Kuwait J. Sci. Eng. **37** (2A) pp. 31-41, 2010

On embedded languages in expert systems for troubleshooting wireless connectivity

MOHAMMED ALMULLA

Dept. of Computer Science, Kuwait University, P.O. Box 5969, Safat 13060, Kuwait.
Email: m.almulla@ku.edu.kw

ABSTRACT

Troubleshooting wireless network connectivity remotely significantly reduces network helpdesk costs and operational expenditures related to the wireless network by practically eliminating the need to travel onsite to troubleshoot wireless network issues. These costs savings can be achieved through several troubleshooting tools. In this paper we propose a connectivity troubleshooting tool (expert system) for all organizational staff, allowing them to perform expert analysis on wireless client connectivity issues and quickly resolve or escalate trouble tickets without requiring extensive wireless expertise. The suggested system is implemented using embedded languages from logic as well as object-oriented programming paradigms, in order to benefit from various features of both paradigms such as security, portability, inheritance, reasoning, and answer justification.

Keywords: Embedded Languages; Expert Systems; Troubleshooting Wireless Network Connectivity.

INTRODUCTION

An Expert System (ES) is defined as an intelligent computer program that applies reasoning methodologies or the knowledge in a specific domain to render advice or recommendations (Abdul-Rahim, 2005). Such a system can help individuals deal with their daily problems without having to reference complicated manuals for problem solving, or having to wait for a troubleshooting expert. The targeted users of such systems can be those who are eager to solve problems themselves, or those who wish to understand where the problem lies or how it can be solved. Such systems clearly save time, effort, and costs.

This paper is about designing and implementing an expert system for troubleshooting problems that might arise during wireless network connectivity in personal mobile computers and devices. The majority of people connect to a

wireless network at least once a day either at home, the office or any public location that offers a wireless internet service. This means there is a significant chance for a connection problem to occur. Running an expert system that directs the user to the source of the problem is of great importance; even if users are unable to solve the problem, they can at least gain some insight into the nature of the problem and be more aware when seeking professional assistance.

Client connectivity issues can be caused by a variety of reasons, many of which are not related to the wireless network (Spyrou *et al.,* 1999). Unfortunately, the wireless network often have connectivity problem experienced by mobile users, and wireless network support staff is then required to devote time troubleshooting the issues, which may not be a wireless problem at all. Thus the client connectivity troubleshooting tool is designed to assist users with limited wireless networking expertise easily identify the exact connectivity problem, allowing them to either resolve it or call upon appropriate IT support staff. The client connectivity troubleshooting module's sophisticated analysis engine will quickly identify device level problems, wireless network health, wireless network availability, wireless network or client configuration, and wired network connectivity issues.

Wireless applications rely on configuration of both wireless and wired network elements to function correctly, and a simple change to the wired network could render wireless applications inoperable. Because network administrators cannot connect to the wireless network to perform the tests required to identify where the problem occurred, troubleshooting can be cumbersome and time-consuming. The Access Point Connectivity Testing segment addresses this issue by asking the user to go though a step-by-step testing procedure to ensure that the device is functioning correctly. By utilizing the radio of the wireless sensor to simulate a wireless client station, true end-to-end network testing can verify all aspects of the wireless application's data path. Connectivity tests can be customized to verify the specific wireless configuration, wired network configuration, and application server availability. These tests can be configured to run automatically on a pre-configured schedule or run on demand to identify and notify configuration changes which impact wireless applications.

APPLICABILITY, ADVANTAGES AND DISADVANTAGES

Large organizations and/or companies can distribute the troubleshooting expert system among their employees to reduce IT costs and to improve operational

efficiency. In addition, such a system can be bundled with certain hardware/software products (e.g. wireless routers) as a method of offering basic support or simply distributed publicly over a certain medium (e.g. web-based, DVD, etc).

