intelliView – An Intelligent Recommender System for Web TV Programs

By

Towhidul Islam
(2004-NUST-BIT-243)

A project report submitted in partial fulfillment of
The requirement for the degree of
Bachelors in Information Technology

In

NUST School of Electrical Engineering and Computer Sciences
National University of Sciences and Technology
Sector H-12, Islamabad. Pakistan
(2009)

CERTIFICATE

It is certified that the contents and form of thesis entitled "intelliView – An Intelligent Recommender System for Web TV Programs**",** Submitted by Mr.Towhidul Islam have been found satisfactory for the requirement of the degree.

Project Advisor: _____
(Mr. Owais Ahmed Malik)

Co-Advisor: _____
(Dr. Zia Ul Qayyum)

Dedication

In the name of ALLAH, the Most Gracious, the Most Merciful

I dedicate my project and all the efforts to my kind parents and family members, who encouraged me to succeed in life.

Acknowledgements

I am deeply grateful to my advisor Mr. Owais Ahmed and especially to co-adviser Dr Zia Ul Qayyum for their continuous assistance, patience and inspiration. Their dedication and guidance really helped me a lot.

I am extremely thankful to all my teachers who gave me guidelines throughout my degree. Their guidance, knowledge and support enabled me to carry out this project.

With humble regards, I offer my deepest thanks to my affectionate parents and all other family members for their encouragement and prayers to Almighty God for my success.

It is almost impossible to make note of all those, whose inspirations have been crucial in the completion of this thesis. I am thankful to all of them.

Table of Contents

Introduction. …….....8

1.1. Introduction ……8

1.2. Background ……8

1.3. Problem Statement ……9

1.4. Project Aim ……..10

Related Work & Literature Survey ……...11

2.1 Current Recommendation System …….....11

2.2 Works on TV Recommendation System ……..... ….11

2.3 Techniques involve in Recommendation System.……..... 12

2.4 Viewing behavior tracking ……..... ….13

2.5 User profiling ……..... ….13

2.6 Comparison of Text-Categorization technique with feature-based Profiling…14

Methodology ……..……..…….15

3.1 System Architecture and Components …….....16

3.2 User Interface …….....17

3.3 Scraping tools …….....18

3.4 Store Viewing history …….....18

3.5 User profile: Implicit …….....18

3.6 Recommender: Main module …….....19

3.7 Use Case Diagram …….....20

3.8 Class Diagrams …….....24

3.9 User Interface Navigation …….....25

3.10 Conceptual Component Model …….....27

3.11 Sequence Diagram …….....28

3.12 ERD Model …….....30

Algorithm Implementation & Results…….31

4.1 Tracking a user and update Viewing History.....31

4.2 Finding Today's Favorite Show 32

4.3 Finding Related Shows to provide recommendation 33

4.4 Implementation of User Profile ..…….....34

4.5 Generating Program profile.…….....34

4.6 Results …….....35

Conclusion……37

Future Works……...38

References ……39

List of Figures

Figure 1 15

Figure 2..... 16

Figure 3 20

Figure 4 24

Figure 5..... 25

Figure 6 26

Figure 7 27

Figure 8 28

Figure 9 29

Figure 10..... 30

Figure 11..... 31

Figure 12..... 35

Figure 13..... 35

Figure 14..... 36

Figure 15..... 36

Abstract

intelliView is a recommender system that suggests TV programs to viewer based on his/her explicit and implicit profiles. This recommender system takes video stream from different Web TV channels and maintains a database of TV programs by scraping program listing from these channels. It manages viewer's explicit profile by storing programs' wish list and the implicit profile is created while a viewer watches different types of programs on different timings at different channels. This viewing history in combination with explicit profile is used to recommend programs to the viewer.

The recommendation technique used for implementing this system is content-based text categorization strategy. In this technique thousands of feature like words or phrases in the program lists are used to generate programs profile. Later user profile, based on tracking viewing history and explicit wish list, is matched against this programs profile to suggest best match. The technologies used for implementation of the algorithms and user interface are J2EE 2.0, JSF and JDBC 2.0.

Chapter 1

INTRODUCTION

1.1.Introduction

Recommendation systems are suggest programs of interests by using statistical and machine learning techniques. They learn from examples of user's likes and dislikes. There are two main approaches used: *collaborative filtering* and *content-based*. Most of existing recommendation systems is based on collaborative filtering. There are two main disadvantages of collaborative filtering. Firstly, the system may not be able to find users with similar preferences to those of the target user. Secondly, there is not enough users' rating to make reliable recommendation. Content-based recommendation systems overcome the limitations of the collaborative filtering by making recommendation based on the content/synopsis of the program and target user's ratings. Two different content-based approaches have been proposed: *feature-based* and *text categorization-based.* Feature-based recommendation systems extract important features from the program synopsis and learn a user's profile (classifier) using a set of pre-classified (according to the user's rating) feature vectors, e.g. genre, leading actor/actress in a TV program recommender system. However, choosing representative features and appropriately encoding them, is not an easy task. Text categorization systems learn from thousands of features like words or phrases in the program synopsis.

1.2.Background

1.2.1 Personalization

It is a very common issue in many aspects of our lives. One aspect of personalization is, determining how much interested a user in a certain piece of information, and e.g. how interesting is this TV program for the viewer? How interesting is this book for the customer? How interesting is this news article for the reader? Etc. Profiling is a very powerful technique to provide personalized user information that really a user can like. There are two most popular way to make user profile, one is explicit and other implicit. Recent days implicit profiling gets more

focus because it gives suggestion without user interaction. So user can save their time and enjoy powerful recommendation.

1.3.Problem Statement

In recent days, numbers of TV channels have increased incredibly, so it is a big problem for viewers to identify television program of their interest. So, we came up with an application that can recommend content of interest to the viewer.

Traditional TV recommender systems recommend TV programs according to explicit interaction with the viewer and update user profile based on viewer's feedback. But the recommendation could be interesting or boring to viewer because user viewing behavior can change from time to time and the user has to interact with the system periodically. So we have developed a web based application that helps user to watch different channels (from Pakistan) and provide program recommendation using explicit/implicit profiling approach.

1.4.Project Aim

Our motivation behind this project is to develop a TV program recommendation system that apply different profiling techniques and provide most accurate results to the user experiences. This project has been developed on J2EE web application architecture so that in future we can apply collaborative filtering approach to provide more powerful program recommendation and better TV watching experience to the user.

Chapter 2

Related Work & Literature Survey

2.1. Current Recommendation System:

There are number of recommender system currently exist in market. Following is the list of recent developed and popular recommender system.

TiVo (2000) is a leading the way of the digital video recorder (DVR).TiVo DVRs provide an electronic television programming schedule, and provide features such as Season Pass recordings (which ensure subscribers never miss an episode of their favorite shows) and WishList searches (which allow the user to find and record shows that match their interests by title, actor, director, category or keyword). It means TiVo use explicit profiling methodology. [9]

Netflix (1997) is an online movie rental web application with an extensive personalized video-recommendation system based on preferences by its customers. [10]

MovieLens (2003) Is a movie recommendation system based on collaborative Filtering technique. It works by matching users with similar thought about movies. [11]

Kasenna – On demand IPTV with recommendation.

INTIMATE- A Web-Based Movie Recommender System.[1]

2.2. Importance of TV Recommendation Systems:

There are a lot of organizations which have spent a huge amount of money and man power to develop their web presence. This information provides edges to user when they have to make some decisions. Hence, there is a need of intermediaries to meet the increasing demand of time and cognitive load on a user. Recommendation systems are such tools that can help user to make their decision. Recommendation system suggest items of interests such as books, movies, CDs, news, pictures, etc by using statistical and machine learning techniques. They monitor users like and unlike.

2.3. Recommendation System Techniques:

2.3.1. Collaborative filtering:

Collaborative filtering technique is one of the most successful techniques to find out the prediction of user preferences in the recommendation systems. We can observe three major processes in the collaborative filtering approach: object data collections and representations, similarity decisions, and recommendation computations. Collaborative filtering helps us at finding the associations among the new individual and the existing data to further determine the similarity and provide recommendations. How can we define the similarity is an important issue. How similar should two objects be in order to finalize the preference prediction? Relationship decisions are concluded differently by collaborative filtering techniques. For example, people who likes and dislikes movies in the same categories would be considered as the ones with similar behavior. The concept of the nearest-neighbor algorithm has been included in the implementation of the collaborative filtering based recommendation systems.[2]

The maximum existing recommendation systems are based on collaborative filtering. They keep a database of user's preferences, find a group of users with similar preferences to the target user and suggest items the group members liked, e.g. Movielens DVD Express, and Amazon.

2.3.2. Content-based filtering:

Content-based recommendation systems overcome the limitations of the collaborative filtering by making suggestions based on the content of the items and target user's ratings. Two different content-based approaches have been proposed:

2.3.2.2.Feature-based: Feature-based recommendation systems fetch important features from the program/movie descriptions and learn a user's profile (classifier) using a set of pre-classified attribute vectors, e.g. genre, leading actor/actress in a movie/program recommender system. However, choosing envoy features and appropriately encoding them, is not an easy task.

2.3.2.2. Text categorization-based: Text categorization approach help to learn from thousands of features (words or phrases), but recent researcher has explored that it is possible to build effective classifiers. Several systems using text categorization (TC) have been developed.

