#6621

#6621

WINDOW DIAGNOSTICIAN: A KNOWLEDGE BASED SYSTEM FOR DIAGNOSING PROBLEMS WITH WINDOWS

K. RUBERG and S. M. CORNICK
Institute for Research in Construction
National Research Council of Canada
Ottawa, Canada

1. Introduction

Diagnosis of building envelope problems demands expertise acquired with *experience* over time and a deep understanding of building physics. Finding the causes of the problem is a process based on a *heuristic* or common sense understanding of the situation, a knowledge of building science and deductive reasoning about the symptoms. Models of diagnosis are difficult to code procedurally, but are well suited to technologies developed in the *artificial intelligence* realm. Expert systems properly known as knowledge based systems (KBS) have been used to diagnose medical problems (a process requiring broad experiential knowledge founded on a deeper scientific understanding) and similar methods appear to be appropriate for diagnosis of building envelope problems.

One of the building envelope components most prone to problems is fenestration. Condensation, leakage, breakage and etching are only some of the prevalent conditions building owners and managers must cope with. The WINdow Diagnostic Expert Knowledge System (WINDEKS) is a classical knowledge based diagnosis system for identifying perceived problems with windows and suggesting corrective measures. Although the system can be expanded to consider all window related problems, at this time it addresses only moisture related situations and some breakage conditions.

The goals of our research include:

1. development of a system for answering professional and public enquiries about window problems. This comprises:
   a. development of knowledge acquisition strategies,
   b. capture of IRC and outside expert knowledge,
   c. development of knowledge representations for building envelope diagnosis,
   d. development of a knowledge base from the acquisition phase,
   e. development of user interfaces to support access to the KBS.
2. deployment of the system at a public agency to support the answering of queries and getting feedback from those answers.
3. evaluation of system performance in this field deployment.

The work presented covers the first goal. A description of the system, the process of developing the system, and conclusions drawn from this development follow.

2. System Description

2.1 Using WINDEKS--What it Looks Like

This KBS system was developed to support the answering of public enquiries by building information agencies or "hotlines". The system user may be an architect or engineer with some experience in building related fields who answers calls concerning problems with buildings. For a problem with windows, the user would turn to WINDEKS, resident on a XEROX Lisp Machine, and have the KBS help in answering the query.

The user is confronted with numerous computer windows (not the building kind) prompting for identification of symptoms. "Ice, water or fog on window" is one of the six general problem categories. Although the other categories are presented, the rule base is not yet complete enough to allow consultation on breakage, visual distortion, visual obscurity, or etching. This screen is shown in Fig. 1, and is one of the first screens the user sees.

Responses to the system are through mouse input on text and graphic menus. In an example consultation, the user identifies the problem context: he has identified the context as recurring condensation on the room side surface of a sealed glazing unit in a commercial setting with square windows. The system queries the user for a condensation pattern, shown in Fig.2. The user picks the "dishing" pattern. Next the user is queried about seasonal occurrence and approximate outdoor temperature. When the user responds with "only during cold weather, around -20 °C", the system establishes the cause as a narrow gap between the glazing planes in the middle of the unit caused by a pressure difference between the space and the surrounding atmosphere, resulting in glass deflection. This has lead to a smaller air space and resulted in higher conductivity at the centre of the glazed unit.

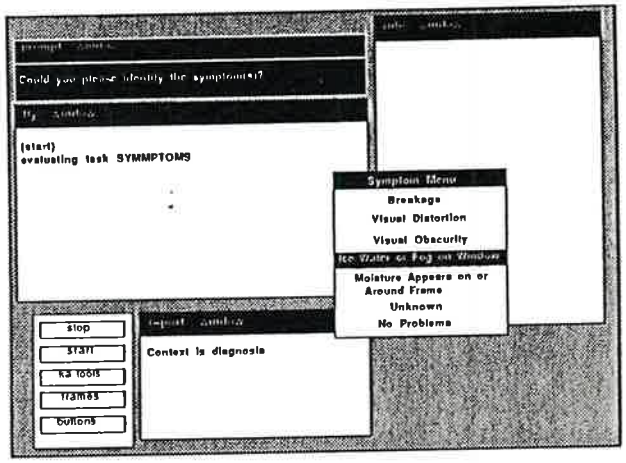

Fig.1. Initial window diagnostic screen prompting for general symptoms.

