ADAPTIVE FAULT ISOLATION WITH LEARNING

W. R. Simpson J. R. Agre

ARINC Research Corporation
Annapolis, Maryland 21401

ABSTRACT

Fault-isolation strategies are often developed from estimated data, with application factors assumed or, sometimes, completely ignored. As a consequence, fielded systems exhibit failure pat-terns different from those predicted. Application and environmental data differ from location to location and from user to user; and fault-isolation strategies conceived during design are not always updated with field experience. A fault-isolation strategy that "learned" would adapt to field conditions and change on the basis of actual failures and the use of maintenance resources, permitting more efficient maintenance and reducing mean time to repair.

This paper describes an enhancement of the ARINC System Testability and Maintenance Program (STAMP) to provide fault-isolation strategies that change and conform to evolving failure data. The enhancement is designed to produce an on-line fault-isolation strategy that "learns" through repeated application.

INTRODUCTION

Testing of fielded systems can be characterized by excessive maintenance times, fault-isolation problems, and high false-alarm or "re-test OK" (RTOK) rates. As much as 50 percent of the time expended is attributable to fault-isolation difficulties. Troubleshooting remains a complex and poorly formulated process, despite advances in automatic test equipment 1,2,3,4

Testability is an emerging engineering discipline. The ARINC Research System Testability and Maintenance Program (STAMP) is a computer-based model that analyzes testability and provides fault-isolation strategies. STAMP techniques have been described in previous papers 5,6,7,8 and will not be described in detail here.

There are three major sections in this paper. The first is an overview of the topological dependency-analysis methods implemented in STAMP, including design for testability. The second describes the development of fault-isolation strategies, including the basis for information-theory strategies. It also addresses the weighting of information theoretic to obtain certain fault-isolation objectives. The weighting algorithms can be used to optimize test costs or test times, or to incorporate data on mean time between failures (MTBF). The third section describes the techniques needed to provide information feedback for "learning" from previous applications of the interactive system. Examples are employed to illustrate the techniques.

STAMP DEPENDENCY ANALYSIS

To describe the STAMP dependency analysis, we use a sample system. The functional block diagram of the system, shown in Figure 1, could represent a large system such as an aircraft, with subsystems such as navigation, fuel, and environmental control. It could also represent a single printed circuit board in a computer, with the boxes representing electronic components on the board. We use the term "component" to represent each block C1. . .,C9. The nodes T1. . .,T8 represent test points, and the arrows indicate functional flow.

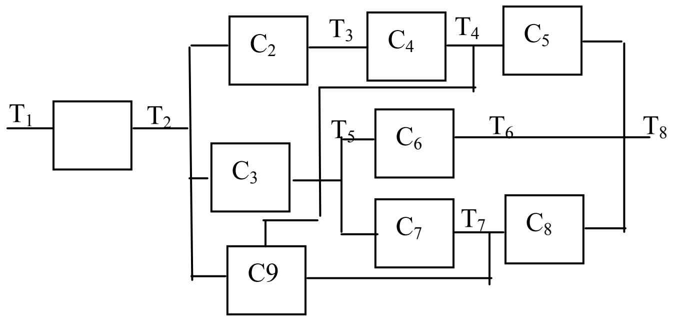


Figure 1. Sample System

The STAMP analysis requires input of the information on the functional block diagram. Given these inputs, STAMP employs a mathematical algorithm to identify all higher-order dependencies. A full range of testability measures is then produced through analysis of the higher-order dependencies.

A detailed description of the testability analysis, including actual output and redesign considerations for the sample system, has been provided by Simpson and Balaban 7.

Component ambiguities arise when the tests are inadequate to separate failures between components or groups of components, complicating not only testability but fault isolation. It is important to note, for purposes of this paper, that there are two sets of component ambiguities. For the sample system of Figure 1, the small feedback loop makes it impossible to distinguish failures in C3, C7, and C9, while the tests needed to distinguish between C5 and C8 are inadequate. In this paper these ambiguity groups are identified by placing an asterisk after C5 and two asterisks after C3.

FAULT-ISOLATION STRATEGIES

A fault-isolation strategy is a sequence of tests that permits identifying the failed component. An information-based, adaptive algorithm is employed in STAMP to compute efficient fault-isolation strategies that will minimize the number of tests required. (Details of the algorithm are presented in Reference B.) The algorithm examines the information content of all unknown tests. Since the test outcome is uncertain, the test is selected by performing an optimization procedure on the information content.

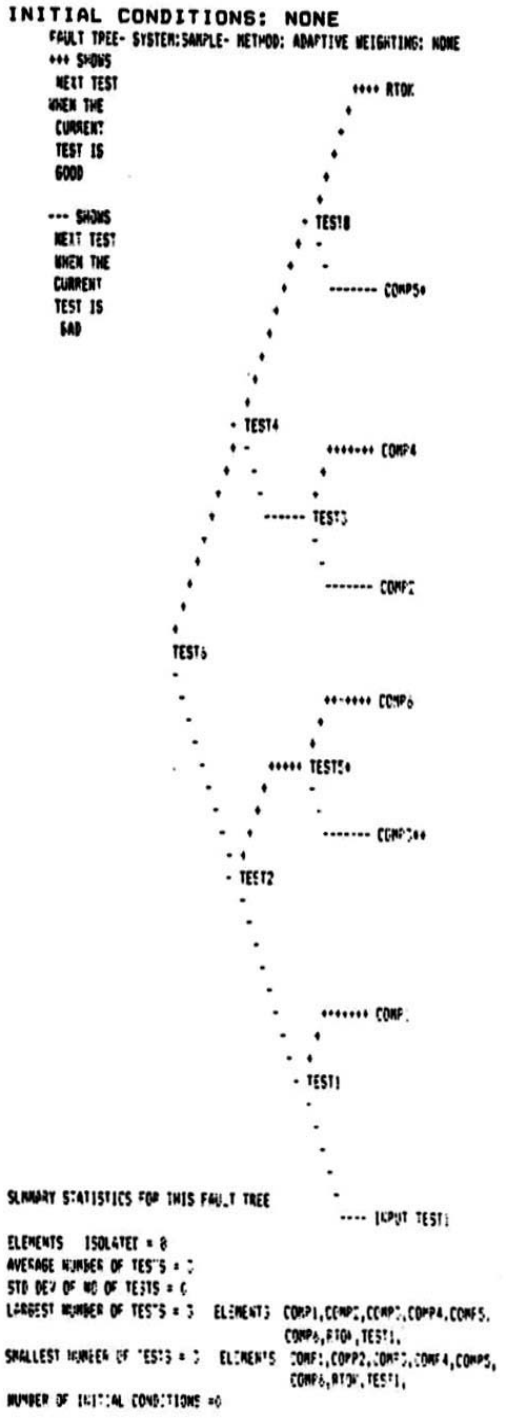
The fault tree for the sample system produced by the STAMP adaptive algorithm is presented in Figure 2. The first test to be performed is T6. If T6 is good, the "+" path is followed to the next test, T4. If T6 is bad, then T2 is performed. For example, the sequence T6 good, T4 bad, T3 good isolates the fault to component C4. The fault tree always uses three tests to isolate one of eight outcomes: components C1, C2, C4, and C6; ambiguity groups C3** and C5*; the input T1; and RTOK (retests OK).

Weighted Fault-Isolation Strategies

Underlying our discussion has been the assumption that failure rates, test costs, and test times are equally important. In many real systems this assumption is not valid. Some tests are complicated, lengthy, or costly; other tests are simple; certain components may fail more often than others. A fault-isolation strategy should be able to incorporate and employ that information. The STAMP approach is to convert such information into a factor that will assign each test a relative weight eased on the importance of the fault data it yields. STAMP has five weighting schemes:

(a) Test Cost - The value of the information is inversely proportional to the cost of the test. This is represented by wi, the weight applied to test 1, and is given by where c_i is the cost of test 1 and a is a normalization constant

$$w_1 = a/c_i$$



```
INITIAL CONDITIONS: NONE
    FAULT TREE- SYSTEM:SAMPLE- METHOD: ADAPTIVE WEIGHTING: NONE

SUMMARY STATISTICS FOR THIS FAULT TREE

ELEMENTS   ISOLATED = 8
AVERAGE NUMBER OF TESTS = 3
STD DEV OF NO OF TESTS = 0
LARGEST NUMBER OF TESTS = 3   ELEMENTS  COMP1,COMP2,COMP3,COMP4,COMP5,
                                        COMP6,RTOK,TEST1,
SMALLEST NUMBER OF TESTS = 3  ELEMENTS  COMP1,COMP2,COMP3,COMP4,COMP5,
                                        COMP6,RTOK,TEST1,
NUMBER OF INITIAL CONDITIONS =0
```

Figure 2. Fault Tree for Adaptive Strategy

(b) Test Time - The value of the information is inversely proportional to the time required to perform the test. The weight applied, w_i, is given by

$$w_i = \beta/\tau_i$$

where T_i is the time to perform test i and B is a normalization constant.

(c) MTBF - The value of the information from a test is directly proportional to the probability of test failure and inversely proportional to the number of components examined. The probability of test failure is given by

$$\rho_i = \frac{1}{\gamma} \sum_{j \epsilon O_i} \frac{1}{MTBF_j}$$

where 0_i is the set of all elements (components, RTOK, or inputs) on which test i depends and $1/MTBF_j$ is the failure rate or occurrence rate of the jth element.* The normalization constant is defined as:

$$\gamma = \sum_{j=1}^{N} \frac{1}{MTBF_j}$$

where N is the total number of elements (i.e., 9 components plus 1 input plus RTOK). In addition, let n_i be the number of elements that affect test i. Then the MTBF weighting for test i, w_i, is given by

$$w_i = \rho_i/n_i$$

(d) Minimum expected cost - The product of weights defined in a and c.

(e) Minimum expected time - The product of weights defined in b and c.

Each of these weightings can be used to develop fault trees with specific objectives. For example, we could use a or d to reduce the cost of fault-isolation, b or e to achieve minimum mean time to repair (MTTB), and c to achieve the least complicated test approach (fewest tests).

A specific performance measure associated with each of the weighting schemes is computed from the fault tree and used as the measure of effectiveness. Given a fault tree, we define the branch of the tree for outcome i, B_i, to be the set of tests in the fault path required to isolate outcome i. Let $[B_i]$ denote the number of tests in branch B_i.

*$MTBF_j$ for RTOK is an algorithmic convenience and is the inverse of the frequency of occurrence of RTOK. Every test is assumed to depend upon RTOK.

