TM-0491

An Expert System Architecture for Switching System Diagnosis

by

S. Wada & Y. Koseki

April, 1988

AN EXPERT SYSTEM ARCHITECHTURE FOR SWITCHING SYSTEM DIAGNOSIS

Shin-ichi Wada, Yoshiyuki Koseki,
C&C Systems Research Laboratories

and Tetsurou Nishida
Integrated Switching Development Division

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

ABSTRACT

This paper presents the architecture of a troubleshooting expert system called SHOOTX, for the NEC NEAX61-series telephone switching system. It helps inexperienced maintenance technicians to perform diagnosis tasks. Modeled on human experts' diagnosis methods, it enables location and repair of a fault which cannot be detected by built-in diagnosis functions. It figures out the suspected components, based on an abstract signal-flow model of the target switching system. Since several kinds of knowledge are represented in a unified network, the knowledge base is compact and modifiable. The paper also describes the user-oriented man-machine interface.

1. Introduction

In accordance with the recent growth in hardware technology, modern electronic devices like an ESS (Electronic Switching System) have become complex. Therefore, it is difficult for an inexperienced maintenance technician to locate a defective component in an ESS and repair it. Although diagnosis functions are built into an ESS, a defective component cannot always be clearly detected by using these functions.

On the other hand, a skilled human expert can locate a defective component and repair it by using various diagnosis strategies. For example, by watching and analyzing the symptom, he can infer what the possible defective components are. Based on this, a diagnosis expert system, modeled on human experts' strategies, is foreseen, which would enable an inexperienced maintenance technician to diagnose a fault that cannot be diagnosed by the built-in functions.

In addition, many kinds of ESS operations for diagnosis, such as typing commands at the console, and analyzing messages printed out from an ESS, are difficult and troublesome for a maintenance technician. It is also desirable to release the maintenance technician from these troublesome tasks by automatic ESS operations via on-line interface with an ESS.

These reasons motivated the development of a diagnosis expert system, called SHOOTX, for NEAX61-series ESS.

In the diagnosis domain, many expert systems have been developed applying artificial intelligence technology. Early diagnosis expert systems, like MYCIN[Shortliffe76], were rule-based systems which used symptom-cause association rules. They seemed promising. However, in reality they required the description of too many rules for complex systems like an ESS. A large number of rules are necessary because of the number of symptoms, components and tests. It is a difficult task to develop and maintain so many rules.

As a contrast to this approach, a "*reasoning in first principle*" method was developed[Genesereth84, Davis84]. This method reasons diagnostic behavior, based on structural and behavioral description of the target system. This approach seemed promising. However, there are difficulties in this approach with describing the structure and behavior of complex systems like an ESS, which includes CPU's and various kinds of software.

Instead, SHOOTX incorporates design experts' knowledge as well as skilled maintenance technicians' empirical knowledge. Using both kinds of knowledge, it helps maintenance technicians to find defective components. Especially, it has an abstract signal-flow description of the target ESS as design knowledge, and it figures out suspected components, based on the description. Both the complexity of description and the diagnosis resolution, are achievable for complex systems like an ESS.

In addition, SHOOTX has the following features:

1)Heuristic test strategies, modeled on human experts diagnosis method:

Following the diagnosis strategies employed by human experts, SHOOTX can accomplish various tests to narrow down the possible causes of the trouble.

2)Network-style knowledge representation:

Structure, symptom and test knowledge regarding the target ESS are represented in a unified network, based on the network knowledge representation facility in PEACE(Prolog based Expert AppliCations Environment[Koseki87b]). By this method, the description is compact and modifiable, while only the necessary objects in the network are accessed according to the diagnosis phase.

3)Man-machine interface usable for inexperienced maintenance personnel:

SHOOTX provides a menu-driven and graphic-oriented user interface. In determining a subsequent action to perform, it recommends inexperienced maintenance personnel the most appropriate action, while leaving sufficient flexibility to choose other actions when deemed necessary.

