CHAPTER 1

INTRODUCTION

This chapter discusses briefly about the recommendation applications and its various aspects. In this chapter we have discussed about existing system, problems in existing system and about challenges in building recommendation system for LAN.

1.1 INTRODUCTION TO LAN

The increasing demand and use of computers in universities and research labs in the late 1960s generated the need to provide high-speed interconnections between computer systems. The development and proliferation of personal computers using the CP/M operating system in the late 1970s, and later DOS-based systems starting in 1981, meant that many sites grew to dozens or even hundreds of computers. The initial driving force for networking was generally to share storage and printers, which were both expensive at the time. There was much enthusiasm for the concept and for several years, from about 1983 onward, computer industry pundits would regularly declare the coming year to be "the year of the LAN".

A **local area network (LAN)** is a computer network that interconnects computers in a limited area such as a home, school, computer laboratory, or office building using network media. The defining characteristics of LANs include their usually higher data-transfer rates, smaller geographic area, and lack of a need for leased telecommunication lines. A LAN is a high-speed data network that covers a relatively small geographic area. It typically connects workstations, personal computers, printers, servers, and other devices. LANs offer computer users many advantages, including shared access to devices and applications, file exchange between connected users, and communication between users via electronic mail and other applications.

A common scenario in LAN is difficulty in proper management of shared files in LAN. For example every time a person adds a new file to share in an already shared folder others are not notified of the new file being shared. Also there is currently no system for LAN that recommends content to users on the basis of popularity and personal preferences.

1.2 INTRODUCTION TO RECOMMENDATION SYSTEM

The abundance of information available on the Web and in Digital Libraries, in combination with their dynamic and heterogeneous nature, has determined a rapidly increasing difficulty in finding what we want when we need it and in a manner which best meets our requirements.
As a consequence, the role of user modeling and personalized information access is becoming crucial: users need a personalized support in sifting through large amounts of available information, according to their interests and tastes.
Recommender systems can now be found in many modern applications that expose the user to a huge collection of items. Such systems typically provide the user with a list of recommended items they might prefer, or predict how much they might prefer each item. These systems help users to decide on appropriate items, and ease the task to find preferred items in the collection.

For example, the DVD rental provider **Netflix** displays predicted ratings for every displayed movie in order to help the user decide which movie to rent. The online book retailer **Amazon** provides average user ratings for displayed books, and a list of other books that are bought by users who buy a specific book. **Microsoft** provides many free downloads for users, such as bug fixes, products and so forth. When a user downloads some software, the system presents a list of additional items that are downloaded together. All these systems are typically categorized as recommender systems, even though they provide diverse services.

Systems implementing a content-based recommendation approach analyze a set of documents and/or descriptions of items previously rated by a user, and build a model or profile of user interests based on the features of the objects rated by that user. The profile is a structured representation of user interests, adopted to recommend new interesting items. The recommendation process basically consists in matching up the attributes of the user profile against the attributes of a content object. The result is a relevance judgment that represents the user's level of interest in that object.

The Content Recommendation systems seek to predict the 'rating' or 'preference' that a user would give to an item(**Music, eBooks, Videos, documents**) and provide an user interface which enables the users to view the newly available contents over the LAN and the content previously liked by user in the past.

1.3 GOALS AND OBJECTIVES:

- To build a system which will overcome all limitations in LAN regarding the recommendation of distributed content and minimizes user's efforts to provide feedback.
- To maximize the user's satisfaction by displaying appropriate content at the right time.
- System involves collection of user preferences for the available content and to learn user behaviour from it.
- Provides recommendations to users on the basis of their behaviour & content popularity and also provides searching mechanism.

CHAPTER 2

REVIEW OF LITERATURE

Recommender systems or **recommendation systems** are a subclass of information filtering system that seek to predict the 'rating' or 'preference' that user would give to an item (such as music, books, or movies) or social element (e.g. people or groups) they had not yet considered, using a model built from the characteristics of an item or the user's social environment .

Recommender systems have become extremely common in recent years. A few examples of such systems:

- When viewing a product on Amazon.com, the store will recommend additional items based on a matrix of what other shoppers bought along with the currently selected item.
- Pandora Radio takes an initial input of a song or musician and plays music with similar characteristics (based on a series of keywords attributed to the inputted artist or piece of music). The stations created by Pandora can be refined through user feedback (emphasizing or deemphasizing certain characteristics).
- Netflix offers predictions of movies that a user might like to watch based on the user's previous ratings and watching habits (as compared to the behavior of other users), also taking into account the characteristics (such as the genre) of the film.

2.1 OVERVIEW

Recommender systems typically produce a list of recommendations in one of two ways - through collaborative or content-based filtering. **Collaborative filtering** approaches to build a model from a user's past behavior (items previously purchased or selected and/or numerical ratings given to those items) as well as similar decisions made by other users, then use that model to predict items (or ratings for items) that the user may have an interest in. **Content-based filtering** approaches utilize a series of discrete characteristics of an item in order to recommend additional items with similar properties.

The differences between collaborative and content-based filtering can be demonstrated by comparing two popular music recommender systems - Last.fm and Pandora Radio.

- **Pandora** uses the properties of a song or artist (a subset of the 400 attributes provided by the Music Genome Project) in order to seed a "station" that plays music with similar properties. User feedback is used to refine the station's results, deemphasizing certain attributes when a user "dislikes" a particular song and emphasizing other attributes when a user "likes" a song. This is an example of a content-based approach.
- **Last.fm** creates a "station" of recommended songs by observing what bands and individual tracks that the user has listened to on a regular basis and comparing those against the listening behavior of other users. Last.fm will play tracks that do not appear in the user's library, but are often played by other users with similar interests. As this approach leverages the behavior of users, it is an example of a collaborative filtering technique.

Each type of system has its own strengths and weaknesses. In the above example, Last.fm requires a large amount of information on a user in order to make accurate recommendations. This is an example of the cold start problem, and is common in collaborative filtering systems. While Pandora needs very little information to get started, it is far more limited in scope (for example, it can only make recommendations that are similar to the original seed).

Recommender systems are a useful alternative to search algorithms since they help users discover items they might not have found by themselves. Interestingly enough, recommender systems are often implemented using search engines indexing non-traditional data.

Montaner provides the first overview of recommender systems, from an intelligent agents perspective. Adomavicius provides a new overview of recommender systems. Herlocker provides an additional overview of evaluation techniques for recommender systems.

Recommender system is an active research area in the data mining and machine learning areas. Some conferences such as RecSys, SIGIR, KDD have it as a topic.

2.2 CONTENT-BASED FILTERING

Content-based filtering methods are based on information about and characteristics of the items that are going to be recommended. In other words, these algorithms try to recommend items that are similar to those that a user liked in the past (or is examining in the present). In particular, various candidate items are compared with items previously rated by the user and the best-matching items are recommended. This approach has its roots in information retrieval and information filtering research.

Basically, these methods use an item profile (i.e., a set of discrete attributes and features) characterizing the item within the system. The system creates a content-based profile of users based on a weighted vector of item features. The weights denote the importance of each feature to the user and can be computed from individually rated content vectors using a variety of techniques. Simple approaches use the average values of the rated item vector while other sophisticated methods use machine learning techniques such as Bayesian Classifiers, cluster analysis, decision trees, and artificial neural networks in order to estimate the probability that the user is going to like the item.

Direct feedback from a user, usually in the form of a like or dislike button, can be used to assign higher or lower weights on the importance of certain attributes (using Rocchio Classification or other similar techniques).

A key issue with content-based filtering is whether the system is able to learn user preferences from user's actions regarding one content source and use them across other content types. When the system is limited to recommending content of the same type as the user is already using, the value from the recommendation system is significantly less than when other content types from other services can be recommended. For example, recommending news articles based on browsing of news is useful, but it's much more useful when music, videos, products, discussions etc. from different services can be recommended based on news browsing. Such cross-content recommendation has been productised by Leiki.

As previously detailed, Pandora Radio is a popular example of a content-based recommender system that plays music with similar characteristics to that of a song provided by the user as an initial seed. There are also a large number of content-based recommender systems aimed at providing movie recommendations, a few such examples include Rotten Tomatoes, Internet Movie Database, Jinni, Rovi Corporation.

2.3 MOBILE RECOMMENDER SYSTEMS

One growing area of research in the area of recommender systems is mobile recommender systems. With the increasing ubiquity of internet-accessing smart phones, it is now possible to offer personalized, context-sensitive recommendations. This is a particularly difficult area of research as mobile data is more complex than recommender systems often have to deal with (it is heterogeneous, noisy, requires spatial and temporal auto-correlation, and has validation and generality problems). Additionally, mobile recommender systems suffer from a transplantation problem - recommendations may not apply in all regions (for instance, it would be unwise to recommend a recipe in an area where all of the ingredients may not be available).

One example of a mobile recommender system is one that offers potentially profitable driving routes for taxi drivers in a city. This system takes as input data in the form of GPS traces of the routes that taxi drivers took while working, which include location (latitude and longitude), time stamps, and operational status (with or without passengers). It then recommends a list of pickup points along a route that will lead to optimal occupancy times and profits. This type of system is obviously location-dependent, and as it must operate on a handheld or embedded device, the computation and energy requirements must remain low.

Mobile recommendation systems have also been successfully built using the Web of Data as a source for structured information. A good example of such system is SMARTMUSEUM. The system uses semantic modelling, information retrieval and machine learning techniques in order to recommend contents matching user’s interest, even when the evidence of user's interests is initially vague and based on heterogeneous information.

2.4 THE NETFLIX PRIZE

One of the key events that energized research in recommender systems was the **Netflix prize**. From 2006 to 2009, Netflix sponsored a competition, offering a grand prize of $1,000,000 to the team that could take an offered dataset of over 100 million movie ratings and return recommendations that were 10% more accurate than those offered by the company's existing recommender system. This competition energized the search for new and more accurate algorithms. On 21 September 2009, the grand prize of US$1,000,000 was given to the BellKor's Pragmatic Chaos team.

The most accurate algorithm in 2007 used an ensemble method of 107 different algorithmic approaches, blended into a single prediction:

Predictive accuracy is substantially improved when blending multiple predictors. Our experience is that most efforts should be concentrated in deriving substantially different approaches, rather than refining a single technique. Consequently, our solution is an ensemble of many methods.

Many benefits accrued to the web due to the Netflix project. Some teams have taken their technology and applied it to other markets, such as 4-Tell, Inc.'s Netflix project-derived solution for ecommerce websites.

2.5 PROBLEM STATEMENT

To develop a system that will recommend the content after learning the behaviour, activities and preferences of user and to predict what users will like based on their similarity to other users.This chapter explains the problems in designing and implementing Content Recommendation System for LAN.

The System is developed in four main Modules -

1- Distributed System Module

2- Data Collection Module
3- Training Module
4- User Interface Module

It also contains three smaller modules (Searching & Sorting, Web Server and File Info Database)

There are various problems which are specific to a particular module,

- Time efficiency for retrieval of shared files in LAN is very crucial, so making it efficient is one major issue specific to Searching.
- To retrieve other systems information after pinging them through a system and to make the retrieval of information efficient is a problem specific to Distributed System.
- To maintain the scalability of the system on increasing the number of resources is specific to Distributed System.
- Distribution of data among computers and Error handling i.e., if network fails , then data to be collected may be lost etc. are specific to Data Collection Module.

2.6 SOLUTION TO PROBLEMS

Development and implementation of successful distributed system requires one exercise "caution and sound planning". Simplicity of Distributed System will be maintained and Synchronization and logging is used for error handling. To decrease the complexity of searching it is implemented using single system only.

CHAPTER 3

MATERIALS AND METHODS

3.1 PROCESS MODEL

We have used spiral model for our project. The spiral model is a software development process combining elements of both design and prototyping-in-stages, in an effort to combine advantages of top-down and bottom-up concepts. Also known as the spiral lifecycle model (or spiral development), it is a systems development method (SDM) used in information technology (IT). This model of development combines the features of the prototyping and the waterfall model.

