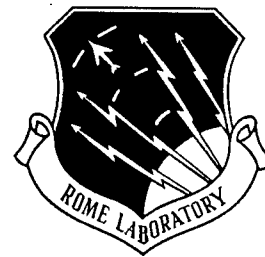


**RL-TR-94-232**  
**Final Technical Report**  
**December 1994**



# **R&M DESIGN EXPERT SYSTEM**

**Martin Marietta**

**Angela M. Kitchen, Jeff Steelhammer, Doug Doscocil,  
Eric Freeman, Glenn Elias, and Rich Ziegler**

**DTIC**  
**ELECTE**  
**MAR 22 1995**  
**G D**

*APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.*

**19950321 157**

*DTIC QUALITY INSPECTED 1*

**Rome Laboratory**  
**Air Force Materiel Command**  
**Griffiss Air Force Base, New York**

This report has been reviewed by the Rome Laboratory Public Affairs Office (PA) and is releasable to the National Technical Information Service (NTIS). At NTIS it will be releasable to the general public, including foreign nations.

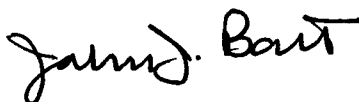
RL-TR-94-232 has been reviewed and is approved for publication.

APPROVED:



DALE W. RICHARDS  
Project Engineer

FOR THE COMMANDER:



JOHN J. BART  
Chief Scientist, Reliability Sciences  
Electromagnetics & Reliability Directorate

If your address has changed or if you wish to be removed from the Rome Laboratory mailing list, or if the addressee is no longer employed by your organization, please notify RL ( ERSR) Griffiss AFB NY 13441. This will assist us in maintaining a current mailing list.

Do not return copies of this report unless contractual obligations or notices on a specific document require that it be returned.

## REPORT DOCUMENTATION PAGE

Form Approved  
OMB No. 0704-0188

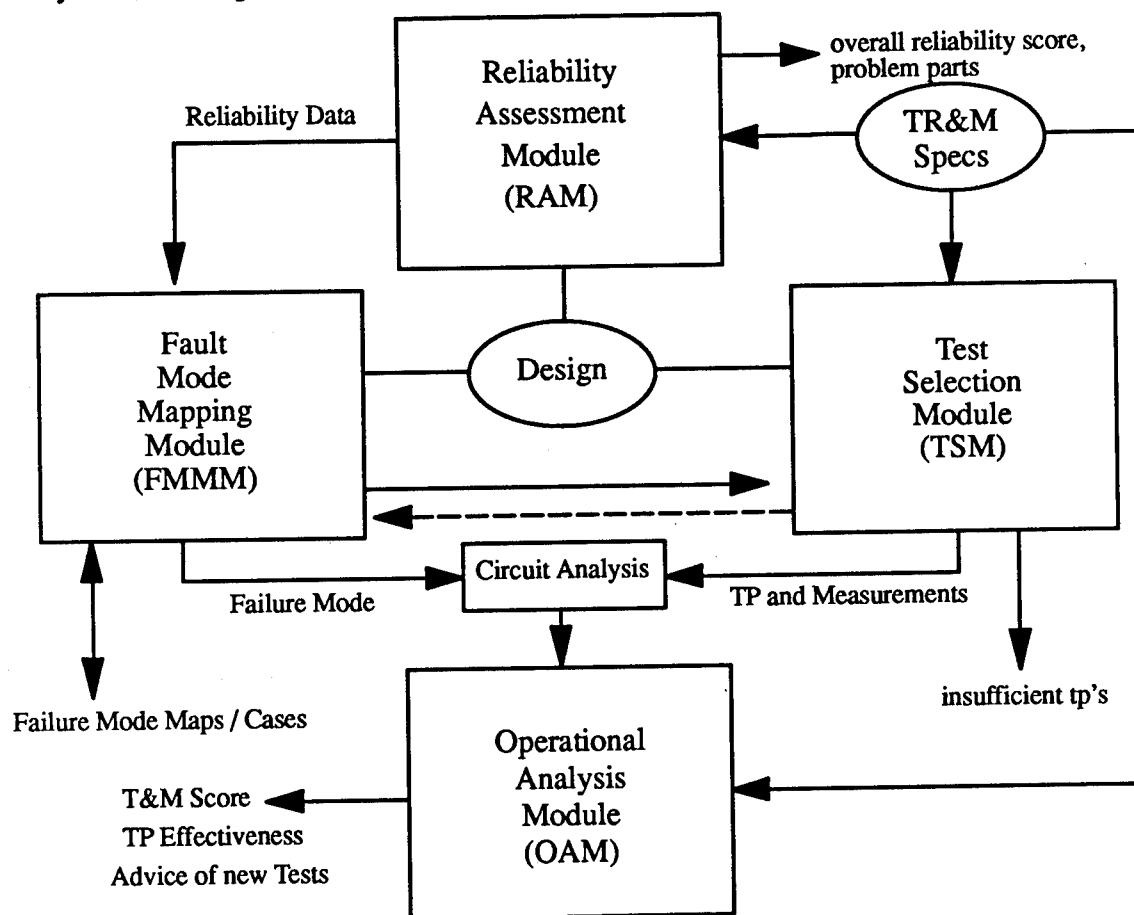
Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188), Washington, DC 20503.

1. AGENCY USE ONLY (Leave Blank)		2. REPORT DATE December 1994		3. REPORT TYPE AND DATES COVERED Final Sep 91 - Sep 94	
4. TITLE AND SUBTITLE  R&M DESIGN EXPERT SYSTEM				5. FUNDING NUMBERS C - F30602-91-C-0159 PE - 62702F PR - 2338 TA - 02 WU - 5G	
6. AUTHOR(S) Angela M. Kitchen, Jeff Steelhammer, Doug Daskocil, Eric Freeman, Glenn Elias, and Rich Ziegler					
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Martin Marietta Martin Marietta Laboratories Moorestown NJ 08057				8. PERFORMING ORGANIZATION REPORT NUMBER  N/A	
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) Rome Laboratory (ERSR) 525 Brooks Rd Griffiss AFB NY 13441-4505				10. SPONSORING/MONITORING AGENCY REPORT NUMBER  RL-TR-94-232	
11. SUPPLEMENTARY NOTES Rome Laboratory Project Engineer: Dale W. Richards/ERSR/(315) 330-3476					
12a. DISTRIBUTION/AVAILABILITY STATEMENT Approved for public release; distribution unlimited.				12b. DISTRIBUTION CODE	
13. ABSTRACT (Maximum 200 words)  This effort investigated and developed a proof-of-concept software system that combines reliability, testability, and maintainability analysis capabilities into a single tool. The system recommends tests and testpoints based on potential faults (associated with a component within the design) and a functional-block description of the design. The system directs the user in injecting faults into a simulation model. The tests and testpoints are then used to perform fault isolation using the results of the simulation. Case-based reasoning techniques were used as a basis for the testability and test coverage aspects of the program. The casebase is a repository of "past knowledge" gained from expert designers and testability experts. The system adds to this repository if an unfamiliar situation occurs, allowing the system to "learn" over time. The expert knowledge was encoded into a Fault Mode Map that represents how the experts relate faulty behavior of the circuits to actual failures. Rome Laboratory Oracle (a reliability prediction tool), Mentor Graphics Design Architect (a schematic capture tool), and Analogy Saber (a circuit simulation tool) were integrated to allow the user to exercise all system capabilities from a single user perspective. The system was tested against Martin Marietta power supply design.					
14. SUBJECT TERMS Reliability, Testability, Computer-aided design, Case-based reasoning, Power supplies, Fault analysis, Electronics				15. NUMBER OF PAGES 124	
				16. PRICE CODE	
17. SECURITY CLASSIFICATION OF REPORT UNCLASSIFIED	18. SECURITY CLASSIFICATION OF THIS PAGE UNCLASSIFIED	19. SECURITY CLASSIFICATION OF ABSTRACT UNCLASSIFIED	20. LIMITATION OF ABSTRACT  UL		

## Executive Summary

The Reliability and Maintainability Design Expert System (R&M Expert System) is a recently concluded, three-year program sponsored by Rome Laboratory. Martin Marietta Laboratories • Moorestown developed the contract, which resulted in a proof-of-concept system that combined three types of analyses into a single tool; these analyses were reliability, testability, and maintainability.

The R&M Expert System recommends test and test points based on potential faults (a component within the design) and a functional block. It direct users to inject faults into a simulation model, and it uses tests and testpoints defined by users and the results of the simulation to isolate faults. We used a power supply design from a division of Martin Marietta to test the system, which performed to specification.





## Concurrency is the main requirement for testability, reliability, and maintainability (TR&M) issues

Electronic systems designed for TR&M require specialists to participate from the beginning of the design effort; otherwise, they are unable to successfully address issues that may arise. Specification requirements, which include government standards, drive reliability design. Standard testability analysis involves a structured process to evaluate testability performance. Primary testability performance parameters are fault-detection coverage, fault isolation, and false alarm percentage. The designer considers the following when defining a hardware testpoint and selecting a software test on the final product:

- Mechanical accessibility considerations
- Testability requirements and analyses
- Hardware/software test design trade-offs
- Test equipment capabilities
- Cost of test and repair processes.

Accession For	
NTIS	CRA&I <input checked="" type="checkbox"/>
DTIC	TAB <input type="checkbox"/>
Unannounced <input type="checkbox"/>	
Justification _____	
By _____	
Distribution /	
Availability Codes	
Dist	Avail and/or Special
A-1	

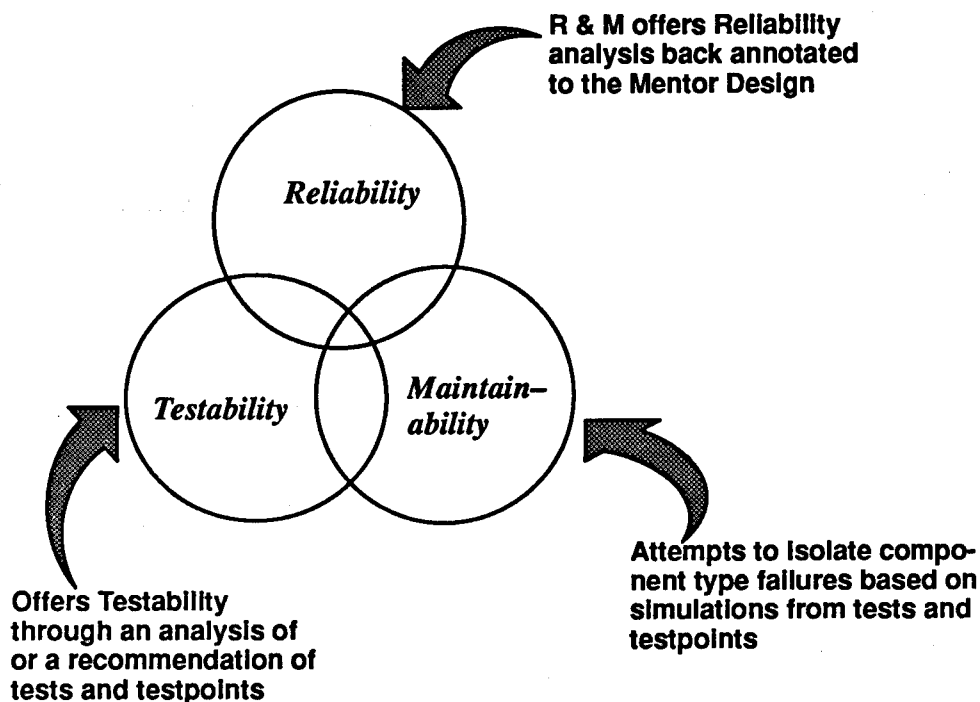
The R&M Expert System program contained three phases:

- **Phase I** - A 12-month effort that defined the system concept based on existing human processes in TR&M design and on tools to support the designer. The effort examined links between human processes and tools so that the concept had maximum impact. Also conducted was a survey of available TR&M tools.
- **Phase II** - An 18-month effort that designed and implemented the skeleton system. Tool interfacing, data extraction, and data representation were major design elements. A critical development was the Fault-to-Failure-Mode Map (FFMM), which was the foundation of the system's knowledge base.
- **Phase III** - A 6-month effort refined system concepts.