This paper describes the problems involved in switching system diagnosis. Next, it describes the architecture of SHOOTX, namely, the diagnosis strategy modeled after human experts, diagnosis architecture, knowledge representation, and man-machine interface.

2. Problems in Switching System Diagnosis

Modern ESS's (Electronic Switching Systems) have become more and more complex in accordance with the recent growth in hardware technology. Therefore, ESS maintenance has also become a difficult task.

When a trouble occurs in an ESS, one or more fault messages are printed out from an ESS console. They do not always pinpoint the original cause of the trouble but only indicate the detection of malfunctions affected by the trouble. A maintenance technician has to determine the original cause.

Another case of trouble detection is based on subscriber complaints. In this case, the maintenance technician should determine the essential problem to achieve accurate diagnosis.

In order to determine the cause of the trouble, an ESS has built-in diagnosis functions for locating defective components. However, the performance of these diagnosis functions is limited, and a defective component cannot always be clearly detected. The following classes of failures are not detectable by built-in diagnosis functions:

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

Even for a failure in these classes, a skilled maintenance technician can find out the defective component and repair it. He performs diagnosis by various strategies based on his knowledge and skill.

However, highly skilled maintenance technicians are very few. In addition, since the life span of an ESS is over ten years, it is hard to retain highly skilled maintenance personnel in a specific telephone office for a long time.

As a way to support inexperienced personnel, an effort to organize a set of manuals has been promoted. However, it takes a long time to perform diagnosis through reading manuals. This is because even a single task requires reading many volumes. In addition, manuals lack expressiveness; they cannot dynamically indicate diagnosis operations according to the situation. Thus, diagnosis by manuals is not the best way to support inexperienced personnel.

Therefore, an expert system is desired which assists inexperienced maintenance technicians to perform diagnosis of an ESS, based on the strategy used by human experts.

3. Diagnosis by Human Experts

Human experts can pinpoint and repair faults which cannot be located by built-in diagnosis functions. Their know-how and diagnosis strategies were acquired for SHOOTX through interviews with the experts by the knowledge engineers. The interviews were carried out in the following style:

At the beginning, the experts were asked to explain their procedure for one symptom precisely. They could easily imagine a concrete situation and present their methods. Next, the methods were enhanced to be applicable to various kinds of troubles.

As a result of the interviews, it is known that the diagnostic methods used by experts are as follows:

1)Analysis of symptom:

In order to determine the cause of the trouble, an expert considers detailed information regarding the symptom, ESS status, and so on. He sometimes uses past experience about symptoms and components' faultability.

2)Carrying out effective tests to narrow down suspected components:

In order to determine a cause among many suspects, that is, the possible causes of the trouble, a human expert thinks of effective tests to narrow them down. The word *test* means any action to reduce the number of suspects. Some of the tests carried out by human experts are as follows:

- Viewing the occurrences of symptom and their distribution
- Acquisition of various kinds of information in the ESS, like operation status for duplicated units
- Making use of built-in diagnosis functions
- Typing commands into the ESS and watching the ESS behavior in reaction to them
- Changing the operation status for the duplicated units and looking for the occurrence of the trouble
- Viewing alarm indicators on the packages and panels

3)Trial and error in repair actions such as replacing fault FRU's(Field Replaceable Units) with spares and resetting the switches

4)Recognition and consideration of external conditions such as thunderbolts and air-conditioner faults

By combining various methods, human experts narrow the suspects down, and finally repair the trouble by replacement of suspected FRU's and so on.

4. Basic Diagnosis Method for SHOOTX

4.1 Basic Diagnosis Flow

The basic diagnosis flow in SHOOTX is shown in Fig-1. During the diagnosis, *suspected components*, that is, candidates which might have caused the trouble, are considered and updated.

