ImmGnosis: Architecture for a Stateless Web-based Expert System for Immigration Law

by

Vineet Khosla

(Under the direction of Walter D. Potter)

Abstract

This thesis documents the design and implementation of ImmGnosis, a stateless Web-based expert system that reasons over matters involving United States immigration law. The emphasis of this thesis is on the stateless architecture of the system which allows for multiple consultations while using minimum machine resources. This thesis also documents the modifications made to an existing state dependent expert system shell (XSHELL) in order to make it stateless and deployable over the Web. The system is currently capable of determining a user's U.S. citizenship status, admissibility status, and eligibility for naturalization and a visa recommendation module is currently under development.

Index words: Expert System, Legal Expert System, Expert System Shell, Stateless Architecture, Portable Blackboard, Immigration Law, Rule-Based Reasoning, Prolog, Java

ImmGnosis: Architecture for a Stateless Web-based Expert System for Immigration Law

by

Vineet Khosla

B.S., Pittsburg State University, 2002

A Thesis Submitted to the Graduate Faculty
of The University of Georgia in Partial Fulfillment
of the
Requirements for the Degree

Master of Science

Athens, Georgia

2005

ImmGnosis: Architecture for a Stateless Web-based Expert System for Immigration Law

by

Vineet Khosla

Dedication

I would like to dedicate this thesis to my family. Mere words cannot express my sincere feelings nor the gratitude I owe to them.

Acknowledgments

I would like to express my gratitude to all of my committee members. I would like to thank Dr. Nute for the invaluable advice he gave me throughout this project. Without his knowledge, patience, and guidance this project would not be possible. I also wish to thank Dr. Potter for not only providing me with valuable AI skills but also sharing his valuable study strategies for maintaining the highest academic standards. Many thanks are owed to Dr. Estes for giving his support and time to this project.

I would like to acknowledge Blair Dorminey for funding this research and providing us with a team of domain experts. I would like to thank David Tooley, Aschalew Nigussie, Yongnam Park, and Yannick Morgan for providing the expert knowledge for this project.

I am also thankful to my research partner Daniel Dejuan for working endless hours and ensuring that our work is of highest quality.

I would also like to thank all the faculty and friends at the Artificial Intelligence Center at University of Georgia for providing me with an intellectually stimulating environment. Special thanks are owed to Makiko Kaijima for proofreading this thesis and providing me with valuable suggestions throughout the thesis writing process.

Finally, I would like to express my gratitude to all the people who have not been a direct part of this thesis but influenced me during my undergraduate degree at Pittsburg State Univeristy. I thank Dr. Maeve Cummings for introducing me to the exciting field of AI. I also thank Mr. Greg Brown and Dr. Ramon M. Figueroa-Centeno for being my teachers, my mentors, and above all, my friends.

My sincere thanks to you all.

Preface

This thesis is a part of a team project done at University of Georgia's Artificial Intelligence center by myself and Daniel DeJuan. I have tried to include only those topics in this thesis on which I have either worked alone or was an equal contributing member. Topics which are not my work but are pertinent to the complete understanding of the system are clearly marked.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Problem Domain

1.1.1 Background

The domain of this expert system is immigration law. More specifically the United States Immigration and Nationality Act Title 8 of the U.S. Code (8 USC), which was first created in 1952 and subsequently amended over the years. This act along with other immigration laws, treaties, and conventions of the United States relates to the immigration, temporary admission, naturalization, and removal of aliens. This act applies to all United States citizens and people wishing admission to the United States for any reason.

1.1.2 Problem Statement

The Immigration and Nationality Act is a labyrinth of laws, bylaws, and amendments, which make the task of researching a case rather time intensive. It is hard even for a legal professionals to ensure whether a case has been researched completely and no important information is left out. By automating this process, not only the temporal cost of research will be reduced, but it will also ensure a more thorough and valid research.

1.2 Expert System Approach

1.2.1 Expert Systems: A Brief Overview

An expert system is a computer program that incorporates the specialist's knowledge in order to solve problems or give advice. An expert system may either completely replace the human expert or play the role of an assistant. Expert systems consist of five parts: a knowledge base, an inference engine, a blackboard, an explanation facility, and a user interface. The knowledge base contains an expert's knowledge (facts) and problem-specific heuristics (rules or cases). The inference engine enables step-by-step logical reasoning about this information. During this logical operation, the intermediate results are kept on the blackboard (working memory). The contents of the blackboard are updated if a new (intermediate) result is derived. The explanatory facility justifies the conclusions reached by displaying the reasoning or the rules used for reaching a conclusion. The user interface facilitates all communication between the user and the system by obtaining necessary information from the user through queries and by displaying recommendations and any information supplied by the explanatory facility.

Of all the components mentioned above, only the knowledge base is dependent on the domain while all the other components can be developed irrespective of the domain of the expert system. This feature is particularly important in constructing an expert system, because it allows programmers to construct "expert system shells" which then allows knowledge engineers to construct fully functional expert systems rather quickly.

1.2.2 Case-Based vs. Rule-Based Expert Systems for Law

Though law has long been a popular domain for expert systems, it is also a difficult domain due to the complexity of legal reasoning. The arguments made in the court are either based on specific laws or previous cases. Previous cases are especially valuable when the laws are not precise and the only way to reason is to cite previous cases. Based on their reasoning mechanisms, the majority of legal expert systems can be classified in two categories, case-based reasoning systems and rule-based reasoning systems.

Like a human expert, a case-based expert system uses previously known cases from its knowledge base to reason about the present case, by comparing the present case with them. Case-based reasoners store abstract knowledge of cases such as the facts of the case, the result, and possibly the reasons for reaching that result. SHYSTER, developed by Popple

(1993), reasons over Australian law and represents the newest of case-based statistical legal reasoning systems. One of the advantages of using case-based reasoning in a legal expert system is that it can deal with a case, about which laws are not clearly defined or which allow for various interpretations when applied. However in case-based reasoning, there is considerable difficulty in obtaining a sufficiently large number of cases and there are often no predetermined criteria for deciding which features of a case are most important and should be included in its feature set. Case-based reasoners also have less sophisticated conflict resolution mechanisms and their performance has been noted to degrade, when reasoning with a large number of cases in their knowledge base (Jackson 1999).

In rule based systems, knowledge gathered from the experts of the domain is stored as rules in "IF - THEN" format and reasoning is done by matching these rules to data written to the working memory or the blackboard. In rule-based reasoning systems we can either do backward chaining or forward chaining to reach a conclusion. In backward chaining we start with a goal state and then try to look for all the relevant, supporting processes that will lead us to achieve our goal state. On the other hand, forward chaining is a more data driven approach where we try to see what conclusion can we draw with the current data. Rule-based legal systems have been applied to domains like immigration law and property law where laws are precisely defined and do not leave space for various interpretation. One of the earliest rule-based legal expert systems, JUDITH, was created by Popp and Schlink (1975) at Stanford University. The JUDITH system is a general legal expert system designed for use by a lawyer. Its primary use was for researching open-textured terms[1] using a rule based approach. This was followed by TAXMAN, a system created by McCarty (1977) that focuses on the taxation of corporate restructuring as legislated in subchapter C of chapter I of the Internal Revenue Code of 1954 in the United States of America. Rule-based reasoning systems are more popular owing to the general notion that intelligent behaviour is rule governed and experts find it extremely useful to present their knowledge as a set of rules.

[1]A term is open-textured if the system of rules does not contain an explicit definition of the term

Consequently, there are more expert system shells available that use rule-based reasoning rather than case-based reasoning. The biggest disadvantage of rule-based systems comes from the inconsistency in the rules supplied by the experts.

1.3 Description of study

1.3.1 Purpose and significance

Dealing with immigration law is complicated and practitioners would benefit from an automated system to efficiently search statutes applicable to a particular case. Immigration law can be effectively broken down into smaller and more manageable sub-domains such as citizenship status of an individual, citizenship by naturalization, admission qualifications for aliens, travel control of citizens and aliens, immigrant and non-immigrant visas, etc.

Trying to use only case-based reasoning or only rule-based reasoning for the entire domain of law is an inappropriate approach. There are some sub-domains of law, in which case-based reasoning would better serve the purpose because laws in that area are not precisely defined; hence, the only way to reason is to cite previously decided cases. Yet for other sub-domains where laws are extremely precise, a rule-based system would be a better approach. Since immigration laws are fairly precise, a rule-based system would be a better approach than case-based reasoning. Additionally, case knowledge in immigration law is not far removed from the actual Immigration and Nationality Act statutes and since the laws are extremely precise, all the cases would look the same. Consequently, rules derived from the Immigration and Nationality Act offer a better representation for our domain knowledge.

Hence ImmGnosis, which is a rule-based legal expert system, was designed and developed. Such a system has several potential practical applications. It can be employed as an in-house tool for immigration law firms that replaces the traditional preliminary consultation and removes the temporal cost of individually evaluating each new case that presents itself. This allows potential clients to make an initial determination of the worth of pursuing their cases while circumventing, or at least minimizing, the financial and temporal costs associated with initiating a traditional consultation with a human attorney. Alternatively, a firm could offer public access to such a system over the Web, allowing a far greater number of users to access the system. For this second application, the system would diagnose a potential client's case and if the system concludes that the case is worth pursuing, it can refer the user to the firm providing the service and can store his or her consultation in a database for future reference. It also benefits legal practitioners by automatically sorting potential cases at a lower cost to the firm (Dejuan et al. 2005).

