Form R-396 (1/1/87)

Standard Title Page -- Report on State Project

541

Report No.	Report Date	No. Pages	Type Report: Final	Project No. : 9510
VTRC90-R1	Sept. 1989	49	Period Covered: Nov. 87 - Sept. 89	Contract No.:

Title and Subtitle
An Expert System as Applied to Bridges:
Software Development Phase

Author(s)
Zuk, W. & Newbrough, J.

Performing Organization Name and Address

Virginia Transportation Research Council
Box 3817, University Station
Charlottesville, Virginia 22903-0817

Sponsoring Agencies' Names and Addresses

Va. Dept. of Transportation	University of Virginia
1221 E. Broad Street	Charlottesville
Richmond, Virginia 23219	Virginia 22903

Key Words
Abandon
Bridges
Expert System
Improve
Maintain
Rehabilitate
Replace

Supplementary Notes

Abstract

This report describes the results of the third of a four-part study dealing with the use of a computerized expert system to assist bridge engineers in their structures management program. In this phase of the study, software (called DOBES) was written incorporating the knowledge base presented in the second phase of the study (VTRC Report No. 88-R6, November 1987) relating to the disposition of old bridges. DOBES provides recommendations as to whether a bridge should be rehabilitated, improved, replaced, abandoned, or simply maintained. A future study will test, evaluate, and refine this program for eventual operational use.

FINAL REPORT

AN EXPERT SYSTEM AS APPLIED TO BRIDGES: SOFTWARE DEVELOPMENT PHASE

William Zuk
Faculty Research Scientist

and

Jonathan Newbrough
Highway Research Scientist Assistant

Virginia Transportation Research Council
(A Cooperative Organization Sponsored Jointly by the Virginia Department of Transportation and the University of Virginia)

Charlottesville, Virginia

September 1989
VTRC 90-R1

The research reported here, because of the scope, does not fall within the purview of any currently established advisory committee. However, the research was supported by the Council administration in connection with the conceptual research mission.

ABSTRACT

This report describes the results of the third of a four-part study dealing with the use of a computerized expert system to assist bridge engineers in their structures management program. In this phase of the study, software (called DOBES) was written incorporating the knowledge base presented in the second phase of the study (VTRC Report No. 88-R6, November 1987) relating to the disposition of old bridges. DOBES provides recommendations as to whether a bridge should be rehabilitated, improved, replaced, abandoned, or simply maintained. A future study will test, evaluate, and refine this program for eventual operational use.

FINAL REPORT

AN EXPERT SYSTEM AS APPLIED TO BRIDGES: SOFTWARE DEVELOPMENT PHASE

William Zuk
Faculty Research Scientist

and

Jonathan Newbrough
Highway Research Scientist Assistant

INTRODUCTION

This four-part expert system study was initiated in 1986 with an overview entitled "Expert Systems as Applied to Bridges and Pavements" (VTRC Report No. 86-R31). The first phase was followed by a study in 1987 and 1988 of specific applications to bridges, which was entitled "Expert Systems as Applied to Bridges" (VTRC Report No. 88-R6). Phase three deals with the development of a computer software program recommending to a user whether to rehabilitate, improve, replace, abandon, or simply maintain an old highway bridge. Since thousands of such old bridges have to be evaluated yearly, an expert system of this nature would be of considerable value to VDOT.

The fourth and last phase of the study, to follow the current one, will test and refine the system so that it can become operational.

GENERAL PROCEDURE

In the 1988 report "Expert Systems as Applied to Bridges: Knowledge Acquisition Phase," a detailed knowledge base was created. It was obtained from VDOT bridge engineers, and it is based on how decisions are reached with regard to the disposition of old bridges. These decision procedures have been formulated into a set of complex rules, which is required by expert system analysis.

Writing an expert system program from scratch appeared to be a formidable task, so attempts were made to acquire a suitable expert system shell from outside sources into which it was hoped that this study's knowledge and inference requirements could be inserted. Unfortunately, none of the numerous shells investigated proved to be useful for this study.

By good fortune, a computer programmer was found who already had considerable experience in developing expert systems. He undertook to write a program specifically for this project, using the computer language known as

LISP. He has also written three reference manuals detailing the operation of this program, which is called DOBES (Disposition of Old Bridges Expert System) (see the Appendix). The first of these manuals is a general user manual, the second is a developer user manual, and the third is a technical reference manual.

DOBES was written to be a useful and friendly program. It operates on any IBM or compatible personal computer with one or more floppy disk drives. Essentially, all the user has to do is answer a series of questions about the attributes of the bridge under investigation. All of the questions designated as "primary" must be answered. However, those designated as "secondary" should be answered if possible; but if this is not possible, the program will still provide a recommendation regarding the disposition of the bridge, followed by a list of any unknown secondary attributes. The user can then exercise his or her own judgment as to how important the missing information is. If it is deemed to be important with respect to the recommendation stated, then the missing information should be ascertained and the program should be rerun.

The entire expert system program is contained on two 5-1/4 in floppy disks: one for the program itself and one for inputted data. They cannot be distributed at this time because certain licensing agreements have yet to be worked out.

DOBES provides a recommendation regarding a given bridge as to one of five possible courses of action. They are as follows:

1. Rehabilitate: To rehabilitate a bridge is to restore it to its original condition. The recommendation may stipulate rehabilitation of the deck, the superstructure, the substructure, or any combination of the three.

2. Improve: To improve a bridge is to improve its characteristics over and above its original condition. This is usually either by strengthening, widening, or both.

3. Replace: To replace a bridge is to remove or bypass the old bridge and replace it with a completely new one on the same or on a new alignment.

4. Abandon: To abandon a bridge is to take it out of service from the VDOT highway system. An abandoned bridge may be destroyed or put to a new use, possibly at another site.

