DEPENDENCY MODELLING PITFALLS

by

William R. Simpson
Institute for Defense Analyses
1801 N. Beauregard St.
Alexandria, Virginia 22311-1772
Phone: (703) 845-6637 Fax: (703) 845-6788 E-Mail: rsimpson@ida.org

and

Mr. John W. Sheppard
ARINC Research Corporation
2551 Riva Road
Annapolis, Md. 21401
Phone: (410) 266-2099 Fax: (410) 266-4010 E-mail: sheppard@arinc.com

Abstract- **Model based diagnostic systems are becoming quite popular. Partially because of the large number of successful applications. Many tools exist for the use of the model based approach, each with a diagnostic advisor. However, experience at modelling over 250 systems has shown that the process of modelling is not straight forward and requires a number of considerations not generally dealt with in the literature. Since the modelling process involves some aspects of art and some aspects of science, the ability to model incorrectly and obtain inaccurate answers is great. This is especially true in the area of automated model development where the concept of a test may be improperly or poorly defined. In this paper we will cover the some of the more subtle nuances of the modelling approach. We will also provide illustrative examples to demonstrate these subtleties.**

I. BACKGROUND

Recent years have seen the proliferation of a wide variety of model-based diagnostic systems. ARINC developed the *System Testability and Maintenance Program* (*STAMP*®) in 1981 to provide a tool for modeling diagnostic information in complex systems and assessing the system testability [1]. In 1988, the *Portable Interactive Troubleshooter* (*POINTER*™) was developed to process models from STAMP for interactive diagnosis [2]. The model used by *STAMP* and *POINTER* is called the *information flow model* and has been described in detail in [3]. Early forms of the information flow model go by names such as *logic model* [4], *causal model* [5], or *dependency model* [6] and are processed by such systems as *LOGMOD* (Logic Model) [4], *STAT* (System Testability Analysis Tool) [7], *WSTA* (Weapon System Testability Analyzer) [8], and *ADS* (Adaptive Diagnostic System) [9]. More recent entries in the field include *DARTS* (Diagnostic Analysis and Repair Tool Set) [10] and AITEST (Artificial Intelligence Test) [11]. This proliferation of tools using the same basic approach has come about because of the large measure of success they have had in solving diagnostic problems.

The information flow model is a *failure-oriented* diagnostic model in which specific failure modes of a system are specified as the possible conclusions to be drawn. Tests detect faults when they are present in the system. The underlying assumption of the failure-oriented model is that at least one failure exists in the system. The diagnostic task proceeds by proving subsystems in the system have not failed. Diagnosis results by examining the set of candidates remaining after all test outcomes are known.

From this discussion we can envision multiple tests detecting the same faults and the failed test outcomes *depending* on the presence of a fault. Further (as will be discussed below), we may be able to directly infer the outcomes of tests based on previous testing thus leading us to a notion of diagnostic information flowing from a fault through a set of tests. This flow corresponds to inference traces that may result from testing.

Common to most of the systems mentioned above is the assumption that diagnosis proceeds from a single failure assumption. One notable exception is *STAT* which is able to perform multiple failure diagnosis by processing a *success-oriented* diagnostic model. Such a model tends to contradict the notion of diagnostic information flow. Tests do not detect failures; tests verify functionality. Unfortunately, the algorithms used by *STAT* are proprietary, so a description of the approach used for multiple failure diagnosis is unknown. The most common notion of dependency has wide variations in

0-7803-1910-9/94 $3.00 © IEEE

application and it is partly for this reason that we have undertaken the writing of this paper.

