

A WWW-Based Expert System Advisor for the Diagnostic of Network Communication Problems

Anibal Morales, Antonio Creus, Juan Pablo Carvajal

Department of Electrical and Computer Engineering
University of Puerto Rico at Mayagüez
Mayagüez, PR 00680

anibal@exodo.upr.clu.edu, acreus@exodo.upr.clu.edu, juanpc@exodo.upr.clu.edu

Abstract

Internet users who encounter network communication problems sometimes do not have the experience or expertise available on site to approach and solve those problems. NeXPert is a WWW-based expert system advisor developed with the intention of providing access to expert knowledge to a broad user community. It provides advisory knowledge to help users solve these problems. It was implemented with freely available tools such as CLIPS, JESS and Java. The WWW-based design proves the feasibility of developing expert systems in a distributed environment and gives access to the knowledge base to the broad Internet community.

1. Introduction

On a daily basis, users connected to the Internet confront network communication problems. Most of the time these problems are common and easy to solve for experienced system administrators. On the other hand, they tend to be a real nuisance to inexperienced ones. As a consequence, there is the need for expert knowledge that could assist inexperienced system administrators and end users in approaching and solving such problems. Expert systems running on servers can support a large group of users who communicate with the system over the network. Using WWW browsers users will not need special hardware or software to consult these services. NeXPert is created with that purpose in mind. It provides advising based on a diagnostic knowledge base of network communication problems.

2. Problem Domain

The problem domain targeted by NeXPert is a typical network of inter-connected computers that looks like the diagram in Figure 1. Computers in this network are using the TCP/IP protocol to connect to the rest of the world. The situation modeled is a typical situation encountered by system and network administrators abroad managing workgroups of computers connected to the Internet. In this environment users are constantly accessing popular Internet services such as the World Wide Web, FTP, Gopher, E-Mail, and other services. If a problem occurs somewhere along the path of communication between the user computer and a target Internet server somewhere on the world, the user will encounter a communication problem that the application software will promptly inform as a communication error. The user will take one of two possible courses of actions. Either the user has the technical knowledge and it will try troubleshooting steps to isolate the experience, he/she will call the administrator right away. We target both situations by providing NeXPert as an expert system that can be used by both the end user and the administrator from their own locations at the same time.

As we can see in Figure 1, we presumably have a Local Area Network (LAN) of computers connected by a shared Ethernet connected with a Hub. This LAN is in turn connected to a Router, which may act as an arbitrator between different network segments. For simplification, we will assume only our local segment. The Router is connected via a Wide Area Network (WAN) link to the Internet (the cloud of Figure 1). We can also identify define the Network Access Point (NAP) as the last step along the path to the Internet to aid in the troubleshooting. A NAP is a place where different communication companies (providers) agree to locate their equipment in order to centralize and organize the backbone connection across the Internet. The NAP is the connection point where the WAN link of our network (or our service provider's network) connects with the backbones of the Internet.