5. Maintain: To maintain a bridge is to keep it functioning essentially in the condition that it is in. The recommendation may call for maintenance of the deck, the superstructure, the substructure, or any combination of these three components.

The rules used to make the recommendation can be called forth by command. In addition, with the use of a printer attached to the computer, anything appearing on the monitor screen at any stage can be printed out.

Because of time and budget constraints, the following limitations had to be imposed on the current DOBES program.

- o A few of the rules are couched in overly precise terms. For example, several rules set the ADT (average daily traffic) at a precise 100. Obviously a count of 99 or 101 would make little difference in the final result. However, an incorporation of "fuzzy logic" in the program would add considerable complexity, which is not warranted in this initial program.

- o Interactive questions and answers concerning how the rules were arrived at, where data can be obtained, or the reasoning behind the inference have been left out. Should such information be necessary in an operational program, it could be added at a later date. Much of this information, however, already is available in the report of the knowledge acquisition phase of this study (VTRC report 88-R6, November 1987, Charlottesville, Virginia).

- o A few relatively rare bridge types have not been included, such as covered wooden bridges, arch bridges, cable-suspended bridges, and plastic bridges. So few of these exist in Virginia that it is believed these special ones could be handled individually.

  The program is applicable to all single or multiple span girder or truss highway bridges of wood, metal, or concrete (either normally reinforced or prestressed). If in a multiple span bridge, one span (or more) is of a different basic material and/or of a different structural type from the rest of the bridge, that span or spans should be treated as if it were a separate bridge. The remaining span or spans would then also be treated as a separate bridge.

- o Recommendations concerning which of the many old bridges deserve expenditure of funds in a given year are not part of this program. Such recommendations would require that all of the old bridges be evaluated by this DOBES program, after which the total required costs would be compared to the total available funds on some rational basis such as the current bridge management system using deficiency point ratings as a guide to prioritization. An expert system prioritization is quite possible, but until it is done, the current method in use will work, given some discretion.

Should changes in any of the rules or attributes be needed, including additions or deletions, they can be made following the procedures outlined in the documentation manuals in the Appendix. Such provisions should enable DOBES to stay up to date for ongoing operational use in the future.

INFERENCE PROCEDURE

The recommendation generated by this expert system is arrived at by applying each of the separate rules to each of the separate attributes of the bridge. However, some rules conflict or overlap with others, and some rules and attributes are more important than others. It was therefore necessary to develop an inference or evaluation process to deal with these factors. A multiweighting system was chosen.

First, each attribute and rule is designated as primary (P) or secondary (S). These determinations are based on the judgments of the bridge engineers involved in this study. Each status of P or S can be changed if necessary by minor reprogramming of DOBES. When the rules are applied to the attributes, the possible combinations of options (such as replace, rehabilitate, etc.) involve primary rule with primary attribute (PP), primary rule with secondary attribute (PS), secondary rule with primary attribute (SP), and secondary rule with secondary attribute (SS). For any given bridge, a list is made of the various resulting options generated in applying the rules to the attributes, along with their (PP), (PS), (SP), (SS) category.

To provide for a wide range of weighting factors, 15 constants (potentially adjustable) were applied in the following manner to the 5 basic options: rehabilitate, improve, replace, abandon, and maintain. For example, with regard to rehabilitation, a numerical constant _A_ is assigned as a multiplier or weighting factor to the sum of all the times on the option list cited that call for rehabilitation in the (PP) category. Then another numerical constant _B_ is assigned as a multiplier to the sum of all the times on the option list that call for rehabilitation in the (PS) or (SP) category. The categories (PS) and (SP) are assumed to be of equal weighting value. Still another numerical constant _C_ is assigned as a multiplier to the sum of all the times in the option list that call for rehabilitation in the (SS) category. By adding all the terms above involving _A_, _B_, and _C_, a new sum value _D_ is determined.

In like manner, four other sum values for the improve, replace, abandon, and maintain options can be determined.

The final recommendation would be the option with the largest numerical sum value. In the DOBES program as currently written, the constants _A_, _B_, and _C_ were chosen as 10, 7, and 3, respectively. Parallel constants for the improve option are 8, 5, and 2. For the replace option, the constants are 10, 7, and 3. For the abandon option, they are 6, 4, and 1. Finally, for the maintain option, the constants are 7, 5, and 2. The larger the constant, the more important is the related option. These constants were chosen intuitively as trial values. These values will be adjusted to more accurately reflect realistic conditions after further testing and evaluation in a study to follow.

Two checks related to cost are built into the program.

1. If the rehabilitation or improve option has the highest sum value, but the cost exceeds 75 percent of the cost of a replacement bridge, then replacement is selected as the final recommendation.

2. If the rehabilitate or improve option has equal sum values, and these sum values exceed all others, then the rehabilitate or improve option costing the lesser amount is selected as the final recommendation.

CONCLUSIONS

As a result of this phase of the study, an easy-to-use expert system was developed that enables a user to quickly and consistently determine the best course of action to take regarding the disposition of an old highway bridge. The five basic courses of action are rehabilitation, improvement, replacement, abandonment, and ordinary maintenance.

The system is designed to be used on any IBM or compatible personal computer with a floppy disk drive.

However, for the program to be operational, additional testing and evaluation should be undertaken as some changes and refinements may need to be made, particularly in adjustments of the various weighting factors. Details of such a testing program will be described in the Work Plan for phase four. It is anticipated that such testing will be completed by December 31, 1990.

APPENDIX

PART I--DOBES USERS MANUAL
PART II--DOBES DEVELOPER USERS MANUAL
PART III--TECHNICAL REFERENCE MANUAL

DOBES Users' Manual

Jonathan Newbrough

1 June 1989

556

Contents

1 Purpose

This document is intended to aid and instruct anyone wishing to use the Disposition of Old Bridges Expert System (DOBES). It contains general information about DOBES and specific instructions for the use of the program.

