracle is a large-scale relational database based on SQL, meaning that Oracle manages large amounts of data using SQL, a logical query language. Oracle uses the most popular architecture, client-server architecture. Oracle applications cover many fields such as e-commerce, financial, and human resources and are widely used by companies that need to manage large amounts of information. Oracle is adaptable in both Windows-based operating systems, such as Windows 2000 and Windows XP, and UNIX-based systems such as Linux. Oracle has improved much since 1979, when the first version of Oracle database software (Oracle 2) was released; this software was the first commercial SQL RDBMS. The latest edition of Oracle is Oracle 10g. Across several recent Oracle editions, object features have been introduced gradually, and these features have convert Oracle from a pure relational database management system to an
object relational database management system to cater to the new requirements of data management. Object relational database management systems (ORDBMSs) are solutions that are transitional from relational to pure object-oriented database management systems. In an ORDBMS, the concepts of object-oriented programming, such as class, inheritance and object are adopted when designing, a system. Programmers can define new data types in object-oriented languages and then use SQL queries to retrieve those objects from a database. The downside of Oracle is its high price for small and medium sized companies. The price of the Oracle Enterprise Edition is more than 60,000 dollars according to Oracle website[22].

4.1.2 Microsoft Access

Microsoft Access is a database product that is a part of the Microsoft Office package (since 1992), and it was also the first PC database for Windows. Because Microsoft Access is bundled into Microsoft Office, Access is widely used in Microsoft’s Windows-based operating systems. Access is also a relational database. Unlike Oracle, Access caters to small sized companies’ requirements for databases. Thus, developers have made its operation easier compared to Oracle and much cheaper than Oracle. A significant function of Access allows end users to create tables, queries, forms and other components using a graphical user interface (GUI), which is easier and more interesting than some input some commands. Additionally, Access also supports SQL interfaces into which the database designer can type SQL queries[23]. Limits of Access include the limitation on database size and its inability to run on UNIX-based operating systems. The maximum Access database size is 2GB; whereas other commercial vendors have much larger database size limitations. For example, MySQL has a 4 GB limitation, and Oracle has no limitations on its database size.

4.1.3 MySQL

MySQL is an open source database, as Linux is an open source operating systems. MySQL is currently the most used database worldwide. MySQL is an adequate database system for daily use. However, compared to the former two databases, MySQL has less support for elementary features such as basic data types and SQL language features[24].
In 2008, MySQL was purchased by Oracle to target low-to-middle end users. Subsequently, MySQL has become much more popular in recent years. MySQL was free which made it one of the favourite choices for lab experiment, and many companies currently use MySQL as their database solution[25].

4.1.4 Sybase
Mark B. Hiffman and Robert Epstern established Sybase in 1984, and after three years, the Sybase database was realised. Sybase can run on Windows-based operating systems, Novell Netware environments and Unix-based operating systems. Currently, the most popular editions run on UNIX is Sybase 10 and Sybase 11 for SCO UNIX. Sybase is based on client-server architecture. Sybase can work with other types of databases in a system.

4.1.5 Serialisation and De-serialisation in .Net
Serialisation and de-serialisation is a technology in .Net. Serialisation records the status of an object and then stores that object’s status to a persistent storage place for future use. In contrast, de-serialisation restores the status of an object from a file such as XML. Currently most object-oriented programming languages, such as C# and Java, support this technology[26].

Serialisation and de-serialisation also support SQL-like queries such as add, update and delete in applications. These queries are accomplished by deserialising the whole object into a stream to perform an SQL-like query. After these SQL-like query methods have been written, a client can call these methods to perform an SQL query; however, this might become a problem when the file is growing. When an object is deserialised into a stream, other resources in same network cannot access it, and that resource is exclusively locked at that moment.

This method of processing an object also creates a performance issue. If the database is not extremely large huge, there will be no significant decrease in performance without the database's indexing capabilities compared to regular database management systems.
The key advantage of this technology is that it provides an alternative way to store data on the client side without much configuration work, which makes it perfect for clients who require light-weight databases with simple SQL query functions.

4.1.6 Database Decision

Because the peer-to-peer system will create a system for a company with a small budget, several factors should be considered. First, configuration and installation on the client side should be minimised. Second, the price of the database should be kept low because small companies usually do not have enough money to afford expensive software and prefer cheaper databases even if the performance suffers.