Like all other systems, expert systems have advantages and disadvantages. The advantages include (1) 24-hour availability (2) multimedia elements make the system more accessible and user-friendly (3) expert systems provide explanation on how problems were caused or solved (while human solvers might not deliver a clear cause of the problem or how it was solved) (4) knowledge in an expert system is permanent and scalable (Phan *et al.,* 2002), while knowledge of a human expert is not (5) scalability allows the system to be easily modified to handle more complex problems, and lastly, (6) cross-platform portability adds to the advantages of the expert system. On the other hand, the disadvantages of such a system may include (1) expert systems cannot automatically adapt to changing environment as they must be explicitly updated (2) the cost of regular maintenance (3) A limited scope of the expert system.

PREVIOUS WORK

Malakooti & Tsurushima (1989) argue that despite the quantitative nature of expert systems, the ability to handle multiple conflicting goals might resemble experts' cognitive treatment of subjective and uncertain preferences. However, there have been previous attempts at designing expert systems that can handle complex tasks. For example, Heragu and Kusiak (1990) present a Knowledge-based Machine Layout (KBML) system that is capable of solving relatively larger problems. It employs both quantitative and qualitative data. However, the crisp nature of data means it cannot adequately capture subjective and uncertain dynamics of the problem domain. Furthermore, conflicting preferences require user intervention. KBML employs various models and algorithms, each of which is suitable to some specific scenario, with a hope that a collection of models would cover most of the scenarios. Other systems focus on one model that possibly fits all scenarios. Our proposed system belongs to this group. It uses an intuitive graphical user interface aided by informative images and simple questions with only yes and no answers, as shown in Fig. 1.

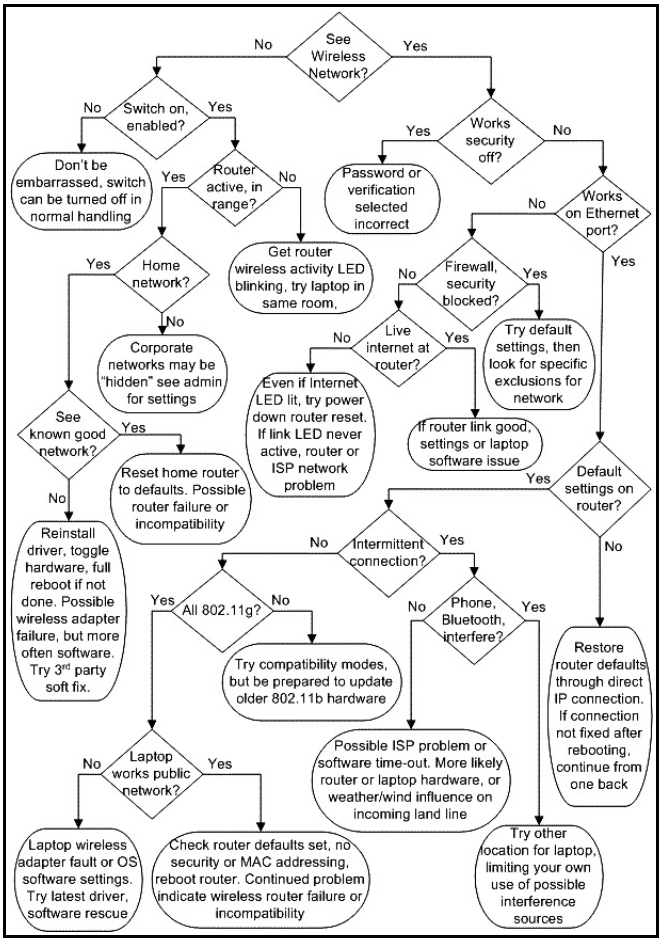

Fig. 1: Rules of the Troubleshooting Expert System

There have been attempts at designing computer-problems troubleshooting expert systems due to the predictable nature of computer troubleshooting and because of the ready availability of precise expert knowledge. One such commercial system is the Sniffer Wireless Expert system which analyzes 802.11a and 802.11b frames all the way from the physical network layer to the application network layer (Teşekkür, 2002). Expert analysis provides greater visibility into network anomalies and facilitates automatic problem solving. The AirDefense Advanced Troubleshooting module is another industry's toolset to perform expert analysis of wireless connectivity issues and perform end-to-end network testing from the wireless perspective (Sinha *et al.,* 2006). Unfortunately, not all of these attempts have proved to be successful due to various reasons. For example, there is the problem of using simple user interfaces such as command lines or pages of text that tend to confuse the users instead of helping