2 Overview

Many of the numerous bridges under the jurisdiction of the Virginia Department of Transportation (VDOT) are deficient in one way or another, and adequate funds to replace them all are not available. The safety and financial risks involved necessitate careful and correct decisions concerning their disposition. These decisions are now made by civil engineers in VDOT based on information about the bridge. Their criteria vary and they often disagree in their decisions. A uniform set of criteria for the state upon which the decision of disposition may be based would insure that VDOT uses available funds to its best advantage.

Dr. William Zuk, a faculty research scientist at the VDOT's Research Council, began a project to establish these criteria. Gathering information from state experts and current civil engineering publications, Dr. Zuk formulated a set of criteria. His findings are incorporated in DOBES and govern its results. DOBES evaluates bridge information supplied by the user and makes a recommendation for the disposition of a given bridge.

Five basic recommendations are possible:

- To rehabilitate (to restore the bridge to its original condition).
- To improve (to strengthen or widen the bridge over and above its original condition).
- To replace (to replace the old bridge with a totally new bridge).
- To abandon (to take the bridge out of service in the VDOT system).
- To maintain (to keep the bridge operational in its present condition).

Expert systems are so called because they are intended to replicate the decisions of experts in some field. DOBES should replicate the decisions of state experts. Since even the experts disagree, the user may not agree with the recommendation of DOBES. In any event, the system's recommendation should be taken as advice only—certain factors particular to a bridge may not have been considered. Overall, DOBES is a tool providing the probable recommendation of the majority of experts. Use of this information and final disposition of the bridge is still under the judgment of the user.

560

3 Program use

3.1 Use of fonts

Within this document and particularly in the following sections, typefaces or fonts are used to clarify intent. Any reference to a particular key appears in a slanted font, such as the *ENTER* key. Text exactly reprinted from the microcomputer screen will appear in a typewriter-style font, like this: `c:\DOBES`.

3.2 Installation

DOBES was written to work on most configurations of IBM and compatible microcomputers. It can be configured for two floppy drives or one floppy and a hard drive, for systems with or without a printer, and for color or monochrome (black and white) monitors.

To ease the installation process, specific installation instructions are provided on the DOBES distribution floppy disks. These instructions will guide you through the installation procedure for a particular machine and configure DOBES for features of your particular system. Read the file `READ.ME` on the first diskette for more information on the installation procedure.

3.3 Invocation

If DOBES has been installed on a hard disk, go into the `\GCLISP` directory and type `GCLISP`. If DOBES has been installed on floppy disks, insert disk one into drive A, make that the current drive, and type `GCLISP`. The system may take a minute or more to load, depending on the speed of the computer.

3.4 DOBES environment

The screen is generally divided into three areas. The top line or title, the midsection or main area, and the bottom or information line. The top line indicates the current action of the system. For example, when analyzing a record, the title line will display `Analyzing` record: *record name.* The bottom line indicates which keys may invoke special actions at that time. When the computer pauses, the bottom line tells which keys will provide the needed input. At most times, help is available on the screen by pressing the *H* key. When in an editing window, it is available by pressing the *ESC* key. Many actions can be canceled by pressing *ESC* instead of selecting an item from a menu.

3.5 Windows and menus

Information in DOBES is often displayed on the screen with a double border and title box. Windows display one or more lines of textual information. Edit windows display information and await input. Menus allow the user to select one item from a list of several possible choices.

Edit windows allow the user to input one or more lines of textual information. The amount of information is limited by the size of the window, usually the user will be allowed to enter exactly one or up to six lines of text. To enter information, type it from the keyboard. To make corrections, the *BACKSPACE* key deletes the character to the left of the cursor. The *DEL* key deletes the character at the cursor. The *INS* key inserts a space after the cursor. *F3* clears all text in the window. *F2* redraws the text in the window. *F1* accepts the text in the window as done and tells the program to continue. Pressing *ESC* causes the system to display helpful information on the screen—instructions for using the editing commands available. Pressing *ENTER* is equivalent to pressing *F1* when the window has only one line.

Menus display a column of items from which the user is expected to choose one. The current selection is highlighted to stand out from the others. To move the current selection up or down, press the arrow keys. To accept the current selection and go on, press enter. When more items are in the list than fit on the screen, the window will have << `more` >> on the top or bottom. To reach the next windowful of items, move the selection to the bottom item and press the down arrow again. To return to earlier windowfuls, move the selection to the top and press the up arrow again. Left and right arrows also skip between windowfuls.

3.6 Editing the database

The bridge information in DOBES is called the database. You may add, change, or delete entries in the database to alter this bridge information.

Add record To add a new bridge to the database, choose `Add record` from the main menu and supply information about the bridge to the system as prompted.

Change record To change information about a particular bridge, select `Change record` from the main menu. The system will then ask the user to select the record to be changed and the attribute (aspect) of that record to be changed. Then the system will ask for the new value of that attribute of the record.

Delete record To delete a record from the database, select `Delete record` from the main menu. The system will then ask the user to select the record to be deleted. Once deleted, the record is permanently lost from the system.

Inspect record To look at a database entry select `Inspect record` from the main menu. Again, the system will ask the user to select a record to inspect. Then the system will display the value of each of the attributes of the record.

List records To list the records in the database, select `List records` from the main menu.

Print record To print a record to the printer select `Print record` from the main menu. The system will ask the user to select the record to print. This command will not be available on systems without a line printer.

Once a database editing command is finished, the system returns to the main menu.