A comprehensive consideration of the database requirements of small companies, serialisation and de-serialisation are chosen in .Net for three reasons. First, serialisation and de-serialisation in .Net is a function of Microsoft Visual Studio, so company does not need to spend extra money if they want to use another database such as Oracle or Sybase. Second, there is no installation and configuration on the client side, which simplifies the work. Finally, the number of SQL query functions is limited, and database tables are simple.

4.2 Programming Language Comparison and Justification

4.2.1 C++

C++ originated from C, the C programming language was very popular because it is efficient. Then, Bjarne Stroustrup wanted to develop an object-oriented C language, so he modified the core of the C language to cater to the requirements of object-oriented programming[27]. Now, after several years of modification, C++ is a mature programming language. The C++ language is composed of two main parts, the core language and the C++ standard library. The core language is a set of keywords reserved for special purposes in a programming language. The C++ standard library is a modified version of the C library and the Standard Template Library[28].
4.2.2 Java
Java has been a popular programming language in recent years. Java is also an object-oriented programming language and is similar to C++. The three main features of object-oriented languages are encapsulation, inheritance and polymorphism. C++ supports both high-level and low-level programming features, which makes it an intermediate-level programming language. In contrast, Java focuses on the high-level and has limited low-level support. The Java programming language is less error prone compared to C++ because of its cross-platform features, decreased dependency on the platform and lack of pointers[29]. Java runs on top of the Java virtual machine, meaning that it needs to be compiled on the client side, which makes it slower. This is the only drawback, and Java has many advantages over C++ and Visual Basic because it is entirely free and adaptable to any operating system [30].

4.2.3 Microsoft Visual C#
Microsoft Visual C# is a Windows programming language with features similar to those of Java. Visual C# is a high-level programming language. “Visual” means it is easy to develop GUIs. Visual C# provides sufficient toolkits for the development of any type of application including individual small size applications, web-based applications, etc.[19]. Visual C# supports more features than Java including data types, functional programming and dynamic binding[31].

4.2.4 Programming Language Decision
After considering the drawbacks and advantages of these three programming languages, Microsoft Visual C# is chosen as the system’s programming language. The reasons for this decision are as follows:
1. C# is easier to learn. Due to time considerations, a rapid application approach is chosen.
2. C# has functions sufficient to develop a peer-to-peer file sharing system.
3. The company’s current software is written in C#, which will ease the future
integration and testing of the prototype in the current version of the software.

4.3 Hardware & Software Setup

Now a general statement about the hardware and software environment in the computing project will be given.

4.3.1 Hardware Requirements

Specification:
Processor: Intel Pentium III 800MHz
RAM: 630M SDRAM
Disk space: 40 GB (running in 15 GB partition)

4.3.2 Software Requirements

Specification:
Operating system: Microsoft Windows XP Professional
Application approach: Microsoft Visual Studio 2005
Database approach: Serialisation and de-serialisation in .Net with XML
Section 5 Implementation & Testing

The details of the database in the system have been discussed in the design section, and a suitable database and application approach are chosen in the solution proposal section. Obviously, it has been proved sufficiently for starting the implementation of the software. Although this is a confidential software development project, the implementation will still be discussed above the code-level. After the implementation section, a brief discussion about testing will be given.

5.1 Implementation

The following is a list of steps taken in the development of the peer-to-peer file sharing system.
- GUI design
- Database schema
- Required features
- How the system works

In the following, these steps will be discussed briefly because some of these steps have been discussed in-depth in the design section.

5.1.1 GUI Design

The purpose of GUI design is to provide a friendly interface for end users. In this stage, the following rule has been tried to follow: keep the interface as simple as possible while including essential information that will help to test the system in the testing phase.

The interface will display information for logging in, error messages, user authentication, etc. Logging in is mainly used for tracking how the system works, and
the log information will be printed on the console and written to a log file located on the node. Error messages will be printed on the console, and the system will break at that point. User authentication will show whether an end user has logged onto the system successfully or not.

When testing the system, it will be easy to determine whether all functional parts have behaved correctly on the console window.

5.1.2 Database Schema

In the earlier design stage, it is decided that the system would be composed of four tables; the details of each table and the relationships between tables are discussed in the design section. After deciding to use serialisation and de-serialisation in .NET with XML as the database choice, these four tables are made by creating four classes in C# code.

