NRC Publications Archive
Archives des publications du CNRC

Diagnosing window problems: building an expert system

Ruberg, K.; Cornick, S. M.

This publication could be one of several versions: author's original, accepted manuscript or the publisher's version. / La version de cette publication peut être l'une des suivantes : la version prépublication de l'auteur, la version acceptée du manuscrit ou la version de l'éditeur.

Publisher's version / Version de l'éditeur:

Proceedings of the 4th Conference on Building Science and Technology: 18 February 1988, Toronto, Ontario, Canada, pp. 101-20, 1988

NRC Publications Archive Record / Notice des Archives des publications du CNRC :
https://nrc-publications.canada.ca/eng/view/object/?id=c9ed0b28-7fda-4332-a8cf-397d3b4b1cc4
https://publications-cnrc.canada.ca/fra/voir/objet/?id=c9ed0b28-7fda-4332-a8cf-397d3b4b1cc4

Access and use of this website and the material on it are subject to the Terms and Conditions set forth at
https://nrc-publications.canada.ca/eng/copyright
READ THESE TERMS AND CONDITIONS CAREFULLY BEFORE USING THIS WEBSITE.

L'accès à ce site Web et l'utilisation de son contenu sont assujettis aux conditions présentées dans le site
https://publications-cnrc.canada.ca/fra/droits
LISEZ CES CONDITIONS ATTENTIVEMENT AVANT D'UTILISER CE SITE WEB.

Questions? Contact the NRC Publications Archive team at PublicationsArchive-ArchivesPublications@nrc-cnrc.gc.ca. If you wish to email the authors directly, please see the first page of the publication for their contact information.

Vous avez des questions? Nous pouvons vous aider. Pour communiquer directement avec un auteur, consultez la première page de la revue dans laquelle son article a été publié afin de trouver ses coordonnées. Si vous n'arrivez pas à les repérer, communiquez avec nous à PublicationsArchive-ArchivesPublications@nrc-cnrc.gc.ca.

National Research Council Canada
Conseil national de recherches Canada

National Research Council Canada
Conseil national de recherches Canada

Institute for Research in Construction
Institut de recherche en construction

Diagnosing Window Problems: Building an Expert System

by K. Ruberg and S.M. Cornick

Reprinted from
Proceedings of the Fourth Conference on Building
Science and Technology
Toronto, Ontario. February 18-19, 1988
p. 101-120
(IRC Paper No. 1534)

Price $5.00

NRCC 28920

Canada

8316376

RÉSUMÉ

Les "systèmes experts" ou systèmes à base de connaissances (SBC) sont des programmes informatiques incorporant une connaissance pratique d'un domaine sous une forme facilement accessible. Cette méthode de programmation comprend la "saisie" des connaissances d'un expert, l'organisation de cette connaissance sous une forme logique et la conception d'une interface adaptée à l'usager éventuel du système.

Les systèmes experts se sont avérés efficaces pour recueillir, organiser et transférer les connaissances dans des domaines autres que l'industrie du bâtiment. Les SBC sont utilisés en médecine, par exemple, pour les diagnostics, en géologie pour la découverte de gisements minéraux et pétroliers, en gestion pour l'établissement des calendriers d'employés, en analyse des systèmes pour l'établissement des horaires des lignes aériennes, et en finance pour la planification des investissements. Dans l'industrie de la construction, "Mentor" a été créé par la société Honeywell pour diagnostiquer les problèmes des compresseurs de refroidissement. De la Finlande jusqu'à l'Australie, des efforts sont déployés pour construire des systèmes de diagnostic des désordres du bâtiment et pour aider à la conception. La plupart des systèmes demeurent des outils de recherche. Les systèmes éprouvés sont coûteux à élaborer et sont traditionnellement utilisés par des ordinateurs spécialisés. Toutefois, cette situation est en train de changer à mesure que
disparaît la distinction entre

À l'IRC, nous avons conç gnostiquer les
problèmes reliés aux fenê rche visant à
découvrir et documenter a science du
bâtiment. Le système d nombre de
caractéristiques d'élabora t, de la saisie
des données jusqu'à l'im ssances et la
vérification finale de la n ordinateur
spécialisé mais a été ada es formes et
applications possibles de on, ainsi que
l'élaboration d'un systèm e modèle.

DIAGNOSING WINDOW PROBLEMS:

Building an Expert System

Kalev RUBERG[1] and Steven M. CORNICK[2]

ABSTRACT

"Expert systems" or knowledge-based systems (KBS) are computer programs embodying the practical knowledge of a field in a readily accessible form. This programming method comprises "capturing" the knowledge of an expert, organizing the knowledge in a logical form and designing a computer interface suitable for the ultimate user of the system.

Expert systems have been found effective for capturing, organizing and transferring knowledge in fields other than the building industry. KBS are used, for example, in medicine for diagnosis; in geology for finding mineral and oil deposits; in management for shop floor scheduling; in systems analysis for airline scheduling; and in finance for investment planning. In the construction industry, "Mentor" was built by Honeywell for diagnosing cooling compressor problems. Efforts are ongoing from Finland to Australia to build systems for diagnosing building problems and aiding building design. Most systems remain research tools. Verified systems are expensive to build and have traditionally run on specialized computers. This situation is changing as the distinction between micro and minicomputers begins to disappear.

At IRC we have built one expert system, called WINDEKS, for diagnosing problems with windows. The program remains a research tool to discover and document KBS development methods particular to building science. The window diagnostic system illustrates many of the characteristics of expert system development for the building industry, from knowledge acquisition through knowledge representation to final verification of the knowledge-base and deployment. This system was developed on a specialized computer, but was ported to a micro-computer environment. This paper describes the possible forms and applications of expert systems in the construction industry, and illustrates the development of a knowledge-based system using WINDEKS as an example.