Text categorization-based filtering follows four steps to create user profile. First using steaming and stop-word removal algorithm we clean unnecessary word from program synopsis then feature representation technique apply to create weighted vector for every synopsis. Bag of words is the simplest and most frequently used feature representation in TC. Removing the less informative and noisy terms reduces the computational cost and improved the classification performance. All synopsis vectors' features are ranked according to the feature selection mechanism and those with value higher than a threshold are selected. Information gain is our preferred feature selection technique to rank the feature. Then comes feature weighing which can be done by binary weighting technique. At the end using above information we can create implicit profile for every user.[1]

There are two main disadvantages of collaborative filtering. Firstly, the system may not be able to find users with similar preferences to those of the target user. Secondly, there is not enough users' rating to make reliable recommendation.

2.4. Viewing behavior tracking:

In order to recommend appropriate TV programs, a user profile that reflects his / her preference on selecting TV programs should be estimated. The typical features used for generating user profile are watching time and attributes of TV programs. It is assumed that a user watches a TV program for a long time if s/he found it interesting, while a user switches to another channel when s/he found it uninteresting. Attributes of TV programs, such as genres and performers (actors), are used to estimate common features of TV programs that a user is interested in. Such attributes can be obtained from metadata. Users' behaviors of watching TV programs (viewing behaviors) have also been investigated. In the studies, various types of viewing behaviors, such as concerned viewing (The programs regularly watched by user e.g Drama series) and diversion viewing (The programs user not concern about), are observed according to period of time in a day. It is also observed that user's watching behavior in weekday is more stable than that in weekend.

2.5. User profiling:

2.5.1. **Implicit:** Implicit Profile is created on the basis of a user's viewing history. It contains the relevant information about the favorite programs the user has watched, e.g. type, time, date, programs, etc. For example, in an implicit profile, the types of the most often watched programs by a user are recorded as the favorite types in the profile. That is, if a user often watches "The Office" and "Friends", it is very possible that the "Comedy" type will be recorded under "Type" feature. The system also adds the most often watched programs into the "Favorite" Programs" section of the implicit profile. In this case, "The Office" and "Friends" may be included in this user's "Favorite programs" in his/her implicit profile.

2.5.2**. Explicit:** Explicit Profile is created during the first interaction between the system and a user through an interactive interface of TV or a web page access. This profile contains the general interests of a user for all genres of programs, the preferred time and date to watch TV and so on. After the first connection, user profile can also be modified and updated by users, i.e. users can change their preferences. In an explicit profile, a user can specify which program s/he wants to watch, time, date, channel, etc. And these programs will always be recommended to the active user by the explicit recommendation system agent. Explicit method tends to be more accurate than implicit method, while the user's workload for using implicit method is much lower than explicit method.

2.6. Comparison of Text-Categorization technique with feature-based Profiling:

Text-Categorization involves representing each document (movie/drama synopsis) as a feature vector by selecting feature representation, and representing them with weights. There are three approaches for feature representation, 'bag of words' is one of them. Then three techniques for feature selection and information gain give us best result. For feature weighting we could use binary weighting mechanisms. And then we will be eligible to implement machine learning algorithm, in this case we will implement Decision tree algorithm.

Chapter-3

Methodology

3.1. System Architecture and Components:

Our recommender system is composed of following different modules.

a. User Interface.
b. Program List scraping component.
c. Recommendation Module.
d. Data Repository.
e. User Profile Management
    a. Explicit Profile
    b. Implicit Profile

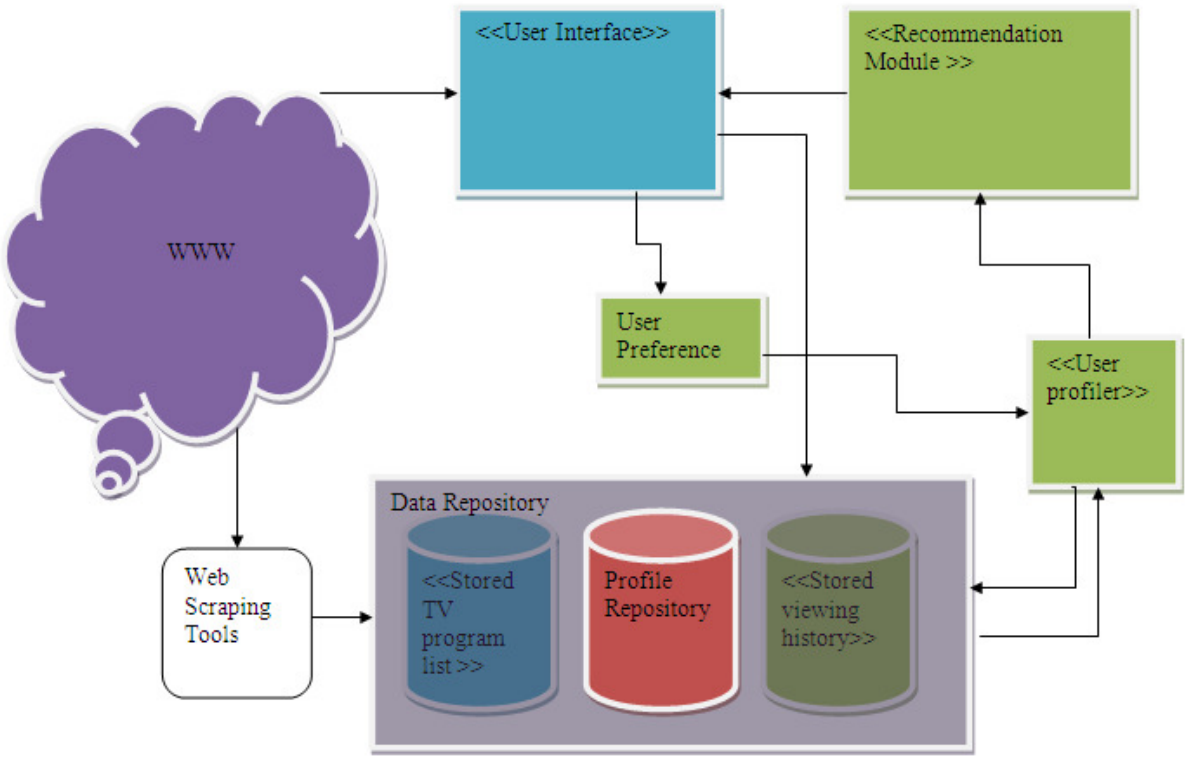


Figure 1

The component diagram shoes the interconnection between components. Detail description of these component describe below,

3.2. User Interface

User interface has been developed on JSF Framework. This framework has ability of dynamic content generation and event-driven application interface. When user selects a TV channel from TV program list, TV Panel display the required channel by online streaming from the web.

