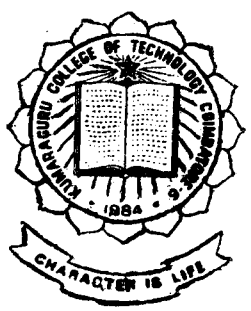
Information Brokerage by Software Agents

Project Report 1997 - 98

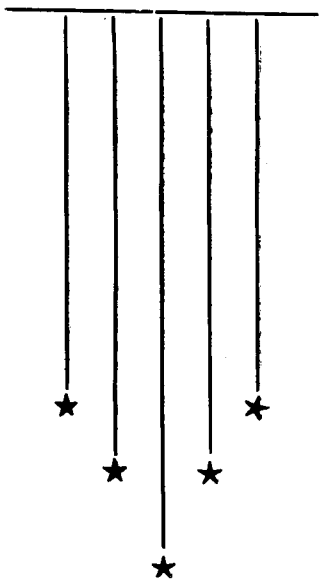


Submitted by

Jacqueline Helen Pinheiro
S. Nithya
J. Arun Swaran

Guided by

Prof. P. Shanmugam,
B.E., M Sc (Engg), M.S. (Hawaii)
M.I.E.E., M.I.S.T.E.



In partial fulfilment of the requirements
for the award of the Degree of
BACHELOR OF ENGINEERING
IN COMPUTER SCIENCE AND ENGINEERING
of Bharathiar University

Department of Computer Science and Engineering
Kumaraguru College of Technology
Coimbatore - 641 006

Department of Computer Science and Engineering

Kumaraguru College of Technology

Coimbatore - 641 006

Project Work 1997 - 98

Name ..... Register No. .....

Certified that this is the Bonafide Record of the
Project Work Done by

.....

In partial fulfilment of the requirements for the
award of the Degree of Bachelor of Engineering
in Computer Science and Engineering
of the Bharathiar University.

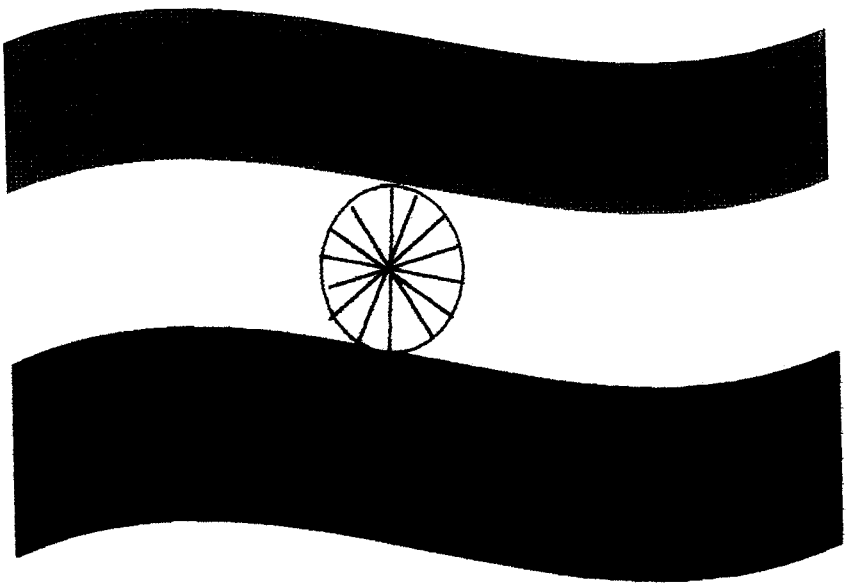
.....
Head of the Department

.....
Guide

Submitted for the University Examination held on .....

.....
Internal Examiner

.....
External Examiner



Dedicated

To Our Beloved Parents

And

Mother Land

ACKNOWLEDGEMENT

ACKNOWLEDGMENT

We sincerely thank our principal, **Dr. S. Subramanian**, B.E., M.Sc (Engg), Ph.D., MISTE for the patronage and innumerable facilities given by him for our project work.

Our heartiest thanks are due to our professor and guide **Prof. P.Shanmugam**, B.E., MSc(Engg), M.S (Hawaii) M.I.E.E.E, M.I.S.T.E , Head of Department of Computer Science and Engg. for all the pains he had taken to provide us with all the required material and facilities needed for the project .We record our deep gratitude to him for his valuable guidance and suggestions given through out the project.

We express our extreme gratefulness to our class advisor **Miss R.Sumathi** B.E, M.S, for the helping hand she had extended to us throughout the project without which the successful completion of the project was unimaginable.

Our heartiest thanks are due to **Mrs. Aruna Sankaranarayanan** of Brandies University, Massachusetts for introducing us to *software agents* and her valuable suggestions during the initial stages of the project. We extend our heartiest gratitude to Ms. Kerla Kaumudi for showing interest in this product even before it was released.

We also thank all the members of the faculty for their kind help and encouragement. We express our sincere thanks to all the non-teaching staff for supporting us and the indispensable services they rendered day and night through out the project.

We would like to mention the timely help provided by **Mr.Sundaramurthy,** Head of the Department of Electrical and Electronics Engineering, Coimbatore Institute of Technology, for the timely help offered by him in getting some manuals when we needed it the most.

We are pleased to thank Mr.Rampi and the members of **Center for Advanced Technologies (CAT), Wipro Systems,** Banglore for providing us with all information regarding Java, agents and Internet working needed by us to proceed with the project.

We sincerely thank all our friends who contributed in the form of ideas, suggestions and encouragement for the successful completion of this project successfully.

CONTENTS

CONTENTS

SYNOPSIS

1. SYNOPSIS

The product ***KCT INFO AGENT 1*** is a highly sophisticated and user friendly package built using JDK1.1 (Java Development Kit) and Front Page. This is basically a ***software agent used for information brokerage.***

Prime facie, the user enters the web-site and inputs all the relevant information the user's personal details i.e. the user's characteristics, expectations and contact details. In his expectations the user specifies the characteristics he/she prefers for his/her prospective i.e. his/her age, income, qualification, profession, language, height etc. The user enters his bio-data and also gives his contact details. The agent then obtains the relevant information regarding the user's preferences and based on this the agent performs a ***fuzzy*** based search. The search is based mainly on three characteristics of the user, which are his/her age, income and height. After performing the search the user is presented with a list of other users who closely satisfy his preferences. The user can then obtain the contact details of his/her prospective by simply clicking on the list.

The agent performs the search on the local server then the agent ships itself to the remote server and then searches the remote database based on the user's preferences. This search is again a fuzzy based search. The agent then travels back to the local server and displays the results of the search.

An epitome of the product features is presented to the user apriori to make the capabilities of the agent perspicuous to the user. The core features are enhanced by an easy to use graphical user interface and on-line help. These enhance the user friendliness and the interactivity of the system. Further, the user has nothing to strain but his fingertips to key in his data.

PREFACE

2. PREFACE

In this section we give a brief description on the organization of the documentation. In the first section namely the Need For The Product we give a presentation of the need of the product and its advantages with regard to the people involved in the product namely the organization, the end user and the developer. In the next section, we elaborate on the benefits enjoyed by the user by using the product.

In the technical overview section we present a general overview of the technical concepts we have used to develop the product. All the concepts used in developing the product namely JDBC, Applets security box and Agent concepts are discussed in detail.

In the Design Overview section we give an overview of design of the product which basically consists of the design of databases, fuzzy expert system and GUI. In the project implementation section we discuss in detail the implementation of JDBC, Applets sandbox, GUI and Agents with respect to our project.

The basic requirements to be satisfied to run the product is discussed in detail in the next section namely Requirements For The Product.We speak of the requirements of the product from the view point of the organization and the client. The advantages and drawbacks of our product are discussed in brief in the following sections. Scope for upgradation and future expansion are also discussed in the next section.