them. This is particularly important to users with limited computer troubleshooting knowledge who aren't concerned about the details. Secondly, these systems have difficulty pursuing directed goals. This is evident in the Windows troubleshooter for example, where a fixed set of solutions are presented to the user and the system gives up when the user has exhausted all possible solutions. In contrast, our system asks directed questions and uses a logical elimination process to reach a specific solution. Lastly, the lack of scalability and the fixed nature of many of these systems make them hard to update and expand. Conversely, this newly proposed system is easily extendable and can be modified to suit any problem that could be modeled as a flowchart.

FUNCTIONAL SYSTEM DESCRIPTION

A traditional ES has five basic components, namely a Knowledge Acquisition Module, a Knowledge Base, an Inference Engine, an Explanation Facility, and an Interactive User Interface (Negnevitsky, 2002). Upon running the Wireless Troubleshooting Expert System, Java will run first to initialize the system and to draw the graphical user interface. Next, Using JPL, the Java VM calls Prolog and asks it to consult the roles prolog file, a file containing the roles by which a solution can be obtained from certain answers. The first question is displayed on the screen, and the system waits for the user to pick an answer (yes or no).

The user's answer is written to another prolog file called "Answers", which will hold user's answers to the questions asked so far. These answers are later used to determine what questions to ask next, what solutions to propose, and to explain the rationale behind these solutions. A sample screenshot demonstrating a question posed by the system is shown in Fig. 2.

Fig. 2. Sample Screenshot of the Troubleshooting Expert System

Prolog will then consult the "Answers" file, and using the knowledge from the rules file, it is able to find either the next question to ask, or the solution if it is reached. Please note that the expert system will keep asking questions, processing them in Prolog queries, until it reaches a solution, at which point the solution is displayed. If a solution is found, the users can accept the solution and close the program, or can ask the system to explain the solution and to provide a list of answers that lead to this conclusion, or to reset the process from the start. Fig. 3 illustrates the processing flowchart of the system.

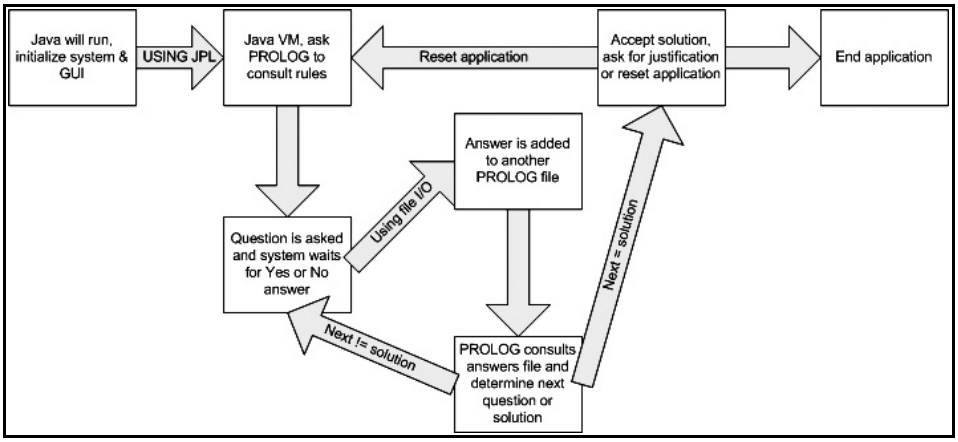

Fig. 3. System Processing Flowchart.

IMPLEMENTATION ISSUES

Java is chosen as the programming language for network computers (NC) and has been perceived as a universal front-end for the enterprise database. Java is considered a much simpler and easier-to-use object-oriented programming language when compared to C++. Java has replaced the complexity of multiple inheritance in C++ with a simple structure called interface, and also has eliminated the use of pointers. Java uses automatic memory allocation and garbage collection, whereas C++ requires the programmer to manually allocate memory and to collect garbage. Additionally, the number of language constructs in Java is relatively small for such a powerful language. The simple syntax makes Java programs easy to write and read.

