CHAPTER-4

Design & Implementation of Rule Based & Cased Based Expert System

Introduction

In the following sections, a technology used for the implementation, design, diagrams and logic of the rule-based, case-based and the hybrid / composite is included. Finally the chapter ends with conclusion.

4.2 Technology Used for Implementation

The technology used for the implementation is categorized in to three types,

i) Technology for the implementation of rule-based reasoning (RBR)
ii) Technology for the implementation of case-based reasoning (CBR)
iii) Technology for the implementation of hybrid / composite model

4.2.1 Rule Based Reasoning: Description, diagrams and logic

It is difficult to construct knowledge-bases and inference engines with conventional programming languages. Languages such as ALGOL are based on defining data and procedures for working with that data. Such languages have no basic structures for encoding logical relationships. PROLOG is advanced computer language for application of knowledge-based techniques. In turn knowledge-based techniques are methods of applying the computer to conditions or situations described by people in logical, structured statements. One important application of knowledge-based techniques lies in programming expert system for computer implementations **[56].** In 1972, Alain Colmerauer, a French researcher, and Robert Kowalski of Edinburgh University in Scotland created Prolog, short for "Programming in Logic"). In Prolog, it is easy to state relationships between objects.

For example:

father (X, Y) :- parent (X, Y), is male (X) grandfather

*(X, Y) :- father (X, **l**), parent **(l,** Y)*

For example, the fact that Joe is the father of Bill would be written as: The programmer then defines logical rules that apply to the facts. The first rule states that a person X is the father of a person Y if he is the parent of Y and is male. The second rule says that X is Y's grandfather if he is the father of a person Z who in turn is a parent of Y. When a Prolog program runs, it processes queries, or assertions whose truth is to be examined. Using a process called unification, the Prolog system searches for the facts or rules that apply to the query and then attempts to create a logical chain leading to proving the query is true. If the chain breaks (because no matching fact or rule can be found), the system "backtracks" by looking for another matching fact or rule from which to attempt another chain. Prolog became widely used in the late 1970s and 1980s, spurred on by the Japanese decision to use it for their massive fifth generation computer program. Although this attempt to build a new type of super logic computer ultimately failed, Prolog continues to be used in a number of areas of artificial intelligence research, particularly the construction of expert systems. In world of knowledge-based systems, logic has a special meaning and set of special implications **[56].**

4.3 Intelligent Flowcharts: What is VisiRule?

VisiRule is a tool for creating decision support software purely by drawing flowcharts. The end result is Flex or Prolog code which is automatically generated, compiled and ready to run, but this can also be copied and used in a separate program, VisiRule can be used by people with minimal programming skills. Once you are familiar with the tool, building an application is like creating a graphical machine. VisiRule also enhances productivity by considerably reducing the time it takes to produce a decision support system **[57].**

VisiRule is an intelligent flowcharting tool in two senses. Firstly, it is used to create knowledge-based systems and, secondly, it intelligently guides the construction process by constraining what you can do and can't do on the basis of the semantic content of the emerging program. This means for example, that you cannot inadvertently construct invalid links. As well as this real time semantic checking, VisiRule also checks the syntax of expressions as they are entered. VisiRule provides the automatic construction of menu dialogues from questions. These are populated by items inferred from expression boxes throughout the flowchart tree which have a path to the question.

VisiRule also offers:

- A wide variety of question types including single and multiple choice, numeric and integer entry, text and set entry
- A powerful expression handling logic
- Statement boxes for computable answers which are not decided by questioning the user
- Code boxes for procedural code and external functions
- Modularity allowing multiple charts to define one executable program

In particular we can build decision trees, classifiers and diagnostic systems of arbitrary complexity using these simple tools **[57].**

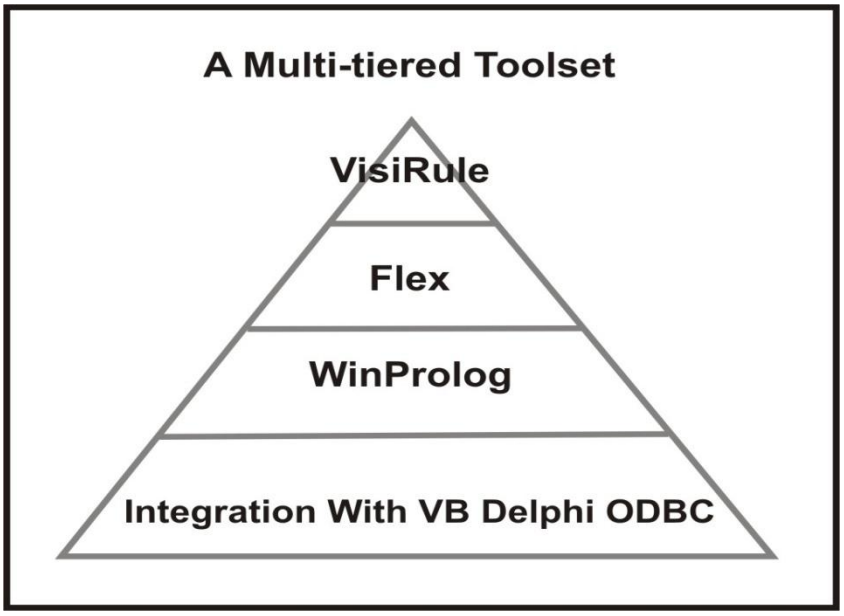

Figure 4.1: VisiRule architecture

VisiRule lets you generate code in flex which in turn gives you access to Prolog

i) Simple Chart

The simplest VisiRule charts consist of a start box, one or more question boxes, some expression boxes and some end boxes which are the conclusions drawn from the answers to the questions.