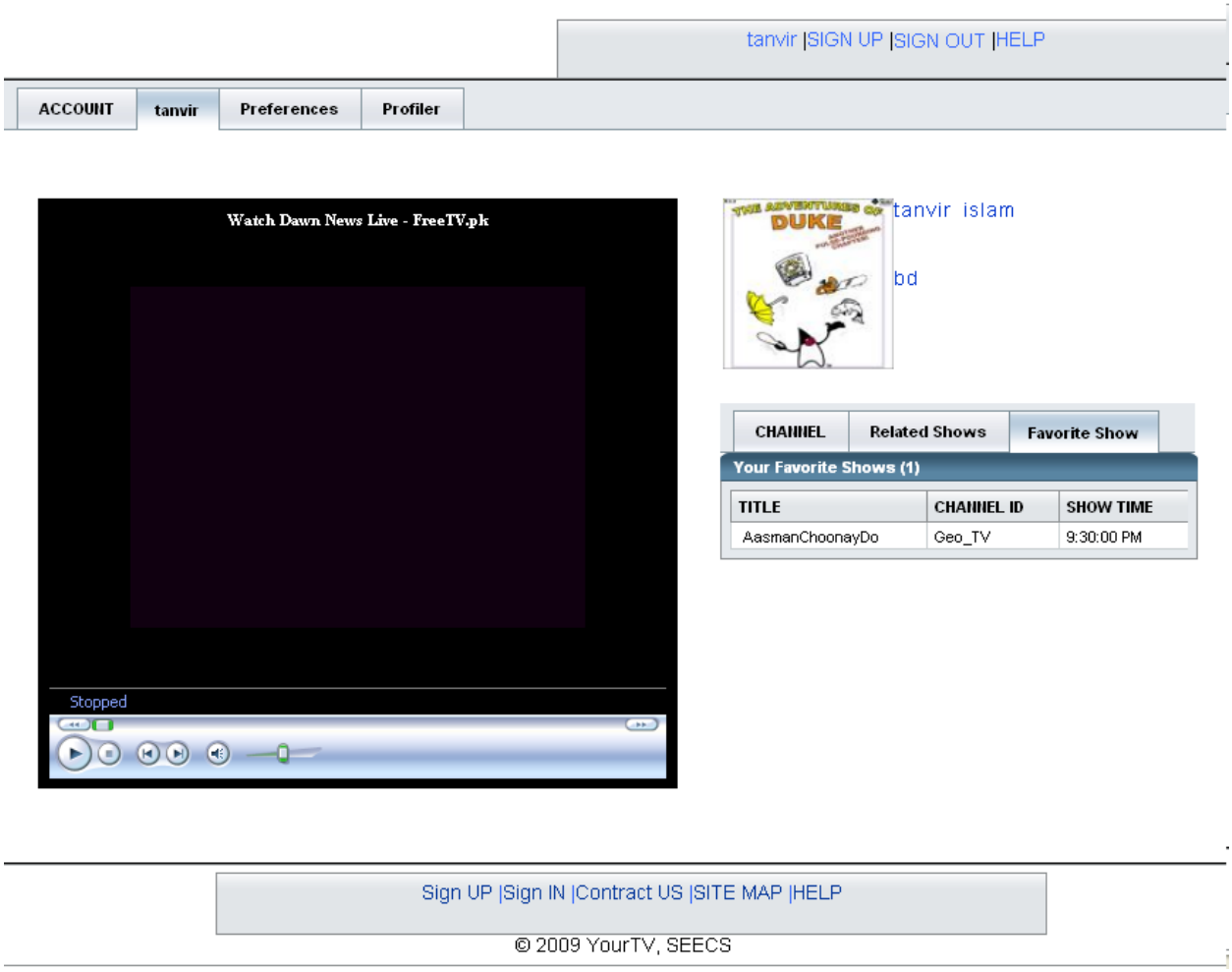


Figure 2

On the right hand side different tabs are showing the navigation to view channel list, Related Shows, Favorite Shows. On the top tab bar shows Account, preference, profile tabs to edit and view other facilities. User Interface provides a convenience way to interact with application. User interface also controls the sequence of recommendation algorithm's execution.

User interface has been developed on J2EE 2.0 and use JSF, JDBC 2.0, Hibernate framework and Ajax api for jsf.

3.3. Scraping tools

This module has been developed to extract TV program schedule from TV channel's web page. First we fetch an html page from a channel that contains schedule of the programs then convert it into XML file and using XPath we query the desire data from the page. This is a limitation of web scraping tool, because of the inconsistency in html pages of different TV channels. Due to this problem a generic algorithm for all channels is not possible.

Technology is used: NUX Java XML API + TagSupe framework.

3.4. Store viewing history

To store different types of information for our system a database is maintained. It store TV program schedule of different TV channels and user profile (explicit and implicit). When a user login to the system and select a TV channel from the list, we can predict logs in that moment which programs the user is going to watch, as we have the list about TV program. So a back ground process/module can create a log about users viewing history. Here we can track some metrics like program's name, type, actors, director, rating etc by assuming that data in the database are updated. Following table shows different fields of user interest. And this table is dynamic in nature, when the user change his/her behavior, records in this table will change accordingly.

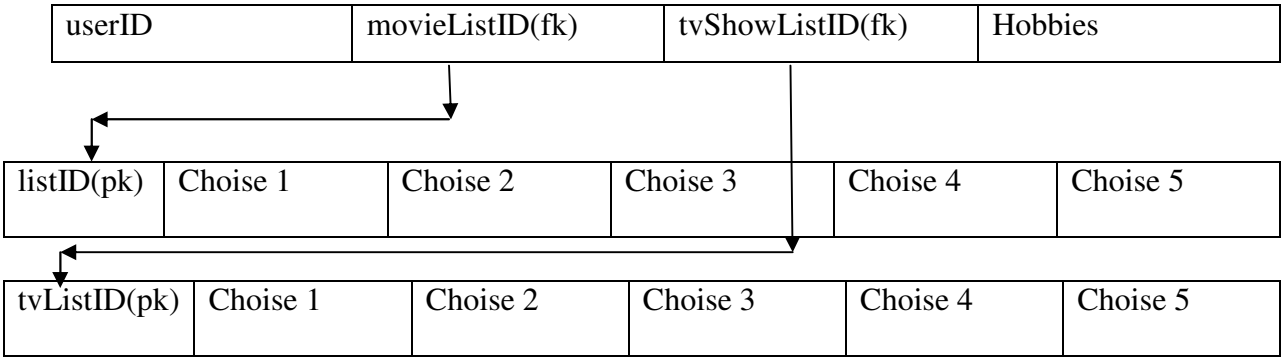
TITLE	SHOW_TIME	START_TIME	DURATION	LASTCHANNELID	USERID	RATING	Date	

3.5. User profile: Implicit

Implicit Profile is created on the basis of a user's viewing history. It contains the relevant information about the favorite programs the user has watched, e.g. Type, time, date, programs, etc. For example, in an implicit profile, the types of the most often watched programs by a user are recorded as the favorite types in the profile. That is, if a user often watches "Nadia Khan Show" and "Morning with Hum", it is very possible that the "TV Magazine Show" type will be recorded under "Type" feature. The system also adds the most often watched programs into the "Favorite Programs" section of the implicit profile. In this case, " Nadia Khan Show " and "Live with Talat" may be included in this user's "Favorite programs" in his/her implicit profile.

3.6. User profile: Explicit

Explicit profile is created using primary information of user. This information manually entered by the user at time of account creation or any time later. When user asks to input his/her favorite programs/movies, there will be five choices, which help us to create more successful user profile. Following tables show the different fields of explicit profile tables.



3.7. Recommender: Main module

User profile and stored information of TV program list are used to create recommendations. CB RS (Content-based recommendation system) selects the programs from the available programs on the basis of the user's implicit profile. For example, a user likes action, adventure, drama etc the preferred time and date is in the evening of weekend, and then we have a movie Indiana Jones which is an action, Adventure drama movie and will be available at 9:00 this Saturday, then there is a possibility that this movie will be recommended to the active user since this movie is very similar to the user's favorite programs.

3.8. Use Case Diagram:

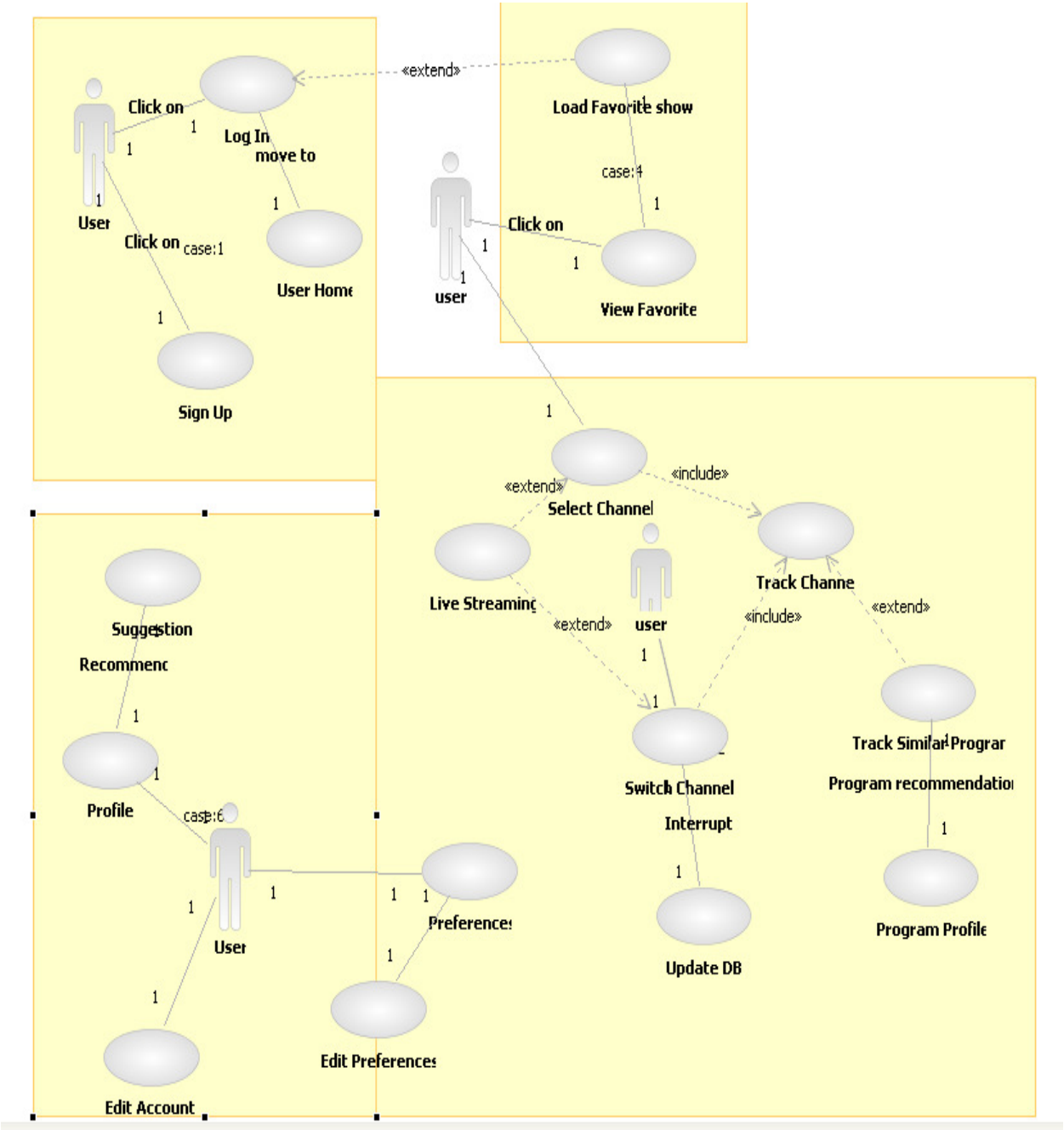


Figure 3

Case 1:

Name: Primary Interaction with system.
Entry Condition:

a. **Account already created, only login to system.**

b. **Account not created, first create account then login to system.**

Flow of Events:

- User click to login button, display login page.
- Type user id and password to enter into application.
- Otherwise create new account to login.
- If login is successful , then display UserHome.jsp
- Which contain information like TV list, preference, user profile is displayed.

