INCORPORATING MODEL-BASED REASONING IN INTERACTIVE MAINTENANCE AIDS

by

John W. Sheppard
William R. Simpson

ARINC Research Corporation
2551 Riva Road
Annapolis, MD 21401

ABSTRACT

Until recently, there have been only three types of maintenance decision aids: the electronic fault tree, the electronic simulation, and the rule-based expert system. Electronic fault trees present fault-isolation material about a system according to a precomputed diagnostic strategy. An electronic simulation uses a mathematical representation of the system to identify anomalous behavior. Finally, rule-based expert systems process a set of heuristics, or rules-of-thumb, specified by an expert diagnostician to determine how "best" to diagnose the system. In this paper, we propose an approach to generating an "intelligent" maintenance decision aid, which we have implemented in a tool called POINTER™: Portable Interactive Troubleshooter. The approach involves developing a mathematical model of the system to be tested and then reasoning about the system to determine which test to choose and what one can deduce from the test outcome. The mathematical model is based on the flow of information through the system and the type of information available at each "test point" in the system. The corresponding test dependencies are then identified. This model is used to guide the fault-isolation process. In addition to the basic fault-isolation capability, the model provides tremendous flexibility for diagnosing a system. Failures can be hypothesized. Test choices can be overridden or delayed. Various optimization criteria can be specified. Detailed explanations of the current state of the problem can be provided. In addition, new optimization criteria and dependency relationships can be learned. Each of these capabilities will be discussed in detail.

INTRODUCTION

Background

The ARINC Research System Testability and Maintenance Program (STAMP®) is a computer-based model that is used to conduct testability analyses and develop fault-isolation strategies to improve system maintenance. The dependency modeling approach incorporated by STAMP permits analysis of a wide range of systems, including digital, analog, digital/analog hybrid, electrohydraulic, and electromechanical. STAMP has been used to analyze systems in various stages of the acquisition process: preliminary design, prototype, redesign, and operational. Several of these applications have included analyses of built-in test (BIT) or have used built-in test equipment (BITE) and other forms of automatic and semiautomatic test equipment. In addition, the particular levels of analyses have varied from macro (full system) to micro (piece-part level), with several levels in between (e.g., line and shop replaceable units [LRUs and SRUs] or weapons and shop replaceable assemblies [WRAs and SRAs]).

For many systems that STAMP has analyzed, significant improvements have been achieved, and for some systems, order-of-magnitude improvements have been achieved [1, 2]. The software is mature and has been used on more than 50 systems. It is currently being used by ARINC Research in a number of applications. Figure 1 shows the functional flow of a STAMP analysis. As the figure shows, the key element in providing STAMP analyses is the development of a system model that serves as the STAMP knowledge base.

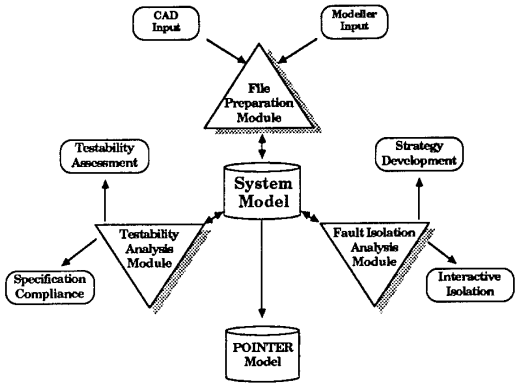

Figure 1. STAMP Functional Flow

POINTER™ (Portable Interactive Troubleshooter) was derived from STAMP. It also uses the system model for its knowledge base. With the system model, POINTER interactively presents test material and guides the maintenance process through fault isolation. At each step, POINTER examines the current system state to determine the next best test to perform. The basic processes of modeling and knowledge-base development, as well as the testability and fault-isolation output, are described in depth in the literature [3–5] and are not discussed in this paper.