An interpreter is needed in order to run Java programs. The programs are compiled into Java Virtual Machine code called bytecode. The bytecode is machine independent and is able to run on any machine that has a Java interpreter. Normally, a compiler will translate a high-level language program to machine code and the code is able to only run on the native machine. If the

program is run on other machines, the program has to be recompiled on the native machine.

Java is one of the first programming languages to consider security as part of its design. The Java language, compiler, interpreter, and runtime environment were each developed with security in mind. The compiler, interpreter, and Java-compatible browsers all contain several levels of security measures that are designed to reduce the risk of security compromise, loss of data and program integrity, and damage to system users. Considering the enormous security problems associated with executing potentially untrusted code in a secure manner and across multiple execution environments, Java's security measures are far ahead of even those developed to secure military systems.

The sizzle of Java is MULTIMEDIA - Sounds, Images, Graphics and Video. In this growing age of multimedia, new computers are known as "multimedia ready" with CD-Rom drives, sound cards, 3D accelerator cards and other new special sound or graphic technology capabilities. Multimedia demands incredible computing power and only recently have affordable computers of this kind are become available. We also need programming languages that make creating multimedia easy. Most programming languages do not have built-in multimedia capabilities. Java, however, through the packages of classes that are an integral part of the Java programming world, provides extensive multimedia facilities that will enable a programmer to start developing powerful multimedia applications immediately. Among the image formats supported by Java is the Graphics Interchange Format.GIF and Joint Photography Experts Group.JPEG. Among the audio formats are AIFF, AU and WAV. Music formats currently supported are MIDI Type 0, MIDI Type 1, and Rich Music Format (RMF). Sound formats now allow 8- and 16-bit audio data, in mono and stereo, with sample rates from 8 kHz to 48 kHz. Capability to play MPEG- Layer 2 and 3 formats are also available. And the list goes on, and continues to grow.

Prolog is also used in programming this application. It is a declarative language. By stating the facts and rules which relate objects in the problem domain to each other, you construct your Prolog program. Its meaning is the set of logical consequences of these program statements, and this is computed by the inference engine at run-time. One need not to be concerned with telling the machine how to solve the problem, nor where to put data in memory. This allows the programmer to concentrate more on the problem at hand rather than on software concerns. Furthermore, scoping rules are simple and uniform in Prolog, and declaration of variable names is not required. This reduces code size and opportunities for error. Prolog programs tend to be from five-to-ten times smaller than the equivalent procedural programs. This reduces the opportunity for human error and reduces maintenance costs. Finally, Prolog is not

specifically for Artificial Intelligence. It is a powerful general-purpose programming language with efficient implementations available on most computing platforms today.

The connection between the two programming languages is through the JPL library - using the SWI-Prolog foreign interface and the Java jni interface. That provides a bidirectional interface between Java and Prolog; it can be used to embed Prolog code in Java programs as well as to embed Java code in Prolog programs (Singleton *et al.,* 2004). In both cases it provides a reentrant bidirectional interface. There are many purposes for using this interface, such as enabling Prolog applications to exploit any Java classes, instances, methods etc. (without requiring any wrappers, metadata, etc. to be set up first). In addition, it enables Java applications to manipulate any standard Prolog libraries, predicates, etc. (without requiring any wrappers, metadata, or other to be set up first). It also enables hybrid Prolog-Java applications to be designed and implemented so as to take best advantage of both programming paradigms, in addition to it being testable, debuggable, maintainable, etc, and to minimize the impact of deployability. In other words, the runtime support for Prolog-Java applications must be a position-independent, self-sufficient file-store tree, requiring no changes to registries, system libraries, system configuration files, etc. Also, to minimum the dependency deployability as with JVMs, the Prolog-Java runtime support must depend upon nothing which cannot be taken for granted in healthy OS installations. Finally, to minimize the vulnerability deployability, that is the Prolog-Java runtime, support must be immune to legitimate variations in its environment (PATH settings, other applications and libraries including other Prolog-Java apps, etc). The available documentation is outdated, however (2004), and doesn't correspond to the most recent version of SWI-Prolog (which contains JPL).