Exit Condition: User click "Sign OUT ".

Identify Objects Related to Case 1:

Entity Objects: user_info, channel_list, user_preference, user_profile
Control Objects: HTML Input Controls, Mysql connection pool.
Boundary Objects: Home.jsp, Login.jsp, createNewAC.jsp…..etc

Case 2:

Name: Select Channel to View.
Entry Condition: User click TV Channel list.
Flow of Events:

- Initialize Tracker Object with basic information like userId, channelId,Time etc
- Start a thread with tracker object to track a program on selected channel.
- Send request to live streaming server to start TV channel.
- Tracker thread track a channel till a program finished itself or user switch the channel
- In any case an interrupt will occur and according to duration a program watched by the user, assign a rating to that program.

Exit Condition: program is over or user switches him/her self.

Identify Objects Related to Case 2:

Entity Objects: viewing_history, user_info, program_list
Control Objects: tvProgTracker , tvProgTrackerThread
Boundary Objects: UserHome, SessionBean1, ApplicationBean1

Case 3:

Name: Switch channel
Entry Condition: User click to change channel
Flow of Events:

- Whenever user change channel, last Tracker Thread will interrupted and program data updated to database.
- For each tracker thread there is an another thread , responsible for finding similar program running at that time on other channel
- Bag of word is a text categorization technique to find out similarity between programs synopsis, so that we mark them as a related program and create programs profile, its help us to make recommendation without user profile information.
- Two conditions must apply, programs are must running and old profile must erase.

Exit Condition:

Identify Objects Related to Case 3:

Entity Objects: program_profile, program_list
Control Objects: FindSemilarProgThread, tvProgTracker
Boundary Objects:

Case 4:

Name: load favorite program into user session
Entry Condition: user login or click on "Favorite Show" tab on UserHome.jsp page
Flow of Events:

- When user login to the system session object initiated and load a data base table with favorite program list.
- This list has variable number of programs, those have been changed time to time depends on user viewing history of user.
- List of favorite shows are change according to viewer's habit.

Exit Condition:

Identify Objects Related to Case 4:

Entity Objects: favorite_program_list, program_list, user_info
Control Objects: SessionBean1, ApplicationBean1
Boundary Objects: UserHome

Case 5:

Name: load user profile and make suggestions (recommendations)
Entry Condition: click on user profile.
Flow of Events:

- ¬ Initiate profile loader object that create decision tree from training data set.
- ¬ Training dataset populated from viewing history table and from detail program tables.
- ¬ Then using decision tree make suggestion or recommendation to user.

Exit Condition:

Identify Objects Related to Case 5:

Entity Objects:
Control Objects: Create Profile, Manage Profile.
Boundary Objects:

3.9. Class Diagrams:

3.9.1 Class Diagram of main system module

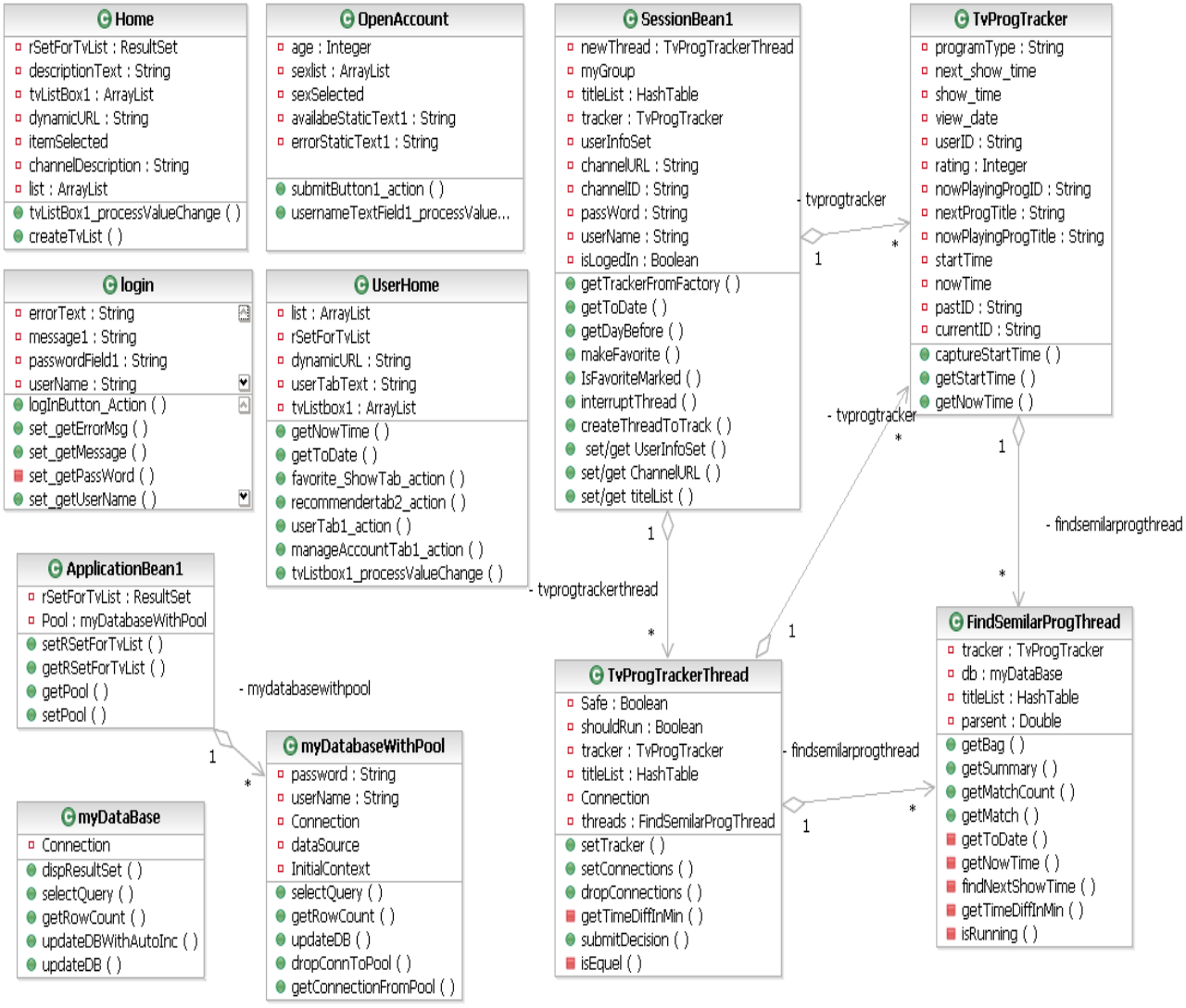


Figure 4

3.9.2 Class Diagram of scraping module

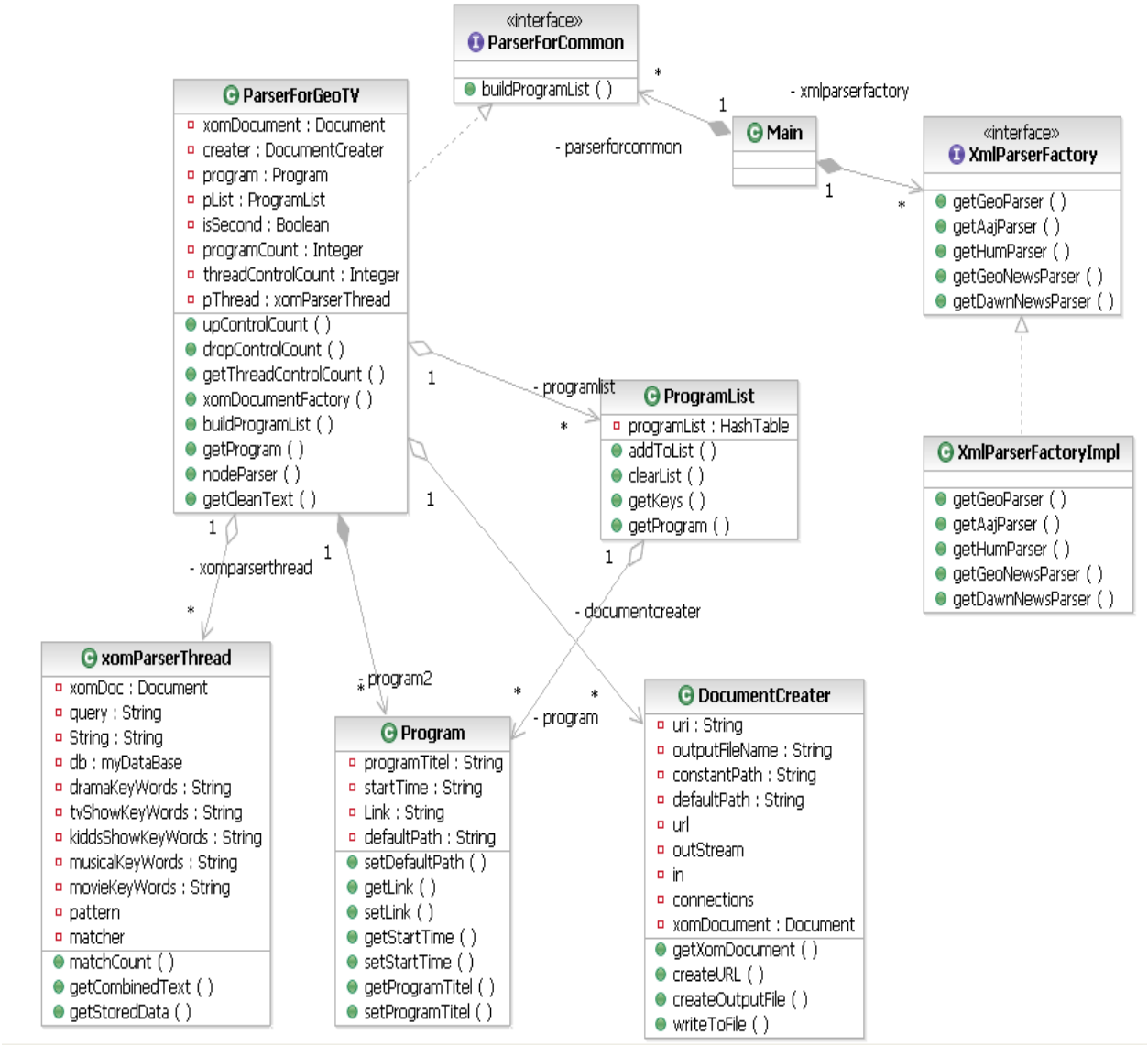


Figure 5

3.10. <u>User Interface Navigation:</u> Following diagram shows the UI navigation for the recommender system.

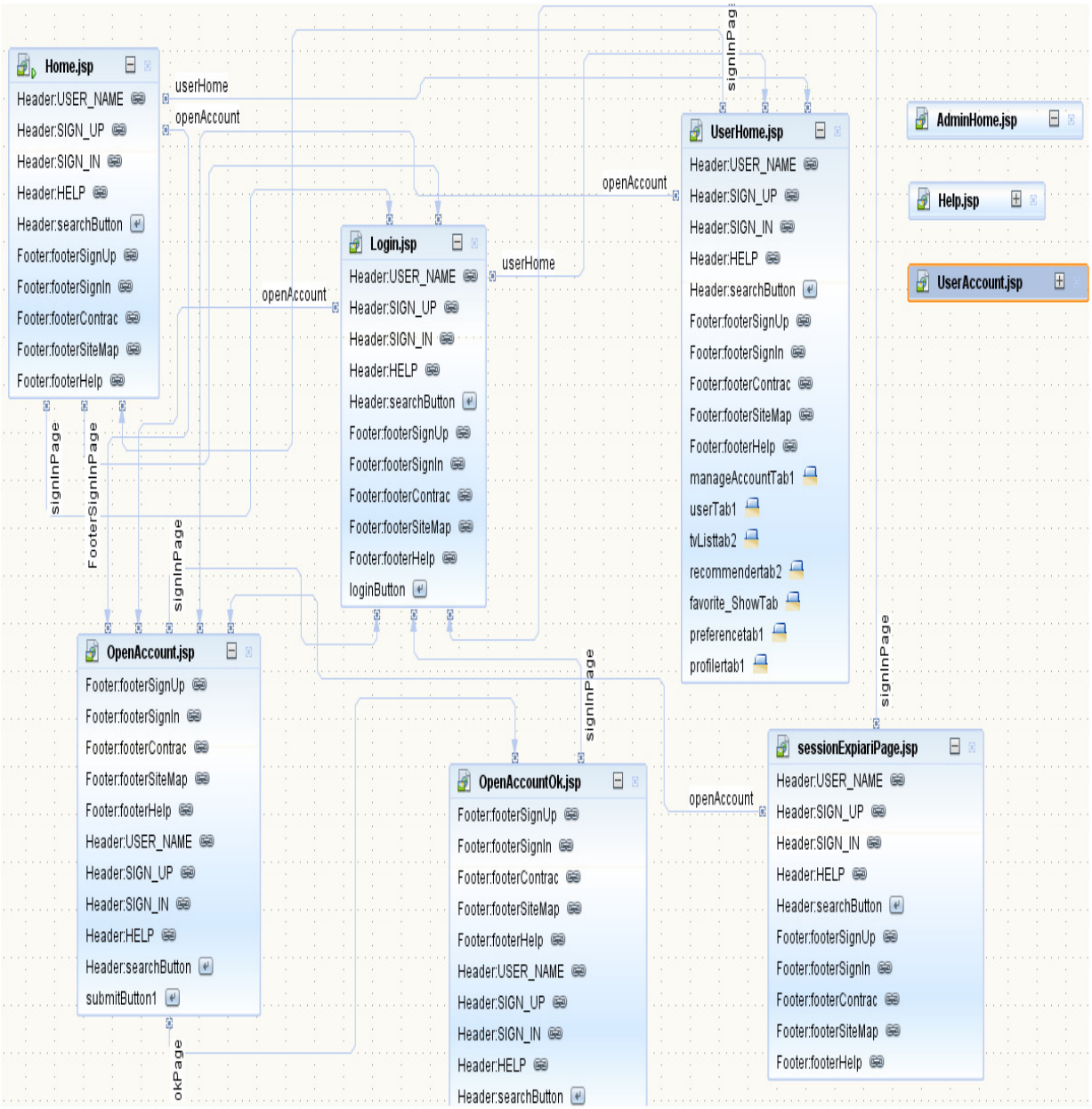


Figure 6

3.11. Conceptual Component Model:

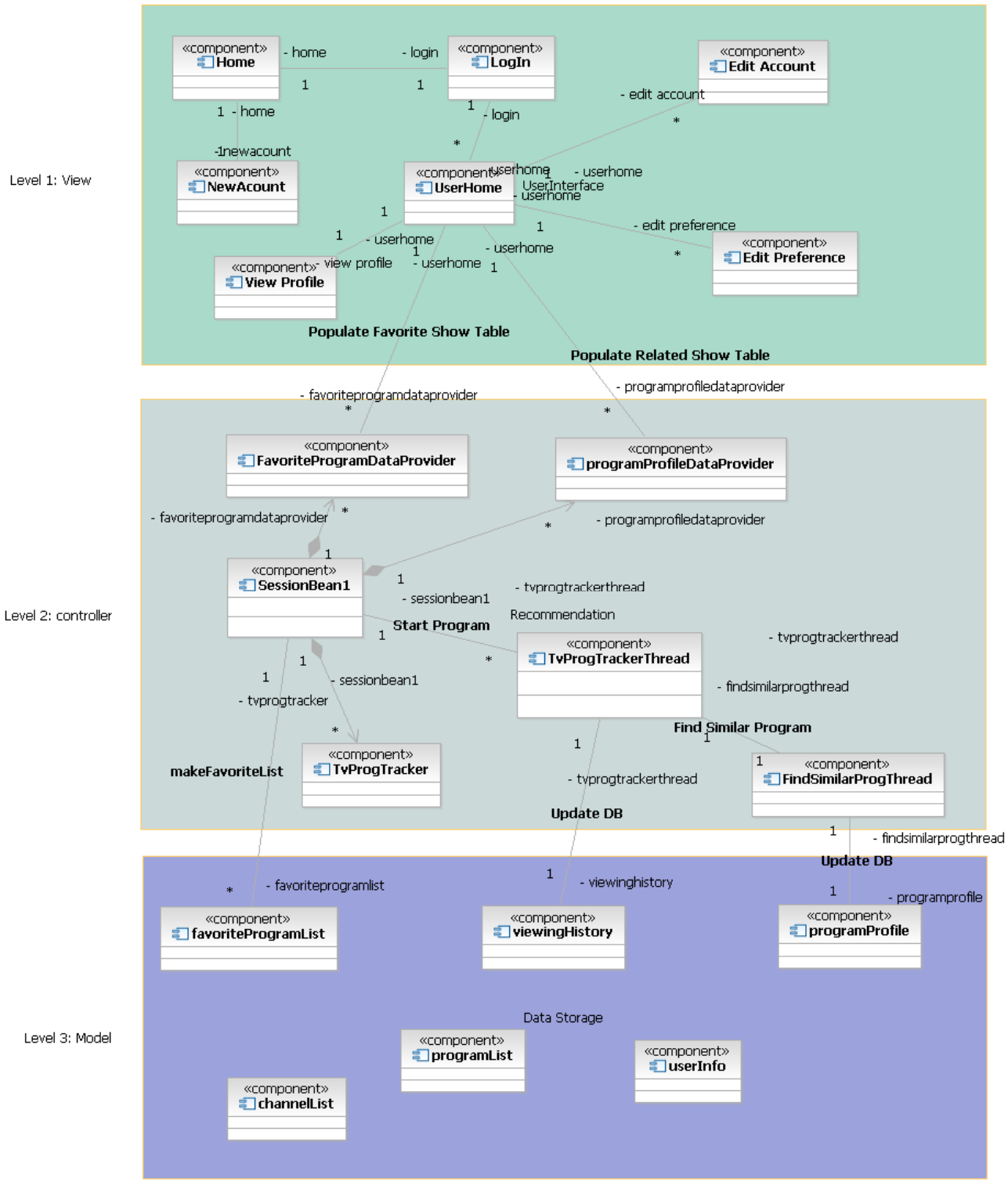


Figure 7

3.12. Sequence Diagram:

3.12.1 Interaction 1: User selects a channel to watch live TV show. In this sequence diagram we show the interaction between user and application. Threads are starting to track the current TV program watching by the viewer and create similar TV program profile for recommendations.

