POINTER—AN INTELLIGENT MAINTENANCE AID

William R. Simpson
John W. Sheppard
C. Richard Unkle

ARINC Research Corporation
2551 Riva Road
Annapolis, Maryland 21401

ABSTRACT

After eight years of development and application, the ARINC Research Corporation System Testability and Maintenance Program (STAMP®) has been expanded to provide a portable, intelligent, interactive diagnostic tool—the portable interactive troubleshooter (POINTER). POINTER is an artificially intelligent, portable maintenance aid that utilizes the STAMP system model, test choice, and inference algorithms to provide a dynamically tailored fault-isolation process. With POINTER, the user may declare test results, hypothesize failures, and override or delay the specified test, and have POINTER select another test to perform. During fault isolation, the user receives the fault identification and may request repair procedures, isolation recap with verification procedures, or a variety of analyses for multiple failures. In addition to the ability to dynamically tailor the fault-isolation session to the current situation, the user has access to a complete explanation facility. Finally, POINTER logs the entire analysis and can modify its search strategy by incorporating, through a learning algorithm, the actual test times and failure rates experienced.

INTRODUCTION

POINTER BACKGROUND

The ARINC Research Corporation-developed portable interactive troubleshooter—POINTER—was derived from the System Testability and Maintenance Program (STAMP®), also developed by ARINC Research. STAMP is a set of computer-based analysis techniques and tools that are used to conduct testability analyses and develop fault-isolation strategies to improve system maintenance. The tools and techniques work from a functional dependency model of the system being analyzed. Because the functional model is the primary input to STAMP, a wide variety of systems may be analyzed, including digital, analog, hybrid, electromechanical, and electrohydraulic. In addition, STAMP may be used to analyze systems in various stages of the acquisition process: preliminary design, prototype, redesign, and operational.

The applications for which STAMP has been used are quite varied and cover the various phases of a system life cycle. These applications have also included analyses of built-in test (BIT) or have used built-in test equipment (BITE) and other forms of automatic and semi-automatic test equipment. In addition, the particular level of analysis has 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 weapon- and shop-replaceable assemblies [WRAs and SRAs]). The STAMP software is mature, having been used to analyze more than 50 systems. For many of these systems, significant improvements have been achieved, and for some systems, order-of-magnitude improvements have been achieved.[1,2]

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 knowledge base to STAMP. 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.

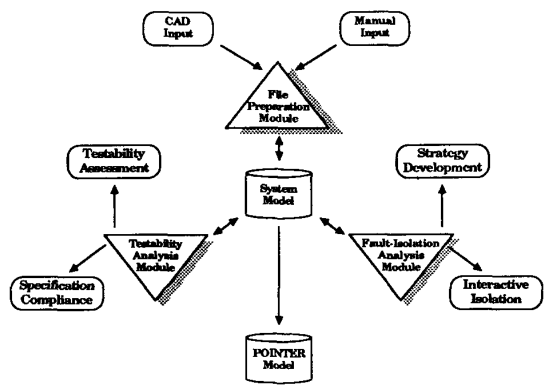


FIGURE 1. STAMP ANALYSIS FUNCTIONAL FLOW

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

CH2568-4/89/0000-0026 $1.00@ 1989 IEEE

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 are also unable to learn. Details of two such applications by ARINC Research are contained in reference 6.

- *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 of use in fully automatic, semi-automatic, 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.

- *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 prognostication 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 problem context and user input. It allows a full range of user options for both isolation and repair and can be tied to logistics documentation systems. It can be completely tailored to the individual application through a series of application-specific menus, and separately executed programs can be run from within POINTER. Figure 2 shows the functional flow of a POINTER application. The remainder of this paper expands upon the elements in that figure.

POINTER PROBLEM-DEFINITION MODULE

INPUT

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

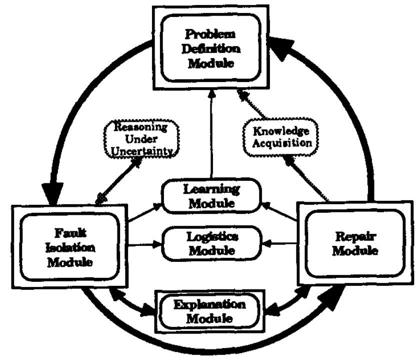


FIGURE 2. POINTER PROCESS FUNCTIONAL FLOW

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 links to computer-aided design (CAD), hypertext, and other packages.