The measures of effectiveness are:

- Expected cost (uniform failures)

$$\bar{C} = \frac{1}{N} \sum_{i=1}^{N} \sum_{j \epsilon B_i} c_j$$

- Expected time (uniform failures)

$$\bar{T} = \frac{1}{N} \sum_{i=1}^{N} \sum_{j \epsilon B_i} \tau_j$$

- Expected fault-path length

$$\bar{L} = \sum_{i=1}^{N} |B_i| P_i$$

Where P_i is the probability of outcome i.

$$P_i = \frac{\frac{1}{MTBF_i}}{\sum_{j=1}^{N} \frac{1}{MTBF_j}}$$

- Expected cost

$$\hat{C} = \sum_{i=1}^{N} P_i \sum_{j \epsilon B_i} c_j$$

- Expected time

$$\hat{T} = \sum_{i=1}^{N} P_i \sum_{j \epsilon B_i} \tau_j$$

Weighting Example for the Sample System

The sample system (Figure 1) that is being fielded is described here to illustrate the application of STAMP weighting techniques. Information about the system's component failure rate will evolve through time and stages of development. Two stages for which failure-rate data are examined are (1) initial uniform weighting and (2) field data. For each stage, STAMP is used to generate weighted fault trees. The fault trees are directed at the smallest number of tests that can be used to isolate the more frequently failing components. The ability of the weighting algorithm to assimilate and adapt to changes in the failure data is shown.

At the first stage of development, no attempt is made to distinguish individual component MTBFs. For the 11 elements (9 components, 1 input, 1 RTOK), equal probability of outcome is assumed. However, because of the ambiguities between components 3, 7, and 9 (referred to as COMP3**) and between components 5 and B (referred to as COMPS*), the ambiguity groups have slightly higher probabilities of outcomes. The probabilities are given in Table 1.

Table I: Initial Data for Sample System

Outcome	Probability of Outcome
COMP1	.091
COMP2	.091
COMP3**	.273
COMP4	.091
COMP5*	.182
COMP6	.091
OUTPUT T1	.091
RTOK	.091

The high-failure-rate items are the two ambiguity groups. The fault tree resulting from the weighting algorithm is the same as the unweighted tree shown in Figure 2. According to the weighting algorithm, the uniform tree, with an expected fault-path length, L, of three, is the minimum tree. Examination of alternative fault trees shows an increase in the expected path length. The higher failure probabilities of the ambiguity groups do not compensate for the information value of the first test, Test 6, in the fault tree of Figure 2.

At some point in the system development, engineering estimates of the MTBF values might have been derived, possibly from a data base such as MIL-STD 217.9 Derived or estimated values could be used in lieu of the initial uniform failure-rate assumptions given.

At a later stage of development, field data on the MTBFs might have been collected. The maintenance technicians now have a feel for the system. They are guided somewhat by their experience to examine the most frequently failing items. A good fault tree should reflect this knowledge.

For the sample system, Table 2 shows the measured component MTBFs. In the absence of specific data, the input T1 and RTOK are assumed to have MTBFs equal to the maximum component MTBF. Table 3 lists the resulting probabilities of outcomes.

Table 2: Field MTBF Data

Element	MTBF	Element	MTBF
COMP1	45	COMP7	55
COMP2	55	COMPS	50
CO.MP3	10	COMP9	45
COMP4	50	OUTPUT	100
COMPS	65	RTOK	100
COMP6	100		

As shown in Table 3, failures in the ambiguity group of COMP3** are the source of more than 50 percent of the fault-isolation outcomes. The weighted fault tree for this case is illustrated in

Table 3: Probability of Fault-Isolation Outcome

Outcome	Probability of Outcome
COMP1	.083
COMP2	.068
COMP3**	.528
COMP4	.075
COMPS*	.133
COMP6	.038
INPUT T1	.038
RTOK	.038

Figure 3, in which the expected fault-path length, L, has been reduced to 2.64. As desired, COMP3** has been isolated in two tests, the minimum possible in this system. This, however, occurs at the expense of COMPS* and RTOK, which require four tests to isolate.

In larger systems with many more components and test points, the benefits of weighting are expected to be substantial. The next section describes a technique for automatically generating and incorporating the required data.

INTERACTIVE FAULT ISOLATION WITH LEARNING

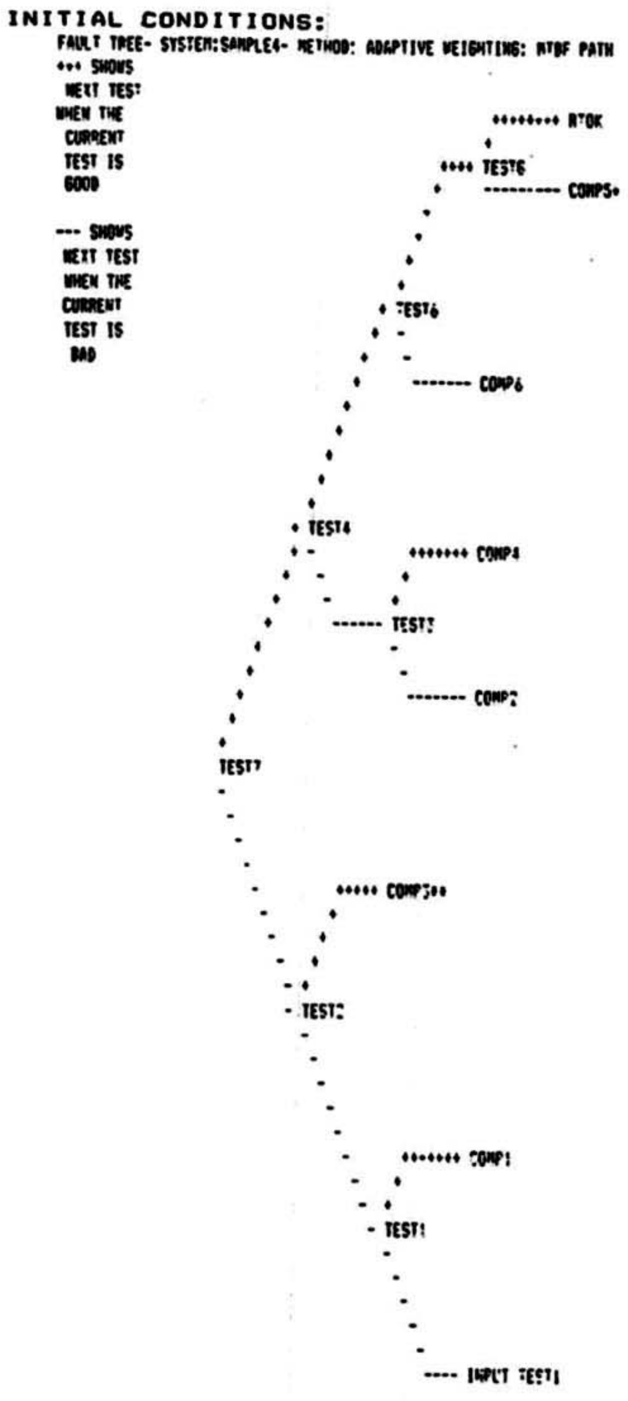
The STAMP has incorporated procedures that provide interactive assistance to maintenance personnel. The basic algorithms are resident in a microcomputer with a CRT display. In the interactive mode, initial conditions are entered into the system and the maintenance technician is "walked through" the fault tree. The primary advantage is that all potential fault trees resulting from any combination of initial conditions and unusable tests are available.8

The implementation of a learning fault tree requires only the updating of test times and test costs. When a fault has been isolated, the program will automatically update a data base with the new data acquired since the last maintenance action was performed for that failure. This update will, in turn, modify the fault tree for the next fault-isolation action.

For the sample system, then, the initial interactive isolation will follow the fault tree of Figure 2. At a later date, following many maintenance field actions, the interactive procedure will follow the fault tree of Figure 3. Still later, it may follow a different procedure depending on what has been learned from its environment.

SUMMARY

The STAMP capability for testability assessment and the development of fault-isolation strategies has been briefly described. The strategy can be influenced by a choice of weighting schemes, which include test time, test cost, component MTBF, and their combinations. The weighting schemes


INITIAL CONDITIONS:
FAULT TREE- SYSTEM:SAMPLE4- METHOD: ADAPTIVE WEIGHTING: MTBF PATH
+++ SHOWS NEXT TEST WHEN THE CURRENT TEST IS GOOD
--- SHOWS NEXT TEST WHEN THE CURRENT TEST IS BAD
RTOK
TEST6
COMP5+
TEST6
COMP6
TEST4
COMP4
TEST3
COMP2
TEST7
COMP3++
TEST2
COMP1
TEST1
INPUT TEST1

```
SUMMARY STATISTICS FOR THIS FAULT TREE

ELEMENTS   ISOLATED = 8
AVERAGE NUMBER OF TESTS = 3.125
STD DEV OF NO OF TESTS = .599478941
LARGEST NUMBER OF TESTS = 4   ELEMENTS  COMP5,RTOK,
SMALLEST NUMBER OF TESTS = 2   ELEMENTS  COMP3,
NUMBER OF INITIAL CONDITIONS =6
```

Figure 3. MTBF Weighted fault Tree

provide an effective method for deriving fault trees that reflect knowledge gained from field experience and should make maintenance personnel confident in using automated fault-isolation procedures.

The weighting algorithms have been used in conjunction with an interactive fault-isolation procedure. On-line data collection can be incorporated to permit STAMP to learn from the system's environment. A learning, interactive fault-isolation procedure has not yet been applied directly to a fielded system. However, the STAMP procedure offers the potential for reducing training requirements and improving maintenance procedures by providing a maintenance aid that employs concepts of artificial intelligence.

REFERENCES

1. George Smith II, "Testability Analysis, Predict It More Closely," <u>1979 Proceedings, Annual Reliability and Maintainability Symposium,</u> Washington, D.C., January 1979.

2. William L. Kiener and Anthony Coppola, "Joint Services Program in Design for Testability," <u>1981 Proceedings, Annual Reliability and Maintainability Symposium,</u> Philadelphia, Pennsylvania, January 1981.

3. Thomas N. Cook and John Ariano, "Analysis of Fault Isolation Criteria/Techniques," <u>1980 Proceedings, Annual Reliability and Maintainability Symposium,</u> San Francisco, California, January 1980.