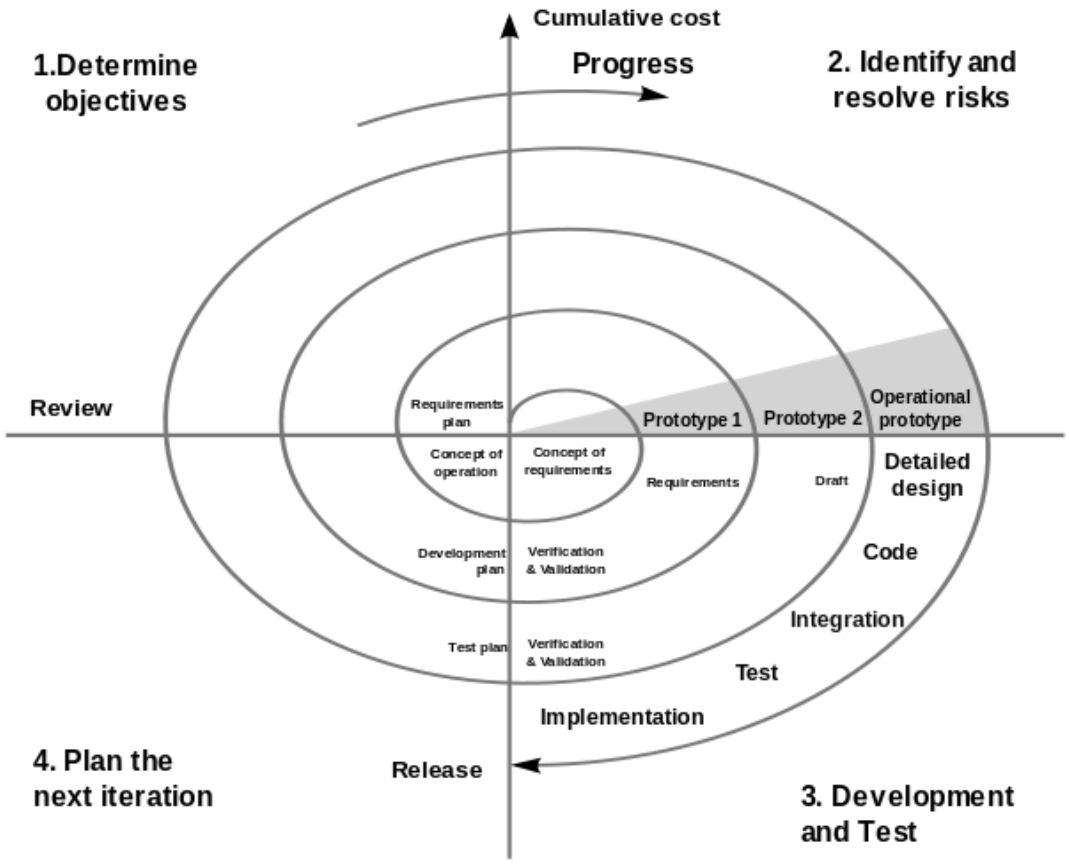

Fig.3.1 Spiral Model

3.2 REQUIREMENT ANALYSIS ANDFEASIBILITY STUDY

Requirements analysis is a software engineering task that bridges the gap between system level requirements engineering and software design. Requirements engineering activities result in the specification of software's operational characteristics (function, data, and behaviour), indicate software's interface with other system elements, and establish constraints that software must meet. Requirements analysis allows the software engineer sometimes called analyst in this role) to refine the software allocation and build models of the data, functional, and behavioural domains that will be treated by software. Requirements analysis provides the software designer with a representation of information, function, and behaviour that can be translated to data, architectural, interface and component-level designs. Finally, the requirements specification provides the developer and the customer with the means to assess quality once software is built.

Software requirements analysis may be divided into five areas of effort:
(1) Problem recognition
(2) Evaluation and synthesis
(3) Modelling
(4) Specification
(5) Feasibility Study

3.2.1 PROBLEM RECOGNITION

Initially, the analyst studies the System Specification (if one exists) and the Software Project Plan. It is important to understand software in a system context and to review the software scope that was used to generate planning estimates. Next, communication for analysis must be established so that problem recognition is ensured. The goal is recognition of the basic problem elements as perceived by the customer/users.

The analyst must define all externally observable data objects, evaluate the flow and content of information, define and elaborate all software functions, understand software behaviour in the context of events that affect the system, establish interface characteristics, and uncover additional design constraints. Each of these tasks serves to describe the problem so that an overall approach or solution may be synthesized.

The basic problems which we have recognized which lead to the Development of this tool are:

PROBLEM 1: It's very difficult and time consuming to find appropriate content according to one's choices.

PROBLEM2:There is no provision to find which content is popular (having more views) in LAN

PROBLEM3:Proper management of shared is difficult because of huge amount of shared data present in LAN.

3.2.2 EVALUATION AND SYNTHESIS

Problem evaluation and solution synthesis is the next major area of effort for analysis. The analyst must define all externally observable data objects, evaluate the flow and content of information , define and elaborate all software functions, understand software behavior in the context of events that affect the system, establish system interface characteristics, and uncover additional design constraints. Each of these tasks serves to describe the problem so that an overall approach or solution may be synthesized.

3.2.3 MODELING

During the evaluation and solution synthesis activity, the analyst creates models of the system in an effort to better understand data and control flow, functional processing, operational behaviour and information content. The model serves as a foundation for software design and as the basis for the creation of specifications for the software.

3.2.4 USE CASE DIAGRAM

Actor

We can picture an actor as a user of the recommendation system. Because individual persons are irrelevant for the model, they are abstracted. So the actors are called “check-in employee” or “passenger”. Actors represent roles that users take on when they use the recommendation system, e.g., the role of a recommending client or administrator. It is important for the recommendation system in which role a person is acting i.e., whether as a client or as an administrator. In each case access to the appropriate functionalities (use cases) is granted.In our recommendation system for LAN, there are two actors i.e., Server(Admin) and the client.

Use Case

Use cases describe the interactions that take place between actors and recommendation system during the execution of processes. A use case represents a part of the functionality of there commendation system and enables the user (modelled as an actor) to access this functionality. Anything that users would like to do with Recommendation System on LAN has to be made available as a use case (or part of a use case). Functionalities that exist in this system, but that are not accessed by means of use cases, are not available to users. Even though the idea behind use cases is to describe interactions, flows of batch processing, which generally do not include interactions, can also be described as use cases. The actor of such a batch use case is then the one who initiates batch processing.

Association

An association is a connection between an actor and a use case. An association indicates that an actor can carry out a use case. Several actors at one use case mean that each actor can carry out the use case on his or her own and not that the actors carry out the use case together. According to UML, association only means that an actor is involved in a use case. We use associations in a restricted manner.

Include Relationships

An include relationship is a relationship between two use cases. It indicates that the use case to which the arrow points is included in the use case on the other side of the arrow. This makes it possible to reuse a use case in another use case.

Use Case Diagram

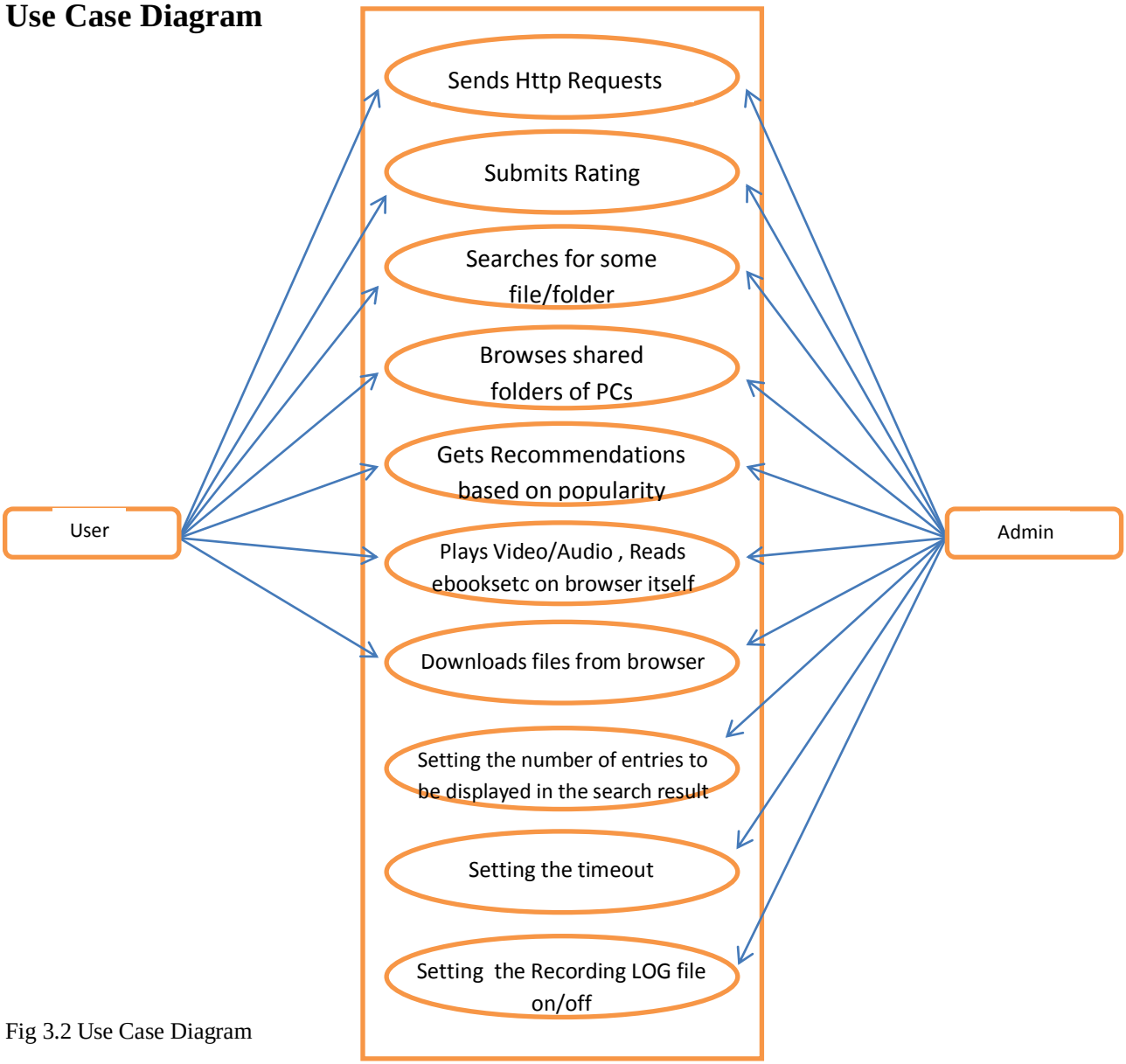

Fig 3.2 Use Case Diagram

3.2.5 SPECIFICATION

3.2.5.1 Hardware Specification

--System with minimum 512MB RAM

3.2.5.2 Software Specification

-- Eclipse Ide

--Notepad++

-- Firefox

-- VLC plugin

-- JDK 1.7

--Jcifs library

3.2.5.2.1 Eclipse IDE

In computer programming, Eclipse is a multi-language software development environment comprising a base workspace and an extensible plug-in system for customizing the environment. It is written mostly in Java. It can be used to develop applications in Java and, by means of various plug-ins, other programming languages including Ada, C, C++, COBOL, Fortran, Haskell, JavaScript, Perl, PHP, Python, R, Ruby (including Ruby on Rails framework), Scala, Clojure, Groovy, Scheme, and Erlang. It can also be used to develop packages for the software Mathematica. Development environments include the Eclipse Java development tools (JDT) for Java and Scala, Eclipse CDT for C/C++ and Eclipse PDT for PHP, among others.

The initial codebase originated from IBM Visual Age. The Eclipse software development kit (SDK), which includes the Java development tools, is meant for Java developers. Users can extend its abilities by installing plug-ins written for the Eclipse Platform, such as development

toolkits for other programming languages, and can write and contribute their own plug-in modules. Released under the terms of the Eclipse Public License, Eclipse SDK is free and open source software (although it is incompatible with the GNU General Public License). It was one of the first IDEs to run under GNU Classpath and it runs without problems under IcedTea.

3.2.5.2.2 Notepad++

Itis a text editor and source code editor for Windows. It aims to be a lightweight and robust editor for a variety of programming and scripting languages. One advantage of Notepad++ over the built-in Windows text editor Notepad, is that Notepad++ supports tabbed editing, which allows working with multiple open files. Notepad++ opens big files significantly faster and can be used as Windows Notepad replacement.

3.2.5.2.3 Firefox

Mozilla Firefox is a free and open source web browser developed for Windows, OS X and Linux, with a mobile version for Android, by Mozilla Foundation and its subsidiary, the Mozilla Corporation. Firefox uses the Gecko layout engine to render web pages, which implements current and anticipated web standards.

3.2.5.2.4 VLC media player

VLC media player (informally just VLC) is a highly portable free and open-source cross-platform media player and streaming media server written by the VideoLAN project.VLC media player supports many audio and video compression methods and file formats, including DVD-Video, video CD and streaming protocols. It is able to stream over computer network and to transcode multimedia files.

3.2.5.2.5 JDK (1.7)

The Java Development Kit (JDK) is an implementation of either one of the Java SE, Java EE or Java ME platforms released by Oracle Corporation in the form of a binary product aimed at Java developers on Solaris, Linux, Mac OS X or Windows. Since the introduction of Java platform, it has been by far the most widely used Software Development Kit (SDK).On 17 November 2006, Sun announced that it would be released under the GNU General Public License (GPL), thus making it free software.