The four table classes defined are ACL, File, User and DeleteIp. The purpose of these table classes is to control access to other nodes in the system. The access control helps to ensure security in the network. The file table class contains a list of files that a node would like to download and upload. The record of table classes will be added/modified/deleted based on the job defined by the end user the media publishing network. The User table class contains the current node’s IP address. The purposes of this table class are to check whether a client is an authorised user and whether the client can access another specific resource on the network by comparing the table with the ACL table on the other side. The DeleteIp table class was designed to check whether a node in the network has disappeared. There is a predefined counter in the system, and if the number of times the node found to be unreachable after trying to connect to reaches the counter value, that node will be deleted from the network.

Additionally, SQL-like queries need to be written, such as add, delete and update, in C#. Because there are not many or complicated queries, it will not require much work to write each query separately in the system.
5.1.3 Required features

Designing GUIs provides me with a view of my future testing work. The core of the system is programming. Programming should carry out the functions needed by the system. This part occupied much more time than the rest of the project because I am not good at programming, and I had no prior knowledge of C#. The following is a list of the features that have been achieved in the system:

- Candidate Delete IP
- Candidate Supernode IP
- Fastest Download Node Selection
- MD5 Hash Attribution of Files
- Multithreading
- Scheduled Traffic Schemes
- Secure User Authentication
- Similar IP Finding
- Virtual Network Topology

For details about these features, please refer to Section 3’s function specification.

5.1.4 How the P2P system works

As the features required by the system have been described, now it is the time to illustrate how the system works.

1. System Deployment: Before the system could be run it had to be deployed as it would be during use on a public network. The Visual Studios is used to build executable applications for project implementation deployment.

2. After the previous step was completed, two executable applications were generated, the tracker server and node applications. Those two applications should be installed depending on the node's role in the network. There will be one tracker server and at least one node in the system.

3. When a client logs into a node through console for the first time, the client is required to type in a pre-defined password and to change that password afterwards.

4. After logging in successfully, the node will open its port and listen to any incoming requests from the tracker server.
5. When tracker server receives a scheduled task, it will first check which client needs to display that multimedia file by checking the IP address.

6. If the IP address can be found, the tracker server will then send a request to that client with a message that says "You need to display File A at time T". Once the node gets this message, it will try to find that file from the network.

7. If the file can be found on any node in the network, the file will be transferred from that node to the requested node. If the file cannot be found on any node in the system, the file will be transferred from the tracker server to the requesting node.

8. When file transfer is complete, the requesting node will check the file's MD5 checksum. If the file is identical to the transferred file, the node will tell the tracker server that it has already obtained the file.

9. The tracker server will then update its database to say that the requested node has received the file and that that node can be a contributor if there are any requests for that file. From that time, a normal node will become a supernode.

5.1.5 Classes Specification

In the following specification, a brief high-level description of what classes are on the client and server nodes will be given. Finally, several main functions in the system will be described here to cater for the prototype’s requirement.

5.1.5.1 Client side classes

Here is a figure containing all classes on client side:
LivefileClient.cs
LivefileClient is the main class for a node in this network to share files with others. Its main tasks are to share file between nodes, check end user’s authentication, control bandwidth and report information such as new node, new file to livefile server. Below are the main functions in the class:
- User authentication (login control)
- Bandwidth control
- Location control
- File downloading
- Network status update (new file, new user, new supernode, etc)
- XML database (ACL, File, User)

file.cs
File class defines what a file is in such system. A file is the object will be shared in such a system. It contains essential information including MD5 checksum to verify file integrity and a unique file name to identify it in the network.

```csharp
public class file
{
    [XmlElement(Nullable = false)]
    public string ip_address;
    [XmlElement(Nullable = false)]
    public string file_name;
    [XmlElement(Nullable = false)]
    public string md5;
}
```

message.cs
Message class defines a data packet will be sent to another node in such a network. The purpose of it is to make sure the other node is online before the sender shares information with it. It contains receiver’s IP address, how long it should wait to get the response and if the message has been received on the receiver side.
Figure 5.3 Message class