4. Michael L. Labit, G. T. Harrison, and B. L. Retterer, Special Report on <u>Operational Suitability (OS) Verification Study Focus on Maintainability,</u> ARINC Research Publication 1751-01-2-2395, Annapolis, Maryland, February 1981.

5. William R. Simpson, "The Application of the Testability Discipline to Full System Analyses," <u>1983 IEEE Automatic Test Program Generation Workshop,</u> San Francisco, California, March 1983.

6. William R. Simpson, "Testability and Fault Diagnosis of Airline Avionics," <u>Special Edition, Plane Talk News,</u> March 1983.

7. William R. Simpson and Harold S. Balaban, "The ARINC Research System Testability and Maintenance Program (STAMP)," <u>1982 Proceedings of the IEEE AUTOTESTCON Conference,</u> Dayton, Ohio, October 1982.

8. H. Balaban and W. R. Simpson, "Testability/ Fault Isolation by Adaptive Strategy," <u>1983 Proceedings, Annual Reliability and Maintainability Symposium,</u> Orlando, Florida, January 1983.

9. MIL-HDBK-217, "Reliability Prediction of Electronic Equipment."

TESTABILITY AND FAULT DIAGNOSIS OF AIRLINE AVIONICS

Dr. William R. Simpson
ARINC Research Corporation
Advanced Research and Development Group

ABSTRACT

Several studies have shown that fault isolation or troubleshooting of aircraft avionic systems is a primary consumer of maintenance resources. The major determinants of the time consumed in fault isolation are the testability design, the efficiency of the fault isolation strategy, and the skill level and training of maintenance personnel. A dependency modeling approach to improving some of those factors has been developed by ARINC Research Corporation.

The ARINC Research System Testability and Maintenance Program (STAMP) is a computer-aided testability design and fault diagnosis system. It follows a basically functional approach to dependency analysis. From basic inputs of functional relationships, STAMP generates all higher-order dependencies and their implications. That permits testability assessment through automatic identification of component ambiguity groups, redundant or unnecessary test points, and feed-back loops. The model also provides several overall measures of testability. It generates automatic testability and fault isolation reports for use in an iterative application for testability design improvements. STAMP also develops fault isolation strategies that may be weighted for specific maintenance objectives such as minimum cost or time.

STAMP has been applied to systems at various levels of development, including those in preliminary design, prototypes, and fielded systems. STAMP has been used to improve testability design, write self-test ATE programs, and reduce mean time to repair. STAMP is flexible enough to be applied to the piece-part level, while general enough to be applied at the subsystem interface level.

INTRODUCTION

Maintaining avionics systems is becoming more difficult and costly as a result of increased complexity and sophistication. This is true despite advances in automatic test equipment.1,2 System testability design is usually approached from the bottom up with component and board testability designed in, while isolation to the individual unit in the full system is becoming more and more difficult. Current system and test design often results in high ambiguity levels for fault isolation and long test times. False alarm and re-test OK (RTOK) rates of 40 percent and higher are not uncommon in many avionic systems. Recent studies involving the F-16 aircraft3 and the CH-54 helicopter4 have shown that troubleshooting action can consume as much as 50 percent or more of the total man-hours spent for repair. Avionics Maintenance Conference reliability reporting statistics indicate similar trends in avionic repairs for the scheduled air carriers. Those figures suggest that there is a large potential return on an investment in improved test-ability assessment and fault isolation procedures.

The notion of testability is being recognized as a valid and viable engineering discipline. An equipment has good testability when existing faults can be confidently and efficiently identified. Confidence is achieved by frequently identifying only the failed components or parts with no removals of good items. Efficiency is achieved by limiting the resources required (including manpower, man-hours, test equipment, training, etc.). The number and information content of tests together with the location and accessibility of test points defines the testability potential of an equipment and is obviously a design-related characteristic. In addition, there must be a strategy (either implied or explicit) for using tests to verify and isolate faults.

This paper discusses a philosophy, a procedure, and a computer-based process, System Testability and Maintenance Program (STAMP), for assessing testability and conducting fault isolation strategies.

TESTABILITY AND FAULT ISOLATION BASICS

What is Testability?

This section will present a brief explanation of testability for a simple system. The simple system is serial in design (each element feeds the next element downstream with no feedbacks or parallel paths) and contains only tests that can provide a good-bad or pass-fail indication. This example precludes the use of such techniques as waveform or signature analyses that rely primarily on failure symptoms. Figure 1 shows the simplified system, consisting of five components, one input, and four tests. A test will be good if all elements that feed it are good, and bad if any element that feeds it is bad. For now, only a single failure in this system is considered.

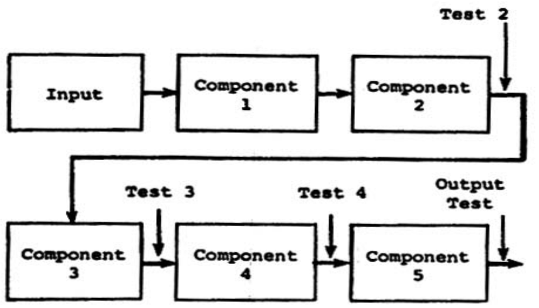


Figure I. SERIAL SYSTEM
(PARTIALLY TESTABLE)

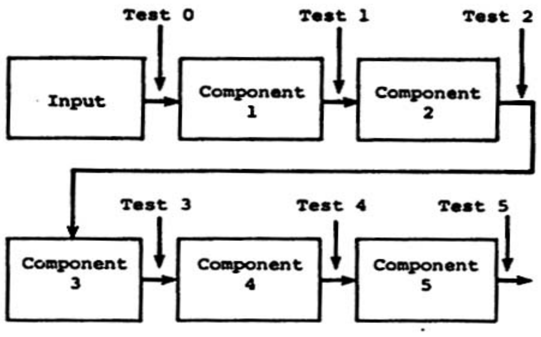


Figure 2. SERIAL SYSTEM
(FULLY
TESTABLE)

The system can be tested by initially looking at the output test. If the equipment passes this test, then it is functioning correctly and no failures are present. If the equipment fails this test then we know only that there is a failure. Any one of the elements (the input and five components) could be responsible for the bad value of the output test. For this system, one can with certainty isolate failures in Components 3, 4, and 5 by obtaining a good test value preceding the component and a bad test value after the component. This is not true for Components 1 and 2 and the input. A bad test value at Test 2 will only allow us to con-clude that one of the three preceding elements is the cause. Whenever tests allow only such a conclusion, the members of the group in doubt are called an ambiguity group. If the failure is in that group, a technician might replace all three elements. Since only one actually failed, a high RTOK rate (two out of three) will result at the component repair facility. An alternative might be to replace each of the three items one at a time until the system can pass the output test. This could be not only time consuming but may increase the chances of mishandling or breaking fragile elements or creating an improper system configuration. A much pre-ferred solution would be to provide tests that eliminate the ambiguities, as is shown in Figure 2, where the system has been given two additional tests.

In more complex systems with parallel paths and feedback loops, the solution is not so simple, but the goal is to eliminate or minimize the number of ambiguity groups.

Testability Summary

Testability is that property of a system that describes the number and type of faults that can be efficiently and confidently identified as well as those that cannot be efficiently and confidently identified. It is also the discipline by which one can analyze or improve the testability of a system.

How Are Isolation Strategies Developed?

An isolation strategy is a road map of how to use the available test points in determining what in the system (if anything) has failed.
A number of different strategies are available. The most common approach is to proceed along the functional flow. In the example of Figure 2, an evaluation of Test 5 is made. If it is "good," we are done and the result is a RTOK. If the test is "bad," we proceed to Test 4 (the next one in the functional flow). If that test is "good," we are done and we know that Component 5 is bad. If the test is "bad," we proceed to Test 3, and so on. This sequential or signal tracing approach is called a Directed Search. It is the method employed by most technicians in the absence of detailed procedures. For a series system such as our example, if all events* are equally likely, it will take an average of $(n-1)/2 + (n-1)/n$ tests to isolate one of the events. For the example, this is 3.86 tests with the least number of tests being 1 (RTOK) and the largest number of tests being 6 (Input and Component 1).

An alternative to this strategy would be to use system partitioning.** That procedure says that any given test, depending upon its results,

*An event is the failure of a component, the failure of an input, or RTOK. **A** total of n events is possible. For Figure 2, n = 7 (5 components, 1 input, and RTOK).

**Partitioning concepts are dealt with more formally in the section on STAMP fault isolation.

eliminates certain events from being the failure cause. Each test partitions the possible results into two states, feasible and not feasible. In the example, if Test 3 is "bad," under a single failure assumption, the failure could not have been caused by Component 4 or Component 5. Those two results go into the unfeasible category. RTOX is also unfeasible so it goes into the unfeasible category. If, on the other hand, Test 3 is "good," Input, Component 1, Component 2, and Component 3 go into the unfeasible category. Under these criteria, fault isolation 18 achieved when only one result remains feasible. We can expect to put the largest number of results in an unfeasible state, regardless of outcome, if we choose a test near the middle of the system. This would either be Test 2 or Test 3 for our example, depending upon how one rounds off to get the middle or half-way point. This method of partitioning to the middle or half-way point is termed Half-Interval.

A fault isolation strategy can be conveniently represented by a tree. Fault isolation trees determined by both the Directed and Half-Interval methods are shown in Figure 3. The procedure in either search case is to start with the first listed test and, depending on the result, take one of the branching paths. This proceeds until an event (circled) is identified.