This condensation pattern results from manufacturing the glazing unit under lower pressure than is found at the building site, or from adsorption of nitrogen by the dessicant in the sealed glass unit space, or because of a sealed unit with an initial air space too narrow to accommodate the expected glass deflection. Since it occurs only during extremely cold weather, when the contraction of air volume

between glazings results in the smallest air gap, the probable causes are manufacture at low pressures or a narrow initial air gap. Although nitrogen adsorption may still contribute to the difference in pressure, the probability of this effect is small as there is no condensation on the outside of the pane during very hot, humid days when the inside surface of the pane is exposed to cooling.

In this case, the solution calls for replacement of the unit, with inclusion of a clause in the specification calling for a breather tube during transport from the assembly plant to the construction site, addition of a third storm sash, or a replacement unit with a wider air gap. (As the user did not specify initial air gap width, the system could not infer more than was allowed by the known facts). In a residential setting, other solutions (including plastic film) would have been presented.

During this consultation, the user could query the system for explanations about the phenomenon (as he did with the dishing), or ask how it arrived at the conclusion or how it might arrive at a different conclusion. In the latter two cases, the system responded with the *rules* used during the deductive process.

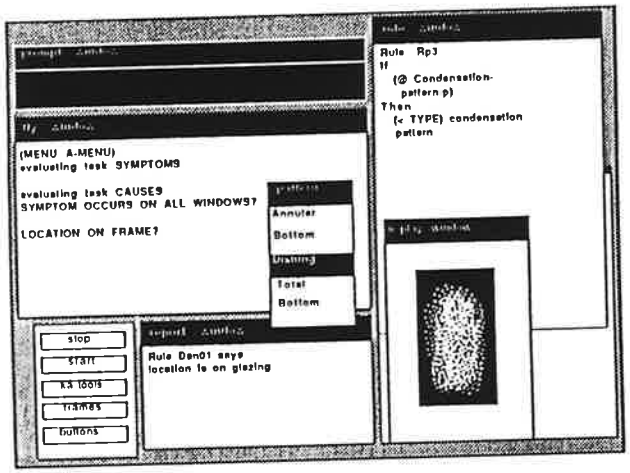

Fig.2 Graphic menu prompt asking for condensation pattern on window.

3. Building WINDEKS

3.1 The Process

Building the system comprised three stages of the six typically found in KBS development:

1. acquiring knowledge in the domain. This includes familiarization with the literature and with practising experts. One of the leading experts in diagnosing moisture related window problems resides at IRC.

2. experimenting with representations of the problem. A number of KBS representations are possible, from semantic nets through frames to rule based systems. Rule based systems are well suited to diagnosis.Prior experience in medical diagnosis has proved the viability of rule-based systems.
3. developing the knowledge base.
4. designing an interface for the system user.
5. field testing to determine the robustness of both the knowledge base and the user interface.
6. continued expansion and support of the knowledge base.

WINDEKS is currently at stage four. Limited field tests (87 cases) with the current knowledge base provided satisfactory explanation for 85% of the condensation related problems. A thorough field test is planned.

3.2 Knowledge Acquisition

Much work in the artificial intelligence research community has been directed towards acquiring knowledge so as to understand the process of an expert's problem solving approach. Two approaches were used; *observational* and *intuitive*.

Observational techniques or procedural simulation (also known as protocol analysis) comprise recording and verbally describing the expert's actions while solving a problem. It is unrealistic to expect to observe all possible solutions by means of this approach, but it provided most of the rules used in the system. The second strategy, forward scenario simulation (archetype acquisition), was used to "walk through" problem solving scenarios.

Introspection by the expert on his actions during problem solving forms the basis for intuitive methods. This is limited by the expert's own perception of how he solves problems, because his knowledge is "compiled" (1) or in a form that cannot be formally expressed. Our first expert relied on problem scenarios and past experience as guides to the knowledge he was providing. To a large extent, he distilled the knowledge as rules.

Our second expert contributor concentrated on the physical window structure and described the potential mechanical problems arising from application of materials. The third expert described a diagnostic process based on comparison of an "ideal" model window and its detailing with that found in a problem situation. He diagnosed failures in real-world situations by comparing differences in detailing or poor implementation. Structuring the "ideal model" knowledge and designing an interface supported the comparisons eluded us.

Automated knowledge acquisition was tried on a small scale. The Participant Construct System (2) developed from Shaw's (3) personal construct theory proved to be more useful for developing a window chooser and priortizing criteria rather than development of diagnostic systems. Systems based on personal construct theories force bi-polar decision making and do not support the type of reclassification of the problem necessary for efficient rule base development.