---

[1] Kalev RUBERG, M.Arch. is an Associate Research Officer at the Institute for Research in Construction, National Research Council of Canada

[2] Steven M. CORNICK, M. Civil Eng. is a Research Associate at the Institute for Research in Construction, National Research Council of Canada

DIAGNOSING WINDOW PROBLEMS:

Building an Expert System

Kalev RUBERG and Steven M. CORNICK

1. A COLD AND WINDY DAY

"Jessica stood by the window. The window clouded over just enough to give the world a foggy disposition. It had happened before. Ugly rivulets were already forming from the malignant watery growth. Water spawning on her window. She tried wiping it off. Just made it worse. Sunny and cold they had said. Sure couldn't tell from in here..."

Condensation. It happens on windows, in walls and roofs, and in basements. Along with fungus growth, leakage, acid attack, and leaching it is only one of the "diseases" buildings endure. Finding the causes requires detective work [1] -- a mixture of experience, research and logic. Once the culprit is identified, remedial measures are suggested--again based on experience and taking into account performance and cost.

Building scientists engaged in diagnosis of building problems must bring to the task both their experience and knowledge about buildings. The methods they use are analogous to those used in the medical field to diagnose patients. A "building pathologist" eliminates causes of lesser potential and then concentrates on gathering evidence to support the most likely cause. Diagnosis of buildings is less well defined because no accepted pathologies have been defined and each building differs enough from another to defy simple generalizations.

2. BUILDING DIAGNOSIS

A case of condensation in sealed glazed units illustrates the diagnostic process:

> In a building in Montreal that has large, sealed glazed units with metal frames (longer than 100 cm in length) and an air gap of about 13mm, water was condensing on the roomside surface of the inner lite (numbered from the outside, surface; surface number 4). The condensation occurred as a patch in the middle of the lite. This occurred during the first winter, after the unit was installed in the summer. During the summer, the owner had not noticed any particular deflection of the inner light. In winter, the condensation disappeared as the sun hit the light, but it recurred on cold days, and particularly when there was a rapid drop in temperature. The interior temperature was 20°C with 30% relative humidity. The outside temperature was at -10°C when condensation occured.

First, the mechanism of the window malady (or condensation) was identified. Because of the size of the unit, and because there was no condensation at the edge, the mechanism must be increased heat conduction in the middle of the window panes. This condensation pattern could only be caused by decreasing the width of the air gap in the middle, or by non-uniform radiant cooling of the window. The second mechanism was rejected because there were no obstructions or shading devices that might lead to non-uniform cooling. Therefore the mechanism was one of air gap size reduction. This is caused by a difference in pressure between the air-gap and the atmosphere. This hypthesis was further supported by the large window size allowing the glass to deflect.

In diagnosing the condensation problem, a number of possible causes were rejected. Any air leakage by the sealant would have resulted in some condensation in the a cavity, the lites remaining parallel, and a different resultant condensation pattern. The probability of pumping ction by the window resulting in a dishing pattern was low, as no condensation was discernible on the surfaces o he lites facing the air space. Nitrogen adsorption by the dessicant would occur over time (about four mont) during the first year and would not increase in severity. If nitrogen adsorption had been the mechanism, defl tion of the lites would have been noticed during the fall.

An hypothesis for the cause was formulated. Mar acturing the unit at a lower pressure may not have been immediately evident in the warmer weather. The duced cavity size would be accentuated by lower air space temperatures. The building scientist would concl e that the problem was due to manufacturing the unit at atmospheric conditions significantly different fro hose at the end use location. Manufacturing the unit on a horizontal surface could not be ruled out as a caus In the latter case, the upper lite would sag under its own weight and lead to a reduction of the air gap width.

The solution would be to drill a hole in the spacer ar of the unit and equalize the pressure in the air cavity. This operation should be performed in a dry, controlle nvironment to prevent moisture or dust from entering the unit. The hole would have to be resealed with a sealant ecommended by the sealed unit manufacturer.

The example illustrates the rejection of hypotheses and the gathering of evidence to support a final conclusion. The above example could be re-written in the form of IF...THEN... rules. In the preliminary evaluation of mechanisms the rules could be stated as follows:

> IF the condensation pattern is a patch in the middle of the lite AND the unit is double-glazed sealed unit THEN the mechanism is air-gap width reduction (with a certainty of 50%) OR the mechanism is non-uniform radiant cooling (with certainty of 50%).
>
> IF the condensation mechanism is air-gap width reduction (with a certainty greater than 50%) THEN the *cause* is manufacture at a lower barometric pressure (75%) OR the *cause* is nitrogen adsorption (25%).
>
> IF condensation increases in severity during the first 4 months and the lites are still concave during the warm weather THEN the *cause* is nitrogen adsorption (75%).
>
> IF condensation occurs immediately after light is installed AND continues to be a problem AND the lite is large (short side longer than 100 cm) THEN the unit is probably sealed in a horizontal position resulting in a smaller air space (75%) [2].

"Expert systems" provide the technology to represent this model of expert problem-solving as a computer program. These programs integrate common-sense knowledge, as used in the example above, with calculation methods required for decreasing the uncertainty about a diagnosis. In the example above, the expert may have calculated the window's dewpoint (using psychrometrics ca. 1.9°C) and calculated the nominal internal glass temperature without solar gain (using thermal analysis algorithms [3]) and concluded that a normally behaving 13mm gap would not support condensation on the inside glass under the given conditions.

At IRC we have built an "expert system" for diagnosing problems with windows. As a prototype, WINDEKS [4] (WINdow Diagnostic Expert Knowledge System) has been implemented on three machines and in as many different languages. The prototype has pointed to many problems with "expert system" implementation, but it has also revealed the power of this programming technique for the dissemination of expertise.

3. EXPERT SYSTEMS for DIAGNOSIS

Knowledge-based system technology was first implemented in the medical field and was used for diagnosing diseases and suggesting remedies. The following chapter outlines the terms used in the "expert system" realm and some of the techniques used for writing useful programs.

3.1 Expert Systems

Research in Artificial Intelligence has spun-off a method of computer programming that mimics the expert's problem-solving process. These programs are popularly known as "expert systems". Very few programs built using this technology have achieved an "expert" level of performance. Most are useful as knowledge sources and provide guidance to professionals who make the final decisions. The technology is more appropriately known for building knowledge-based systems (KBS). These are programs that diagnose medical problems, guide geological exploration, advise on chemical spill containment or financial planning, and design electronic circuits and mechanical objects. The technique is well suited to record and organize information in a well defined field where experts are already solving problems. These systems will NOT solve problems not solved before. They cannot (as yet) create knowledge because the key to their operation is they mimic an expert's behaviour, but they do not model his thinking.