The average number of tests required by the Half-Interval technique is 2.86, or one less than the Directed. Additionally, the Half-Interval requires no more than three tests while the Directed may require as many as six. Partitioning is an extremely powerful technique for producing fault isolation strategies. In fact, if one were able to choose the test point that evenly partitioned the feasible and unfeasible events at each point, it can be shown[5] that the number of tests required is log2n, or 2.807 for the example of Figure 2. The Half-Interval is thus shown to have much

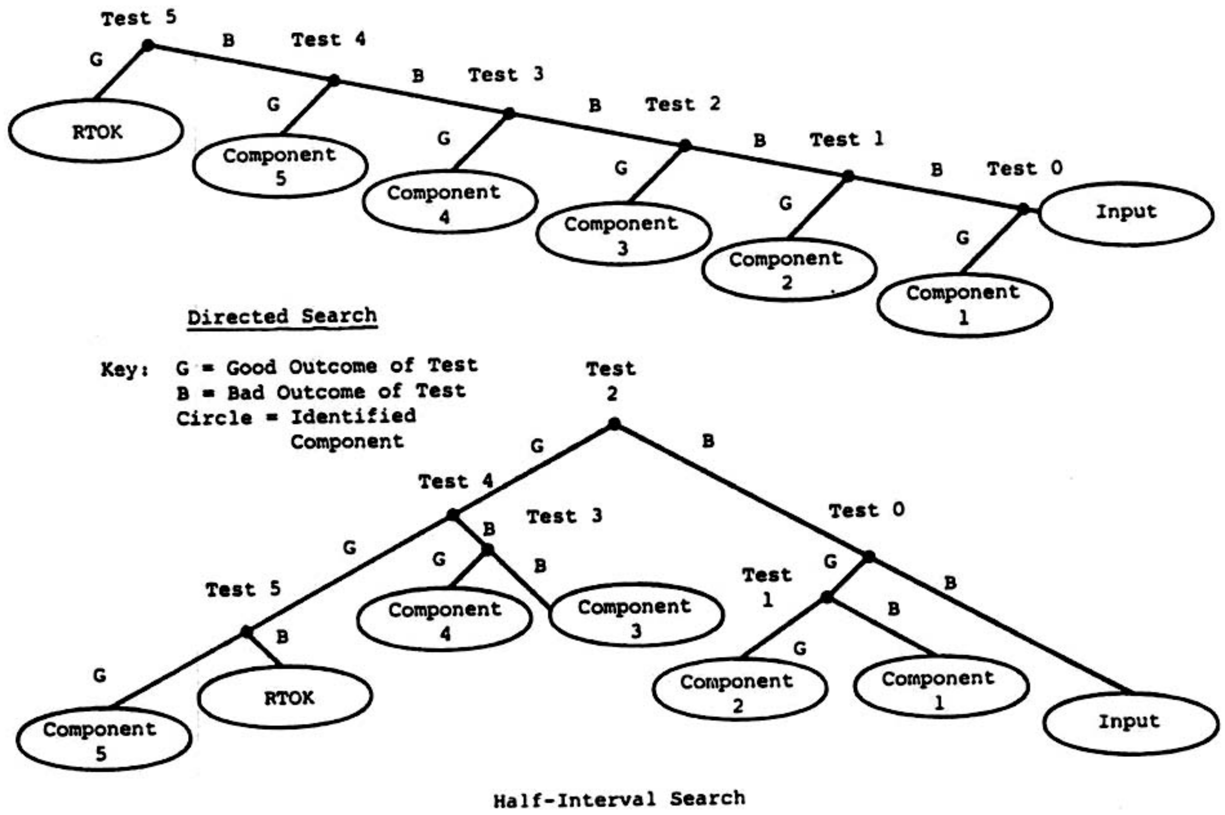


Figure 3. FAULT ISOLATION TREES FOR SYSTEM OF FIGURE 2

better theoretical characteristics than the Directed, at least for this serial system. The Half-Interval technique, however, suffers from a very serious flaw; namely, for complex equipments it is difficult, if not impossible, to reasonably define half-way. The problem arises in systems with multiple inputs and outputs, parallel paths, crossovers, or feedback loops. We say that such designs are not well ordered. Use of the Half-Interval technique requires at least partially ordered systems.*

There are other search approaches (see Table 1) that include some characteristics of the two basic approaches.

These few paragraphs show the basic issues of testability and fault isolation that STAMP addresses. One is to perform a testability analysis to identify weaknesses, such as component ambiguities, and allow for the formulation of corrective actions to improve the testability design. The other is to develop fault isolation strategies that make maximum use of whatever testability the system has.

Table 1. SEARCH STRATEGIES

Type of Search	Ordering Required	Strategy
Directed	Yes	Test all test points sequentially from right to left in HOC, beginning with first known fault, except skip test points where result can be inferred from previous tests.
Half-Interval	Yes	Test at mid-point of remaining test points. Go right to left after a good test and left to right for a bad test. Skip tests whose results can be inferred from previous tests.
Half-Interval, Directed Combination	Yes	Test at mid-point of remaining test points until a bad test is encountered. Use directed search from left to right on tests upon which bad test depends, except skip tests whose results can be inferred from previous tests.
Exponential, Directed Combination	Yes	Select test closest to 63 percent partition from left to right of remaining test points. Continue until a bad test is encountered, then use directed search from right to left.
Exponential	Yes	Same as above, except continue testing with 63 percent partition.
Adaptive or Information Theoretic'	No	For each test point, ask how much information can be inferred from either good or bad tests. Select test points to optimize answer.
Random	No	Use uniformly distri-buted random numbers to choose test points.

This technique is dealt within greater detail in STAMP Fault Isolation, page 9.

Overview of STAMP Application[6]

Figure 4 presents an overview of the STAMP application. A brief summary of the steps is given below with details provided in the following sections:

- System Topology. The functional block diagram of the system, including test points, forms the basis for developing the inputs to STAMP.
- STAMP First-Order Dependencies. For each test point, all first-order (immediate) predecessor test points are entered with embedded components.
- Testability Assessment. Basic test-ability characteristics inherent in the design are analyzed and design improvement areas are identified. The step includes generation of a testability report.
- Redesign or Fault Isolation. A decision by project personnel is made to redesign the system to improve its testability or to develop a fault isolation strategy.
- Testability Redesign. The information and recommendations provided by the STAMP testability assessment is acted upon by the engineering department to improve testability characteristics.

.

- Fault Isolation Analysis. On the basis of dependency analysis and weighting factors, an analysis is performed to develop efficient fault isolation strategies.
- Fault Isolation Strategy. The detailed step-by-step sequence of test procedures

*The distinction between partially ordered and well ordered is one of uniqueness. A partially ordered set may have several permutations by which ordering is satisfied. A well ordered system has only one ordering

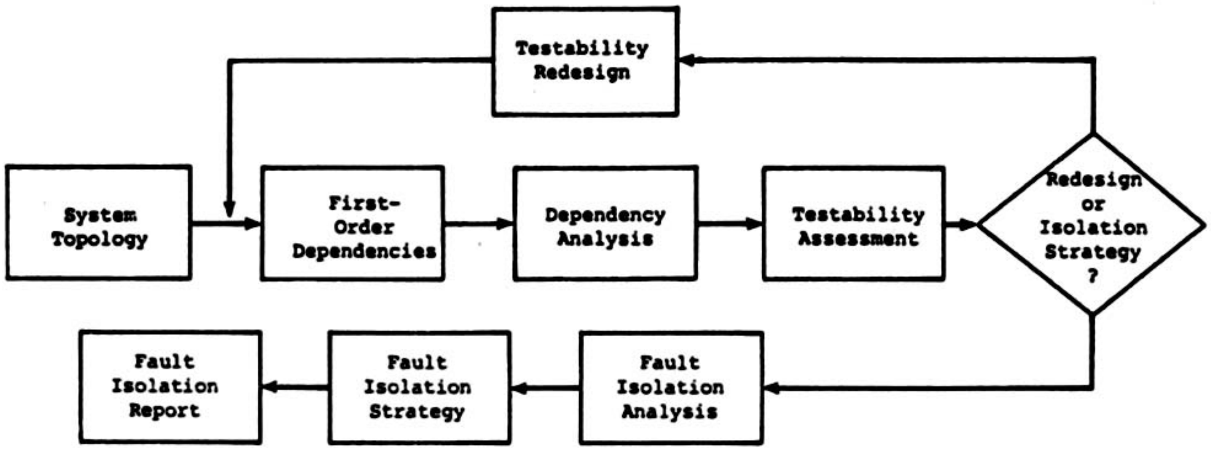


Figure 4. STAMP APPLICATION OVERVIEW

to follow to isolate faults is determined. A handbook containing the fault isolation strategy in the form of fault trees with alternatives on initial conditions or an interactive computer-based procedure can guide the maintenance technician.

- Fault Isolation Report. A summary report is automatically generated, describing the testability and fault isolation characteristics of the system, including multiple failure strategies.

STAMP TESTABILITY ANALYSIS

System Topology and First-Order Dependency Input6

Figure 5 is a functional block diagram of a sample system that we will use in this paper for illustrative purposes. This diagram might represent a full aircraft avionics system, and the blocks could represent such subsystems as navigation, weather, fuel, and environmental control. The diagram could also represent such a low-level item as a PC board within an air data computer, with electronic components being represented by the blocks. For our purposes, we will use the term "components" to represent the individual blocks, C1, C2, .. C9. The nodes represented by T1 , T2, .. , T8 are test points and the signal or dependency flow is indicated by the arrows.

STAMP will consider two types of tests:

- Functional Tests: Tests that indicate the correct functioning of all system elements that "feed" the test point
- Special Tests: Tests that have dependencies that are not readily described by a standard functional diagram.

For the example of Figure 5 each test is considered to be functional test. In general, a functional test can be "placed" on a flow diagram, while a special test cannot.

Also shown in the figure are examples of the input for functional tests. For Test Point T2, the immediate predecessor test point is T1, and Component C1 is embedded. Similarly, Test Point T6 has one immediate predecessor test point, T5, and one embedded component, C6. Test Point T4 is fed by two branches: one branch has T3 as an immediate predecessor with C4 embedded, and the other branch is fed by T5 with no embedded components.

No special tests are shown for the sample system, but they may be included by listing their individual dependency chains.

In addition to these inputs, the user also must enter any weighting information that may be required. STAMP can develop a fault isolation strategy that considers failure rate, test costs, and test times, and such factors must be provided if they are to be considered.

Dependency and Testability Analysis[6]

Given these inputs, STAMP employs a mathematical algorithm to obtain all higher-order dependencies through a manipulation of a matrix representation of the functional and special test dependency relationships. A full range of testability measures is then generated through analysis of the higher-order dependency matrix. Table 2 lists the measures, gives a brief definition of them, and shows the values obtained for the sample system.

