Expert systems for engineering codes

Autor(en): **Sharpe, Ron / Marksjö, Bertil**

Objekttyp: **Article**

Zeitschrift: **IABSE reports = Rapports AIPC = IVBH Berichte**

Band (Jahr): **58 (1989)**

PDF erstellt am: **21.06.2024**

Persistenter Link: https://doi.org/10.5169/seals-44925

Expert Systems for Engineering Codes

Systèmes experts pour les normes en génie civil

Experten-Systeme für Bauingenieurnormen

Ron SHARPE
Senjor Princ. Res. Scientist
CSIRO
Melbourne, Australia

Ron Sharpe, born 1943, got his PhD in civil engineering degree at Southampton University after graduating from Melbourne University. He now leads a knowledge based systems research group.

Bertil MARKSJÖ
Princ. Res. Scientist
CSIRO
Melbourne, Australia

Bertil Marksjö, born 1939, graduated from the Royal Institute of Technology in Stockholm and followed up with a PhD. He is deputy leader in the same knowledge based systems research group.

SUMMARY

Both users and developers of complex engineering codes have much to gain from the use of expert systems to help ensure correct interpretation and avoidance of obscurities, contradictions and omissions. The paper discusses the development of PC based expert systems for the Australian wind and building codes including the problems encountered. The former system is complete and being introduced into design offices.

RESUME

Dans le cas des normes en genie civil, l'utilisation d'un système expert peut être profitable aussi bien aux utilisateurs qu'au personnes en charge de les établir, en les aidant à garantir une interprétation correcte du code et à éviter les contradictions, les omissions ou les parties obscures. Ce document discute les développements de deux systèmes experts sur PC, l'un étant lié aux standards australiens du bâtiment, le second à ceux des effets du vent sur les édifices. Le développement de ce dernier a maintenant été complété et le système est actuellement présenté à des bureaux d'études.

ZUSAMMENFASSUNG

Um Bauingenieurnormen besser zu interpretieren und Unklarheiten und Widersprüche zu vermeiden, können Experten-Systeme eine wichtige Rolle spielen, sowohl bei den Benutzern als auch bei den Sachverständigen selber. In dieser Veröffentlichung wird die Entwicklung von Experten-Systemen für die australischen Wind und Baunormen beschrieben, einschliesslich der dabei auftretenden Probleme. Das WINDLOADER-System wird zur Zeit in Ingenieurbüros eingeführt.

1. INTRODUCTION

The problems in developing codes are well known [1] and expert systems can reduce these as well as assist users [2,3]. For the developer they encourage greater precision in code specification, and exposure of inconsistencies, omissions and ambiguities. For the user they offer accurate and thorough checking of relevant clauses faster than possible by hand, plus an ability to explore more design alternatives.

In spite of these benefits few expert systems for codes (and also for engineering in general) have been completed and implemented even though many small prototypes have been developed in academic and research institutions. One of the reasons is that the development of full-scale systems usually requires extensive resources well beyond that required for the initial prototype and these are usually not available. Also many engineering systems combine logic with mathematical, database and graphical procedures. While expert systems are designed to handle logic, they also need to be integrated with the other procedures in order to achieve wider usage [4].

2. WINDLOADER

2.1 History of WINDLOADER

Over the past five years WINDLOADER has been developed in parallel with the latest revision of the Australian wind loading code, AS 1170.2 [5]. The code is complex and a study [6] has shown that designers can make significant errors in its interpretation. This code was previously revised in 1975 and 1983, and the latest version represents a major revision involving a Simplified Procedure for small buildings less than 15 metres in height, and Detailed Procedures for both static and dynamic analysis. WINDLOADER is restricted to the Detailed Static Analysis section since this is where most code users are likely to need assistance. (WINDLOADER may be extended to cover dynamic analysis in the future if there is sufficient demand.) Most of the static analysis section has also been incorporated into the New Zealand loading code, and it is expected that a version of WINDLOADER will be developed for that country.

The complexity of the code has increased substantially in the latest version and users are expected to experience much difficulty, especially in complicated applications, in the next year or so with the text version of the code. A series of seminars and a code commentary have been prepared to assist in the transition period, and WINDLOADER is also expected to greatly assist users.