3.2.5.2.6 JCIFS Library

In computer networking, Server Message Block (SMB), also known as Common Internet File System (CIFS) operates as an application-layer network protocol mainly used for providing shared access to files, printers, serial ports, and miscellaneous communications between nodes on a network. It also provides an authenticated inter-process communication mechanism. Most usage of SMB involves computers running Microsoft Windows, where it was known as "Microsoft Windows Network" before the subsequent introduction of Active Directory. Corresponding Windows services are the "Server Service" (for the server component) and "Workstation Service" (for the client component).

The Server Message Block protocol can run atop the Session (and lower) network layers in several ways : directly over TCP, port 445; via the NetBIOS API, which in turn can run on several transports : on UDP ports 137, 138 & TCP ports 137, on several legacy protocols such as NBF (incorrectly referred to as NetBEUI).

3.2.6 SUPPORTED OPERATING SYSTEMS

--Windows XP ,Vista , 7

-- Linux

3.2.7 FEASIBILITY STUDY

Feasibility study is the process of determination of whether or not a process is worth doing. Feasibility studies are under taken within tight time constraints and normally culminate in a written and oral feasibility report. It took one week to determine feasibility for this project. Feasibility study helped as a sound basis for deciding how to precede the project. It helped in taking decisions such as which software to use, hardware combinations, etc. The feasibility analysis starts with the user set of requirements. With this, the existing system is also observed. The next step is to check for the deficiencies in the existing system. By evaluating the above points a fresh idea is conceived to define and quantify the required goals. The user consent is very important for the new plan. Along with, for implementing the new system, the ability of the organization is also checked. Besides that, a set of alternatives and their feasibility study is an important part in the software development.

3.2.7.1 Economical Feasibility

Economic feasibility determines whether there are sufficient benefits in creating to make the cost acceptable, or is the cost of the system too high. This signifies cost-benefit analysis and savings. On behalf of the cost-benefit analysis, the proposed system is feasible and is economical. During economical feasibility test, a balance between Operational economical feasibilities is maintained, as the two are conflicting. For example the solution that provides the best operational impact for the end user may also be the most expensive and, therefore, the least economically feasible. For calculating the development cost, the certain cost categories were evaluated

3.2.7.2 Operational Feasibility

Operational feasibility criteria measure the urgency of the problem (survey and study phase) or the acceptability of a solution. There are various aspects of operational feasibility to be

considered. These decide the effectiveness of the system. These measures can be collectively called PIECES.

P (Performance): The code developed gives the desired output for a particular input.

I (Information): This software provide end-users with timely, accurate and user formatted information i.e. accurate encrypted cipher text.

E (Economy): This software offer adequate service level and capacity to reduce the costs.

C (Control): This software offers guaranteed accuracy.

S (Services): This software provides desirable reliable service to those who need it. The code is flexible and expandable.

3.3 DESIGN

Software design is an iterative process through which requirements are translated into a "blueprint" for constructing the software. Initially, the blueprint depicts a holistic view of software. That is, the design is represented at a high level of abstraction— a level that can be directly traced to the specific system objective and more detailed data, functional, and behavioural requirements. As design iterations occur, subsequent refinement leads to design representations at much lower levels of abstraction.

3.3.1 FUNCTIONALITY MODULES

The system will be developed in 4 main modules-

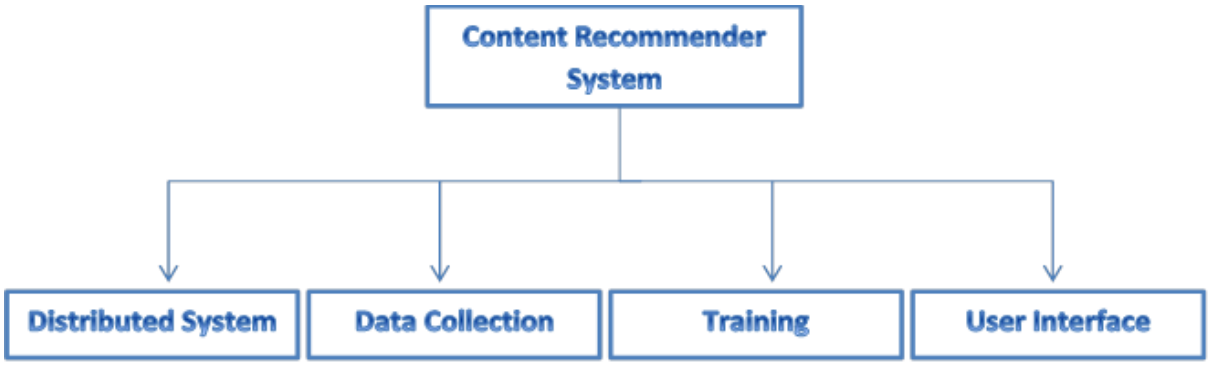

Fig 3.3 Main Functionality Modules

Along with these main modules, system will also contain three smaller modules (Searching & Sorting, Web Server and File Info Database).

3.3.1.1 - DISTRIBUTED SYSTEM

A distributed system is a collection of independent computers that appears to its users as a single coherent system.A distributed system will be designed which will distribute the operations like learning, recommending amongst different computers in order to get best performance.

Parameter properties will be saved in permanent storage in each of the user's computer. Learning will be performed at some regular definite intervals or when a particular threshold of ratings is submitted. The existing files having parameter properties at each computer will be updated after the learning is performed. After calculation of recommendations, the information will be shared between computers using this system. System will be loosely coupled.

Challenges in Distributed System

- **Security:**
  Many of the information resources that are made available and maintained in distributed systems have a high intrinsic value to their users. Their security is therefore of considerable importance. Security for information resources has three components: confidentiality, integrity and availability.
- **Scalability :**
  A system is described as *scalable* if it will remain effective when there is a significant increase in the number of resources and the number of users.
- **Failure Handling:**
  Failures in a distributed system are partial – that is, some components fail while others continue to function. Therefore the handling of failures is particularly difficult.
- **Concurrent execution of processes.**

3.3.1.2 - USER INTERFACE

A user interface is the system by which people (users) interact with a computer . The user interface includes hardware (physical) and software (logical) components. User interfaces provide a means of allowing the users to manipulate a system and allowing the system to indicate the effects of the users' manipulation.We have chosen to build a web based interface as web based interface offers much more flexibility as compared to an application based interface. For example web based interface offers adding plugins for video players, music players etc and through web based interface one person can access his account from other's computer. It will show content like e-Books, videos, music etc in the browser itself. User Interface provides the functionality of downloading the files and Searching.

3.3.1.3 - DATA COLLECTION

The main purpose of this module is to obtain a list of items that a user has listened to or watched on his/her computer, to observe the items that a user viewed and also to analyze the item/user viewing times. It will also ask users to rate items on a scale of 5 after they have viewed the item. For each user a rating vector is stored in the computer which contains the ratings submitted by the user. When a new rating is collected it is distributed to all systems.

3.3.1.4 –BEHAVIOR LEARNING

The algorithm for recommending content learns theparameter vector (to define the content) for content. On the basis of current rating the algorithm learns user behavior to predict the rating for unrated content so that the highly rated content is provided to the user. It learns to minimize the error in calculation of rating .The algorithm works separately for different type of content. Average rating will be given to the content for the new users.

Challenges in Behavior Learning

- Determining when to run learning algorithm
- Determine time when maximum computers are available

- Ensuring system does not perform too much processing.
- Determining practical accuracy of system

3.3.1.5 SEARCHING

The interface contains a search option to search for a file or a folder in the current directory or in all the connected computers. Search results displayed in a sorted manner on the basis of their average rating given by users and also according to their popularity. Category based searching is also available.

CHAPTER 4

RESULTS AND DISCUSSION

This chapter discusses the experiences of the project, in a wholly subjective manner. It is divided into six parts. It explains about what findings we gathered when we put Content Recommendation system for LAN to testing purposes. Due to time and resource limitations the project, though fully functional in current state, suffers from minor limitations and some remaining tweaks which can be applied to further enhance the utility of the project.

The user can use Recommender System for following activities –

- User can seek the help of system to see the recommended movies, music when he is confused about what to watch/listen.
- User can search for files/folders in all the computers in the network and find sorted results easily and in faster manner.
- It can be used for educational purposes – to see recommended books , lectures etc.
- It can be used for Entertainment purposes i.e., to watch movies , music videos etc.
- User can browse the shared files/folders of other computers.
- User can rate the files according to his choice so that other similar content is recommended to him.

4.1 IMPLEMENTATION

The implementation of the project is the task of converting idea conceived on paper during design phase into a real world application that can be used. The implementation phase of software development is concerned with translating design specification into source code. The preliminary goal of implementation is to write source code and internal documentation so that conformance of the code to its specifications can be easily verified, and so that debugging, testing, and modifications are eased. This goal can be achieved by making the source code as clear and straightforward as possible.

Simplicity, clarity, and elegance are the hallmarks of good programs; obscurity, cleverness, and complexity are indications of inadequate design and misdirected thinking.

4.1.1USER INTERFACE

User Interface shows all the connected PCs in a list format , when a PC is selected a list will show all the files and folders shared by that PC in addition to showing the recommended files to user in another pane. When a media is clicked, the media will be available to user through a media player add-on in the browser.

4.1.2 DISTRIBUTED SYSTEM

The main function of distributed system is the **ping mechanism,** which will be helpful to determine the connected PCs. Through the ping mechanism the source PC (which Broadcasts the Ping Request) will get the resource information of the other connected PCs.

Two types of Ping UDP packets have been used –

1- Ping Request Packet
2- Ping Reply Packet

Ping Request Packet have a fixed length identifier which is used to check if the ping reply received corresponds to the same ping request sent.

The Ping Reply Packet will contain following information-IP address, Latency, Authentication (byte array), Total memory, free memory, Processer and system information. This method is implemented using serialization (Serializable Interface) in java.

Output for ping in two systems is as follows –

Ping Started

Ping request Packet Received

getFreePhysicalMemorySize = 892522496

Ping response Packet Received

processingTime153

Ping response Packet Received

System:0-- Processing:2133 Intel(R) Core(TM) i3 CPU M 330 @ 2.13GHz ,address:/192.168.1.164,Lat:49,Memory total:3.13534464E9 ,Memory free:9.31467264E8 ,NoCores:4

System:1-- Processing:2200 Intel(R) Core(TM)2 Duo CPU T6600 @ 2.20GHz ,address:/192.168.1.163,Lat:14,Memory total:3.218112512E9 ,Memory free:8.92522496E8 ,NoCores:2

Ping request Packet Received

getFreePhysicalMemorySize = 991379456

processingTime623

Other main function of the distributed system is to synchronize the file information database which contains ratings, number of times opened/downloaded etc. for each file/folder which is/are rated. This is done using TCP. Each time the system is run on a new computer the file information database present in already running systems is synced with its own. Each time a user opens a file or rates a file/folder the distributed system will send a UDP broadcast to all other system informing the changes.