PROBLEM-SPECIFIC LAYERED MENUS

Once the proper input files are created and processed, the troubleshooting aid, POINTER, is ready to use. The problem-definition module presents layered menus that allow the user to choose the specific knowledge base to be executed and then a separate set of menus from which to choose from among the symptoms presented by the maintenance problem. Figure 3 shows the symptoms and conditions menu for the dc generating section of a fuel cell power plant (see applications section and reference 7). The user may choose from among these symptoms and conditions or enter with no symptoms and conditions. The menu presented may be multipaged and contain up to 999 different symptom sets. A menu choice, in addition to setting up the symptoms, will initiate logging files, learning, and fault-isolation criteria, such as minimum skill level or minimum time to fault-isolate.

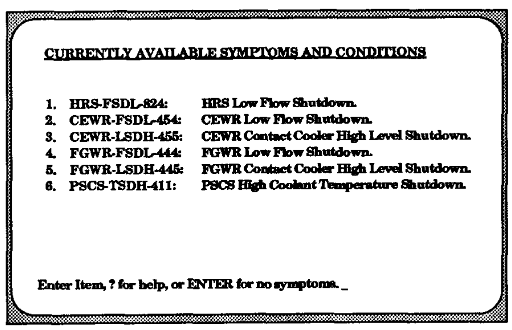


FIGURE 3. POINTER DIAGNOSTIC SYMPTOMS AND CONDITIONS MENU

POINTER FAULT-ISOLATION MODULE

BASIC APPROACH

POINTER uses the STAMP algorithms for fault isolation to determine the "best" test to perform at the current point in a fault-isolation session. The STAMP algorithms use an information theoretic approach to choosing tests and are flexible enough to allow for the following:

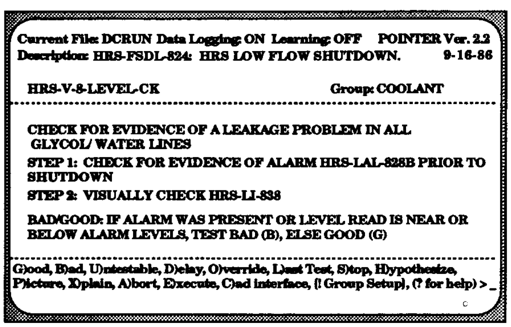
- Multicriterion Optimization (such factors as skill level, test time, and test cost may be included in the optimization)
- Test Grouping and Sequencing
- Replaceable Unit Isolation
- Multiple Failure Fault Trees
- Hypothesis-Directed Searches

These factors are discussed generally in reference 5 and specifically in references 8 through 11. The multicriterion optimization can also be used in a learning mode where test times and failure rates may be learned through experience. This learning process is detailed in reference 12.

USER OPTIONS DURING FAULT ISOLATION

After a test is chosen, the user is presented with a test procedure to perform as shown in Figure 4. 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 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, equipment required momentarily out of reach). (POINTER may choose the delayed test at a later point, if needed.)


Current File: DCRUN Data Logging: ON Learning: OFF POINTER Ver. 2.2
Description: HRS-FSDL-824: HRS LOW FLOW SHUTDOWN. 9-16-86

HRS-V-8-LEVEL-CK Group: COOLANT

CHECK FOR EVIDENCE OF A LEAKAGE PROBLEM IN ALL GLYCOL/ WATER LINES
STEP 1: CHECK FOR EVIDENCE OF ALARM HRS-LAL-828B PRIOR TO SHUTDOWN
STEP 2: VISUALLY CHECK HRS-LI-838
BAD/GOOD: IF ALARM WAS PRESENT OR LEVEL READ IS NEAR OR BELOW ALARM LEVELS, TEST BAD (B), ELSE GOOD (G)

G)ood, B)ad, U)ntestable, D)elay, O)verride, L)ast Test, S)top, H)ypothesize, P)icture, X)plain, A)bort, E)xecute, C)ad interface, {! Group Setup}, (? for help) > _

FIGURE 4. POINTER TEST PROCEDURE INSTRUCTIONS

- *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 if testing were 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. A picture appropriate to the test procedure of Figure 4 is presented in Figure 5.