The goal of this study is to develop an efficient expert system shell and build a Web-based expert system using that shell which will reason over various facets of American immigration law.

1.3.2 Organization of Study

This thesis documents the design and implementation of an expert system shell and ImmGnosis, a stateless Web-based expert system built using that shell that reasons over matters involving United States immigration law. Chapter 2 documents the expert system development lifecycle with emphasis on knowledge acquisition. Chapter 3 documents the system architecture and the physical setup of the system. Chapters 4 and 5 discuss the various components of the expert system and describe the original shell and the modified shell on which this expert system has been built. Chapter 6 concludes the thesis by presenting the results of preliminary evaluation and suggests future plans for ImmGnosis.

Chapter 2

Expert System Development Lifecycle

The development life cycle for ImmGnosis followed the life cycle suggested by Awad (1996) which is itself derived from the Buchanan life cycle (Buchanan et.al 1983). Awad's life cycle involves five steps: problem identification, knowledge acquisition, knowledge representation and prototyping, verification and validation, and testing and implementation. Note that this lifecycle is not purely sequential. Knowledge acquisition is a unique step since it runs parallel to all the other steps. Knowledge acquisition is not a one time step where complete knowledge of the domain is obtained before the development of the expert system. Whenever new knowledge is obtained from the expert, it is then represented as a rule (step 3), verified by the expert (step 4) and finally implemented in the system (step 5).

2.1 Problem Identification

In this first stage, the characteristics of the problem were identified. Emphasis was placed on understanding the problem domain, scope of the problem and what resources would be available to solve the problem. In this stage, there were a series of informal meetings with the expert where indepth knowledge about the domain and the problem were obtained.

2.2 Knowledge Acquisition

In the second stage, the knowledge engineer either interviews the expert in order to elicit knowledge or elicits knowledge from other credible sources. This is unarguably the toughest stage for the knowledge engineer. In some expert systems, the task of knowledge elicitation has been assigned to cognitive psychologists in order to ensure high quality knowledge is gathered from the experts. Unfortunately such luxury was not available for this system and the task of knowledge acquisition was pursued by myself and the co-creator of this system, Daniel DeJuan.

2.2.1 Knowledge Acquisition Methods

The knowledge was obtained from experts either through direct interview or from documents prepared by them. The experts would write their interpretations or directly take statues from the Immigration and Naturalization Act and write them in an "IF - THEN" format. These rules were then transformed into knowledge base rules which the inference engine could interpret.

The primary knowledge acquisition method used for ImmGnosis was the later method of extracting rules from documents written by experts. This method is better suited than direct interview for the domain of immigration law since most of the knowledge that the experts had came from the Immigration and Naturalization Act. Direct interviews with the experts consisted of resolving ambiguities in these documents.

2.2.2 Knowledge Acquisition Results

The experts identified four distinct yet interdependent sub-domains of immigration law to be pursued first. These are not the only sub-domains which together constitute immigration law, but were the ones which were identified by the client to be more important than other possible sub-domains. These four sub-domains are listed below.

1. Citizenship status: This sub-domain dealt with finding out whether a user was already a citizen of the United States of America.

2. Citizenship by naturalization: This sub-domain dealt with finding out whether an individual was eligible for United States citizenship by way of naturalization.

3. Admission qualifications for aliens: This sub-domain dealt with finding out whether an individual alien was eligible to enter the United States or not.

4. Visa recommendation: This sub-domain dealt with recommending the type of entry visa appropriate for an alien visitor.

Breaking the domain into smaller sub-domains allowed for development of knowledge bases in parallel which were later combined to form a complete and comprehensive knowledge base in a relatively short period of time. Knowledge acquisition and development of knowledge bases for the first three sub-domains was pursued by Daniel DeJaun while the author mainly dealt with the Visa recommendation sub-domain. More detailed discussion about the first three sub-domains can be found in (Dejuan 2005).

The Visa recommendation sub-domain is a unique sub-domain since an individual can be eligible for more than one type of Visa; hence multiple recommendations are possible. On the other hand, a decision from other sub-domains like “citizenship status” would be a “yes” or a “no” decision. This sub-domain is also one of the bigger sub-domains; hence there was a need to evaluate it thoroughly and identify relationships between the objects in order to build an efficient system. Figure 2.1 and Figure 2.2 present a brief overview of the scope of this sub-domain and how a consultation in this sub-domain proceeds. Representing the sub-domain as a tree helped facilitate exchange of knowledge between the knowledge engineer and the domain expert. It also helped to ensure that the sub-domain was covered completely and in an efficient manner.

For the sub-domain of visa recommendation, the experts identified seven possible reasons why a potential client will initiate contact with an immigration attorney. These seven reasons are exhibited at Level 1 of Figure 2.1 and are the starting grounds for a consultation in this sub-domain. When a user starts a consultation, the first question they are asked is “What is the purpose of your consultation?” The choices available to them are: to obtain a new visa, to sponsor a visa for an employee, to change status of an existing visa, to reinstate an expired visa, to get an extension on an existing visa, to determine the validity of a visa, or to apply for legal permanent residence. If they choose “to obtain a new visa”, then the next question will enquire about the purpose of their stay in the United States. The responses available to the users are all the nodes in Level 2 of Figure 2.1. If the users chooses “entertainment or athletic competition”, then the expert system will try to determine whether a P1 or P2 visa is appropriate for them. Figure 2.2 displays a similar process for an employer wishing to sponsor an individual for work in the United States.

Note that the levels in these trees don’t imply the depth of the search tree for this sub-domain, but are used to understand and model the domain better. The terminal nodes of the tree in Figure 2.1 and Figure 2.2 marked in bold font indicate those visa categories whose rule sets are complete and have been tested. In the current state, this sub-domain has 55 rules in it.

2.3 Knowledge Representation and Prototyping

The main objective of this stage is to map key concepts and relations on to formal representations and to select appropriate tools. In this stage the knowledge engineer designs structures to organize knowledge such that the inference engine can use it to draw conclusions. For ImmGnosis, this stage ran in parallel to stage 2 and the decision to use and modify XSHELL (Covington et al. 1997), an existing expert system shell for Prolog was made. The original XSHELL is a rudimentary shell which uses prolog’s built in inference engine. It provides a limited text interface and explanatory facility and is not built to handle multiple users. The modifications to the shell included improving all those features and also allow for consultations by multiple users.

2.4 Verification and Validation

For the fourth stage the verification and validation for ImmGnosis rules was done in two steps. First the domain experts verified the rules by reading them. Since the rules of the ImmGnosis knowledge base have an “IF - THEN” structure, the experts could easily read