GENERAL OVERVIEW OF THE PRODUCT

3. GENERAL OVERVIEW OF PRODUCT KCT INFO AGENT-1

3.1 Need for the Project

The people who will be benefited from the creation of the product *Kct Info Agent-1* can be viewed as falling into three categories:

1) The Organizations (who would host this product in their web servers)
2) The End Users of the product
3) The Developers

3.1.1. ***Organizations:***

Today, computers rule the world and information superhighway is in vogue. Many companies, firms and other organizations do a lot of gimmicks just to attract surfers to their websites and make their organization more popular. Hence, the hosts or the organizations i.e. the daily have decided to use our product *Kct Info Agent-1* as part of a strategy to attract many people. This product is particularly aimed at South Indian Dailies which want to maintain a matrimonial column in their daily.

Online Dailies can install this product in their web servers and can help their users. Hence, as the service offered by these dailies increases, which in turn increases the number of users and which in turn increases the number of hits per day for the web pages of the online dailies.

BENEFITS FOR GENERAL PUBLIC

4. BENEFITS FOR THE GENERAL PUBLIC

The soul objective of our software is to enable the end user to enjoy the following aspects of our product.

a) Computerization of the entire process

The manual system involves a sequential process of first putting an advertisement in the daily, then obtaining the responses after a short period of time and then filtering these responses based on the user's preferences.

The entire process happens to be time consuming besides involving a large amount of work in the search procedure, there is also a small amount of cost involved in the process. Almost all of the above mentioned problems are overcome by the *KCT Info Agent-1* and the entire process is greatly simplified. The user only has to enter the data the remaining process is taken care of by the agent who acts on behalf of the user.

b) Easy Accessibility

The next major advantage is the easy availability of the product on the World Wide Web. All the relevant information about the user is stored in the database available at the web site of the organization, and the search operation is performed on the database.

5. TECHNICAL OVERVIEW

The *KCT Info Agent* was build using the following ***software***'s:-

a) Programming language - JDK 1.1.4
b) Database - MS SQL Server or other RDBMS
c) Creating Interactive Web pages - Frontpage '97
d) Design & Simulation of Fuzzy Expert System - MATLAB
e) JDK 1.1 Enabled Browser- Internet Explorer 4.0 or Netscape Navigator 5.0

☆ The major ***concepts*** we have dealt with in developing the product *KCT Info Agent* are the following:-

a) **JDBC**
b) **Applets that can run in the sandbox of Java.**
c) **Software Agents**
d) **Fuzzy Logic**

In this section, we provide a brief account of the above mentioned concepts and in the sections that follow we discuss how these concepts have been implemented with respect to this project.

5.1 JDBC (Java Database Connectivity)

5.1.1 ***Introduction:***

JDBC is a Java API for executing SQL statements. (As a point of interest, JDBC is a trademarked name and is not an acronym; nevertheless, JDBC is often

thought of as standing for "Java Database Connectivity".) It consists of a set of classes and interfaces written in the Java programming language. JDBC provides a standard API for tool/database developers and makes it possible to write database applications using a pure Java API.

Using JDBC, it is easy to send SQL statements to virtually any relational database. One can write a single program using the JDBC API, and the program will be able to send SQL statements to the appropriate database. And, with an application written in the Java programming language, one also doesn't have to worry about writing different applications to run on different platforms. The combination of Java and JDBC lets a programmer write it once and run it anywhere.

5.1.2 Why Java for Database Connectivity?

Java, being robust, secure, easy to use, easy to understand, and automatically downloadable on a network, is an excellent language basis for database applications. What is needed is a way for Java applications to talk to a variety of different databases. JDBC is the mechanism for doing this. JDBC extends what can be done in Java. For example, with Java and the JDBC API, it is possible to publish a web page containing an applet that uses information obtained from a remote database, or an enterprise can use JDBC to connect all its employees (even if they are using a conglomeration of Windows, Macintosh, and UNIX machines) to one or more internal databases via an Intranet. With more and more programmers using the Java programming language, the need for easy database access from Java is continuing to grow.

MIS managers like the combination of Java and JDBC because it makes disseminating information easy and economical. Businesses can continue to use their installed databases and access information easily even if it is stored on different database management systems. Development time for new applications is short. Installation and version control are greatly simplified. A programmer can write an application or an update once, put it on the server, and everybody has access to the latest version. And for businesses selling information services, Java and JDBC offer a better way of getting out information updates to external customers.

5.1.3 What does JDBC do?

Simply put, JDBC makes it possible to do three things:

* Establish a connection with a database.
* Send SQL statements to database for execution.
* Process the results.

5.1.4 Why use JDBC?

At this point, Microsoft's ODBC (Open Database Connectivity) API is probably the most widely used programming interface for accessing relational databases. It offers the ability to connect to almost all databases on almost all platforms.
So why not just use ODBC from Java?

The answer is that you can use ODBC from Java, but this is best done with the help of JDBC in the form of the JDBC-ODBC Bridge, which we will cover

shortly. The question now becomes, "Why do you need JDBC?" There are several answers to this question:

1) ODBC is not appropriate for direct use from Java because it uses a C interface. Calls from Java to native C code have a number of drawbacks in the security, implementation, robustness, and automatic portability of applications.

2) A literal translation of the ODBC C API into a Java API would not be desirable. For example, Java has no pointers, and ODBC makes copious use of them, including the notoriously error-prone generic pointer "void *". You can think of JDBC as ODBC translated into an object-oriented interface that is natural for Java programmers.

3) ODBC is hard to learn. It mixes simple and advanced features together, and it has complex options even for simple queries. JDBC, on the other hand, was designed to keep simple things simple while allowing more advanced capabilities where required.

4) A Java API like JDBC is needed in order to enable a "pure Java" solution. When ODBC is used, the ODBC driver manager and drivers must be manually installed on every client machine. When the JDBC driver is written completely in Java, however, JDBC code is automatically installable, portable, and secure on all Java platforms from network computers to mainframes.

In summary, the JDBC API is a natural Java interface to the basic SQL abstractions and concepts. It builds on ODBC rather than starting from scratch, so programmers familiar with ODBC will find it very easy to learn JDBC. JDBC retains the basic design features of ODBC. The big difference is that JDBC builds on and reinforces the style and virtues of Java, and, of course, it is easy to use.

5.1.5 Two-tier and Three-tier Models:

The JDBC API supports both two-tier and three-tier models for database access. In the two-tier model, a Java applet or application talks directly to the database. This requires a JDBC driver that can communicate with the particular database management system being accessed. A user's SQL statements are delivered to the database, and the results of those statements are sent back to the user. The database may be located on another machine to which the user is connected via a network. This is referred to as a client/server configuration, with the user's machine as the client, and the machine housing the database as the server. The network can be an intranet, which, for example, connects employees within a corporation, or it can be the Internet.

In the three-tier model, commands are sent to a "middle tier" of services, which then send SQL statements to the database. The database processes the SQL statements and sends the results back to the middle tier, which then sends them to the user. MIS directors find the three-tier model very attractive because the middle tier makes it possible to maintain control over access and the kinds of updates that can be made to corporate data. Another advantage is that when there is a middle tier, the user can employ an easy-to-use higher-level API which is translated by the middle tier into the appropriate low-level calls. Finally, in many cases the three-tier architecture can provide performance advantages.

Until now the middle tier has typically been written in languages such as C or C++, which offer fast performance. However, with the introduction of optimizing compilers that translate Java bytecode into efficient machine-specific code, it is becoming practical to implement the middle tier in Java. This is a big plus, making it possible to take advantage of Java's robustness, multithreading, and

security features. JDBC is important to allow database access from a Java middle tier.