## **Martin Marietta developed a unique methodology to integrate the three types of analysis**

Martin Marietta developed a unique methodology to integrate analyses of testability, reliability, and maintainability. The methodology combined these analyses in new ways; for example, using reliability to guide the maintainability analysis. The R&M Design Expert System is extensible to different designs, but it currently concentrates on power supplies.

Martin Marietta Laboratories • Moorestown developed a method to examine designs from the perspective of TR&M. This allowed a framework to consider how each of the three impacted the others. It also gives designers a new perspective on the design through the combination of the different fields. The methodology allows users to see which areas of TR&M do not have tools to assist the designers.



## **We created a casebase of past knowledge that grows with system use**

The R&M Expert System uses past knowledge through Case-Based-Reasoning technology. The system's casebase is a repository of past knowledge gained from expert designers and testability and maintainability experts. The system adds data to its repository when unfamiliar situations occur. The system uses its casebase in two ways:

- To recommend tests and test points, based on components and functional blocks
- To isolate faults based on a test, test point, and signature behavior.

The design reduces the number of data repositories, making updates to the casebase easier.

We formed a Fault-to-Failure-Mode Map (FFMM) that represents how experts relate circuit faults to that which may actually be wrong with the circuit. The FFMM is a central part of the R&M System, and it is used by the testability and test coverage modules.

### **The Elements in the Cases:**

Name : Type of fault (e.g. open diode)

Functional Block: Functional block fault occurred

Failure Mode: Effect on the Circuit

Measurement Point: Point where measurement is taken

Measurement Type: Type of measurement (e.g. DC voltage)

Signature Behaviour: observed behavior

Measurement Devices: devices used in measurements

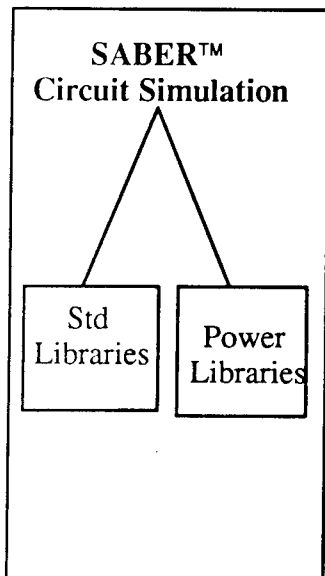
## The R&M Expert System interfaces with Mentor Graphics' Design Architect, Analogy's Saber, and Rome Laboratory's Oracle

The interface with Mentor Graphics' Design Architect, Analogy's Saber, and Rome Laboratory's Oracle are a significant part of the R&M Expert System. Without this interface, the system would not be able to get the information that is required for the expert system.

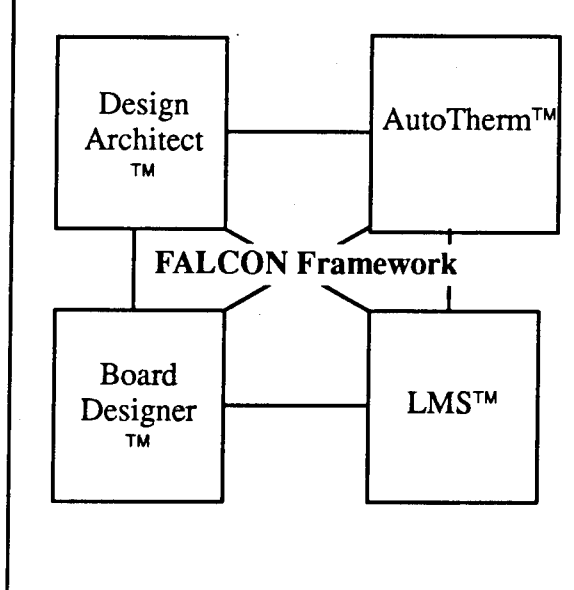
The interface with Design Architect is seamless and transparent after initial set-up. Design Architect provides most of the system's information. Martin Marietta implemented the interface early in the program to lay the groundwork for the informational structure used in the program.

There is a standardized procedure to interface with Saber and Oracle through system files. The Oracle interface allows users to gather reliability data, which reduces mistakes and time required for the analysis. The Saber interface is also transparent by interacting with it solely through automatically generated files. Oracle provides the system with reliability data on the design. Saber provides circuit simulation results for test coverage analysis.

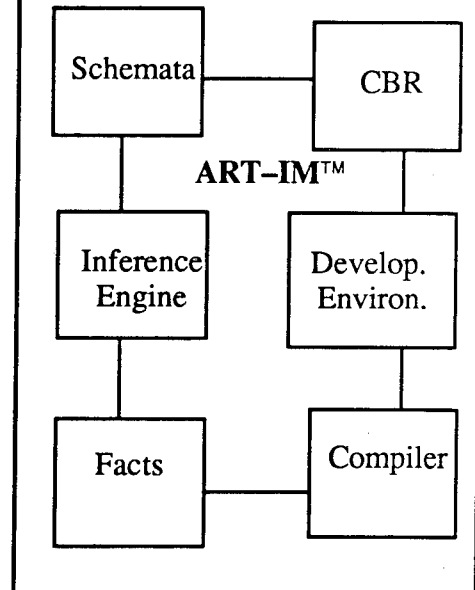
### Analogy Tools



### Mentor Graphics Tool Suite



### Inference Corp. Tools



## **Martin Marietta overcame many challenges in developing the R&M Expert System, including interfacing tools, developing a design methodology, and integrating the whole into a useful prototype**

A major challenge was interfacing Mentor Graphics' Design Architect, Analogy's Saber analysis tool, and Martin Marietta's R&M system. Issues focused on differing representations of design in each tool. Interfacing the tools was a major accomplishment early in the program; it allowed us to assist the designer for the remainder of the program. Accomplishments also included developing a general interface could enable commercial-off-the-shelf (COTS) tools to replace Mentor Graphics or Analogy tools without a total rework of the system.

A major challenge was developing a way to bring the circuit designer into the TR&M analyses earlier. The prototype is an implementation of the resulting design methodology, which contains areas that were not implemented in the proof-of-concept prototype. However, it clearly defines these areas and how they would interact with the system. The R&M Expert System implemented the areas of reliability, testability, and a subset of maintainability. We were unable to implement the overlapping areas of the methodology.

We developed the prototype system to bring the circuit designer closer to the TR&M practices. The prototype provides a common framework in which designs can perform "what-if" analysis of their designs. By integrating the system with common used CAD/CAE tools, the prototype allows designers full freedom for development.

# Table of Contents

<b>1. Introduction .....</b>	<b>1</b>
1.1 Program Scope and Objectives .....	1
1.2 The R&M Design Expert System Concept .....	3
1.3 Report Overview .....	3
<b>2. Background .....</b>	<b>4</b>
2.1 The Human Process in Design: An Overview .....	4
2.1.1 The Specification .....	4
2.1.2 Design Process with Testability, Reliability & Maintainability in Mind ...	5
2.1.2.1 Reliability .....	5
2.1.2.2 Testability .....	6
2.1.2.3 Test Point and BIT Selection .....	7
2.1.2.4 Automated Test Equipment -Army and Navy Standards .....	10
2.1.3 Summary .....	10
2.2 A Survey Of The Tools That Aid The Design Process .....	12
2.2.1 Overview .....	12
2.2.2 Concurrent Engineering And Embedded Tools .....	12
2.2.2.1 Reliability Analysis .....	13
2.2.2.2 Maintainability Analysis .....	18
2.2.2.3 FMECA .....	20
2.2.2.4 Testability .....	22
2.2.2.5 Other Related Tools .....	23
2.2.3 Hierarchy, Division Of Expertise, Data Flow & Concurrent Engineering .	23
2.2.4 An Observation On The Growth Of R&M Analyses .....	24
2.2.5 Summary .....	25
2.2.6 Conclusions .....	26
2.3 The Testability Process: A Detailed Look .....	27
2.3.1 Introduction .....	27
2.3.2 Programmatic Testability Issues .....	27
2.3.3 Fault Coverage Analysis .....	28
2.3.4 Diagnostic Development Process .....	30
2.3.5 Technical Testability Issues .....	31
2.3.6 Typical Fault Coverage Database .....	31
2.3.7 Multiple Viewpoints of Diagnostic Data .....	32
2.3.8 Functional and Physical Model Development .....	33
2.3.9 Testability in an Integrated Product Team Environment .....	34
2.3.9.1 R&M Functions in a CAD/CAE Environment .....	35
2.3.10 Summary .....	36
<b>3. System Concepts and Methodologies .....</b>	<b>38</b>
3.1 Supportability Analysis .....	38
3.1.1 Reliability .....	40
3.1.2 Testability .....	41
3.1.3 Maintainability Metric .....	41
3.1.4 Reliability Metric .....	42

3.1.5 Testability Metric .....	42
3.1.6 Maintainability Metric .....	45
3.1.7 Reliability/Testability Combined Metric .....	46
3.1.8 Reliability/Maintainability Metric .....	48
3.1.9 Testability/Maintainability Metric .....	49
3.1.10 Testability/Reliability/Maintainability Metric .....	49
3.2 The Concept: The R&M Design Expert System .....	50
3.2.1 Scope: Focus on Test Selection and Coverage .....	50
3.2.2 Concept Design Drivers: Tools Used in the System .....	52
3.2.2.1 Mentor Tool Suite: The Base CAD/CAE Environment .....	52
3.2.2.2 Saber Analog Simulation .....	52
3.2.2.3 RL—Oracle .....	53
3.2.2.4 ART—IM: The Knowledge—Based System Tool .....	53
3.2.3 Application of Knowledge—Based Tools Electronic System Design .....	53
3.2.4 Required System Inputs .....	54
3.2.4.1 A Hierarchical Design to Analyze .....	54
3.2.5 Additional Design Representation .....	55
3.2.5.1 TR&M Specifications .....	56
3.2.6 System Modules .....	56
3.2.6.1 Reliability Assessment Module(RAM) .....	56
3.2.6.2 Fault Mode Mapping Module (FMMM) .....	57
3.2.6.3 Test Selection Module (TSM) .....	58
3.2.6.4 Operational Analysis Module (OAM) .....	59
3.2.7 Assumptions .....	60
<b>4. Implementation .....</b>	<b>61</b>
4.1 Development Overview .....	61
4.2 Interfacing with Mentor Graphics .....	63
4.3 Interfacing with the Design .....	63
4.4 Fault—to—Failure—Mode Map .....	65
4.4.1 Creating the Fault—to—Failure—Mode Map .....	66
4.4.2 Using the Mode Map for Testability .....	66
4.4.3 Using the Mode Map for Fault Isolation .....	67
4.5 Casebase Methodologies and Views .....	68
4.6 Casebase Views .....	68
4.6.1 Testability View .....	69
4.6.2 Test Coverage View .....	70
4.7 Reliability Assessment Module .....	70
4.8 Test Selection Module .....	71
4.9 Operational Analysis Module .....	71
4.9.1 OAM Databases .....	72
4.9.2 Parsing the Saber Output .....	73
4.9.3 Fault Behavior Analysis .....	74
4.9.4 Fault Isolation .....	77
4.9.5 Update Mapping Casebase .....	80
<b>5. Lessons Learned .....</b>	<b>81</b>
5.1 Working with the Tools .....	81