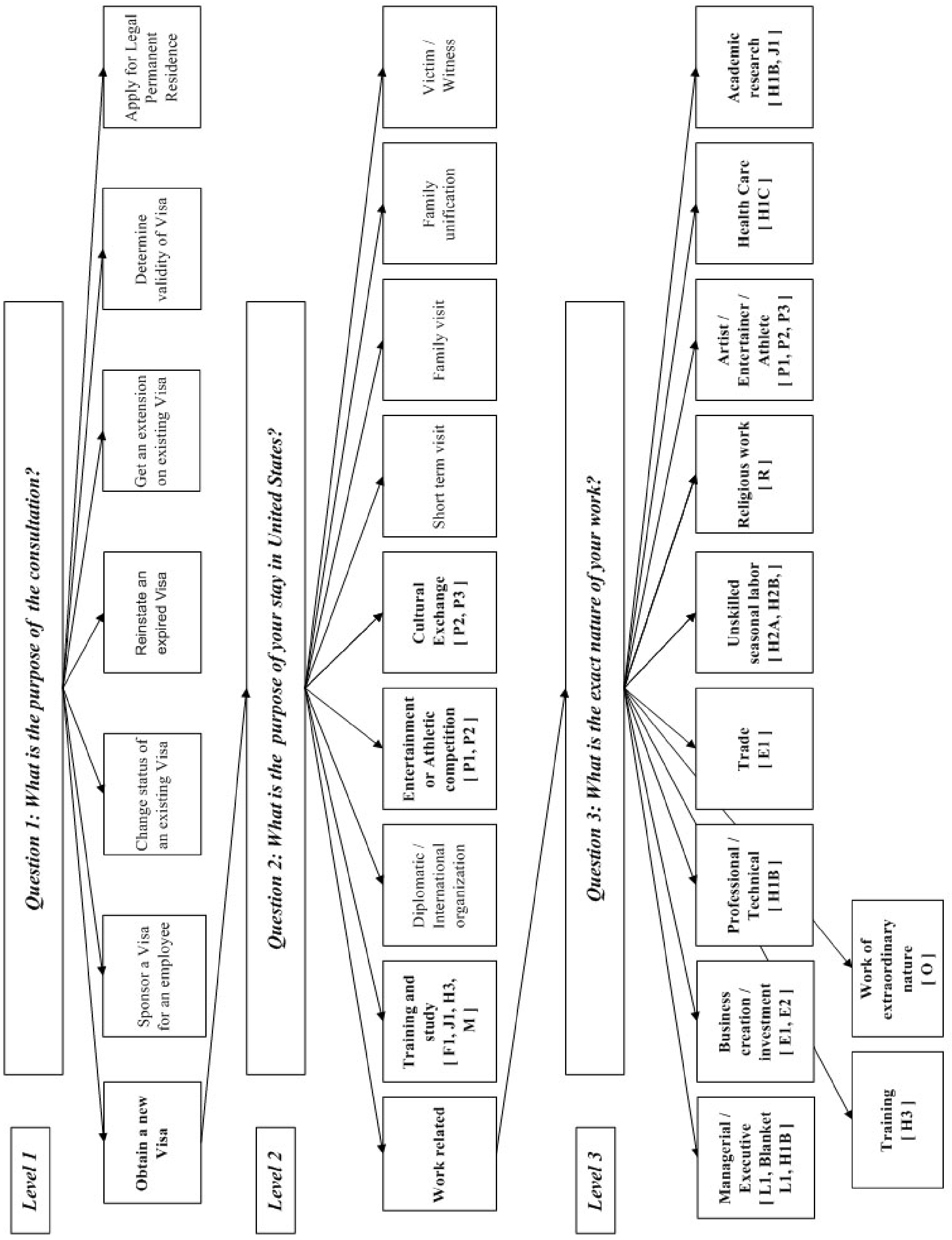

Level 1
Question 1: What is the purpose of the consultation?
Obtain a new Visa
Sponsor a Visa for an employee
Change status of an existing Visa
Reinstate an expired Visa
Get an extension on existing Visa
Determine validity of Visa
Apply for Legal Permanent Residence
Level 2
Question 2: What is the purpose of your stay in United States?
Work related
Training and study [F1, J1, H3, M]
Diplomatic / International organization
Entertainment or Athletic competition [P1, P2]
Cultural Exchange [P2, P3]
Short term visit
Family visit
Family unification
Victim / Witness
Level 3
Question 3: What is the exact nature of your work?
Managerial / Executive [L1, Blanket L1, H1B]
Business creation / investment [E1, E2]
Professional / Technical [H1B]
Trade [E1]
Unskilled seasonal labor [H2A, H2B,]
Religious work [R]
Artist / Entertainer / Athlete [P1, P2, P3]
Health Care [H1C]
Academic research [H1B, J1]
Training [H3]
Work of extraordinary nature [O]

Figure 2.1: Work related Visa: Employee perspective

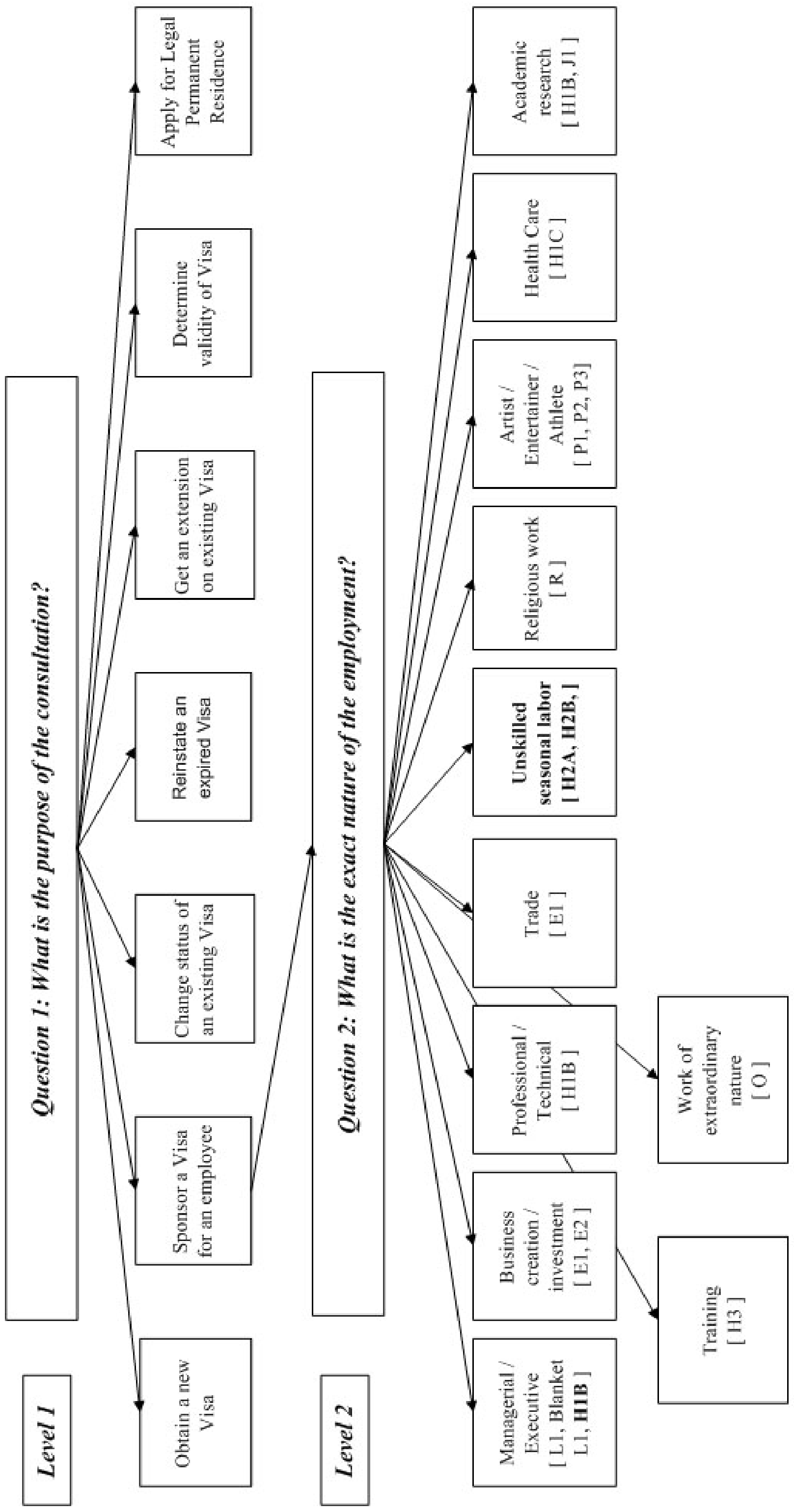

Level 1
Question 1: What is the purpose of the consultation?
Obtain a new Visa
Sponsor a Visa for an employee
Change status of an existing Visa
Reinstate an expired Visa
Get an extension on existing Visa
Determine validity of Visa
Apply for Legal Permanent Residence
Level 2
Question 2: What is the exact nature of the employment?
Managerial / Executive [L1, Blanket L1, H1B]
Business creation / investment [E1, E2]
Professional / Technical [H1B]
Trade [E1]
Unskilled seasonal labor [H2A, H2B,]
Religious work [R]
Artist / Entertainer / Athlete [P1, P2, P3]
Health Care [H1C]
Academic research [H1B, J1]
Training [H3]
Work of extraordinary nature [O]

Figure 2.2: Work related Visa: Employer perspective

the rules and verify them. Once these rules were verified by the expert, they were added to the knowledge base and then they were validated by running the system and evaluating its response against the expert's own responses to the given case. This process was informal and no written record was kept since the aim was to catch obvious errors in the knowledge base.

2.5 Testing and Implementation

In this last stage, the rules that embody the knowledge are validated by user testing, peer reviews; and any other testing method that the client may desire. Since this is a yet to be launched commercial product, comprehensive testing by peer review was not allowed. The completed system was tested by the domain experts who supplied the expertise for the development of the knowledge base.

Chapter 3

System Architecture

A rule-based expert system that deals with immigration laws (ImmGnosis) was designed and implemented in order to efficiently evaluate the legal merits of a case and reduce the time required for research by legal professionals . ImmGnosis is a Web-based system which allows multiple users to access the system simultaneously. The advantages of using a Web-based system are that it

1. can handle multiple users efficiently;
2. can store proprietary information securely; and
3. permits immigration law stored in the knowledge base to be updated easily.

Another option to a Web-based system was a downloadable program, but that would have made the task of securely maintaining the proprietary information difficult. Furthermore, a downloadable program would have made updating the immigration law knowledge base difficult and time consuming because immigration laws change often. In order to provide accurate up-to-date recommendations, we need to modify the knowledge base at the central server only once instead of requiring the users to constantly install updates. All the above mentioned considerations, plus the need to store user consultations in order to monitor the accuracy of the system and identify areas of future research based on user trends, led to a Web-based system with major client side load.

3.1 Main components of ImmGnosis (Java Server Pages, Intelligence Server and Prolog Inference Engine)

As shown in Figure 3.1, the three important server components of ImmGnosis includes Java Server Pages (JSP), the Intelligence Server (IS), and a Prolog inference application (i.e., knowledge base and inference engine). JSP at the front end, deployed on an Apache Tomcat 5.0 Web server[1], governs the user interface.

JSP, a part of the Java technology family, allow Web designers and developers to develop and maintain dynamic Web pages. JSP are created using syntax very similar to HTML and have Java embedded in them. They are an extension of Java Servlet technology and are configured to work with an HTTP server that passes CGI complaint input parameters to it (McPherson 2000). The role of JSP is to retrieve the inputs provided by the user using the GET evocation and display the response from the Prolog-based inference application.

Logic Programming Associates' IS at the back end is an integral part of the system and facilitates the interaction between the server and the system's Prolog-based inference application. The IS provides a simple text interface between Java and the Prolog language instead of requiring a direct mapping between their complex data structures. In this model, Java sends queries to Prolog and receives output from the by Prolog programs as text strings (Westwood and Steel 2004).