5.2 Applets & Security

One of the most highly important characteristics that is required of this product is its security. This is especially important because of the distributed nature of KCT *Info Agent-1*. Without an assurance of security, one certainly wouldn't download code from a site on the Internet and let it run on his computer. Our software was designed with security in mind, it provides several layers of security controls that protect against any malicious code, and allow users to comfortably run our programs as applets.

At the lowest level, security goes hand-in-hand with robustness. Our programs do not forge pointers to memory, or overflow arrays or read memory outside of the bounds of an array. The second line of safety is that the byte-code verification process that the Java interpreter performs any time it loads the code. These verification steps ensure that the code is well-formed and does not contain illegal byte- codes. Another layer security protection is commonly referred to as the ***"sandbox model".***

5.3 Software Agents

5.3.1 Introduction:

Since the beginning of early history people have been fascinated with the idea of non-human agencies. Popular notions about androids, humanoids, robots,

cyborgs and science fiction creatures permeate our culture, forming an unconscious backdrop against which software agents are perceived.

Today, researchers are taking to new fields like distributed artificial intelligence, robotics, artificial life, distributed object computing, human-computer interaction, intelligent and adaptive interfaces, intelligent search and filtering, information retrieval, knowledge acquisition and a growing list of other fields. As "agents" of many varieties have proliferated, there has been an explosion in the use of the term without a corresponding consensus on what it means. However the concept of agenthood can be summed up the following definition

"An AGENT is a computational entity which:-

- acts on behalf of the other entities in an ***autonomous*** fashion
- performs its action with some level of ***proactivity*** and ***reactivity***
- exhibits some level of the key attributes of learning, co-operation and mobility"

SOFTWARE AGENTS are software systems that loosely conform to above definition and can basically be described as inhabiting computers and networks, assisting users with computer based tasks.

Researchers and software companies have set high hopes on these so-called software agents, which "know" user's interest and can act autonomously on their interests. Instead of exercising complete control, people will be engaged in a co-operative process in which both human and computer agents initiate communication, monitor events and perform tasks to meet user's goals.

There are several kinds of agents based on their application or on the way they are implemented. Some basic categories of agents are collaborative agents,

interface agents, mobile agents, information/internet agents, reactive agents, hybrid agents, smart agents etc.

In the following section we give an overview of the intelligent and the mobile agent, whose characteristics we are incorporating into our project.

5.3.2 Intelligent Agent:

In recent years "intelligent agents" have moved from being a somewhat obscure research topic to being an important element in the product strategies of the world's largest technology companies. Intelligent agents are agents that perform the operations specified or given to them with a certain level of intelligence.

The intelligent agent accepts the user's statement of goals and carries out the tasks delegated to it. At a minimum, there can be some statement of preference performed by the agent. At a higher level of intelligence there is a user model, reasoning and more. Further out on the intelligence scale are systems that can "learn" and "adapt" to their environment, both in terms of the user's objective, and in terms of the resources available to the agent.

5.3.3 Mobile Agent:

Today, it seems the need of the hour is a new breed of computer software, which would help in making available new products to be placed in the hands of consumers. This computer software called communicating application will adeptly use public networks to which they give access to find and interact with people and interact with people and information on the consumer's behalf. This technology may also be called "Push Technology".

Today's networks pose a barrier to the development of communicating applications. This barrier stems from the need for such applications to physically distribute themselves these applications run not only on the computer dedicated to themselves but also on the computers that users share, the servers. Hence arises a new communication paradigm, which provides the organizing principle for a public network and a new communication software technology.

A mobile agent can be viewed as a piece of program code, or as a procedure along with its parameters which can ship itself from one computer to another computer in the network, and thus can be executed locally on the new machine. If required it can abruptly halt the execution on the current computer and ship itself to another computer in the network and continue its execution. These procedure calls are "local" rather than "remote". The procedure and its state is called a mobile agent to emphasize that they represent the sending computer even while being present in the receiving computer.

5.4 Graphical User Interface

This Application is intended to be used by users from all walks of life. Some may be computer illiterate. So, this should have a very good user interface. The user should feel comfortable while using this product. The web pages should be interactive and make the product user friendly.

The tasks that need to be carried out in order to make the pages lively include the following:

- Creating a links from a page currently open to other pages of user's interest
- Associating applications to open image, sound, video and other files
- Importing an existing web and automatically updating its links

- Inserting advanced objects such as ActiveX Controls, Java Applets, Plug-ins and Power Point animations.
- Creating text hyperlinks to pages and files in the web or anywhere in WWW.
- Creating clickable images and arranging for either the browser or the Web Server to implement the jump to the target file.
- Creating Frame sets displaying more than one page during display.
- Inserting forms to collect information from users.
- Including WebBots.
- Accepting inputs from users.

5.5 Fuzzy Logic

5.5.1 What is fuzzy logic?

Fuzzy Logic is a superset of conventional (Boolean) logic that has been extended to handle the concept of partial truth-truth values between "completely true" and "completely false". Dr. Lotfi Zadeh of UC/Berkely introduced it in the 1960's.

5.5.2 Fuzzy subsets:

Just as there is a strong relationship between Boolean logic and the concept of a subset, there is a similar strong relationship between fuzzy logic and fuzzy subset theory. In classical set theory, a subset U of a set S can be defined as a mapping from the elements of S to the elements of the set {0, 1},

U: S → {0,1}

This mapping may be represented as a set of ordered pairs, with exactly one ordered pair present for each element of S. The first element of the ordered pair is an element of the set S, and the second element is an element of the set {0, 1}. The

variables. Most tools for working with fuzzy expert systems allow more than one conclusion per rule. The set of rules in a fuzzy expert system is known as the rule base or knowledge base.

The general inference process proceeds in three or four steps.

1) FUZZIFICATION
2) INFERENCE
3) COMPOSITION
4) DEFUZZIFICATION

5.5.4 Where are Fuzzy Expert Systems used?

To date, fuzzy expert systems are the most common use of fuzzy logic. They are used in several wide-ranging fields, including:

- Linear and Nonlinear Control
- Pattern Recognition
- Financial Systems
- Operation Research
- Data Analysis

DESIGN OVERVIEW

6. DESIGN OVERVIEW

6.1 Database Design

The Database can be maintained by any software (preferably RDBMS) by the host. It should have the following tables:

Mainad:

This table stores the username, address, password and date on which the user became a member. The primary key used is the username.

Maind:

This table stores the all information of the user. It has details of users' age, education, martial status, religion, caste, salary, height, etc., The username here is the foreign key for the username of Mainad table given above.

Maine:

This table has details of what the user expects of his prosepective. Data stored here includes, age, deviation in age, height, deviation in height, religion, caste etc., The username is the foreign key of the username of Mainad table.

A Detail description of these tables is given in the E-R (Entity – Relationship) diagram given in fig 6.1.

6.2 Fuzzy System Design

6.2.1 Model:

Fig 6.2 gives an overview of the fuzzy model used. The model accepts a set of inputs (age, income and height) as its knowledge about each person. The model

itself executes a series of fuzzy propositions or rules stored in application code – the vocabulary of fuzzy sets and formally defined variables. The model executes these propositions in a way that simulates parallel processing (all the rules contribute to the final value of the solution variable). This means that the value of the solution variable (Rank) is not available for use until all the rules have been executed.

6.2.2 Variables:

At present, we are dealing with three variable viz., age, salary and height. The membership functions used are PERFECT, VERY NEAR and NEAR based on the closeness to the expectations. These are represented in Figures 6.4 (a), (b) & (c).

The output variable is rank which have five possible membership functions so that there can be a wider range for the ranking of different prospectives. The ranking is graded as FIRST, SECOND, THRD, FOURTH & FIFTH.