Given the symptom information, SHOOTX first considers the suspected components by analyzing the symptom information. After that, a test effective for them is chosen and carried out. By interpreting the test result, suspected components are narrowed down; the list of suspected components is updated to smaller ones. According to the updated list of suspected components, effective tests are reconsidered. In this way, tests and result interpretations are repeated. After several repetitions, a small number of suspected components can be obtained. Finally, corresponding FRU's, such as equipment packages or connector cables, are replaced, and the trouble is repaired.

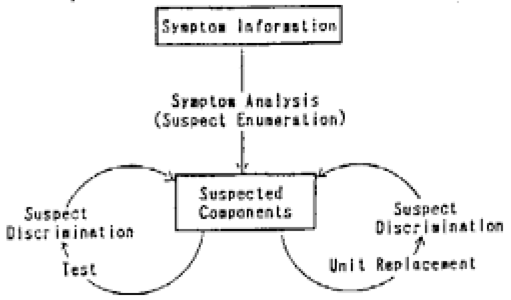


Fig-1 : Diagnosis flow

4.2 Abstract Signal-Flow Based Reasoning

In order to implement the reasoning about suspected components according to the symptom and test results, SHOOTX has an *abstract* signal-flow description, which represents the functional structure of the ESS. The expert system looks into it in considering suspected components. An abstract signal-flow description is represented in a directed graph. Fig-2 indicates an extraction of the abstract signal-flow description. In it, each node represents a component, and each arc represents a signal-flow between components.

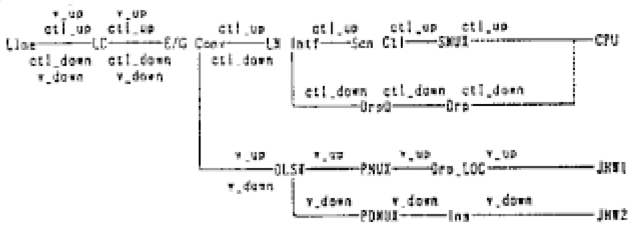


Fig-2 : Abstract signal flow description

With an abstract signal-flow description, suspected components for a given symptom can be enumerated. For example, consider a case wherein the following symptom is occurring.

SYMPTOM: no-dial-tone

(meaning)

"Although a subscriber picks up a receiver, no dial-tone is heard."

In order that a dial tone should be normally heard, the following signal path must not be defective:

SIGNAL PATH:

"a control signal path from the line to the CPU"

The relation between *no-dial-tone* symptom and this signal path is compactly described as symptom-specific knowledge, which is graphically shown in Fig-3.

The components on the path can be listed by searching for the abstract signal-flow description. By enumerating the nodes along the path, from *Line* to *CPU* through *ctl_up* arcs in Fig-2, the following components are listed as suspected components for the no-dial-tone symptom.

SUSPECTED COMPONENTS:

Line, LC, E/G Conv, LM Intf, Scn Ctl, Ins, SMUX, ... , CPU

The components in the abstract signal-flow description do not need to be so precise as logic gates but may be functional blocks divided into suitable sizes. Since the final purpose of the diagnosis is to find and replace a faulty FRU, such as a package or a connector cable, functional blocks only need to be divided precisely so that they can satisfy the following conditions:

1)Different FRU's commonly include no functional blocks.

2)Functional blocks, which have different properties, are partitioned as different components, so that each block will fall into one category in discriminating between suspects.

3)Signal paths can be represented with the functional blocks.

The abstract signal-flow method is also applicable to symptoms like parity-error and pilot-error messages. For instance, the suspected signal path for a pilot-error is a path from the pilot signal generator to the pilot signal checker. The blocks on the path are enumerated as suspected components of the error.

For the kind of tests which check signal flow functions, narrowing the suspected components is easily accomplished. In the case when the test result is no error, that is, the signal flow works normally, the components on the flow will be eliminated from the suspected components. In the other case, the defective component must be on the signal flow, and the components out of the flow will be eliminated from the suspected components.

By the abstract signal-flow method, structural knowledge for the ESS is naturally represented in suitable precision. In addition, the total amount of the knowledge is kept small. This is because a signal-flow description is shared by many kinds of symptoms, leaving brief symptom-specific descriptions.