The LPA WIN-Prolog application at the back end is the knowledge base and inference engine of the expert system and it is explained more in detail in Chapters 4 and 5.

3.2 Stateless Architecture with Portable Blackboard for Efficient Multiple User Access

ImmGnosis allows multiple users to access the system at the same time with minimum server side load by incorporating a portable blackboard in a stateless architecture. We benefited

[1]The Apache Jakarta Project. Web: http://jakarta.apache.org/tomcat/

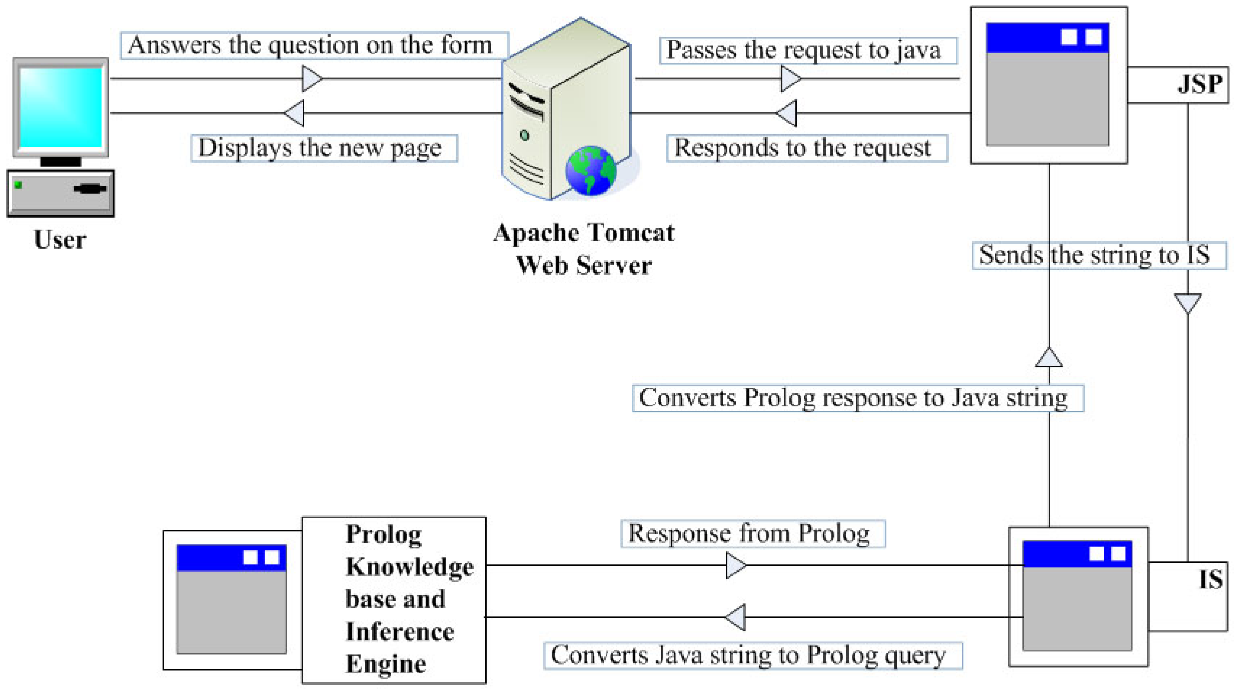

Answers the question on the form
Displays the new page
User
Apache Tomcat Web Server
Passes the request to java
Responds to the request
JSP
Sends the string to IS
Converts Prolog response to Java string
Prolog Knowledge base and Inference Engine
Response from Prolog
Converts Java string to Prolog query
IS

Figure 3.1: System Architecture

from reading Jennings (2002), which describes an earlier implementation of a stateless architecture for Web-based expert systems. This is the most efficient method since only a single instance of the knowledge base and inference engine can support multiple users.

3.2.1 Stateful Architecture

An expert system can be stateful or stateless. In a stateful system, a current state and preceding events (i.e., history of user's interaction with the system) are used to determine the next query or to make a recommendation. One of the major disadvantages of using a stateful architecture for ImmGnosis is that it cannot handle multiple users at the same time. If only one user is using the system, the interactions between the user and the system are easily tracked by just updating the working memory or blackboard where the dynamic facts of that interaction are being stored. Hence, the system is able to provide appropriate query and proceed to the next state without a problem. However, if multiple users are accessing the system simultaneously, this will be a problem since facts from different users will be written to the same working memory thereby generating inappropriate conclusions. Hence it is impractical to use a stateful architecture for a Web-based expert system that needs to process multiple clients' queries in parallel.

3.2.2 Stateless Architecture with Portable Blackboard

ImmGnosis is a stateless system designed around a portable blackboard architecture. ImmGnosis incorporates features which allow for distinguishing each user while using a single inference engine and knowledge base to support multiple users hence using minimum machine resources.

In a stateless system, neither the current state nor the previous events are stored. The statelessness of ImmGnosis is derived from its portable blackboard. In its structure, a portable blackboard is just like a regular system blackboard, which is permanently located inside the system. The differences between the portable blackboard and system blackboard are that a portable blackboard is assigned to each user and it travels back and forth between the user and the system. Each portable blackboard contains all the facts pertaining to the consultation and is copied to the system blackboard when the system receives new inputs from a user. When the system has satisfied the current query either by generating a conclusion or a new question, the information in the system blackboard is copied back to the portable blackboard and the information in the system blackboard will be deleted in order to process inputs from other users.

The blackboard assigned to each individual was made portable in order to avoid server clutter and reduce server side load. If the blackboard is maintained at server side and the internet connection gets disconnected or the user decides to leave the consultation in the middle, the server would maintain the user's blackboard in its last state indefinitely, and eventually the server will get cluttered with inactive blackboards. Additionally it will have to distinguish between all incoming requests and find the appropriate blackboard to add new information obtained from the user.

In ImmGnosis, each user is tracked over the period of a consultation using a unique Java Session ID[2] assigned during the initial consultation. This is necessary because HTTP is a stateless protocol that opens a separate connection to the Web server each time a user retrieves a Web page. The server does not automatically maintain contextual information about a client. Session ID was also used to distinguish portable blackboards assigned to different individuals.

"Stateful" and "stateless" are derived from the usage of state as a set of conditions at a moment in time. Computers are inherently stateful in operation; so these terms are used in the context of a particular set of interactions, not of how computers work in general. It is important to note that the claim to statelessness applies only to the expert system and not to the Website through which the system is delivered. The Website is indeed stateful and it keeps track of users moving from one page to another and performing actions other

[2]User tacking by Session ID is a mechanism provided by JSP to maintain states about series of requests originating from the same Web browser.

than conducting a consultation with the expert system. It is the expert system which does not need to store previous knowledge of interactions with the user in order to successfully process their query.

3.3 Data-flow in a Sample Interaction

Data flow in ImmGnosis is presented in Figure 3.2. Whenever a new consultation is started, the user fills out a generic form, which asks for the user's date of birth, place of birth, and the sub-domain of immigration law that the user wants to explore. When the user submits the initial form, Java assigns the user a unique Java Session ID and generates a portable blackboard for the individual. Java then takes the user's answers from the original form and packages them as a text string. This text string is then sent to the IS, where it is converted to a Prolog query string. This query string is then sent to Prolog and the retrieved information is posted on the blackboard. The Prolog inference engine then tries to reach a conclusion based on the information currently on the blackboard. If the information is not sufficient for reaching a recommendation, then Prolog generates a new JSP file (i.e., Web page) and writes a question (along with its possible answers) in a form. In addition, Prolog copies the contents of the system blackboard to a portable blackboard, which is inserted in the JSP file as a hidden control field. The memory address where this file is stored is sent to the IS, JSP, and passed back to the Web server. Then the Web server displays the page to the appropriate user based on the Session ID associated with this file. When a user answers a new question, Java appends the answer to the hidden blackboard string and sends it to Prolog where the whole process is repeated until a conclusion is reached. When Prolog finally reaches a conclusion, it creates a new JSP file that has the conclusion and the answers to all the previous questions in it. This JSP file is then sent back to the Web server. At this point the user has the option of either saving their consultation to a database or permanently deleting it from the system. If a user chooses to save their consultation, their current blackboard is saved to the database and in subsequent visits they can retrieve it and view or edit that consultation. Note that, in either case, Prolog deletes all the facts about that user from its blackboard after sending the JSP files to the Web server in order to handle requests coming in from other users.