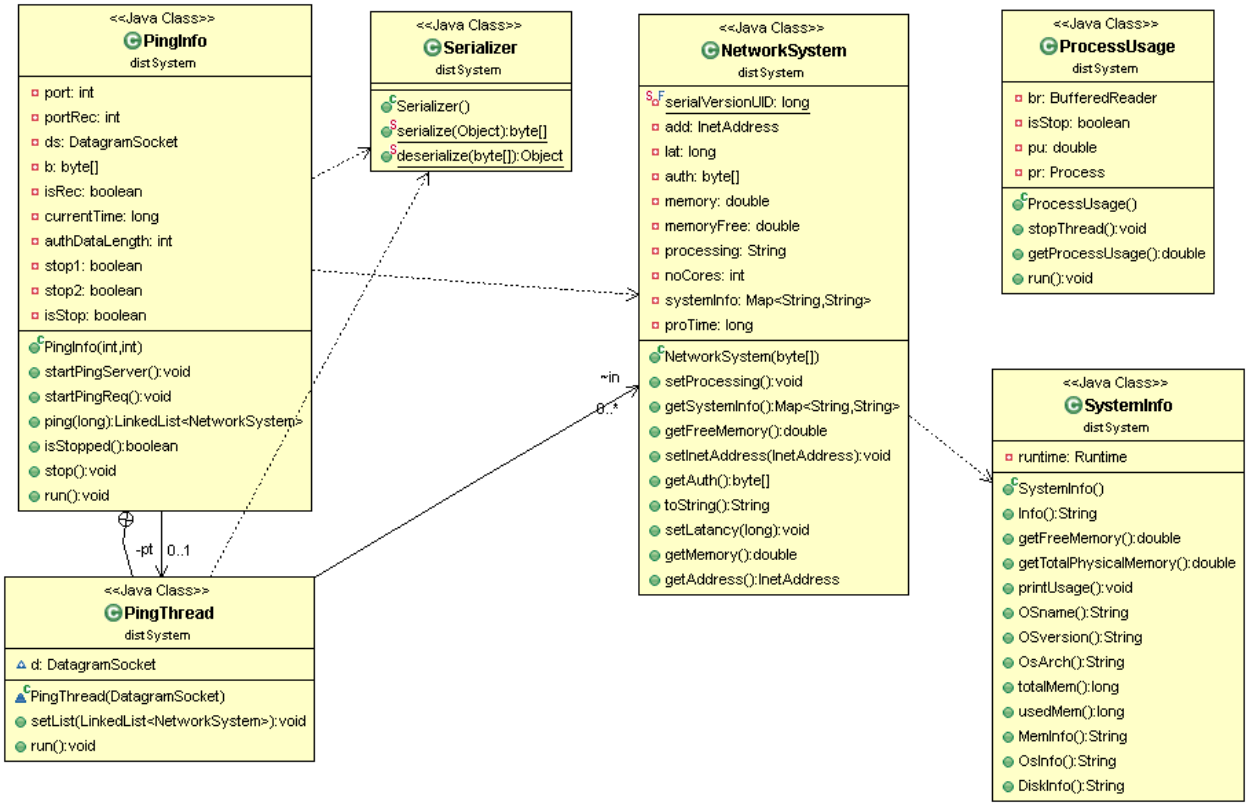

Fig 4.1 above shows the class diagram for the distributed system package, which consists of various classes and shows the relationship between these classes.

4.1.3 SEARCHING & SORTING

As system is accessing the shared files in LAN so a library for SMB (Server Message Block) protocol is required. For this jcifs (http://jcifs.samba.org) library is used. We have used **depth first search algorithm**for searching.

Depth-first search (DFS) is an algorithm for traversing or searching tree or graph data structures. One starts at the root (selecting some node as the root in the graph case) and explores as far as possible along each branch before backtracking.

The search results are sorted according to their popularity/average ratings. Two types of searching are available to users; user can either search for a file in the currently opened folder or can search for the file in all the computers, user can also perform a partial search which searches for the search term in the filename anywhere or a full search which completely matches the search term with the file name .

For sorting the list of files and folders, four parameters are used –

1- Number of times(n) file is downloaded / Number of times folder is opened .
2- Rating(r) given by the users in a 1-5 scale (zero if not rated).
3- Average rating(a) amongst all users.
4- Number of users (u)who rate the file/folder.

All these features are given equal importance , so to compare two files following formula is used :-

Value = 0.25*(n+r+a+u)

The file having higher value appears higher in the sorted results.

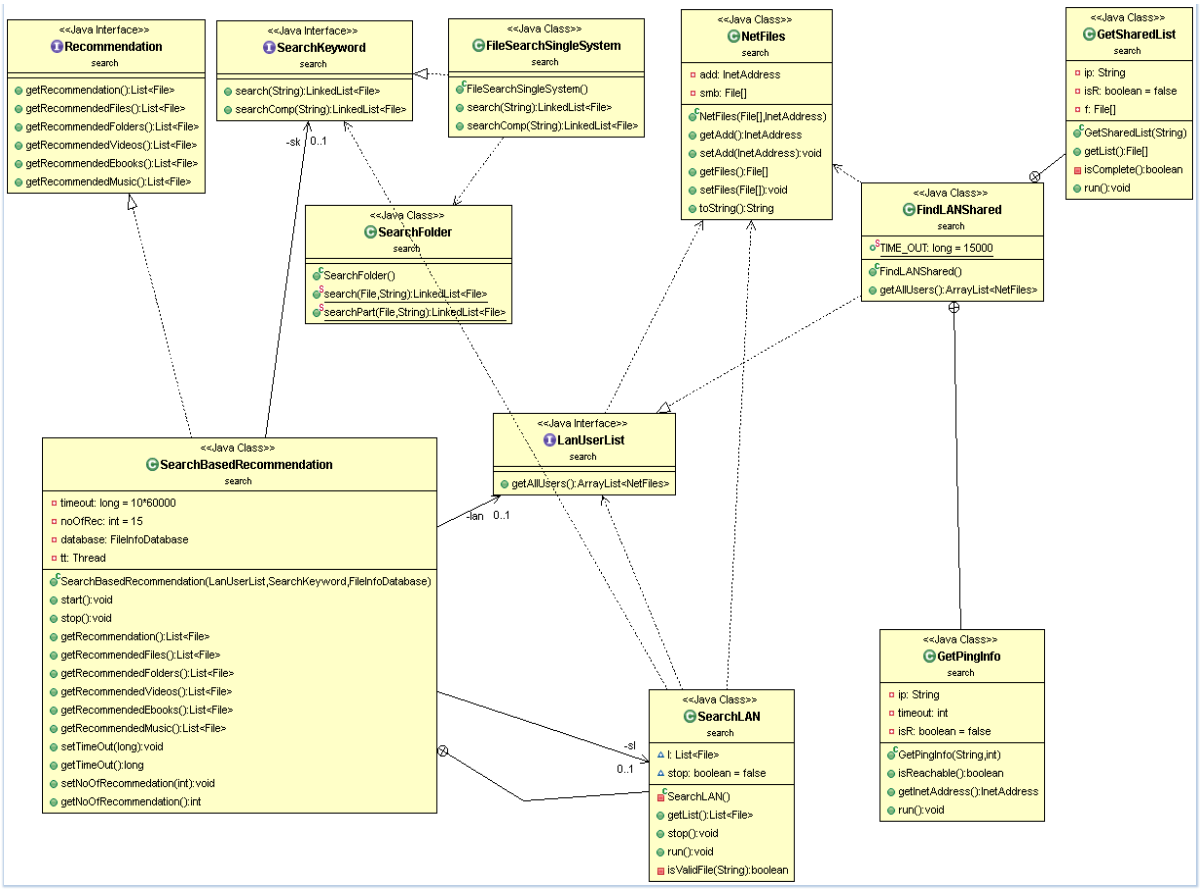

Fig 4.2 shows the class diagram of the Search package.

4.1.4 RECOMMENDATION

System provides general recommendation on the basis of the computers online in the network. It provides six types of recommendation:-

1- Recommendation of all files and folders.
2- Recommendation of files only.
3- Recommendation of folders only.
4- Recommendation of Videos
5- Recommendation of e-Books
6- Recommendation of Music.

A searching based recommendation system is applied for the recommendation in which a thread is kept running which continuously scans all the files within LAN and sorts them according to sorting algorithm specified in section- 4.1.3 and filtering them accordingly. Such a system a very flexible, scalable and simple in terms of application and use. The flexibility of system allows itself to adapt to new changes in a very short time. For example if a computer disconnects from the network then the recommendation from that system must be removed immediately from the recommendation results. The scalability of the system allows it to work as fast and as accurate in large system as it works in a smaller system and simplicity allows it to be implemented easily.

This system can later be extended to be a learning based recommendation system which can provide better recommendations for each user. But such system is too complex and needs very high processing power which will increase linearly with the increase in the number of computers which are sharing in the network, so it will take a lot of time to learn .Such system will also not be as flexible as the current system as it will not be able to adapt to the changes in the network immediately and will take a lot of time.

3.4.5 FILE INFORMATION DATABASE

To identify a file, a key is used which consists of its (name, size). It is assumed that (name, size) composite is different for each file. If these two are same for two files then they are considered same by the system. For folders only name attribute defines the key.

A **hashmap / hash table** is used to map the key for a file with the information contained in that file. Hash map is used with a load factor of 0.75 which gives a good balance between memory usage and performance of hashmap.

To differentiate between users, their computer's MAC addresses are used. So MAC address is used as a key for each user and A hashmap / hash table is maintained for mapping MAC address with the ratings submitted by that user.

After a particular time-out the settings (rating database) are saved.

Hashmap / Hash Table - a hash table (also hash map) is a data structure used to implement an associative array, a structure that can map keys to values. A hash table uses a hash function to compute an index into an array of buckets or slots, from which the correct value can be found.

Ideally, the hash function should assign each possible key to a unique bucket, but this ideal situation is rarely achievable in practice (unless the hash keys are fixed; i.e. new entries are never added to the table after it is created).In a well-dimensioned hash table, the average cost (number of instructions) for each lookup is independent of the number of elements stored in the table. In many situations, hash tables turn out to be more efficient than search trees or any other table lookup structure. For this reason, they are widely used in many kinds of computer.

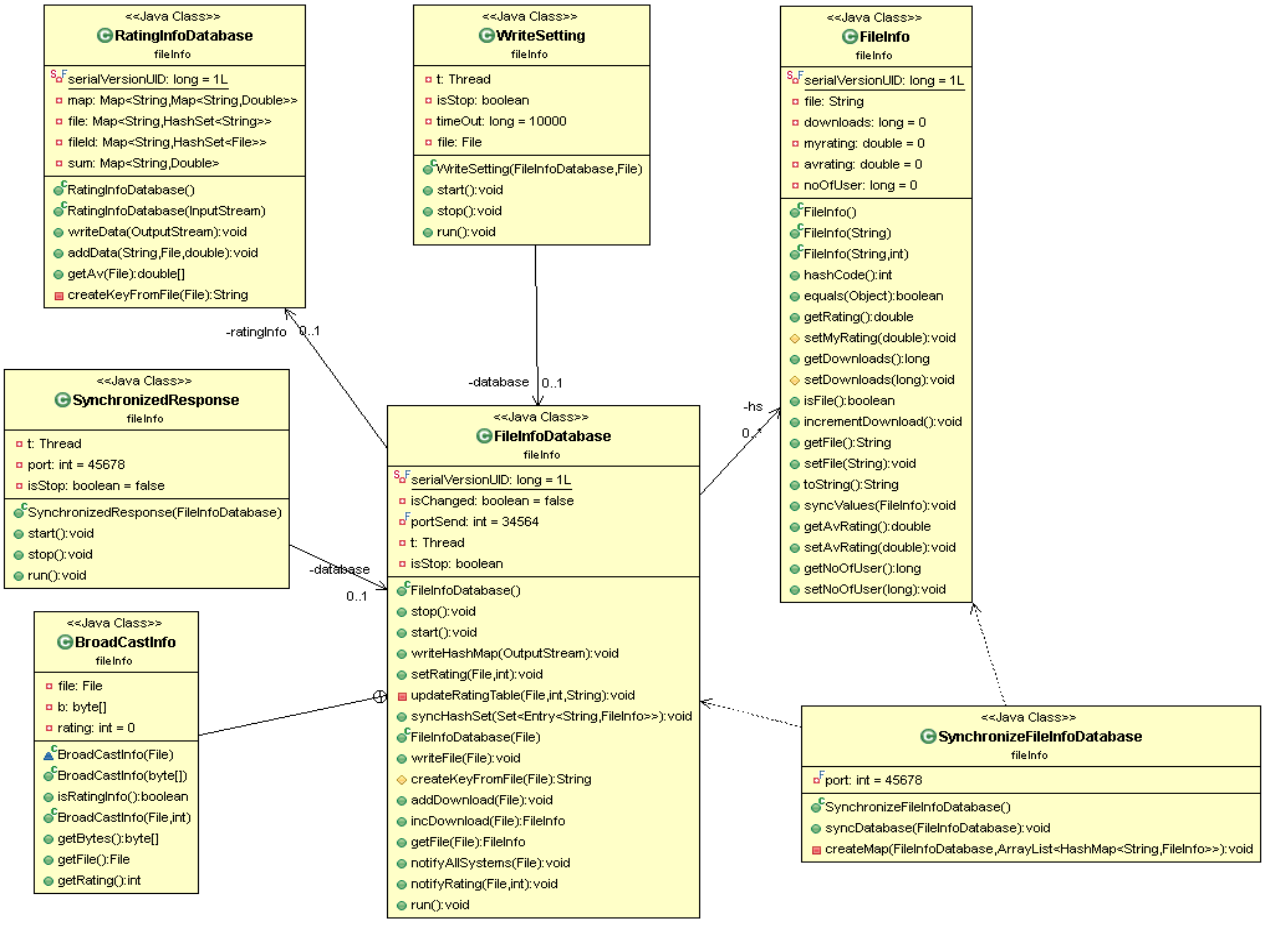

Fig 4.3 above shows the class diagram for File information Database package.

4.1.6 WEB SERVER

Web Server is built from scratch in the system to reduce the complexity, processing power and time. The web server will process all the requests and provide appropriate response. It is used for putting the dynamic html code in the html files for example – in getting the list of all the connected pcs and their shared folders and then showing it in html file.