II. "INHERENT TESTABILITY" AND THE NETLIST

The diagnostic model must be built upon the concept of information sources (tests) and conclusions (failure modes or improper behaviors). The concept of a test is paramount to accurate analysis. Only the concept of a test can provide an accurate representation for a diagnostic system. Those techniques based primarily on signal flow will be **inaccurate.**

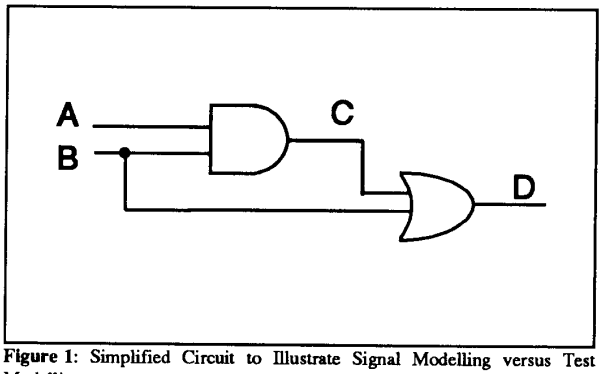

Figure 1: Simplified Circuit to Illustrate Signal Modelling versus Test Modelling

There are several well known examples that can illustrate this. Figure 1 shows one of the examples:

Signal flow alone will provide an answer that is incorrect! For example, assume that there is a fully comprehensive signal test at D (t_D). The dependencies to be listed for t_D will be AND, OR, Input A, and INPUT B. A failed test of t_D will result in the ambiguity of AND, OR, Input A, and INPUT B. A passed test will clear all of the listed elements. The analysis will state that the system is fully testable, yet there is clearly non-detectable faults! Only if we model based upon the concept of a test can we determine the none detections. In this case there are 4 tests that can be performed at D they consist of the D output with each combinations of A and B inputs (i.e., D00, D01, D11, D10). These tests are shown in Table 1.

This means that any modeling approach that relies on net list input will provide an inaccurate model. An accurate model may be obtained by simulation using the above output as an "attribute map". It is in this direction that hope lies for automated model production. Any reading of VHDL files or other net list information can only be considered a start on the modeling process that requires manual intervention to complete.

Table 1: Delineation of the Four Possible Tests in The Sample Circuit

Node	Tests			
	D00	D01	D11	D10
A	0	0	1	1
B	0	1	1	0
C	0	0	1	0
D	0	1	1	0
Fault at D detects	Csa1 Dsa1 Bsa1	Dsa0 Bsa0	Dsa0 Bsa0	Bsa1 Csa1 Dsa1
No Fault at D clears	Csa1 Dsa1 Bsa1	Dsa0 Bsa0	Dsa0 Bsa0	Bsa1 Csa1 Dsa1
Not Detected	Asa0, Asa1, Csa0			

If you note, as some do, that only "non-relevant" failures are non-detectable (a failure that does not effect functional performance is termed non-relevant), you have missed the point. Suppose, for example that tests D00 and D10 were not available, then we could not verify the operation of either the AND or the OR gate since D sa 1 and C sa 1 are not detectable. In fact, the primary input B is not verifiable! While we have demonstrated fallacies in the signal flow analysis, we should point out that the answer is not even close, since the signal flow analysis reported ambiguities greater than the actual test situation would provide. For this reason, the concept of "inherent testability" as an implicit upper bound on testability that includes no definition of a test is particularly bad and should be avoided altogether.

It has been further noted, that special tests could be devised that take advantage of the time delay between the two lines to ascertain whether the devices are working properly. These tests will detect each of the non-detectable items listed in the table, and make the system fully testable. Under these conditions, the testability analysis has worked admirably, pointing out a testing deficiency which the test engineer then eliminated by writing additional sophisticated tests. However, it would be incorrect to assume that this can always be done. Figure 2 shows a slight modification to the circuit that takes the time delay differential away by adding a unit time delay to the lower line. In this circuit the testability analysis (left as an

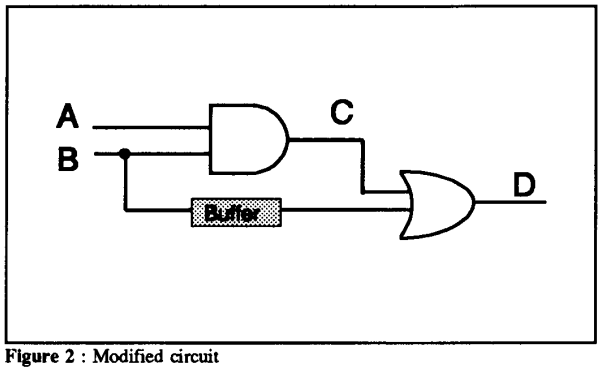

Figure 2 : Modified circuit

exercise to the reader) will note additional non-detections that some clever test engineer may or may not be able to overcome. The important point is that the model itself must be based upon the concept of a test to provide meaningful information.

