TM-0244

SHOOTX: A MULTIPLE KNOWLEDGE BASED DIAGNOSIS EXPERT SYSTEM FOR NEAX61 ESS

by

Y. Koseki, S. Wada, Nishida and H. Mori

NEC Corp.

January, 1987

SHOOTX : A MULTIPLE KNOWLEDGE BASED DIAGNOSIS EXPERT SYSTEM FOR NEAX61 ESS

Yoshiyuki Koseki, Shin-ichi Wada, *Tetsurou Nishida and Hiroyuki Mori*

C&C Systems Research Laboratories Integrated Switching Development Division

NEC CORPORATION
4-1-1 Miyazaki, Miyamae-ku
Kawasaki, 213 JAPAN

ABSTRACT

This paper describes a diagnosis expert system for the NEC NEAX61-series digital switching system. The system is founded on a multiple-knowledge based approach, utilizing *design-knowledge* represented on an *abstract-signal-flow model* of the switching system, and domain specific *symptom-knowledge* and *test-knowledge*. All kinds of knowledge are uniformly described in a network-based representation which is more flexible than the conventional rule-based representations. The system provides graphic-oriented and menu-driven man-machine interface for easy operation.

INTRODUCTION

The role of electronic equipment, such as telecommunication switching systems, has been growing in today's modern society. Modern electronic switching systems (ESS) are designed to achieve high reliability. For instance, most of their components are duplicated, so the back-up components can take over when a failure occurs. They also have built-in diagnosis functions to locate defective components.

However, the performance of these diagnosis functions is limited, and a defective component cannot always be clearly discriminated. Some classes of failures not detectable by built-in diagnoses are:

- Intermittent failure
- Failure in an interface unit
- Failure in a non-duplicated unit
- Undetected failure due to insufficient diagnosis

In such a case, a skilled maintenance technician must isolate the failure using his experience and knowledge.

Much work has been done previously in the area of automatic diagnosis of electronic equipment. Early work includes the D-Algorithm, a test-pattern generation algorithm [Roth67]. But the applicability of the algorithm is limited because it applies only to combinational logic circuits rather than sequential logic circuits and it is only effective for stuck-at faults.

Therefore, in most of the modern switching systems, functional tests are mainly employed for built-in diagnoses. Since functional tests employ only known input/output patterns, it is not guaranteed to cover all the failures.

Moreover, the intermittent failures can not always be diagnosed by the test patterns. Since the failure occurs intermittently, it is not guaranteed that the same failure occurs again when the test pattern is applied. In this case, an intensive symptom analysis is the principle means for diagnosis.

Recently, as an application of Artificial Intelligence technology, expert-system techniques have been attracting attention. Early diagnosis expert systems like MYCIN were rule-based systems which used a set of empirical symptom-cause rules [Shortliffe76]. They seemed promising, but in reality, it was a very time-consuming task to develop and maintain a practical-sized rule set for real systems like an electronic switching system.

As a contrast to this approach, the so-called first-principle-based approach was developed [Genesereth84, Davis84]. This employs structural and behavioral descriptions of the target system, and reasons about the diagnosis without having empirical knowledge. This approach seems a good alternative; however, we do not believe it can be applied to complex systems like the ESS, because the behavioral description of such a system is too difficult to describe.

The diagnosis expert system SHOOTX utilizes design knowledge as well as skilled maintenance technician's empirical diagnosis knowledge. Using both types of knowledge, it helps maintenance technicians to find defective components, which are not detected by built-in diagnosis functions.

The system was designed to be highly interactive and provides many alternatives for the user. Therefore, novice users can perform a difficult diagnosis task by just following the suggestion from the system, whereas expert users can use it as an aid to make their own decisions.

This system falls into the same category as some recently developed systems, like LES [Laffey86]. However it has many unique features:

The expert heuristic knowledge is represented in a semantic network by utilizing an inheritance mechanism instead of rules. In this way, more powerful heuristics can be described in a compact form.

- It leaves much flexibility regarding the user's subsequent action, by providing multiple choices sorted on a priority basis.
- It provides a graphics-oriented user interface for understandability.

In this paper, we describe how the expert's diagnosis knowledge is represented in SHOOTX, and explain its user interface and implementation.

EXPERT'S DIAGNOSIS KNOWLEDGE

Through interviews conducted with skilled maintenance technicians, we found that their diagnosis method generally follows the flow diagram, shown in Fig. 1.