6.2.3 Knowledge Base:

There are 26 rules governing the output of the system. All the rules are given below:

1. If (AGE is PERFECT) and (INCOME is PERFECT) and (HEIGHT is PERFECT) then (RANK is FIRST)

2. If (AGE is PERFECT) and (INCOME is VERYNE) and (HEIGHT is PERFECT) then (RANK is FIRST)

3. If (AGE is PERFECT) and (INCOME is PERFECT) and (HEIGHT is VERYNE) then (RANK is FIRST)

4. If (AGE is VERYNE) and (INCOME is PERFECT) and (HEIGHT is PERFECT) then (RANK is FIRST)

5. If (AGE is PERFECT) and (INCOME is PERFECT) and (HEIGHT is NEAR) then (RANK is FIRST)

6. If (AGE is PERFECT) and (INCOME is NEAR) and (HEIGHT is PERFECT) then (RANK is SECOND)

7. If (AGE is NEAR) and (INCOME is PERFECT) and (HEIGHT is PERFECT) then (RANK is SECOND)

8. If (AGE is PERFECT) and (INCOME is VERYNE) and (HEIGHT is VERYNE) then (RANK is SECOND)

9. If (AGE is VERYNE) and (INCOME is PERFECT) and (HEIGHT is VERYNE) then (RANK is SECOND)

10. If (AGE is VERYNE) and (INCOME is VERYNE) and (HEIGHT is PERFECT) then (RANK is SECOND)

11. If (AGE is PERFECT) and (INCOME is VERYNE) and (HEIGHT is NEAR) then (RANK is THIRD)

12. If (AGE is NEAR) and (INCOME is VERYNE) and (HEIGHT is PERFECT) then (RANK is THIRD)

13. If (AGE is PERFECT) and (INCOME is NEAR) and (HEIGHT is VERYNE) then (RANK is THIRD)

14. If (AGE is PERFECT) and (INCOME is NEAR) and (HEIGHT is NEAR) then (RANK is THIRD)

15. If (AGE is NEAR) and (INCOME is PERFECT) and (HEIGHT is NEAR) then (RANK is THIRD)

16. If (AGE is NEAR) and (INCOME is NEAR) and (HEIGHT is PERFECT) then (RANK is FOURTH)

17. If (AGE is VERYNE) and (INCOME is NEAR) and (HEIGHT is NEAR) then (RANK is FOURTH)

18. If (AGE is NEAR) and (INCOME is VERYNE) and (HEIGHT is NEAR) then (RANK is FOURTH)

19. If (AGE is NEAR) and (INCOME is NEAR) and (HEIGHT is VERYNE) then (RANK is FOURTH)

20. If (AGE is VERYNE) and (INCOME is VERYNE) and (HEIGHT is NEAR) then (RANK is FOURTH)

21. If (AGE is VERYNE) and (INCOME is NEAR) and (HEIGHT is VERYNE) then (RANK is FIFTH)

22. If (AGE is NEAR) and (INCOME is VERYNE) and (HEIGHT is VERYNE) then (RANK is FIFTH)

23. If (AGE is NEAR) and (INCOME is NEAR) and (HEIGHT is VERYNE) then (RANK is FIFTH)

24. If (AGE is NEAR) and (INCOME is VERYNE) and (HEIGHT is NEAR) then (RANK is FIFTH)

25. If (AGE is VERYNE) and (INCOME is NEAR) and (HEIGHT is NEAR) then (RANK is FIFTH)

26. If (AGE is NEAR) and (INCOME is NEAR) and (HEIGHT is NEAR) then (RANK is FIFTH)

6.2.4 ***Rule Examples:***

Rules 1, 6 & 26 are discussed here as examples of different rules. Based on the membership values of these variables various rules are fired. Fig 6.3 (a),(b),(c) give the firing of Rules 1, 6, 26.respectively.

RULE-1:

IF(AGE IS PERFECT) AND (INCOME IS PERFECT) AND (HEIGHT IS PERFECT)THEN (RANK IS FIRST)

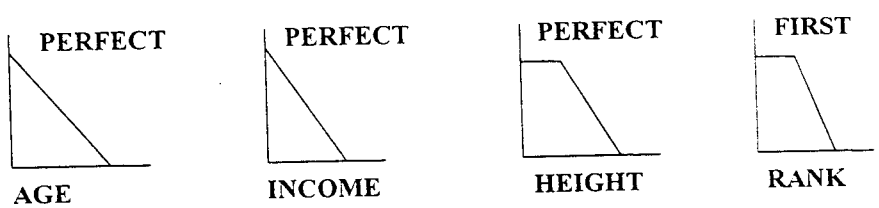


FIGURE –6.3(a)

RULE-6:

IF (AGE IS PERFECT) AND (INCOME IS NEAR) AND (HEIGHT IS PERFECT) THEN(RANK IS SECOND)

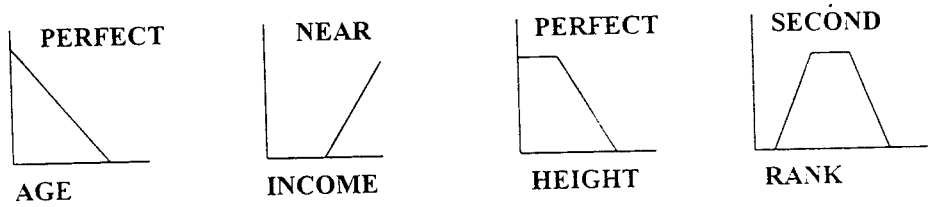


FIGURE-6.3(b)

RULE-26:

IF(AGE IS NEAR) AND (INCOMR IS NEAR) AND (HEIGHT IS NEAR) THEN (RANK IS FIFTH)

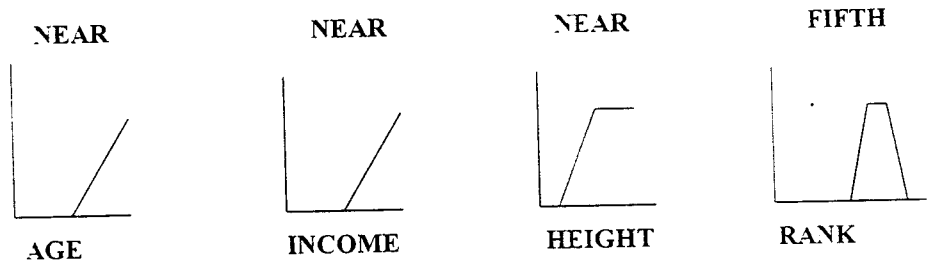


FIGURE-6.3(c)

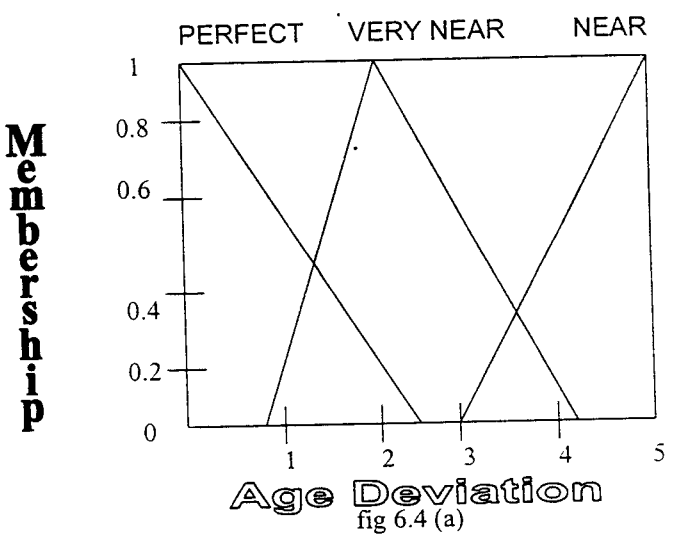


fig 6.4 (a)