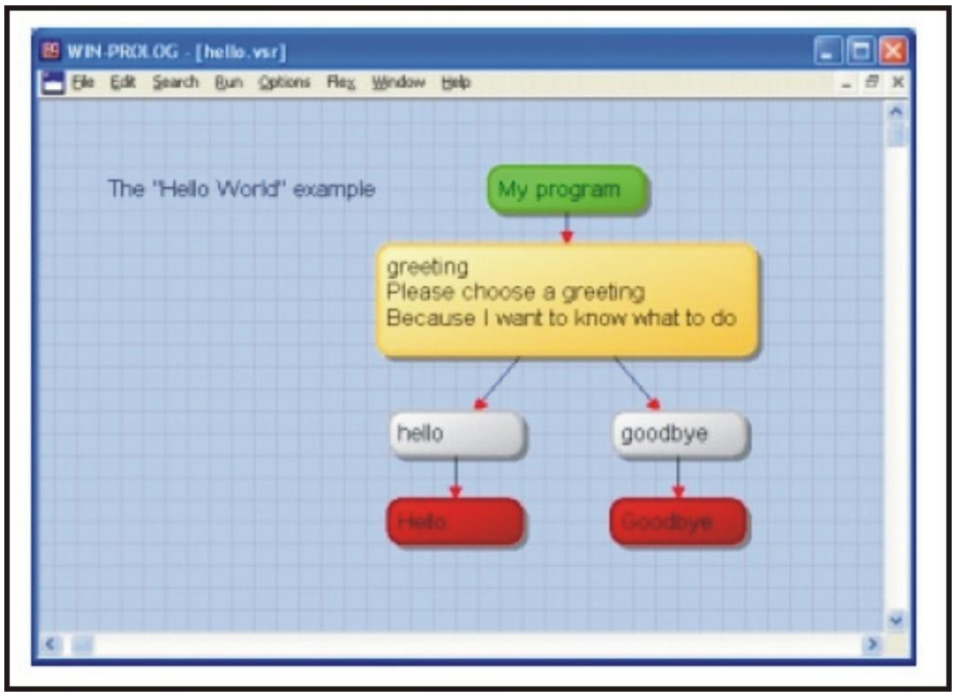

Figure 4.2: Sample Chart

4.3.1 Implementation of Rule Based Reasoning

The extensive developmental study of the modules was conceived and the VisiRule software was selected for the development purpose. In rule based part 10 modules are developed covering all aspects of transfer of property act in Indian legal domain using the rule-based reasoning concept. The modules consist of nearly 87 rules.

The modules which have been developed are a follows

1) Main Module
2) Individual Module
3) Authority Module
4) Document Module
5) Government Module
6) Guardian Module
7) Company Module
8) Free-Consent Module
9) Consideration Module
10) Karta Module

VisiRule implementation of the modules mentioned above is given below.

i) The **GREEEN** boxes indicate the **START,**

ii) The **YELLOW** boxes indicate the **QUESTION**

iii) The **RED** box indicates the **CONCLUSION** (i.e. either the property can be purchased or not) drawn after series of inferences.

iv) The **WHITE** boxes indicate **OPTIONS** to be selected by the questions.