The system was tested in one of the buildings on the Khaldia Campus (namely Kh/11) of Kuwait University. This building has four floors of lecture halls, including the basement, for all academic courses offered by the College of Science. The building is equipped with wireless network connectivity on each floor. Both students and faculty members connect their mobile devices to the Internet in order to access the Black-Board learning management system which is installed by the university. Many connection attempts fail throughout the day due to various reasons such as network failure, incompatibility of the CISCO connection agent (which is required to establish the connection), or the lack of upgrade of the hosting operating system. In an experiment, the system was installed on twenty-five students' laptops, and the students were asked to connect to the Internet. Seven laptops were able to connect directly without any need for the help of the system. Among the remaining cases, the expert system

made the right suggestions for seventeen cases, which means the system offered sufficient help to 95% of the users.

CONCLUSION

In this paper we demonstrated the construction of an expert system for remotely troubleshooting wireless network connectivity. The system combines features obtained from two different programming paradigms, namely object-oriented and logic programming. Java was selected to represent the first paradigm and Prolog was used to cover the second paradigm. Once the scope of the system was specified, the task became finding a suitable library that will bridge Java and Prolog. We have selected JPL for its maturity and relatively good documentation.

Nonetheless, most of the existing knowledge-based systems are not very robust or flexible, as users might want or as state of affairs might necessitate (Shenassa & Khakpour 2008). Such lack of robustness and flexibility are a result of various factors. Some of the more salient factors include Scope, Scalability, Quality of Alternatives, Transparency, Learnability and Reusability.

FUTURE WORK

There is great potential for future enhancements that could enrich the construction of expert systems. It would be relatively easy to support other languages mapping by storing the questions and answers in swappable files and having translators translate those files and feed them to the program. Additionally, the rules could be made more complex so that multiple answers could be reached and assigned weights to offer the user of the expert system alternatives and to simulate the uncertainty faced by human experts.

More detailed pictures or even animations could also be used to help the user. A step-by-step process could probably be displayed as a slideshow when instructing the user to perform a certain task. The system is truly scalable, meaning that the Java code is almost completely independent from the Prolog rules set, so it is possible to integrate other areas (such as other hardware problems like hard disks, video cards, motherboards, etc.), where the expert system can decide the right solution to such problems by creating new rules sets designed to solve other problems.

REFERENCES

Abdul-Rahim, A. 2005. An Intelligent Expert System for Decision Analysis and Support in Multi-Attribute Layout Optimization, **Ph.D. thesis**, University of Waterloo, Canada.

Heragu, S. S. & Kusiak, A. 1990. Machine layout: An optimization and knowledge-based approach. International Journal of Production Research **28**(4): 615-635.

Malakooti, B. & Tsurushima, A. 1989. An expert system using priorities for solving multiple-criteria facility layout problems. International Journal of Production Research **27**(5): 793-808.

Negnevitsky, M. 2002. Artificial Intelligence: A Guide to Intelligent Systems. Pearson, Sydney.

Phan, T., Zorpas, G. & Bagrodia, R. 2002. An Extensible and Scalable Content Adaptation Pipeline Architecture to Support Heterogeneous Clients. Proceedings of the 22nd International Conference on Distributed Computing Systems, Austria.

Shenassa, M. H. & Khakpour, K. 2008. Knowledge base expert system for tuning PID controllers using wireless technology. International Conference on Computer and Communication Engineering, ICCCE 2008.

Sinha, A., Haddad, I., Nightingale, T., Rushing, R. & Thomas, D. 2006. Wireless intrusion protection system using distributed collaborative intelligence. Performance, Computing, and Communications Conference, IPCCC 2006.

Singleton, P., Dushin, F. & Wielemaker, J. 2004. JPL: A bidirectional Prolog/Java interface. http://www.swi-prolog.org/packages/jpl/.