3.3 Knowledge Representation

The type of knowledge representation is dependent on the nature and structure of the domain. A general diagnostic system for building envelope problems would comprise a hybrid approach utilizing frames or objects to construct hierarchical models where sub-domain components are represented using rules.

In choosing a knowledge representation, a problem solving strategy is hypothesized in which the domain knowledge is represented in a form best suited to the problem. The simplest type of KBS are

rule-based systems. In these systems, like WINDEKS, domain knowledge is represented as a collection of IF-THEN rules. The problem solver chains these rules until a goal is reached.

Like all KBS, WINDEKS exhibits self reasoning (explanation) but (unlike most KBS) incorporates an empirical model for condensation potential. A shell surrounding WINDEKS supports editing and maintenance of the knowledge base, and provides the inferencing mechanism for chaining the knowledge base.

3.4 The WINDEKS KBS Shell

The WINDEKS system is written in InterLISP D. The inference engine is comprised of forward and backward chaining algorithms. These algorithms are used in combination in order to find specified context, cause and solution goals defined in WINDEKS. The relationship of the knowledge base, the inference engine and the interface is shown in Fig. 3.

In WINDEKS the end goal is to determine a solution for a given window problem. The shell was intended primarily for diagnostic purposes; consequently the system's operation is divided into three main tasks. Reflecting these tasks, the shell's inference mechanism attempts to satisfy three goals:

1.- find the symptom
2.- find the cause
3.- find a solution

Knowledge in the form of production rules is entered by the knowledge engineer using a structure editor. Rules can be included in the knowledge base as they are found. Rudimentary knowledge acquisition tools allow knowledge to be rapidly encoded into a production rule format. WINDEKS allows the knowledge engineer to compile the rule knowledge base. The inference engine can traverse a compiled and networked rule base three times more quickly than a partially compiled base, but unlike some expert system tools the rule compilation stage is optional. Run time units consist of completely compiled knowledge bases.

In addition to the knowledge acquisition tools, WINDEKS has a capacity to be retrospective about a diagnosis. The system is capable of retracing the reasoning path followed by allowing the user to inquire about how certain facts were arrived at. The system is also able to explain why it requires certain information, showing the user the current goals.

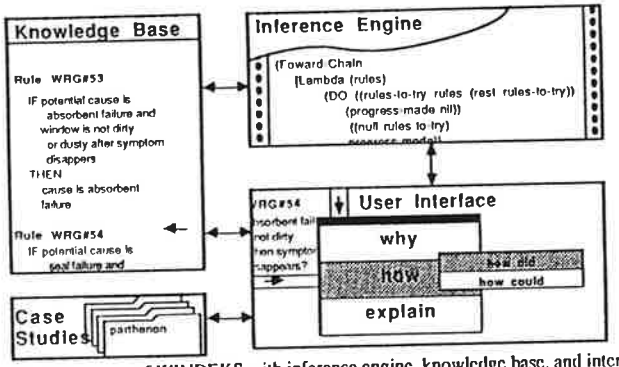

Fig. 3 Structure of WINDEKS with inference engine, knowledge base, and interface relationships.

3.5 The Knowledge Base

3.5.1 Developing the Knowledge Base

Informal techniques were used to develop the knowledge base. *Reclassification* of the problem into distinct packets and observables led to the creation of four unique classes of moisture related problems. This is in contrast to formal automated techniques of developing classification trees (4). In retrospect, the classification tree technique would have been more useful for developing specific diagnostic rules. Most commercially available knowledge based shells support this type of classification (5).

3.5.2 Rules

Knowledge about windows in WINDEKS is represented as a collection of production rules. For moisture related problems, there are 172 rules in 11 condensation and moisture categories. The syntax for the production rules was relatively straightforward:

RULE <rule-identifier>
IF <clause$_1$>,<clause$_2$>..<clause$_n$>
THEN <clause$_1$>,<clause$_2$>..<clause$_n$>

A rule can be considered by the system if all the IF clauses are satisfied and at least one THEN clause is left unsatisfied. Clauses can have several different forms.

1.- <Condensation Occurs on Inside of Inner Panes>
2.- <Window Unit Is (*Glazing-Type*)>
3.- <@ CONDENSATION-PATTERN-P>
4.- <@ @@ (QUOTE RH$_i$) (QUOTE SET) 30.0>

The first type of clause is simple fact. For this clause to be true the fact must be known. The second type of clause has a variable (eg. *Glazing-Type*). For this clause to be true the variable *Glazing-Type* must be bound. If *Glazing-Type* is set to *Triple-Glazed* then the clause is true and assumes the value *Window Unit Is Triple-Glazed*. The third clause is an example of a clause that executes a function. The '@' character at the beginning of the clause indicates executable LISP code. In this case WINDEKS will execute a function called CONDENSATION PATTERN P. The 'P' ending identifies a function that is to be executed only once. The fourth clause executes a function (@@) that binds the variable RH$_i$ (indoor relative humidity) to 30.0.