v) The **BLUE** boxes indicate the **STATEMENT**

vi) The **BROWN** boxes indicate the **NEXT MODULE** for execution

Sample module: – Main Module and related Source code

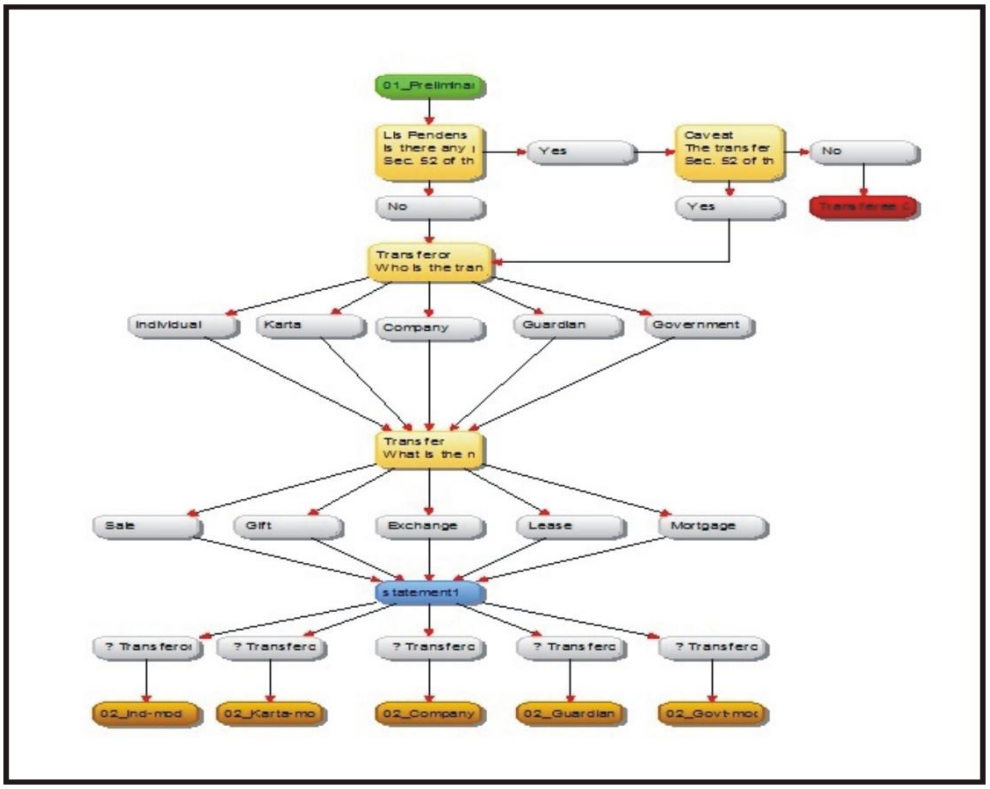

Figure 4.3: VisiRule Implementation Module-1

Table 4.1 Pseudo code of module 1

```
do ensure_loaded( system(vrlib) ) .

relation '01_Preliminary'( Conclusion ) if
  'q_Lis Pendens'( Conclusion ) .

relation 'q_Lis Pendens'( Conclusion ) if
  the answer to 'Lis Pendens' is _ and
  check( 'Lis Pendens', =, 'No' ) and
  q_Transferor( Conclusion ) .

relation 'q_Lis Pendens'( Conclusion ) if
  the answer to 'Lis Pendens' is _ and
  check( 'Lis Pendens', =, 'Yes' ) and
  q_Caveat( Conclusion ) .

relation q_Transferor( Conclusion ) if
  the answer to 'Transferor' is _ and
  check( 'Transferor', =, 'Individual' ) and
  q_Transfer( Conclusion ) .

relation q_Transferor( Conclusion ) if
  the answer to 'Transferor' is _ and
  check( 'Transferor', =, 'Karta' ) and
  q_Transfer( Conclusion ) .

relation q_Transferor( Conclusion ) if
  the answer to 'Transferor' is _ and
  check( 'Transferor', =, 'Company' ) and
  q_Transfer( Conclusion ) .

relation q_Transferor( Conclusion ) if
  the answer to 'Transferor' is _ and
  check( 'Transferor', =, 'Guardian' ) and
  q_Transfer( Conclusion ) .

relation q_Transferor( Conclusion ) if
  the answer to 'Transferor' is _ and
  check( 'Transferor', =, 'Government' ) and
  q_Transfer( Conclusion ) .

relation q_Transfer( Conclusion ) if
  the answer to 'Transfer' is _ and
```

Table 4.2 Pseudo code of module

```
 check( 'Transferor', =, 'Guardian' ) and

   '02_Guardian-mod'( Conclusion ) .

relation s_statement1( Conclusion ) if

   vrl_vrt_statement( $(statement1), c_statement1(X), X )  and

   check( 'Transferor', =, 'Government' ) and

   '02_Govt-mod'( Conclusion ) .

relation q_Caveat( Conclusion ) if

   the answer to 'Caveat' is _ and

   check( 'Caveat', =, 'Yes' ) and

   q_Transferor( Conclusion ) .

relation q_Caveat( Conclusion ) if

   the answer to 'Caveat' is _ and

   check( 'Caveat', =, 'No' ) and

   Conclusion = 'Transferee Quits'  .

group group1

   'Yes', 'No' .

  question 'Caveat'

   'The transfer will be subject to the outcome of the litigation. Do you still want to continue?' ;
choose one of group1

because 'Sec. 52 of the TP Act provides that if any suit or proceeding is pending in any court in
India in respect of title to any property, such property or any interest therein cannot be transferred
by any party to such suit or proceeding. Any such transfer shall be subject to the outcome of the
case.~M~J' .  group group2    'No', 'Yes' .
```