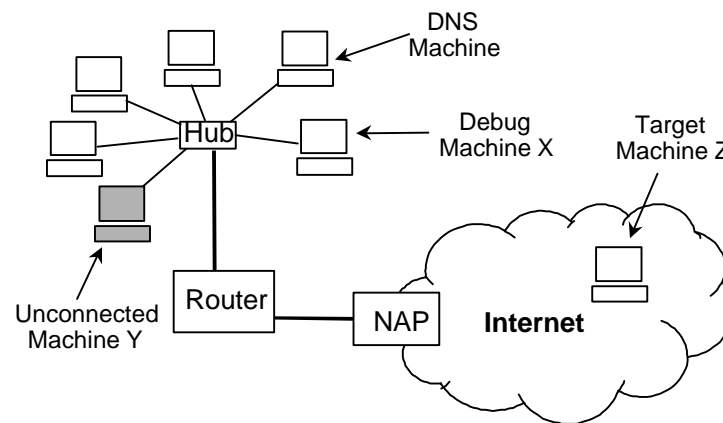


Figure 1: Simplified Network Model

A typical situation will be a user machine, called Machine Y, trying to connect to a Machine Z located somewhere on the Internet, for example a Web server with a certain home page. As part of the troubleshooting procedure, the user or administrator will probably make use of another computer as a type of “control” in an experiment. This machine will be Machine X. When the end user or administrator starts the troubleshooting procedure, most probably, he/she will use the PING utility to query the target address of the machines and communication equipment along the way. The PING utility uses the Internet Control Message Protocol (ICMP) as a kind of “sonar” that returns a message that confirms the target machine responds to the query. The PING utility is a vital aid in the troubleshooting of these systems, which is why NeXPert will ask the user of the system to utilize it as well. This environment also includes a Domain Name Server (DNS) machine who will map machine names with Internet Protocol (IP) addresses. The DNS machine is an integral part in network applications such as the World Wide Web.

3. Conceptual Model

Knowledge in an expert system can be represented in several ways such as production rules, semantic networks, frames, etc. Trees can be used to represent knowledge and classify objects if the problem domain is hierarchical in nature. In this type of domain higher level alternatives at the top nodes are examined first and then a narrowing process begins until an answer is found. A decision tree could be both a knowledge representation and a reasoning method. We use this approach for its advantages of being very efficient, compact, and relatively easy to implement.

3.1 Decision Trees

A decision tree is composed of nodes, leaves and branches. Inside the nodes some decision occurs that transfer control via any of its branches to other nodes or leaves [Giarratano, 93]. A binary decision tree is one where each node of the tree has only two transition branches. Typically, these binary trees are used to implement knowledge systems where you answer yes/no questions. The hierarchical nature of network communication problems adapts very naturally to binary decision trees. Usually the heuristics dictate that one problem leads to consequences that in turn lead to other problems and that eventually leads to a cause and a reasonable guess or solution to the problem. We use the approach of [Giarratano, 93] and call the nodes *decision nodes* and the leaves *answer nodes*. A decision node has associated with it a question. An answer node has associated with it an answer. All the nodes can be identified with labels in order to keep track of tree traversals and ease the implementation. A binary tree with these characteristics will look like the one shown in Figure 2.

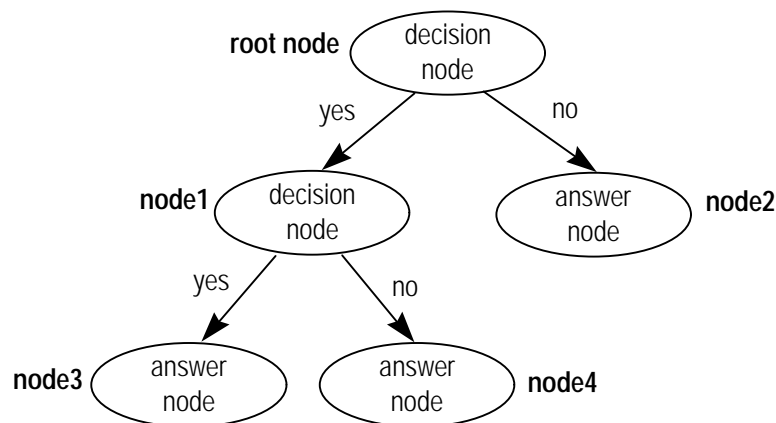


Figure 2: The Decision Tree Structure

The inference process in decision trees starts by setting an initial location, typically the root node, and following the yes/no transitions until we arrive at an answer node. If the current location is a decision node then the question associated with that node is presented. If the question is answered with a yes then the current location is set to the child node associated with the yes branch. If this location is a decision node then the process is repeated. If it is an answer node then the process comes to an end and the answer is presented. To begin the process again the current location is set to the root node.

4. NeXPert Development

NeXPert development process followed the linear life cycle model. This life cycle consists of four major areas: Planning, Knowledge Definition, Coding and Evaluating the Expert System.