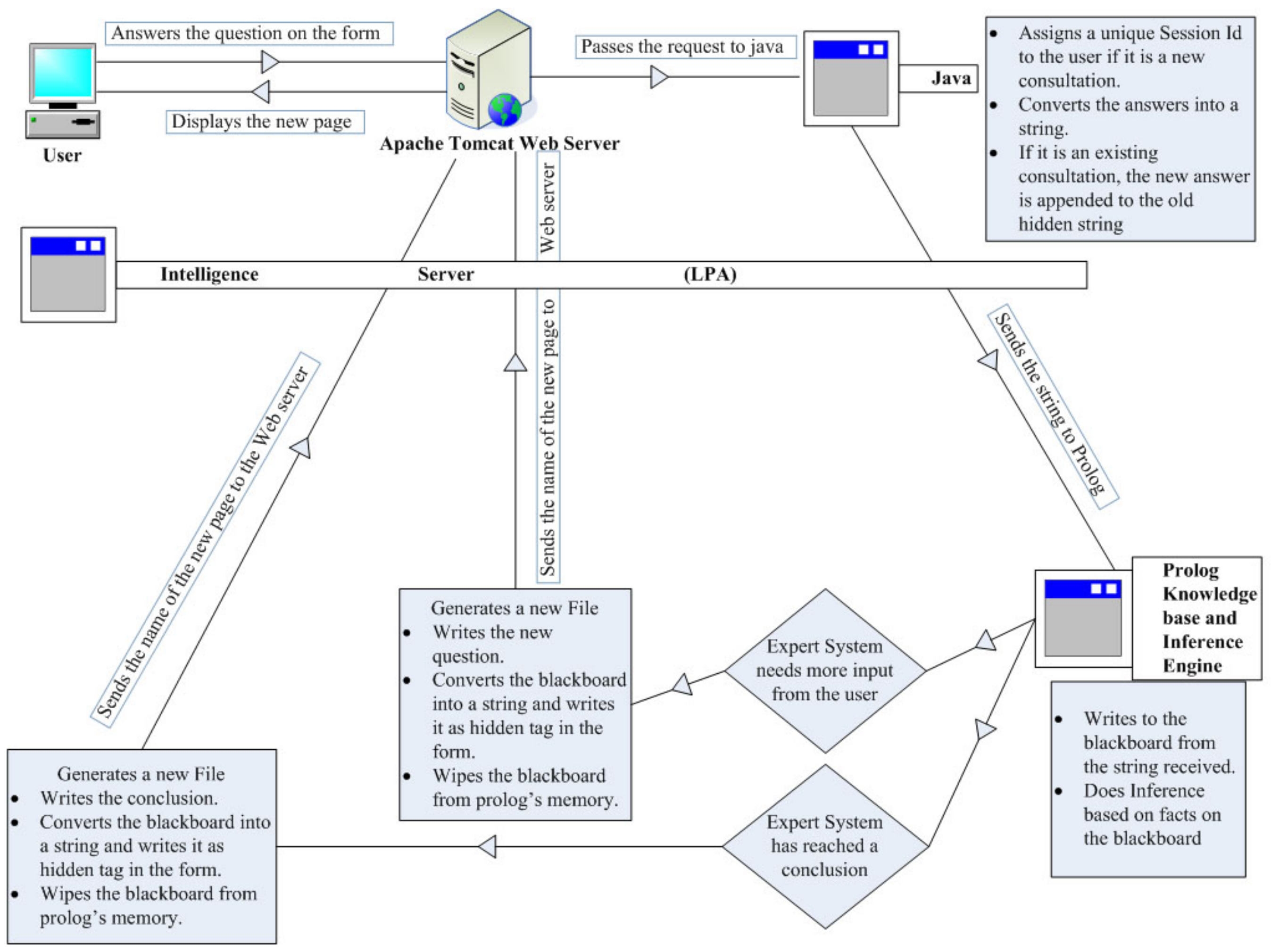

Answers the question on the form
Displays the new page
User
Apache Tomcat Web Server
Passes the request to java
Java
• Assigns a unique Session Id to the user if it is a new consultation.
• Converts the answers into a string.
• If it is an existing consultation, the new answer is appended to the old hidden string
Intelligence Server (LPA)
Web server
Sends the name of the new page to
Sends the string to Prolog
Sends the name of the new page to the Web server
Prolog Knowledge base and Inference Engine
• Writes to the blackboard from the string received.
• Does Inference based on facts on the blackboard
Expert System needs more input from the user
Expert System has reached a conclusion
Generates a new File
• Writes the new question.
• Converts the blackboard into a string and writes it as hidden tag in the form.
• Wipes the blackboard from prolog's memory.
Generates a new File
• Writes the conclusion.
• Converts the blackboard into a string and writes it as hidden tag in the form.
• Wipes the blackboard from prolog's memory.

Figure 3.2: Data-flow in a Sample Interaction

Chapter 4

XSHELL Design

An expert system shell typically consists of an inference engine, a user interface, and sometimes an interface to build knowledge bases, but has no knowledge base. Shells provide a platform to build a knowledge base by simply filling in the domain knowledge. Hence, using a shell is an efficient way of building an expert system because it takes the burden of writing an inference engine and user interface away from the knowledge engineers and system developer. There are many commercial shells available, such as CLIPS[1], EXSYS[2], and FLEX[3].

ImmGnosis was developed using XSHELL (Covington et al. 1997), which is a shell written in Prolog and is freely available. It does not have several features found in other high end shells. But an advantage of XSHELL is that it can be modified easily, because usually it is hard to modify commercial shells. The rest of this chapter briefly describes the design of XSHELL, while Chapter 5 documents the changes made to XSHELL. A more detailed discussion about XSHELL can be found in Covington et al. (1997).

4.1 Inference Engine and Procedure

XSHELL employs a depth-first backward-chaining inference mechanism, which is built into Prolog. A backward-chaining system starts its inference from a goal state and tests if all the relevant supporting facts can be obtained. The inference engine picks a rule in lexical order and tries to match the antecedent of the rule to the information in the current working memory in order to determine if the premises of the rule are satisfied. If the premises are

[1]CLIPS. Web: http://www.ghg.net/clips/CLIPS.html
[2]EXSYS. Web: http://www.exsys.com/
[3]Logic Programming Associates. Web: http://www.lpa.co.uk/flx.htm

not satisfied, the inference engine will start looking for another rule that can be used. This process continues until either all the premises of some rule are satisfied or the inference engine runs out of rules. Note that inefficient ordering of rules can result in a longer path to the conclusion because rules are traversed lexically.

In XSHELL, a single query to the inference engine leads to a sequential building of the blackboard by searching through the entire knowledge base. A query to an expert system developed upon XSHELL would lead to the display of introductory text followed by a call to `xkb_identify/2`[4] "The two place predicate `xkb_identify` defines the knowledge component of XSHELL" (Jenning 2002). Each `xkb_identify/2` clause is a rule that defines an answer to a set of user inputs. Prolog tries to satisfy each clause in the order it appears in the rule base. A query on a rule results in a query on the clauses in that rules body in order. Each clause represents a certain condition, which is checked by the inference engine to identify whether that condition is already satisfied or negated according to previous user input. If the condition is satisfied, then the inference engine continues on to the next one. If the condition is negated, then the rule fails and the engine backtracks to the next rule (if present). Once all conditions for some rule are satisfied, the rule succeeds and returns an answer to the user. However, if no rule succeeds given the user input, the system simply reports that no further conclusion can be reached.

Hence a single query to the system starts a process, which will be terminated when the system reaches a conclusion. During this process, every action by the system is dependent upon the previous state of the system, thereby making it stateful. A direct consequence of the stateful architecture of an expert system developed using XSHELL is that it can support only one user at a given time.

[4]This is standard Prolog notation where the number "2" indicates the arity of the predicate.

4.2 Knowledge base

Since XSHELL is an expert system shell and does not contain actual knowledge of a domain, the discussion in this section is limited to the structure of rules only.

Rules in XSHELL's knowledge base are `xkb_identify/2` clauses. The first argument of this clause is an integer that represents the rule number. In XSHELL, this number is not used in computation and simply provides an identification for the system developer. These numbers have to be unique and sequential; however, the order of rules in the knowledge base does not matter. The second argument is a list of atoms that the inference engine uses to display appropriate output if the rule succeeds.

Every `xkb_identify/2` clause can have eight types of predicates in its body. These predicates are `prop/1`, `parm/3`, `parmrange/3`, `parmset/4` and their negations. `prop/2` is used to define Boolean properties. `parm/3` represents multi-valued properties, which can be a menu choice, an atom, or a number as specified in their second argument. `parmrange/3` and `parmset/4` specify that a value must fall within a certain range or a value is a member of a specified set, respectively. The rule can also have something called a "complex condition", which is itself defined in terms of the above mentioned four predicates and their negations. Complex conditions are useful because they allow programmers to take repetitive knowledge and put it in one clause, thereby making rules neater and more manageable.

An XSHELL knowledge base contains clauses for two other predicates: `xkb_question/4` and `xkb_menu/4`. Both predicates contain information that is used to formulate a question as well as generate rule explanations. The first argument of `xkb_question/4` is an atom, which is used by the inference engine to match against the argument of `prop/1` when `xkb_identify/2` is processed. The second argument of `xkb_question/4` is a list, which contains a question asked to the user in order to obtain information about the argument of `prop/1`. The third argument of `xkb_question/4` clause is a short piece of text string stating that the subject of the consultation has the property, while the fourth argument is a text stating that the subject of the consultation does not have the property. The first argument of `xkb_menu/4` is an atom, which is matched against the first argument of `parm/3` when the body of `xkb_identify/2` is processed. The second argument of `xkb_menu/4` is a list which contains a question asked to a user in order to obtain information about the first argument of `parm/3`. The third argument of `xkb_menu/4` is a list of strings each of which states a menu item. The fourth argument of `xkb_menu/4` is a text string which is used by the explanatory facility of the system.

The structure of rules in XSHELL's knowledge base is not far removed from the structure of rules in ImmGnosis' knowledge base because all the above mentioned predicates are used to create ImmGnosis' rules too. An example of an ImmGnosis rule is given in the next chapter.

4.3 User Interface

XSHELL provides a basic text-based interface. The user is required to type in their response to a question displayed on the screen. This interface is Prolog's own user interface with the additional capability of taking predefined inputs from users.