Table 4.3 Pseudo code of module

```
1)  question 'Lis Pendens'
'Is there any pending litigation in respect of the title to the property?' ;  choose one of group2

because 'Sec. 52 of the TP Act provides that if any suit or proceeding is pending in any court in India in respect of title to any property, such property or any interest therein cannot be transferred by any party to such suit or proceeding. Any such transfer shall be subject to the outcome of the case.~M~J' .

group group3

'Individual', 'Karta', 'Company', 'Guardian', 'Government' .


2)  question 'Transferor'
'Who is the transferor?' ;  choose one of group3

because 'The transferor may be an Individual, an HUF, a Company, a Partner in a Firm, a Guardian of a Minor, or a Government. Each type of transferor is subject to different sets of rules of law. Each one has to fulfil different sets of requirements.~M~J' .

group group4

'Sale', 'Gift', 'Exchange', 'Lease', 'Mortgage' .
```

Table 4.4 Pseudo code of module

```
1)  question 'Transfer'
'What is the mode of transfer?' ;  choose one of group4

because 'Sale, Exchange and Gift are complete transfers of property while Lease and Mortgage are partial transfers. Further, each mode of transfrer has its own requirements. Therefore, sets of rules applicable to different modes of transfer are different.~M~J~M~J' .
```

4.4 Case-Based Reasoning: Description, diagrams and logic

4.4.1 Description, diagram

For the development of the case-based reasoning the Java NetBeans and Microsoft Access are used. These two software's are very popular software's hence the more details are not included.

i) JAVA

Java is a programming language and computing platform first released by Sun Microsystems in 1995. It is the underlying technology that powers state-of-the-art programs including utilities, games, and business applications. Java runs on more than 850 million personal computers worldwide, and on billions of devices worldwide, including mobile and TV devices **[58].**

ii) Microsoft Office Access,

The Microsoft Access is used to store the cases. Previously known as Microsoft Access, is a relational database management system from Microsoft that combines the relational Microsoft Jet Database Engine with a graphical user interface and software-development tools. It is a member of the Microsoft Office suite of applications, included in the Professional and higher editions or sold separately **[59].**

4.4.2 Implementation of Case Based Reasoning

Case-based reasoning component of this research work has the following THREE modules.

i. Searching cases with Keyword and related variations
ii. intelligent learning part
iii. Web-Search of related cases:

Module 1: Searching cases with Keyword and related variations

This module discusses the implementation of the case-based reasoning part. This module is divided in to two parts