The measures are normalized values ranging from 0 to 1. Where applicable, the testability characteristic of a comparably sized full-serial system is used as a basis for

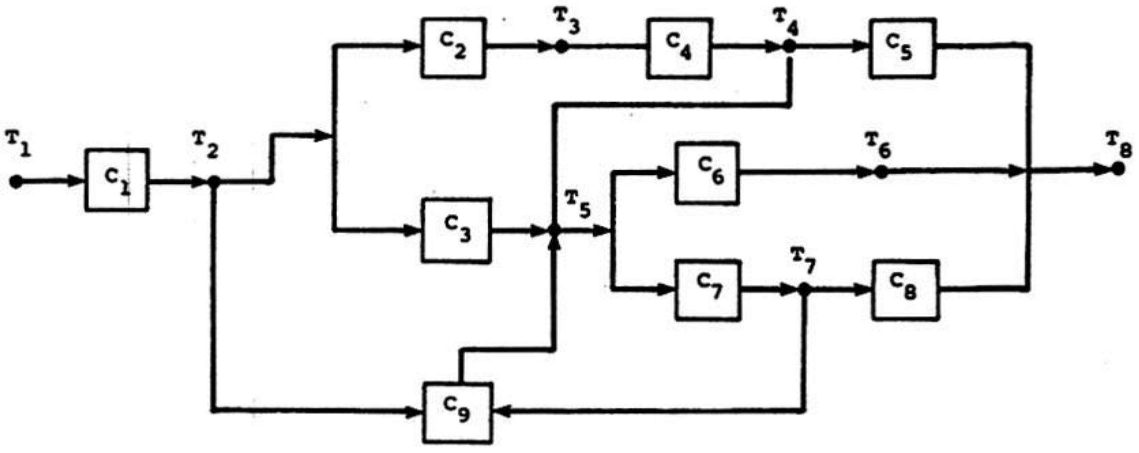


Figure 5. SAMPLE SYSTEM

Table 2. SYSTEM TESTABILITY MEASURES

Measure	Definition	Sample System Value
Dependency	A measure of test point connectivity	0.56
Component Leverage	Percentage of components uniquely fault isolatable (observability)	0.67
Modified Component Leverage	Percentage of component groups uniquely fault isolatable	0.86
External Dependency Factor	A measure of dependency on external factors	0.11
Test Point Redundancy	The percentage of test points which contain unique dependencies	0.86
Test Point Leverage	A relative measure of the degree to which the test point set meets theoretical fault isolation limits (controllability)	0.67
Modified Test Point Leverage	Test point leverage for a repackaged system	0.86

normalization or analysis. For example, consider a system consisting of eight components. If all components are directly in series, a Half-Interval partition can be used to show that three tests is the minimum necessary for full fault isolation.* If each component required a separate test, a maximum of eight tests would be required, irrespective of the design. The limits of three and eight then form the basis for developing the test point leverage measure. The normalized

*See the section How Are Fault Isolation Strategies Developed? on page 2.

measures allow comparison across competing designs and also provide measures of progress as design iterations take place.

STAMP automatically generates a testability report providing six major outputs:

- Testability Measures and Discussion
- Component Ambiguity Groups
- Test Point Redundancies and Excess Test Points
- Feedback Loops

• Signature Analysis for Hidden Failures

• Recommendations for Testability Improvement

Table 2 provides a list of measures and the values for the sample system. Examples of paragraphs discussing these measures developed automatically by STAMP are provided in the following paragraphs.

Dependency Measure (DEP = 0.555)

The moderate value of dependency indicates several gaps in dependency. Specialized or adaptive techniques could sharply improve fault isolation. Adaptive techniques may yield the best fault isolation strategies, approaching the theoretical limits of between three and four required tests.*

Component Leverage (CL = 0.666 MCL - 0.857)

Several component ambiguity groups exist, as is shown in Table 3. The table lists those components whose faults cannot be individually isolated with the current set of test points. Those components with an asterisk are tied up in one or more feedback loops. They should be packaged together to reduce the number of good components removed. With such packaging the good component removal rate due to component ambiguity is 0.142; without such pack-aging the rate is 0.333. That rate could be reduced further by adding test points to separate those components not tied up in feedback loops or by repackaging them.

Table 3. COMPONENT AMBIGUITY GROUPS

Group Number	Cross Reference*	Component Number
1	T_7	C3,** C7,** C9*,
2	--	C5, C8

*Refers to a specific feedback loop in Table 4 or an element of that feedback loop.
**Indicates part of feedback loop.

Table 3 illustrates the identification of the component ambiguity groups and feedback loop information. Table 4 shows the information provided through the analysis of test point redundancies and excess test points. A redundant test point is one for which another test point provides identical information. An excess test point is one whose information content is not necessary for fault isolation. As an example, Test Point T2 is excess for this design because combinations of other tests (e.g., T3, T4, and T7) can be used to provide the same information. This type of conclusion is not easily reached by inspection even for such a simple design as is shown in our illustration.

*See The STAMP Approach, page 9.

Table 4. TEST POINT REDUNDANCIES

Group Number	Cross Reference*	Redundancy Group
1	C_3	T_5,** T7**

Test point analysis indicates one or more of the following test points are not needed:

Test 2, Test 5

*Refers to a specific feedback loop in Table 3 or an element of that feedback loop.
**Indicates part of feedback loop.

Signature Analysis for the Multiple Failure Case[6]

Another analysis performed by STAMP is identification of potential hidden failures and false component failure indications. This is done through analysis of the component failure signatures. A component failure signature as used here can be mathematically defined as a vector

$$\underline{K}_i = (K_{i1}, K_{i2}, \ldots, K_{in})$$

where Kij is equal to 1 if the jth test would fail, given Ci has failed, and Kij = 0 other-wise. Component Ci is said to dominate Ck if Kij < Kkj for j = 1, 2, ..., n. Any reasonable fault isolation procedure will isolate to the dominant component first, which means that if multiple failures were possible, the failure of any dominated components would be hidden. We also have the possibility that failure of two or more components may lead to a false indication that another component has failed. This would occur when there exists a subset of components, say S, such that

$$\bigcup_{j \in S} \left[\underline{K}_j\right] = \underline{K}_i$$

That is, the failure signature of a group of components "adds up" to the signature of another component.

STAMP identifies potential hidden and false failures as shown in Table 5 for the sample

STAMP Fault Isolation[7]

General Discussion[7]

Fault isolation can be mathematically described as a partition process. This process was described generally in the earlier section on How Are Isolation Strategies Developed? This section will deal more formally with partitioning processes. Let C = (C_l, C_2,..., C_n) represent the set of components. After the jth test, a fault isolation strategy partitions C into two classes defined as follows:

$\underline{F}^j = (c_1^j, c_2^j, \ldots c_n^j)$ = the set of components that are still failure candidates after the j test (feasible set)

$\underline{G}^j = \underline{C} - \underline{F}^j$ = the set of components found to be good after the j^{th} test (unfeasible set)

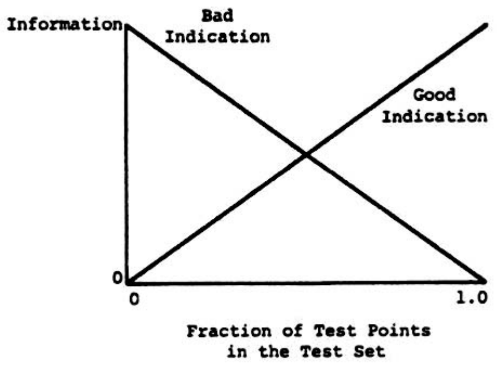
By this structure, a strategy will have isolated to the failure when $\underline{F}^j$ consists of a single element or a component ambiguity group. From our earlier discussion of series systems, we can see that the Directed Search strategy may only reduce the size of $\underline{F}^3$ by one component at a time. In the Half-Interval technique, $\underline{F}^j$ is reduced in half (approximately) after each test, an obvious advantage.

It can be proved that for a completely ordered system the Half-Interval technique will provide the minimum number of tests. However, as was pointed out earlier, such an ordering rarely exists. The approach used by STAMP employs an adaptive, information-based strategy.

Information Concepts[7]

Results of a test impart information. The type, amount, and quality of such information should be a consideration in developing a fault isolation strategy. For our purposes, we assume equal quality in the sense that the good-bad indication of a test actually reflects the state of the unit under test. (This assumption is quite significant; it suggests that STAMP may have limited usefulness in the analysis of degrading systems.)

The amount of information, however, is quite variable. Referring back to Figure 2, if the first test reading was at Test 5 and it was bad, the only inference we can make is that one or more of the six elements are bad and that RTOK is not possible. On the other hand, a bad reading of Test 0 tells us specifically that the input is bad. However, it is obvious that we cannot conclude that Test 0 is a better test to start with. If we take the case of good readings, Test 5 good tells us that all elements are good (RTOK) while Test 0 good tells us only that the input is good. This type of information distribution leads to the basic premise that a good test at the end of the functional flow is information rich, as is a bad test early in the information flow. If we can hypothesize a linear variation in information content, we have a relationship similar to that shown in Figure 6.



Fraction of Test Points
in the Test Set

Figure 6. LINEAR INFORMATION DISTRIBUTION

The fault isolation process involves an uncertain outcome and, therefore, a reasonable strategy is to balance the information content; that is, choose a strategy that gives equal information regardless of the test out-come. Indeed, that is precisely the basis for the theoretically optimum Half-Interval test as discussed in an earlier section. Unfortunately for complex designs, a linear information assumption may not be appropriate.

The STAMP Approach[7]

In seeking to overcome the limitations of the Half-Interval technique, it became apparent that if all dependencies in a system were known, then the information content of each test could be calculated. If a test is per-formed, the set of dependencies allows us to draw conclusions about a subset of components. The process of drawing conclusions about the system from limited information is called inference. For any test sequence the STAMP process allows us to compute (j1, j2 . . . , jk), the set of remaining failure candidates; namely $\underline{F}^{j1}$, $\underline{F}^{j2}$,. . . , $\underline{F}^{jk}$. A complicated algorithm has been developed to look at the information' content of all remaining tests so that the number of remaining tests that have to be performed to isolate faults is minimized over the set of potential failure candidates. This adaptive approach embodies several

Table 5. EQUIVALENCE

Number	Failure Indication	Dominated Components	Retest with the Following Tests as Initially Given Good
1	T_i (Input)	C_1, C_2, C_3, C_4, C_5, C_6	T_i
2	C_1	C2, C_3, C_4, C_5, C_6	T_1
3	C_2	C_4, C_5	T_6
4	C_3	C_4, C_5, C_6	T_3
5	C_4	C_5	T_6, T_3
6	C_6	C_5	T_4

system. In this case there are no false indication possibilities (an asterisk would be used to identify them), but failures in six of the components and the input can hide other failures. The recommended procedure is to determine for which cases it is likely that multiple failures will occur because of physical or environmental dependence. For such cases, it may be desirable to replace both compo-nents; e.g., if the power supply is failed, always replace the simple resistor upon which it depends, or else it may be worthwhile to employ a special test to determine if there is a hidden failure of the resistor.