5. The Architecture of SHOOTX

The whole architecture of SHOOTX is shown in Fig-4. Diagnosis is cooperatively carried out by a maintenance technician and the expert system. Only the essential decision and physical operations are imposed on the maintenance technician, while the reasoning tasks and detailed ESS operations via interface with the ESS are carried out by the expert system. The following are the features of this architecture.

(a)Trouble data maintenance with log file

The expert system collects fault messages printed out by the ESS and stores them in a *fault message log*. It also maintains complaint information from subscribers in a *complaint log*. Therefore, a maintenance technician's task for designating a symptom to diagnose is only selecting a malfunction in the fault message log or in the complaint log. After that, the expert system diagnoses the designated symptom.

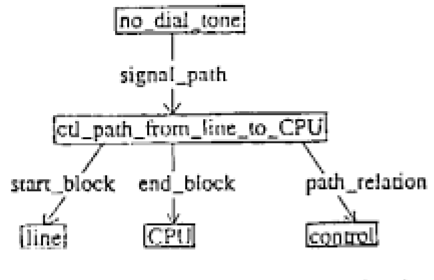


Fig-3: Relation between symptom and signal-path

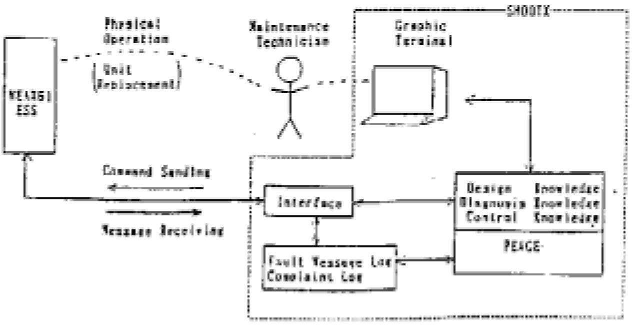


Fig-4 : Whole architecture of SHOOTX

(b)Flexible man-machine interaction

In the course of diagnosis, the following choices appear concerning a subsequent action:

- Test or FRU replacement
- Which test to carry out(when test is selected)
- Which FRU to repair(when FRU replacement is selected)

In selecting a subsequent action, the expert system reasons about possible actions and recommends the most appropriate one, leaving the maintenance technicians flexibility to select an alternative action. By such flexible man-machine interaction, inexperienced maintenance technicians can carry out troubleshooting merely by following the actions recommended by the expert system, while an expert technician may select actions himself.

Moreover, with the 'Explain', 'Why' and 'Help' functions, the user can freely obtain explanations for the diagnosis execution.

(c)Automatic ESS operations

The expert system handles interface functions with the ESS. Frequent command sending and message receiving to/from the ESS are necessary to carry out various test strategies. However, these operations are often misleading and require special knowledge to carry out appropriately. As shown in Fig-4, they are automatically carried out by the expert system via on-line connection with the target ESS. Using this facility, most of the tasks for tests are automatically carried out by the expert system.

(d)Understandable instructions in maintenance technicians' operation

As an aid to maintenance technicians in accomplishing physical operations, simple and detailed instructions are provided. For example, in instruction for a unit replacement procedure, the physical location of the FRU, such as a package or a connector cable, is displayed in graphic windows and detailed commands such as, *turn the power switch on the package to off*, are given to the user. Throughout the replacement procedure, the ESS status is set ready for package replacement by automatic ESS operations.