Unlike human experts, these systems are bound to relatively well-defined *domains* or fields of problem solving. For example, a medical diagnostic system is incapable of coming to any reasonable conclusions about building diagnosis, whereas a doctor might at least have some understanding of the physical processes associated with condensation. Artificial intelligence research is concentrating on some of these issues, such as developing systems that understand naive physics and natural language. KBS technologies are evolving as well and are increasing in power and breadth of understanding.

The power of KBS lies in their ability to encode *heuristics*. Heuristics are methods used in ill-defined and open-ended problem solving processes. These methods include rules-of-thumb, common sense reasoning, and reasoning under uncertainty. The programming languages are well suited to manipulating symbols (words, numbers and lists of words) rather than only numbers used in traditional computer languages (FORTRAN). These programming methods allow calculation procedures to be embedded or they can call analysis programs. They often integrate graphics processors and support a variety of "user-friendly" interfaces. WINDEKS was programmed to explore the viability of capturing building diagnostic knowledge with rules, to determine the reasoning power and the interface design for diagnostic systems.

Typically, computer programs used in building science are *procedural*. There is a set pattern or *algorithm* to the method of arriving at a solution. With a given set of inputs, there is only one answer. All thermal analysis programs are procedural. Once started, the program follows the given instructions to completion. These programs can be integrated into KBSs to perform analysis or support conclusions. But KBSs differ in the way knowledge is *represented* and in the logic used to tie the knowledge together.

3.2 Parts of a Knowledge-based System

Knowledge-based systems comprise three parts:

- the knowledge-base
- the inference engine
- the user interface

3.2.1 Knowledge-base

The *knowledge-base* is the information used by the system to arrive at conclusions or give advice. Depending on the type of KBS used, its form can vary from rules (looking very much like the example cited above) to "frames" which resemble data-base fields in their structure but are operated on by rules, and can have functions assign values of object properties. These two knowledge structures are illustrated in Fig. 1.

It is the knowledge-base that forms part of the *representation* of the knowledge. In rule-based systems, rules form the knowledge-base. In object-oriented or frame systems, frames are the *representation mechanism.*

Building a knowledge-based system requires building a reasonably complete knowledge-base that represents or models all the knowledge required to solve problems in a particular field. **The accumulation of the rules or frames necessary to achieve this representation (or *knowledge acquisition)* is the most formidable task in building an expert system.**

3.2.2 Inference Engine

The *inference engine* is the reasoning structure or mechanism used to link the knowledge together. Mechanisms have been well defined for rule-based systems, but formal inferencing mechanisms for frames are typically combinations of rule-ordering mechanisms and methods to determine the *inheritance* of frames. Inheritance allows the properties and descriptions of one frame to be passed on to another frame.

For rule-based systems, two well defined techniques are used to move by inference through the rule base:

- *forward-chaining;* where the **IF...THEN...** rules fire when the conditions in the IF part are true.
- *backward-chaining;* where the **IF...THEN...** rules are used to determine which THEN clauses need to be found true to reach a final goal.

Forward-chaining systems are termed *data-driven.* With known facts supporting the truth of the IF statement, the

THEN statement can be considered true (given the rules are valid). The facts produced by determining the truth of the THEN statement then become facts for another IF statement. The last THEN statement becomes the system's conclusion. It has found out as much as it can from the given data.

Backward-chaining systems are *goal driven*. The system has a goal -- for example, to determine the value of a parameter (variable) named *DIAGNOSIS*. To determine that value, a rule that sets *DIAGNOSIS* in the THEN part of the rule is found. The system looks at the known facts to determine whether the IF part of the rule is true. When the truth of the IF statements are unknown, it tries to find rules that determine the facts supporting the IF statement. In turn, the facts supporting the IF statements of those rules are examined. This process continues until no more rules can be *backward-chained*. At that point the system must query the user for a supporting fact. Most KBS diagnosis systems are backward-chaining systems. They try to determine a diagnosis and remedy by working backward through the knowledge-base to find the value in the goal that can be supported logically.

Knowledge-based system technology differs from previous procedural methods by separating the knowledge-base from the logic used to apply the knowledge-base to a problem. This separation makes it possible to maintain the knowledge base (as opposed to re-writing entire programs) by changing specific rules or rule sets. It also makes possible the application of different domains of knowledge to the same inference mechanism.

3.2.3 User Interface

Although often neglected, the "look and feel" of the computer program are more important to the end user than the elegance of the computations. Whereas there are some formal guidelines for the structure of the knowledge-base and the inference engine, few exist for the design of the input and output formats required by users.

The technology is changing rapidly from limited user interaction with screen output and keyboard or mouse pointing devices for input, to innovative voice recognition systems, computer-generated holograms and computer recognition of body positions.

Current systems use bit-mapped graphic displays (similar to the one seen on the Apple Macintosh) for generating menus (for picking commands), graphics, and text. Input is by picking from menus or interacting with the graphics. In lower level systems, keyboard input is required.

As the number of graphic and text display windows on a computer screen increases, so does the confusion. Poor display window management does indeed resemble a desktop; it becomes very messy. A different problem arises when system interfaces become too friendly. For learning to use a system, menus and windows are appropriate. As the user becomes more comfortable with the system and understands some of the logic, the system's response begins to slow the user to a point where interaction is stymied rather than promoted. At some point, command languages and direct keyboard (or voice) interaction become more appropriate. The system should understand a user's proficiency and acclimatize to the user's interface demands. Insufficient knowledge of this area of human-computer interaction means that successful user-interface design is dependent on the programmer's skill and understanding of the user's demands.