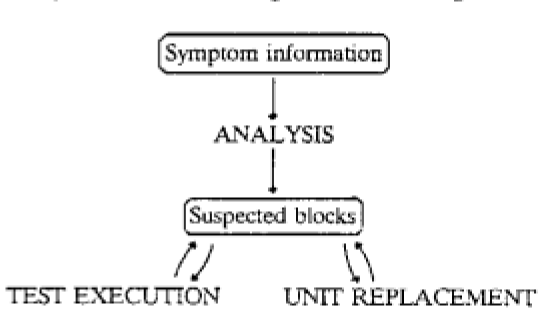


Fig. 1 General Diagnosis Flow

Given the symptom information, the expert thinks of suspected portions which could have possibly caused the defect. This is done by determining which functional blocks are related to the symptoms. To perform this task, he has a model in his mind to represent the structure and behavior for the target ESS.

With the list of suspected blocks, he thinks of the subsequent action to be performed. If the suspected area is relatively small, the subsequent action may be replacing suspected units. If the suspected area is broad, he thinks of a test to reduce the number of suspected blocks. The test to be performed is chosen according to its effectiveness and cost. After the test, he reduces the suspected area by interpreting the test result. A structure-and-behavior model is also used at this stage. He repeats this test-and-reduce cycle until he gets a small number of units to be replaced.

We may conclude that the types of knowledge used are classified as follows:

- Design knowledge: structural and functional description of the target system
- Diagnostic knowledge: symptoms and tests
- Control knowledge: strategy for deciding the subsequent action

SYSTEM OPERATION

When there is a fault found in an ESS, a maintenance technician starts a trouble-shooting session with the expert system by selecting the appropriate symptom in the menu.

Then the expert system asks for the detailed symptom information. With the ESS configuration and status information, it infers the suspected faulty blocks.

When there are many suspected blocks, the expert system suggests appropriate tests in order to discriminate a faulty portion. The test results are then interpreted to reduce the number of suspected blocks. This is repeated until a small number of suspected blocks are identified.

When there are a few suspected blocks, or there are still several suspected blocks but no appropriate test, the expert system suggests an appropriate unit replacement, showing the unit location graphically.

ABSTRACT-SIGNAL-FLOW BASED DIAGNOSIS

To represent structural and functional description, we have used an *abstract-signal-flow* model. This model is used for suspect generation by symptom analysis and suspect discrimination by test result interpretation.

The model is represented in a directed graph as shown in Fig. 2. This example shows four kinds of signal flow to realize a dial tone connection in an ESS. In this example, the line scan signal for a telephone line comes up to the central processor unit (CPU) through the signal flow (a). To make an interconnection between the originating party and the terminating party, three abstract-signal-flows (a), (b), (c) are concerned.

(a) Line scan signal

```
Line -> LC -> E/G Conv -> LM Intf ->
  Scn Ctl -> Ins -> SMUX -> Drp -> Q Ctl ->
    Bus Ctl -> BIU -> CPU
```

(b) Tone source control order

```
CPU -> BIU -> Bus Ctl -> SMC ->
  SPC Intf -> T2 CtlM -> T2Sw
```

(c) Originating subscriber control order

```
CPU -> BIU -> Bus Ctl -> Ord TX -> SRDQ ->
  SDMUX -> Drp -> DrpQ -> SIO -> LM Intf
```

(d) Voice signal path

```
Tone Gen -> PMUX -> Ins -> SMUX -> Pad ->
  T1 -> S1 -> JHW1 -> JHW2 -> S2 -> T2 ->
    SDMUX -> Drp -> PDMUX -> DLSw ->
      E/G Conv -> LC -> Line
```

Fig. 2 Dial tone connection signal-flows

In SHOOTX, this model is represented at functional block level; that is, each node on the flow is a functional block. The entire ESS representation forms a network of functional blocks connected to each other by signal flows.

Suspect generation by symptom analysis

With symptom information, suspect generation is accomplished using the abstract-signal-flow model. For example, to send a dial tone to an originating subscriber, the four abstract-signal-flows, shown in Fig. 2, must not be defective. If there is a no-dial tone symptom observed, there must be a failure in a functional block along with these signal flows. Therefore all of the functional blocks along the flows are enumerated as suspects.