Diagnosis: Art Versus Science

The Art

The general problem of diagnosis is extremely complex. The process of determining the proper tests to perform, performing and evaluating these tests, determining the failure, and correcting the problem, given some set of symptoms, conditions, and operational history, is called a diagnostic strategy. Diagnostic strategies are often developed by the system "experts." Because of the complexity of the problem, there exist wide variations in the strategies for different systems that may have the same symptoms. These variations have been accepted because it is rarely evident whether a particular strategy is optimal or not (e.g., least costly, most efficient in the use of resources, or most complete).

The Science

Early attempts at developing computer-assisted tools for diagnosis sought to emulate the reasoning process of the experts.

CH2881-1/90/0000-1238 $1.00 © 1990 IEEE

Heuristics were coded into rule-based expert systems that attempted to provide the captured expertise to a wider audience. Two landmark examples are MYCIN and INTERNIST. MYCIN is an expert system, developed at Stanford University, that advises on diagnosis and treatment for infectious diseases [6]. INTERNIST, developed at the University of Pittsburgh, is an expert consultant in the area of internal medicine [7].

The problem of optimal design of diagnostic decision trees was addressed as early as 1960 by R. A. Johnson [8]. Johnson proposed constructing these trees by using the information gained per dollar for each test for each successive diagnostic step. Subsequently, some work in the application of information theoretics and the design of diagnostics has been pursued [9, 10]. More recently, W. R. Simpson proposed an approach that treated the problem of diagnosis as an information modeling problem. This approach has been extensively used in the area of electronic, mechanical, and flow system fault isolation by the computer shell called STAMP [11]. The approach was extended to the medical domain in 1986 with the development of a protocol for the diagnosis and treatment of urinary tract infections in female patients [12], and again in 1989 with the development of an emergency room protocol for treating penetrating truncal injuries [13].

Past Approaches to Computer-Aided Diagnosis

To date, the most frequently used type of diagnostic system has been the rule-based expert system. The expert system approach to developing diagnostic protocols involves constructing a knowledge base of IF. . .THEN. . . rules to represent the diagnostic process and then using that knowledge base to determine the best approach for performing the diagnosis [14]. During the reasoning process, chains of related information (or propositions) are analyzed by an inference engine. These propositions consist of the possible outcomes associated with the information sources while the relationships between propositions constitute the links in the chain. The reasoning process involves identifying the chain or chains that connect some goal (i.e., conclusion) with established facts (i.e., propositions whose outcomes are known). Chaining either proceeds from the set of established facts to the goal states *(forward chaining)*, or it proceeds from some hypothesis (goal) to the facts needed to verify the hypothesis *(backward chaining)*. It may also proceed as combinations of forward and backward chaining.

AN INFORMATION THEORETIC APPROACH—Developing an Information Flow Model

The STAMP system differs from rule-based expert systems in a very important way. STAMP processes a mathematical model of the information flow through a system rather than through a set of IF. . .THEN. . . rules. In order to construct an information flow model in STAMP, one of two approaches may be used.* For users who are familiar with a system, a functional model of that system can be constructed. This model consists of tests, a set of diagnostic conclusions, and a set of interdependencies between tests and conclusions. This approach usually requires a level of understanding of a system that is approximately one-half to one level below that required for effective diagnosis. For example, to model a system at the card level, some knowledge of the circuitry on the cards is required.

The second approach begins with a set of tests and diagnostic conclusions (or knowledge of the testing process) and produces a model that is based on the expected test outcomes for the diagnostic conclusions within the model. These outcomes are recorded, and an analysis of the logical relationships is performed by STAMP to determine interdependencies between tests. This analysis procedure is called logical closure and is based on STAMP inference meta rules [16]. Further, this approach requires only a level of understanding at the required level of diagnosis, including a thorough understanding of the specific tests to be performed.

INTERACTIVE MAINTENANCE AIDS

As part of its ongoing Independent Research and Development Program, ARINC Research continues to develop STAMP and STAMP-related technologies. Our most recent research tasks have been in the area of interactive maintenance aids, in general, and intelligent portable maintenance aids, in particular. An interactive maintenance aid is an electronic presentation of fault-isolation and repair material that assists a maintenance technician in diagnosing faults and repairing faulty systems. We have defined three basic categories of maintenance aids:

- ***Electronic Manuals:*** A machine representation of the technical manuals that uses static isolation procedures and test sequences in an on-line text display for use by the field technician. These devices have minimal machine requirements but provide little in the way of flexibility, training, and logistic support; they also cannot learn. Details of two such applications by ARINC Research are contained in Reference 17.