5.2 Using Casebase Reasoning Approach .....	82
5.3 Testability .....	83
5.4 Degrees of Fault Isolation .....	84
5.5 Importance of Standard Naming Conventions .....	85
5.6 Simulation Software Version or Vendor Changes .....	85
5.7 Simulation Accuracy Versus System Performance .....	86
5.8 Casebase Continuity .....	87
5.9 Designer Versus Automation .....	87
5.10 Simulation and Fault Diagnosis Software Integration .....	88
5.11 Measurement Points: Specific to General .....	89
5.12 Effects of Casebase Expansion .....	89
5.13 Manipulation of Saber Output Files .....	89
5.14 System Resource Requirements .....	90
5.15 Conclusions .....	90
 6. Glossary .....	 92
 7. Appendix .....	 97
 Index .....	 100



## Table of Figures

Figure 1. Program objectives. ....	2
Figure 2 Typical testability requirements summary. ....	28
Figure 3. Fault coverage analysis. ....	29
Figure 4. Diagnostic development process. ....	30
Figure 5. Typical fault coverage analysis database. ....	32
Figure 6. Multiple viewpoints of diagnostic data. ....	33
Figure 7. Functional and physical model development. ....	34
Figure 8. R&M and design CAD/CAE functions. ....	36
Figure 9. Design methodology for testability, reliability, and maintainability. ...	38
Figure 10. Reliability Example ....	43
Figure 11. Testability Example ....	44
Figure 12 Failure rate weighted testability example. ....	47
Figure 13 System concept. ....	51
Figure 14. System concept with different view/same functionality Shared Data. ....	51
Figure 15 An example ART-IM representation. ....	55
Figure 16 Reliability Assessment Module(RAM). ....	57
Figure 17 Fault Mode Mapping Module (FMMM). ....	58
Figure 18 Test Selection Module (TSM). ....	59
Figure 19 Operational Analysis Module(OAM). ....	60
Figure 20 The R&M Design Expert System. ....	62
Figure 21. Relationships used in R&M Expert System Mentor Graphics . ....	63
Figure 22 Examples of the models of functional blocks . ....	65
Figure 23 Example of a Fault-to-Failure-Mode Mapping . ....	66
Figure 24. Structure of the casebase used in the testability and test coverage modules. ....	68
Figure 25 Architecture of the Operational Analysis Module (OAM). ....	72
Figure 26 Time versus Voltage display for the baseline and fault circuits as viewed by the designer during the simulation. ....	75
Figure 27 General software architecture of the Fault Behavior Analysis component. ....	77
Figure 28 Example of multiple fault signature associations to a single fault. ...	79

## Table of Tables

Table 1. Test coverage. ....	45
Table 2.. Test coverage with lvalues. ....	48
Table 3. Failure rate weighted maintainability. ....	49

### ***Acknowledgements***

*This report was prepared by Martin Marietta Laboratories • Moorestown and Automated Systems Department (ASD). The descriptions of concepts and developments herein are a direct result from the contribution of the following individuals: Doug Doscocil, Eric Freeman, Eldon Sutphin, Robert Harwell, and Lee Stratton at ASD, and Angela Kitchen, Rich Zeiger, Jeff Steelahmmer, Glenn Elias, Virginie Harris, and Bill Tinney from Martin Marietta Laboratories • Moorestown.*

# 1. Introduction

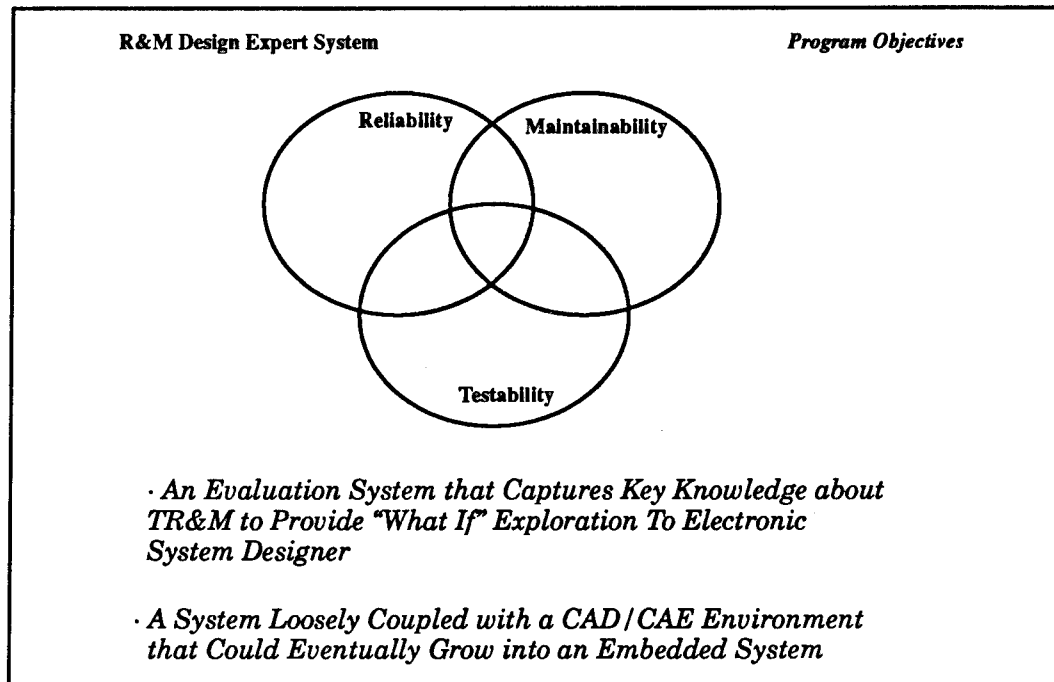
## 1.1 Program Scope and Objectives

In recent years, the defense industry has, appropriately, placed increased emphasis on reliability, maintainability and testability in order to develop and maintain a more cost-and-mission-effective fighting force. The advent of Integrated Diagnostics has resulted in methodologies and tools to assist in supportability requirement refinement and tradeoffs, thereby assisting the front end of the system development cycle. Additionally, many tools have been developed for assisting Automatic Test Equipment (ATE) and Test Program Set (TPS) developers to define and implement better systems once the system under test has been defined. However, the capabilities of computer-aided design tools have increased exponentially, there has been little associated development in reliability, maintainability and testability tools which apply to the detailed design phase. One of the classic dilemmas facing the industry has been that a specialty engineer cannot evaluate a design until it is at least somewhat complete, and yet once it is complete, his ability to affect it is impaired due to the direction already taken by the designers, as well as the pressure to meet schedule.

Therefore, if it were possible to approach a goal of on-line reliability, maintainability and testability (RM&T) effectiveness feedback to a designer while he is making other related design decisions (such as ASIC/SW, analog/digital, Hybrid/PC, ...) then it would be more likely that the appropriate decisions would be made during the detailed design phase. It is thus a purpose of this program to determine the feasibility of integrating RM&T analysis with computer-aided electronic design tools. Since such tools will continue to be developed, it was important to determine whether the interface of such tools, such as the interconnection list, could be exploited for RM&T analysis.

The R&M Design Expert System (F30602-91-C-0159) was a three-year contract with the United States Air Force/Rome Laboratory to explore issues related to coupling knowledge-based system tools with CAD/CAE tools. The program focused knowledge-based system techniques on improving testability in design (including testpoint selection/review, test scenario and equipment selection, and verification), and using information from testability, reliability, and maintainability (called TR&M analysis). Martin Marietta demonstrated these techniques in analog power design using the following tools:

- Mentor Graphics' Falcon Framework™ with Design Architect™ to represent and analyze electronic system designs
- Analogy Inc.'s Saber™ analog circuit simulator to collect data
- Inference Corp.'s ART-IM™ knowledge-based system tool.



*Figure 1. Program objectives.*

Figure 1 shows the program objectives, which included developing an experimental system that captures key TR&M knowledge to provide designers, or other potential users within the electronic system design community, with a "what if" exploration capability in a CAD/CAE loosely "coupled" framework. This capability could eventually be fully embedded with the design tool environment.

We tested the R&M Design Expert System on a power supply design from Martin Marietta's AstroSpace division. This power supply consisted of six functional blocks: reset block, power control block, shutdowns, input stage, transformers, and a regulator. The total number of components was approximately 400.

## **1.2 The R&M Design Expert System Concept**

Our review of the design and tools available to aid the TR&M process showed that applying knowledge-based system techniques impacted testability the most. There are numerous reliability tools available to electronic system designers, but few tools that aid the design with functional testability.

The R&M Design Expert System concept only addresses the diagnosability subset of maintainability; specifically test coverage (e.g., can the system detect and isolate the fault(s) given the testpoints and associated measures?). The system concept focuses on improving testability in design (including testpoint selection/review, test scenario and equipment selection, and verification), and TR&M analysis (diagnosability subset – test coverage only). The R&M Design Expert System has four main system modules. Though we present it in this document as a serial process for discussion purposes, the system can be thought of as a body of shared, interacting knowledge about TR&M, wrapped around a set of "core tools" that use information available from design data.

## **1.3 Report Overview**

Section 2 provides background, including a discussion the human process in design, a survey of available reliability, testability and maintainability tools, and an in-depth look at the testability process.

Section 3 provides an overview of the system concepts and methodologies, including a discussion of our unique TR&M methodology and its application in the Supportability Analysis subsection.

Section 4 details implementing the R&M Design Expert system.

Section 5 discusses the "lessons learned" during the program.

Appendix A list the project deliverables.

## **2. Background**

### **2.1 The Human Process in Design: An Overview**

The design process, starting with a specification for a module of an electronic system, may include a design engineer, mechanical engineer, reliability and maintainability engineer, testability engineer, and parts control specialist. In this context, module refers to an electronic system part dedicated to a particular function within the design. For example, the power supply is one module of an electronic system.

#### **2.1.1 The Specification**

In most cases, the specification for an electronic system will:

- Define system requirements, including electrical performance (e.g., input power, output power, fault performance), physical characteristics (e.g., weight, dimensions, connectors), reliability and maintainability requirements, environmental requirements and stress screening, and design and construction materials and processes. Maintainability requirements typically include specific testability requirements, (e.g., fault detection and fault isolation).
- Specify documents applicable to the design process. These may include military standards and handbooks as well as non-government documents such as manuals and procedures guides. For example, MIL-STD-1629, "Procedures for Performing Failure Mode Effects and Criticality Analysis (FMECA)" and MIL-HDBK-217, "Reliability Prediction for Electronic Equipment," are commonly required with respect to reliability. MIL-STD-471, "Maintainability Verification/Demonstration/Evaluation" and MIL-STD-2165, "Testability program for Electronic Systems and Equipment," address maintainability and testability issues.
- Specify quality assurance provisions, such as special tests and examinations, test procedures, equipment, and facilities.

## 2.1.2 The Design Process with Testability Reliability & Maintainability (TR&M) in Mind

Design engineers usually begin the design process by creating a functional block diagram of the design task based on the module-level specification. Following a concept review, each block is designed in detail and the results used to generate a detailed schematic and parts list (usually using a CAD system). A parts control specialist (if part of the team), helps select parts. The parts specialist provides preferred parts, derating and screening criteria, and specifications and source control drawings for specialty parts. (The customer typically stipulates a system-level derating when it is desirable. Derating refers to using an item whose applied stresses are below the rated values or lowering the rating of an item in one stress field to allow an increase in another stress field.) Careful parts selection is one of the primary TR&M efforts during early design stages.

The parts specialist selects the parts and decides the screening process the parts will undergo. The reliability reflects these choices. For example, if the design engineer selects a part for which no military drawing exists, the design engineer will ask the parts specialist if the new part will work under the required conditions. If it will work under the conditions, a specification control drawing describing the types of screening to be done to the part will be made. (In the screening process the part may be burned in, tested, temperature cycled, etc.) Screening enhances the reliability of the part by eliminating, early in the process, specific pieces that may fail from the batch. Screening does not qualify a part type or batch for a higher reliability rating, however.