- ACL.cs
ACL is the abbreviation of access control list. ACL defines a XML database to store permitted receivers’ IP addresses for a sender. It helps to ensure network security.

```csharp
public class ACL
{
    [XmlAttribute(IsRequired = false)]
    public string allowed_ip;
}
```

Figure 5.4 ACL class

- user.cs
User class defines an end user in the network by XML database. In the database, it stores information such as user name, password, supernode option etc.

```csharp
public class user
{
    [XmlAttribute(IsRequired = false)]
    public string ip_address;
    public string user_name;
    public string password;
    public string supernode_ip_address;
    public bool supernode_option;
}
```

Figure 5.5 Login user information class

- clientThread.cs
Since nodes in such a system support multi-threading, clientThread class is defined to initialise a new thread between sender and receiver. It is a socket communication between two nodes, which is safer than an http connection.
The purpose of files class is to write SQL-like query for XML database in the system. It contains queries such as generate empty table, add, update or delete file in XML database etc.
MyPing class defines a data packet which will be sent to the receiver side to check its availability. It works similar as ping command.

```csharp
public class MyPing
{
    private static byte[] packet(ushort id)

    public static message ping(IPAddress dest_ip)
}
```

Figure 5.7 Files class

- myPing.cs

MyPing class defines a data packet which will be sent to the receiver side to check its availability. It works similar as ping command.

- ACLs.cs

The purpose of ACLs class is to write SQL-like query for XML database in the system. It contains queries such as add, update or delete ACL in XML database for a node.
Figure 5.9 ACLs class

- **users.cs**

The purpose of users class is to write SQL-like query for XML database in the system. It contains queries such as add, update or delete users in XML database for a node.

Figure 5.10 Users class

- **throttledStream.cs**
Download speed control is an important feature in a network when there is high network traffic. By defining throttledStream class, a node will be able to control its download speed by letting the downloading thread sleep periodically.

```
public class throttledStream : Stream
{
    private Stream stream;
    private long max_bps;
    private long byte_count;
    private long start;

    // get & set values for throttledStream
    public throttledStream(Stream raw_stream, long limited_bps) ... |

    // functions of throttledStream (override methods of Stream)

    public throttledStream(Stream raw_stream, long limited_bps) ... 

    protected void Throttle(int bufferSizeInBytes) ... 

    protected void Reset() ... 
}
```

**Figure 5.11** throttledStream class

- fileMD5.cs
The purpose of fileMD5 class is to ensure file integrity by storing a file’s MD5 checksum value in XML database.

```
public class fileMD5
{
    public string file_name;
    public string md5;
}
```

**Figure 5.12** fileMD5 class

- icmp_type.cs
This class defines Internet Control Message Protocol (ICMP) type code. The purpose of ICMP code is to generate response code in order to know the status on the receiver side.

```
public enum icmp_type
{
    echo = 8,
    echo_reply = 0,
    time_exceeded = 11,
}
```
5.1.5.2 Server side classes

Here is a figure containing all classes on server side:

![Server node classes](image)

**Figure 5.14 Server node classes**

- **LivefileServer.cs**
  LivefileServer is the main class for a node which has control ability to the other client nodes in this network. Its main tasks are to ensure files’ integrity in the network, recommend for a supernode to a client node in case its supernode is down. Below are the main functions in the class:
  - File checking
  - Look for supernode in network
  - Inform client node about network status
  - Open and listen on port

- **ipGap.cs**
  This class decides if two IP addresses are geographically close to each other. It helps the network to predict if the transfer speed will be good between these two nodes. However, there is no guarantee on the transfer speed.

```csharp
/// <summary>
/// get the gap value between two ips #lfs#
/// </summary>
class ipGap
{
    public long gap_value;
    public string ip;
}
```

**Figure 5.15 ip gap class**

- **ipGapComparer.cs**
  This class extends IComparer checking similarity on two IP addresses. It is the class
will be called by ipGap class.

```csharp
// <summary> ...
public class ipGapComparer : IComparer
{
    int IComparer.Compare(object a, object b)...
}
```