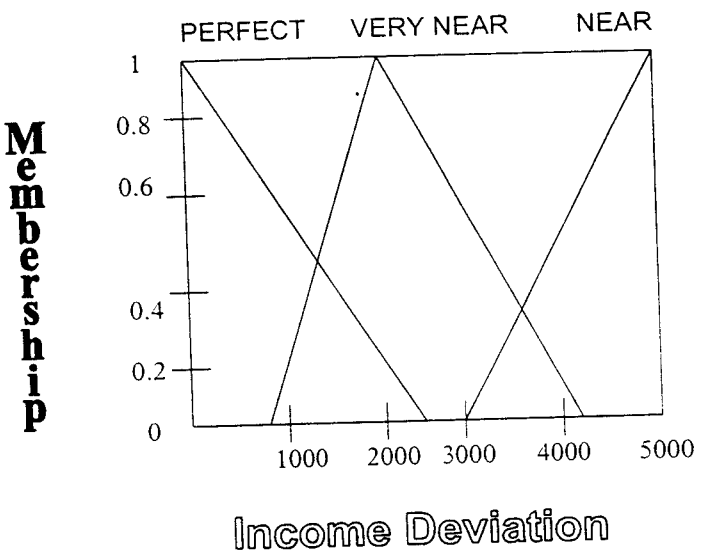


fig 6.4 (b)

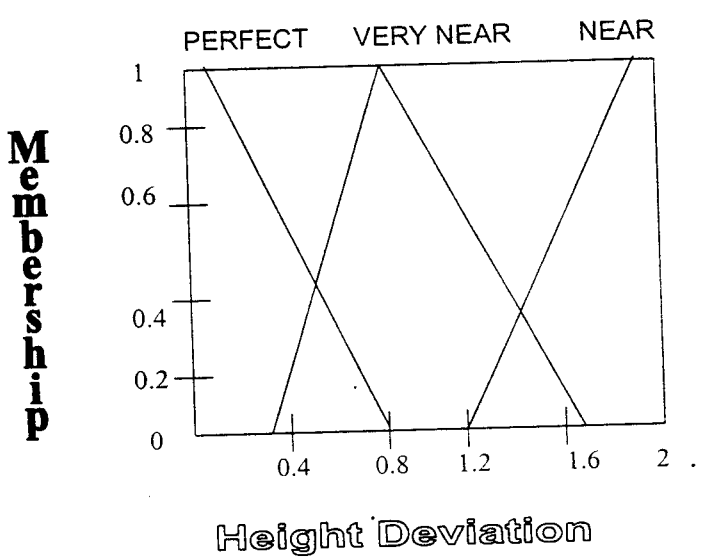


fig 6.4 (C)

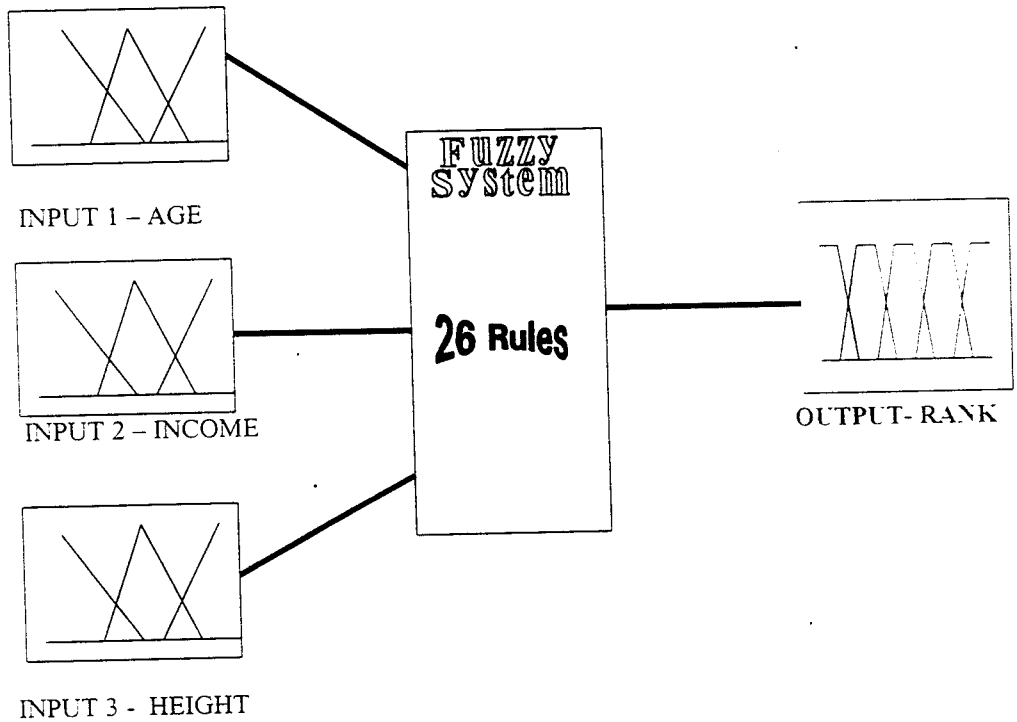

INPUT 1 – AGE
INPUT 2 – INCOME
INPUT 3 - HEIGHT
Fuzzy System
26 Rules
OUTPUT- RANK

Fig 6.2

IMPLEMENTATION DETAILS

7. PROJECT IMPLEMENTATION

In this section we give a brief description of how the technical concepts discussed in technical overview section were implemented.

7.1 JDBC

The JDBC allows the product to be extendible in many aspects. One important feature is that in many practical situations the database available at the web site of host organization does not follow the same format as of the another organization. The JDBC was designed with this in mind. We choose a thee-tier model for JDBC Driver. Fig 7.1 gives a detail representation of the three-tier model that we are using. It suits the requirements of connecting with any database of the host. In this three-tier model, that we are using commands are sent from a browser (user) to a "middle tier" 100% Java driver on the host server. This Driver then sends the SQL statements to the database. The database processes the SQL statements and sends the results back to the 100% Java Driver, which then sends them to the user. This three-tier architecture can provide performance advantages. Also, the host can change the database or other features by just changing the driver involved. Utmost care was taken to connect with the distributed databases spread across the net. We have assumed that any person / organization who wants to use this software will buy any 100% Java Driver and Install it to his Web server.

7.2 Implementation of Mobility of KCT Info Agent -1

Initially we had decided to create the *KCT Info Agent-1* so that it searches for the user's preference in the database maintained by the host organization. Additionally for increasing the user's options, we decided to increase the user's

range by performing the search operation on several databases (that might be spread across the net).

Once the user has input the relevant data and has specified his preference the mobile agent i.e. *KCT Info Agent-1* searches the database present in the server in which it resides. After performing the search operation, it travels over the network and searches other similar databases available on other servers. Once the search is performed the mobile agent returns to the local machine displaying all the results of the search operation. However the agent can search the remote database only if the owners of the server give them the access rights.

The word 'Mobile Agent' actually refers to a piece of program code which along with its input and output parameters can be executed on a remote machine on behalf of the user. Here our agent moves from one server to another searching for a prospective based on the user's expectations. The agent searches these databases and presents the user with a list of other users who have satisfied the current users preferences.

7.3 Agent Security

At the end user's terminal it's the applet that receives the data and perform search operations. Applets loaded over the network are usually considered by the browsers as untrusted code. For this reason, web browsers carefully restrict what an applet is allowed to do. Hence, we designed our applets to run without any security violations in the browser. Hence, the applets we build:

- Didn't try to run any native code.
- Were designed in 100% Java.
- Did not access any local files
- Did not try to manipulate any security identities.

7.4 Fuzzy Model

The Fuzzy model was first designed, simulated and tested using MATLAB. All the parameters were first tested and then later were implemented in Java. The four processes involved in the fuzzy expert system namely fuzzification, inference, compostion, defuzzification were done in Java. The membership functions were designed easily as the complexity involved in this system was very less. MIN-MAX method was used. For Defuzzification Centroid method was used to convert the fuzzy labels back to crisp sets.

The Knowledge base associated with this system was already discussed in the previous section, design overview.