4.4 Explanatory Facility

XSHELL supports user prompted explanations and displays text extracted from the rule used to derive the last conclusion. The third and the fourth arguments of `xkb_question/4` and `xkb_menu/4` are used by the explanatory facility. As mentioned in 4.2, the third argument is an affirmation of the presence of a property, while the fourth argument is an affirmation of absence of a property. In the case of `xkb_menu/4`, the fourth argument is appended to the text string in the third argument that corresponds to the choice made by a user. This is explained further when I discuss the ImmGnosis explanatory facility in the next chapter.

Chapter 5

ImmGnosis System Design

ImmGnosis is a rule-based expert system that deals with immigration laws and was designed and implemented in order to efficiently evaluate the legal merits of a case and reduce the time required for research. ImmGnosis is a Web-based system, which allows multiple users to access the system simultaneously. Depending upon the nature of the sub-domain, ImmGnosis either provides a single diagnostic conclusion or multiple recommendations.

This chapter describes ImmGnosis with respect to the four major components of an expert system and documents the changes made to XSHELL in order to make it useable for ImmGnosis.

5.1 Inference Engine and Procedure

The inference engine of ImmGnosis also uses LPA WIN-Prolog's depth-first backward-chaining inference mechanism. The key difference between XSHELL and ImmGnosis lies in how a query to the expert system proceeds. In XSHELL, a single query to the inference engine leads to a sequential building of the blackboard by searching through the entire knowledge base. In order to handle multiple users, multiple instances of XSHELL are needed since a single instance can support only one user. Hence XSHELL was modified as follows.

The inference process was modified such that when a Prolog query is made to the expert system, the following steps are performed.

1. Contents of the user's portable blackboard are written to the expert system's blackboard.

2. The inference engine processes information in the blackboard and generates appropriate response (another question or a conclusion).

3. Contents of the system's blackboard are wiped from the expert system.

4. The response is displayed to the user.

A call to `stateless_xshell/3` generates a call to five other predicates, culminating in the response of the expert system being displayed to the user. The rest of this section focuses on step 4, while steps 1 to 3 are detailed in Dejaun (2005).

The final predicate called in `stateless_xshell/3` is `generate_file/6`. `generate_file/6` is responsible for generating a new Web page which is displayed to the user as a response to his query.

```
generate_file(Session_ID , flag1 , flag2 , Conclusion_Or_Question_Text ,
                Possible_Answers , Portable_Blackboard)
```

The first argument of `generate_file/6` is a unique Session ID used to identify to whom this Web page will be displayed. The second argument of `generate_file/6` is a flag obtained from `process/5` (Dejaun 2005) indicating whether a conclusion has been reached or more information is needed from the user. The third argument of `generate_file/6` is a flag indicating whether a user is in middle of a consultation or simply viewing a saved consultation. Both of these flags are used to determine the layout of the Web page being generated. If the second argument indicates that more information is needed, then the Web page will have a split screen. A new question and its possible answers are displayed on the left side, and all previously obtained answers are displayed on the right side (Figure 5.1). On the other hand if the second argument indicates that a conclusion has been reached, the Web page has a single screen displaying an option to either save the consultation to the database or proceed for alternate recommendations, depending upon the sub-domain of the consultation (Figure 5.2). The fourth and the fifth arguments of `generate_file/6` are also obtained from `process/5`.

The fourth argument of `generate_file/6` contains the text of the conclusion or the new question. The fifth argument of `generate_file/6` contains the text of possible answers, if applicable. The sixth argument of `generate_file/6` obtained from `wipe_blackboard/2` (DeJaun 2005) contains the most current version of the user's blackboard, which is inserted into the Web page as a "Hidden"[1] control field.

Once `stateless_xshell/3` succeeds, Prolog passes the control back to the Web server which takes over the duty of displaying the newly generated Web page to the user. Prolog is now free to process queries coming from other users.

5.2 Knowledge Base

The domain was broken into smaller sub-domains allowed for development of knowledge bases in parallel, which were later combined to form a complete and comprehensive knowledge base in a relatively short period of time. ImmGnosis currently has over 200 rules in its knowledge base covering four sub-domains as mentioned in Section 2.2.2 (i.e., citizenship status, citizenship by naturalization, admission qualifications for aliens, and visa recommendation).

ImmGnosis provides either a single diagnostic answer or multiple recommendations depending upon the sub-domain. The conclusion provided by the system when dealing with either citizenship status or citizenship by naturalization is a single diagnostic answer indicating whether the user is a citizen or not, or eligible for citizenship or not, respectively. On the other hand for the visa recommendation sub-domain, the system makes multiple recommendations regarding appropriate categories of visas. The distinction lies not in the inference procedure of the system, but in the nature of immigration law.

Table 5.1 summarizes important information related to each sub-domain.

[1]The "Hidden" control field of an HTML form, is neither visible to the user nor can it be interacted with unlike other control fields on a Web page like checkboxes, radio buttons, etc. Authors of a Web page generally use this control type to store information between client/server exchanges that would otherwise be lost due to the stateless nature of HTTP. Since the blackboard is a string of abbreviated attribute/value pairs, displaying it on the screen would serve no purpose except to confuse the user. Hence using a hidden field is an appropriate method.

Table 5.1: Properties of Sub-domains

Sub-domain	**Number of Rules**	**Conclusion Type**	**Unique Conclusion**
Citizenship status	47	Diagnosis	Yes
Citizenship by naturalization	29	Diagnosis	Yes
Admission qualifications for aliens	82	Diagnosis	No
Visa recommendation	55	Recommendation	No

The structure of knowledge base rules is the same for ImmGnosis and XSHELL except for the number of the arguments for `xkb_identify`. Two arguments were added to `xkb_identify/2` of XSHELL making it `xkb_identify/4`. In the ImmGnosis knowledge base, the four place predicate `xkb_identify` defines the knowledge component and has the following structure:

```
xkb_identify(Rule_Number, INA_Statue, Environment, Diagnosis)
```

The first argument is an integer that represents the rule number, which is used by the inference engine to pick out the correct rule for processing. The numbers have to be unique and sequential, though the order of rules in the knowledge base does not matter. The second argument is a list of atoms that refers to the exact Immigration and Nationality Act statue that corresponds to the rule. The third argument is an atom that identifies the sub-domain of the rule. This makes the system's inference procedure more efficient by indexing only the rules pertaining to that particular sub-domain. The final argument is a list of atoms that is used to display appropriate output if the rule succeeds.

Similar to XSHELL, `xkb_identify/4` is also constructed using `prop/1`, `parm/3`, `parmrange/3`, `parmset/4`, and their negations, and it can also have complex conditions in its body. Consequently, any knowledge base written for XSHELL can be accommodated in this system with minor modifications; however, this has not been tested. A benefit of deploying this system over the Web is that the text which goes in all but the first argument of `xkb_question/4` and `xkb_menu/4`, can have HTML tags in them. It is possible to boldface text using the "<b>" tag, etc., or even attach an image or provide hyper-links.

Having complex conditions as a clause in rules was extremely beneficial because every rule in the "citizenship by naturalization" sub-domain has the property of "being admissible". This property is derived from the "admission qualifications for aliens" sub-domain. By putting that property in every rule for naturalization, a module-within-module architecture is achieved where the conclusion from admissibility sub-domain becomes a prerequisite to all the rules in the naturalization sub-domain. Hence an intermediate diagnosis is received from the "admission qualifications for aliens" sub-domain which is then used to derive the final diagnosis. Further discussion about the advantages of complex conditions in ImmGnosis knowledge base can be found in (Dejuan 2005).

Following is a toy rule from the ImmGnosis knowledge base, which is used to make a recommendation for an L1 visa. Owing to the proprietary nature of ImmGnosis, a complete rule cannot be divulged. The `xkb_identify/4` rule states that if the user wishes to obtain a new visa for working purposes as a manager or an executive, the system will give him a recommendation to obtain L1 visa.

```
xkb_identify(1,'INA101(a)(15)(L)',visa,[l1]) :-
        parm(purpose_of_consultation,m,1),           % new visa
        prop(executive_or_manager).
```

```
xkb_menu(purpose_of_consultation,
         'What is the purpose of your consultation today?',
        ['To Obtain a new Visa',
         'Sponsor Visa for prospective employee',
         'Change the status of current Visa', '],
         'Purpose of current consultation: ').
```

```
xkb_question(executive_or_manager,
        ['Did you hold an executive or manager position? '],
         'You held an executive or manager position.',
         'You don't hold an executive or manager position.').
```

Note that `parm/3` has a corresponding `xkb_menu/4` while `prop/1` has a corresponding `xkb_question/4`. If the rule succeeds, then the fourth argument of `xkb_menu/4` will be appended to the appropriate choice from its third argument and used by the explanatory facility. The explanatory facility will also take appropriate text from `xkb_question/4` and display it to provide complete justification for the conclusion reached.

5.3 User Interface

The user interface can be divided into two parts, the expert system interface and the interface which is needed for the proper functioning of a Website.

5.3.1 Expert System Interface

Upon accessing the ImmGnosis home page, a user can conduct a consultation by either initiating an anonymous consultation or by logging in to the system with an existing account. Since ImmGnosis is a Web-based expert system designed for use by the general public, we present the user with a familiar Web interface. The consultation window has two sections (Figure 5.1). The section on the left displays the current question and possible answers, which can be selected from a drop down list or radio buttons. The section on the right displays all the answers provided by the user up to that point. If a user wants to change an answer to a previous question, he or she can use the browser's "back" button to navigate to that question and alter its answer.