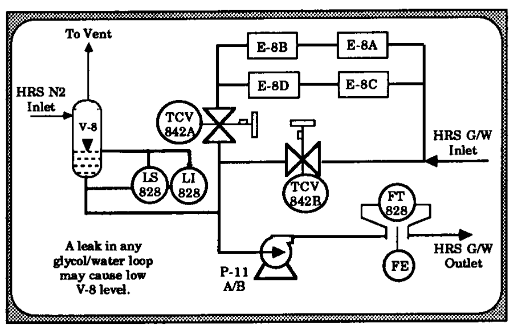


FIGURE 5. REPRESENTATIVE POINTER GRAPHICS CAPABILITY

- *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.

- *CAD Interface:* A program may be linked to POINTER and made available to the user at any point in the diagnostic or repair session. The CAD interface option shown in Figure 4 is one example making use of this feature. The associated prompt may be predefined and placed at the prompt line for access. Other examples include hypertext interface or amplified HELP programs. One engineer at ARINC Research placed a 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 screen within POINTER may have an executable program associated with it. (See reference 5 for more information on test groups.)

- *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, smoothing algorithm. In addition, each step in the diagnostic process is recorded in a log file for future use in logistic analyses.

POINTER REPAIR MODULE

AIDS TO REPAIR

The repair module may be entered upon 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.

RECORDING REPAIRS FOR LEARNING

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 into 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.

POINTER EXPLANATION MODULE

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 it 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 that are 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 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 will provide the user with the minimum number of steps (with their expected outcomes) necessary to verify the isolated failure. 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 the 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.

OTHER MODULES

LEARNING MODULE

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 MODULE

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 failure 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. There are currently four applications in process using POINTER:

- Under work sponsored by the Electric Power Research Institute (EPRI), ARINC Research, in conjunction with the International Fuel Cells Corporation (IFC), is developing an intelligent portable maintenance aid for a multimegawatt phosphoric acid fuel cell power plant.

- Under work sponsored by the U.S. Army, ARINC Research is developing 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 is developing an intelligent portable maintenance aid for the AV-8B stores management system high voltage power supply. The maintenance aid will be used at the intermediate maintenance level repair facility (shop-level repair).

- Also under work sponsored by the U.S. Navy, ARINC Research is developing an intelligent portable maintenance aid for the AV-8B stores management fuzing power supply. This also will be used for shop-level repair.

FUTURE RESEARCH

As exciting as these concepts are, the work is not yet complete. We are continuing research for the development of even more sophisticated tools in the future. Two areas of concentration in the near term are shown in Figure 2 (grey tone) and discussed below:

- *Reasoning Under Uncertainty:* Often when test procedures are inadequate or systems contain the ability to dynamically reconfigure on the basis of different modes of operation and system failure, the outcomes of tests or the system relationships may be uncertain. We have developed an approach to incorporating uncertainty factors for these types of failures.

- *Relationship Learning:* It is possible for system models used in POINTER to be incomplete or inaccurate. Techniques are being explored that will examine the differences between POINTER isolations and actual system repairs to determine where the model failed to represent the actual system. The model can then be modified automatically to contain the new information.

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 is truly integrated over the life cycle of the system. 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. W. R. Simpson, "Active Testability Analysis and Interactive Fault Isolation Using STAMP," 1987 IEEE AUTOTESTCON, San Francisco, California, November 1987.

7. C. R. Unkle, et al., "A Smart Portable Troubleshooting Aid for Power Plant Controls," 1989 Conference on Power Plant Controls and Automation, February 1989.

8. J. W. Sheppard, et al., STAMP Technical Note 213, "STAMP User's Manual," ARINC Research Corporation, Annapolis, Maryland, January 1987, with Supplements 213A (June 1987) and 213B (November 1988).

9. J. W. Sheppard, STAMP Technical Note 332.1, "Multicriterion Optimization with Emphasis," ARINC Research Corporation, Annapolis, Maryland, April 1987.

10. W. R. Simpson, STAMP Technical Note 344.1, "Notes on Hypothesis Verification," ARINC Research Corporation, Annapolis, Maryland, July 1988.

11. W. R. Simpson, STAMP Technical Note 348.1, "Notes on Multiple Failure Groups," ARINC Research Corporation, Annapolis, Maryland, September 1988.

12. C. S. Dowling, et al., "WRAPLE: The Weighted Repair Assistance Program—Learning Extension," IEEE Design and Test of Computers, April 1986.