**Figure 5.16 IComparer class**

- deleteIps.cs

The purpose of deleteIps class is to write SQL-like delete query for storing already deleted IP address in case some client nodes need to reconnect to them again.

```csharp
[Serializable]
public class deleteIps
{
    [XmlIgnoreAttribute()]
    private XmlSerializer mySerializer;

    [XmlIgnoreAttribute()]
    private object critical_section = new object();
    [XmlArray("deleteIp_table"), XmlArrayItem("ip_counter", typeof(deleteIp))]
    public ArrayList array_list = new ArrayList(); //to hold user_table's elements("ip_counter")

    // <summary> ...
    public static void generateEmptyTable()...

    // <summary> ...
    public static bool deleteIpCounter(FileInfo xmlfile, string ip_address)...

    // <summary> ...
    private deleteIps DeserialiseDeleteIp(FileInfo xmlfile)....

    // <summary> ...
    private void SerializeDeleteIp(deleteIps deleteIps, FileInfo xmlfile)...
}
```

**Figure 5.17 delete IP class**

### 5.1.6 Candidate Delete IP

This function is called when a node is not reachable. There is a table called ‘deleteIps.xml’ on the tracker server. This table’s attributes include IP addresses and counters. The inability to reach a node indicates that the node is dead or temporarily blocked. In the latter situation, the node should not simply be deleted. Thus, a counter for each IP has been used in the network. When a ‘not reachable’ signal is received on the server side, this counter will increment by one. When the counter reaches the maximum delete request, the node will be deleted from the network. The max value is
defined as three in the experimental network, which has only five nodes, and the interval for attempting to reach a node was defined as one minute. These values can be changed based on the scale and reliability of the network.

**Figure 5.18** How delete IP works

**5.1.7 Candidate Supernode IP**

This function is similar to the ‘candidate delete IP’ in some respects. In the system, the candidate supernode IP has a list of candidate supernodes. If one of these supernodes fails, the child node can still connect to other nodes.

**Comment:** If Node A’s supernode A is unreachable, it will try to connect to another supernode, in this case supernode B.

**Figure 5.19** How to change supernode
5.1.8 Fastest Download Node Selection

If there are several nodes in the network that hold identical copies of a file, the decision must be made regarding where the file should be obtained from. This problem will be solved by sending ICMP packets to check response times. This process can be called my own version of ‘Ping’.

Comment: If Node A requests a file, which Node B, Node C and Node D all have, it will check which one has the fastest download speed by Ping all these three nodes. And then, Node A will download the file from the closest Node (Node C)

Figure 5.20 Download from fastest node

5.1.9 MD5 Hash Attribute of File

This function is used to check file integrities. When a file is transferred over the network, a 128-bit MD5 hash string (md5_s) will be sent from the sender to the receiver with the file’s content. The string is calculated by the MD5 algorithm for each file. Once the receiver receives the entire file and the md5_1, a local version of the MD5 hash string will be calculated (md5_r). If the md5_s is equal to the md5_r, we can say that the files are the same.

Here is a brief explanation on how it works:

- md5: Input is a variable length of information and output is a fixed length of
128-bit has string.

- **Sender:** Send file a.mpg with md5 code ‘md5_s’ calculated by the file.
- **Receiver:** Receive the file with its remote (sender's) md5 code (md5_r).
- Calculated hash code of the file received ‘md5_r’.
- md5_s = md5_r ? if yes, download succeed. Else download fails, retry or download the file from another node.

**Figure 5.21** File integrity check by MD5 hash

**5.1.10 Multithreading**

Because the tracker server may need to handle several requests at the same time, the support of multithreading is required. Via multithreading, each request will be dispatched to a thread and there will be a back-end cleaning thread that regularly checks the status of each thread. Once the back-end cleaning thread finds the thread’s status is set to finish, it will erase this thread and its relevant resource.
5.1.11 Scheduled Traffic Schemes

We know that peer-to-peer sharing will consume nearly all available bandwidth if there is no traffic control mechanism. The situation is exacerbated if peer-to-peer sharing occurs during peak times; i.e., between 9 am and 5 pm during weekdays. To overcome this problem, a traffic control function will be added to limit the download speed if the current date and time is with the peak times. If it is a peak time, this function will be enabled, and the transfer speed will be limited to a predefined download speed. 30KB/S is used as the transfer speed limit in the experimental network.
5.1.12 Secure User Authentication

This function checks whether the current user is allowed to enter into the network. Besides that, login user’s access level will be determined by the user name and password provided. In the system, there are two types of user, administrator and end user. If a client logs in the administrator name “Admin” and types a correct password, administrator privileges will be granted to that client. As an administrator, the client can add, delete and modify users and files, access the control list of the system and check the system log. In the project, control based on the user will not be implemented because the media publishing network already has a user control method that the peer-to-peer functionality can reuse.