3.7 Analyzing data

To analyze a record and generate a recommendation, select `Analyze record` from the main menu. The system will ask the user to select a record to analyze. Results will be displayed on the screen and may also be printed on a printer, if available.

The first part of the analysis information is the rule application. Each of the rules, or criteria for the recommendations, are tested against the record. Those that apply are listed along with the recommendation for which the rule is a criterion and the weight of the information concerned (primary, secondary, or mixed primary and secondary information). Some attributes of a record are more important than others, and some criteria are more important than others, so each is given a rating of primary or secondary. These weights are used in determining the weights of each of the recommendations.

The next part of the analysis is the comparison of the relative weights of the possible recommendations. Numeric values represent weights—the higher the value, the better the recommendation relative to the others. Subrecommendations are also compared and shown.

The final recommendation is then made based on the point values and consideration of a few special criteria. There may also be a subrecommendation and comments or qualifications of the recommendation. The system then returns to the main menu.

3.8 Inspecting the rule base

The rules in DOBES may be inspected at any time. Inspecting the rules listed by the analysis of a record would help explain why a certain recommendation was made for a particular bridge. To inspect a rule, select `Inspect rule` from the main menu. The system will then prompt for a rule to inspect and ask if the results should be printed on the printer, in addition to the screen. Finally, the rule text and priority are shown and the system returns to the main menu. For more information about rules in DOBES, see the section below entitled "The knowledge base."

3.9 Online help

There are help screens provided to explain commands available to the user. Whenever a menu appears on the screen, the *H* key will bring up the appropriate screen. When in a text editing window, pressing the *ESC* key will display the help screen.

3.10 Leaving the program

To leave DOBES, select `Exit DOBES` from the main menu or press *ESC* from the main menu. The database will automatically be saved and restored the next time DOBES is run.

4 The knowledge base

4.1 General

This section is included in this manual to help the user understand how DOBES works. It is neither a complete tutorial on expert systems nor a technical reference manual for DOBES but rather an attempt to combine the basics of both. Many books are available on expert systems and related subjects of artificial intelligence, propositional logic, and knowledge base programming. For more information about DOBES see the *DOBES Technical Reference Manual.*

4.2 Rules in DOBES

DOBES stores all of its "knowledge" in the form of if-then rules. A hypothetical rule might be:

```
IF THE NUMBER OF LANES IS LESS THAN 2
THEN RECOMMEND WIDENING
```

This rule makes a specific recommendation that is weighted by the importance of the rule, the importance of the factors considered (in this case, the number of lanes), and a weight adjustment for the specific recommendation (widening).

The knowledge base of DOBES has about 40 such rules. The recommendations from all of these rules are then added up and compared, and a final recommendation is made.

In some cases, certain meta-rules are considered. Meta-rules are special rules that make recommendations that are not affected by the weighting system. For example, if the rules are all compared and the trial recommendation is widening but replacement is cheaper, replacement will be recommended. The meta-rule would be:

```
IF THE RECOMMENDATION IS WIDENING AND THE
COST OF REPLACEMENT IS LESS THAN THE
COST OF WIDENING THEN RECOMMEND REPLACEMENT
```

The procedure for breaking a tie and any special conditions to be considered before recommending bridge abandonment are also included as meta-rules.

4.3 Knowledge in DOBES

The rules and meta-rules in DOBES represent a logical compilation of information gathered at the Virginia Transportation Research Council. They are an attempt to formalize the judgments and decisions made by the VDOT's engineers. This process is hindered by several factors, not the least of which is that experts do not always know why they have made a certain decision. That is, many decisions are not a conscious comparison of factors but rather intuition or an educated guess. This makes it more difficult to identify the actual basis for the decisions made. The accuracy and reliability of DOBES and the degree to which it conforms to expert opinion is a matter to be tested by time and by use of the system.

DOBES Developer Users Manual

Jonathan Newbrough

1 June 1989

Contents

1 Purpose

This document is intended to aid and instruct anyone wishing to use the Disposition of Old Bridges Expert System (DOBES) developer program. It contains general information about the DOBES developer, specific instructions for the use of the program, and an overview of expert systems and the workings of DOBES.

2 Overview

Originally, DOBES was intended to be a simple program to display results from certain fixed test conditions against information from various bridges in the Commonwealth of Virginia. These conditions are the subject of a report by the Virginia Transportation Research Council entitled *Disposition of Old Bridges: Phase II—Knowledge Acquisition.*

This solution to the problem of disposition of older bridges in the Virginia highway system was certainly plausible, but the usefulness of a rigid system would depend entirely upon the accuracy of the conclusions of the original conditions specified. Rather than rely on this assumption, DOBES was built with these conditions, or rules, as external data to the program. As such, they can be modified and amended to correct for oversights in the original rules or replaced completely to engage DOBES in the solution of other problems. Thus, DOBES is a general expert system that can be applied to any rule base.

The DOBES developer is the program that allows the user to inspect and modify the rule base. Just as the DOBES user may work with the information about bridges, the DOBES developer user may work with the actual conditions and conclusions of that data.

The DOBES developer should not be used for regular bridge analysis. It would be more costly in terms of memory, disk space, and speed of execution to do so, and it would also risk modifying the rule base accidentally, thus destroying the validity of all future conclusions by the system.

3 Expert systems

An expert system is a program intended to replicate the decisions of an expert in some field. DOBES is intended to replicate the decisions of bridge experts and civil engineers concerning the disposition of old bridges in the VDOT highway system. The computer is programmed to consider and weigh the same factors considered by experts. In order to do this, several types of information about the basis of these decisions were found. This information is called the knowledge base and consists of a rough coding of some of the knowledge of experts in this field.