The development of WINDLOADER began as a prototype system [7] coded in Melbourne University Prolog in 1985 on a HP9000/540 computer, followed by conversion to Prolog-2 on an IBM PC-AT computer with a Professional Graphics Display in 1986. The prototype (which featured colour graphic menus and displays) generated much interest and proved very useful in attracting both funding support and involvement of experts in the development of a commercial system in 1987. At this stage Standards Australia conducted a survey of potential users and found that 85% of potential users had access to an IBM PC or compatible computer, and so it was decided that WINDLOADER should be developed for the PC market. However this meant that Prolog alone would be unsuitable for a system as large as WINDLOADER since it would not be capable of fitting into a PC. It had also been found during the prototype development that Prolog was too slow for recursion and too cumbersome for processing graphics, equations and table interpolations.

In 1987 it was decided to develop an in-house shell (called BX-Shell) written in C and use this for WINDLOADER since no suitable commercial PC shell could be found at that time with the necessary range of functions, speed and low delivery cost. The BX-Shell included special C functions for

processing graphics, equations and tables [8]. The shell also included BX-Prolog for handling logic. BX-Shell was developed on a Sun 3/50 workstation with the intention of later porting it down for use on PCs.

2.2 Use of Crystal Shell

In early 1988 the Crystal PC shell became available and was quickly adopted in place of the BX-Shell since Crystal met most of the desired development and delivery features required. Crystal is written in C and while it did not possess all of the functions required by WINDLOADER for equations and table handling, it was possible to quickly port these across from the BX-Shell.

Crystal is a rule-based shell and this suited the format of the wind code which is also partly rule based.

2.3 Knowledge Representation

While the printed code is intended to be exhaustive and self-explanatory, development of the expert system required that the expert had to be consulted several hours per week for code interpretations over the development period. The printed code deals with a complex domain and even experts may need to spend many hours to ensure that correct interpretations of difficult sections are made for the full range of possible cases.

The wind code is partly rule based and partly procedural. The procedural sections include extensive use of mathematical equations and table interpolations, many of which are non-linear and not continuous. Such procedural sections are often tightly interwoven into the code logic and it is frequently very difficult to encode the knowledge into an easily readable format. This makes it difficult to achieve a close correlation between the expert knowledge (the printed code) and the knowledge representation in the expert system.

For example, consider the following subset of clauses from Section 3.2.6 for determining changes in terrain category:

> *Fully developed gust windspeed multipliers (Mz,cat) only apply at a structure site when the terrain category at the site is uniform upstream for a distance greater than (2500 + xi) metres.*
>
> *When the immediate upwind terrain extent is less than (2500 + xi) metres, corrected windspeed multipliers (Mx) shall be computed using Equation 3.2.6(3).*
>
> *Notwithstanding this requirement, the extent of upwind terrain to be considered need not exceed the larger of either 2500 or 50 times the structure height (ht), provided that the terrain at that limit is Terrain Category 3 or less rough, (assume the windspeed multiplier (Mo) to be the value for fully developed terrain at that limit).*
>
> *If the terrain at that point is rougher than Terrain Category 3, the upwind limit shall be extended until Terrain Category 3 or terrain of less roughness is encountered, or alternatively fully developed Terrain Category 3 may be arbitrarily assumed upwind of that point.*

The logic of these clauses is complex and first it is necessary to simplify the structure in terms of logical operators (IF, THEN, NOT, etc.) and subclauses (A,B,C, etc.) before rearranging it for the expert system. This gives:

THEN.....A.. IF .....B....
IF .....C.. THEN .....D....
NOT WITHSTANDING ....C...,
THEN .....E... OR .....F...,
IF .....G... OR .....H....
IF NOT(. .G... OR .....H..),
THEN .....I.... OR .....K... OR ...L...

The terms 'NOT WITHSTANDING' and 'NOT(...)' can be removed and the qualifying 'IF....' statements are best placed ahead of 'THEN ...' statements to avoid unnecessary evaluation of the latter if the 'IF...' tests fail. This leads to a simpler and more consistent form:

IF .....B..... THEN ....A....
IF .....C..... THEN ....D....
IF .....G..... OR .....H...,
THEN .....E.... OR .....F...,
ELSE ....I..... OR .....K... OR ...L...