When the symptom is a parity-error or a pilot-error, the suspected blocks are along the signal flow from the parity/pilot generator to the checker. In this case, we get relatively small number of blocks as suspects.

Effective test selection

After the suspect generation is carried out, all of the tests which are effective to reduce the number of the suspected units are selected. By the word *test,* we mean any actions to reduce the number of suspected units. According to our knowledge acquisition study, tests are mainly classified into symptom specific tests and functional-block specific tests.

Symptom specific tests are generated by searching through the symptom knowledge hierarchy. Functional-block specific tests are generated according to suspected functional blocks. These generated tests are sorted by priority and shown to the user. Therefore, novice users can mechanically choose the most recommended test, while expert users can use the list only as a guide.

Suspect discrimination by test result

There are two different kinds of strategies for test result interpretation. One strategy is the direct interpretation of the test result to functional blocks; that is, a certain test result indicates whether a set of functional blocks is faulty or not. The other strategy is to use the above mentioned abstract-signal-flow model; that is, a certain test result indicates whether functional blocks along certain signal flows are faulty or not.

KNOWLEDGE REPRESENTATION

All kinds of knowledge in SHOOTX are uniformly represented in a network structure. A network consists of a set of *objects* and their *relations,* as shown in Figure 3.

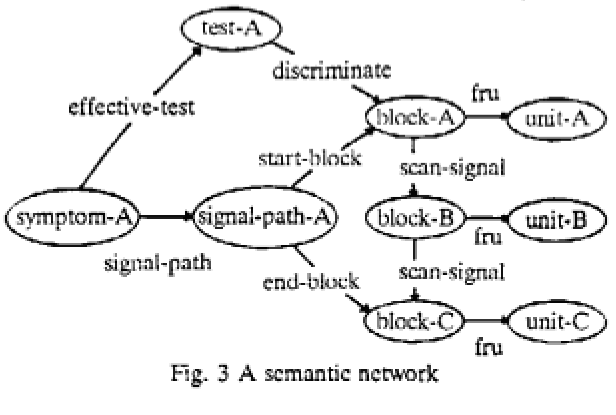


Fig. 3 A semantic network

Each *object* represents a certain lump of knowledge pertaining to a concept or a physical object. That is, it may represent a functional block, a field-replaceable-unit (fru), a symptom, or a test.

Relations literally represent relations between these objects. Some relations are used for diagnostic inference. For instance, relations representing signal flows among functional blocks (*scan-signal* in Fig.3), are used to enumerate suspected blocks according to the symptom information.

Some relations form a hierarchy of objects. For instance, an object which represents a general concept of duplicated functional blocks is a super object of individual functional-block objects. Objects to represent knowledge about tests and symptoms also form a similar hierarchy.

Through these hierarchical relations, lower level objects can *inherit* knowledge concerning their super objects. Therefore, common knowledge about a class of objects must be represented only once in the class representing object. In this way, a compact knowledge representation was achieved.

Design knowledge representation

Design knowledge is a logical and physical structure description of the ESS.

The logical structure description represents properties of all the functional blocks and the logical relations (signal flows) between the blocks. Functional blocks are represented as objects, and logical connections are represented as relations as shown in Fig. 4.

The physical structure description is also described in the design knowledge. Physically replaceable units are also represented as objects in the network, connected to functional blocks by relations. These relations are used to infer actual locations of physical units, such as, packages and cables.

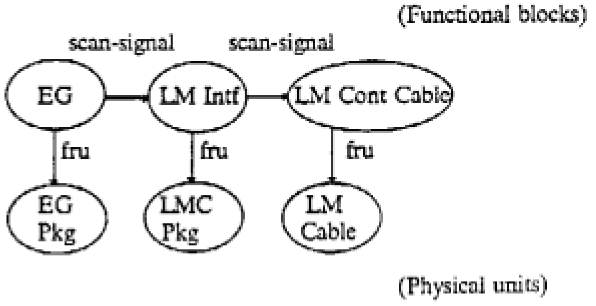
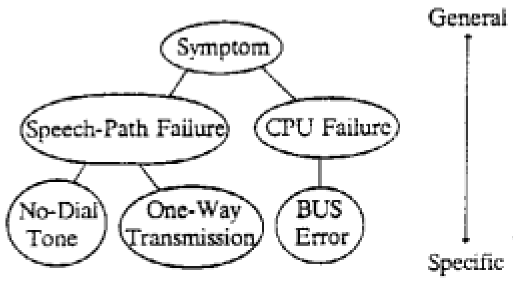