6. Knowledge Representation

6.1 Knowledge Base Structure

This section presents the knowledge representation. SHOOTX is based on multiple kinds of knowledge acquired from both design experts and maintenance experts of the target ESS[Koseki87a]. Structural knowledge regarding the target ESS is acquired from design experts, while the pertinent maintenance knowledge, such as symptoms and tests, is acquired from the maintenance experts. Then, the knowledge base is composed of the following:

- Structural knowledge acquired from design experts
  1)Functional block knowledge
  2)FRU knowledge
- Maintenance knowledge acquired from maintenance experts
  3)Symptom knowledge
  4)Test knowledge
  5)Diagnosis control knowledge

These kinds of knowledge are represented in an expert system tool environment called PEACE (Prolog based Expert AppliCations Environment) [Koseki87b]. PEACE supports several programming paradigms amalgamated in Prolog. The features of PEACE utilized in the knowledge representation are as follows:

- Semantic-network-style knowledge can be represented with objects and relations between them
- Relation-oriented programming is supported
- Compact knowledge representation is possible by multiple inheritance, not only through particular relations, such as *has_class* and *super*, but also through domain-specific relations, like *super-block*.
- Procedural knowledge can be represented in Prolog utilizing pattern matching and backtracking mechanisms

6.2 Network Knowledge Representation

Knowledge, except for diagnosis control knowledge, is represented in a unified network by using PEACE. By describing nodes in the network as objects and by connecting nodes by relations, the network is uniformly represented. Signal-flows between functional blocks can be directly represented as relations. Relations between different kinds of knowledge are also represented in the same style. These relations are between:

- Symptoms and corresponding signal-path names
- Signal-path names and their starting/ending functional blocks
- Functional blocks and corresponding FRU's

With these relations, the network-style representation is constructed as shown in Fig-5. It represents the relations between symptoms, tests, functional blocks and FRU's.

For instance, functional blocks corresponding to Symptom-A can be listed as follows:

Block-A, Block-B, and Block-C

The network can be compactly represented by the PEACE facilities. Maintaining inverse relations is automatically performed by PEACE. That is, when a relation is set or deleted, its inverse relation is automatically set or deleted. Therefore, intricate descriptions for updating inverse relations are unnecessary.

In addition, the knowledge is represented without redundancy in hierarchical structures. For instance, common properties among several symptoms can be described in a general symptom object by *inheritance* facility. Similarly, the test, functional block and FRU knowledge are also described compactly utilizing inheritance. *Multiple inheritance* through

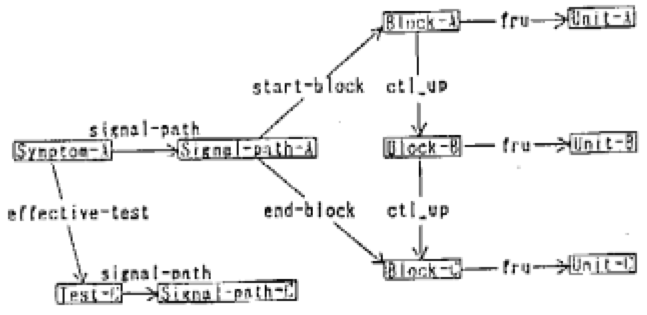


Fig-5 : Network-style knowledge representation specified relations is especially of great use. In representing the complex structure of the target ESS, inheritance through domain-specific relations, such as *fru* and *super-block*, is utilized, as well as inheritance though general relations like *instance_of* and *super*.

In this representation style, the structure, symptom and test knowledge descriptions are very compact, and local modification is easy.

6.3 Details in Knowledge Representation

This section precisely presents PEACE-based representation for each kind of knowledge, namely functional block, FRU, symptom, test and diagnosis control knowledge.

Functional Block and FRU Knowledge

Fig-6 is an example description of functional block and FRU knowledge. In Fig-6, eg, lm_intf, and scn_ctl represent functional blocks, and lmc_pkg represents an FRU. In Fig-6, # denotes the relation with the other object, and : denotes the slot. In a functional block object, the following knowledge are described:

(a)Signal-flow:

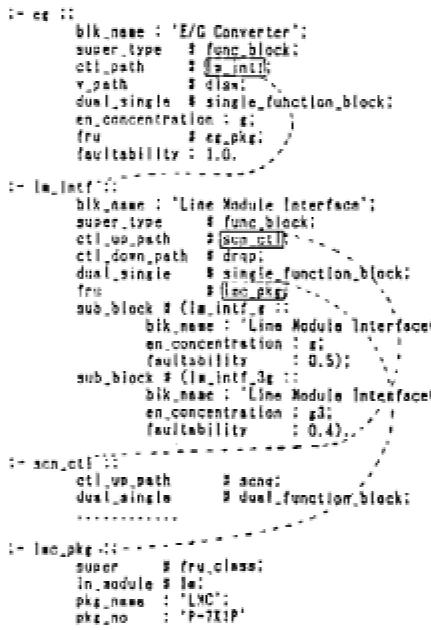
The signal-flow knowledge as shown in Fig-2, is represented with pertinent relations such as *ctl_up_path* and *v_down_path*. Each relation represents a signal-flow connection between components. For example, *lm_intf* is connected with *scn_ctl* by *ctl_path* as shown in Fig-6.

In Fig-6, the generalized name for the relation is used for brief representation. For example, eg is connected with lm_intf by *ctl_path*. This means that eg is connected with lm_intf by both the *ctl_up_path* and *ctl_down_path* relations.

By looking into the description; functional blocks on a specified signal-flow path can be enumerated. For instance, functional blocks on "*ctl_up_path from eg to scn_ctl*" are enumerated as eg, lm_intf, and scn_ctl.

(b)Block's property:

Block's properties are given within each functional block object.



```
:- eg ::
        blk_name : 'E/G Converter';
        super_type   # func_block;
        ctl_path     # lm_intf;
        v_path       # dlaw;
        dual_single  # single_function_block;
        en_concentration : g;
        fru          # eg_pkg;
        faultability : 1.0.
:- lm_intf ::
        blk_name : 'Line Module Interface';
        super_type      # func_block;
        ctl_up_path     # scn_ctl;
        ctl_down_path   # drap;
        dual_single     # single_function_block;
        fru             # lmc_pkg;
        sub_block # (lm_intf_g ::
                blk_name : 'Line Module Interface(g)';
                en_concentration : g;
                faultability     : 0.5);
        sub_block # (lm_intf_3g ::
                blk_name : 'Line Module Interface(3g)';
                en_concentration : g3;
                faultability     : 0.4).
:- scn_ctl ::
        ctl_up_path       # scnq;
        dual_single       # dual_function_block;
        ............
:- lmc_pkg ::
        super     # fru_class;
        in_module # lm;
        pkg_name  : 'LMC';
        pkg_no    : 'P-7E1P'
```

Fig-6 : Sample of structural knowledge description

For instance, the property *dual_single* could have the value *dual_function_block* or *single_function_block*. They may be referred to in narrowing down the suspects according to the test results. The narrowing method is explained in the description of symptom and test knowledge.

(c)Correspondence to FRU:

An FRU corresponding to a functional block is described with the *fru* relation. For example, the FRU for lm_intf is lmc_pkg(Fig-6).

(d)Subblock:

A functional blocks is partitioned into subblocks so that each property within a block has one value. Subblocks of a functional block are also represented with *sub_block* relations. In Fig-6, lm_intf has two subblocks, namely lm_intf_g and lm_intf_3g. Common properties between subblocks are represented in a super block, and they are inherited by subblocks through the *sub-block* relations.

(e)Faultability :

A faultability value for the functional block is described. It is referred to in enumerating possible FRU's.

Symptom and Test Knowledge

In a symptom object, symptom-specific knowledge is described compactly. Signal-paths corresponding to symptom are described in a symptom object. The signal-path description is in the following form:

```
signal_path_A ::
        start_block    : block_B;
        end_block      : block_C;
        path_relation  ; path_relation_D.
```

In addition, effective tests for symptom, and tests to check symptom occurrences are described in symptom objects.

In test objects, test execution methods, and methods for narrowing the suspects down are described.

Narrowing the suspect is performed by either of two methods. One is based on the abstract signal flow. This method is for tests to check the functions of signal flows, as described in 4.2. The other method is suspect elimination by functional blocks' properties. For example, if the result of a test to find the *dual_single* property of the defective component tells that *single_function_block* is not suspected, functional blocks with the dual_single property being *single_function_block* are eliminated from suspects.