3.3 Expert System Shells

When the expert system for diagnosing infections in hospitals (MYCIN) was developed at Stanford [5], it was found that rules could be removed from the system, another set of rules diagnosing different ailments put in its place, and the logical inferencing mechanism would still draw conclusions from this new set of rules. The *knowledge-base* could be extracted, replaced with a knowledge base from another domain, and provide useful answers in this new domain. A KBS *shell* was born. The shell provides a programmer a system with all the inference mechanisms and user-interface support required without the knowledge base. It resembles a very high level computer language.

Using different inferencing mechanisms (forward-chaining or backward-chaining), these *inference engines* were packaged with appropriate *user interfaces* to develop systems for building knowledge-bases. These systems have an empty rule set, an inferencing mechanism, and editors to input rules, descriptions of parameters and links to completely define the knowledge in a domain. Users do not have the inconvenience of building the logical inferencing mechanisms or the rule editors. But control of the rule traversing logic is often sacrificed and custom inferencing methods cannot be implemented. These trade-offs must be weighed when implementing systems for applications.

WINDEKS was initially implemented using a shell developed at IRC. The rules were then applied in a commercially available rule-based shell. Development time on the commercial system was three weeks. For the in-house shell, development time was about one year, but this included acquiring knowledge from experts about windows. Acquiring knowledge for the system proved to be the largest obstacle.

At present, more than 25 commercially available shells can be used on ATs [6]. They range in price from $400 to $10,000 CDN. These high level languages are useful for rapidly prototyping knowledge-based systems, but each shell has its strengths and its drawbacks. Shells with backward and forward-chaining and high-level rule control are more suitable for diagnostic system development. Frames are better suited to modelling applications or real-time system control.

4. DIAGNOSTICS

Expert system technology can be used to mimic expert problem solving in a range of fields from planning, design, teaching, through diagnosis. Diagnostic problem solving was one of the earliest applications of the technology. In the realm of medicine, MYCIN is a system used to diagnose infectious conditions and suggest remedies. It implements certainty conditions. A doctor could specify how certain he was of a condition he was observing. The system uses backward-chaining to reach a goal and assigns a certainty to that goal. An example from MYCIN is shown in Fig.2.

4.1 Building Diagnosis Systems

Knowledge-based system prototypes have been built for diagnosing problems with buildings. These include:

- a dry rot diagnostic system (Lansdown 1982) [7].
- a compressor preventative maintenance and diagnostic system MENTOR (Cochran and Hutchins,1986) [8]
- a fungus growth system (Smith, 1987) [9]
- a system to diagnose problems with windows, WINDEKS (Ruberg and Cornick, 1985) [4]
- a system to diagnose water penetration problems through curtain walls (CSIRO 1986) [10]

Of these, only MENTOR has been field tested and commercially used. All use traditional backward-chaining techniques to determine the diagnostic goal.

4.2 WINDEKS

At IRC an expert system was written to represent knowledge about window problems. The system queries the user for a description of the window problem and then deduces the cause and suggests a remedy. With only 220 rules, the system is capable of analyzing most condensation problems and some breakage problems. Large systems have over 10,000 rules (for example, XCON, a system for specifying computer systems) [11].

4.2.1 What WINDEKS Looks Like

A session begins with the user sitting at an AT computer and being questioned about the description of the problem. The user can change any answer at any time and query the system for an explanation of the question or the description he is being asked about. The system determines the cause and suggests a remedy. The system will determine as much as can be concluded the description given. For example, if the user indicated there was condensation on the frame and the window, and the glazing was a sealed glazed unit (and this is all the information available), the system will conclude, with a low certainty (30%), that the problem is heat conduction through the spacer and the frame. As the user specifies more information about the condition, the cause and the remedy associated with the cause become more specific and uncertainty decreases.

In addition to using heuristics to determine the cause and suggest a remedy, the KBS uses procedural functions to determine the dewpoint in a room (specified by the user) and calculates glass surface temperatures using the Window 2.1 algorithms [3]. Glass temperature variations due to frame types are determined heuristically from the window tests performed at IRC. Weather data from Canadian cities are "binned" according to hours at 2°C intervals. This entire *procedure* is used to estimate the percentage of time that condensation is likely to occur on a particular window. This information allows the user a choice: either live with the condensation problem or accept the suggested remedy.

Three versions of WINDEKS have been developed. The program has the same knowledge-base in each case, but looks different to the user because of the software and hardware configurations. The first version was developed on a *LISP machine* [12]. These special computers support very elaborate interfaces, with graphics in menus and display windows. Because these machines support graphic interactive interfaces, we developed a system that provided the user with a query window, a status window, an answer window, explanation and menu windows. All this information proved too confusing to use. An example of this user interface is shown in Fig.3.

When ported to an AT [13], the available LISP compilers would not support graphics. The new interface was a simpler text interface. Functions were written to provide at least menuing capability for user answers. An example of the user interface is shown in Fig.4.

The third version used a commercially available shell with the developed knowledge-base. Development time was reduced, and the user interface was automatically created by the shell. Running on an AT, the shell allowed the user to pick from menus, and allowed the developers to include graphic explanations or graphic menu choices. An example is shown in Fig.5.

4.2.2 How WINDEKS Works

The inference engine and knowledge-base written at IRC were rule-based. The inference engine is both backward and forward-chaining. The structure of the program is shown in Fig. 6.

The goal for the backward-chaining portion of the program is to determine a remedy for a particular window problem. Three main tasks comprise the system operation:

1. determining the symptom
2. finding the cause
3. determining a solution

Knowledge is entered in the form of IF...THEN (production) rules into a rule editor. The first type of clause found in the rule sentences is a simple fact (example; *condensation occurs on the inner pane*). For this clause to be true, the fact must be known. A second type of clause will bind variables to known values (for example; *glazing-type is sealed-double-glazing*) and a third type of clause will call a function (for example, to calculate the condensation potential). IRC's shell has the capacity to provide explanations, call a dictionary for explaining terms, and determine what rules are required to reach a conclusion or explain how a conclusion was reached.

WINDEKS was conceived to be a tool for professionals answering queries about window problems (such as HEATLINE at Energy Mines and Resources). It would enable consistent responses to problems and provide the staffers with the highest level of knowledge available in the field. WINDEKS's level of knowledge in one area (condensation) has passed limited field tests. Its implementation in a "hotline" environment would require extension of its knowledge to other areas (like chemical etching, leakage and breakage) and a firm commitment by an end user willing to implement it.