3.1 Attributes

An attribute is a feature of a bridge. For example, bridge length is an attribute, and the attribute definition in the computer will tell the computer that this attribute has a numeric value which lies within a specified range. Other types of attributes are not simple numbers: for example, the county in which the bridge is located or the material from which the bridge was constructed. In general, attributes have six components:

1. Name
   The attribute has a name within the system. For example, the bridge length attribute could be called bridge length. The names should indicate clearly the data conveyed by the attribute. No two attributes should have the same name.

2. Description
   The attribute may also have a description to further specify exactly what information the attribute is to convey. This description or explanation is not used by the system but may be used to provide the user of the expert system with additional information about what the attribute means.

3. Question
   When a new record is defined in the expert system, each attribute is given a value by the user. The user gives this value

in response to a question concerning the attribute. For example, "What is the overall length of the bridge?" would be an appropriate question for the bridge length attribute.

4. Data type
   The type of information the attribute is to contain may be one of:

   - SINGLE-LINE-TEXT. The value of the attribute is a single line of text typed by the user.
   - MULTI-LINE-TEXT. The value of the attribute is text edited in the text editing window and may be from one to six lines long.
   - NUMBER. The value of the attribute must be a number.
   - MULTI-CHOICE. The value of the attribute is one of the given choices.
   - Y-N. The value of the attribute must be YES or NO.
   - Y-N-UNKNOWN. The value of the attribute must be YES, NO, or UNKNOWN.

5. Range
   Data types are sometimes qualified with a range. The range specifies particular valid and invalid values for the attribute. For SINGLE-LINE-TEXT, MULTI-LINE-TEXT, Y-N and Y-N-UNKNOWN type attributes, there is no range value. For attributes of data type NUMBER, the range specifies the minimum and maximum values of the attribute. For MULTI-CHOICE attributes, the range is a list of valid choices.

6. Priority
   Some attributes are more important than others. Hypothetically, the strength of a bridge may be more important that local political concerns. To compensate for these differences, each attribute is weighted as primary (most important) or secondary.

3.2 Goals

Goals are all possible outcomes of the decision procedure of the system. For DOBES, the goals are improving, abandoning, maintaining, replacing, or rehabilitating the bridge. In addition, there are various subgoals allowing for more detailed decisions. For example, the goal improve had subgoals improve by widening and improve by strengthening. The goals are defined in the system with three components:

1. Name
   The goal, like the attribute, must have a name. Like the attribute, the name should be descriptive of the possible recommendation it represents. For example, if a certain goal is chosen by the system to mean that the system recommends that a bridge be replaced, then an appropriate goal name would be replacement. No two goals may share a name, nor may a goal and an attribute.

2. Explanation
   The explanation tells the user more information about a chosen goal. This is used to further specify the recommendation reached by the system.

3. Type
   The goal may be a parent goal (or normal goal) or a child goal (subgoal) of another goal. The difference is that if some factor considered by the system weighs heavily that a subgoal should be the final recommendation, its parent goal will also be recommended. For example, DOBES will not recommend improve by widening without also recommending the parent goal, improve. Child goals may have additional levels of subgoals.

4. Weights
   Each goal is given a list of three relative weights. These numeric values determine how differently primary and secondary priorities among rules and attributes are handled. The weights

of a child goal are automatically those of the parent, but parent goal weights must be specified. For more information about the use of these weights, see the section below entitled Recommendation Procedure.

3.3 Records

Records are not defined for the system until after the rules, but an understanding of what records are and how they function in the system is necessary to understand the definition and application of rules. Just as attributes and goals had six and three components, respectively, each of which conveyed certain information about that item, rules have one component for each attribute and the information conveyed by that component is a value of the attribute associated. For example, if attributes bridge length and construction material are defined as type NUMBER and MULTI-CHOICE, then a record would need to be defined with a value for bridge length that is a number within the range specified by that attribute, and a value for construction material that is one of the listed choices for that attribute. Each record contains a unique list of values for each of the attributes in the system. A record is the definition of a unit for which the expert system makes its decision. In DOBES, a record is a set of information about a particular bridge.

3.4 Rules

Rules are representations of simplified bases for decision. They are formal comparisons and tests made on data about a record (in DOBES, each record corresponds to a bridge) and result in recommendations about the record. To best understand rules, it is easiest to simply look at one. The text of DOBES RULE-7 reads:

```
if the deck condition rating is less than 6 then recommend
deck rehabilitation
```

This rule quite simply says that if the value of deck condition rating (an attribute of data type NUMBER) is less than 6, then the expert system should recommend the goal deck rehabilitation, which is a subgoal and implies that rehabilitation should also be recommended. This comparison is made, or the rule is invoked when some record is analyzed by the system. At this time, the value of the attribute in the rule is the value associated with that attribute in the record being analyzed, and the recommendation being made is a recommendation for that record. Rules have two components:

1. Rule text
   This is an if-then statement (as above) specifying some comparison between attributes and finally recommending some goal or performing some other action.

2. Priority
   Like attributes, some comparisons made by rules are more vital than others. One test may determine that a bridge is extremely unsafe and another may determine that it may have more of a tendency to erode over time. These two rules may be differentiated by priorities of primary (most important) and secondary.

4 Program use

4.1 Use of fonts

Within this document and particularly in the following sections, typefaces or fonts are used to clarify intent. Any reference to a particular key appears in slanted font, such as the *ENTER* key. Text exactly reprinted from the microcomputer screen will appear in typewriter-style font, like this: `c:\DOBES`.

4.2 Installation

DOBES was written to work on most configurations of IBM and compatible microcomputers. It can be configured for two floppy drives or one floppy and a hard drive, for systems with or without a printer, and for color or monochrome (black and white) monitors.

To ease the installation process, specific installation instructions are provided on the DOBES distribution floppy disks. These instructions will guide you through the installation procedure for a particular machine and configure DOBES for features of your particular system. Read the file `READ.ME` on the first diskette for more information on the installation procedure.