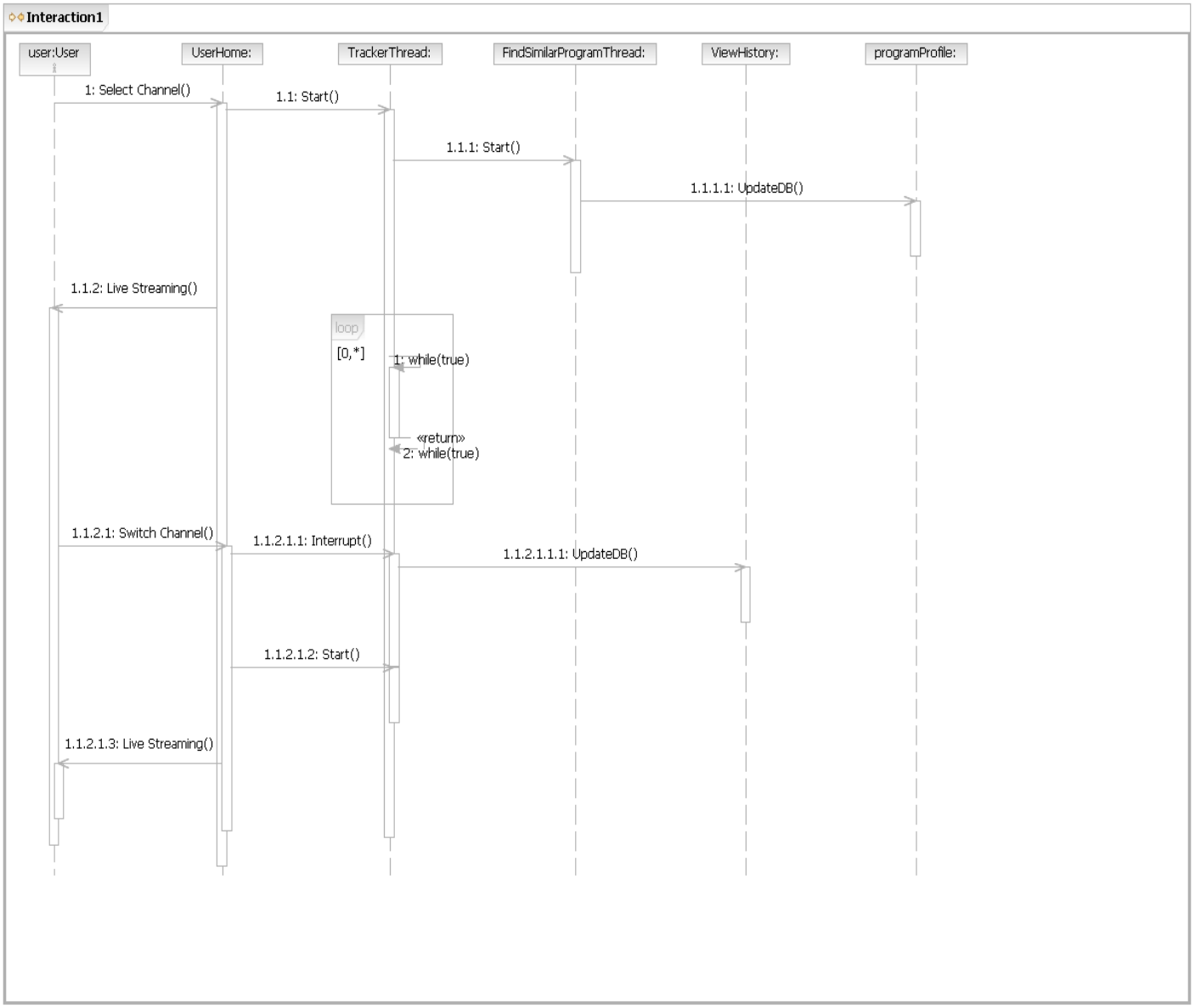


Figure 8

3.12.2 Interaction 2: When a user login to the system, a session is created by application framework. When a session object is created, it calls a function makeFavorite() to load a favorite program list for that particular user. Now whenever user click on favorite show tab it display updated favorite program list.