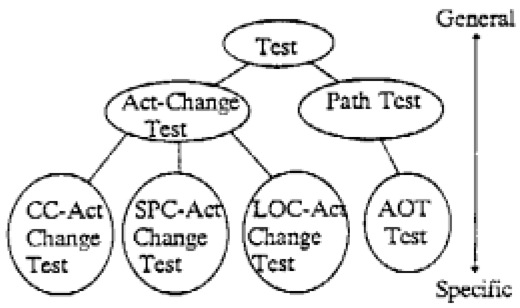
Fig. 4 Design knowledge

Diagnosis Knowledge Representation

This is the heuristic knowledge which distinguishes the expert system from conventional diagnosis program. Diagnosis knowledge consists of symptom knowledge and test knowledge. Each knowledge forms a hierarchical tree. As shown in Fig. 5, a higher object of each concept tree is general knowledge of symptom or test. Lower objects describe individual symptom or test knowledge.



(a) Symptom knowledge hierarchy



(b) Test knowledge hierarchy

Fig. 5 Diagnosis knowledge

Symptom objects hold symptom specific knowledge such as procedures to gather symptom parameters for suspected block generation.

Test objects hold test specific knowledge such as conditions to be effective, test execution procedures, and result interpretation strategies.

Inference mechanism

Unlike other rule-based diagnosis expert systems, this system does not use a general inference mechanism like the forward reasoning or the backward reasoning. Instead, it uses the more domain specific control mechanisms written in corresponding objects as Prolog procedures (so called *methods* in object-oriented programming terms [Goldberg83]).

Through pre-prototyping with a forward reasoning mechanism, we found that it was more natural and flexible to write inference knowledge as methods than as a set of rules. Utilizing the multiple inheritance mechanism through the relations, the knowledge-base becomes more compact and more modifiable.

Inference is done by calling methods one by one, creating the *instances* of the objects.

USER INTERFACE

The principle we have employed in designing the user interface of this system is: *What you see is what is going on.* According to this principle, the graphics-oriented user interface is provided in a multiple window environment as shown in Fig. 6.

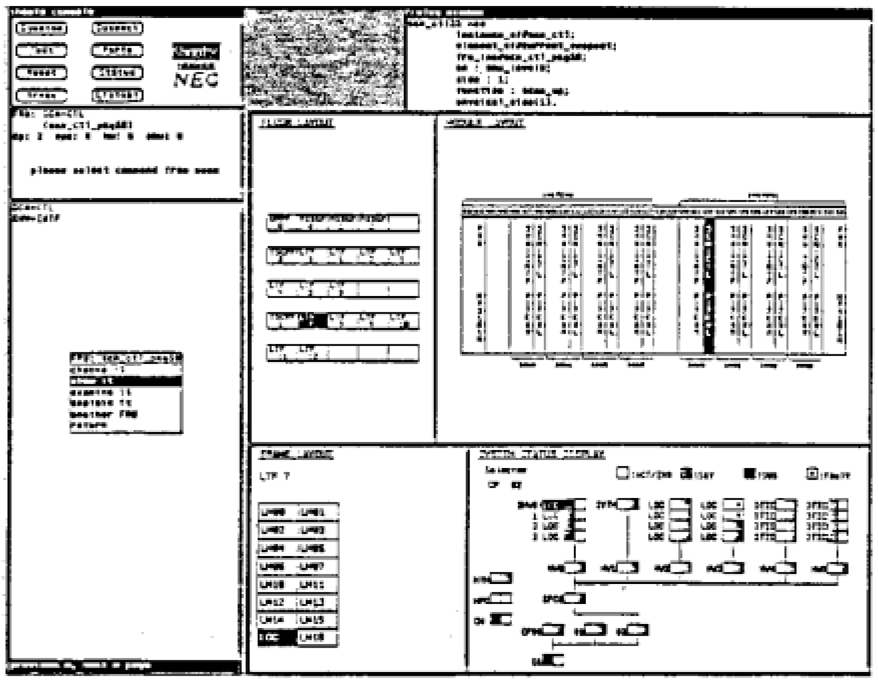


Fig. 6 User interface screen

It shows the detailed location of the unit in the windows. The floor layout window shows the frame location on the floor, the frame layout window shows the module location in the frame, and the module layout window shows the package/cable location in the module. User interaction is done through the menu-window by the menu, sorted by priority, using the *mouse* pointing device.