4.3 Invocation

If DOBES has been installed on a hard disk, go into the `\GCLISP` directory and type `GCLISP`. If DOBES has been installed on floppy disks, insert disk one into drive A, make that the current drive, and type `GCLISP`. The system may take a minute or more to load, depending on the speed of the computer.

4.4 DOBES environment

The screen is generally divided into three areas. The top line or title, the midsection or main area, and the bottom or information line. The top line indicates the current action of the system. For example, when analyzing a record, the title line will display `Analyzing` `record:` *record name.* The bottom line indicates which keys may invoke special actions at that time. When the computer pauses, the bottom line tells which keys will provide the needed input. At most times, help is available on the screen by pressing the *H* key. When in an editing window, it is available by pressing the *ESC* key. Many actions can be canceled by pressing *ESC* instead of selecting an item from a menu.

4.5 Windows and menus

Information in DOBES is often displayed on the screen with a double border and title box. Windows display one or more lines of textual information. Edit windows display information and await input. Menus display one or more items on separate lines and await the selection of one item.

Edit windows allow the user to input one or more lines of textual information. The amount of information is limited by the size of the window. Usually the user will be asked for either a single line or up to six lines of text. To enter information, type it from the keyboard. To make corrections, the *BACKSPACE* key deletes the character to the left of the cursor. The *DEL* key deletes the character at the cursor. The *INS* key inserts a space after the cursor. *F3* clears all text in the window. *F2* redraws the text in the window. *F1* accepts the text in the window as done and tells the program to continue. *ESC* provides a review of these commands on the screen. Pressing *ENTER* is equivalent to pressing *F1* if the window has only one line.

Menus display a column of items from which the user is expected to choose one. The current selection is highlighted to stand out from the others. To move the current selection up or down, press the arrow keys. To accept the current selection and go on, press *ENTER*. When more items are in the list than fit on the screen, the window will have << `more` >> on the top or bottom. To reach the next window of items, move the selection to the bottom item and press the down arrow again. To return to earlier windows, move the selection to the top and press the up arrow again. Left and right arrows also skip between windows.

- 578

4.6 Editing the database

From the main menu, the command `Edit database` will bring the program into the database menu. From this menu, the following commands alter the database of records on the system.

Add record. To add a new bridge to the database, choose `Add record` from the record menu and supply information about the bridge to the system as prompted.

Change record. To change information about a particular bridge, select `Change record` from the record menu. The system will then ask the user to select the record to be changed and the attribute (aspect) of that record to be changed. Then the system will ask for the new value of that attribute of the record.

Delete record. To delete a record from the database, select `Delete record` from the record menu. The system will then ask the user to select the record to be deleted. Once deleted, the record is permanently lost from the system.

Inspect record. To look at a database entry select `Inspect record` from the record menu. Again, the system will ask the user to select a record to inspect. Then the system will display the value of each of the attributes of the record.

List records. To list the records in the database, select `List records` from the record menu.

Print record. To print a record to the printer, select `Print record` from the record menu. The system will ask the user to select the record to print. This command will not be available on systems without a line printer.

Once a database command is finished, the program returns to the database menu. To return to the main menu, press *ESC*.

4.7 Analyzing data

To analyze a record and generate a recommendation, select `Analyze record` from the main menu. The system will ask the user to select a record to analyze. Results will be displayed on the screen and may also be printed on a printer, if available.

The first part of the analysis information is the rule application. Each of the rules, or criteria for the recommendations, are tested against the record. The rules that apply are then listed along with the recommendation. Some attributes of a record are more important than others, and some criteria are more important than others, so each is given a rating of primary or secondary. These weights are used in determining the weights of each of the recommendations.

The next part of the analysis is the comparison of the relative weights of the possible recommendations. Numeric values represent weights, and the higher the value, the better the recommendation relative to the others. Subrecommendations are also compared and shown.

The final recommendation is then made, possibly with a subrecommendation, based on the point values and consideration of a few special criteria. Comments and qualifications of the recommendation may be displayed. The system then returns to the main menu.

4.8 Editing the rules

From the main menu, the command `Edit rules` will cause the system to display the rule menu. From this menu, the following commands allow the user to alter the rule base.

Add rule. To add a rule to the knowledge base, select `Add rule` from the rule menu and input the rule text and the priority.

Delete rule. To delete a rule from the knowledge base, select `Delete rule` from the rule menu and select the rule to be deleted.

Change rule. From the rule menu, `Change rule` will first prompt for a rule, then for the aspect of the rule (rule text or priority) to be changed, and finally for a new value of the chosen aspect.

List rules. To list the rules in the knowledge base, select `List rules` from the rule menu. Press *ENTER* when done.

Inspect rule. To inspect a rule, select `Inspect rule` from the rule menu, then select a rule to inspect. The system will show the rule text and priority of the chosen rule.

Print rule. From the main menu, `Print rule` followed by a rule selection will print the chosen rule to the printer.

After a rule editing command, the program returns to the rule menu. To return to the main menu, press *ESC*.

4.9 Editing the attributes

From the main menu, the command `Edit attributes` will cause the system to display the attribute menu. From this menu, the following commands allow the user to alter the attribute definitions in the system.

Add attribute. Define a new attribute for the system.

Delete attribute. Delete the definition of an existing attribute.

Change attribute. Change an aspect of the definition of an attribute.

List attributes. List the attributes known in the knowledge base.

Inspect attribute. Inspect the aspect values for an attribute.

Print attribute. Print an attribute definition to the printer.

After an attribute editing command, the program returns to the attribute menu. To return to the main menu, press *ESC*.

4.10 Editing the goals

From the main menu, the command `Edit goals` will cause the system to display the goal menu. From this menu, the following commands allow the user to alter the goal definitions in the system.