5. CANNING KNOWLEDGE

Building WINDEKS required a proper knowledge representation to be chosen. It also required acquiring and packaging the knowledge about the domain in the chosen representation. The method used was "ad hoc". The knowledge-base was built from cases based on semi-structured interviews with experts.

This is the method most expert systems builders have relied upon. Formal procedures are appearing in the literature for interview strategies. These include structured interviews and a series of constrained cases. When constraints are imposed on an expert, the methodology often becomes clear. For example, if a time constraint is placed on an expert, the decisions made in the time given indicate the priorities in decision making. Another constraint would be to present only partial information resulting in another set of priorities used by an expert during his decision making process.

The computer science solution is to build another computer program to encode the knowledge more naturally. There is no consensus on whether it is better for the expert to become a programmer familiar with KBS technology and build the system himself or for the expert to work through a "domain-naive" knowledge engineer to avoid biases.

A system for the design of steel frames exemplifies the expert building the system himself. Built by Techtrol in Montreal [14], the LISP based system allows the user to freehand sketch a steel frame outline on the system and then provides the user with guidance on selecting the frame members. Dr. M. Akroyd, the systems designer, is an expert in the field of steel structures and has "picked up" the technology while developing this design system. It alleviates many of the problems inherent with knowledge acquisition and the system's knowledge is very consistent. But it also represents only one man's knowledge and does not take advantage of the technology's ability to integrate knowledge from many sources.

On the other hand, a system was developed to diagnose faults in a telephone switch [15] by "knowledge engineers" who had little experience in this field. They relied on interviews with experts, but were well versed in expert system technology. This system is now field tested and supported by the systems group that built it. Any other knowledge acquisition technique would have been impractical as multi-disciplinary knowledge was required.

5.1 Knowledge Acquisition--The Bottleneck

Knowledge acquisition methods do not always reliably transfer the knowledge to the knowledge-base. Over-generalizing an expert's examples can cause problems while transcribing the descriptions into rules. Using several experts can lead to conflicts of *perceived* information and these conflicts must be resolved. If cases are used as the basis for rules, omissions of common occurrences may result in incomplete knowledge bases with critical gaps.

5.2 Knowledge Acquisition for WINDEKS: Semi-structured Interviews

For WINDEKS, unstructured interviews with experts reviewing cases of window failures were used. With three experts, three different human analysis processes were observed. The first expert related cases where failures had occurred, and how he believed they had occurred. This is a classical method for developing rule based systems based on description of cases. The second expert reasoned by the use of a model of the fault, and the representation was more amenable to object descriptions. The third described ideal scenarios and then compared these scenarios to faults found with the problem window. By deciding to represent all these problem solving methods in a rule-based system the onus was placed on the knowledge engineer to develop a conflict-free knowledge-base. This did not happen. Conflicts and misinterpretation of the experts' descriptions has led to a knowledge-base that has faults and must be verified.

5.3 Automated Procedures for Knowledge Acquisition and Verification

Automated procedures for rule verification have been developed by Nguyen and others for checking rule bases. Verification is an area where further development must occur before widespread application of this technology for building fault diagnosis can occur. Existing algorithms can be used to check for consistency. Redundant rules, subsumed rules, circular rules, unnecessary "IF" conditions, and completeness are some of the automated checks that can be applied to rule bases [16]. These algorithms are be written at IRC and will be applied to the rule bases we have built for WINDEKS .

An automated method for creating rules has been developed at IRC. It uses a comprehensive set of parameters (and their values) to build case descriptions. A classification tree algorithm then traverses the data and decides on the optimum split and the resulting rule base. The shortcomings of this system are:

- the need to define all the parameters and their attributes beforehand.
- the fact that the knowledge domain is "closed" where new inferences from the old set are not possible.

Another automated approach uses the personal construct system [17] to map a set of the experts "beliefs". These "belief structures" are used to build rules. Two computer software systems use this method: ETS (Expertise Transfer System) by Boeing [18] and PCS (Participant Construct System) by the Alberta Research Council [19]. The construct system can be used to develop rules, but it is a less intuitive method for rule production and cannot be used to describe systems as models. This method is not well suited to diagnostics because of the structure of diagnostic tasks. This type of approach would be used to code rules for choosing windows.

6. KNOWLEDGE-BASED SYSTEMS: TO THE RESCUE

"Jessica turned to her desk. The machine's whir was comforting.

"What do I do?" she asked Hal.

"Jessica, I think it's hopeless. This window will have to be replaced, or at the very least taken out and drilled."

The voice was soothing..."

The technology for diagnostic tools in the building related domain is not quite that good, but the potential exists not only to diagnose problems, but to complement designer's during their decision making. This requires research into:

- knowledge acquisition methods
- representation of buildings in knowledge-base structures
- user interfaces to fit into the building delivery context

At IRC, WINDEKS has been a test bed to discover the problems with KBS development. Knowledge acquisition remains the greatest hurdle in the successful application of this technology. This implies developing very generalized modelling systems that experts themselves can handle.

Consensus among experts is difficult to reach because of different ways of describing similar situations. A tool developed at Alberta Research Council for reaching consensus and articulating beliefs is useful for formalizing the different beliefs of experts in a domain [19].

Representation or modelling of buildings using knowledge-based systems requires taxonomy development and the testing of various configurations. Object oriented programming (a type of programming where "objects" are described according to their physical and dynamic propoerties) promises to be a powerful approach to most building-related problems, but the relationship of the objects must be defined, as must the dictionary of building parts.

This initial prototype has been useful for defining the course of research in this area and demonstrating its potential to future users and knowledge contributors. Application of KBS technology in the construction industry requires industry involvement in the research we perform. Industry participation helps focus the problems being defined and provides feedback for future expert systems development.

REFERENCES

[1] Jack Fielding, "A Kind of Detective Story: Diagnosing Building Defects", **Building Research and Practice**, March/April 1987. pp. 93-96.