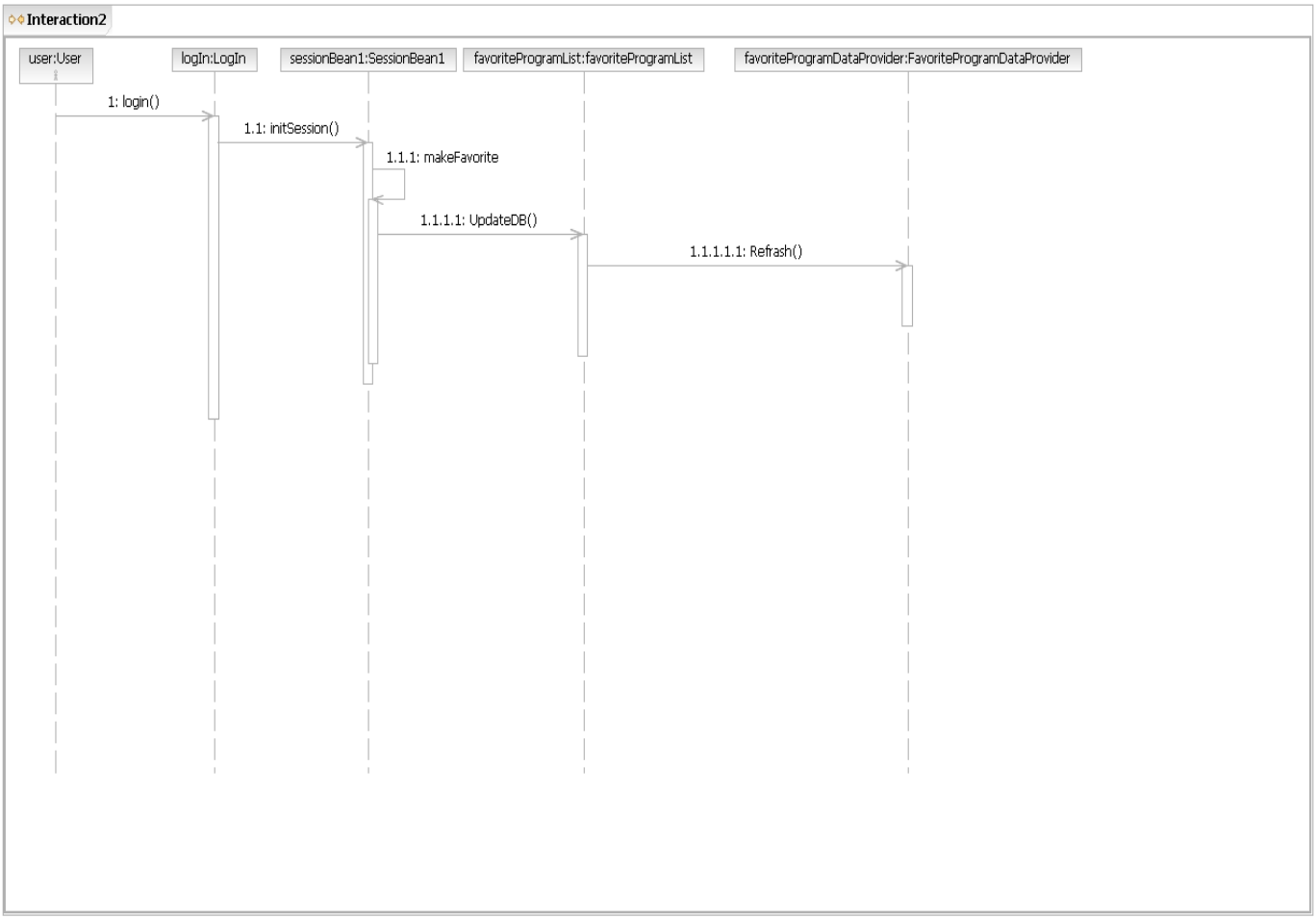


Figure 9

3.13. ERD Model: Following diagram shows ER model for the system.

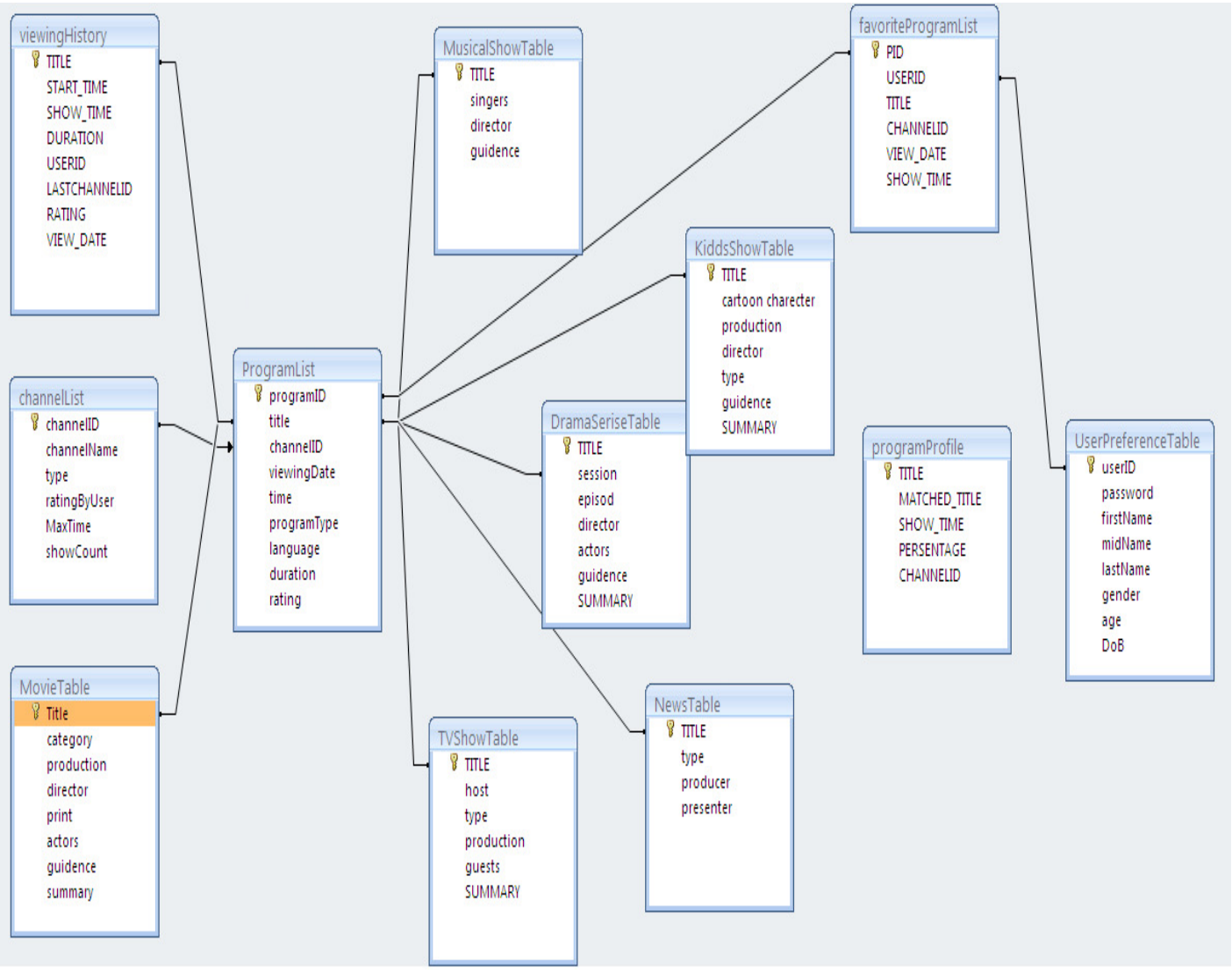


Figure 10

Chapter-4

Algorithm Implementation & Results

In this chapter we'll discuss implementation of different algorithms for tracking user behavior so that we can build user implicit profile and programs profile.

4.1. Tracking a user and update Viewing History:

When user selects a channel, a thread is started on the server, which is responsible for tracking the program currently running on that channel. An object of tvProgTracker class is instantiated with some initial values like channelID, username, startTime of current thread as the program start time. Then two functions findCurrent() and findNext() are used to find the current and the next program. findCurrent() and findNext() function of tvProgTrackerThread then update some other fields to carry information to identify whether current program is running or not. They also help to decide the duration of the program. So that current program can trigger interrupt itself. Otherwise when a user switch as channel, current thread will be interrupted and a new thread will be started. In the mean time we make some ranking points to that program and insert into viewingHistory table or update old value.

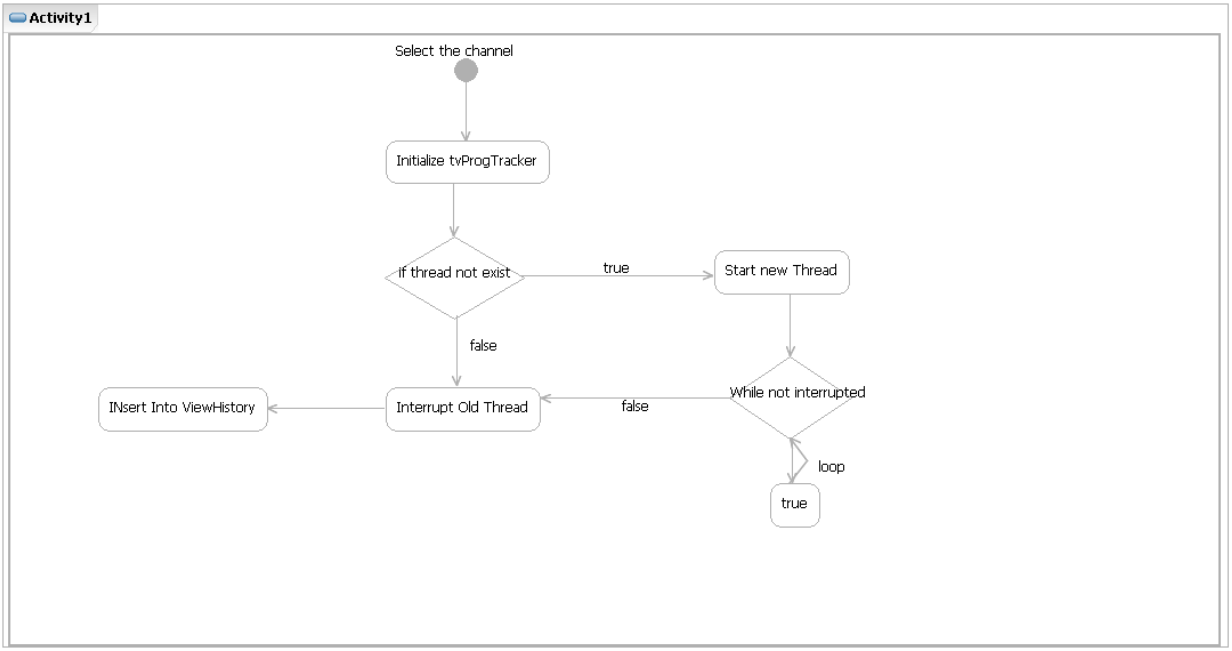


Figure 11

This helps the system to easily change the user mode/choice about a program. So rating can vary daily or weekly basis.

Evaluation of Watched TV Program: user's selection of TV channels during watching TV is stored as the log of the user's TV watching behavior. From the log, the rating of a TV program by the user is estimated based on the watching time, which can be calculated by following equation,

Scorew(*p*): a score based on watching time

The score of a TV program *p* based on watching time is defined by Eq. (1), where *wtime*(*p*) and *airtime*(*p*) indicate the amount of time (minutes) a user watched and its airtime, respectively.

$$Scorew\ (p) = wtime\ (p)/airtime(p) \ldots\ldots (1)$$

4.2. Finding today's Favorite Show:

When user clicks on Favorite Show tab, a session method makeFavorite() is called. This method is responsible for finding today's favorite shows which depends on program rating for last week.

Algorithm:

List Title <= fetch_rows from viewingHistory table when date <= TODATE() AND userID=currentID();

While(Title)

Then check <= is this Title exist or not in programList table.

If (check =exist)

Then inset into favoritePogramList table

Else update into favoritePogramList table

End if

End While

4.3. Finding Related Shows to provide recommendation:

Following algorithm finds related shows to provide recommendations. tvProgTrackerThread pass tracker object to findSimilarProgThread

Algorithm:

Hashtable mainBag = getBag (currentTitle)

RowSet <= fetch_rows from programList when date = todate() AND time=Current() AND programType= pType

While (RowSet exist)

Then tempBag<=getBag(tmpTitle)

If (getMatch (mainBag, tempBag) is true)

Then checkRowSet <= fetch_row from programProfile when Title=currentTitle AND matchedTitle=tempTitle;

If(isRunning(tempTile))

Then

If (checkRowSet not exist)

Then insert into programProfile table

Else Update last entry into programProfile

End if

End if

End if

End while

4.4. Implementation of User Profile:

After successfully tracking viewing history we can develop user profile which gives result partially correct results. To do this we maintain a table called favoriteProgramList with following structure,

PID	TITLE	USERID	CHANNELID	VIEW_DATE	SHOW_TIME

For populating data, we first fetch data from viewingHistory table for last 7 days. Then we check through a loop to find out if this Title exists in today's TV schedule list. If it exists then we submit data into table along with user ID. When a user clicks on "Favorite Show" tab, a session function makeFavorite() is called and favoriteProgramList table is updated with new data or only refresh data.