Further changes are required when the subclauses are expanded. Also while it is not stated, a recursive procedure is implied in the last clause (to see if the building penetrates more than one layer arising from upstream changes in terrain roughness). After recursion is included, the logic may be represented in pseudo code as shown in Figure 1. This can be accessed by the user as a WINDLOADER 'help screen'. Finally the actual coding of the clauses in Crystal is different again because of the need to evaluate equations and to store results speedily and efficiently.

```
        ASSUMPTIONS BEHIND CHANGES IN TERRAIN CATEGORY
                                refs  3.2.6 and E3.2.6

- IF upstream terrain category, Cat, is fully developed at height, Z m
  THEN use multiplier M(Cat,Z).

- IF upstream terrain category, Cat, is undeveloped at height, Z m
  THEN find next terrain further upwind,
     AND interpolate between the upstream multipliers.

- IF next terrain further upwind is undeveloped,
     AND this terrain change occurs beyond point D
         located max(2500,50*Structure height) m from Structure,
     AND terrain category is 3 or less rough at D
  THEN assume a fully developed terrain.

- IF next terrain further upwind is undeveloped,
     AND this terrain change occurs beyond point D,
     AND terrain category is more rough than 3 at D
  THEN find next terrain further upwind until category is 3 or less rough,
     OR arbitrarily assume a fully developed category 3 upwind.

          | ↑    |  Sections ▪▪          | ↓    |          |End      |
          |PgUp  |  Pages    ▪▪          |PgDn  |          |to Quit  |
```

Fig. 1. Changes in Terrain Category HELP screen.
(This and the following images have been dumped using the PrtSc/PrintScreen key to a dot matrix printer. The color and highlight information is not shown).

The code developers anticipated difficulty would be experienced in this and other sections, and subsequently produced examples in a code commentary which were checked by WINDLOADER.

2.4 User Interface

The design of the user interface required much thought and experimentation. Many of the earlier PC expert shells tended to have either scrolling screens or a sequence of screen images with questions and explanations. Often such systems offer the user little control over the order in which the knowledge base is accessed. In deep systems (with many levels of rules), it is easy for the user to lose his overview and become confused by the order of questions. Later shells have introduced menu formats and pop-up windows and these will become more widely available soon.

A menu-based approach is adopted in WINDLOADER. This appears particularly suited to the design environment where users need as much flexibility as possible to test and modify building shapes, location, orientation and so on. Menus are useful for displaying items for selection, data inputs including tables and results. Users can generally move freely around such menus and select items in any order. A HELP item appears on most menus and this enables one or more explanatory screens to be displayed if needed. These are illustrated in the example of the effect of terrain changes on a 50 m building. The building is in a central business district area surrounded by tall buildings (Terrain Category 4) and there are changes to lesser terrain categories in the east and north directions as shown in Figure 2. Further help screens are provided to enhance the explanation of terrain changes including a diagram (Figure 3).