4.1 Planning

NeXPert is an expert system designed for the main purpose of aiding system administrators with daily network problems. Several factors were considered as to create a useful expert system tool. These factors involved:

1. Provide this tool to as many people as possible at the same time, specially to inexperienced system administrators as well as end users.
2. Provide a tool that could be expanded and/or modified with ease as to provide the necessary information to specific problems, in this case the network troubleshooting domain.
3. Provide the capabilities of learning new knowledge while interacting with the user/expert.
4. Use tools and/or languages that provide for the implementation of expert systems with portability and extensibility capabilities.
5. Use tools that are available with little or no cost. Use tools and/or languages that are common to the daily programmer. Use tools and/or languages that are in the process of growing and that will not become obsolete in the near future.
6. Test the feasibility of such a system by researching available tools and conducting an informal product-need research.

These issues provided the necessary information for deciding which tools and programming languages were needed for the creation of NeXPert. NeXPert is designed and developed as a Java applet. The use of the Java Expert System Shell (JESS) was necessary for the integration of the knowledge base and inference engine provided by CLIPS, with the advantages of being Web-based.

4.2 Knowledge Definition

NeXPert's knowledge definition was divided into knowledge elicitation and knowledge representation. The knowledge elicitation process was based on gathering information mainly from articles, books and interviews with network domain experts. These experts were gathered from the Electrical/Computer Department and the Campus Computer Center from the University of Puerto Rico, Mayagüez Campus. The representation structure selected was the decision tree since the domain problem followed a highly hierarchical structure.

4.3 Coding

We began the coding using CLIPS 6.05 for developing the rules and facts that make up the knowledge base. These rules and facts were created following Giarratano and Riley's [Giarratano 93] suggested structure for an expert system using a decision tree with learning capabilities. After implementing this we found inconsistencies in the restructuring of a fact when learning new knowledge. Once the existing fact was modified to include new branches from a particular node, some of the slots in the fact were not modified leaving inconsistencies in the structure of the knowledge base. We fixed this problem and modified the structure of facts to transcend from a binary decision tree to a decision tree with multiple branches, meaning multiple answers.

4.4 Evaluation and Verification

This process consisted on running a series of test scenarios and comparing the results to the answer of domain experts. Initially, fifteen scenarios were created. From these scenarios, only two showed inconsistencies with the conceptual model. The conceptual model was modified to correct the inconsistencies detected and the same fifteen scenarios were tested again. In the second stage, both NeXPert and the domain experts reached the same solutions for all the scenarios presented.

5. Expert System Paradigm

NeXPert differs a little from the classical expert system paradigm by the use of the Internet as the communication channel between the User Interface and the other modules of the Expert System architecture. The integration of the different modules is shown in Figure 3.