- ***Intelligent Maintenance Assistants:*** A computer program with an extensive knowledge base of the unit under test (UUT). These devices provide a considerable amount of flexibility by computing the next maintenance action using the currently known information and the problem context. In order to obtain the desired level of flexibility, a moderate level of effort is required to develop the knowledge base. These devices can be used in fully automatic, semiautomatic, or manual isolation and can provide a direct link to logistics and other data bases. Test procedures must be defined, and machine capabilities are important, although some of today's small portable and laptop computers may be capable of hosting this type of system. The rule-based expert system is an example of this type of maintenance assistant, but other systems are included in this category.

- ***Electronic Simulation:*** A computer program with a detailed physical model of the system. This is the most capable of the maintenance aids. Test procedures may be interactively developed, and some prognosis development may be possible (e.g., time trending and pattern analysis work may be performed). Building the model requires extensive information and effort, and for moderate to large systems, the required machine capabilities will exceed current small computer capabilities. Such maintenance aids are in use in factory production processes and NASA facilities for flight-critical hardware.

POINTER

POINTER is a model-based intelligent maintenance aid that dynamically computes fault-isolation strategies on the basis of

*Although STAMP uses an information flow model to describe a system for diagnosis, we have also developed procedures for processing IF. . .THEN. . . rules directly. This process is described in detail in Reference 15.

problem context using the information theoretic approach previously described. Figure 2 shows the functional flow of a POINTER application.

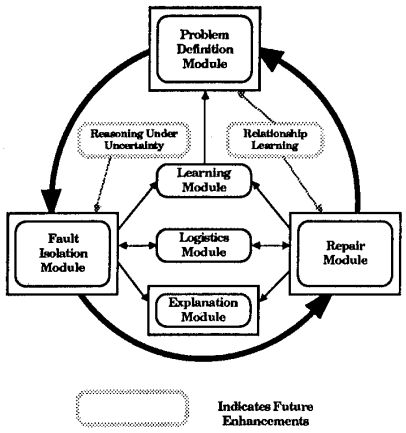

Figure 2. POINTER Functional Flow

POINTER requires a functional dependency model of the system to be tested for its knowledge base. This model is obtained as an output from the STAMP software system. To this may be added textual material for test and repair procedures; graphics material for display at appropriate times in fault isolation and repair; a number of intermediate screens and menus; and programs that can be executed independently to provide specific information, execute tests, initiate reconfiguration, and link to computer-aided design (CAD), hypertext, and other packages.

USER OPTIONS DURING FAULT ISOLATION

After a test is chosen, the user is presented with a test procedure to perform. At each point in the fault-isolation process, a number of options may be available, depending on user privileges and information available to the system. The system can be set up to include any or all of the following user privileges:

- ***Good:*** A test outcome of good may be declared after the test is executed.
- ***Bad:*** A test outcome of bad may be declared after the test is executed.
- ***Untestable:*** A test may be declared untestable if the test is unable to be performed for any reason, such as insufficient test equipment or skill level.
- ***Delay:*** The performance of a test may be delayed whenever conditions warrant (e.g., difficult-to-access test points, required equipment momentarily out of reach.) POINTER may choose the delayed test at a later point, if needed.
- ***Override:*** In some cases, the technician may wish to specify the next test to be conducted. This is accomplished with the override command. If the outcome of that test has been previously provided or inferred during the isolation process, the user will be advised.
- ***Last Test:*** At any point during fault isolation, the user may back up and redisplay the last test. Repeated use of this feature can back up the fault-isolation procedure to any place in the sequence.
- ***Stop:*** Fault isolation may be terminated at any point with this command. If this option is exercised, an answer will be provided that is appropriate to the level of testing completed. However, this answer will contain a larger ambiguity group than when testing is completed.
- ***Hypothesis:*** When users believe they know the cause of the problem, they may enter the suspected failure as a hypothesis, and POINTER will redirect its test choice algorithm to verify that hypothesis. If the failure presented as a hypothesis is no longer in consideration as the probable cause, the user is advised. The user may then go to the explanation facility for a complete explanation of why that failure is not considered. If the result of any test violates the hypothesis, then the user is advised, confirmation is requested, and fault isolation continues under the "no hypothesis" information choice.
- ***Picture:*** When graphics assistance is needed for any reason, this option allows a picture associated with the current test to be displayed; any test may have a picture associated with it.
- ***Explain:*** A great amount of information, concerning the why, what, and how of the current situation, is available through the explanation module. This module is described in a separate section of this paper.
- ***Abort:*** This option terminates the current isolation procedure and returns the user to the problem definition module.
- ***Execute:*** This prompt appears only when a separately executable program has been associated with a test. Such a program may actually execute the test and return the answer to POINTER. In the automatic mode, tests may execute when chosen by POINTER or the user without any interaction with the user.
- ***Program Interface:*** A program may be linked to POINTER and made available to the user at any point in the diagnostic or repair session. The associated prompt may be predefined and placed at the prompt line for access. Examples include hypertext interface or amplified help programs. One engineer at ARINC Research placed a word processor on this prompt so that it was possible to interactively rewrite test procedures while conducting the tests during verification testing in the laboratory.
- ***Group Setup:*** Throughout the execution of POINTER, a number of setup screens may be placed in line for transition. When POINTER first enters a test group, the group setup screen, if it exists, is displayed. Thereafter, it is accessible by entering *!* at the prompt. Note that any setup screen within POINTER may have an executable program associated with it.
- ***Help:*** Every prompt in POINTER has a comprehensive on-line help function.

During the fault-isolation process, each test is timed, and the value is retained for learning purposes. At the completion of repair, these values are used to update the learning file using an 11-point, moving average algorithm. In addition, each step in the diagnostic process is recorded in a log file for future use in logistic analyses.

AIDS TO REPAIR

The repair module may be entered under two conditions:

- User has terminated fault isolation by use of the stop command.
- POINTER has isolated to an element or group of elements that cannot be further refined.

When the repair module is entered, the user is presented with the isolation answer. If more than one element (component, replaceable unit group, or failure group) is present in ambiguity, the user is also presented the computed probability of each failure on the basis of the failure rates stored with the data file or in the learning file. At this point, the user may choose to display a repair procedure. If historical information is available, it is displayed, and the user may review the maintenance file for the previous time that item was repaired. Also, a full series of explanation options is available at this time. (This is further discussed in the section on the explanation module.) If desired, the user may back up to fault isolation from the repair module and, for example, continue fault isolation if the user command *stop* was originally issued.

If the learning option has been invoked, POINTER asks for specific repair information, including what was repaired and the operating hours associated with the repair. These are placed in the log file, and new failure rates are computed for the learning file. Conflicts between actual repairs and isolations are stored in a file to be used in future extensions of learning. Upon exit from the repair module, the technician is prompted for narrative comments. (Note: The technician also has the option to add narrative comments to the log file while at both the fault-isolation and the repair menus.)

EXPLANATIONS DURING ISOLATION

The explanation module has the full power of the inference engine to provide information and analyses to the user. A secondary menu is provided and may include:

- ***Tests Given or Inferred Good:*** Tests determined to be good are listed, and the option is provided to view the reasoning that determined the good outcome.
- ***Tests Given or Inferred Bad:*** Tests determined to be bad are listed, and a reasoning option (as in good outcomes) is available.
- ***Tests Untestable (or Unavailable):*** Tests declared untestable or unavailable are listed, and the reasoning option (as in good outcomes) is available. An analysis of any ambiguity groups resulting from declaring a test or tests untestable is also available.
- ***Components, Replaceable Units, and Multiple Failures No Longer Being Considered:*** Conclusions eliminated from consideration are listed, and an explanation of the elimination reasoning is available.
- ***Tests That Are Unknown and Available:*** Tests still available for performing are listed with or without any associated weighting factors (e.g., test time or skill level).
- ***Actively Considered Answers:*** Conclusions or answers that have not been eliminated from consideration are listed with this option. The list comprises the ambiguity group that would exist if isolation were terminated.
- ***Why This Test:*** This option provides an explanation of the test choice, including the value of weighting factors considered, compared with the range of values available and the differentiation ability of the test in terms of separating the conclusions.
- ***Consequences of Not Doing the Test:*** An analysis of the consequences of declaring a test untestable may be performed with this option. The analysis includes efficiency changes, ambiguity groups, and an analysis of related tests that have been declared untestable.
- ***Hypothesis:*** The steps necessary to verify a specific answer on the basis of current information are given when this option is selected.