Once the conclusion is reached, the recommendation will be displayed in the right section just below the history of answers provided by the user. At this point, the user has the option of saving the consultation or permanently deleting it from the system. The "admissibility" and "visa recommendation" sub-domains have the capability of providing multiple recommendations. For example, a user might be eligible for an H1-B visa and also for an L1 visa. In such a situation the expert system provides the user with the option of either accepting the first recommendation or further continuing their consultation with the aim of getting multiple recommendations. If the user chooses to get multiple recommendations then all the recommendations are written at the bottom right of the screen as shown in Figure 5.1

An advantage of delivering the system over the Web is that the knowledge engineer can use any tag set allowed within the body element of a HTML document to appear in expert system output.[2] This is especially useful in the explanatory facility where words can be highlighted, underlined or bold faced for emphases. Complete URL's linking to other documents can also appear in expert system output and can be used to link to definitions of a concept or a term.

5.3.2 Website Interface

Since this expert system is delivered over the Web, it includes many functionalities which are pertinent to the proper functioning of a Web-site. A user can login to the Web-site and either choose to start a new consultation or view or edit an existing consultation.

Upon reaching a recommendation, the system provides the user with the option of saving the current consultation or deleting it. If the user is already logged in and wants to save the consultation, then it is saved under his or her account. Otherwise, he or she is given the

[2]An expert system "output" is not limited to its conclusions but includes any text displayed to user.

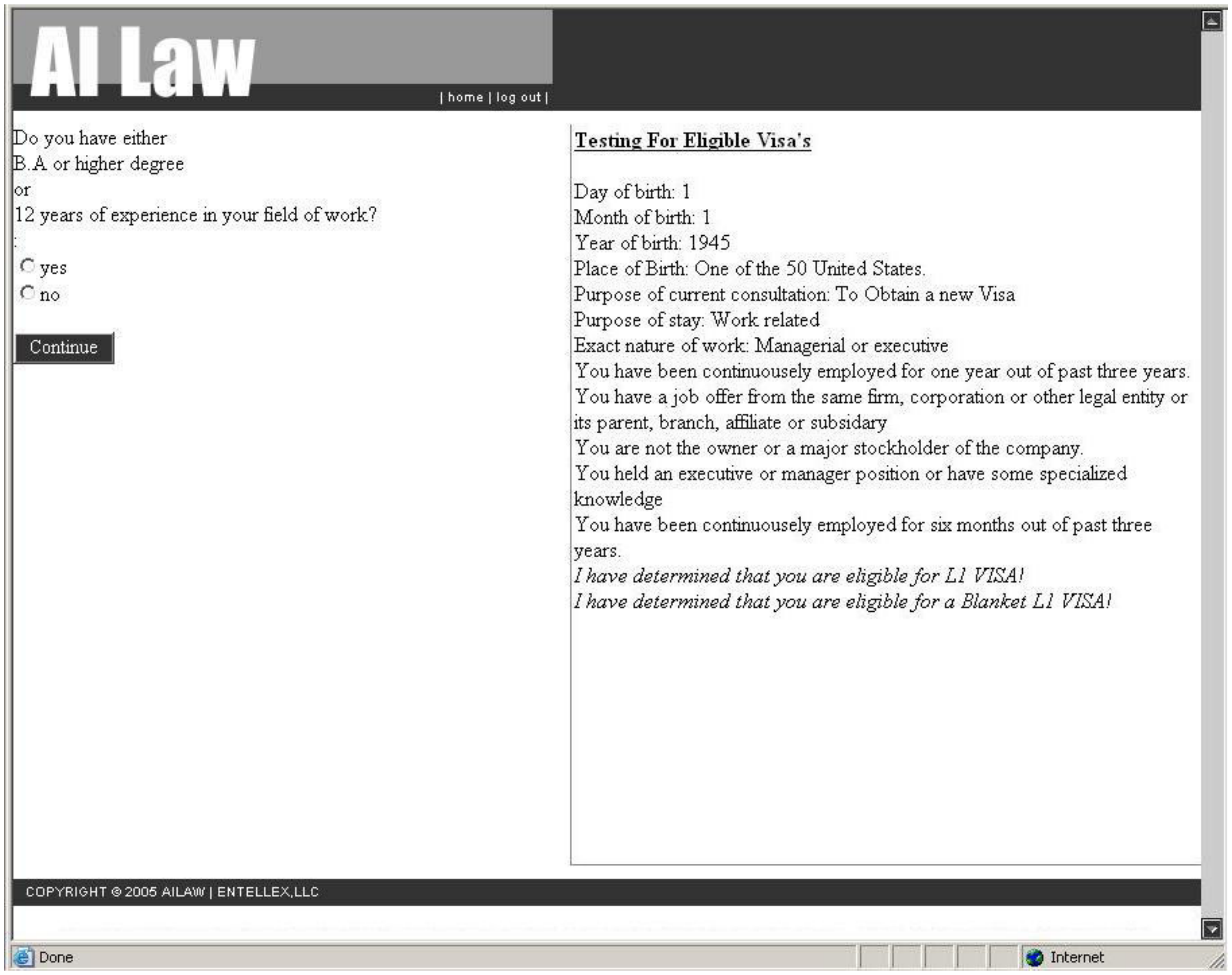
option of creating a new account. In subsequent visits, the user can view, edit, or delete any of his or her previous consultations.

Currently there are two types of roles that a user can possess: a regular user or an administrator. An administrator level user has the option of viewing, editing, and deleting all saved consultations available on the system from all users. The administrator also has the power to start, stop, and restart the expert system. An administrator can also initiate a special verbose mode of consultation where various system parameters are displayed during the consultation. This is specially helpful in debugging the knowledge base and inference engine. The parameters displayed in the verbose mode are

1. session ID assigned to the consultation;
2. the last successfully processed blackboard of this consultation;
3. the new unprocessed blackboard with the new answer; and
4. flags from Prolog indicating whether the new blackboard has been processed successfully or not.

5.4 Explanatory Facility

The explanatory facility of ImmGnosis is similar to the explanatory facility of XSHELL in respect to how the displayed texts are formulated from the Prolog rules in the knowledge base as detailed in Section 4.4. There are two major differences between the explanation provided by XSHELL and ImmGnosis. First, in XSHELL, a user needs to explicitly request an explanation of how the conclusion is reached at the end of the consultation, while ImmGnosis automatically displays the explanation in chronological order as the consultation proceeds. Second, in XSHELL, only the facts that contributed to reaching the final conclusion are displayed, while ImmGnosis displays all the facts collected during the consultation. The advantage of the continuous explanatory feature of ImmGnosis is that the users can see their previous responses during the consultation and can make necessary changes in the middle of the session as discussed in Section 5.4.1. The disadvantage of the explanation facility of ImmGnosis is that it does not show which facts were actually relevant for reaching the final conclusion.