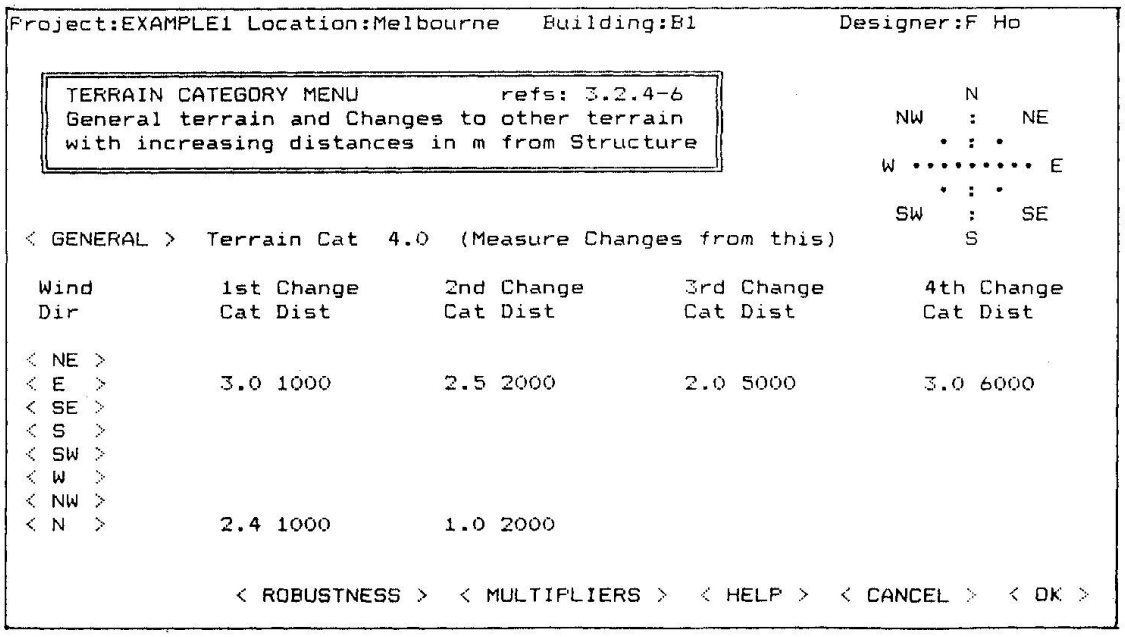

Fig. 2. Input screen for input and checking Terrain Category and changes.

If the MULTIPLIER button is selected in Figure 2, the user will be shown the factors by which the basic wind speeds will be multiplied as a result of the terrain changes. Figure 4 shows that while the protection offered by surrounding buildings in the central business district reduces the basic wind gust speeds by 10 per cent (i.e. to 0.9) at the top of the building, the exposure to the east and north increases the speeds by 7 and 17 per cent respectively. The user can experiment to see how many terrain changes need be considered before the changes in the multipliers become insignificant. The ROBUSTNESS button provides further information on the relative effect of each terrain change. The user can also test the changes in building height and location if wind loads need to be reduced to achieve cost savings or increased safety, especially in cyclone areas.

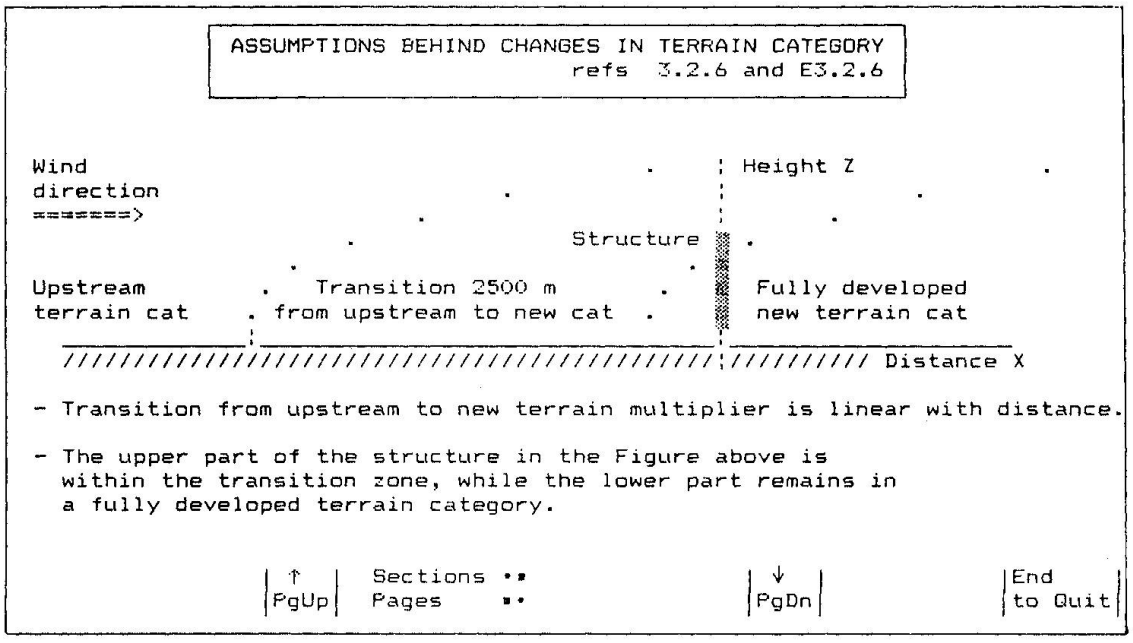

Fig. 3. HELP screen for diagrammatic explanation of terrain change effects on a structure.