Concurrent with the detailed electrical design process, the mechanical engineer creates a design for packaging the product to meet environmental and maintainability requirements. This includes power stresses and thermal analysis that may impact both the reliability and electrical performance of the design.

### 2.1.2.1 Reliability

The reliability engineer becomes part of the design process upon a completion of a preliminary parts list, preliminary electrical design, and preliminary mechanical design. Parts must withstand many stresses - temperature, voltage stresses, etc. - that affect their reliability and which must be accounted for by the reliability engineer. To complete the Failure Modes and Effects Analysis (FMEA), possibly with the help of an automated reliability prediction tool, the engineer:

- Analyzes operational requirements from the specification (e.g., the system will be used in 70°F at 50% humidity)



- Application requirements stipulated by the design engineer (e.g., apart will be used at 24 volts as opposed to its max rating of 36 volts)
- The parts lists, schematics, and other pertinent documents.

The FMEA analysis is used to predict the effect of failures on system operation. Once the FMEA is complete, the reliability engineer then adds criticality factors and executes MIL-STD-1629 (the FMECA), which is the procedure to assure adherence to MIL-HDBK-217. The FMECA is used to predict the effect of failures on system performance (e.g., effect on mission criticality). It is of little use during the design process because it offers little benefit to module-level fault isolation.

Reliability prediction is done on a given board according to the parts list. It is not a check-on functional design; passing the reliability test does not guarantee that a design will work satisfactorily. Note that the interaction between the reliability engineer and others in the design process is often a one-way communication - most often from the design engineer to the reliability engineer. In most cases, if the design meets the specification, the reliability engineer will not suggest design changes since a pass or fail is all that is necessary for the design.

### 2.1.2.2 Testability

Design practices that enhance design testability include: partitioning circuits, analyzing feed-back loops, and providing test points. If a testability engineer is on the design team, the entrance of the engineer into the process varies and is influenced by several factors. The most important of these is the testability requirements specifications. The system specification sets forth comprehensive testability performance requirements. Testability engineering support is a crucial role early in the system design process. Testability engineers, therefore, are part of a concurrent design effort. From the beginning the process is driven by the specification, in which test requirements are clearly defined for all levels of maintenance.

The maintenance level is defined by who or what detects a failure, where the repair or replacement will be done, and what maintenance actions will take place. For example, usual maintenance levels include:

- *User Level*- The user will detect fault but will not isolate or repair
- *On-Site Repair Team* - Trained maintenance technician confirms failure, isolates the fault to a line-replaceable unit (LRU), and replaces the unit

- *Shop Repair* - The LRU is taken to a local shop where further fault isolation can be done down to the shop-replaceable unit (SRU)
- *Depot or Factory Repair* - If economically repairable, the SRU is sent back to the factory where the fault will be traced to the component, which is then replaced. Fault isolation is often the most important parameter at the depot-level testing of a module.

The standard for testability analysis is MIL-STD-2165, which describes a structured process to evaluate testability performance. The primary testability performance parameters are fault detection coverage, fault isolation, and false alarm percentage. The system specification should set values for these performance parameters by identifying pertinent requirements at each maintenance level. For example, a specification may address three of the maintenance levels as follows:

- *Unit (on-site repair)* - Detection 97% - Isolation to LRU 95% - Alarm 5%
- *Shop* - Detection n/s - Isolation to 1 SRU 97% - Alarm n/s
- *Depot or Factory* - Detection/s - Isolation to 1 Part 80% - Alarm n/s

n/s = none specified

In this case, a fault detection rate is set at 97%; a 95% rate is required to isolate the detected fault to the LRU with no more than a 5% false alarm rate. If the system is brought to a repair shop with a detected fault, the isolation rate to a single SRU must be 97%. Similarly, the depot or factory should be able to isolate the fault to a single part 80% of the time. The testability engineer influences the system design so that these criteria can be met.

### 2.1.2.3 Test Point and BIT Selection

Hardware test point and software test selection on the final product is generally based on five factors:

1. Mechanical accessibility considerations
2. Testability requirements and analyses
3. Hardware/software test design trade-offs

#### 4. Test equipment capabilities

#### 5. Cost of the test and repair processes.

Test points are required when part of the system cannot meet its fault detection/isolation requirements. Test point selection is the result of a trade-off analysis. At the module (or board) level, the first issue is how the module will be tested. For example, a board with a processor and built-in-test (BIT) capability may not be able to execute its built-in-self-test when plugged in to a depot's automated test equipment (ATE), which the depot will use for module-level testing. ATE, without built-in-probe or bed-of-nails-test capability, may also require that test points be brought to connectors. There is a practical limit to the number of test points which can be brought out, and it is usually a small fraction of the number of internal nodes that are candidate test points. Therefore, test point selection is often based upon the weighting of the test value of each candidate node, where the weighting criteria includes predicted failure rates for the components in the fault isolation path (FMEA data), and the richness of the diagnostic information contained in the signal at the testpoint.

In an organization (or project) that does not include testability engineers, the design engineer may select tests and test points. Normally, these will be selected to serve the particular needs of the design engineer in the design process, i.e., to determine if the design will work. The design engineer may not consider fault detection and isolation concerns or, if considered, may not have the tools or knowledge to make effective trade-off decisions.

If concurrent engineering practices are observed, the test engineering group of the manufacturing operation (the people concerned with quality assurance for the system) may choose the tests and test points. They will emphasize system performance verification - not fault detection and isolation. Their test and test point selection will often be based on the nature of the ATE available to them in house so that they can easily hook up and exercise a system. If given input yields expected output, the system will be passed. If a problem is discovered and fault isolation is required, the system is failed and will be sent back to engineering for lab testing.

The ideal case for specification of the tests and test points is a testability engineer aware of the specification's requirements for fault detection and isolation, the FMEA results, the system design, the levels of maintenance required, etc. The testability engineer will combine this information into a cogent basis for test point and test selection at the LRU level. Often this is done with the assistance of automated tools for fault coverage prediction. Several tools exist commercially, and Martin Marietta's Automated Systems division developed tools for this ap-

plication. Each tool has its limitations, caveats, and weaknesses and must be thoroughly understood by users for the results to be valuable. Unfortunately, FMEA results are rarely available in time to be of use in this process. Frequently the design has already gone to production before the FMEA results are completed.

Traditionally, the specifications do not designate what technique is required for testability at the board level. The idea of ATE is relatively new and is an answer to the growing complexity of electronic systems. As ATE become more common, so are detailed testability requirements. For example, the Navy now requires equipment to be designed to be testable on ATE called Consolidated Automated Support System(CASS), the Navy standard test equipment (see below under Automated Test Equipment - Army and Navy Standards). The specification for CASS describes what tests are possible. When fault detection and isolation are considered, the designer should be aware of ATE capabilities to assist in the trade-off of the testpoints. Designers will not design a module to be tested by ATE, but will consider the capabilities and limitations of ATE in developing the support strategy and establishing test coverage within the design.

If done properly, fault detection and isolation requirements at the system level will, in large part, define the requirements at lower maintenance levels. This means establishing the scheme for maintenance by the time the board level is reached. The strategy from the beginning of a proper design process is that the testing specified at the system level by the systems engineer will establish the type of testing to be done at the subsystem level. These requirements, in turn, drive the type of testing that will be required at the board level. In other words, the test points and tests selected will be those required to meet the needs of the subsystem plus those necessary to fault isolate the particular board.

Tests supported at the system level may have little impact at the module or board because the software available at the system level is not available at the module level. This can make the module-level fault isolation requirements independent from system-level test considerations. Designer in such a case must trade off the value of the board-level fault isolation versus the cost of adding more hardware to the design. Cost usually wins.

If FMEA or FMECA results are available as input to the test and test point selection process, they allow the testability engineer to put fewer testpoints on more of the reliable circuits and vice versa. The magnitude of this change in the test point mix could be as much as ten percent, allowing the engineer to fine-tune the testpoint selection. In addition, the FMEA data may reveal areas that require real-time testing to prevent catastrophic failure. For example, a sensitive circuit that overheats if current is maintained at a certain level could be

protected by a test circuit which alerts the power supply controller of the overheating condition. The controller could then cut power, thereby maintaining board integrity.

#### **2.1.2.4 Automated Test Equipment -Army and Navy Standards**

The Navy is standardizing on the CASS being built by Martin Marietta in Daytona Beach. CASS provides support for both analog and digital electronic equipment, including radar, navigation gyros, and electro-optic devices. CASS is intended to be a total vertical package; the same instruments and test program sets (TPSs) can be used at certain designated levels of maintenance. (A TPS is the software required to support any interconnect device to the test equipment). A factory element of the CASS system will be developed to enable suppliers of Navy equipment to have a CASS in house. Suppliers may then write TPSs for their own systems, which will support CASS' verticality by providing common software support across maintenance levels.

Similarly, the Army ATE standard is the Intermediate Forward Test Equipment (IFTE) built by Grumman. The objective of IFTE is the same as that of CASS, but IFTE uses a different architecture and implementation and does not offer a total vertical solution. IFTE's components include: 1) a true fixed intermediate system (a rack system), and 2) a base shop test facility (BSTF), which is a van with some complement to the test equipment mounted inside.

#### **2.1.3 Summary**

Designing electronic systems to meet TR&M requirements requires specialists from the beginning of the design effort. To the extent that TR&M engineers are not involved concurrently in the process, TR&M issues are likely to be disregarded or postponed until it is too late to take effective measures.

The design engineer begins the design process by creating a functional block diagram of the design task. Each block is designed in detail and the results are used to generate a detailed schematic and parts list (usually using a CAD system). If parts control specialists are a part of the design team, they help select parts. Concurrent with the detailed electrical design, the mechanical engineer creates a design for packaging the product to meet environmental and maintainability requirements.

Reliability design is driven by specification requirements that commonly include government standards for reliability. MIL-STD-1629 (Procedures for Performing Failure Mode Effects and Criticality Analysis - FMECA) and MIL-HDBK-217 (Reliability Prediction for Electronic Equipment) address reliability. Using a preliminary parts and electrical design produced by the design engineer, the R&M engineer completes a FMEA. This analysis is used to predict the effect of failures on the system operation. Once the FMEA is complete, the R&M engineer adds criticality factors and execute MIL-STD-1629, which is the FMECA procedure to assure adherence to MIL-HDBK-217.

Design for test typically includes practices to enhance testability, circuit partitioning, feed-back loop analysis, and providing test points. The standard for testability analysis is MIL-STD-2165, which prescribes a structured process to evaluate testability performance. The primary testability performance parameters are fault detection coverage, fault isolation, and false alarm percentage. Hardware test point and software test selection on the final product is generally based on five factors:

1. Mechanical accessibility considerations
2. Testability requirements and analyses
3. Hardware/software test design trade-offs
4. Test equipment capabilities
5. Cost of test and repair processes.

Test points are required when part of the system cannot otherwise meet its fault detection/isolation requirements. In the ideal case, a testability engineer aware of the specification's requirements for fault detection and isolation, the FMEA results, the system design, the levels of maintenance required, etc., combines this information into a cogent basis for test point and test selection at the LRU level. Often this is done with the assistance of automated tools for fault coverage prediction.



## **2.2 A Survey Of The Tools That Aid The Design Process**

### **2.2.1 Overview**

This section contains the results of a survey for TR&M circuit analysis tools potentially useful to analog circuit designers.

Included are tool descriptions with brief evaluations, a discussion on concurrent tools, and how tools specific to R&M are used and fit into the general realm of electronic design.