[2] J.P. Ausikaitis, "Insulating Glass Unit Reliability", **Glass Digest**, October 15, 1987. pp 69-78.

[3] Windows and Daylighting Group, Lawrence Berkeley Laboratory, 90-3111, Berkeley, "WINDOW: A computer Program for Calculating U-Values and Shading Coefficients of Windows, USER AND REFERENCE GUIDE, Version 2.0", July 1986.

[4] K. Ruberg and S.M. Cornick, "Window Diagnostician: A knowledge-based System for Diagnosing Problems with Windows", **Proceedings of the Third International Congress on Building Energy Management**, Sept 28, 1987, Lausanne, Switzerland, (Presses Polytechnique Lausanne, Switzerland) Sept. 1987. pp. 356-363.

[5] B. Buchanan, E. Shortliffe, **Rule Based Expert Systems**, Reading Mass, (Addison-Wesley) 1984, pp. 209-232. (reference to MYCIN)

[6] R. Freedman, "Evaluating Shells", **AI Expert**, Vol2., No. 9, September, 1987. pp.68-84.

[7] John Lansdown, **Expert Systems: Their Impact on the Construction Industry**, Report to the RIBA Conference Fund, London, RIBA, 1982. 105 pp.

[8] E.L. Cochran, and B.L. Hutchins, "Testing, Verifying and Releasing and Expert System: the Case History of Mentor", paper submitted to **AAAI'86**, Golden Valley,(Minn.), Honeywell Corporate Systems Development Division. 1986.

[9] J.L. Smith, "Frameworks for Defects DIagnosis", **Building**

Research and Practice, March/April 1987. pp. 85-86.

[10] John V. Thomson, "A Water Penetration Expert System Using Prolog with Graphics", **Proceedings of the Second Australian Conference on Applications of Expert SYstems**, New South Wales Institute of Technology, Sydney, 14-16 May, 1986. pp. 51-73.

[11] S. Polit, "R1 and Beyond: AI Technology Transfer at DEC", **The AI Magazine**, Winter 1985.

[12] Patrick H. Winston, Berthold Klaus Paul Horn, **LISP**, Addison-Wesley, Reading Mass., 1981.
LISP machines are computers designed specifically to run LISP (LISt Processing Language. It uses symbols and lists as its computational syntax.) The computer architecture supports fast and efficient processing of this symbolic language. With typical computer architectures, much memory and fast processors are required to support computation with LISP.

[13] AT is an Intel 80286 based microcomputer with a 16 bit bus specified by IBM (TM).

[14] John H. Grubbs, Michael H. Ackroyd, Kincho H. Law, "An Intrinsic Behaviour-based Model for Interactive Structural Design", **Engineering with Computers, 2**, 185-198 (1987).
and "Use of Artificial Intelligence in Engineering Design", informal paper at NRC from Techtrol Ltd. This expert system incorporates data-base tables, heuristic rules and finite-element analysis to help a designer during the design of a steel structure.

[15] Prerau, David S. Summer 1987. "Knowledge Acquisition in Expert System Development", **AI Magazine**, Vol. 8, No. 2, pp.43-52.

[16] Nguyen, Tin A., Perkins, Walton A., Laffey, Thomas J., Pecora, Deanne, "Knowledge-base Verification", **AI Magazine**, Vol.8.,

[17] The personal construct system was developed by G.A. Kelly in **The Psychology of Personal Constructs**, (Norton, N.Y., 1955). It a method for mapping personal beliefs in terms of opposites and determining conflicts in those beliefs.

[18] John.H. Boose, **Expertise Transfer for Expert System Design**, Vol 3 in the ADVANCES IN HUMAN FACTORS/ERGONOMICS Series, G. Salvendy, Editor, Ellsevier, New York, 1986.

[19] Alberta Research Council,"Participant Construct System, User Manual", ARC-CPC-PCS-MANUAL, Feb.86, ARC, Calgary, Alberta.

RULES

Rule 1
If
there is evidence of insulating_gap_reduced
And first_occurrence is less than 8.00
And symptom is INTERMITTENT
And first_occurrence is greater than or equal to 3.00
Then air_and_nitrogen_adsorption_by_sieve_and_adsorbent
is confirmed.
And solution is set to EQUALIZE_PRESSURE_WITH_A_PRESSURE_TAP

Rule 2
If
condensation_pattern is OCCULAR
And there is no evidence of radiant_obstructions
And there is evidence of insulated_glazing_unit
Then insulating_gap_reduced
is confirmed.

Figure 1a. A rule in a rule based system describing the air gap reduction. The second rule relates the condensation pattern to a particular condensation pattern.