Figure 5.23 Downloading speed limitation
5.1.13 Virtual Network Topology

Because I have previously worked as a network administrator, the importance of defining the virtual network topology of a company is realised to ensure network security. Depending on the different roles of the users, different access rights should be granted. For example, a software developer’s network should not access the company’s financial information. Thus, the system can define the nodes that a specific node can access. This restriction improves the security of the system.

**Figure 5.25** Network control by ACL

**Step 1:**
Node A requests a.mp from Node B.

**Step 2:**
Node B is in Node A’s ACL and the file can be transferred to Node A.
5.2 Testing

Testing consists of many parts, such as testing whether the system can meet the company’s requirements and verifying the system’s data recovery capabilities. Some people may not think that testing is important and that it is quite simple compared to design and implementation, but this is not true. The testing of the system will be discussed in the following two sections.

5.2.1 Test Plan

Before a test is actualised, it is useful to have a test plan. The plan should state the following:

- Objectives
- Software description
- List of functions to be tested
- Test tools to be used

To test the functions, what should be achieved after by the completion of each function needs to be understood. To gain a clear idea of this issue, a description of how the software works should be created and followed during testing.

Testing consists of both automatic and manual testing depending on the function. For functions such as user authentication, manual testing is more suitable. In contrast, for database-related operations and file transferring, it is better to write function test cases. Specifically, all functions should be tested separately and then an integration test that encompasses other functions in the network should be carried out to determine whether all functions are working properly. Because this is a prototype, attention to non-function-related testing such as stability and performance testing will not be given. These parts of testing should be carried out after the peer-to-peer system has been integrated into the main system. Testing tool selection is also a crucial part of software testing, and suitable testing tools for each function can help improve efficiency and reduce costs.

The peer-to-peer system of this project has been already described; thus, it is not
difficult to compile a description of the projects functionality. Furthermore, both automatic and manual test cases will be designed and run on top of the peer-to-peer system after a suitable testing tool from the market is selected.

The testing part will be achieved in the following sections will be described.

5.2.2 Objectives of Testing
The objectives of the testing are quite simple and are clearly stated in the master’s project description. These objectives are as follows:

- Ensuring the system meets the media publishing software company’s requirements. For confidentiality reasons, the company's name cannot be stated here.
- Testing the functions have been implemented in the subsystem.

5.2.3 Software description
The software requirements are clearly stated in this part, and a detailed description of the features that should be included in peer-to-peer functionality is also provided. The software description will be used as an input to testing and is an important part of write appropriate test cases.

5.2.4 Testing tool selection
To select the correct testing tool, it is essential to think about what should be tested. My testing of this subsystem should be performed on both the functional and system levels. Because access to the other code of the system is not granted, only functional tests can be performed. Csunit[32] is chosen and it is a popular Xunit testing framework, to perform these functional tests. Additionally, some manual testing that mainly concerned the UI parts such as login is performed.

5.2.5 Testing method
A bottom-up testing approach is adopted because the functions written were on the low-level. The rationale behind this choice is that the system requirements are always
changing, and it is difficult to begin testing activities from the top.

5.2.6 List of functions to be tested

After learning how the peer-to-peer functionality will work in the system, it became time to list the functions that needed to be tested and write the corresponding test cases. The details will be discussed in the following section.

5.2.7 Details of Testing

The different parts of the testing evaluated different aspects of the system’s functions. Bearing in mind the testing’s objectives, testing was carried out based on several main functions of the system, and the boundary-value problem in the system was also tested when the system’s functions were tested.

The operations tested operations in the system are as follows:

- Login system: ensure sure only authorised users can login to the system and that different users have different privileges in the operating system.
  - Operations: execute the application and login with a predefined password.
  - Expected result: a client can log into the system if a correct password is entered.

- When a node is not found in the system a pre-defined number of times, the node will be deleted from the network.
  - Operations: a client requests a file and discovers that the file is located on Node A with IP address I, a request message is sent to Node A. However, Node A is currently unavailable.
  - Expected result: the counter will be incremented by one on the tracker server, and, if the counter has reached the maximum allowed number, Node A will be removed from the network until it becomes available again.