The table also indicates how fault isolation should proceed given that the indicated failure has been corrected. To illustrate some of these points, consider Line 4 of Table 5. It shows that a failure of Component 3 could hide a failure of Component 4. Assume that engineering analysis shows that Components 3 and 4 are physically dependent in such a manner that failure of Component 4 is likely to cause a failure of Component 3. Therefore, if through fault isolation C3 is identified as having failed and is replaced, it will likely fail again if C4 is a hidden failure. Devising a special test to identify multiple failures may be beneficial if the occurrence probabilities are significant and always replacing both components will be too costly.

Testability Improvement Through Redesign[6]

The following discussion is based on hypothetical characteristics of the sample system; the characteristics were chosen to illustrate testability redesign points. These hypothetical characteristics may or may not be present in real systems.

Referring back to Table 3, we see that Components 3, 7, and 9 are in a feedback loop and that Components 5 and 8 form an ambiguity group. Assume that it is decided to insert a gate that is opened for test purposes to distinguish between Components C7 and C9. That will reduce the rate of removal of good components from 0.33 to 0.22. That change will still leave two ambiguity groups of two (C3, C9 and C5, C8), which may not be satisfactory. Assume that repackaging of C3 and C9 into one replaceable unit is satisfactory, leaving only the C5 and C8 ambiguity group.

Using our multiple failure example, assume it is decided to eliminate the possibility of a C3 failure hiding a C4 failure. To handle this as well as the remaining ambiguity group, a special test may be developed that distinguishes between components 4 and 5.

When these changes are made, the STAMP test-ability analysis will show that all component groups are now fully testable and that a component 4 failure is not hidden by a Component 3 failure. However, STAMP shows that the modified system now is over-specified in that one or more of Test Points T2, T3, and T5 are not needed.

The design engineer should be asked to identify the best candidate for elimination. Assume it is Test T2 because, for instance, it requires expensive access hardware or is time consuming to perform. Rerunning STAMP without T2 will show that Test T5 is still not needed so that it can be eliminated as well.

Redesign Summary[6]

Specific redesign actions for the sample system taken as a result of the STAMP analysis are listed below:

- Insertion of a gate to open the feed-back loop
- Repackaging of two components
- Addition of a special test
- Deletion of two functional tests

These actions caused the following improvements in testability:

- Component isolation was increased from 6 to 8.
- Component package isolation was increased from 66.6 to 100 percent (component leverage = 1.0).
- An undesirable multiple failure depen--dency situation was eliminated.
- Testability complexity was reduced in that the modified system had one less test than the original system (test point leverage increased to 0.75).

artificial intelligence algorithms, including inference and pattern recognition. We can describe it mathematically as follows:

Let D represent the full dependency relationship between components and teat points. This is formulated as a matrix representation.

Let S_k be a sequence of k tests, $(T_{j1}, T_{j2}, \ldots, T_{jk})$.

Let $\underline{F}^k$ be the feasible failure candidate set associated with S_k.

We then develop an information measure for each remaining (unperformed) test (j), which is a function of the dependency relationship and the remaining candidate failure class, say, $I_k^j = f[D, \underline{F}^k]$. The test sequence $\underline{S_k}$ that is derived is obtained by optimizing at each decision point. That is, the next test in the sequence is taken as the test that maximizes I_k^j for the conditions imposed by each previous test outcome and based upon an unknown current outcome. The sequence terminates when adequate information is derived for fault isolation.

Weighted Strategies[7]

To this point, we have considered only the test point location and functional or signal flow in discussing isolation strategies. Underlying this discussion was the assumption that all failures are equally likely and that all tests require equal resources. In practice, such an assumption may not be acceptable. Ideally, a fault isolation strategy should give more weight to tests that can determine the status of components most likely to fail and to tests which are simple to per-form or easily accessible.

The approach upon which STAMP is based allows this type of information to be easily incorporated into the information measure. STAMP allows for data on component failure rates, test time, and costs to be directly incorporated into the search strategy algorithm. The logistics or maintenance manager can then select a strategy that minimizes test resource use through selection of one or more of the weighting factors. The STAMP information-based strategy is the only strategy that allows this form of weighted information.

Consistency Checks

Under certain conditions one may wish to include in the fault tree checks that must be met before a radical action is taken. Often in the fault isolation of a complex system, a high degree of inference is made to draw conclusions, or one is unsure of the quality of test equipment or personnel used.

It may be advisable to verify apparent faults through additional tests before under-taking complex repairs. This can be accomplished in STAMP by overriding some of the inference that is possible. That is generally done by adding tests to the more difficult failure paths and checking the consistency of results. The consistency checks may be requested with any of the fault isolation strategies and with or without weighting.

Application[7]

The fault isolation procedure provided by STAMP incorporates the adaptive information-theoretic approach described above. Two modes of fault isolation are possible.

Interactive - In this mode, the fault isolation strategy resides in a micro-computer with video display. The technician is directed to provide initial known information such as good and bad test values and untestable test points (e.g., a certain test equipment might not be available or a test point may not be accessible). The computer then directs the technician to perform a particular test and explains how to interpret the results, which are than fed back to the program. A procedures file can be incorporated that describes the test, or else refers to the applicable section of a technical order or maintenance manual. When sufficient information has been obtained, the identity of the failed component is revealed and, if desired, detailed repair instructions can be displayed. Failure rate, test cost, and test time weighting can be incorporated, as well as consistency checks.

Fault Isolation Trees - This application provides a set of fault isolation trees that present the step-by-step procedures for locating failures. One tree is required for each set of initial conditions to be considered, leading to a manual or handbook of strategies to follow, depending upon which initial condition is appropriate. The trees are in the form of graphs as well as tabular step-by-step instructions.

Program Implementation

STAMP has been implemented on a microcomputer with 64K memory that can handle up to 170 test points and 248 combined test points and components. A complete analysis including testability assessment, automatic report generation, and fault tree generation for systems with about 200 test points and components takes approximately two hours. For minicomputer and mainframe application, we can expect increases in the number of combined test points and components of one to two orders of magnitude.

For the microcomputer application, a user-friendly menu-driven approach was adopted. A special editor program has been written so that design changes being considered can be easily entered for analysis or fault tree generation.

STAMP Effectiveness[7]

Because of the complexities of the STAMP algorithm, we have not been able to prove optimality characteristics of STAMP except for one special case. It can be shown that for well ordered or straightforward series design, STAMP reduces to the Half-Interval technique, which is known to be optimal for that case. In a number of applications, the adaptive, information-theoretic approach has provided mean and variance of the required number of tests under all failure conditions either equal to or lower than those resulting from other procedures examined.

This is illustrated in Table 6, which compares the fault isolation performance for a number of procedures for a partially ordered system consisting of 17 components and 16 test points.* For this 17-component system, the theoretical minimum number of tests of 3.91 is approached by three of the methods considered: exponential, exponential-directed, and adaptive, with the last providing the lowest mean. The adaptive method had a variance considerably lower than the other alternatives. That quality offers advantages in planning manpower, facility, and equipment resources for maintenance.

Table 6. COMPARISON OF SEARCH STRATEGIES FOR A SYSTEM WITH 18 COMPONENTS AND 17 TEST POINTS

Search Strategy	Average Number of Tests Required	Test Variance
Directed	6.86	5.26
Half-Interval, Directed Combination	5.14	1.26
Exponential, Directed Combination	4.50	1.66
Exponential	4.43	1.24
Adaptive	4.35	0.23
Random	5.71	4.49
Theoretical Limit	3.91	--

Analysis of the Sample System[7]

The sample system of Figure 5 will be analyzed. None of the testability improvements proposed in the previous section are assumed incorporated, although they could have been. The fault tree for the sample system employing the adaptive strategy, shown in Figure 7, shows that the first test to be performed is T_6. To illustrate the sequencing, if a good result is obtained from T_6, then T_4 is per-formed. A bad T_4 result would then be followed by T_3, which isolates between C_2 and C_4. Those entries with an asterisk indicate a tie-up with the feedback loop or the ambiguity group. For this system we can isolate six components and the input. We also have a sequence that leads to a conclusion of no defect (RTOK). Components C_7 and C_9 are not shown since they are tied up with C_3 in the feedback loop. Similarly, C_8 is in an ambiguity group with C_5 and, therefore, it also is not shown.

This particular fault isolation tree is uniform in that each decision requires three tests so that the mean number of tests is three and the variance is zero. By comparison, if the Half-Interval Directed approach (see Figure 8) is used, the average number of tests is 3.38 with a variance of 1.1, and the Directed approach (see Figure 9) requires an average of 3.6 tests with a variance of 1.2.

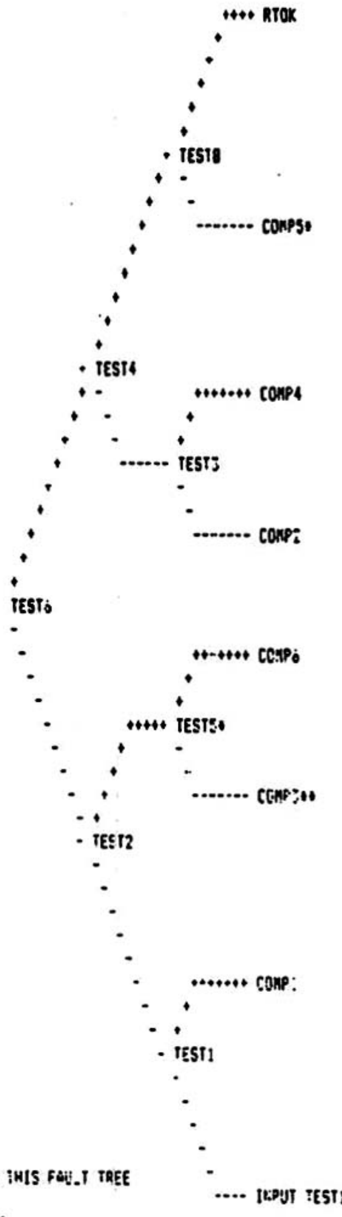
The fault tree generated by the adaptive strategy (but with consistency checks) is given in Figure 10. It can be noted that consistency tests are added in the isolation of C0MP6, COMP1, and the INPUT. Each of the other failure paths is well enough defined so that consistency checks are not required. The additional checks for consistency are the only differences between Figure 7 and Figure 10. Further, in the summary statistics we have increased the average number of tests to 3.375 by adding consistency checks.