3.5.3 Rule Modules

Theoretically, the knowledge base in a rule based system is an amorphous collection of production rules. As the number of rules grows beyond 30, the rule base becomes increasingly difficult to understand and maintain. It proved useful to impose a module structure on the rule base to support updating and editing.

3.6 Developing the Interface

The gateway into this system is through the user interface. With most Lisp workstations, high resolution monochromatic graphics are well supported and can be easily accessed through lisp functions. Choosing items from menus or graphic menus obviates the need for any keyboard interaction. With increasing system familiarity, this becomes too slow and a second level should be supported for taking short-cuts.

Initial versions of the consultant used the windowing capability of the machine to excess. The number of concurrent outputs (a query window, a system status window, a rule tracking window and a report window) proved much too confusing for first time users. Consequently, only a report and query window were retained for the existing version. Subsequent tests using PC based KBS shells with simple query screens indicated that a user interface should contain a minimal amount of information, but have deeper knowledge available on request (6).

4. Conclusions

The goal of developing a prototype system was met. It remains for it to be implemented in a consulting service providing answers to public queries about building problems. Evaluation of the system will take place once it has been put into a consultant postion. Methods for KBS "validation" have not yet been developed that will ensure correct answers. Only field implementation and constant exposure to real-world problems will bring out inconsistencies and changes in the knowledge base.

Knowledge acquisition techniques particularly in the building domain are ad hoc. Automated methods are well suited to consensus and criteria development but not to determining the heuristics of diagnosis. Gaming and unobtrusive evaluation of problem solving are possible future directions.

For diagnosis, rule-based paradigms have been well established in other fields (eg., medicine) and transfer well to the building envelope diagnosis domain. It is the integration of diagnosis, selection and design that will limit the usefulness of this approach and demand a re-evaluation of the representation to an "object oriented" scheme as is common in AI systems and languages such as Xerox's Smalltalk (7.

Although the interface is crucial to the usefulness of a system, few documented guidelines exist for the design of adaptive and evolving interfaces. When designing interfaces only for neophyte users the system designer must realize that the user quickly outgrows the "user friendly" paradigm and looks for structural simplicity and short cuts.

Judging from our experiences in developing this system, strategic decisions about subject area, development procedure and final implementation are crucial. A small well defined problem area like windows, with in-house expertise was essential in this first prototype. Well supported LISP environments provide all the necessary tools for knowledge engineers to try various representations and quickly build interfaces. Finally, a well defined client, a hotline for public enquiries, is necessary for final deployment and evaluation.

Acknowledgments

We are grateful for the time and effort of the contributing experts: Mr. R.P. Bowen of IRC, Mr. M. Glover, and Dr. R. Brand. We thank IRC for their development support and we particularly thank XEROX for their support and machine use.

References

1. J.H. Boose, **Expertise Transfer for Expert System Design**, ADVANCES IN HUMAN FACTORS/ERGONOMICS 3, Elsevier, NY, NY, 1986. p. 26.

2. E. Chang, **Participant Construct System User Manual**, Alberta Research Council, ARC-CPCS-PCS-Manual, Feb. 1986, Calgary, AL. pp. 38.

3. M.L.G. Shaw and C. McKnight, **Think Again: Personal Decision Making and Problem Solving**, Prentice Hall, 1981.

4. B. and W. Thompson, "Finding Rules in Data", **BYTE**, Vol.11, No.12, McGraw Hill, NY, NY, Nov. 1986, p. 149-158.

5. "Chart", **The Spang Robinson Report, Vol. 2, No.10**, Spang Robinson, Palo Alto, CA. USA. Oct. 1986. p.22. The report cites major KBS development shells like KEE, ART, Knowedgecraft, Insight 2 et. al. as supporting classification tree development.

6 A. A. diSessa, "A Principled Design for an Integrated Computational Environment.", **Human Computer Interaction**, Vol.1, No.1, 1985, LEA, London, 1985, pp.1-47.

7 A. Kay and A. Goldberg, "Personal Dynamic Media", **Computer**, Vol 10, No.31, p.41.