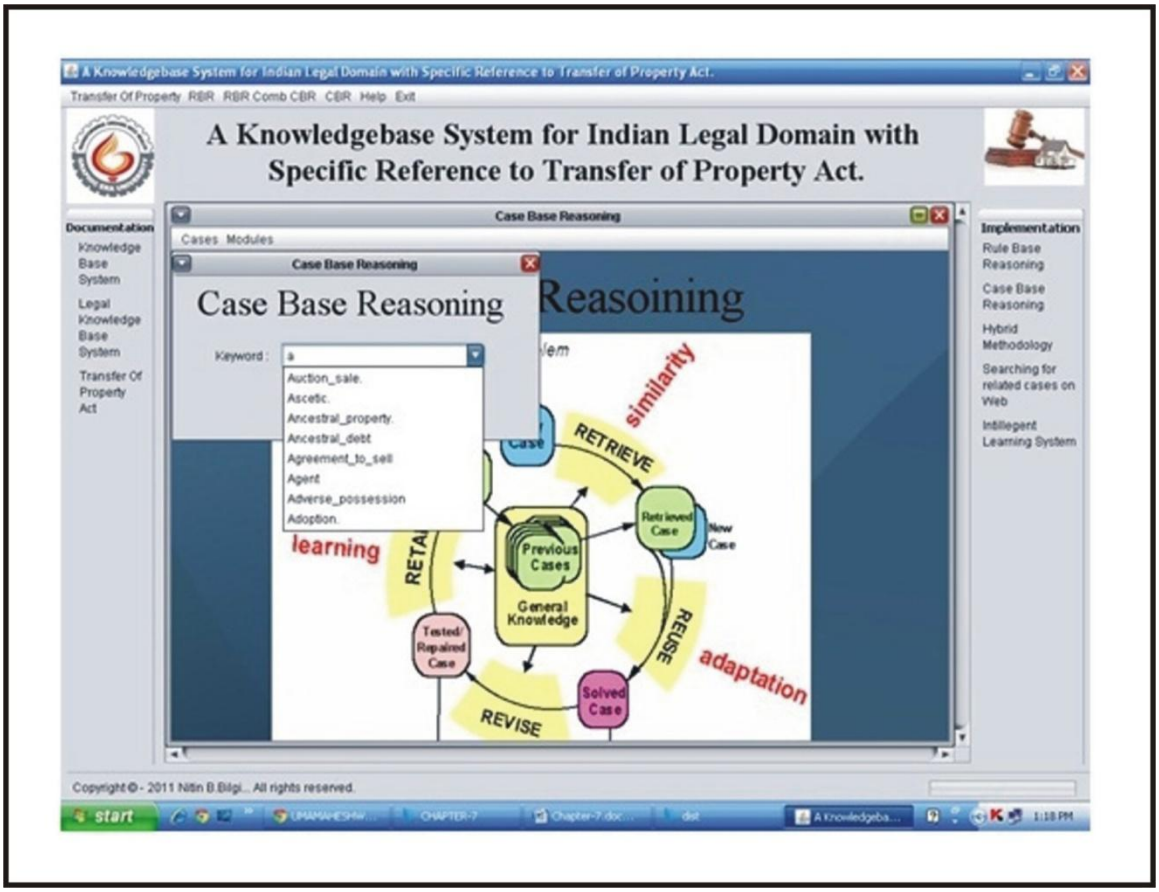

Figure 4.4: Case-based reasoning Screen

i) **Cases:** In this part if KEYWORD which is displayed in the drop down window is to be selected and depending on keyword selected the related cases which fall in its purview are displayed. The illustration is given figure 4.4. There can be one case or may cases which are displayed. If the user clicks on the related cases then information about the case is displayed. A sample case displayed in figure 4.5.

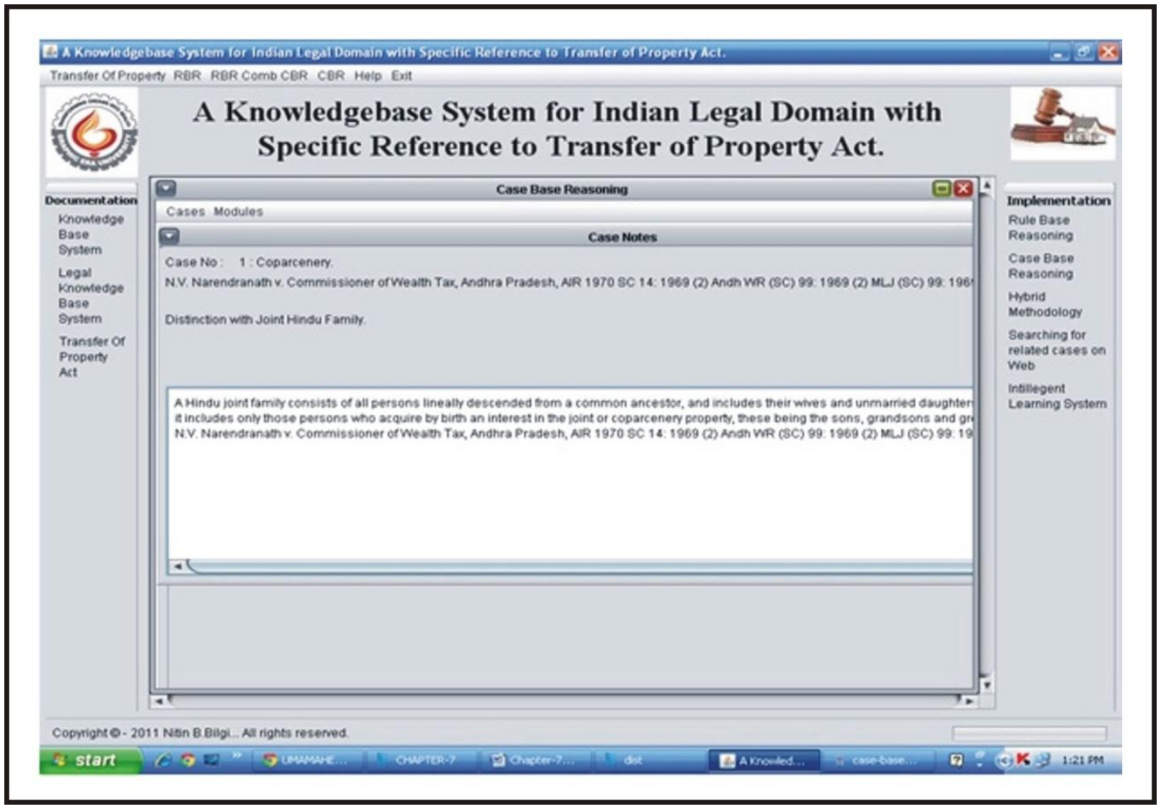

Figure 4.5: Sample Case notes in Case-based reasoning

ii) In this part the user can select any of the 10 modules and then select the related question in those modules. A keyword from that question will be selected. If any additional keyword is to be added by the user, then he can add the same and after that he can press the submit button. As a result of it, the related cases will be displayed. The figure 4.6 illustrates the module part.