As stated earlier, the adaptive strategy also permits known information to be used. Let us assume that it is known that T_4 is good. The fault tree for that case is shown in Figure 11. It is, of course, less complex because of the prior information, requiring an average of 1.7 tests. Only components C_5, C_6, and C_8 could be failed if T_4 is given good. C_8 is tied up with C_5 so that only C_6, C_5, and RTOK are shown in the fault tree. The initial

*Example system derived from Cramer, et al.[5] No weighting of failure rate, costs, or other factors was considered.

```
INITIAL CONDITIONS: NONE
     FAULT TREE- SYSTEM:SAMPLE- METHOD: ADAPTIVE WEIGHTING: NONE
     +++ SHOWS
      NEXT TEST
     WHEN THE
      CURRENT
      TEST IS
      GOOD

     --- SHOWS
      NEXT TEST
      WHEN THE
      CURRENT
      TEST IS
       BAD
```



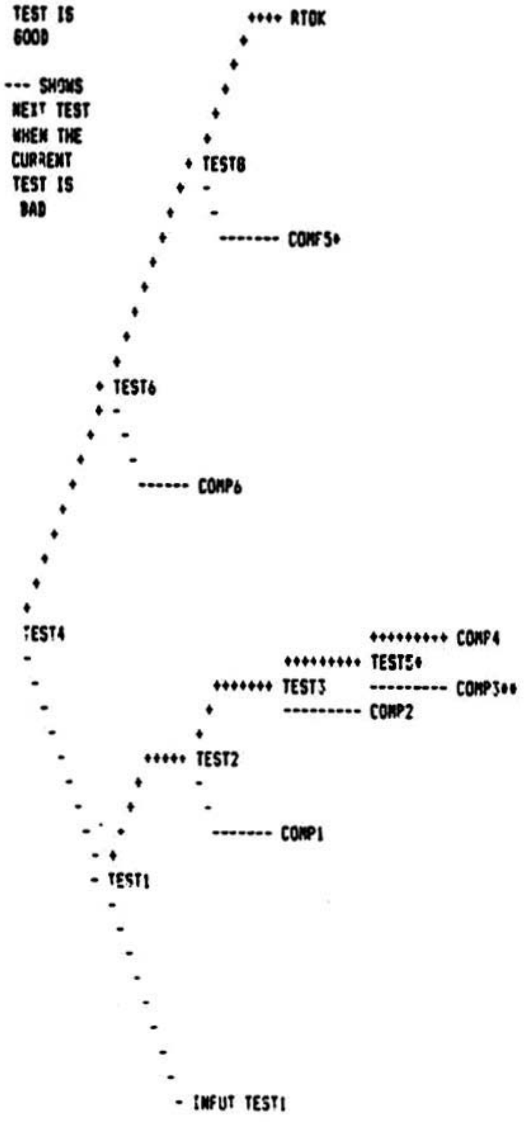
```
SUMMARY STATISTICS FOR THIS FAULT TREE

ELEMENTS   ISOLATED = 8
AVERAGE NUMBER OF TESTS = 3
STD DEV OF NO OF TESTS = 0
LARGEST NUMBER OF TESTS = 3   ELEMENTS  COMP1,COMP2,COMP3,COMP4,COMP5.
                                        COMP6,RTOK,TEST1,
SMALLEST NUMBER OF TESTS = 3   ELEMENTS  COMP1,COMP2,COMP3,COMP4,COMP5,
                                         COMP6,RTOK,TEST1,
NUMBER OF INITIAL CONDITIONS =0
```

Figure 7. FAULT TREE FOR
ADAPTIVE STRATEGY

```
INITIAL CONDITIONS: NONE
     FAULT TREE- SYSTEM:SAMPLE- METHOD: HALF-INTERVAL, DIRECTED
     +++ SHOWS
      NEXT TEST
     WHEN THE
      CURRENT
      TEST IS
      GOOD

     --- SHOWS
      NEXT TEST
      WHEN THE
      CURRENT
      TEST IS
       BAD
```



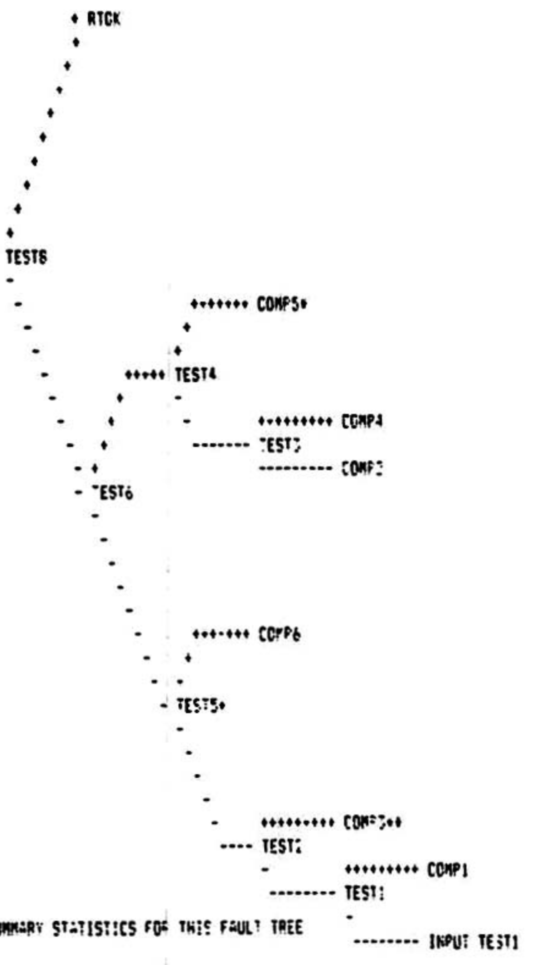
```
SUMMARY STATISTICS FOR THIS FAULT TREE

ELEMENTS   ISOLATED = 8
AVERAGE NUMBER OF TESTS = 3.375
STD DEV OF NO OF TESTS = 1.1110243
LARGEST NUMBER OF TESTS = 5   ELEMENTS  COMP3,COMP4,
SMALLEST NUMBER OF TESTS = 2   ELEMENTS  COMP6,TEST1,
NUMBER OF INITIAL CONDITIONS =0
```

Figure 8. FAULT TREE FOR HALF-INTERVAL
DIRECTED STRATEGY

```
INITIAL CONDITIONS: NONE
     FAULT TREE- SYSTEM:SAMPLE- METHOD: DIRECT SEARCH FROM FAILURE
   +++ SHOWS
    NEXT TEST
   WHEN THE
    CURRENT
    TEST IS
    GOOD

   --- SHOWS
    NEXT TEST
    WHEN THE
    CURRENT
    TEST IS
     BAD
```



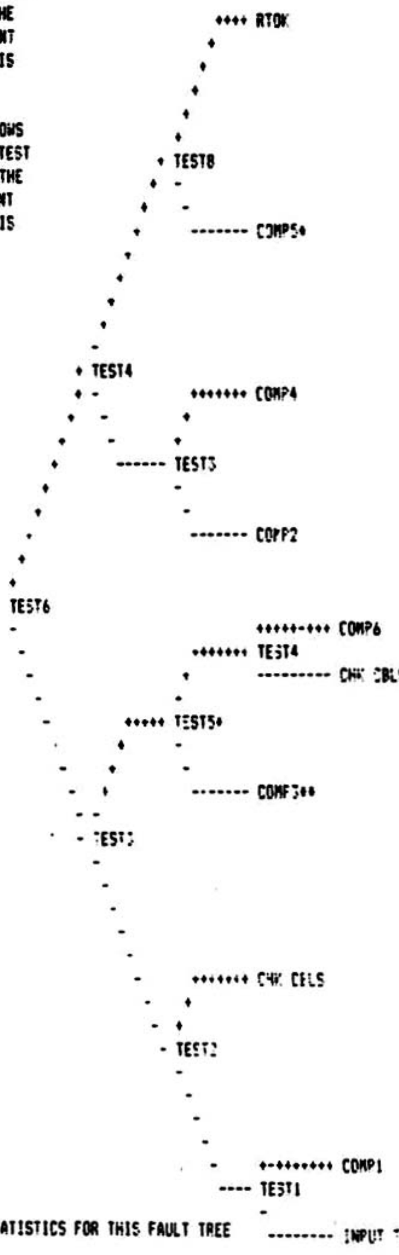
```
SUMMARY STATISTICS FOR THIS FAULT TREE

ELEMENTS   ISOLATED = 8
AVERAGE NUMBER OF TESTS = 3.625
STD DEV OF NO OF TESTS = 1.21334929
LARGEST NUMBER OF TESTS = 5   ELEMENTS  COMP1,TEST1,
SMALLEST NUMBER OF TESTS = 1   ELEMENTS  RTOK,
NUMBER OF INITIAL CONDITIONS =0
```

Figure 9. FAULT TREE FOR DIRECTED STRATEGY

```
INITIAL CONDITIONS: NONE
     FAULT TREE- SYSTEM:SAMPLE- METHOD: ADAPTIVE WEIGHTING: NONE
   +++ SHOWS
    NEXT TEST
   WHEN THE
    CURRENT
    TEST IS
    GOOD

   --- SHOWS
    NEXT TEST
    WHEN THE
    CURRENT
    TEST IS
     BAD
```



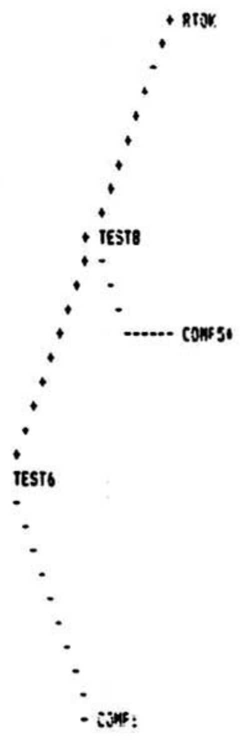
```
SUMMARY STATISTICS FOR THIS FAULT TREE

ELEMENTS   ISOLATED = 8
AVERAGE NUMBER OF TESTS = 3.375
STD DEV OF NO OF TESTS = .48412291
LARGEST NUMBER OF TESTS = 4   ELEMENTS  COMP1,COMP6,TEST1,
SMALLEST NUMBER OF TESTS = 3   ELEMENTS  COMP2,COMP3,COMP4,COMP5,RTOK,
NUMBER OF INITIAL CONDITIONS =0
```

Figure 10. FAULT TREE FOR ADAPTIVE STRATEGY WITH CONSISTENCY CHECKS

```
INITIAL CONDITIONS:TEST4 GIVEN GOOD;
     FAULT TREE- SYSTEM:SAMPLE- METHOD: ADAPTIVE WEIGHTING: NONE
     +++ SHOWS
      NEXT TEST
     WHEN THE
      CURRENT
      TEST IS
      GOOD

     --- SHOWS
      NEXT TEST
      WHEN THE
      CURRENT
      TEST IS
       BAD
```



```
SUMMARY STATISTICS FOR THIS FAULT TREE

ELEMENTS   ISOLATED = 3
AVERAGE NUMBER OF TESTS = 1.66666667
STD DEV OF NO OF TESTS = .471404517
LARGEST NUMBER OF TESTS = 2   ELEMENTS  COMP5,RTOK,
SMALLEST NUMBER OF TESTS = 1   ELEMENT:  COMP6,
NUMBER OF INITIAL CONDITIONS =1
```

Figure 11. FAULT TREE FOR ADAPTIVE STRATEGY USING KNOWN INFORMATION

condition has reduced the feasible set to three possibilities.