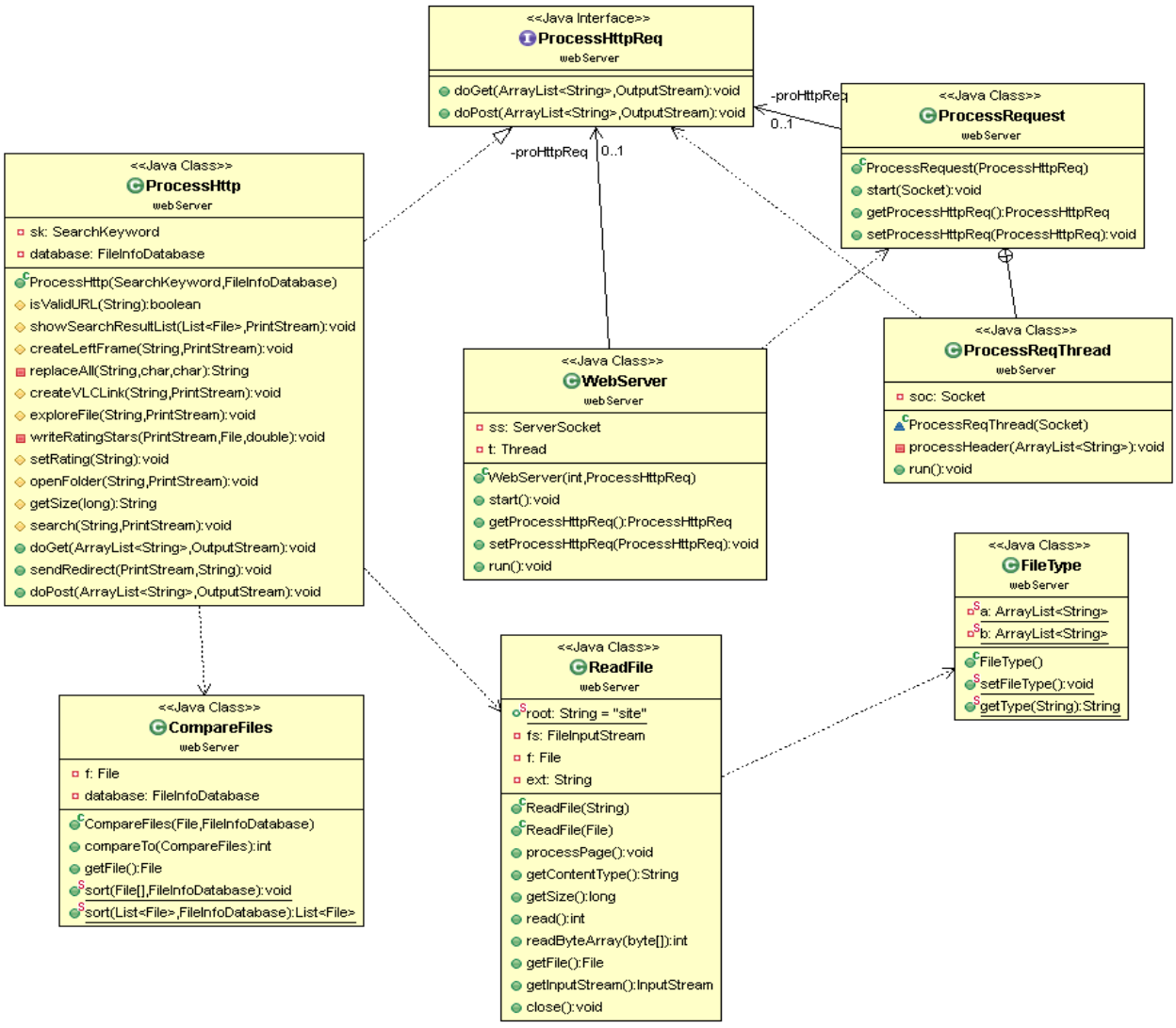

Fig 4.4 above shows the class diagram of the web server package.

4.1.7 LOGGING

All the activities like requests received, number of computers connected at a moment, the destination of UDP packets etc. are logged in a file so that if some problem occurs then state of the system can be resumed or the system can take actions according to the log. The time is noted at each log entry.

4.1.8 JAVA DOCS

Javadoc is a documentation generator from Oracle Corporation for generating API documentation in HTML format from Java source code. The HTML format is used to add the convenience of being able to hyperlink related documents together.

4.2 TESTING

Once software has been built, it cannot be allowed to go into services without checking it for presence for any errors. It must be checked for errors, if any. In fact testing software is operating the software under controlled Conditions, to

(1) Verify that it behaves "as specified"
(2) To detect errors, and
(3) To validate that what has been specified is what the user actually wanted.

Software testing is formally defined as "Testing is the process of analyzing a software item to detect the differences between existing and required conditions (that is defects /errors /bugs) and to evaluate the features of the software item."
Software testing, depending on the testing method employed, can be implemented at any time in the development process. However, most of the test effort occurs after the requirements

have been defined and the coding process has been completed. As such, the methodology of the test is governed by the software development methodology adopted.

The "Content Recommendation System for LAN" was similarly checked after completion for presence of errors. The possible errors may include incorrect links, use of variables in wrong place, incorrect logic, mistaken name of form elements and use of deprecated functions. There were basically two types of testing methodologies used:

--Black box Testing

--White box Testing

4.2.1 BLACK BOX TESTING

Black box testing takes an external perspective of the test object to derive test cases. These tests can be functional or non-functional, though usually functional. The test designer selects valid and invalid input and determines he correct output. There is no knowledge of the test object's internal structure. This method of test design is applicable to all levels of software testing: unit, integration, functional testing, system and acceptance. The basis of the Black box testing strategy lies in the selection of appropriate data as per functionality and testing it against the functional specifications in order to check for normal and abnormal behaviour of the system. In order to implement Black Box Testing Strategy, the tester is needed to be thorough with the requirement specifications of the system and as a user, should know, how the system should behave in response to the particular action.

Viewing connected Computers-

The viewing of connected computers on the web page was checked i.e., whether the system was able to display all the computers that are connected in the network.

Browsing shared files in Computers–

Browsing all shared files for different computers was checked.

Searching–

Searching was performed on all connected computers and searching option was checked.

4.2.2 WHITE BOX TESTING

It is a software testing technique whereby explicit knowledge of the internal workings of the item being tested is used to select the test data. Also called clear box testing, glass box testing, transparent box testing, or structural testing, it can be applied at the unit, integration and system levels of the software testing process. However it is usually done at the unit level. It can test paths within a unit, paths between units during integration, and between subsystems during a system level test.

White box testing involves looking at the structure of the code. When the internal structure of a product is known, tests can be conducted to ensure that the internal operations are performed according to the specification. And all internal components have been adequately and properly exercised.

Sychronization of file information database–

System was checked for synchronization of file information database between different systems in the network.

Testing for correct working of sorting algorithm –

Testing was performed for correct working of the sorting algorithm.

4.2 RESULT

User Interface

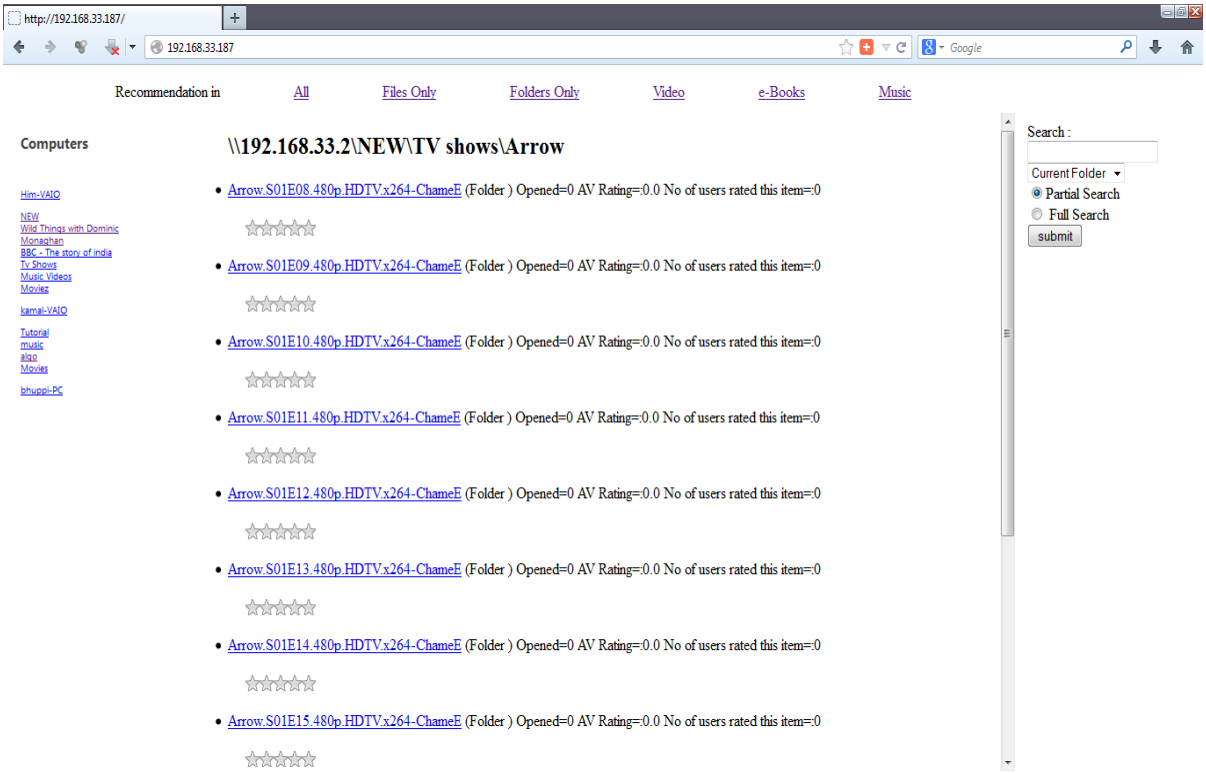

Fig 4.5 - Shows the path browsing in the user interface.

As visible in Fig 4.1 user interface consists of 4 main regions , top region shows options of recommendations , left region shows all the connected computers , right region shows the options for searching and middle region shows the files in the folder currently opened or search results (in case of search). For each file/folder shown various attributes of that file/folder are also visible like the number of times the file/folder is downloaded/opened , their average rating and the number of users who have rated the file/folder.

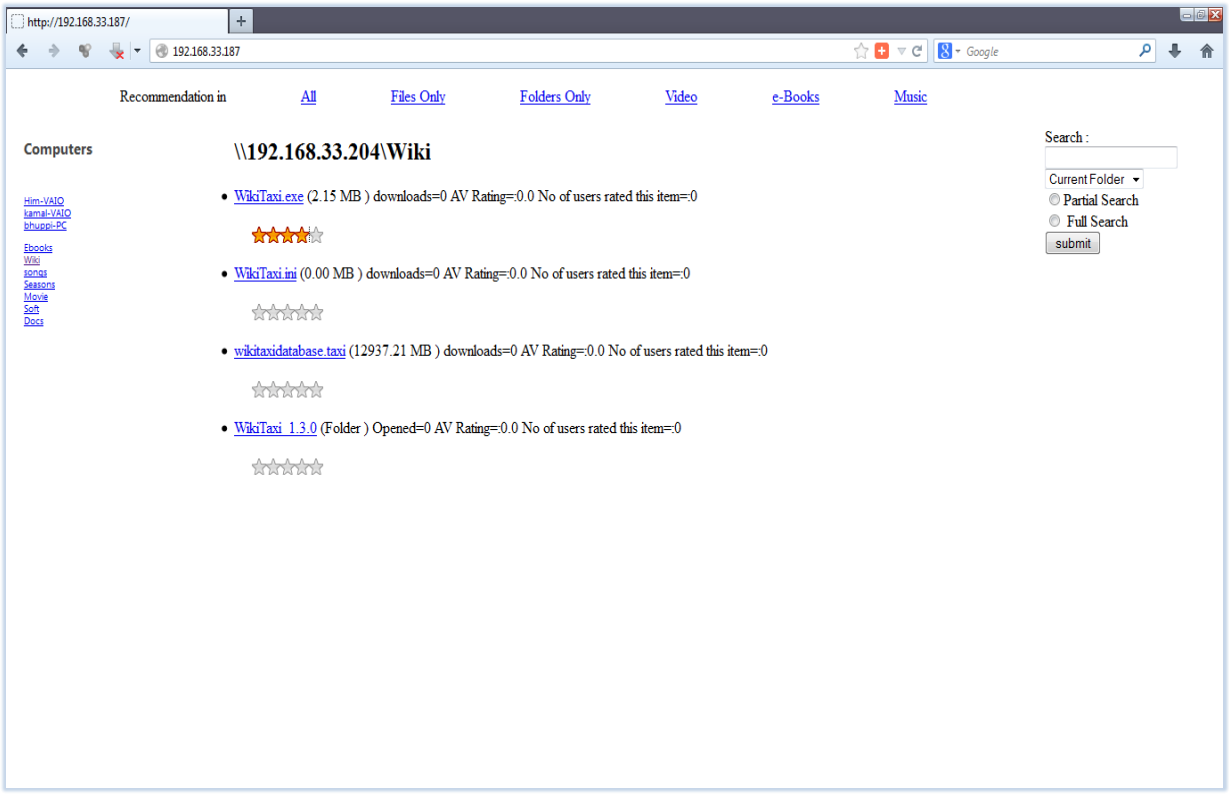

Fig 4.6 above shows that a user can rate any file/folder by clicking the stars which are given with each file name. In this case the file WikiTaxi.exe is rated 4 by clicking 4 stars.