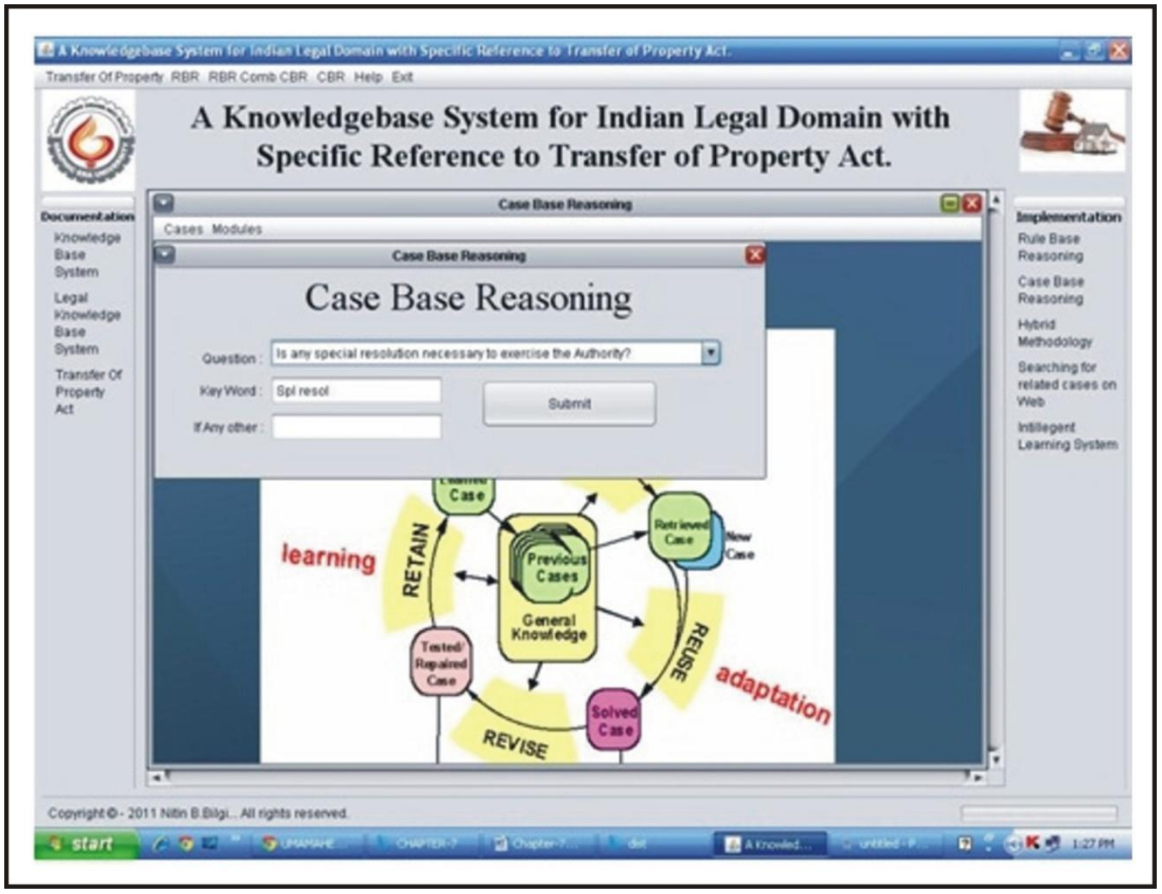

Figure 4.6: Module part in Case based reasoning

Module-2 intelligent learning part

In this module the user can load a case which should be in the form of .txt file in the window or even paste a case of text in window. We can analyze the text of data and find the statistics details of the case i.e. it is possible to find the occurrence of the important legal keywords and number of times they are being used. By this we can come to a reasonable conclusion that this case falls in a particular section of the transfer of property act. The figure 4.7 illustrates the same.