4.5. Generating Program profile:

To populate program profile we use programProfile table with following structure,

TITLE	MATCHD_TITLE	SHOW_TIME	Percentage	CHANNELID

There is a class FindSimilarProgThread which has responsibility to clean up stop words (e.g. like you, me, he etc) from synopsis and then create beg of word with weighed vector, which is iteratively matched by other program's bag of word. If get match successfully then it is insert into programProfile table.

4.6. Results:

4.6.1. Finding Related Programs

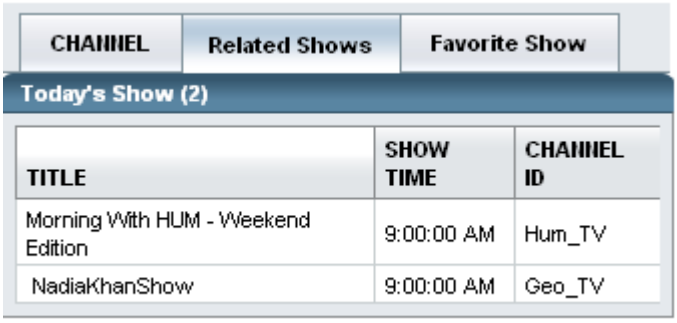


Figure-12

This figure shows that 9 am "NadiaKhanShow" is running on Geo TV, and besides that application shows that "Morning With HUM….", is a similar program currently running on Hum TV so the user can watch that show.

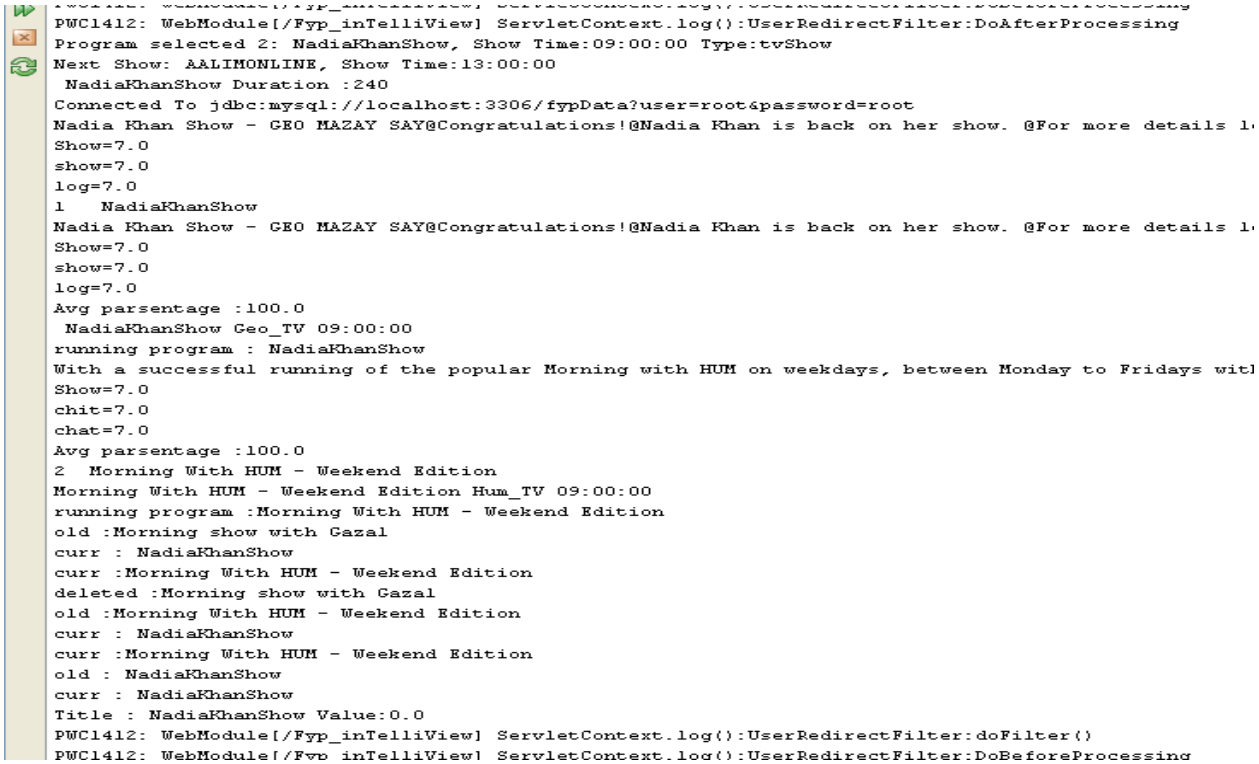


Figure 13

Figure 14 shows the percentage of content matched between "NadiaKhanShow" and "Morning With HUM…." TV show. We implemented feature representation techniques here.

Favorite Show:

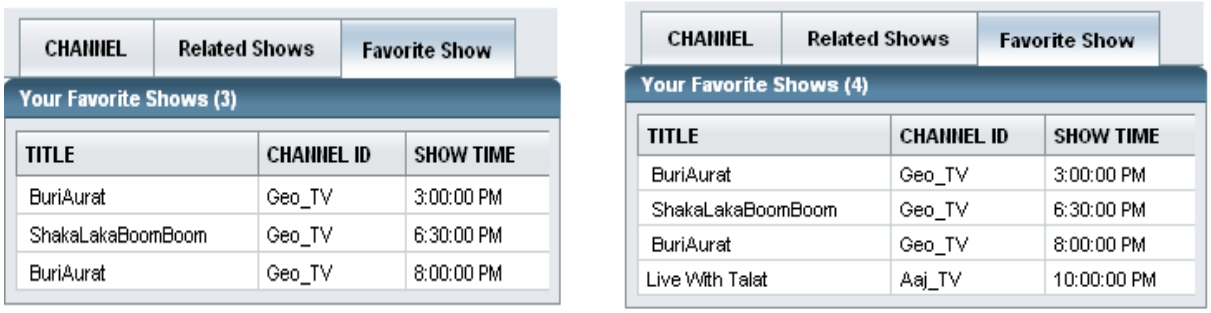


Figure 14

We have developed user profile based on their TV watching pattern. When a user logs in to the system we refresh his/her profile with updated data. Which is shown in the table, 1st table was captured when user logs in first time and after watching some time may be his profile being updated, and so 2nd table reflects that behavior.

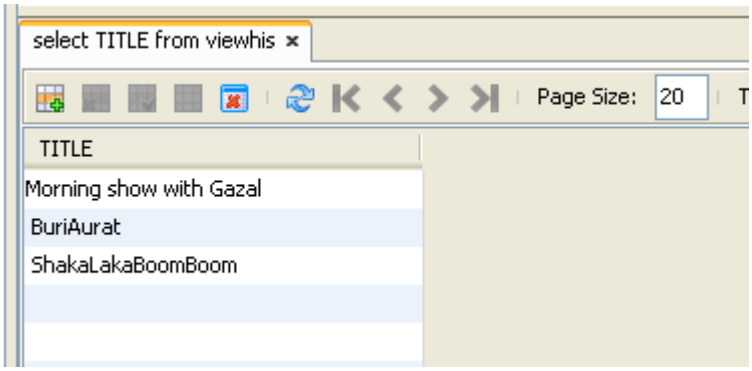


Figure-15

Chapter-5

Conclusion

Currently there are number of recommendation systems available in market, which implement different techniques and algorithms to develop their application. We assume that best prediction could be possible by merging different algorithms together.

We have followed content based approach for recommendation and implement text categorization techniques to gain best performance. On the other hand most of the current implementations of recommendation systems follow collaborative approach which does not give good performance. But combination of both gives best result. Our application currently provide recommendation using text-categorization algorithm, on the other hand we collect the user profiles on common database, which eligible us to make collaborative filtering and combine both result to achieve best performance of our recommender system.

Chapter-6

Future Works

We have partially implemented user profile but once we are able to gather cleaned categorize metadata about programs then we can implement decision tree algorithm to implement complete implicit user profile. Future works includes,

- Implementation of decision tree algorithm to complete implicit user profile.
- Design Training data from Viewing History table.
- Design Test data from program list table.
- Merge both Implicit and Explicit profile to achieve better prediction.

References

[1] INTIMATE: A Web-Based Movie Recommender Using Text Categorization by Harry Mak, Irena Koprinska and Josiah Poon.

[2] Collaborative Filtering for Information Recommendation Systems by Anne Yun-An Chen and Dennis McLeod

[3] A Fuzzy Linguistic Summarization Technique for TV Recommender Systems by A. Pigeau, G. Raschia, M. Gelgon, N. Mouaddib, R. Saint-Paul

[4] Mining Association Rules from TV Watching Log for TV Program Recommendation by Yasufumi Takama and Shunichi Hattori.

[5] Recommendation Based on Personal Preference by Pei Wang

[6] Recent Development of Recommender Systems by Cheng-Ting Wu1, Hsiao-Fan Wang 2

[7] Recommender systems by Alexander Felfernig and Gerhard Friedrich

[8] Dynamic Personalized TV Recommendation System by Weizhen Dai and Robin Cohen

[9] http://en.wikipedia.org/wiki/TiVo

[10] http://en.wikipedia.org/wiki/Netflix

[11] http://www.movielens.org/