7.5 Graphical User Interface (GUI)

The graphical user interface that we are using was designed with the following motives:

- Front end for users
- Interactive GUI
- Comfortable Environment
- Input Validation

Front End For Users

Front Page'97 was used to create the web pages. The pages were created using Front Page Editor. The pages containing forms to accept input from the user was created using Form field components. The hyperlinks are created to various pages for the user to easily browse through the pages.

Interactive GUI

The pages contain buttons and images (related to the topic) as hyperlinks which when clicked in the browser goes to the respective page. The form field components like Menus, single line text box, scroll text box and push button are included for improving the user's interactiveness.

Comfortable Environment

The pages are made more attractive and impressive to the user by including the audio and video clippings. The audio clippings were inserted as background sound. The video clippings were inserted using Active X controls.

Input Validation

The forms were created in the Front Page enabling the validation of input for every field. Error message occurs when the user enters the data incorrectly. Since the input is validated before being submitted at the host end, the network traffic is reduced to a greater extent.

REQUIREMENTS TO RUN KCT INFO AGENT-1

8. REQUIREMENTS FOR THE PRODUCT

We can view the basic requirements to run the product from the two ends i.e. the backend which is the host and the frontend which is the client.

8.1 Requirements Of The Organization

The basic requirement for the organization which is going to host this product are the following :

a) *Internet Server*

At the host organization a server connected to internet is required because this is the basic requirement for any organization to have a web site.

b) *100% Java Driver*

At the host organization a 100% java driver is necessary. This facilitates communication between the agent and the server without any additional requirements at the client side.

c) *Database*

A database is required to store information about the users. This database can be of any kind.

8.2 Requirements at the Client End

At the client end the basic and only requirement is Java 1.1 Enabled browser. Only the Java 1.1 browser can support the JDBC bridge and other advanced features of Java.

ADVANTAGES OF KCT INFO AGENT-1

9. ADVANTAGES OF KCT INFO AGENT-1

The major advantage of the *Kct Info Agent-1* is its availability. Anyone who can explore the cyberspace can use this product. The next major advantage that the users of *Kct Info Agent-1* enjoy is that they pay nothing but can use every thing present. The product will be freely available on web and can be accessed by anyone at any time and anywhere.

The users need not browse different newspapers, need not log into many web sites, need not search everywhere to find his prospective and need not pay for his advertisements. All he needs to do his is to log into any site that hosts our Agent and give his details and expectations to the agent.A list of prospectives is provided to him within no time.

A very good Graphical User Interface, Security and Agent based search on different web sites are other advantages.

DRAWBACKS OF THE PRODUCT

10. DRAWBACKS OF THE KCT INFO AGENT

Given below is a brief account of the various flaws present in our product of which we are aware of:-

a) The major weakness of our product is that the organization . which is hosting the agent, must meet all the requirements to run the product . This includes a100% Java driver, mentioned in JDBC specification of this document to support the three-tier model implemented. This 100% Java browser happens to be a released by some other software manufacturers. Thus, although the agent can travel to any web server , it can be executed only in those servers which have a100% Java JDBC Driver.

b) The second drawback of our product is that the number of users who can use this product is dependent on the database size of the host organization. If there is no memory space in the database the user will not be able to store his details and hence he will not be able to use the agent.

c) A minor drawback in the product is the non-inclusion of images of users. This was done purposefully considering the sociological attitude of the users.

d) As a security measure, the users are given a password the keep their data safely. But, in case they forget their password, there are no means by which they can get back their password. They have to start a new account by logging in as a new user.

SCOPE FOR EXPANSION & UPGRADATION

11. EXPANSION AND UPGRADTION

Periodic upgradation of the product will be done taking into consideration the suggestions and complaints of the users. If the user has any doubts or is not satisfied with the performance of the product , he would be requested to make it known at the ~~KCT~~ web site, Other features that are planned to be implemented include:

- At present the software was designed for exclusive use by South Indian Dailies in India and abroad. In future we hope to extend it to be used by all users all over the world.
- An option for the user to include his/her image.
- Horoscope matching if the user specifies it.
- Including some general information about users like hobbies, interests etc.,

SOURCE CODE

12. SOURCE CODE

The Source Code for Some of the important implementations of the project are given below:

- ***For the Implementation JDBC:*** **[Java Code]**

```
public class searchbothfuzzy extends Applet
{
        TextField na =new TextField(5);
        List id;
        String url="jdbc:odbc:Testsql";
        String url1="jdbc:odbc:arun1";
        Connection con,c,con1,c1;
        Button Bsearch = new Button("SEARCH");
        ResultSet rs,r,r1,re3,re4;
        String msg="";
        TextArea displ=new TextArea(7,35);

        public void init()
        {
            id=new List(4,false);
            try
            {
                Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
                                        //Establishes the driver to
                                    Connect

            }
            catch(ClassNotFoundException e)
            {
            }
```

- ***For Making JDBCConnections to Different Servers Registered with the Agent: [Java Code]***

```
// CONNECTING TO KAUMUDI(TEST SQL-NTS2)
            con = DriverManager.getConnection(url,"arun","arun");
```

```
            DatabaseMetaData dma = con.getMetaData ();
//CONNECTING TO KCT(ARUN1 NTS1)
            con1 = DriverManager.getConnection(url1,"arun1","arun1");
            DatabaseMetaData dma1 = con1.getMetaData ();
```

- *Code for Agent Search:*

```
//IF BUTTON SEARCH IS CLICKED ON
        if(evt.target==Bsearch)
        {
            try
            {
            //OBTAIN THE NAME OF USER
                String temp=na.getText();
            //OBTAIN HIS EXPECTATIONS
                PreparedStatement stmt=con.prepareStatement("SELECT
age,dage,sex,income,din,ht,dht FROM maine where name=?");
                stmt.setString(1,temp);
                rs=stmt.executeQuery();


                while(rs.next())
                {
                    age=rs.getInt("age");
                    dage=rs.getInt("dage");
                    sex=rs.getString("sex");
                    incom=rs.getInt("income");
                    din=rs.getInt("din");
                    ht=rs.getInt("ht");
                    dht=rs.getInt("dht");
```

```
                    }
                    uage=age+dage;
                    lage=age-dage;
                    lin=incom-din;
                    uin=incom+din;
                    lht=ht-dht;
                    uht=ht+dht;
                //SEARCH FOR PROSPECTIVE IN NTS2
                    PreparedStatement stmt1=con.prepareStatement("SELECT
name FROM maind where sex=? AND age < ? AND age > ? AND income < ?
AND income > ? AND ht < ? AND ht > ?");
                    stmt1.setString(1,sex);
                    stmt1.setInt(2,uage);
                    stmt1.setInt(3,lage);
                    stmt1.setInt(4,uin);
                    stmt1.setInt(5,lin);
                    stmt1.setInt(6,uht);
                    stmt1.setInt(7,lht);

                    r=stmt1.executeQuery();
                    while(r.next())
                    {
                        ag=r.getString("name");
                        id.addItem(ag);
                     }
                    r.close();
                    stmt1.close();
                    rs.close();
                    stmt.close();
```

```
con.close();

  //SEARCH FOR PROSPECTIVE IN NTS1
 PreparedStatement stmt3=con1.prepareStatement("SELECT name,age,income,
FROM maind where sex=? AND age < ? AND age > ? AND income < ? AND
income > ? ");
//CODE ASSUMES ATLEAST THE SUPPORT 0F ANSI SQL 2.0
                stmt3.setString(1,sex);
                stmt3.setInt(2,uage);
                stmt3.setInt(3,lage);
                stmt3.setInt(4,uin);
                stmt3.setInt(5,lin);

                r1=stmt3.executeQuery();

                while(r1.next())
                {
                    ag=r1.getString("name");
                    id.addItem(ag);
                 }
                 r1.close();

                 r.close();
                 stmt1.close();
                 rs.close();
                 stmt.close();

                 stmt3.close();
```

```
                    con1.close();
                }
                catch(SQLException e)
                {
                }

                return true;  //EVENT HANDLED SUCCESSFULLY
        }
        if (evt.target instanceof List)
            {
            try
            {
            // CONNECTING TO BOTH DATABASES
               c = DriverManager.getConnection(url,"arun","arun");
               DatabaseMetaData dma = c.getMetaData ();
               c1 = DriverManager.getConnection(url1,"arun1","arun1");
               DatabaseMetaData dma3 = c1.getMetaData ();

               idx=id.getSelectedIndexes();
               for(i=0;i<idx.length;i++)
                   ni +=id.getItem(idx[i]);
             // OBTAINING DETAILS FROM KAUMUDI(NTS2)
               PreparedStatement stmt4=c.prepareStatement("SELECT
name,poboxno FROM mainad where name=?");
               stmt4.setString(1,ni);
               re3=stmt4.executeQuery();


               while(re3.next())
```

```
{
    msg +=" SITE       : KAUMUDI";
    msg +="\n NAME     : ";
    name1=re3.getString("name");
     msg +=name1;
     msg +="\nPO BOX NO :";
     po=re3.getInt("poboxno");
     msg +=Integer.toString(po);

 }

 //OBTAINING DETAILS FROM KCT(NTS1)
    PreparedStatement stmt5=c1.prepareStatement("SELECT
name,poboxno FROM mainad where name=?");
    stmt5.setString(1,ni);

    re4=stmt5.executeQuery();


    while(re4.next())
    {
        msg +=" SITE      : KCT";
        msg +="\n NAME      : ";
        name1=re4.getString("name");
         msg +=name1;
         msg +="\nPO BOX NO:";
         po=re4.getInt("poboxno");
         msg +=Integer.toString(po);
```

```
            }


                re3.close();
                stmt4.close();
                re4.close();
                stmt5.close();
                c.close();
                c1.close();
            }
            catch(SQLException e)
            {
                    setBackground(Color.green);
             }
             catch(NumberFormatException  ex)
              {
              }
              disp(msg);
              return true;
              }
        else
        {
            return super.handleEvent(evt);
        }

    }
```