To illustrate the weighting procedure, let us assume that test time and test costs are as shown in Table 7. Note that test costs are 10 times test time except for T_4, which has a 4 to 1 ratio. In manually directed, labor-intensive tests, this correspondence is reasonable. T_4 might be a semiautomatic test or a test that requires only periodic manual intervention such as reading a gage 12 hours after pressurizing a system.

Table 7. TEST TIME AND TEST COST DATA

Test	Test Time	Test Cost
T_1	10	100
T_2	20	200
T_3	30	300
T_4	25	100
T_5	50	500
T_6	10	100
T_7	5	50
T_8	10	100

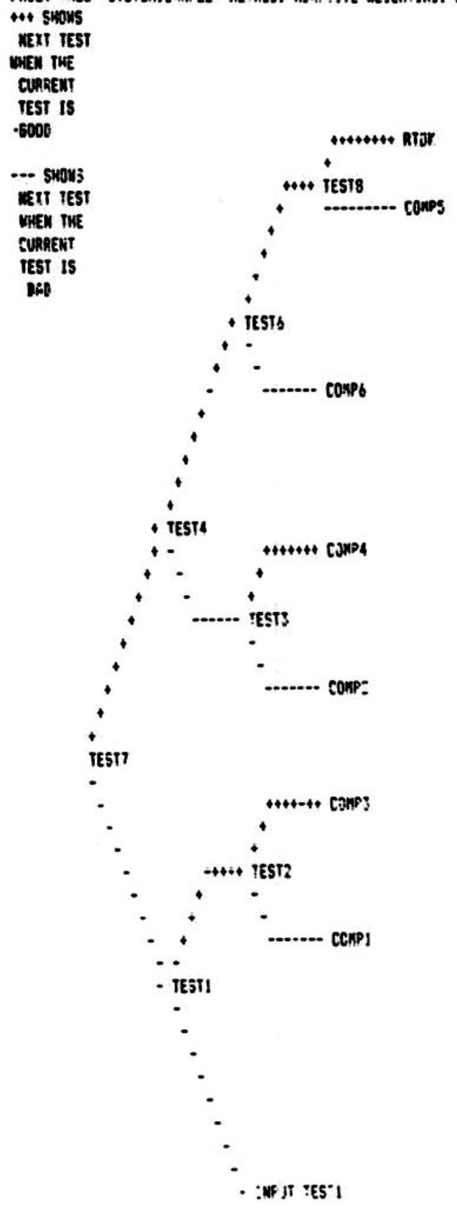
Figure 12 presents the fault isolation strategy when the search is weighted by the test cost data, and Figure 13 presents the test-time weighted search. For these two cases, the first test is T_7, which is the fastest and least costly. The trees differ thereafter because T_4 does not rate as high for quickness of performance as it does for low cost. The asterisks for grouping are eliminated when weighting is performed because the weighting factors tend to make each item unique.

<u>Summary</u>[7]

STAMP, employing an adaptive strategy technique for fault isolation, provides the system user with a powerful tool to ensure or improve testability and fault isolation. Its features include the following:

- Simplified inputs -- single level dependency
- Testability assessment report -- computer generated
- Identification of ambiguities, redundant and excess test points, and feed-back loops
- Fault isolation application

```
INITIAL CONDITIONS: NONE
     FAULT TREE- SYSTEM:SAMPLE- METHOD: ADAPTIVE WEIGHTING: COST MIN
```

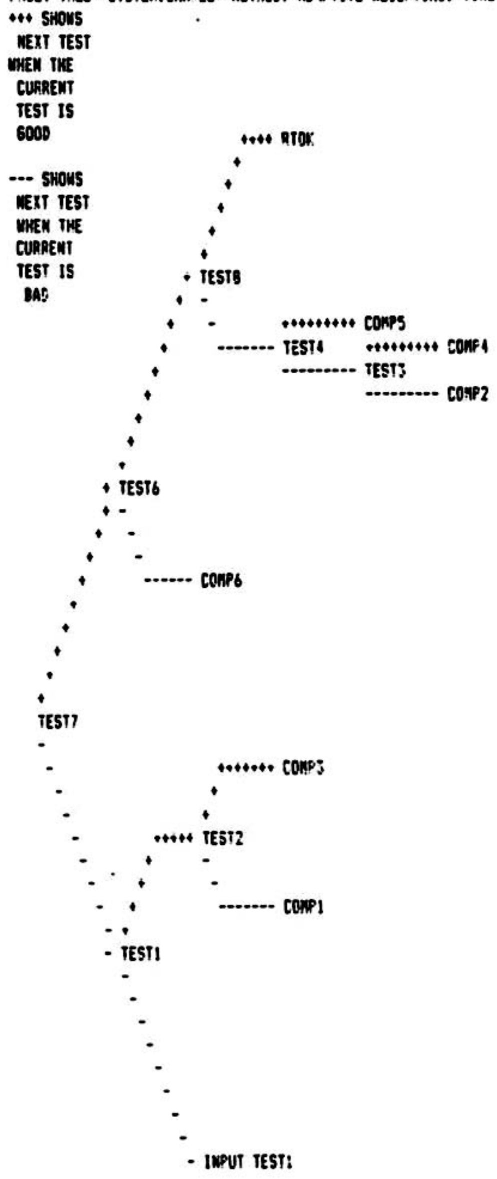

+++ SHOWS NEXT TEST WHEN THE CURRENT TEST IS GOOD
--- SHOWS NEXT TEST WHEN THE CURRENT TEST IS BAD
RTOK
TEST8
COMP5
TEST6
COMP6
TEST4
COMP4
TEST3
COMP2
TEST7
COMP3
TEST2
COMP1
TEST1
INPUT TEST1

```
SUMMARY STATISTICS FOR THIS FAULT TREE

ELEMENTS   ISOLATED = 8
AVERAGE NUMBER OF TESTS = 3.125
STD DEV OF NO OF TESTS = .599478941
LARGEST NUMBER OF TESTS = 4   ELEMENTS  COMP5,RTOK,
SMALLEST NUMBER OF TESTS = 2   ELEMENTS  TEST1,
NUMBER OF INITIAL CONDITIONS =0
```

Figure 12. FAULT TREE WITH SEARCH WEIGHTED BY TEST COST DATA

```
INITIAL CONDITIONS: NONE
     FAULT TREE- SYSTEM:SAMPLE- METHOD: ADAPTIVE WEIGHTING: TIME MIN
```


+++ SHOWS NEXT TEST WHEN THE CURRENT TEST IS GOOD
--- SHOWS NEXT TEST WHEN THE CURRENT TEST IS BAD
RTOK
TEST8
COMP5
TEST4
COMP4
TEST3
COMP2
TEST6
COMP6
TEST7
COMP3
TEST2
COMP1
TEST1
INPUT TEST1

```
SUMMARY STATISTICS FOR THIS FAULT TREE

ELEMENTS   ISOLATED = 8
AVERAGE NUMBER OF TESTS = 3.375
STD DEV OF NO OF TESTS = 1.1110243
LARGEST NUMBER OF TESTS = 5   ELEMENTS  COMP2,COMP4,
SMALLEST NUMBER OF TESTS = 2   ELEMENTS  COMP6,TEST1,
NUMBER OF INITIAL CONDITIONS =0
```

Figure 13. FAULT TREE WITH SEARCH WEIGHTED BY TEST TIME

- Specified initial conditions
- Special or probe tests
- Failure-rate, test-time, or cost weighting
- Multiple failure analyses

The procedure has been used on a number of avionics programs, including both fielded systems and those undergoing development. It has also been applied to the test equipment of avionics systems in devising self-test and self-diagnosis procedures.

STAMP has been useful in identifying system testability improvements, BIT/BITE short-comings, and procedural changes for reduced MTTR, and in developing ATE driver programs for avionics and avionic test equipment. It not only applies to the obvious hardware applications (electronic and nonelectronic) but can be used at the subsystem interface of hybrid systems (for example, where a stability augmentation system meets a hydraulic system).

REFERENCES

1. George Smith II, "Testability Analysis: Predict It More Closely," 1979 Proceedings, Annual Reliability and Maintainability Symposium, Washington, D.C., January 1979.

2. William L. Kiener and Anthony Coppola, "Joint Services Program in Design for Testability," 1981 Proceedings Annual Reliability and Maintainability Symposium, Philadelphia, Pennsylvania, January 1981.

3. Michael L. Labit, G. T. Harrison, and B. L. Retterer, Special Report on Operational Suitability (OS) Verification Study Focus on Maintainability, ARINC Research Corporation Publication 1751-01-2-2395, Annapolis, Maryland, February 1981.

4. Thomas N. Cook and John Ariano, "Analysis of Fault Isolation Criteria/Techniques," 1980 Proceedings Annual Reliability and Maintainability Symposium, San Francisco, California, January 1980.

5. Myron L. Cramer, et al., Logic Model Analysis and Standard Maintenance Information Display System (SMIDS), U.S. Army Applied Research and Technology Labs (USAAVRADCOM-TR-81-D-45), Fort Eustis, Virginia, April 1982.

6. William R. Simpson and Harold S. Balaban, "The ARINC Research System Testability and Maintenance Program (STAMP)," 1982 Proceedings of the IEEE AUTOTESTCON Conference, Dayton, Ohio, October 1982.

7. Harold S. Balaban and William R. Simpson, "Testability/Fault Isolation By Adaptive Strategies," 1983 Proceedings, Annual Reliability and Maintainability Symposium, Orlando, Florida, January 1983.