EXPLANATIONS AFTER ISOLATION

A number of analyses and explanations are available after isolation has been achieved. These are again provided by the explanation module using the full power of the inference engine.

- ***Verification Steps:*** This feature provides the user with the minimum number of steps (with their expected outcomes) necessary to verify the isolated failure. The technician has the option of verifying with or without previously declared tests. This is a good feature for quality maintenance.
- ***Inference Trace:*** The user is able to step through the testing that was just completed and examine a list of the components under consideration at each step. This is a good feature for training by determining the points at which certain conclusions are eliminated.
- ***Hidden Failure Analysis:*** A list of failures whose presence cannot be identified with the current failure isolation is given when this option is selected. If any of the members of this list could be the root cause of the isolated failure (i.e., a failure of a hidden unit would cause the failure of the isolated unit), a maintenance problem can be avoided by repairing both failures.
- ***Failure Combinations with Identical Symptoms:*** This feature scans the knowledge base for combinations of failures that have identical symptoms. If one of these multiple failures exists, repair of the isolated components will not restore the system. The maintenance technician can then consider a multiple repair in a logical fashion.

LEARNING

During the fault-isolation session, POINTER times the tests as they are performed. The test times are then recorded and combined with previously recorded test times to derive a new test time measure. This measure is used by POINTER to improve fault isolation when attempting to minimize the time required to fault-isolate.

In addition, POINTER records the repairs made to the system with the current number of hours of system operation. Failure rates are then recomputed on the basis of the repairs and operating hours, and the new failure rates are used by POINTER when attempting to isolate failures using failure probability.

LOGISTICS

In addition to recording test times and failure rates for improving fault-isolation performance, POINTER maintains two sets of files that can be used in logistics documentation. First, the learning file associated with each POINTER model contains

information on test times, skill level, failure rates, number of recorded failures, the most recent operating hours for each repair, and a link to a log file. The second set of files comprises log files.

Each fault-isolation session creates a log file containing information about that session. Each log file includes a description of the setup conditions for fault isolation, a record of the test sequence, a list of failures identified, any repair actions, and comments provided by the technician. Further, the test sequence information includes the times to perform each test, all POINTER actions taken by the technician, and the test outcome. If learning takes place, information on how test times and failure rates changed are also included. Finally, each log file is linked to the previous log file associated with a repair of the same failure (if one exists).

The information provided in these files does not include the results of any logistics analyses. It does, however, provide some of the data required for such analyses, or other files can be used with a separate documentation system that records and analyzes logistics information.

POINTER APPLICATIONS

POINTER is a new product, first developed in 1988, whose potential as a maintenance and troubleshooting tool is significant. Four applications have been completed using POINTER:

- Under work sponsored by the Electric Power Research Institute (EPRI), ARINC Research, in conjunction with the International Fuel Cells Corporation (IFC), developed an intelligent portable maintenance aid for a multi-megawatt phosphoric acid fuel cell power plant.
- Under work sponsored by the U.S. Army, ARINC Research developed an intelligent portable maintenance aid for the Remote Relay System (RRS) of the GUARDRAIL V COMINT aircraft.
- Under work sponsored by the U.S. Navy, ARINC Research developed an intelligent portable maintenance aid for the AV-8B stores management system high voltage power supply. The maintenance aid is being used at the intermediate maintenance level repair facility (shop level repair).
- Also under work sponsored by the U.S. Navy, ARINC Research developed an intelligent portable maintenance aid for the AV-8B stores management fuzing power supply. This is being used for shop level repair.