- ***Matlab Source Code for Simulation:***

```
clear a
a= newfis('CLASSIFIED');
a=addvar(a,'input','AGE',[0 5]);
a=addmf(a,'input',1,'EXACT','trimf',[0 0 2]);
a=addmf(a,'input',1,'VERYNE','trimf',[0.5 2 4.1]);
a=addmf(a,'input',1,'NEAR','trimf',[3 5 5]);
whitebg([0.1 0.3 0]);
%plotmf(a,'input',1);
%figure;

%whitebg([0.2 0.3 0]);
a=addvar(a,'input','INCOME',[0 5000]);
a=addmf(a,'input',2,'EXACT','trimf',[0 0 2000]);
a=addmf(a,'input',2,'VERYNE','trimf',[500 2000 4100]);
a=addmf(a,'input',2,'NEAR','trimf',[3000 5000 5000]);

%plotmf(a,'input',2);
%figure;
a=addvar(a,'input','HEIGHT',[0 2]);
a=addmf(a,'input',3,'EXACT','trapmf',[0 0 .2 .85]);
a=addmf(a,'input',3,'VERYNE','trimf',[0.3 1 1.7]);
a=addmf(a,'input',3,'NEAR','trapmf',[1.25 1.85 2 2]);

%whitebg([0.2 0.4 0.11]);
%plotmf(a,'input',3);
%figure;
a=addvar(a,'output','RANK',[1 100]);
```

```
a=addmf(a,'output',1,'FIRST','TRAPMF',[1 1 12 20]);
a=addmf(a,'output',1,'SECOND','TRAPMF',[12 21 32 41]);
a=addmf(a,'output',1,'THIRD','TRAPMF',[32 39 50 59]);
a=addmf(a,'output',1,'FOURTH','TRAPMF',[52 60 72 81]);
a=addmf(a,'output',1,'FIFTH','TRAPMF',[72 81 100 100]);
whitebg([0.4 0.1 0]);
%plotmf(a,'output',1);
%figure;
ruleList=[
        1 1 1 1 1 1;
        1 2 1 1 2 1;
        1 1 2 1 3 1;
        2 1 1 1 4 1;
        1 1 3 1 5 1;
 % first
        1 3 1 2 6 1;
        3 1 1 2 7 1;
        1 2 2 2 8 1;
        2 1 2 2 9 1;
        2 2 1 2 10 1;
 % second
        1 2 3 3 11 1;
        3 2 1 3 12 1;
        1 3 2 3 13 1;
        1 3 3 3 14 1;
        3 1 3 3 15 1;
 % third
        3 3 1 4 16 1;
        2 3 3 4 17 1;
        3 2 3 4 18 1;
       3 3 2 4 19 1;
        2 2 3 4 20 1;
 %  fourth
        2 3 2 5 21 1;
        3 2 2 5 22 1;
        3 3 2 5 23 1;
        3 2 3 5 24 1;
        2 3 3 5 25 1;
         3 3 3 5 26 1
```

```
% fifth
 ];

a=addrule(a,ruleList);
RULE_BASE=showrule(a)
%hold on;
%writefis(a,'classified')
%plotfis(a);
OUTPUT_RANK=evalfis([1 1 1 ; 4.9 200 0.2],a)
%gensurf(a)
% open the system in fuzzy editor of Matlab.
fuzzy(a)
```