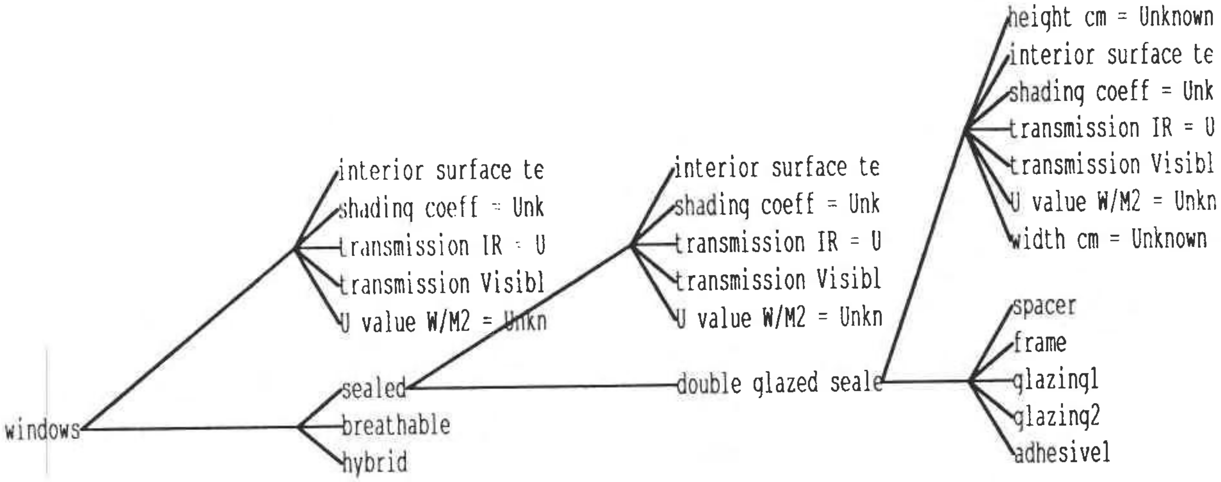
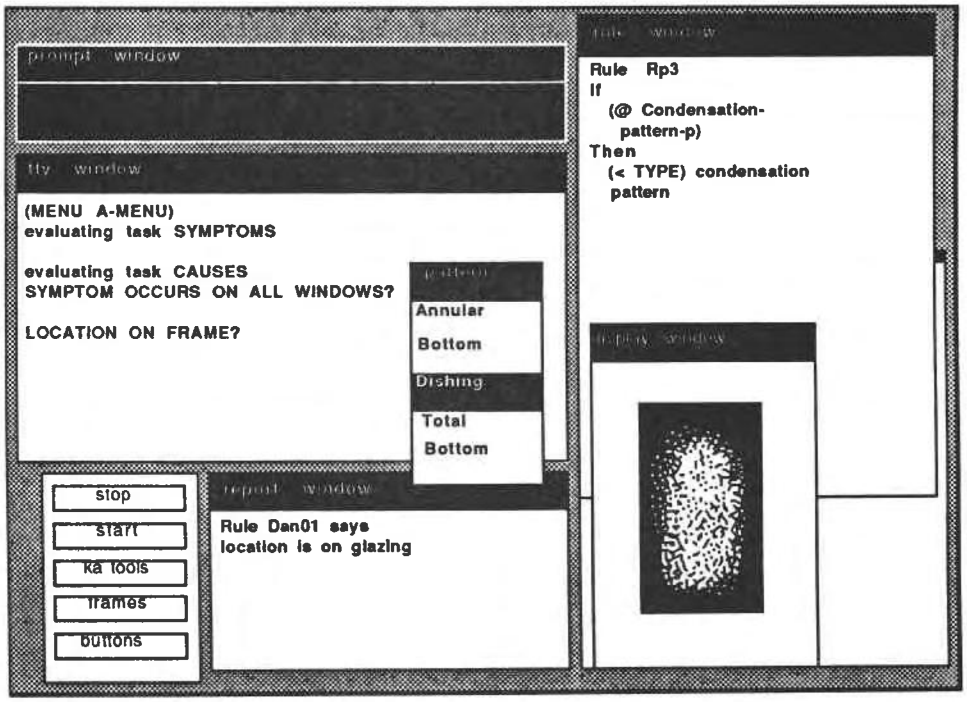


Figure 1b. Object oriented representation of windows. The properties are shown above and increase in specificity as the object descriptions become more specific. Rules may be applied to the objects.

IF: **1) The stain of the organism is grampos, and**
2) The morphology of the organism is coccus, and
3) The growth conformation of the organism is chains

THEN: **There is suggestive evidence (0.7) that the identity of the organism is streptococcus**

Figure 2. Rule from MYCIN, an expert system used for diagnosing g bacterial infection in patients.

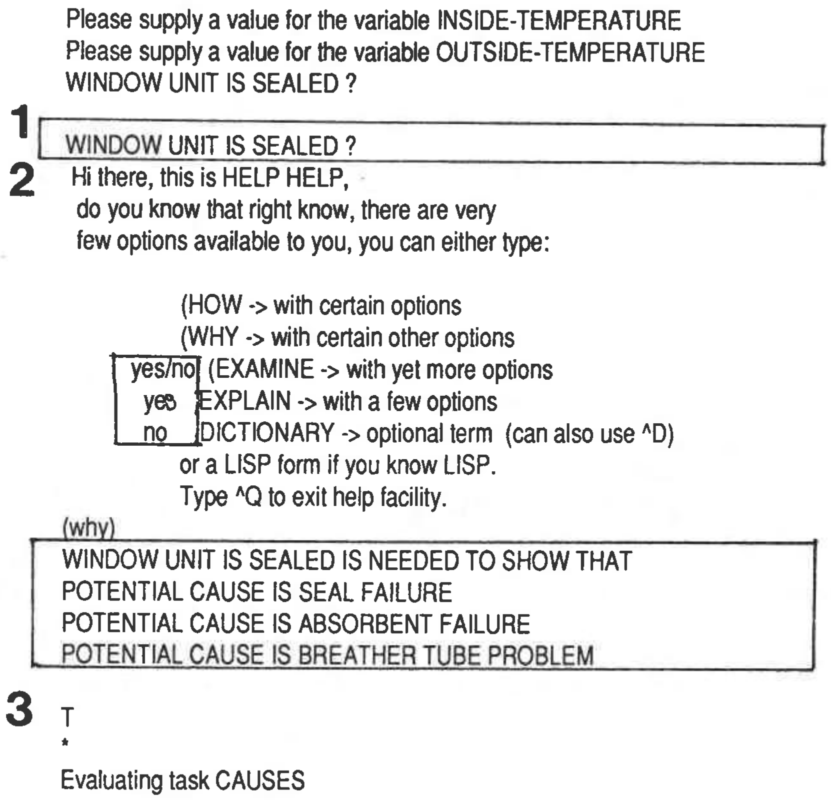


```
(MENU A-MENU)
evaluating task SYMPTOMS

evaluating task CAUSES
SYMPTOM OCCURS ON ALL WINDOWS?

LOCATION ON FRAME?
```

```
Rule  Rp3
If
  (@ Condensation-
    pattern-p)
Then
  (< TYPE) condensation
  pattern
```

```
Rule Dan01 says
location is on glazing
```

Figure 3. Typical screen from WINDEX found on a LISP machine. The prompt window is on the upper left, the responses are selected from a mouse window, and the rules being "fired" are found on the upper right. The condensation pattern is shown on the query window.