- A node will download a file from the node that provides the fastest download speed.
  - Operations: a client sends a request to the tracker server. If more than one
supernode holds the requested file, the tracker server will ask the client to ping all of those nodes.

- Expected result: The supernode with the fastest transferring speed will be selected to transfer the file.

- When a file is downloaded from a sender, the receiver will determine whether the file is the same as sender’s copy.
  - Operations: along with the file, an MD5 checksum code will be sent from the supernode to the client. The client will call the hash function locally on the received file and compared it with the MD5 checksum.
  - Expected result: if the MD5 checksum generated locally is the same as the one received from the supernode, the files are the same, and the download action is completed successfully.

- Each node can handle several tasks simultaneously.
  - Operations: if a client needs to perform more than two tasks simultaneously, a separate thread will be created for each task.
  - Expected result: each task should be performed successfully without disturbing other tasks.

- The download speed can be limited to a predefined value.
  - Operations: the application supports limited transferring speeds. This limitation will normally will be enabled during peak times; i.e., working hours during working days.
  - Expected result: transfers speeds will be decreased from unlimited to the predefined rate.
Section 6 Conclusion

Now that the project is almost finished, it is the time to look back at what have been done and what can be improved in the future.

6.1 Functional Achievements

As mentioned in section 1, the aim of this project was to develop a prototype peer-to-peer file sharing function for an existing media distribution software. To meet the requirements of the system, several functions need to be implemented. SSL is a protocol that was originally developed by Netscape. This protocol encrypts messages sent between two nodes in a network. SSL was designed to create a VPN among nodes in a network. Without SSL, data transferred over network can be captured and modified by hackers, which could potentially lead to security issues for a business network solution. Maximum thread control is a mechanism for controlling the maximum allowed number of threads on each node. Limiting the number of threads can prevent potential issues such as performance and hardware failures. Currently, Secure Sockets Layers (SSL) and maximum thread controls are not implemented in the system. The system-to-system variation in appropriate thread limits is too large to determine an appropriate thread limit for the system. Although the MD5 checksum method has been implemented in the subsystem, I insist that other security mechanisms should also be implemented to secure the network. A performance test needs to be designed and carried out to measure the maximum allowable thread number, but this is outside of the current project’s scope.

All other features were implemented successfully, indicating that all functions were achieved in the system. The system’s database was also created successfully by persisting objects into the database via XML serialisation. Connecting the database to the system enabled the implementation of several useful functions in the system. These
functions include candidate IP selection, fastest download node selection, MD5 hash, download speed control and user authentication.

6.2 Future Improvements

6.2.1 Functional Improvements
Although the system is only a prototype, its architecture is amenable to future improvement. In the system architecture design, a two-tier architecture that is a widely used architecture in system development is used. Further, the programming language, Microsoft Visual C#, has a friendly interface for creating additional functions in an existing system. Users can create additional function with minimal amounts of codes.

6.2.2 Security Considerations
Although a login console and the MD5 checksum are used in this system, it is still felt that the system is insufficiently safe. Login consoles require all clients to enter a password to log into the system for the first time. The MD5 file checksum checks data on the receiving side. However, all data transferred through network is not encrypted. Hackers can intercept streaming data and modify that data if they so choose. The simplest way to ensure the online safety of the system is to ask the ISP (internet service provider) to provide a VPN, but companies have to pay annual or monthly fees for this service. Another solution is the use of encryption, but much more knowledge of math and algorithms would be required to achieving that.