- ***HTML Source Code for a Web page to get User's Expectations :***

```
<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML//EN">
<html>

<head>
<meta http-equiv="Content-Type"
content="text/html; charset=iso-8859-1">
<meta name="FORMATTER" content="Microsoft FrontPage 2.0">
<meta name="GENERATOR" content="Microsoft FrontPage 2.0">
<title>expect</title>
</head>

<body background="Gray_MarbleB0.gif">

<h1 align="center"><font color="#FF0080"><em><img
src="Mail23.gif" width="41" height="32">YOUR
EXPECTATIONS</em></font></h1>
```

```
<p align="center"><font color="#FF0080"><em><img
src="images/Burst78.gif" width="573" height="18"></em></font></p>

<form action="expect.idc" method="POST" name="FrontPage_Form1"
onsubmit="return FrontPage_Form1_Validator(this)">
  <p align="center"><font
color="#FF0080"><em><strong>USERNAME</strong></em>
  <!--webbot bot="Validation" s-display-name="USERNAME"
  s-data-type="String" b-value-required="TRUE"
  i-minimum-length="2" i-maximum-length="15" --><input
  type="text" size="20" maxlength="15" name="name"
  language="JavaScript" onchange="FrontPage_Form1.age.focus()">
  </font></p>
  <div align="center"><center><table border="0" cellpadding="8"
  cellspacing="8" width="100%" bordercolor="#FFFFFF"
  bordercolordark="#FFFFFF" bordercolorlight="#FFFFFF">
    <tr>
      <td><font color="#FF0080"><em><strong>AGE <!--webbot
      bot="Validation" s-display-name="age"
      s-data-type="Integer" s-number-separators="x"
      b-value-required="TRUE" i-minimum-length="2"
      i-maximum-length="3" --><input type="text" size="3"
      maxlength="3" name="age"> </strong></em></font></td>
      <td><p align="left"><font color="#FF0080"><em><strong>AGE
      DEVIATION</strong></em> <!--webbot bot="Validation"
      s-display-name="AGE DEVIATION"
      b-value-required="TRUE" --><select name="dage"
      size="1">
```

```
<option> 1 </option>
<option> 2 </option>
<option> 3 </option>
<option> 4 </option>
<option> 5 </option>
<option> 6 </option>
</select> </font></p>
</td>
<td><p align="left"><font color="#FF0080"><em><strong>SEX
<!--webbot bot="Validation" s-display-name="SEX"
b-value-required="TRUE" --><select name="sex"
size="1">
<option value="M"> Male </option>
<option value="F"> Female </option>
</select> </strong></em></font></ p>
</td>
</tr>
<tr>
<td colspan="2"><p align="left"><font
color="#FF0080"><em><strong>QUALIFICATION
<!--webbot bot="Validation"
s-display-name="QUALIFICATION"
b-value-required="TRUE" --><select name="qual"
size="1">
<option> UG </option>
<option> PG </option>
<option> PHD </option>
</select> </strong></em></font></p>
</td>
```

```
<td><p align="left"><font color="#FF0080"><em><strong>PROFESSION
<!--webbot bot="Validation"
s-display-name="PROFESSION" b-value-required="TRUE"
--><select name="prof" size="1">
   <option> DOCTOR </option>
   <option> ENGINEER </option>
   <option> AUDITOR </option>
   <option> BUSINESSMAN </option>
   <option> LAWYER </option>
   <option> OTHERS </option>
 </select> </strong></em></font></p>
 </td>
</tr>
<tr>
   <td valign="top" colspan="2"><p align="left"><font
   color="#FF0080"><em><strong>INCOME <!--webbot
   bot="Validation" s-display-name="INCOME"
   s-data-type="Integer" s-number-separators=","
   b-value-required="TRUE" i-minimum-length="3"
   i-maximum-length="15" --><input type="text" size="10"
   maxlength="15" name="income"> </strong></em></font>< p>
   </td>
   <td valign="top"><p align="left"><font
   color="#FF0080"><em><strong>INCOME VARIATION <!--webbot
   bot="Validation" s-display-name="INCOME VARIATION"
   b-value-required="TRUE" --><select name="din"
   size="1">
      <option> 1000 </option>
      <option> 2500 </option>
```

```
            <option> 5000 </option>
          </select> </strong></em></font></p>
          </td>
        </tr>
        <tr>
          <td colspan="2"><p align="left"><font
color="#FF0080"><em><strong>RELIGION
          <!--webbot bot="Validation"
          s-display-name="RELIGION" b-value-required="TRUE" --><select
          name="religion" size="1">
            <option> HINDU </option>
            <option> MUSLIM </option>
            <option> CHRISTIAN </option>
          </select> </strong></em></font></p>
          </td>
          <td width="60%"><p align="left"><font
 color="#FF0080"><em><strong>CASTE
          <!--webbot bot="Validation" s-display-name="CASTE"
          b-value-required="TRUE" --><select name="caste"
          size="1">
            <option> GOUNDER </option>
            <option> CHETTIAR </option>
            <option> IYER </option>
            <option> IYENGAR </option>
            <option> PILLAI </option>
            <option> NAIDU </option>
          </select> </strong></em></font></p>
          </td>
        </tr>
```

```
<tr>
  <td colspan="2"><font color="#FF0080"><em><strong>HEIGHT
  <!--webbot bot="Validation" s-display-name="HEIGHT"
  s-data-type="Integer" s-number-separators=","
  b-value-required="TRUE" i-minimum-length="2"
  i-maximum-length="5" --><input type="text" size="5"
  maxlength="5" name="ht"> </strong></em></font></td>
  <td><font color="#FF0080"><em><strong>HEIGHT
  VARIATION <!--webbot bot="Validation"
  s-display-name="HEIGHT VARIATION"
  b-value-required="TRUE" --><select name="dht"
  size="1">
    <option> 2 </option>
    <option> 3 </option>
    <option> 4 </option>
    <option> 5 </option>
    <option> 10 </option>
  </select> </strong></em></font></td>
</tr>
<tr>
  <td colspan="2"><p align="left"><font
color="#FF0080"><em><strong>MOTHER
  TONGUE <!--webbot bot="Validation"
  s-display-name="MUTHER TONGUE"
  b-value-required="TRUE" --><select name="lang"
  size="1">
    <option> TAMIL </option>
    <option> KANNADA </option>
    <option> TELGU </option>
```

```
            <option> ENGLISH </option>
            <option> MALAYALAM </option>
          </select> </strong></em></font></p>
          </td>
          <td><p align="left"><font
color="#FF0080"><em><strong>COMPLEXION
          <!--webbot bot="Validation"
          s-display-name="COMPLEXION" b-value-required="TRUE"
          --><select name="compl" size="1">
            <option> DARK < option>
            <option> BROWNISH </option>
            <option> FAIR < option>
          </select> </strong>< em></font></p>
          </td>
        </tr>
        <tr>
          <td colspan="2"><p align="left"><font
color="#FF0080"><em><strong>MARITAL
          STATUS <!--webbot bot="Validation"
          s-display-name="MARITAL STATUS"
          b-value-required="TRUE" --><select name="status"
          size="1">
            <option selected> SINGLE </option>
            <option> DIVORCED </option>
            <option> WIDOW </option>
            <option> WIDOWER </option>
          </select> </strong>< em></font></p>
          </td>
          <td><font color="#FF0080"><em><strong>RESIDENT OF
```

```
            <option> ENGLISH </option>
            <option> MALAYALAM </option>
          </select> </strong></em></font></p>
          </td>
          <td><p align="left"><font
color="#FF0080"><em><strong>COMPLEXION
          <!--webbot bot="Validation"
          s-display-name="COMPLEXION" b-value-required="TRUE"
          --><select name="compl" size="1">
            <option> DARK </option>
            <option> BROWNISH </option>
            <option> FAIR </option>
          </select> </strong></em></font></p>
          </td>
        </tr>
        <tr>
          <td colspan="2"><p align="left"><font
color="#FF0080"><em><strong>MARITAL
          STATUS <!--webbot bot="Validation"
          s-display-name="MARITAL STATUS"
          b-value-required="TRUE" --><select name="status"
          size="1">
            <option selected> SINGLE </option>
            <option> DIVORCED </option>
            <option> WIDOW </option>
            <option> WIDOWER </option>
          </select> </strong></em></font></p>
          </td>
          <td><font color="#FF0080"><em><strong>RESIDENT OF
```

CONCLUSION

13. CONCLUSION

Several project runs were made and the outputs were verified and validated with manual reports. All efforts to make the project highly reliable have been made.

This product was developed to simplify most of the tedious jobs, which the user should have performed otherwise. The core features of the product are a fuzzy based search and easy to use graphical user interface. Provisions were made so that it can search different database in different servers.

Provisions have been made for future enhancement, which will be made periodically so as to increase the performance of the product.

REFERENCES

AGENTS

1.Jeffery.M.Bradshaw. ***Software Agents.*** MIT Press 1997

2.Shoam, Yaov. ***Agent-Oriented Programming***

http://www.csli.stanford.edu/csli/9495reps/interface9495_shoam.html

3.White, J.E (1996). ***Mobile Agents - White Paper***

http://www.genmagic.com/agents/whitepaper/whitepaper.html

4.Shaw Green, Hurst.L, Somas.F et al, ***Software Agents: A review***

5.Stan Franklin and Art Greaser. *Is it an agent, or just a program?* In proceedings of the Third International workshop on Agent Theories, Architecture and Languages.

6.Chess.D, David L.Eviev: ***IBM Research Report: Itinerant Agents for Mobile Computing***

7.The agents home page at

http://www.agents.org

8.UMBC Agent Web

http://www.cs.umbc.edu/agents/introduction.html

9.MIT's Software Agent Group

http://www.media.mit.edu

JAVA

1.Gary Cornell and Cays Horstmam. ***Core Java 1.1*** Sun MicroSystems Press

2.David Flanagam. ***Java in a Nutshell.*** O'Reily publications 1997

3.Java developer's site

http://www.developer.com

http://www.gamelan.com