### **2.2.2 Concurrent Engineering And Embedded Tools**

“Concurrent engineering is an approach to design, manufacture, and support that emphasizes interaction and communication among the various agents involved in these processes. The general nature of this approach is to restructure the design process from one geared toward iterative, sequential tasks into one emphasizing incremental, cooperative processes.” [4]

True concurrency has yet to be fully realized in CAD/CAE tools; the single most important aspect of a software product has been its ability to satisfactorily perform its designated function. Only recently have vendors concerned themselves with the first step required to build a concurrent tool: the tool's ease of use and ability to transfer data within and between existing tools.

The ease of data transfer of any software product is directly related to whether it executes on the same platform as other required software. In this case it means that the value of a circuit analysis tool is strongly linked to its availability on the same platform as other CAD/CAE tools. Platforms are now multi-faceted and include windowing software packages and operating system and hardware combinations. Popular examples of windowing packages for engineering workstations include X Windows (including several standards such as Openlook and Motif) and Microsoft Windows. Popular hardware/operating system combination platforms include IBM (DOS), SUN Microsystem (Solaris, Sun OS), VAX/VMS (DCL), etc.

The need for data transfer has led to families of integrated circuit analysis tools offered by a single vendor. The enormous benefit of integrating software tools is clear. When possible, these analysis tools could be integrated with the circuit design tools for a further optimized design flow.

Where complete integration is impossible or impractical, a common platform at least makes inter-machine transfers unnecessary, and also provides a foundation to develop automated format translation programs to patch the data flow. Often such programs require minimal effort to develop, but require the common platform.

A tool with more automated data entry and more flexible file formats is more appealing, as is one that is easy to learn and more user-friendly. These criteria can be used to decide between two tools that have both satisfied the more important requirements.

### **2.2.2.1 Reliability Analysis**

Reliability is defined as "the conditional probability, at a given confidence level, that equipment will perform its intended functions satisfactorily or without failure, i.e. within specified performance limits, at a given age, for a specified length of time, function period, or mission time, when used in the manner and for the purpose intended while operating under the specified application and operation environments with their associated stress levels." [4]

"Reliability engineering provides the theoretical and practical tools whereby the probability and capability of parts, components, products, and systems to perform their required functions in specified environments for the desired period without failure can be specified, designed-in, predicted, tested, and demonstrated, and the results fed back to engineering, manufacturing, quality control, inspection, testing, packaging, shipping, purchasing, receiving, sales, and service for improvements and necessary corrective actions." [3]

Reliability analysis is a critical step in electronic system development because it predicts the proper operation of the final product over its intended life cycle. Prediction of system behavior is necessary to assure that the design meets specifications before manufacturing starts (so that redesign can prevent costly remanufacturing). Reliability analysis is a probabilistic science that predicts system behavior using the known behaviors of the component parts. These behaviors are measured using figures such as the failure rate, the Mean Time Between Failures (MTBF), the Mean Time to Failure (MTTF) or the Mean Time to First Failure (MTTFF), the Mean Time Between Critical Failures (MTBCF), the Mean Time Between Downing Events (MTBDE), the Mean Time Between Demands (MTBD), the Mission Time To Restore Functions (MTTRF), the Mean Time Between Removals (MTBR), and the Mean Time Between Maintenance (MTBM). These terms are defined in MIL-STD-721C, Definition of Terms for Reliability and Maintainability.



Reliability analysis tools provide the MTBF and other reliability figures for a circuit by mathematically combining the established reliability data of the individual components. These tools require established component reliability data. (The connectivity of components is NOT addressed by reliability analysis.)

There are several standards in the reliability industry. The dominant one is MIL-HDBK-217E, "Reliability Prediction of Electronic Equipment." Other DoD standards include MIL-STD-756, "Reliability Modeling Prediction," MIL-STD-785, and "Reliability Program for Systems and Equipments Development and Production." A useful reliability design tutorial can be found in MIL-HDBK-338, "Electronic Reliability Design Handbook."

Where does reliability analysis fit into the general realm of electronic system design? Reliability analysis is required during the validation (or verification) phase of circuit development, after a circuit or subcircuit has been designed. The results are desired as soon as possible for iterative circuit or subcircuit design. The circuit validation phase, including reliability analysis, falls near the end of the system and circuit functional design, but near the beginning of the mechanical assembly design process. This process includes developing the physical implementation of the electronic circuit, including parts selection, board layout, thermal analysis, mechanical assembly design, magnetic effects analysis, test point selection, failure mode mapping, operationalization of failure modes, etc. In short, reliability analysis is required to complete the circuit design, which must be completed before implementation and testing can be accomplished.

### 2.2.2.1.a Reliability Analysis Tools

Almost all reliability analysis tools claim to adhere to, or implement, MIL-HDBK-217E. They include RL-ORACLE, REAP, Viable, Relex, and MIL-HDBK-217E Reliability Predictor (also called PC-Predictor). RL-ORACLE is even mentioned in MIL-HDBK-217E and was exercised recently at the contractor's site in the IBM PC environment. Systems Effectiveness Associates' (SEA's) REAP is frequently and successfully used by Martin Marietta's Automated Systems Department on IBM PCs. SEA also offers REAPmate, a version of REAP for IBM PCs. Management Sciences, Inc. integrates 20 of its PREDICTOR family tools into its CAE TOOL-KIT and includes PC-Predictor, the "MIL-HDBK-217E Reliability Predictor." Innovative Software Designs, Inc. offers its Relex product line, which includes Relex 217, Relex Bellcore, Relex CNET, Relex Mechanical, FMECA, BETAsoft (Thermal Analysis), Risk Spectrum, WeibullSMITH, and PartsCount 217. Cadence integrates its reliability analysis program, Viable, with its thermal analysis tool, Thermax, as well as its various CAD and simulation tools.

A brief description of these tools follows.

### RL-ORACLE [4]

Rome Laboratory's RL-ORACLE (Optimized Reliability and Component Life Estimator) is a Government-Furnished Product(GFP) program that implements MIL-HDBK-217E, "The Reliability Prediction of Electronic Equipment."

The RL-ORACLE program is written in FORTRAN and includes versions for the IBM PC or VAX/VMS. Its outputs are failure rate, MTBF, and availability. It is applied primarily during the validation phase of design.

The inputs to RL-ORACLE include:

- The modules making up the system (a module can be a piece part, a printed board, an equipment, a chassis, etc.)
- The individual piece parts making up the module
- Part application-dependent parameters (stress, environment, temperature, etc.)
- Part-dependent parameters (resistance, capacitance, number of gates, etc.)
- Miscellaneous part failure rates (failure rates of parts not covered by MIL-HDBK-217E)
- Level of indenture description (serial configuration without repair and/or serial parallel configuration with repair) of the electronic system.

The outputs of RL-ORACLE include:

- Detailed piece part listing that contains the value of the parameters used in the failure rate model: constants, pi factors, base failure rates, piece part failure rate, and assumptions made
- Configuration output containing: tabular listing of the pi factors, base failure rate and total failure rate of the individual equipments making up the system; equipment failure rate and MTBF; percent contribution of each piece part type; level of indenture configuration; total system failure rate and MTBF; percent contribution of each part type to the total system failure rate; total number of parts making up the system; percent contribution of each screening level to the total system failure rate; and over stress part flagging

- For reliability model - serial Parallel Configuration with Repair: equipment steady state failure rate; equipment steady state MTBF; equipment availability; equipment mean repair rate; equipment mean time to first failure; all of the above for the total system.

The preprocessing that RL-ORACLE needs includes only data entry of component piece parts list, environment parameters, and the characteristics of components that are not in the libraries. Post-processing might include statistical analysis.

### THERMOSTATS

ThermoSTATS originated with Valid Logic Systems, Inc., which was bought by Cadence Design Systems. It was a software option to the AllegroDesign Engineering System. It conducts thermal, reliability, and noise margin analyses of layouts designed on the Allegro system. It supports comprehensive heat transfer analysis of conduction, convection (both forced and natural), and radiation, mechanisms.

The results of the thermal analysis drive the reliability and subsequent reliability analyses, based on MIL-HDBK-217E guidelines. Both graphical and report data aid the user in pin-pointing high-heat components and MTBF.

ThermoSTATS requires the Allegro Design Engineering System and is included in the Allegro-Engineer configuration.

### THERMAX

Cadence offers the Thermax (once called Amadeus Thermax) tool for 3D and transient heat transfer simulation for printed circuit boards (PCB). Integration with Prance (PCB Component and Placement) provides a complete solution for simulating the thermal behavior of almost any existing PCB module, including double-sided component PCBs and multiple boards, as well as simultaneous simulation of the product environment along with the PCB.

### REAP

REAP is the "Reliability Effectiveness Analysis Program) sold by SEA. It is written in FORTRAN, as are all the SEA programs.

REAP allows engineers to understand how parameters such as junction temperatures, stress, part screening, technology, packaging, and anticipated environment influence product reliability.

## VIABLE

Cadence Design Systems sells an integrated reliability predictor called Viable (originally developed at Valid Logic Systems), which offers both pre- and post-layout analysis to predict the reliability of designs at the component, board, and system level in accordance with MIL-HDBK-217E. Viable also offers Mission Profile Analysis to accurately predict system behavior in real-world complex circumstances.

Programmable reliability equations and flexible formatting capabilities enable users to modify calculation methods to incorporate in-house knowledge. Customized reports can be generated to comply with corporate requirements. With reliability data provided by Viable, users may perform Mission Profile Analyses to accurately predict system behavior in real-world complex circumstances. Viable's user-friendly interface, graphical and text reports, and speedy execution facilitate exhaustive "what-if" reliability analysis.

Viable supports hierarchical designs and operates from an integrated design database.

## PC-PREDICTOR or PREDICTOR MIL-HDBK-217 Reliability Prediction

PC-PREDICTOR or "Predictor MIL-HDBK-217 Reliability Prediction" is offered by Management Sciences, Inc.(MSI) and it derives MTBF estimates from part information in data files. Up to 99 scenarios can be performed in one pass of data. The scenarios for each input can include input for thermal, quality, stress, and environmental conditions. It can access the Predictor Electronic Parts database and recognize more than 20 million parts described in the database.

PC-PREDICTOR requires the Predictor Modeling Language with all its software. The source code is available (FORTRAN).

## RELEX

Relex (or Relex Parts Stress to differentiate it from the new Relex Product Line), offered by Innovative Software Designs, Inc., is a complete MIL-HDBK-217 part stress analysis reliability prediction package. Relex is user-friendly, and it is a complete implementation of MIL-HDBK-217 - "there are no shortcuts or limitations."

Very similar to Relex Parts Stress in the Relex family is Relex Bellcore. This reliability analysis product is based on the handbook from Bell Communications Research (Bellcore) titled "Reliability Prediction Procedure for Electronic Equip-

ment.” The mathematical models in the Bellcore document are in part derived from MIL-HDBK-217. The MIL-HDBK-217 models were enhanced and expanded to create the Bellcore document. Both Relex Parts Stress and Relex Bellcore are offered and distributed by Innovative Software Designs, Inc., which has no connection with Bell Communications Research.

The Relex Product Line offered by Innovative Software Designs, Inc. includes products for thermal analysis (BETAsoft), fault trees (RiskSpectrum), failure mode analysis (Relex FMECA), and Relex Mechanical Reliability Analysis package. The maintainability analysis product is underdevelopment and expected “within several months.”

Relex runs on an IBM PC with DOS version 2.0 or above, 640K of RAM, and a 1.2M hard disk. [7]

### BETASOFT

BETAsoft-R and BETAsoft-S are thermal analysis tools developed by Dynamic Soft Analysis and distributed (integrated) by Innovative Software Designs, Inc. BETAsoft-R assesses the thermal characteristics of PCBs, while BETAsoft-S complements BETAsoft-R by providing thermal analysis for an electronic system. The thermal modeling is 3D, and gives an accuracy of 3° Celsius as validated by wind-tunnel and infrared image tests and by users. It interfaces with Mentor, Valid, P-CAD, PADS-PCB, Autocad, Maxipc, Tango, ORCAD, and generalized.