Add goal. Define a goal for the system.

Delete goal. Remove a goal definition from the system.

Change goal. Alter an aspect of a goal.

List goals. List the goals in the knowledge base.

Inspect goal. Show the aspect values of a goal.

Print goal. Print a goal definition to the printer.

After a goal editing command, the program returns to the goal menu. To return to the main menu, press *ESC*.

582

5 Recommendation procedure

The only useful output from an expert system is the recommendation it makes. Definitions of the rules, attributes, goals, and records are necessary to provide information for the system to make its recommendation, and this information is considered in a careful and calculated manner in order to determine the final conclusion.

First, each of the rules in the system is tested against the attribute values of the record being analyzed. If the antecedent, or "if" part of the rule is true, it executes the consequent, or "then" part, possibly making a recommendation. This recommendation is weighted in several ways. Each rule and each attribute is rated primary or secondary. The combination of a rule and its attributes can be rated primary (both rule and attributes are primary), mixed (one or more attributes or the rule but no attributes are secondary), or secondary (the rule and one or more attributes are secondary). These three categories are then assigned numeric values by the goal, and the appropriate number added to the cumulative weight of the goal(s) recommended.

After all of the rules have been applied, and the goals have built up relative weight values, several special conditions, or metarules, are tested. These metarules are not part of the knowledge base but rather part of the expert system program. The metarules include routines for breaking ties, special checks for unknown data or other tests that are not simple relations between attributes. The metarules can determine a final recommendation in some cases.

If the metarules have not determined a final recommendation, it may be assumed there is no tie for highest point value (metarules would break the ties), so the parent goal with the highest cumulative point value becomes the primary recommendation. If the primary recommendation has subgoals, the subgoal with the highest point value becomes the secondary recommendation. Ties for secondary recommendation are listed as equal.

6 Alternate applications of DOBES developer

As a general development package, DOBES developer can work with most expert system problems. Currently, it is tailored for the specific problem defined by Dr. William Zuk in a report entitled *Expert Systems as Applied to Bridges and Pavements—an Overview.* As such, the records of the system refer to bridges, attributes refer to aspects of a bridge, goals refer to possible dispositions of an old bridge, and rules are a means to determine those goals. The system could however be adapted to another problem without excessive effort or redesign.

In order to adapt the DOBES program to another problem, three things must be done: the problem must be clearly defined (goals), knowledge on the subject must be attained (attributes and rules), and special features of the program must be altered (metarules). The sections below serve as a general guide to this procedure but can not completely guide the process.

6.1 Problem definition

The selection of an appropriate problem for the expert system is a difficult task. In general, the problem should be a question with a small number of possible answers. Each of these possible answers would then be goals for the system, and the problem would direct the remaining research.

6.2 Knowledge acquisition

Once the problem has been isolated, an extensive study must be made of current methods used to solve the problem. Current periodicals, extensive statistical studies, and interviews with experts in the field show methods by which a particular outcome is reached. Each trend and its outcome codes to a rule, and the factors considered by these rules are the needed attributes of the system. Once the rules, attributes, and goals of the problem are found, they may

584

be entered into the system.

6.3 Adjustments

Finally, the metarules for this system will most likely not apply to another problem. They may be simply removed or changed to add features to the system solving another problem. In either case, a knowledge of the Lisp programming language and an understanding of *The DOBES Technical Reference Manual* is recommended.

DOBES Technical Reference Manual

Jonathan Newbrough

1 June 1989

Contents

1 Purpose

This document is intended to provide additional information about the Disposition of Old Bridges Expert System (DOBES). Information included is of a technical nature appropriate for tasks such as (but not limited to) minor alterations of DOBES, extending DOBES with new features and capabilities, adding new metarules to DOBES, and applying the DOBES framework to another expert system problem.

2 Overview

DOBES was developed over a period of months; it represents a compilation of various features to make the program easy to use, easy to understand, and portable across machine configurations. With these considerations, several of the program's main features were created:

- visual menu system of program control
- on-line documentation available throughout the program
- natural, English language representation of expert system rules
- record "swapping" to allow machines with limited memory to work with large databases.

All of these features have been implemented in Lisp in a logical symbolic manner, making them easier to understand and easier to alter and replicate.

This document assumes a familiarity with Lisp programming and with expert systems. For documentation on either of these, many sources are available; for general information about DOBES, see the *DOBES Users' Manual* and the *DOBES Developer Users' Manual.*

This document does not attempt to cover every aspect of DOBES program execution, but instead covers the main method of user interface and program control (command menus) and some of the more complex functions of the expert system. A working knowledge of Lisp programming and a copy of the commented Lisp source should be adequate for understanding other areas of the program operation.

3 Program control

Command menus allow the DOBES user to control the program. These menus are Lisp functions that show lines of text on the screen and invoke an appropriate function corresponding to a line selected by the user. The syntax for a command menu is:

```
(command-menu title names functions help)
```

where:

`title` is a string to be displayed as the title bar of the screen

`names` is a list of strings, each listed as a menu choice

`functions` is a list of symbols, each of which is a function of no arguments to be invoked if the corresponding name is chosen by the user

`help` is an integer or symbol corresponding to an entry in the auxiliary text (help) file.

The functions invoked sometimes call other command menus. These are then referred to as submenus, and the title usually includes the parent menu to the submenu in lower case to indicate the current position in the program menu hierarchy.

4 Knowledge base

The data known by the expert system is stored in Lisp as values (bindings) and properties (associations) of symbols.