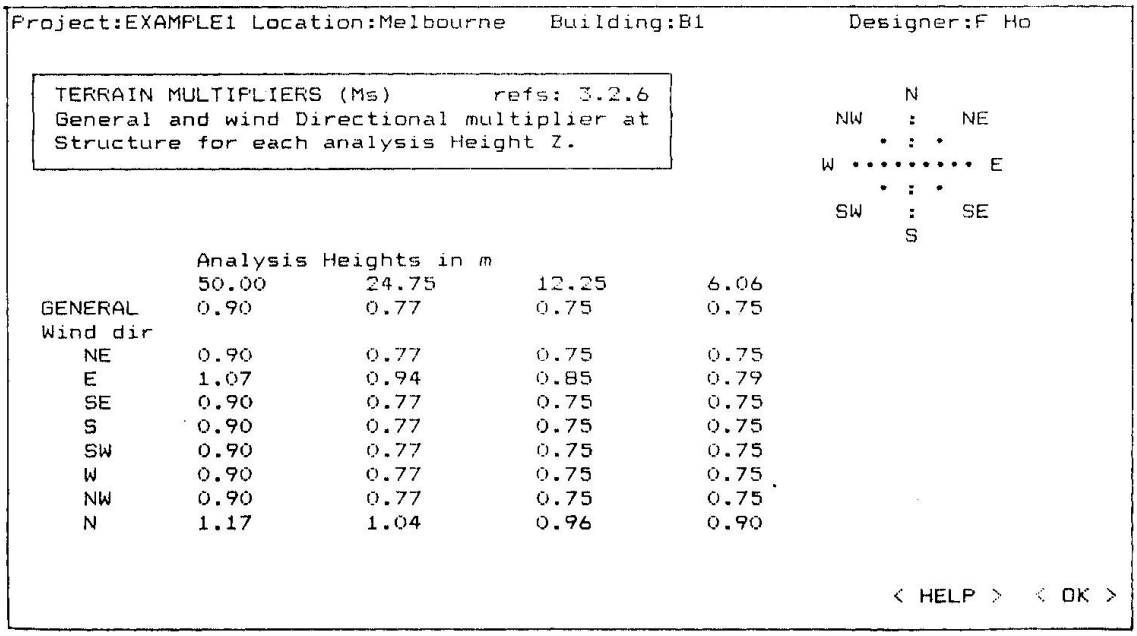

Fig. 4. MULTIPLIERS explanation screen showing impact of terrain changes in east and north directions.

Most of the user interface tends to be procedural as in algorithmic software systems including the HELP screens which are of a fixed format. While Crystal permits the underlying logic to displayed as a HELP screen at any stage by pressing the F1 function key, this key has been disabled in the final version since the coded rules are generally meaningless except to the knowledge engineers. However this ability to check the logic is crucial to the knowledge engineers during program development and is a major advantage over algorithmic software systems

2.5 Testing and User Acceptance of WINDLOADER

Just as with algorithmic software, WINDLOADER must prove itself to be cost effective in the design office in order to gain widespread use. As well as having key experts on the advisory committee, five practising engineers have closely assisted in the design of the user interface and testing of the software. Four prototypes were released over the development period and tested on routine designs ranging from houses to multistorey buildings.

The code developers were greatly assisted by the development since over 20 logical inconsistencies, ambiguities and omissions were detected. All of these would probably have remained obscure until the code was released and engineers started to use it since such errors can be very difficult to spot from reading the code. Omissions were the main problem and the programming style of expert systems makes detection of these more obvious than algorithmic programming unless a truth table type approach is adopted.

In one case a contradiction was also discovered in which three successive tables gave different pressures on a flat roof of a symmetrical square building for the same wind speed from different directions. As a result, columns had to be removed from two of the tables and a new column added to the third.