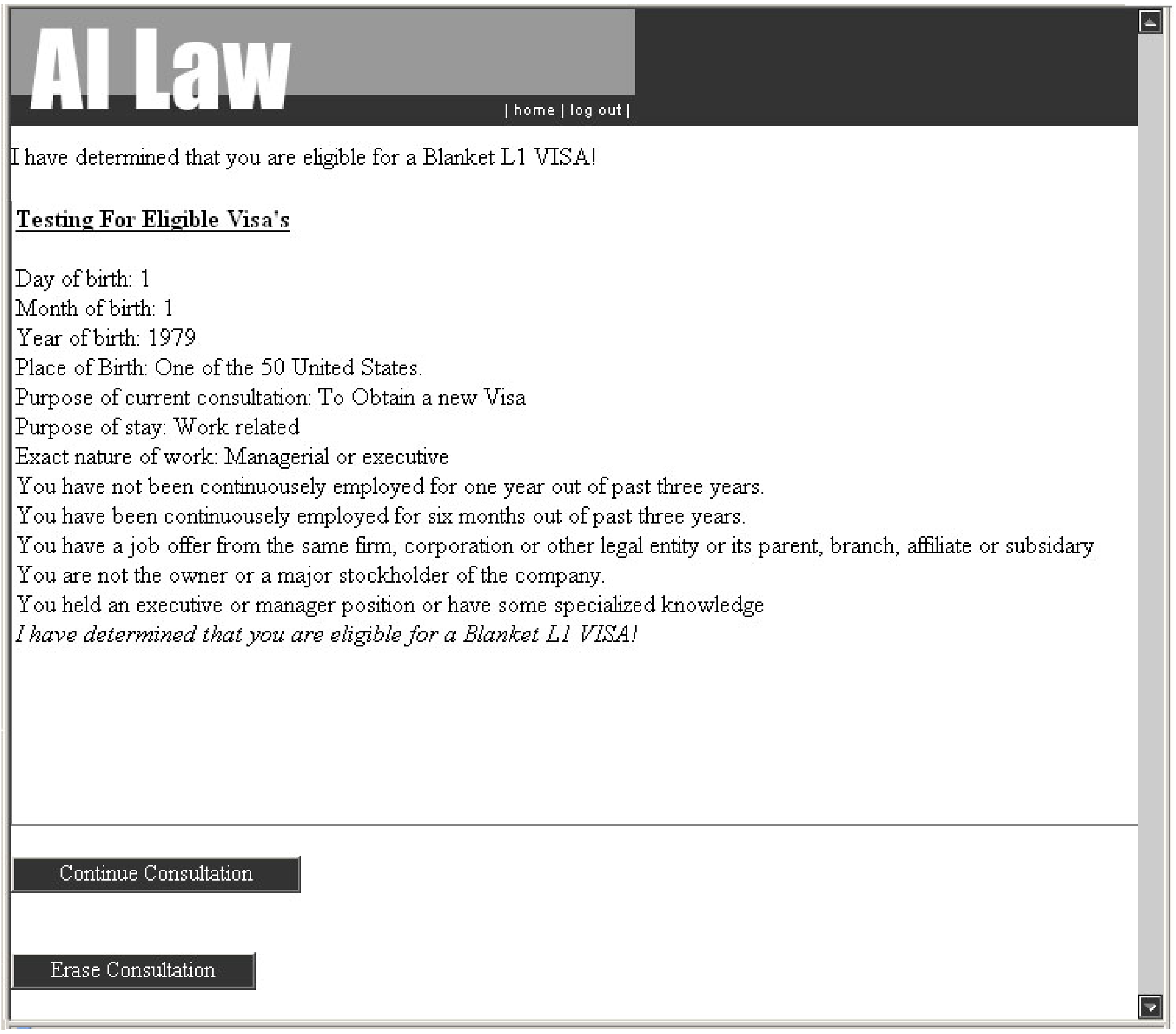


Do you have either
B.A or higher degree
or
12 years of experience in your field of work?
:
yes
no

Testing For Eligible Visa's

Day of birth: 1
Month of birth: 1
Year of birth: 1945
Place of Birth: One of the 50 United States.
Purpose of current consultation: To Obtain a new Visa
Purpose of stay: Work related
Exact nature of work: Managerial or executive
You have been continuously employed for one year out of past three years.
You have a job offer from the same firm, corporation or other legal entity or its parent, branch, affiliate or subsidary
You are not the owner or a major stockholder of the company.
You held an executive or manager position or have some specialized knowledge
You have been continuously employed for six months out of past three years.
I have determined that you are eligible for L1 VISA!
I have determined that you are eligible for a Blanket L1 VISA!

Figure 5.1: User Interface



I have determined that you are eligible for a Blanket L1 VISA!

Testing For Eligible Visa's

Day of birth: 1
Month of birth: 1
Year of birth: 1979
Place of Birth: One of the 50 United States.
Purpose of current consultation: To Obtain a new Visa
Purpose of stay: Work related
Exact nature of work: Managerial or executive
You have not been continuousely employed for one year out of past three years.
You have been continuousely employed for six months out of past three years.
You have a job offer from the same firm, corporation or other legal entity or its parent, branch, affiliate or subsidary
You are not the owner or a major stockholder of the company.
You held an executive or manager position or have some specialized knowledge
I have determined that you are eligible for a Blanket L1 VISA!

Figure 5.2: Multiple Recommendations View

Chapter 6

Conclusion

I now conclude this thesis by providing the results of testing and identifying areas of future development, some of which are already in development stage.

6.1 Evaluation

ImmGnosis is evaluated on three criteria: correctness, completeness, and the usefulness of the system. At present, ImmGnosis has not undergone intensive evaluation regarding the accuracy of its diagnoses owing to the propriety nature of the system. So far, only the experts involved in the knowledge acquisition process have validated the rule base throughout its development by reading the rules and running the system. ImmGnosis passes the correctness criteria so far but a larger test base of users is needed before making any objective claims. ImmGnosis' knowledge bases are complete in respect to their sub-domains except for the "visa recommendation" sub-domain. With respect to United States immigration law, completeness is a matter of investing more resources. The system has been used by the experts to evaluate a few real world cases pertaining to the citizenship status of their clients. According to the experts, the system made accurate diagnosis and in far less time than it would have taken the experts themselves to reach the same conclusion, hence demonstrating the useful nature of the system.

The performance of the modified expert system shell and its stateless architecture has been extensively tested. The stateless architecture coupled with the portable blackboard has proven to be robust and lived up to its expectations of handling multiple users while using minimum machine resources. It can be stated confidently that this shell can be used to develop other expert systems, too.

6.2 Future Development

The primary area of development is to complete the visa recommendation sub-domain followed by adding more sub-domains to the existing system and making it complete with respect to United States immigration law. Using the current sub-domains, it has been proved that the system is capable of reasoning and providing accurate recommendations or diagnoses. Adding more sub-domains is a matter of investing more time in the knowledge acquisition process.

In the sub-domains where multiple recommendations are sought, these recommendations are not ranked or displayed with a confidence value. Since the inference engine tries to satisfy the rules in lexical order, the rules with the higher number are attempted first, hence often solved first. There are situations where the later recommendations may be more appropriate but the user may assume that the first recommendation is the best. This problem can be solved using traditional methods such as MYCIN style certainty values, Dempster-Schafer Theory of Belief Functions, or Fuzzy Logic. I would like to take a different approach and get users to attach a “desirability value” to different attributes. For example, for an alien who wants to work in the United States, an L1 visa is easier to process and obtain than an H1-B visa, but the maximum duration of stay on an L1 is only one year compared to three years for a H1-B. A user could be asked to identify what is more important to them, ease of obtaining a visa or the maximum duration of stay on a visa. These values coupled with an expert’s opinion on ease of processing could influence our ranking methodology in multiple recommendations.

Another shortcoming that became apparent during the parallel development of knowledge bases was the repetition of knowledge structures. Since the knowledge base for sub-domains were developed independently, there were a few concepts which were repeated. For example, in the “Citizenship status” sub-domain, a user is asked about their parent’s citizenship status and then asked again in the “Citizenship by naturalization” sub-domain, albeit in different words. This problem exists due to lack of communication between knowledge base developers and through not having proper naming conventions. Though this does not effect the correctness or completeness criteria of an expert system, asking the same question twice seems rather unintelligent for an intelligent system. This problem can be solved by developing an ontology for the system and identifying relationships between objects so the domain knowledge is appropriately encoded and reused. The development of such an ontology is currently under way.

A rule-based expert system (ImmGnosis) that deals with United State immigration laws was designed and implemented in order to efficiently evaluate the legal merits of a case and reduce the time required for research. An existing Prolog expert system shell (XSHELL) was modified such that it would allow multiple users to access the system simultaneously through a Web interface with minimum server side load by incorporating a portable blackboard in a stateless architecture. The system is capable of determining a user’s U.S. citizenship status, admissibility status for an alien visitor, and eligibility for naturalization as well as giving multiple visa recommendations. ImmGnosis was evaluated on three criteria: correctness, completeness, and the practicality of the system. The results of the tests indicate that the diagnoses and recommendations made by the system are correct, and the knowledge base is complete as mentioned. In addition, it has been shown that the system can make an accurate diagnosis in far less amount of time when dealing with real cases pertaining to citizenship status; hence demonstrating the practical nature of the system.

References

Awad, E. M. (1996) *Building Expert Systems: Principles Procedures, and Applications.* St. Paul: West Publishing Company.

Brown, S.; Burdick, R.; Falkner, J.; Galbraith, B.; and Johnson, R. (2001) *Professional JSP (2nd ed.)* Birmingham: Wrox Press.

Covington, M. A.; Nute, D.; and Vellino, A. (1997) *Prolog Programming in Depth.* New Jersey: Prentice-Hall Inc.

DeJuan, D. M. (2005) *ImmGnosis: Knowledge Engineering for A Stateless Web-based Expert System for Immigration Law.* M. S. thesis, University Of Georgia.

DeJuan, D. M.; Khosla, V.; Potter, W. D.; Dorminey, B.; and Nute, D. (2005) ImmGnosis: A Stateless Web-based Expert System for Immigration Law. *Proceedings of the 2005 International Conference on Artificial Intelligence-1*, 249-255.

Buchanan, B. G.; Barstow, D.; Bechtal, R.; Bennett, J; Clancey, W.; Kulikowski, C.; Mitchell, T.; and Waterman, D. A. (1983) Constructing an Expert System. In Hayes-Roth, Fredrick; Waterman, A. Donald; and Lenat, B. Douglas, eds., *Building Expert Systems*, pp. 127–167. Reading, M.A.: Addison-Wesley Publishing Company.

Jackson, P. (1999) *Introduction to Expert Systems.* 3rd ed. Harlow, England: Addison Wesley Longman Limited.

Jennings, D. (2002) *JXSHELL: a Web-based expert system platform.* M. S. thesis, University Of Georgia.

McCarty, L. T. (1977) Reflections on Taxman: An Experiment in Artificial Intelligence and Legal Reasoning. *Harvard Law Review* 90.5: 837–893.

McPherson, S. (2000, April 2000) Java Server Pages: A Developer's Perspective. Sun Microsystems. Web: http://java.sun.com/developer/technicalArticles/Programming/jsp/

Popp, W. G., and Schlink, B. (1975) Judith, a computer program to advise lawyers in reasoning a case. *Jurimetrics Journal* 15.4: 303–314.

Popple, J. (1993) *Shyster: A pragmatic legal expert system.* The Australian National University, Canberra.

Westwood, A., and Steel, B. D. (2004) LPA Win-Prolog 4.500 Intelligence Server. Logic Programming Associates. Web: http://www.lpa.co.uk/ftp/4500/int_ref.pdf