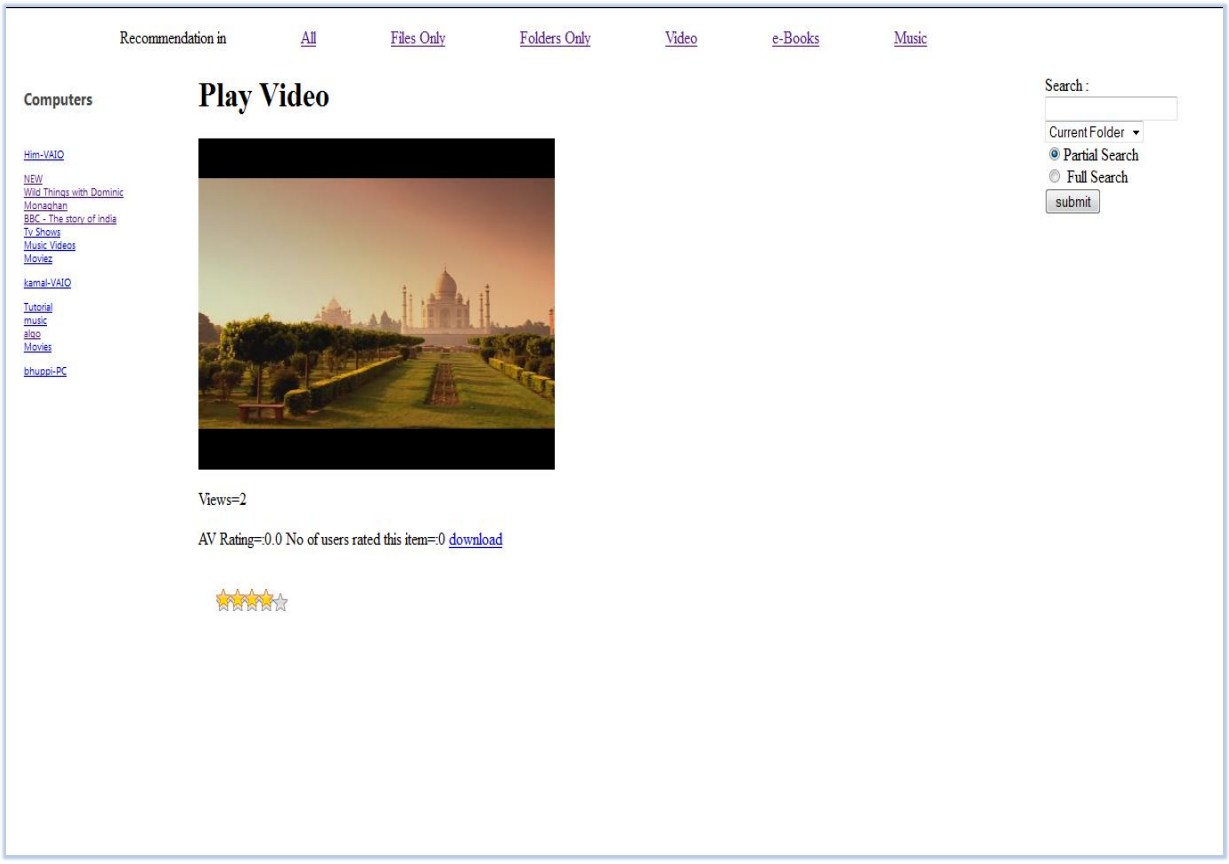

Fig 4.3 above shows that a user can play videos / audio files in the browser itself after clicking the filename, which will show the player in the middle region in place of the directory listing. User can also rate the files while viewing them and can also download the file by clicking the download link. VLC plugin in the web browser makes it possible for user to view the media file. The window will also show the average rating of the file, the number of users who have rated the file and the number of views. User can also view the file in full screen by clicking on it.

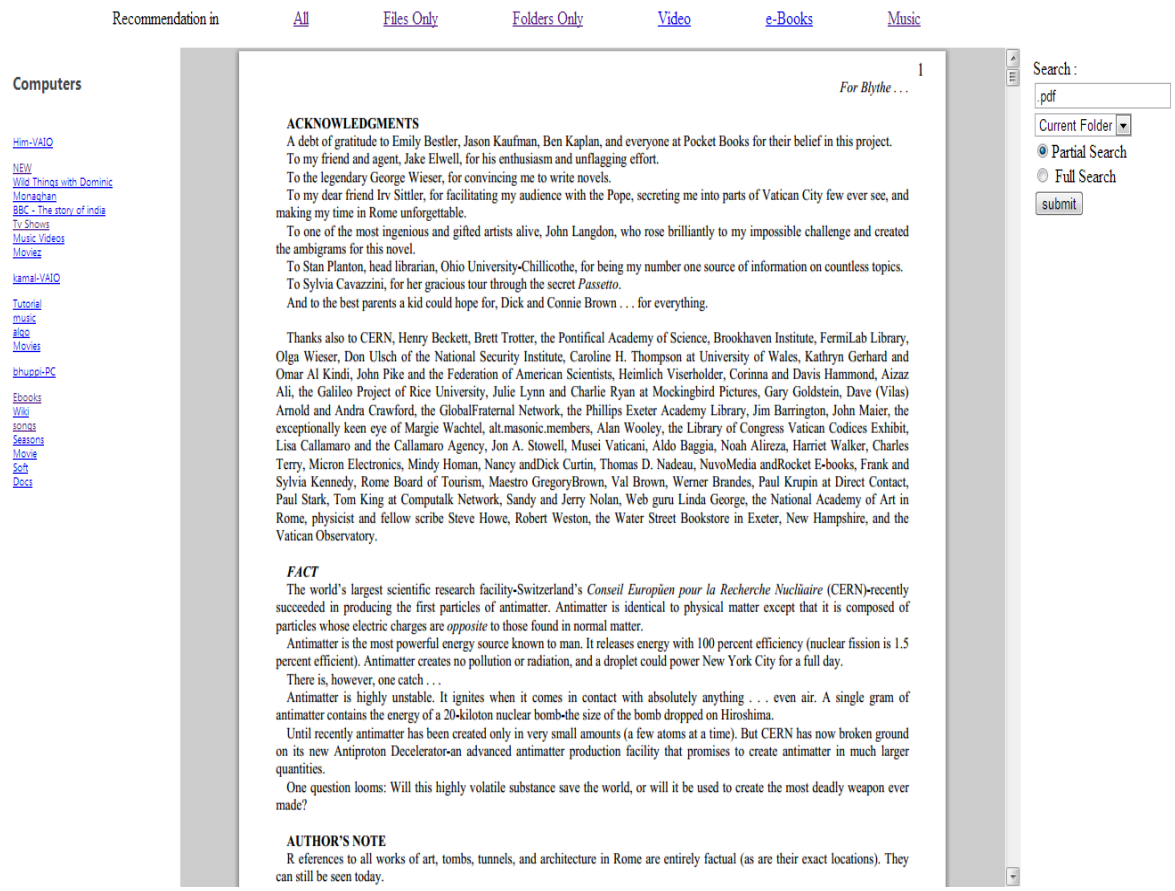

Fig 4.7 above shows a pdf file can also be displayed in the middle frame when needed to be opened.

Ping Mechanism

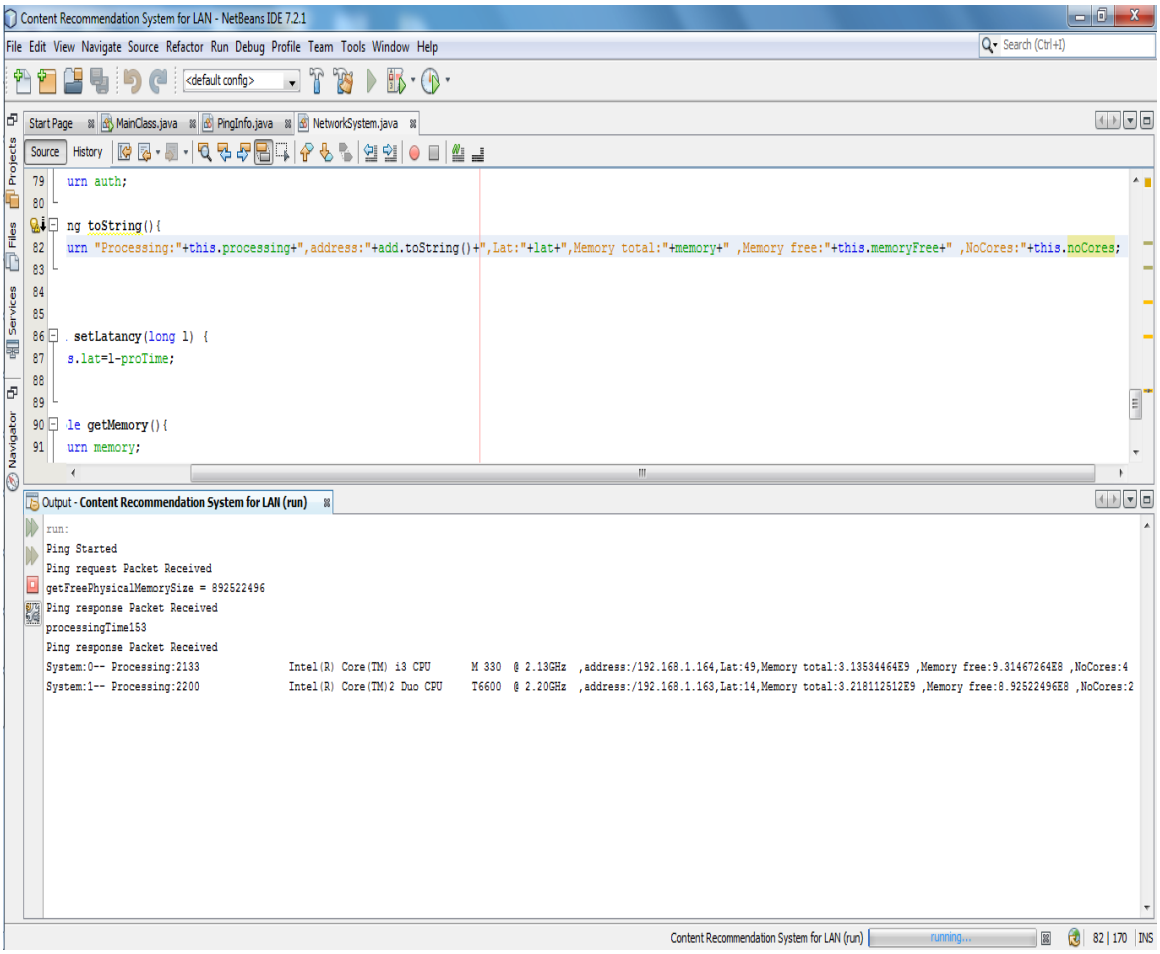

Fig 4.9 above shows the output of ping between two computers which will be helpful to determine the connected PCs.