A benefit of the software development is that WINDLOADER can more readily evaluate the full range of wind directions and resulting pressures on building faces whereas this would be too time consuming by hand. For example, in the case of 8 wind directions with different wind speeds for the capital cities, a building with 6 faces (4 walls and 2 pitched roof faces), may have in excess of 48 pressure combinations, all of which can be calculated by WINDLOADER. This will allow all pressure combinations with dead and live loads to be considered in order to determine the worst loads on a structure. However users following the code manually are likely to only consider a few of the pressure combinations and hence they can never be certain if the worst loading cases have been overlooked.

3. BUILDING CODE OF AUSTRALIA (BCA)

An expert system for the Building Code of Australia is being developed in collaboration with the Australian Uniform Building Regulations Coordinating Council (AUBRCC). The BCA will be simultaneously restructured to become a performance-based code and the project is expected to take about three years. A demonstration prototype for a small part of the code has been developed using the Crystal PC expert system shell and this was instrumental in AUBRCC deciding to collaborate in further development.

The BCA does not have any complex equations like WINDLOADER but this is offset by the greater size of the BCA. The BCA is also undergoing extensive logic restructuring and one of the key experts assisting with this restructuring will also be assisting with the expert system. The development of the prototype has indicated that while the BCA is rule based and thus could be

directly coded into a rule-based system with depth-first search, it is much more convenient for the user if a menu-based format is adopted to allow much faster processing and removal of unnecessary questions.

4. CONCLUSION

The project has proved that a PC-based expert system with a reasonably fast response time can be developed for a relatively complex design code and integrated with algorithmic procedures. However some of the features desired in an expert system, such as reasoning from a transparent knowledge base, have to be compromised as a result of having equations, table interpolations and large arrays of data to be stored and processed. As a result the user can only access formatted help screens but this is offset by the convenience of other features such as graphics to enhance explanations and a more user friendly interface via menus.

It would not have been possible to complete this work in reasonable time without the use of a commercial expert system shell which permitted extensions to be added in C. Other more powerful shells are becoming available and it is felt that these will make the task easier in future, especially for those shells using a common portable language such as C.

ACKNOWLEDGMENTS

The authors are grateful to Standards Australia and its subcommittee, BD/6/2, plus the Department of Industry, Technology and Commerce for their support of this project over the 1987-89 period. Speeial thanks are due to Ms F. Ho, Dr J. Holmes, Ms C. Jeammes and the SA/CSIRO WIND-LOADER advisory committee for their extensive efforts in making the project a success.

REFERENCES

1. FENVES S.J., RANKIN K. and TEJUJA H.K., The Structure of Building Specifications, NBS Building Science Series 90, National Bureau of Standards, Washington DC, 1976.

2. MARKSJÖ, B.S. and HATJIANDREOU, M., Developing Knowledge-Based Systems for Building Design Approval, Proceedings of the I.E.Aust National Engineering Conference, Melbourne, March, pp.206-210, 1985.

3. STONE D. and WILCOX D.A., Intelligent Systems for the Formulation of Building Regulations. Proceedings of the 4th Int. Symp. on Robotics and Artificial Intelligence in Building and Construction, Israel, 22-25 June 1987.

4. FENVES S.J., What is an Expert System, in Expert Systems in Civil Engineering (eds Kostem, C.N. and Maher, M.L.), American Society of Civil Engineers, New York, pp. 1-6, 1986.

5. STANDARDS AUSTRALIA, Minimum Design Loads on Structures, Part 2 - Wind Forces, AS 1170.2, 1989.

6. MELCHERS R.E., Human Error in Structural Analysis, Proceedings of Seminar on Quality Assurance, Codes, Safety and Risk, Dept of Civil Engineering, Monash University, pp.91-94, 1984.

7. LE TEXIER J.Y., DOMAN A., MARKSJÖ B.S. and SHARPE R., Use of Prolog with Graphics including CAD, Proceedings of Ausgraph 85, Third Australian Conference on Computer Graphics, Brisbane, pp.81-85, 1985.

8. THOMSON J.V., BIRD S., THOMSON D., MARKSJÖ B.S. and SHARPE R., Extending Prolog to Provide Support for Design Code Expert Systems, Microcomputers in Civil Engineering, 3, pp.93-109, 1988.

Leere Seite
Blank page
Page vide