Spyrou, C., Samaras, G., Pitoura, E. & Evripidou, P. 1999. Wireless computational models: Mobile agents to the rescue. Proceedings of Tenth International Workshop on Database and Expert Systems Applications.

Teşekkür, E. 2002. Sniffer Netasyst Network Analyzer, Tepum Secura, http://www.tepum.com.tr/Etkinlikler/SnifferNetasyst.pdf

Submitted : *30/9/2009*
Revised : *22/5/2010*
Accepted : *1/6/2010*

اللغات الداخلة في النظم الخبيرة لصيانة أعطال الإتصال اللاسلكي

محمد الملا

قسم علم الحاسوب – جامعة الكويت ص. ب. 5969 الصفاة – 13060 – الكويت

خلاصة

صيانة شبكة الإتصال اللاسلكي عن بعد يخفف كثيراً من تكاليف الدعم الفني للشبكة والنفقات التشغيلية المتعلقة بها، ذلك من خلال التخلص فعلياً من ضرورة التواجد في الموقع لصيانة مشاكل الشبكة اللاسلكية. يمكن تحقيق هذه الوفورات في التكاليف من خلال عدة أدوات مختصة بصيانة الأعطال. نقترح في هذه الورقة أداة لاكتشاف وإصلاح أعطال الإتصال (عن طريق نظام خبير) تسمح لموظفي المؤسسات بإجراء تحليل دقيق لمعطيات عميل الإتصال اللاسلكي ومعالجة الوضع بسرعة أو رفع أمر إصلاح دونما الحاجة لخبرة لاسلكية واسعة النطاق. وقد تم تنفيذ النظام المقترح باستخدام لغات داخلة تابعة لنموذجي برمجة هي برمجة المنطق وبرمجة الكائنات، وذلك للاستفادة من مزايا هذه النماذج البرمجية مثل الحماية والنقل والاستدلال وتبرير الإجابة.

مجلة الشريعة والدراسات الإسلامية

فصلية علمية محكمة تصدر عن مجلس النشر العلمي بجامعة الكويت
تعنى بالبحوث والدراسات الإسلامية

صدر العدد الأول في رجب ١٤٠٤هـ - أبريل ١٩٨٤م

* تهدف إلى معالجة المشكلات المعاصرة والقضايا المستجدة من وجهة نظر الشريعة الإسلامية.
* تشمل موضوعاتها معظم علوم الشريعة الإسلامية: من تفسير، وحديث، وفقه، واقتصاد وتربية إسلامية، إلى غير ذلك من تقارير عن المؤتمرات، ومراجعة كتب شرعية معاصرة، وفتاوي شرعية، وتعليقات على قضايا علمية.
* تنوع الباحثون فيها، فكانوا من أعضاء هيئة التدريس في مختلف الجامعات والكليات الإسلامية على رقعة العالمين: العربي والإسلامي.
* تخضع البحوث المقدمة للمجلة إلى عملية فحص وتحكيم حسب الضوابط التي التزمت بها المجلة، ويقوم بها كبار العلماء والمختصين في الشريعة الإسلامية، بهدف الارتقاء بالبحث العلمي الإسلامي الذي يخدم الأمة، ويعمل على رفعة شأنها، نسأل المولى عز وجل مزيداً من التقدم والازدهار.

جميع المراسلات توجه باسم رئيس التحرير

ص.ب ١٧٤٣٣ - الرمز البريدي: 72455 الخالدية - الكويت هاتف: ٢٤٨١٢٥٠٤ - فاكس: ٢٤٨١٠٤٣٤
بدالة: ٢٤٨٤٦٨٤٣ - ٢٤٨٤٢٢٤٣ - داخلي: ٤٧٢٣

العنوان الإلكتروني: E-mail - josais@kuniv.edu.kw

issn: 1029 - 8908

عنوان المجلة على شبكة الإنترنت: http://pubcouncil.kuniv.edu.kw/JSIS

اعتماد المجلة في قاعدة بيانات اليونسكو Social and Human Sciences Documentation Center

في شبكة الإنترنت تحت الموقع www.unesco.org/general/eng/infoserv/db/dare.html