### Risk Spectrum

Risk Spectrum is a trademark of Relcon Teknik and is distributed by Innovative Software Designs, Inc. It is a PC tool for risk, reliability, and availability assessment. Risk Spectrum is available in three configurations. Risk Spectrum FT provides a complete fault tree analysis package, including state-of-the-art user interface, graphical fault tree editor, flexible output, and powerful calculation capabilities. RiskSpectrum FT+ includes all the features of Risk Spectrum FT, plus an extremely fast minimal cut set calculation module. Risk Spectrum PSA (Probabilistic Safety Assessment) includes all the capabilities of the FT+ versions and more, including a complete set of event tree capabilities.

## **2.2.2.2 Maintainability Analysis**

Maintainability is defined as “the measure of the ability of an item to be retained in or restored to a specified condition when maintenance is performed by

personnel having specified skill levels, using prescribed procedures and resources, at each prescribed level of maintenance and repair." [6], [11]

"This is directly analogous to repairability. The difference is merely that maintainability is based on the total downtime (which includes active repair time, logistics time, and administrative time), while repairability is restricted solely to active repair time. Repairability is defined as the probability that a failed system will be restored to operable condition in a specified active repair time." [11]

According to these definitions, maintainability analysis relies on the results of reliability analysis with the addition of active repair time, logistics time, and administrative time. In other words, reliability is concerned with designing an item to last as long as possible without failure, and maintainability is concerned with designing an item so that failures are easily and rapidly overcome.

Maintainability analysis and its analogous relationship with reliability analysis are summarized on pages 5-38 and 5-39 of MIL-STD-338, "Electronic Reliability Design Handbook."

Some quantitative measures of maintainability include the Mean Time to Repair (MTTR), the Mean Time To Service (MTTS), the Mean Time To Restore System (MTTRS), the Mean Time Between Maintenance (MTBM), the Mean Time Between Maintenance Actions (MTBMA), maintainability, and availability.

The nature of maintainability analysis places it in the verification phase of the development of the electromechanical subassemblies and systems. It combines reliability data from functional subsystems that are partitioned according to assembly structures.

The established standard for maintainability analysis procedures is MIL-HDBK-472, "Maintainability Prediction." Others standards might be MIL-STD-470, "Maintainability Program Requirements (for Systems and equipment)," and MIL-STD-471, "Maintainability Verification/Demonstration/Evaluation."

Because of their commonalities, maintainability analysis and reliability analysis are performed at roughly the same time and dealt with together, as R&M analysis.



### 2.2.2.2.a Maintainability Analysis Tools

There are very few tools that directly address maintainability; only two maintainability analysis tools are addressed in this section: MEAP from SEA, and PC-Maintainability from Management Sciences, Inc.

We used MEAP (SEA) and recognize it as a premier tool in the maintainability realm. MEAP runs on VAX, MicroVAX II, HP/Apollo, Intergraph, or SUN hardware platforms. Its software requirements include VMS, UNIX, or Domain Operating System. As with all Management Sciences, Inc.'s Predictor products, PC-Maintainability is available for IBM PCs, SUN SPARCstations, VAX/VMS computers, RISC 6000 workstations, DEC workstations, HP 9000, and others. The CAD tools that it interfaces with include ORCAD on IBM PCs, and Mentor Graphics and Daisixon SUN.

#### MEAP

MEAP is the Maintainability Effectiveness Analysis Program sold by SEA. It is written in FORTRAN, as are all the SEA programs.

The MEAP tool allows engineers to predict, validate, and study serviceability parameters for complex systems. MEAP offers the unique opportunity to measure the emphasis placed on serviceability at any point during the life cycle. MEAP also helps to specify and manage each task that must be performed to make a system fully operational. MEAP implements MIL-HDBK-472 to compute maintenance times for electronic and electromechanical assemblies, subassemblies, and systems.

#### PC-MAINTAINABILITY

PC-Maintainability, offered by Management Sciences, Inc., automates the maintainability prediction mathematics specified in MIL-HDBK-472. This program evaluates the maintenance tasks related to equipment repair concepts.

This tool is integrated with Management Sciences, Inc.'s other R&M tools in its CAE TOOL-KIT environment.

### 2.2.2.3 FMECA

FMECA is a reliability procedure that includes FMEA (Failure Mode and Effects Analysis) and CA (Criticality Analysis). The FMEA "documents all possible failures in a system design within specified ground rules. It determines, by failure

mode analysis, the effect of each failure on system operation and identifies single failure points, i.e., those failures critical to mission success or crew safety.”[11]

The two most important results of FMEA are 1) identification of single failure points critical to mission success or to crew safety, and 2) a basis to design and locate performance monitoring and fault sensing devices and other built-in ATE.

FMEA differs from fault tree analysis in that FMEA uses inductive logic on the bottoms-up approach that starts with the lowest level in the design (the component level), while fault tree analysis uses deductive logic in atop-down approach. (In FMEA, component failures lead to system performance effects or failures; in fault tree analysis, system failures are traced to possible causes on the component level.)

FMECA is a reliability procedure executed during the design verification phase. It can be viewed as a part of reliability analysis for many purposes. However, for the purpose of an R&M tool survey, specific FMECA capabilities require identification.

### **2.2.2.3.a FMECA Tools**

Available FMECA tools include Relex FMECA package (from Innovative Software, Inc.) and PC-FMEA/FMECA (from Management Sciences, Inc.). No other reliability analysis tools we surveyed claimed FMECA capabilities. Most reliability analysis tools claimed only reliability prediction in accordance with MIL-HDBK-217E

#### **PC-FMEA/FMECA**

PC-FMEA/FMECA, offered by Management Sciences Inc., is an interactive program ideal for the revision considerations inherent in developing an FMECA. It is part of a comprehensive package to automate the application of MIL-STD-1629A, “Procedures for Performing a Failure Mode Effects and Criticality Analysis,” in designing a system.

#### **RELEX FMECA**

Innovative Software Designs, Inc’s Relex Product Line includes a tool called FMECA. It can be used to perform an FMEA, a CA, or both. It can also “perform a Damage Modes and Effects Analysis,” it can “supplement a Safety Analysis,” and it can “evaluate Testability Effectiveness.”



The FMECA tool interfaces easily to the other tools in the Relex family. Outstanding features include full on-line, context-sensitive help, complete pull-down menu control, function key usage, and file management features.

#### 2.2.2.4 Testability

Testability tools include STAT (System Testability Analysis Tool) and GADDS (Generic Adaptive Diagnostic Development System), both from Detex Systems, Inc. The DoD standard for testability analysis is MIL-STD-2165, "Testability Program for Electronic Systems and Equipment."

##### STAT

STAT is intended to evaluate system testability, generate reports appropriate to MIL-STD-2165, and provide optimal test sequences for test program set design. It also provides information on ambiguity group sizes and highlights design features of the system under test that could be modified to improve testability (particularly closed loops).

STAT operates on a functional representation of the System Under Test (SUT), which is usually entered manually after being generated by separate analytical processes (DETEX also alludes to links with CAD/CAE/CASE tool databases, but no information on this feature is available at this writing). The tool is most effective when used at the lowest design level. It was created to work from the bottom up, matching the FMECA approach. Full system representation can be generated by concatenating subsystem models, but the tool is incapable of top-down, functional, or architectural decomposition.

Once the database has been created, the tool permits various diagnostic case studies to compare the cost and benefit attributes of alternate designs and diagnostic strategies. Available reports provide diagnostic figures of merit, design improvement suggestions for the SUT, and optimized testflow charts for translation into test software or procedures.

##### GADDS

Detex Systems, Inc. announced GADDS, a new adjunct tool that reworks the STAT model for a system into a diagnostic model that supports an interactive diagnostic advisory implementation. The diagnostic advisory tool is intended for implementation on a Portable Maintenance Aid (PMA) for use by maintenance personnel.

### 2.2.2.5 Other Related Tools

#### CHEAP

CHEAP is the Cost Effectiveness Analysis Program from SEA. It allows engineers and managers to assess the life-cycle cost of equipment relative to profitability.

#### RAMCAD

RAMCAD is SEA's CAD/CAE/DBMS system interface. It provides a direct, user-definable link between CAD/CAE systems and REAPmate reliability analysis packages. As a peripheral tool, it may be useful in coordinating tools, but is not a main player itself.

### 2.2.3 Hierarchy, Division Of Expertise, Data Flow And Concurrent Engineering

Completing a circuit or subcircuit design initiates the design verification phase. The reliability and maintainability analyses are both performed in this phase. In reality, many iterations of alternating design and verification phases are often needed to optimize an electronic design. Aiding the flow of data between these two phases can reduce design time enormously.

The iterative nature of the various analyses that comprise circuit verification, and the different expertises required, typically combine to require a parallel, iterative effort on the part of at least four cooperating contributors: the circuit designer, the components (parts) engineer, the R&M engineer, and the mechanical engineer.

The components engineer screens the components. The mechanical engineer designs the preliminary mechanical assembly. The R&M engineer coordinates the operational requirements and specifications. The circuit designer takes these results as inputs and designs the electronic circuit.

Reliability analysis requires only identification of the circuit's components and a reliability analysis tool that contains the MIL-HDBK-217E component database. The circuit designer provide this partslist.

The maintainability analysis requires mechanical assembly data as input. This requires the result of the preliminary mechanical assembly developed by the mechanical engineer.

A high degree of communication between these contributors is required during this design verification phase. Concurrent tools aid the necessary data flow. Embedded and integrated tools remove the database boundaries between machines and reduce the possibility of error during data translation from one computer or database to another.

#### **a. R&M Data Feedback**

Once R&M analyses are completed, the results must be fed back to the reliability engineer and the electronic circuit designer. This information is most easily fed back using concurrent or embedded tools that significantly speed the use of this information in optimizing a design for R&M.

The expertise of a reliability engineer is still required to interpret the results of the R&M analysis, to isolate components for substitution and subcircuits for redesign, and to help the electronics designer select superior design alternatives.

With the aid of quicker, more embedded R&M analysis tools, the circuit designer can obtain the R&M results faster, but an expert system is required to reduce the design engineer's need for R&M experience.

### **2.2.4 An Observation On The Growth Of R&M Analyses**

As Michael Johnson points out [2], "The expanding state of the art in computer-aided design and in the overall power of computers and software is making our dreams of infinite attention to detail and concurrent design for R&M possible."

The power of integration is forcing commercial software vendors to adopt open-door policies and open architectures. The future evolution of these policies is unknown, but their benefit to the philosophy of concurrent engineering is already being noticed. The inevitability of concurrent design is assured by its inherent natural and logical essence.

We predict that the availability of integrated and embedded R&M analysis tools will stimulate an increase in the development and use of R&M analysis tools.

We think that software in the planning stage of development should be considered for open architecture techniques that could broaden its applicability.

## 2.2.5 Summary

Of the vendors that we surveyed, four offer complete R&M tool families on a single hardware platform: SEA, Management Sciences, Inc., Innovative Software Designs, Inc., and Cadence.

SEA offers REAP (for reliability analysis), MEAP (for maintainability analysis), CHEAP (for cost analysis), RAMCAD (for interfacing CAD/CAE tools to REAP-mate), and THERMAL (for thermal analysis).

Management Sciences, Inc. offers 20 tools embedded in its CAE TOOL-KIT environment. Some of these include PC-Predictor (for reliability analysis), PC-Maintainability, PC-FMEA/FMECA, PC-Availability, Results and Tree-Master (for fault tree analysis), RMC-1 (for Monte Carlo Simulation), PPCM (for logistics: cost of maintenance), Corida (for logistics: cost of deployment), Bloodhound (for configuration management), BDE (for availability: block diagram evaluation), and FRACAS (for Failure Report and Corrective Action System).