4.1 Attributes

Each attribute is associated with a unique symbol of the form `ATTR-`*n*, where *n* is an integer. When an attribute is defined, each symbol in the list `*ATTRIBUTE-PROPS*` becomes a property of the new attribute symbol. Consider hypothetically that `*ATTRIBUTE-PROPS*` is bound to the list `(NAME EXPLANATION QUESTION TYPE PRIORITY)` and that `NAME` has the property list `(NAME "name" TYPE SINGLE-LINE-TEXT QUESTION "Input name" EXPLANATION 14)`. Then the `NAME` property of the attribute would be given a value supplied by the user as a `SINGLE-LINE-TEXT` response to the prompt `Input name`.

Once all of the symbols in the list `*ATTRIBUTE-PROPS*` are given values as properties of the attribute, the attribute definition is complete. Attribute symbols have no values. That is, their information is completely within their properties and not a bound value.

A list of all attributes in the system is bound to the symbol `*ATTRIBUTES*`.

4.2 Rules

Rule definition is identical to attribute definition except that the rule symbols are of the form `RULE-`*n* and the properties of rule symbols are in a list bound to `*RULE-PROPS*`. Also, the symbol `NATLAN` (natural language representation of the rule) in `*RULE-PROPS*` has `CONDITIONAL` as the value for its `UPDATE` property. This means that any time the `NATLAN` property of a rule is changed (including initial definition), the `CONDITIONAL` property of the rule is updated. The `CONDITIONAL` property of a rule is a Lisp representation of the rule generated from the natural language rule text.

A list of all rules in the system is bound to the symbol `*RULES*`.

4.3 Goals

Goals are defined like attributes, but they have symbols of the form `GOAL-`*n*, and their properties are listed as the value of the symbol `*GOAL-PROPS*`.

A list of goals in the system is bound to the symbol `*GOALS*`.

4.4 Records

Records are defined like attributes, but they have symbols of the form `RECORD-`*n*, and their properties are the attributes of the system, listed in the symbol `*ATTRIBUTES*`.

A list of all records in the system is bound to the symbol `*RECORDS*`.

5 Inference procedure

After a record has been chosen to be analyzed by the system, the analysis proceeds as follows:

1. Each of the attribute symbols is bound to the value of the corresponding property of the record. For example, the symbol `ATTR-3` will be bound to the value of the property `ATTR-3` of the record analyzed. (This is the value originally supplied as the value for the attribute when the record was defined.) Each of the goal symbols is bound to the integer 0.

2. Each rule is checked in turn for application. If the first element of the `CONDITIONAL` property of the rule `EVAL`'s to `t`, then the second element of the property is `EVAL`-ed. The first element of the property tests the values of attributes (the IF part of the rule), and second part is usually a recommendation and/or makes a comment concerning the record (the THEN part of the rule).

3. When a rule recommends a certain goal, that goal increases its value by a weight determined by the priority of the rule and the attributes it uses. For example, if the rule has a value of `PRIMARY` for the property `PRIORITY` and one or more of the attributes referred to by the rule definition have a value of `SECONDARY` for their `PRIORITY` properties, then the rule application has a net priority of `MIXED`, and the value of the `MIXED` property of the recommended goal is the weight added to the value of the goal.

4. After all rules have been tested and all recommendations have been made, the metarules are tested. These are not symbol properties like the rules, but rather a section of the function `ANALYZE-RECORD`. The metarules break ties and test special conditions of attributes and rule application.

5. Finally, the highest valued parent goal is printed as the system recommendation, and its child goals are printed as subrecommendations. Comments and point values are displayed.

6 Natural language parsing

The natural language text of a rule is parsed into lisp code before it is understood by the system. Though this text is intended to look and read like natural English, it is a restricted subset of the language with specific syntax and use.

The natural language rules of the system must be statements of the form:

  IF *value comparison value*

    [AND *value comparison value* ...]

THEN *verb object*

    [AND *verb object* ...]

Note: [ ]=optional

Value is either the name of an attribute or a constant attribute value. *Comparison* is one of: `IS, IS MORE THAN, IS LESS THAN, IS GREATER THAN OR EQUAL TO, IS LESS THAN OR EQUAL TO` or `IS NOT`. *Verb* is either `RECOMMEND` or `COMMENT`. *Object* is either a goal (if *verb* is `RECOMMEND`) or a string (for `COMMENT`).

The parser generates Lisp statements in a straightforward manner from statements in this syntax. Consider, for example, the following name properties:

- `(getf 'ATTR-1 'name) = "number of lanes"`
- `(getf 'ATTR-2 'name) = "county"`
- `(getf 'GOAL-1 'name) = "widening"`

and the rule text

```
if the number of lanes is less than 3 and the county
is Albemarle then recommend widening
```

594

From this input, the following CONDITIONAL Lisp statement would be parsed by the function MAKE-CONDITIONAL:

```
((AND (LT ATTR-1 3) (EQ ATTR-2 "Albemarle"))
(RECOMMEND GOAL-1))
```

7 Swapping

Although Lisp and DOBES can operate on machines with large amounts of memory (RAM), it was not expected that users would have over one megabyte (1MB) available. To keep the system from running out of memory if many records were defined, a swapping method was added so that only a few of the records were in memory at one time. The rest were saved in disk files until needed.

In order to do this, records are divided evenly into folders when defined. There are 50 records to a folder (though this value may be changed by changing the value of `*RECORDS-PER-FOLDER*`) and all records in a folder are treated as a group. The symbol `*FOLDER-RECORDS*` lists the symbols of the records currently in memory or those in the current folder. To access a record, the symbol must be in `*FOLDER-RECORDS*` or the folder containing the record must be loaded as the current folder. In this way, only 50 records at a time are defined in the system and memory usage is reduced.

Some functions controlling folder use include `USE-FOLDER`, which loads the appropriate folder to access a certain record, and `LOAD-FOLDER` and `SAVE-FOLDER`, which load and save the current folder.