6.3 Learning Outcomes
A lot has been learnt from this project, which was certainly the aim of the project. During the time working on this project, some prior knowledge are used and new technical and non-technical knowledge are gained. The most significant knowledge gained concerned the steps of developing a system; many more aspects should be considered than I previously believed. Additionally, the importance of research in developing a system has been aware of; research can help me to find current problems in related fields of study. Additionally, this project helped me to increase my
programming skills because more tools are used in the design of this system than in any systems designed previously.
# Appendix A: Table of Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1</td>
<td>How the company's MDN works</td>
<td>9</td>
</tr>
<tr>
<td>1.2</td>
<td>Supported content file formats</td>
<td>10</td>
</tr>
<tr>
<td>2.1</td>
<td>Central index algorithm</td>
<td>17</td>
</tr>
<tr>
<td>2.2</td>
<td>Flooded requests algorithm</td>
<td>18</td>
</tr>
<tr>
<td>2.3</td>
<td>Distributed Hash Table (DHT) Algorithm</td>
<td>19</td>
</tr>
<tr>
<td>2.4</td>
<td>UDP hole punching</td>
<td>21</td>
</tr>
<tr>
<td>2.5</td>
<td>How a VPN crosses Internet</td>
<td>23</td>
</tr>
<tr>
<td>3.1</td>
<td>Two-tier system architecture</td>
<td>33</td>
</tr>
<tr>
<td>5.1</td>
<td>Client node classes</td>
<td>49</td>
</tr>
<tr>
<td>5.2</td>
<td>File class</td>
<td>50</td>
</tr>
<tr>
<td>5.3</td>
<td>Message class</td>
<td>51</td>
</tr>
<tr>
<td>5.4</td>
<td>ACL class</td>
<td>51</td>
</tr>
<tr>
<td>5.5</td>
<td>Login user information class</td>
<td>51</td>
</tr>
<tr>
<td>5.6</td>
<td>Client thread class</td>
<td>52</td>
</tr>
<tr>
<td>5.7</td>
<td>Files class</td>
<td>53</td>
</tr>
<tr>
<td>5.8</td>
<td>myPing class</td>
<td>53</td>
</tr>
<tr>
<td>5.9</td>
<td>ACLs class</td>
<td>54</td>
</tr>
<tr>
<td>5.10</td>
<td>Users class</td>
<td>54</td>
</tr>
<tr>
<td>5.11</td>
<td>throttledStream class</td>
<td>55</td>
</tr>
<tr>
<td>5.12</td>
<td>fileMD5 class</td>
<td>55</td>
</tr>
<tr>
<td>5.13</td>
<td>ICMP class</td>
<td>55</td>
</tr>
<tr>
<td>5.14</td>
<td>Server node classes</td>
<td>55</td>
</tr>
<tr>
<td>5.15</td>
<td>ip gap class</td>
<td>56</td>
</tr>
<tr>
<td>5.16</td>
<td>IComparer class</td>
<td>57</td>
</tr>
<tr>
<td>5.17</td>
<td>delete IP class</td>
<td>57</td>
</tr>
<tr>
<td>5.18</td>
<td>How delete IP works</td>
<td>58</td>
</tr>
<tr>
<td>5.19</td>
<td>How to change supernode</td>
<td>58</td>
</tr>
<tr>
<td>5.20</td>
<td>Download from fastest node</td>
<td>59</td>
</tr>
<tr>
<td>5.21</td>
<td>File integrity check by MD5 hash</td>
<td>60</td>
</tr>
</tbody>
</table>
Figure 5.22 Multithreading support on tracker server ————————————————————61
Figure 5.23 Downloading speed limitation ————————————————————62
Figure 5.24 Simple login window ————————————————————63
Figure 5.25 Network control by ACL ————————————————————63
Appendix B: Index of Tables

Table 3.1 Entity Required---------------------------------------------36
Table 3.2 Entity Relationship Types------------------------------------37
Table 3.3 Entity Relationship Diagram----------------------------------37
Table 3.4 Create Attribute List for Each Entity Type--------------------37
Table 3.5 Attribute Domains--------------------------------------------38
Table 3.6 Entity Key Identification------------------------------------39
Table 3.7 Logical Database Model as DBDL-----------------------------39
Appendix C: Bibliography


19. Microsoft, MSDN Library, 2003,


22. Oracle, Oracle Technology Global Price List, 2013,

23. Where to type SQL statements in Microsoft Access, 2007,
http://www.jaffainc.com/SQLStatementsInAccess.htm

24. DCDB Group, Comparison of Oracle, MySQL and Postgres DBMS, 2006
http://dcdbappl1.cern.ch:8080/dcdb/archive/ttraczyk/db_compare/db_compare.html#Basic+data+types


26. Wikipedia, Serialization, 2013,
http://en.wikipedia.org/wiki/Serialization#Programming_language_support


30. Martin Schöberl, Java and the JVM, 2007,
http://www.jopdesign.com/teaching/JavaJVM.pdf

31. Wikipedia, Comparison of C Sharp and