Innovative Software Designs, Inc. offers a number of R&M products, out release of their new maintainability product is still awaited. The degree of integration of their products has not been established, yet, but Relex still appears to be limited to IBM/DOS environments.

The product lines offered by these four vendors, and the R&M tools developed at Rome Laboratory, represent comprehensive or near-comprehensive tool sets. They cover reliability prediction, maintainability prediction, FMEA/FMECA (or fault tree analysis), and thermal analysis. The four commercially-available product lines or families stand out as integrated analysis tool sets that embody the concurrent engineering design philosophy.

Of these four commercially available product lines, Management Sciences, Inc.'s PREDICTOR programs are designed using open architecture techniques. This means that the tools behave similarly on all the platforms that they run on, and that they are designed to interface well with various CAD packages. All of the Management Sciences, Inc. tools in the CAD Tool-Kit run on IBM PCs, SUN workstations, Risc 600 workstations, DEC workstations, HP 9000 workstations. Some of the CAD packages that can be interfaced to include ORCAD on PCs, Mentor Graphics, and Daisix on SUN workstations. Management Sciences, Inc. claims a close relationship with Mentor Graphics, and it seems to have the widest variety of analysis tools and the greatest flexibility of platforms.

SEA's REAP (REAPmate), all the Relex Products, all the Management Sciences, Inc. tools surveyed, and RL-ORACLE are available on IBM PCs.

SUN workstation versions are available for all the SEA tools, Cadence's Viable, and Management Sciences, Inc.'s Predictor MIL-HDBK-217E.

Versions of RL-ORACLE, and all the SEA tools surveyed are available for execution in the VAX/VMS environment.

Cadence claims to integrate its analysis tools with its CAD tools. The Viable reliability analysis tool is integrated with, and shares the design database of, the front-end logic design tool.

## 2.2.6 Conclusions

Selecting reliability and maintainability tools requires goal definition and selection criteria. Goals include procuring tools that satisfy a need for reliability and maintainability analyses. Selection criteria might include

- Inherent capability to meet analysis requirements
- Size and accuracy of component reliability libraries/databases
- Flexibility to substitute or enhance component reliability libraries/databases
- Completeness of a set of tools in meeting a complete set of analysis requirements
- Integration or concurrency of analysis tools with each other and with adequate CAD/CAE tools (or ease of data transfer to CAD/CAE tools)
- Executability on a desired hardware platform
- Ease of use
- Cost.

The more complete families of R&M tools (i.e. SEA, Management Sciences, Inc., Innovative Software Designs, Inc.) address the common-platform and integrated-tool criteria. We have not assessed their capability to perform their intended functions.

We successfully used the REAP and MEAP tools on IBM PCs. The libraries upon which the tools' innate value rests are reported to be "very extensive."

Ideally, complete sets of the R&M analysis tools offered by SEA, Management Sciences, Inc., Innovative Software Designs, Inc., and Cadence should be evaluated in the computing environment of choice by R&M experts to determine their relative usefulness and their interfaces to preferred CAD/CAE tools.

## **2.3 The Testability Process: A Detailed Look**

### **2.3.1 Introduction**

This section documents approaches to the testability process as they apply to engineering tool development. The purpose is to assist in defining requirements for testability design tools. We first describe the emphasis of the section, because it focuses on the design implementation and not the systems engineering or other related aspects. Programmatic testability issues are then discussed, including such topics as requirements definition, fault coverage analysis, critical testability factors, and the overall process. The last topic we describe are the technical issues, including the concept of functional and physical model development.

Testability is an inherent part of all program phases, all levels of a weapon system, and all levels of diagnostics and maintenance. Program phases include concept exploration, demonstration/validation, full-scale development, and operation and maintenance. Weapon system levels include the weapon system itself, system/segment, LRU, and SRU. Diagnostic levels include operator/crew (BIT), organizational, intermediate, depot, and factory.

This section addresses only the detailed design phase of full-scale development, only the lower levels of the weapon system (LRU and SRU), and only factory and depot equipment. Many of the concepts discussed are applicable elsewhere, but the focus is on these elements.

### **2.3.2 Programmatic Testability Issues**

Activities that must occur before detailed design include defining the diagnostic system requirements and identifying the scope of the testability effort. It is assumed that definitions of such terms as fault detection, fault isolation, and false alarms have been established, the levels of maintenance have been identified



(i.e. the maintenance concept), and that there is traceability for the testability requirements to mission or weapon system requirements.

Typical testability requirements are shown in Figure 2. Note that testability levels such as these have never been truly achieved, typically because of lack of technology capability, lack of program commitment, or lack of structured fault coverage analysis and failure mode analysis. Another question is whether achieving these levels of testability is cost- and mission-effective, due to the excessive hardware and software costs associated with such levels. It is always necessary to analyze the testability requirements for applicability to the target maintenance system, and to optimize operational availability and life cycle cost.

<u>Maintenance Level</u>	<u>Fault Detection</u>	<u>Fault Isolation</u>	<u>False Alarms</u>
User	90%	Product Level	< 5%
On-Site Repair	95%	1 LRU	80%
		2 LRU	90%
		3 or more	95%
Shop Level	95%	1 LRU	80%
		2 LRU	90%
		3 or more	95%
Depot Level	95%	1 LRU	80%
		2 LRU	90%
		3 or more	95%

*Figure 2 Typical testability requirements summary.*

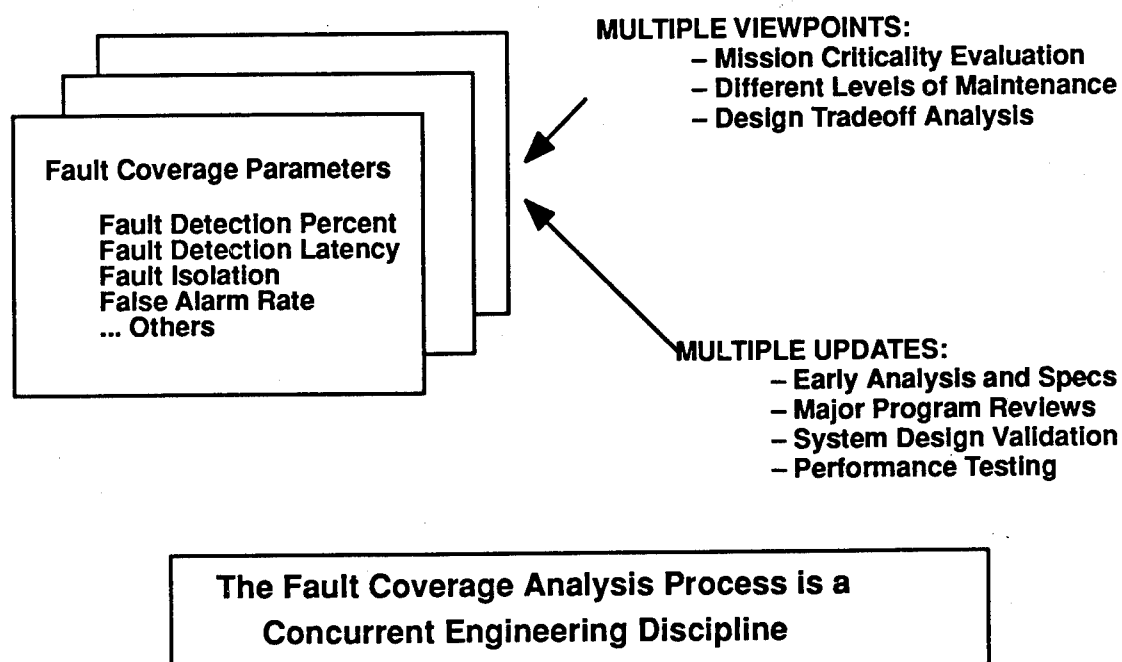
### 2.3.3 Fault Coverage Analysis

Fault coverage analysis is critical to every testability program because it represents the data upon which trade-offs are performed, and ultimately upon which hardware and software is designed.

Figure 3 shows the many viewpoints associated with the data from fault coverage analysis.

The parameters associated with fault coverage analysis include the classic testability figures of merit. Fault coverage percent is the total number of detectable faults divided by the total faults which occur. Detection latency is the time from fault occurrence to the time that the testing system declares that the fault has occurred. Fault isolation is a measure of how accurately and effectively the system can identify the faulty element, once a detection has been made. False

alarm rate has many definitions, none of which are totally verifiable. Note that these parameters are applicable whether the analysis is performed for on-board BIT or off-board ATE. They are normally defined by either the specification, which will determine the requirements, or by the contract Statement of Work, which will determine the type of analysis to be performed.



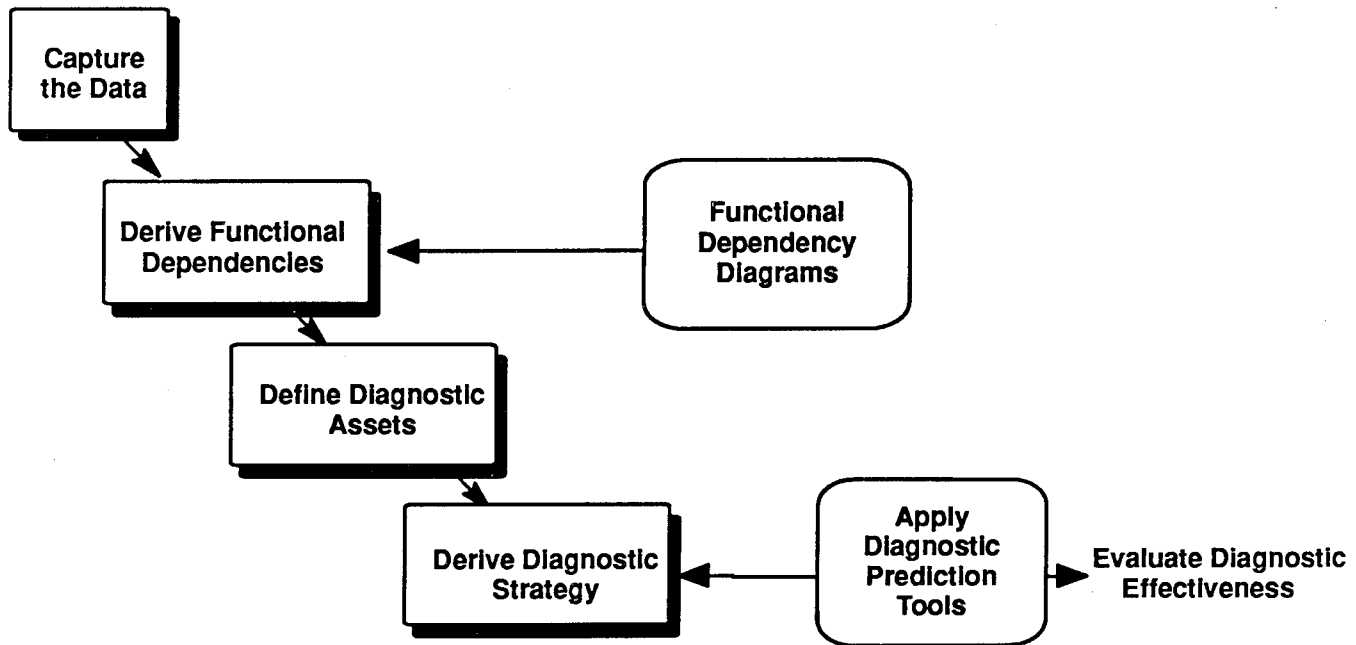
*Figure 3. Fault coverage analysis.*

There are normally many updates to a fault coverage analysis, depending upon the contractual requirements and the level of design concurrency. Initial fault coverage analysis is normally required by contract to be achieved by PDR. For requirements analysis purposes, however, it is necessary to perform some level of analysis by SRR. The most important analysis to the contractor is that performed for system validation, since this will determine the system's adherence to the requirements. However, additional analysis may also be performed when abnormalities occur long after the system is fielded. If these analyses are performed without regard for one another, they may be inconsistent and repetitive. The goal, therefore, is to keep the analyses based upon the same data.