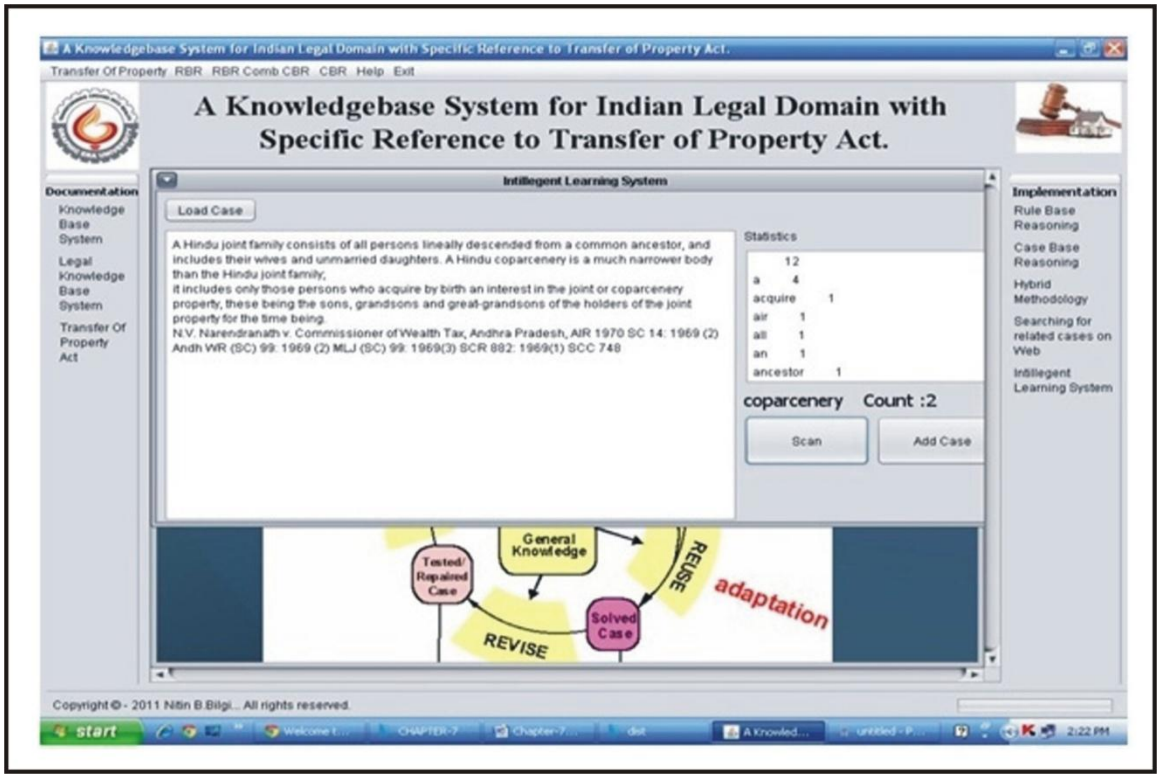

Figure 4.7: Intelligent learning part in Case-based reasoning

Module-3 Web-Search of related cases:

The web-search module is very useful module for the user as he can find current related cases available on the internet. This will give a greater insight to user so that he can refer to current cases which are available on HIGH COURT as well as on the SUPREME COURT website as well as some popular websites such as www.valkilno1.com . This is very useful feature for the lawyers as he can get some finer points from the cases on the internet and also he can be aware of the current cases. The same is illustrated in figure 4.8.

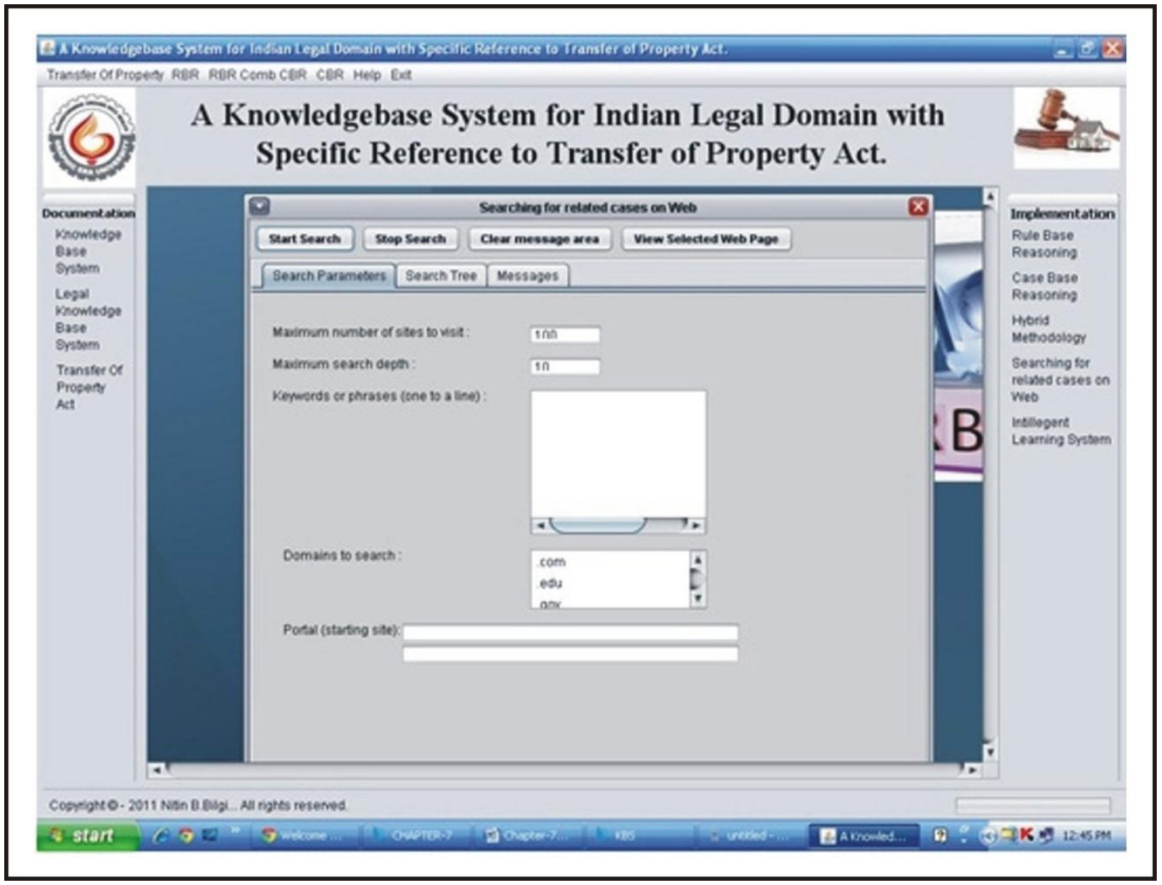

Figure 4.8: Web-Search Screen

4.4.3 Hybrid model development process

The first reasoning modality to be successfully integrated with CBR was rule-case based reasoning. The earliest CBR/RBR systems were built for statutory legal domains, where statutes naturally correspond to the rules and legal precedents naturally correspond to various cases. CABARET used a rule-based agenda mechanism to integrate past cases with legal regulations in the domain of United States tax law **[23].**