SUMMARY

The POINTER intelligent maintenance aid system offers a significant advance in the state-of-the-art in field maintenance. Its direct linkage to STAMP and the STAMP modeling approach provides a diagnostic concept that can be truly integrated over the life cycle of the system to which it is applied. STAMP can be applied at all points in the life cycle, from concept formulation to fielding. And now when fielding occurs, the POINTER system can be used to guide the troubleshooting and repair process.

REFERENCES

1. W. R. Simpson, "STAMP Testability and Fault Isolation Applications 1981-1984," *1985 Proceedings of the IEEE AUTOTESTCON*, Uniondale, Long Island, New York, October 1985.
2. E. A. Esker, "Testability Analysis: Applications in the Real World," *Proceedings of the IEEE Integrated Diagnostics Symposium*, Dayton, Ohio, December 1985.
3. W. R. Simpson, "Testability and Fault Diagnosis of Airline Avionics," Special Edition of *PLANE TALK*, AMC Open Forum, Houston, Texas, March 1983.
4. W. R. Simpson, et al., "Multidimensional Context Representation of Knowledge-Base Information," 1987 Data Fusion Symposium, Laurel, Maryland, June 1987.
5. B. A. Kelley, et al., "The Use of Information Theory in Propositional Calculus," 1987 Data Fusion Symposium, Laurel, Maryland, June 1987.
6. R. Davis, B. Buchanan, and E. Shortliffe, "Production Rules as a Representation for a Knowledge-Based Consultation Program," *Artificial Intelligence*, vol. 8, no. 1, 1988, pp. 15-45.
7. H. E. Pople, "The Formulation of Composite Hypotheses in Diagnostic Problem Solving: An Exercise in Synthetic Reasoning," *Proceedings of the Fifth International Joint Conference on Artificial Intelligence*, 1977, pp. 1030-1037.
8. R. A. Johnson,"An Information Theory Approach to Diagnosis," *Proceedings of the 6th Annual Conference on Reliability and Quality Control*, 1960, pp. 102-109.
9. C. P. Hartman, et al., "Application of Information Theory to the Construction of Efficient Decision Trees," *IEEE Transactions on Information Theory*, vol. IT-28, no. 4, July 1982, pp. 565-577.
10. T. R. Addis, "Knowledge Refining for a Diagnostic Aid (An Example of Applied Epistemics)," *The International Journal of Man-Machine Studies*, vol. 17, 1982, pp. 151-164.
11. M. Cramer, R. M. Hecht, R. H. Kirschman, W. R. Simpson, and S. R. Troy, *Applications Guide, Logic Model Analysis and Standard Maintenance Information Display (SMIDS)*, ARINC Research Corporation, 1784-01-1-2394R, October 1981.
12. W. R. Simpson, B. A. Kelley, E. M. Simpson, and I. R. Horowitz, "An Artificial Intelligence Approach to Developing Medical Protocols," *Proceedings of the 5th World Congress on Medical Informatics*, 1986.
13. J. W. Sheppard, "Learning Diagnostic Information Using a Matrix-Based Approach to Knowledge Representation," Master of Science Thesis, G. W. C. Whiting School of Engineering of the Johns Hopkins University, October 1989.
14. W. R. Simpson and B. A. Kelley, "Multi-dimensional Context Representation of Knowledge Base Information," *Proceedings of the Tri-Service Data Fusion Symposium*, Johns Hopkins Applied Physics Laboratory, Laurel, Maryland, 1987.
15. John W. Sheppard and William R. Simpson, "A Verifiable Representation of Knowledge," submitted to the Eighth National Conference on Artificial Intelligence, Boston, Massachusetts, 1990.
16. William R. Simpson, "Notes on Logical Closure," STAMP Technical Note 340A, ARINC Research Corporation, November 1988.
17. W. R. Simpson, "Active Testability Analysis and Interactive Fault Isolation Using STAMP," 1987 IEEE AUTOTESTCON, San Francisco, California, November 1987.