IMPLEMENTATION

We have developed an expert system tool environment called PEACE (PROLOG based Engineering Applications Environment) on an engineering work-station. It is a semantic-network based knowledge representation system built on the logic programming language PROLOG. It supports multiple programming paradigms such as:

- Object-oriented programming style with multiple inheritances through user defined relations between objects.
- Data-oriented programming style on *slots* of objects.
- Logic-programming style using PROLOG pattern matching and a backward chaining mechanism with backtracking.

The reasoning mechanism and knowledge-base of the system are written in PEACE which offers a more productive environment than LISP-based tools to develop expert systems.

It is notable that the object-oriented programming in PEACE is different from the conventional ones. In the conventional object-oriented programming system like SMALLTALK-80 [Goldberg83], relations between objects which can inherit information are restricted to some typical cases, namely, meta-class/class relation, super-class/class relation, and class/instance relation.

However to represent real world knowledge, more kinds of relations must be represented. For instance, in this system, some information in an object for a suspected functional block must be inherited to the objects for the corresponding physical units.

Figure 7 shows the example description of a function block in PEACE.

```
eg has
        super_type # func_block;
        control_path # lm_intf;
        voice_path # dlsw;
        dual_single # single_function_block;
        en_concentration : eg;
        fru # eg_pkg;
        faultability : 1.0.
```

Fig. 7 An example functional block description

In this description, # denotes the relation with the other object. The inheritance can be made through these multiply defined relations.

STATUS AND CONCLUSION

The prototype system was completed and it has been in a demonstrative evaluation phase. In this period, it has been demonstrated to many NEC operation and maintenance technicians and design experts. By operating it by themselves, they have realized that it greatly improves efficiency of operation and maintenance.

They also recognized that its diagnosis strategy and knowledge representation model are close to their thinking process. After the demonstration, these domain experts became cooperative to give knowledge and the knowledge-base-development productivity was greatly increased.

Unlike other rule-based systems, the different types of knowledge are clearly classified and are explicitly represented in a unified network. Therefore, their development and maintenance task is relatively easy.

However, we feel that the future research is necessary for easier knowledge acquisition. The design knowledge is with ESS designers, whereas, the diagnostic knowledge is with field maintenance technicians. Therefore, knowledge engineers have to understand their knowledge and interpret it to develop the knowledge-base. For efficient and error-free knowledge-base development and maintenance, the future research must be made to make knowledge-base manipulatable by these different kinds of experts directly.

ACKNOWLEDGEMENTS

We thank all of the members of the SHOOTX implementation team for enabling the system to be realized, namely, Mr. N. Wakasugi, Mr. M. Kondo, Mr. M. Oishi, and Mr. T. Hisada. We thank Mr. K. Toudou, Mr. H. Igarashi and their colleagues for their patient cooperation on the knowledge acquisition study. We also express deep appreciation to Mr. A. Kitamura, Mr. H. Mori, and Dr. S. Goto for continuous encouragement and support. Finally, we would like to thank Mr. Y. Iwashita of Institute for New Generation Computer Technology.

REFERENCES

[Roth67]
J. P. Roth, W. G. Buricius, and P. R. Schneider, "Programmed algorithms to compute tests to detect and distinguish between failures in logic circuits," *IEEE Trans. Electron. Comput.*, vol. EC-16, no. 10, 1967, pp. 567-580.

[Shortliffe76]
E. J. Shortliffe, *Computer Based Medical Consultations: MYCIN*, Elsevier, New York, 1976.

[Genesereth84]
M.R. Genesereth, "The Use of Design Descriptions in Automated Diagnosis," *Artificial Intelligence* 24, 1984, pp. 411-436.

[Davis84]
R. Davis, "Diagnostic Reasoning Based on Structure and Behavior," *Artificial Intelligence* 24, 1984, pp. 347-410.

[Laffey86]
T. J. Laffey, W. A. Perkins, T. A. Nguyen, "Reasoning About Fault Diagnosis with LES," *IEEE Expert* vol. 1, No. 1, 1986, pp. 13-20.

[Goldberg83]
A. Goldberg, and D. Robson, *SMALLTALK-80: The Language and its Implementation*, Addison-Wesley, 1983.