Searching

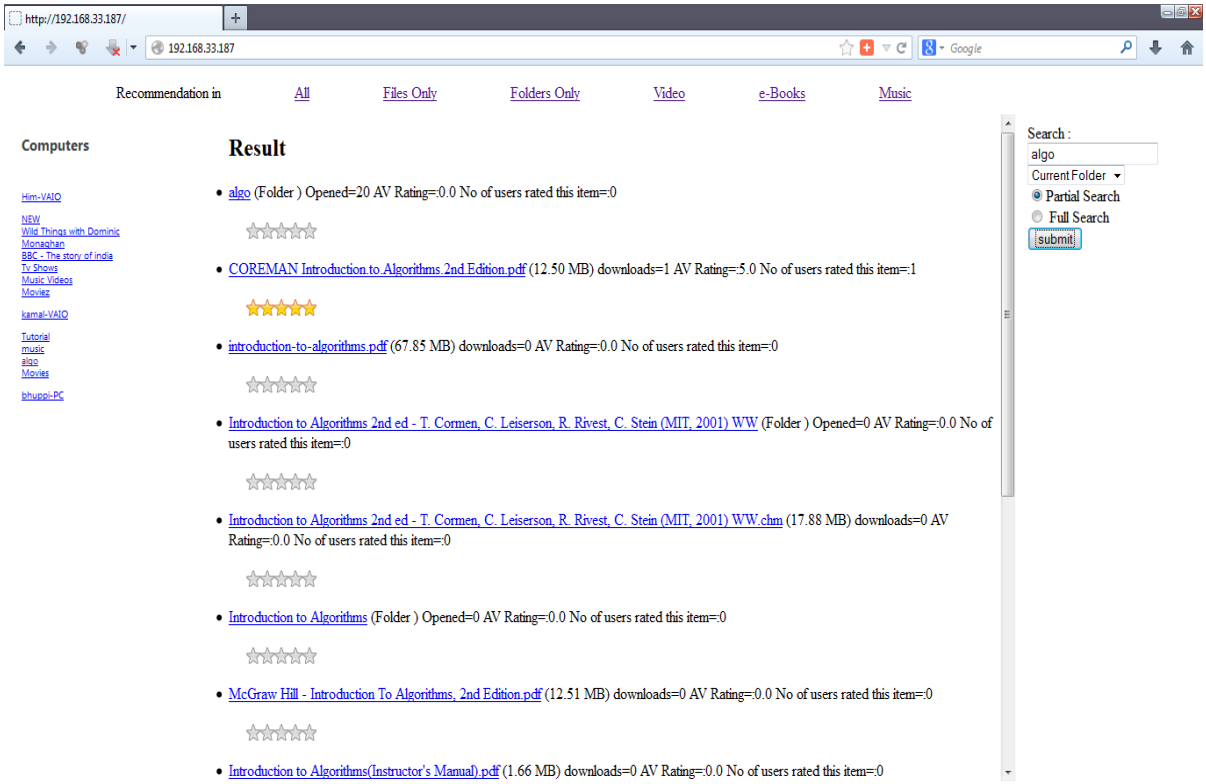

Fig4.10 above shows the results returned by performing a partial search for the term “algo”.

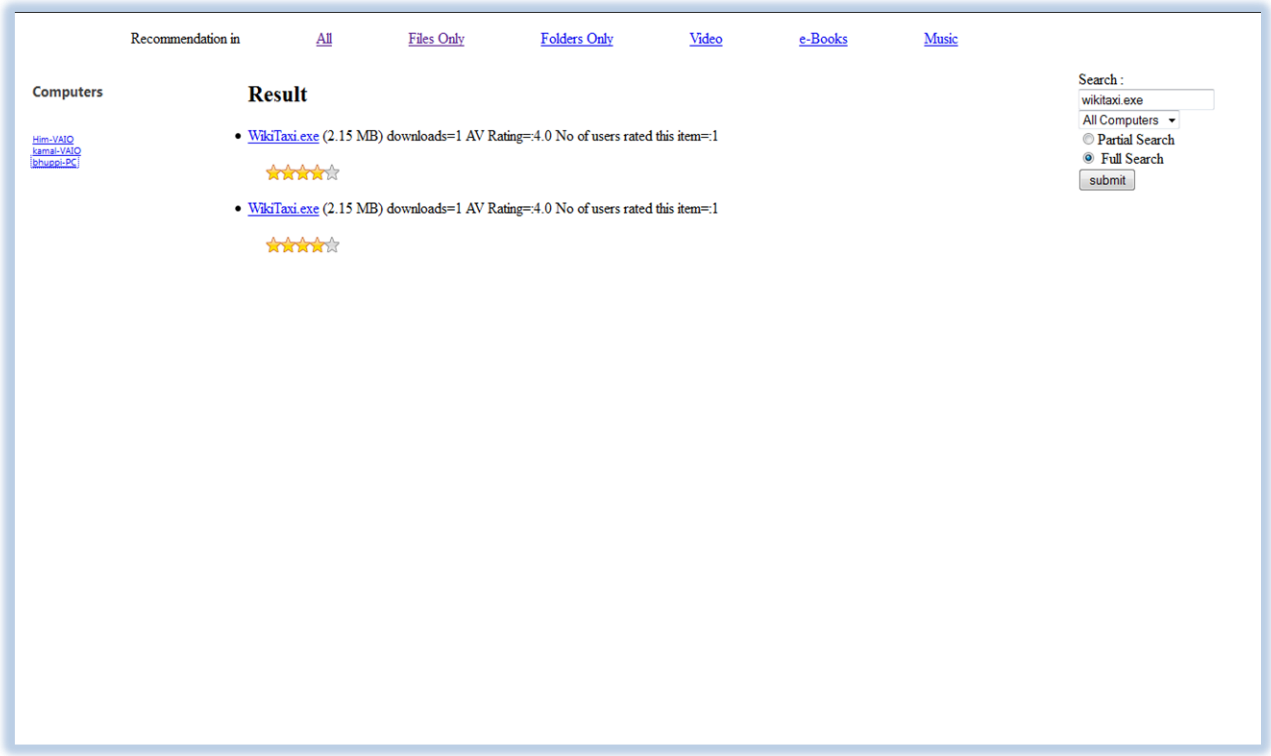

Fig 4.11 above shows the search results returned by performing a full search for the term "wikitaxi.exe" in all computers. User can perform a full search which completely matches the search term with the file name.

Recommendation

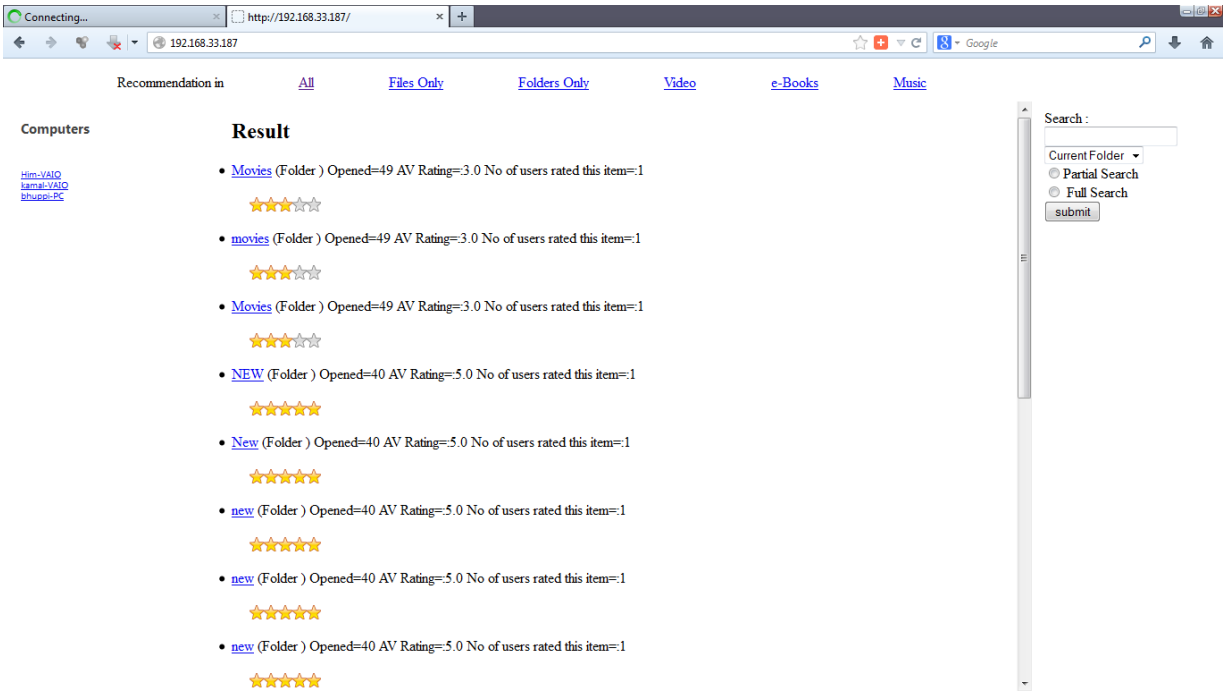

Fig 4.12 above shows the recommendation of all files and folders.

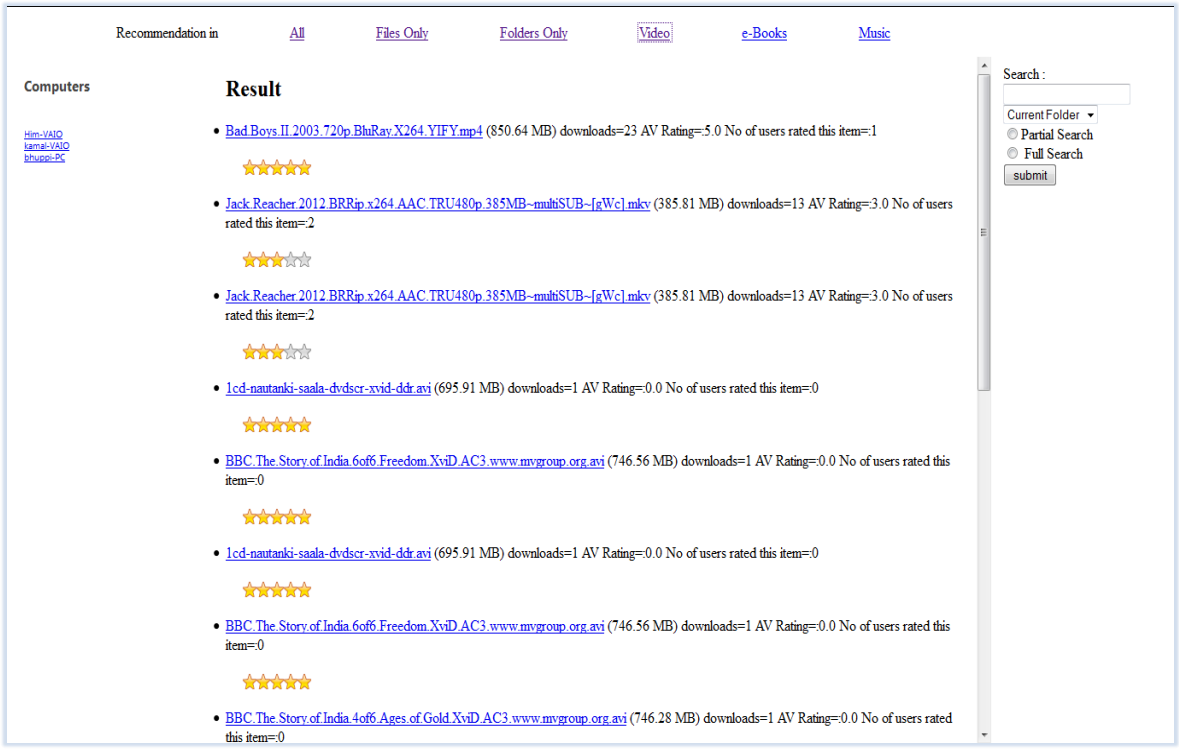

Fig4.13 above shows the recommendation of videos in the three computers connected.

File Type

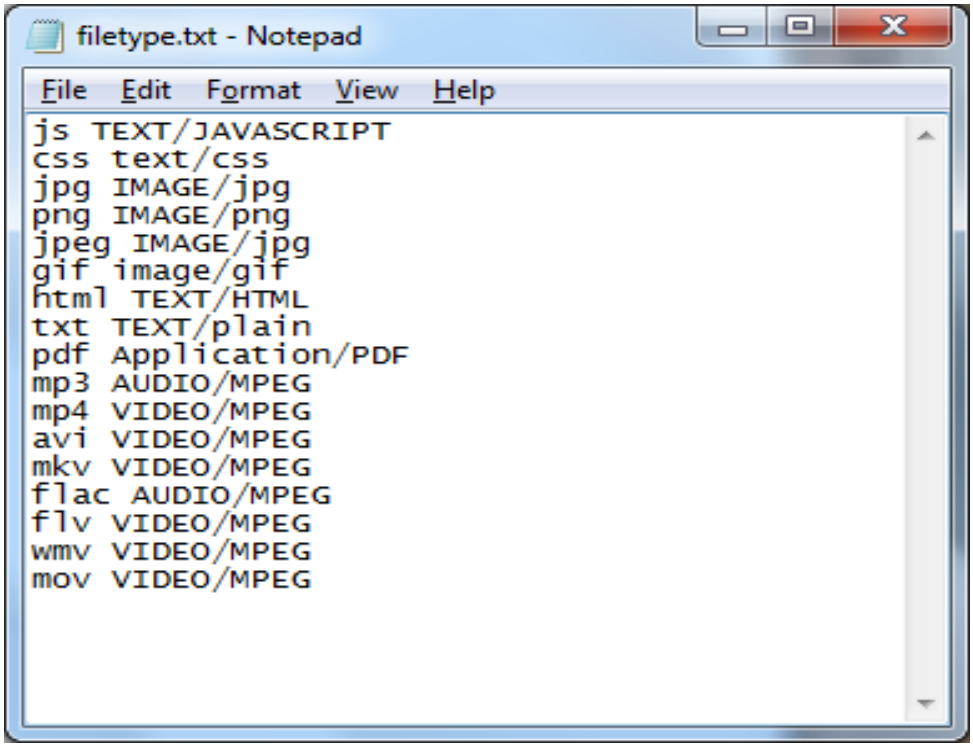

Fig 4.14 above shows a file (filetype.txt) which maps the file type with their internet media type so that it can be identified by the browser and accessed by the browser.