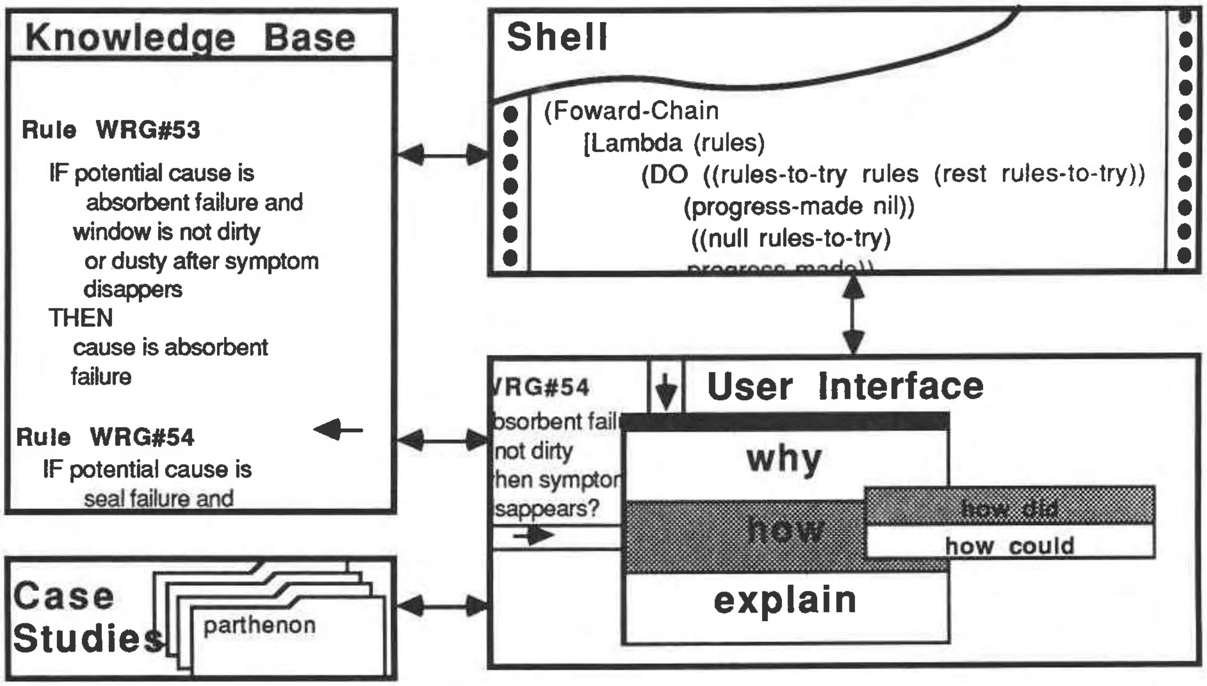


```
Please supply a value for the variable INSIDE-TEMPERATURE
Please supply a value for the variable OUTSIDE-TEMPERATURE
WINDOW UNIT IS SEALED ?

WINDOW UNIT IS SEALED ?
Hi there, this is HELP HELP,
 do you know that right know, there are very
 few options available to you, you can either type:

               (HOW -> with certain options
               (WHY -> with certain other options
      yes/no   (EXAMINE -> with yet more options
      yes      EXPLAIN -> with a few options
      no       DICTIONARY -> optional term  (can also use ^D)
               or a LISP form if you know LISP.
               Type ^Q to exit help facility.
(why)
WINDOW UNIT IS SEALED IS NEEDED TO SHOW THAT
POTENTIAL CAUSE IS SEAL FAILURE
POTENTIAL CAUSE IS ABSORBENT FAILURE
POTENTIAL CAUSE IS BREATHER TUBE PROBLEM

T
*
Evaluating task CAUSES
```

Figure 4. IRC's shell has three parts, a query line (1) an answer area (2) and an evaluation block (3). The mouse is used to enter answers and help is available to query why certain questions are being asked.

```
                    WINDEKS

--------------------------------------------------------------------------
What is the water-related problem with the window?


     Yes
        Condensation on glazing
     •  Misty or oily film between the glazing layers
     •  Condensation on frame
     •  Water on frame
     •  Water on wall around the window frame


  1. Use arrow keys or first letter of item to position cursor.
  2. Select all applicable responses.
  3. After making selections, press RETURN/ENTER to continue.
--------------------------------------------------------------------------
                    WINDEKS

 Conclusions:----------------------------------------------------------------

 The probable cause of the problem is as follows:  The seal in your sealed
 unit may be off-gasing and deposting a film on the window.  (40%)

 An interim recommendation to solve the problem is as follows:  The unit has
 probably failed and may have to be replaced.  (40%)

 Cause of the condensation or breakage pattern is as follows:  The air gap
 between the inside and outside light (glass layer) has been reduced by inward
 glass deflection. This has increased conduction of heat and lowered internal
 glass temperatures in the middle of the glazing. Nitrogen absorption by the
 dessicant has lead to a lower pressure inthe sealed cavity resulting in
 greater deflection.  (90%)  The air gap between the inside and outside light (
 glass layer) has been reduced by inward glass deflection. This has increased
 conduction of heat and lowered internal glass temperatures in the middle of
 the glazing. The nominal unit width is too small for the pressure differential it is being subjected
 to.  (80%)

 Cause of problem located where it is in the glazing system is as follows:
 Sealant failure
  ** More - RETURN/ENTER to continue
 --------------------------------------------------------------------------
```

Figure 5. Query screens from an expert system "shell". Simple lists are presented to the user for choices. After conclusions are presented, the user may retract any statements made.



Rule WRG#53
IF potential cause is
absorbent failure and
window is not dirty
or dusty after symptom
disappers
THEN
cause is absorbent
failure

Rule WRG#54
IF potential cause is
seal failure and

```
(Foward-Chain
    [Lambda (rules)
        (DO ((rules-to-try rules (rest rules-to-try))
            (progress-made nil))
            ((null rules-to-try)
            progress-made))
```

VRG#54
bsorbent fail
not dirty
hen symptor
sappears?

Figure 6. Structure of IRC's knowledge based system WINDEKS with the 3 components (the knowledge base, the shell and the user interface)