In addition, criteria for the effectiveness of tests, and effectiveness values are also described in test objects.

Diagnosis Control Knowledge

The diagnosis control knowledge specifies diagnosis flow and criteria for subsequent actions.

Unlike other rule-based diagnosis expert systems, control knowledge is implemented with domain-specific control mechanisms written in corresponding objects as Prolog procedures, namely *methods*. This is motivated by the fact that through pre-prototyping with a forward reasoning rules, the authors found that it was more natural and flexible to write inference knowledge as methods than as a set of rules. Since this also allows us to utilize the multiple inheritance mechanism through the relations, the knowledge base becomes more compact and more modifiable.

7. Man-machine Interface

A graphic-oriented and menu-driven man-machine interface for easy operation is provided, based on the multiple window environment of an engineering workstation (NEC EWS4800) as shown in Fig-7. It shows several kinds of information in separate windows on the EWS. Several windows displaying the following information are provided as follows:

- System recommendation for a subsequent action
- System status and messages
- Fault message log, or complaint log
- Suspected components
- FRU location(floor, frame and module layout windows)
- Explanation for the action
- ESS input/output
- Menu input

In particular, a detailed location of the suspected FRU is graphically shown by floor layout, frame layout and module layout diagrams. Using these displays and the menu-driven interface, even an unaccustomed user can easily use the system.

8. Concluding Remarks

This paper has presented the architecture of SHOOTX. By combining a reasoning facility modeled on know-how of human experts and interface functions with the target ESS, SHOOTX enables diagnosis of faults which cannot be diagnosed by built-in functions.

The prototype system without full interface functions to ESS has been developed, and is in a demonstrative evaluation phase. It has been demonstrated to many experts. By operating it themselves, they have realized that it greatly improves operation and maintenance efficiency. Moreover, it is also helpful in training maintenance personnel due to flexible and easily understandable man-machine interface facilities.

Further research is necessary for easier knowledge acquisition. Adding and modifying pertinent knowledge requires special care, though the methods are relatively easier than rule-based representation. Prolog-based descriptions of procedures is not very understandable. For more efficient development and maintenance of the knowledge base, research on more effective representation and learning method is desired.

Acknowledgments

The authors thank Mr. N. Wakasugi, Mr. M. Kondo, Mr. M. Oishi, and Mr. T. Hisada for SHOOTX implementation. The authors thank Mr. K. Toudou, Mr. H. Igarashi and their colleagues for their patient cooperation during the knowledge acquisition study. The authors also express deep appreciation to Mr. A. Kitamura, Mr. Hiroyuki Mori, Mr. T. Ando, Mr. Y. Souda, Dr. S. Goto, and Mr. Hajimu Mori for continuous encouragement and support. Finally, the authors would like to thank Mr. Y. Fujii, of the Institute for New Generation Computer Technology for various kinds of support.