III. ALTERNATE TEST PARADIGMS

Signal flow analyses also ignore some very real properties of tests. In this section we will describe four properties of real test, that a signal flow analysis must ignore. Not all tests that we may define for a system can be handled in the same way mathematically. Properly describing test behavior may make or break a diagnostic system.

Implicit in the above discussion is the concept of a *binary test outcome*. That is, a test may be considered to pass or fail. Multiple test outcomes are a reality and need to be handled either implicitly or by representing a multiple test outcome as a series of binary outcomes that are tied together and executed concurrently. An example would be a voltage test with nominal, high, and low voltage outcomes. It can be modeled as three binary tests (voltage nominal or not, voltage high or not, and voltage low or not) each with its own set of dependencies or inferences. In fact each could have other characteristics as listed below.

At least in the above model, the tests are symmetric. By symmetric, we mean that when a test fails, it points to a list of causes (the dependency list), when a test fails, it clears that same list. An *asymmetric test* would have two separate lists one for possible causes and one for elimination from consideration. While multiple outcome tests can be modeled as combinations of symmetric tests, asymmetric test inferencing needs to be specifically addressed. The concept of an asymmetric test can most easily be understood by example. It was first discovered when the methodology was being applied to a medical problem and had to do with a diagnostic test that detected the presence of bacteria in urine. If such a detection was made, bacteria was a primary suspect in disease symptoms. But if the test did not detect bacteria, bacterial infection could not be ruled out. At first this was thought to be an artifact of the medical problem, but was quickly found to exist in almost every diagnostic situation. Most commonly, warning lights that detect anomalies are asymmetric in that the lack of an indication cannot be interpreted as the absence of the anomaly. While 80-90% of the tests do follow a symmetric inference, each test must be examined in light of this problem. The difference is inaccurate diagnosis.

An additional area that we must consider is test *cross-linkages*. The normal dependency formulation will make test-to-test inferences on only like test outcomes. That is, a good test outcome will be used to imply additional good test outcomes, while a bad test outcome will be used to imply additional bad test outcomes. A number of cases exist where a opposite or unrelated test outcomes can be inferred. For example, a detection light coming on (bad test outcome) will accuse a list of elements, but can also be used to infer a good light and voltage source. These linkages generally will improve diagnostic efficiency, but must be developed when safety is concerned where a certain test outcome may imply other tests are unsafe to do.

A final area of concern is *conditional outcomes*. The interpretation of certain test may be different for differing situations. For example, the presence of mode switches, system configuration changes or environmental conditions (the latter is encountered in night vision circuits under varying light conditions). These can be handled implicitly or through very careful definition of the test conditions.

IV. ANALYSES FOR FEEDBACK

The diagnostic model is a simplified representation of a system that may or may not use a topological approach. Feedback in information can be identified, but the models as they now stand do not have enough information to do any physics analysis. In electrical circuits, feedbacks may be in place for stability and, as such, can not easily be broken for testing. Other methods of resolution are available, including using the special test paradigms discussed above. There may be some possibility if the modeling tool were tied directly to a simulation or physical model, but as of this writing, the authors know of no such tool set. The best recommendation we can give is to view any feedback analysis with extreme caution. These models are simply not robust enough to handle the problem. An extensive analysis of an example system with feedback is contained in reference [12].