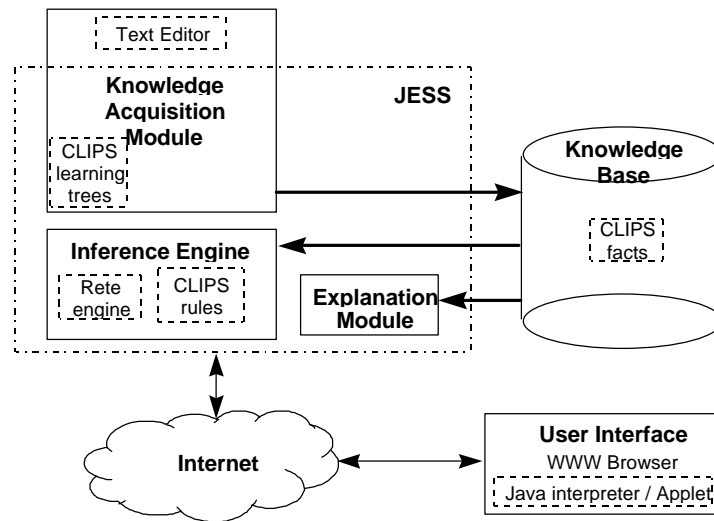


Figure 3: NeXPert Expert System Model

6. Implementation

NeXPert was implemented to be a second generation expert system like other Diagnostic Expert Systems such as MYCIN, etc. It has all the characteristics to narrow the possible causes of a problem by asking a series of questions to the user. An early prototype was developed to test the knowledge gathered from the experts. After corroborating the validity of the data gathered a new approach was designed. Our first design used rules to express questions and possible answers. This design limited the flexibility for future growth since new knowledge meant writing new rules. A new structure was developed using a generic frame for handling decisions and answers without having any dependency on the content of the question or answer. As a result, the questions and answers were created as facts in NeXPert's domain of knowledge. The format in which the facts are presented are called nodes. These nodes have the necessary information to know where they connect from and to in the decision tree.

We implemented the decision tree concept with several rules that decide whether a decision needs to be made, decide if a valid answer has been reached, decide if the user input is valid or not, and if new knowledge needs to be incorporated to the knowledge base. Due to limitations in the implementation of JESS we cannot write to the knowledge base from a Java applet. For this reason the learning process can only be performed through the use of CLIPS alone. The *save-facts* function used for this purpose in the CLIPS rules is one of the many routines not yet implemented in JESS.

7. Conclusion

Our work has led to the development of a WWW-based expert system which exhibits an ability compared to the human domain experts giving advice as to the cause or probable cause of network communication problems. We succeeded in integrating the tools JESS, CLIPS, Java, and the WWW to provide wide access to the tool. The NeXPert home page is located at the location <<http://exodo.upr.clu.edu/~anibal/nexpert>>

8. Future Work

We have identified three areas of improvement for NeXPert: more complete coverage of the networks Domain Problem, changes in the design of NeXPert and issues with the language implementation. A future version of the NeXPert could implement real connection to the live network in order to obtain itself the answers to some of the questions, for example: *The machine responds to a Ping?* could be easily answered by the expert system generating that ping itself. Also, if the user is able to connect to the NeXPert system via the WWW, then some deductions could be made about the ability of the user's machine to connect at least to some sites. Real-time network monitoring that provide advice by analyzing network traffic, making it a Network Management expert system. A Network Design [Ceri, 90] module could also be implemented for completeness. Also, the inclusion of knowledge about network systems more complex than the one presented in this project is feasible, as for example networks that include multiple segments, multiple routers, dial-up communication servers, combinations between shared media and switched media, other types of networks beyond Ethernet, other topologies, other protocols, etc.

The design of NeXPert could be improved in several ways. Our implementation of decision trees has the foundations laid for multiple branch trees by including a list of the accepted answers. The CLIPS code can be expanded to look for a list of answers and branch to different nodes. We believe that the utilization of Object-Oriented techniques such as the CLIPS object oriented language (COOL) could lead to better maintenance of the code. JESS does not support this feature yet but since it is a continuing development effort we think it is going to be implemented in the near future.

9. References

- [Ceri, 90] Ceri, S.; Tanca, L. *Expert Design of Local Area Networks*. IEEE Expert, Vol. 5, No. 5, 1990. PP 23-33.
- [Eriksson, 96] Eriksson, H. *Expert Systems as Knowledge Servers*. IEEE Expert, Vol. 11, No. 3, 1996.
- [Giarratano, 93] Giarratano, J.; Riley, G. *Expert Systems. Principles and Programming*. Second Edition. PWS Publishing Company. 1994. Ch 1, 7, 12.
- [Riley, 93] Riley, Gary. B. Donnell et. al., CLIPS Reference Manual, JSC-25012, Lyndon B. Johnson Space Center, Houston, Texas. June 1993.
- [Wick, 89] Wick, M.R.; Slagle, J.R. *An Explanation Facility for Today's Expert Systems*. IEEE Expert, Vol. 4, No. 1, 1989. Pp 26-36.