REFERENCES

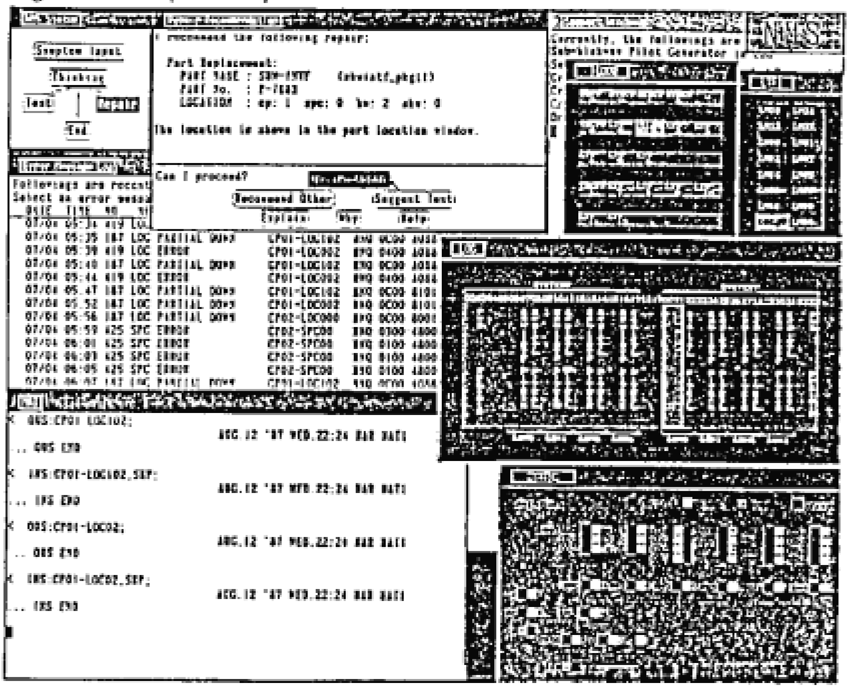
[Davis84]
R. Davis, "Diagnostic Reasoning Based on Structure and Behavior," *Artificial Intelligence* 24, 1984, pp. 347-410.

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

[Koseki87a]
Y. Koseki, S. Wada, T. Nishida and H. Mori, "SHOOTX: A Multiple Knowledge Based Diagnosis Expert System for NEAX61 ESS," *Proc. of the International Switching Symposium*, Phoenix, March 1987, pp. 78-82.

[Koseki87b]
Y. Koseki, "Amalgamating Multiple Programming Paradigms in Prolog," *Proc. of the 10th International Joint Coference on Artificial Intelligence*, Milan, Italy, August 1987, pp. 76-82.

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



```
I recommend the following repair:

 Part Replacement:
   PART NAME : SOM-19TF    (ehwsatf_pkg11)
   PART No.  : P-7603
   LOCATION  : sp: 1  spc: 0  bw: 2  sh: 0

The location is shown in the part location window.

Can I proceed?
```

```
Followings are recent
Select an error messa
 DATE  TIME  NO
07/04 05:34 419 LOC
07/04 05:35 187 LOC PARTIAL DOWN     CP01-LOC102  ANQ 0C00 A088
07/04 05:39 419 LOC ERROR            CP01-LOC002  BNQ 0400 A088
07/04 05:40 187 LOC PARTIAL DOWN     CP01-LOC102  BNQ 0C00 A088
07/04 05:44 419 LOC ERROR            CP01-LOC002  BNQ 0400 A088
07/04 05:47 187 LOC PARTIAL DOWN     CP01-LOC102  BNQ 0C00 8100
07/04 05:52 187 LOC PARTIAL DOWN     CP01-LOC002  BNQ 0C00 8100
07/04 05:56 187 LOC PARTIAL DOWN     CP02-LOC000  BNQ 0C00 8000
07/04 05:59 425 SPC ERROR            CP02-SPC00   BNQ 0100 4800
07/04 06:01 425 SPC ERROR            CP02-SPC00   BNQ 0100 4800
07/04 06:03 425 SPC ERROR            CP02-SPC00   BNQ 0100 4800
07/04 06:05 425 SPC ERROR            CP02-SPC00   BNQ 0100 4800
07/04 06:07 187 LOC PARTIAL DOWN     CP01-LOC102  BNQ 0C00 A088
```

```
< OBS:CP01-LOC102;
                         AUG.12 '87 WED.22:24 BAR BAT1
... OBS END
< INS:CP01-LOC102,SEP;
                         AUG.12 '87 WED.22:24 BAR BAT1
... INS END
< OBS:CP01-LOC02;
                         AUG.12 '87 WED.22:24 BAR BAT1
... OBS END
< INS:CP01-LOC02,SEP;
                         AUG.12 '87 WED.22:24 BAR BAT1
... INS END
```

```
Currently, the followings are
Sub-blocks Pilot Generator
```

Fig-7 : User interface screen