V. ANALYSES FOR BIT

BIT analyses can be performed with this model,but it is important to understand how these analyses are performed and what they mean. The BIT analyses advertised by these techniques will take a set of tests and rank them in order of importance. The result is a rank ordered list that one could use to imbed diagnostics BIT in a system. The analysis assumes that all tests in the model are candidates for such BIT, and the modeler should take care to verify this before running the analysis. Any test that is not a candidate for BIT should be eliminated. The primary process is by use of a tree pruning algorithm. Many such algorithms exist, but we have encountered only two that are used for the BIT analysis. The first is Test Point Utilization (TPU) which simply takes a count of the number of times a test is used in a tree to fault isolate. Thus the vertex of the tree is the most important test. This means that the criteria behind developing the tree is most important. The second is more sophisticated and called the Optimized Resolution Analysis (ORA). This latter technique not only accounts for detection versus isolation but failure rate information, and delineates the details of the faulty tree derivation that should precede the ORA analysis. A complete analysis of the TPU and ORA with examples is contained in reference [13].

VI. SUMMARY

Although a large number of successful analyses have been performed with model based diagnostic tools. The proliferation of such tools is dangerous. Modelling of diagnostic systems is not straight forward and any tools that purports to provide analyses based upon topological descriptions, including net-lists should be viewed with suspicion. It is definitely possible that the analysis of testability in a system too early can provide false information with all the consequences therein. In order to obtain useful information about the diagnosis of a system, the design should have proceeding to at least the point where tests can be defined, if not physically, at least conceptually. Without a definition of tests, it is impossible to get an accurate representation of testability. In fact, the representation obtained from strictly topological information cannot be characterized as conservative or liberal, only inaccurate. Feedback analysis has only limited utility, and BIT analysis must be carefully understood. Finally, the use of output data must be very carefully checked with actual fault insertion if at all possible. The modeling approach is only a mathematical representation of reality and subject to all of the shortcomings of such approaches.

REFERENCES

[1] William R. Simpson and Harold S. Balaban. 1982. "The ARINC Research System Testability and Maintenance Program (STAMP)," *Proceedings of the IEEE AUTOTESTCON*, New York: IEEE Press, pp. 88-95.

[2] William R. Simpson, John W. Sheppard, and C. Richard Unkle. 1989. "POINTER—An Intelligent Maintenance Aid," *Proceedings of the IEEE AUTOTESTCON*, New York: IEEE Press, pp. 26-31.

[3] John W. Sheppard and William R. Simpson. 1991. "A Mathematical Model for Integrated Diagnostics," *IEEE Design and Test of Computers*, Vol. 8, No. 4, pp. 25-38.

[4] Ralph DePaul. 1985. "Logic Modeling as a Tool for Testability," *Proceedings of the IEEE AUTOTESTCON*, New York: IEEE Press, pp. 203-207.

[5] Yun Peng and James A. Reggia. 1990. *Abductive Inference Models for Diagnostic Problem-Solving*, New York: Springer-Verlag.

[6] William Keiner. 1990. "A Navy Approach to Integrated Diagnostics," *Proceedings of the IEEE AUTOTESTCON*, New York: IEEE Press, pp. 443-450.

[7] Detex Systems, Inc. 1990. *Testability Through STAT*, internal documentation.

[8] J. Franco. 1988. "Experiences Gained Using the Navy's IDSS Weapon System Testability Analyzer," *Proceeding of the IEEE AUTOTESTCON*, New York: IEEE Press, pp. 129-132.

[9] Anthony Magliero. 1987. "ADS—The IDSS Adaptive Diagnostic System," *Proceedings of the IEEE AUTOTESTCON*, New York: IEEE Press.

[10] Dr. Li Pi Su, Gregory deMare, and Mary Nolan, 1993, "DARTS: An Enabling Technology for Concurrent Engineering," *Proceeding of the IEEE AUTOTESTCON*, New York: IEEE Press, pp. 383-388.

[11] I. Beniaminy, M. Ben-Bassat, M. Bodenheimer, and M. Eshel, 1993, "Experience in Diagnosing a Remote, Tele-Controlled Unit Using the AITEST Expert System," *Proceeding of the INTERNATIONAL TEST CONFERENCE*, New York: IEEE Press, pp. 37-44.

[12] William R. Simpson, and John W. Sheppard, *System Test and Diagnosis*, Kluwer Academic Press, Boston, Mass., 1994.

[13] John W. Sheppard and William R. Simpson. 1993. "A Systems Approach to Specifying Built-In Tests," *Test Facilities Working Group Conference*, Las Vegas, Nevada, June 1993.