More file types can be appended to the file to extend the functionality of the system.

Logging

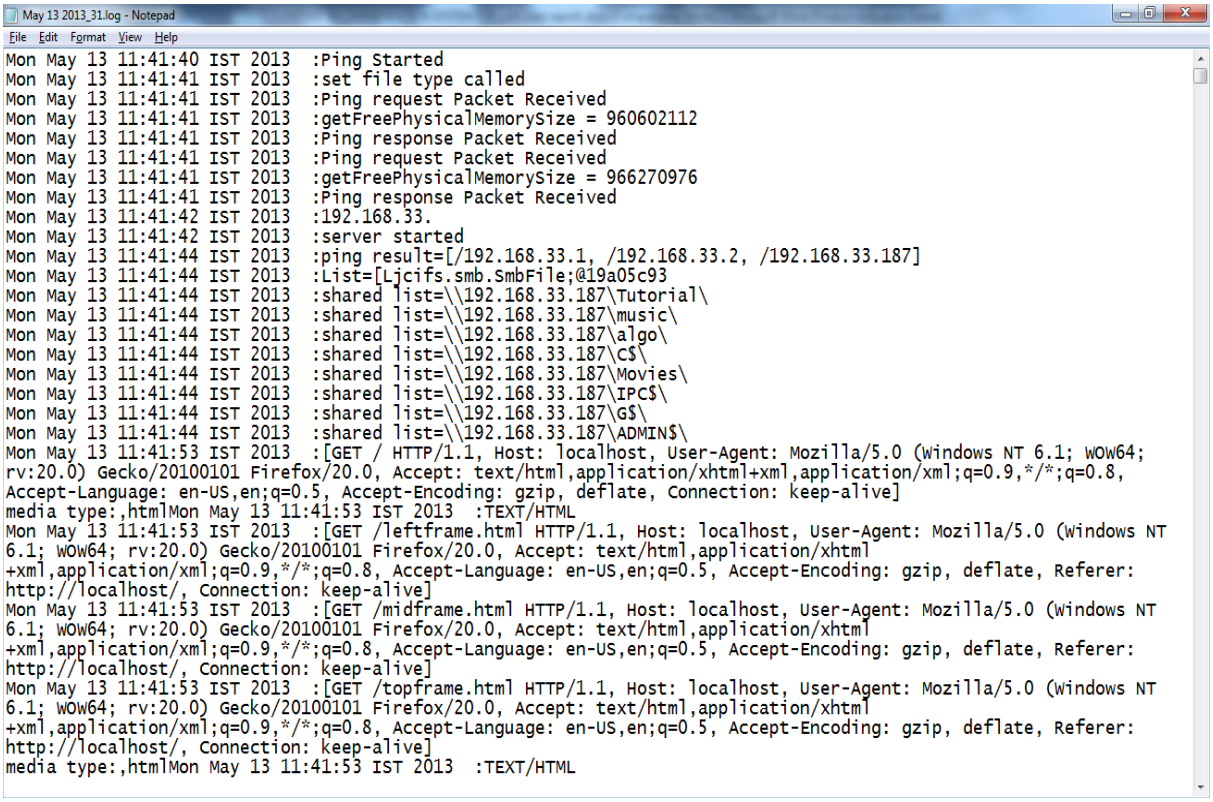

Fig 4.15 above shows the log file for the system. All the activities like requests received, number of computers connected at a moment, the destination of UDP packets etc. are logged in a file.

Java Docs for our project has the outline as given in following figure.

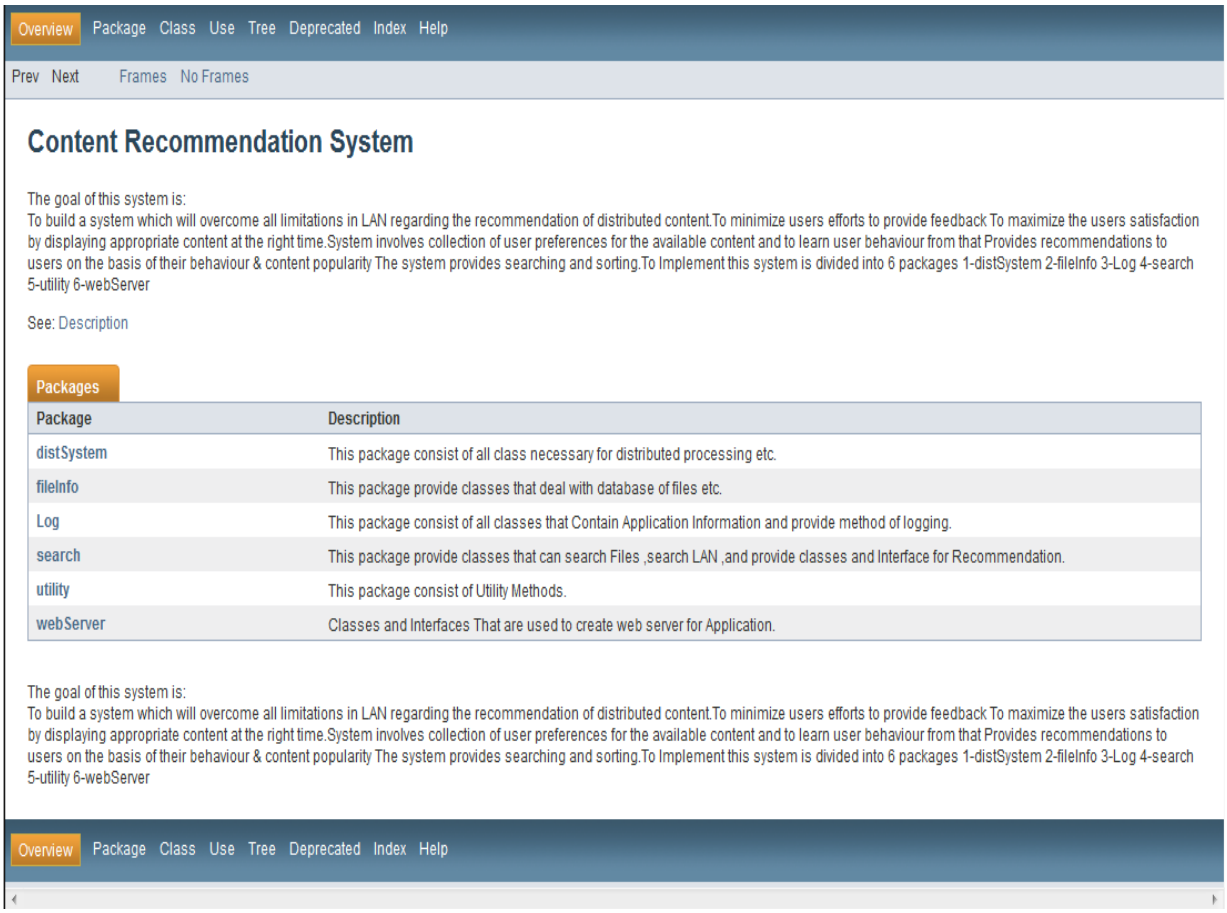

Fig 3.16 shows the description of the main packages used in the system.

For complete JavaDocs , refer to project documentation.

CHAPTER 5

SUMMARY AND CONCLUSION

The aim of our project is to provide an efficient system for recommending the content to users in LAN according to popularity and user behaviour, which will overcome all limitations in LAN regarding the recommendation of distributed content, minimize user's efforts to provide feedback and maximize the user's satisfaction by displaying appropriate content at the right time; which will involve collection of user preferences for the available content and learn user behaviour from it.

Content-based recommendation systems share in common a means for describing the items that may be recommended, a means for creating a profile of the user that describes the types of items the user likes and a means of comparing items to the user profile to determine what to recommend. The profile is often created and updated automatically in response to feedback on the desirability of items that have been presented to the user. Thus recommender systems have the effect of guiding users in a personalized way to interesting objects in a large space of possible options.

5.1 FUTURE IMPROVEMENT

This project can be enhanced in following ways

- The distributed system can be extended so that processing can be shared with other computers.
- User Interface can be improved to be more user friendly and attractive looking.
- Searching can be made more efficient by applying it in the distributed system.
- The Recommendation can be improved by including **behavior learning**.

Behavior learning can later be implemented by using following algorithm –

Problem Formulation –

n_u = Number of users

n_m = Number of Content(separate for different type of content)

r(i,j)=1 if user j has rated content I otherwise 0

$y^{(i,j)}$ = Rating given by user to content (1 to 10)

$\theta^{(i)}$ = feature vector for user i

$x^{(j)}$ = parameter vector for content j

α = learning rate

λ = regularization parameter

Cost function for learning algorithm is as follows –

$$J(x^{(1)},\ldots,x^{(n_m)},\theta^{(1)},\ldots,\theta^{(n_u)}) = \frac{1}{2}\sum_{(i,j):r(i,j)=1}((\theta^{(j)})^T x^{(i)} - y^{(i,j)})^2 + \left(\frac{\lambda}{2}\sum_{j=1}^{n_u}\sum_{k=1}^{n}(\theta_k^{(j)})^2\right) + \left(\frac{\lambda}{2}\sum_{i=1}^{n_m}\sum_{k=1}^{n}(x_k^{(i)})^2\right).$$

By applying gradient descent on this cost function -

1. Initialize $x^{(1)}, \ldots, x^{(n_m)}, \theta^{(1)}, \ldots, \theta^{(n_u)}$ to small random values.
2. Minimize $J(x^{(1)}, \ldots, x^{(n_m)}, \theta^{(1)}, \ldots, \theta^{(n_u)})$ using gradient descent:

$$j = 1, \ldots, n_u, i = 1, \ldots, n_m$$

$$x_k^{(i)} := x_k^{(i)} - \alpha \left(\sum_{j:r(i,j)=1} ((\theta^{(j)})^T x^{(i)} - y^{(i,j)}) \theta_k^{(j)} + \lambda x_k^{(i)} \right)$$

$$\theta_k^{(j)} := \theta_k^{(j)} - \alpha \left(\sum_{i:r(i,j)=1} ((\theta^{(j)})^T x^{(i)} - y^{(i,j)}) x_k^{(i)} + \lambda \theta_k^{(j)} \right)$$

3. For a user with parameters θ and content with (learned) features x, predict a star rating of $\theta^T x$.

Finding related content

For each product i, we learn a feature vector $x^{(i)} \in \mathbb{R}^n$.

How to find content j related to content i?

5 most similar content to content i:
Find the 5 content j with the smallest $\|x^{(i)} - x^{(j)}\|$

LITERATURE CITED

[1] Andrews, Gregory R. (2000), Foundations of Multithreaded, Parallel, and Distributed Programming, Addison–Wesley, ISBN 0-201-35752-6.

[2] Arora, Sanjeev, Barak, Boaz (2009), Computational Complexity – A Modern Approach, Cambridge, ISBN 978-0-521-42426-4.

[3] Cormen, Thomas H.; Leiserson, Charles E.; Rivest, Ronald L. (1990), Introduction to Algorithms (1st ed.), MIT Press, ISBN 0-262-03141-8.

[4] Russell, Stuart J.; Norvig, Peter (2003), Artificial Intelligence : A Modern Approach (2nd ed.), Upper SaddleRiver, New Jersey: Prentice Hall, ISBN 0-13-790395-2

[5] http://stderr.org/doc/libjcifs-java-doc/api/overview-summary.html Documentation of jcifs API (http://*jcifs.samba.org*) -

ABBREVIATIONS

TCP	Transmission Control Protocol
HTML	Hypertext Markup Language
HTTP	Hypertext Transfer Protocol
IDE	Integrated Development Environment
API	Application Programming Interface
LAN	Local Area Network
UDP	User Datagram Protocol
CRS	Content Recommendation System
SMB	Server Message Block
CIFS	Common Internet File System
DS	Distributed System
UI	User Interface