Most popular types of integration involve the combination of rule-based with case-based reasoning methodologies. The efforts to combine symbolic rules and cases have yielded advanced knowledge representation formalisms. The effectiveness of those approaches develops from the fact that rules and cases are alternatives in representing application domains and solving the problems. Rules represent general knowledge of the domain, whereas cases represent the specific knowledge. Rule-based systems solve problems from scratch, while case-based systems use pre-stored situations to deal with similar new instances. Hence, the integration of both approaches turns out to be accepted and useful way of expert system development **[23].**

The research work has combined rule-based system developed in VisiRule and the case-based in Netbeans. The output or the conclusion generated by the VisiRule and trace of the questions asked are redirected to a .txt file. This .txt file is read and the keywords are searched from this file. Then the keywords are compared with the cases and related cases are displayed. The representation of the same is given in following figure 4.9.

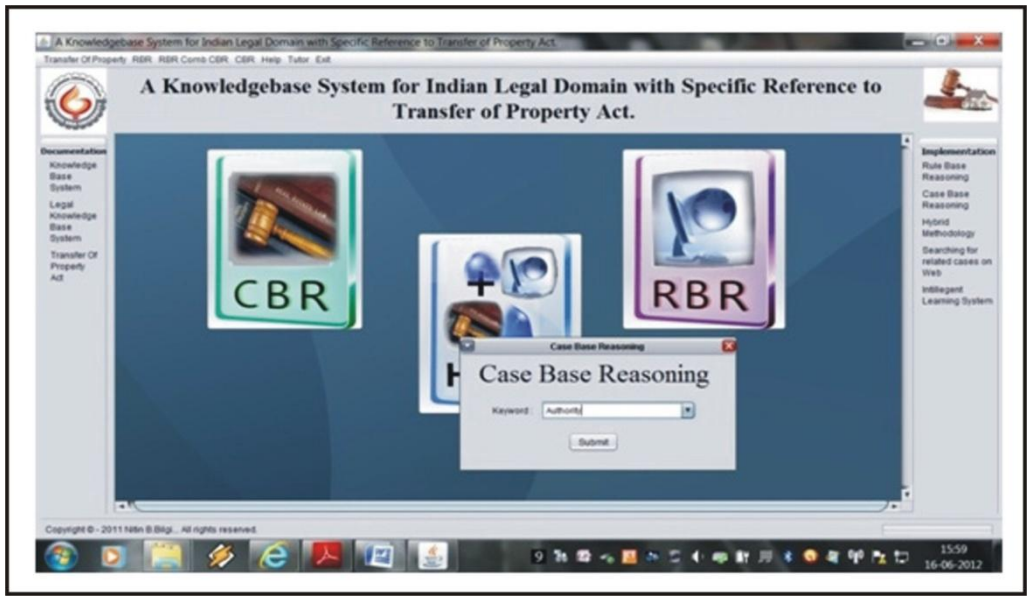

Figure 4.9: Hybrid model Screen

4.5 Conclusion

Rule-based and case-based reasoning are alternative ways of expressing knowledge. The hybrid approaches have managed to solve successfully the problems in application domains where rules and cases are available. Each representation formalism needs the assistance and/or completion of the other to work effectively. This research work presents a way of using rule base and case base knowledge representation. This analysis is based on the philosophy of human memory organization and utilizing for solving problems. A human generally represents knowledge in more than one form, in order to be more efficient to solve a problem. Also it is found that for any domain the knowledge can't be in one form. The proposed Hybrid Knowledge Representation Scheme, utilizes the combination of forward chaining reasoning and backward chaining reasoning, makes the knowledge-based systems and expert systems more flexible and efficient and also utilizes the flexibility of organization of knowledge-bases and the flexibility of problem solving methodologies. The applications of case-based reasoning in developing knowledge-based systems and the expert systems have been widely adopted in various domains and other application areas.

Thus this research work has developed a methodology for constructing a rule-based legal knowledge-based system using VisiRule. The rule framework as stated from the statues of books of law on Indian legal domain with a particular reference to the transfer of property act are converted in to the rule form and implemented using the VisiRule. The developed rule-based system is tested and evaluated by the lawyers. This system can be used by the non-law literate to verify the prima facie title to the property and absence of any encumbrances thereon. When the user is satisfied with these aspects, he can approach the advocate for final confirmations. This saves a lot of time and money. The expert system can be used by the lawyers to make decision faster. The related cases can be referred by the users of the system using the case-based modules.