### 2.3.4 Diagnostic Development Process

The diagnostic development process is shown in Figure 4. It shows the major elements of diagnostic development that are performed regardless of the type of system to be tested and the current phase of the program.



*Figure 4. Diagnostic development process.*

The initial process step is to capture the available data, which is often physical data such as schematics and assembly diagrams, or functional data such as specifications and software performance documentation. From this data, which is usually incomplete, the functional dependencies are derived. Functional dependency diagrams are used often, but the quality and structure of these diagrams varies from engineer to engineer. The purpose of the dependency diagrams is to define the functions of the system, determine how they affect the overall functions and mission of the system, and derive the relationships among the functions. If the physical elements of the functions may be defined, and the functional and physical assets mapped, then the associated fault isolation analysis may be performed.

The diagnostic assets may be defined after the functional dependencies are derived. There may be built-in assets, or other available stimulus, measurement, and computing assets which may be assumed to be available. If not, additional assets must be required.

The diagnostic strategy development is where the engineer applies the assets to the functional system relationships to perform the required diagnostic functions. The strategy includes the general types of tests to be performed, and the required for fault detection, fault isolation, and false alarm reduction.

From this strategy, the engineer applies diagnostic prediction tools to evaluate performance. The tools range from a full model to simple matrices defining tests versus elements. If the proposed strategy meets the requirements with an acceptable cost and mission effectiveness, the detailed design process will follow. Otherwise, the process will be iterated.

### **2.3.5 Technical Testability Issues**

The discussion of technical testability issues first describes the fault coverage database previously referenced. This is followed by a discussion of the multiple viewpoints of this diagnostic data, and then a description of the functional and physical model development process.

### **2.3.6 Typical Fault Coverage Database**

Figure 5 is a typical fault coverage database, used with a recent full scale development program. It was initially used to determine if the system met its testability requirements, then later to define the software and hardware that would perform the test. The data was kept on a relatively simple spreadsheet program.

The first column of the database is a top-level list of the functional units. Note that some deviation from strict functional decomposition is allowed, but there are functional descriptions of these entities that are not part of this database. All the elements break down into lower elements, but only a breakout of the VME master board is shown here for clarity. The allocated reliability figures were initially used for the Failures Per Million Hours (FPMH) values, but later in the program the actual predicted numbers were inserted. The use factor is a percentage of how much the associated function is used, since there were functions in this system that had to be present although they were not used in this version. Therefore, testability coverage of them was not required.

There were three modes of BIT in this system; the first was called Self Test. The percentages of coverage were input, based on the type and quality of the associated tests, and the FD coverage was calculated as a product of the coverage and reliability in FPMH. A different page of the spreadsheet was used for each mode

of BIT, identifying the software and hardware tests that were applied to each functional entity. The aggregate self test coverage is summed at the bottom.

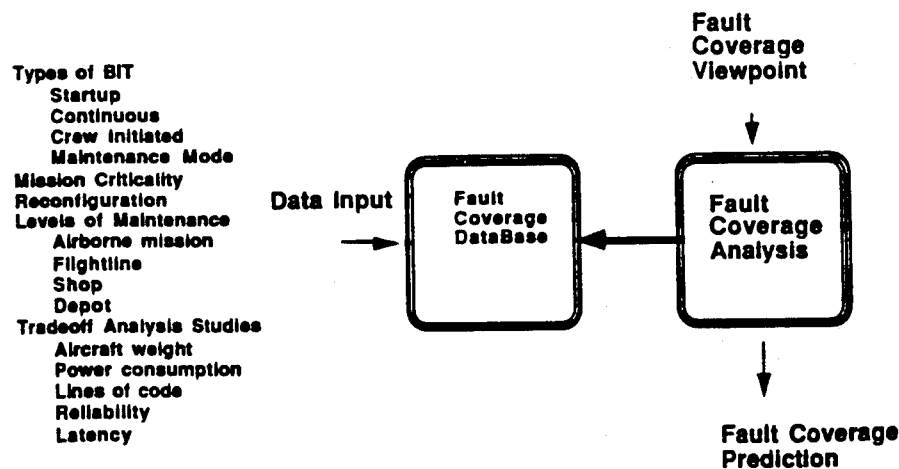
FUNCTIONAL UNITS	REL FPMH	USE FACT BASIS	REL FACT BASIS	SELF-TEST		INITIATED-BUILT-IN-TEST				BIT SUM	FPMH NOT COVER
				% COV.	FD COV.	NON-INTACT. BASIC	INTACT. DELTA				
VME Master Board	4.0	1.0	4.0	50%	2.0	98%	3.8	3%	0.1	3.9	0.08
1553 Bus Interface	0.5	1.0	0.5	50%	0.3	0%	0.0	90%	0.5	0.5	0.05
1553 Transmitter #1	0.5	1.0	0.5	50%	0.3	0%	0.0	90%	0.5	0.5	0.05
1553 Transmitter #2	1.0	1.0	1.0	50%	0.5	99%	1.0			1.0	0.01
Microprocessor	2.6	1.0	2.6	50%	1.3	99%	2.6			2.6	0.03
RAM Memory	2.9	1.0	2.9	50%	1.5	99%	2.9			2.9	0.03
EEPROM Memory	2.9	1.0	2.9	50%	1.5	99%	2.9			2.9	0.03
PROM Memory	2.0	1.0	2.0	20%	0.4	99%	2.0			2.0	0.02
Timer/Counter	0.7	1.0	0.7	30%	0.2	99%	0.7			0.7	0.01
Clock & Reset	1.1	1.0	1.1	0%	0.0	99%	1.1			1.1	0.01
QUART	1.1	0.5	0.6	0%	0.0	0%	0.0			0.0	0.55
RS232 Drivers	1.2	1.0	1.2	50%	0.6	98%	1.2			1.2	0.02
VMEbus Interface	0.5	1.0	0.5	20%	0.1	50%	0.3	30%	0.2	0.4	0.10
PC & Connectors											
Board Totals →	21.0		20.5		8.5		18.3		1.2	19.5	0.99
Graphics Board	28.0		28.0		5.9		16.6		8.5	25.1	0.87
Panel Interface Board	25.0		25.0		5.7		15.6		8.6	24.2	0.81
Power Supply	20.0		18.0		13.2		16.9		0.3	17.2	1.29
Display Assy	50.0		48.0		25.0		35.0		14.5	49.5	0.50
Panel	25.0		25.0		0.0		7.5		17.3	24.8	0.25
Backplane	5.0		5.0		2.5		4.0		0.5	4.5	0.50
Harness & Connectors	10.0		10.0		1.0		2.0		7.0	9.0	1.00
Total Assy →	184.0		179.5		61.8		115.9		57.9	173.8	6.21
Fault Coverage →											
						34%	65%	97%			
						Self Test	Non-In BIT	Total BIT			

Figure 5. Typical fault coverage analysis database.

The next columns perform similar analysis for the other modes of BIT, and delta coverages and percentages not covered are also displayed.

### 2.3.7 Multiple Viewpoints of Diagnostic Data

As mentioned earlier, there are many different reasons to perform fault coverage analysis. A number of them are delineated in Figure 6, which suggests that a common database is useful in the many analyses which are performed. For example, the same data is required for both a coverage analysis and the BIT software definition, so it makes sense to share the data. Additionally, design changes are made often in a concurrent engineering environment, and all members of the engineering team must have insight into the effect of such changes.



*Figure 6. Multiple viewpoints of diagnostic data.*

Specific types of data include the different modes of BIT, the relative criticality of the functions, and the levels of maintenance. The implementation tradeoffs that must be performed include all those to optimize the testability implementation with respect to other system parameters, such as flyaway cost, life cycle cost, reliability, weight, and system performance. A consistent set of valid data is imperative for such analyses.

### 2.3.8 Functional and Physical Model Development

At the core of the diagnostic system design is developing the functional and physical models. This is not to imply that a model-based expert system will always be applied to the process, but there is always some level of modeling, even if it is only performed within the designer's head.

In general, the testability analysis often has to determine if a set of tests, which evaluate functional performance, can detect and/or isolate failure conditions at the physical level.

Early in the program development, the functional definition of the system may be the only well-defined model and must be the primary basis for testability analysis. Later, the physical model will be developed and analysis will extend to the component level.

The process flow for functional and physical model development is shown in Figure 7. The process is similar to functional decomposition, and is often only performed to the level required by the current phase of the effort.

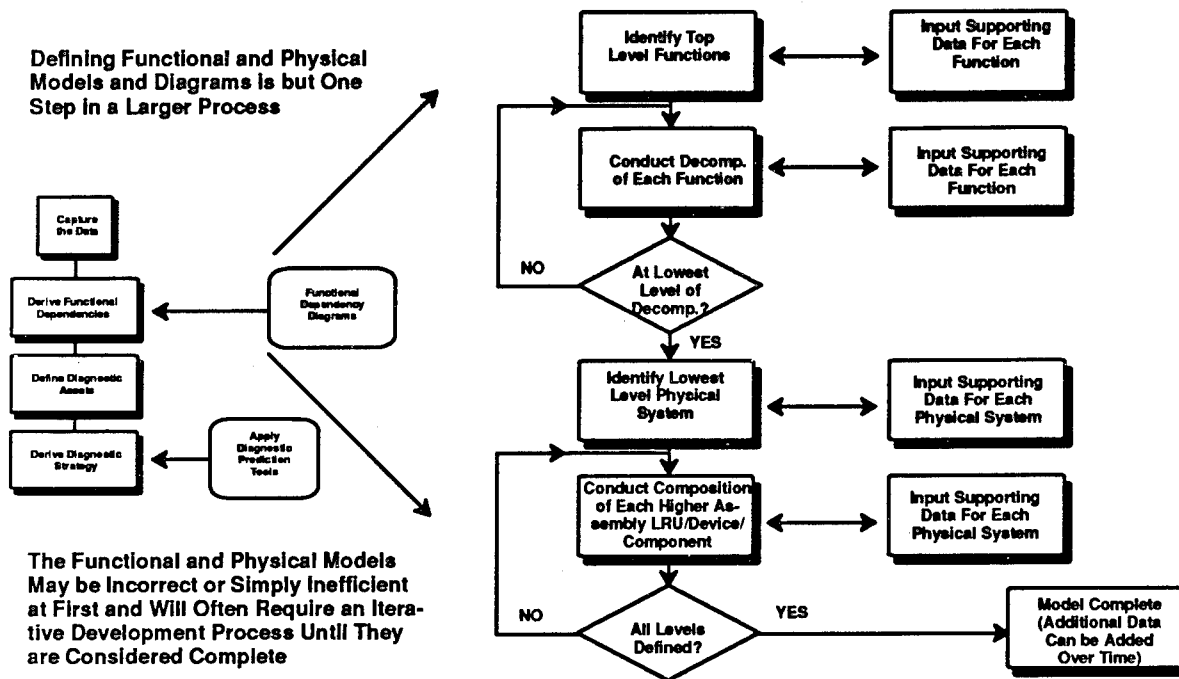


Figure 7. Functional and physical model development.

The functional and physical models may be incorrect or simply inefficient at first, and will often require an iterative development process until they are considered complete.

The ultimate goal is to determine if a fault can be properly isolated, and it is often necessary to be able to measure a symptom which uniquely defines the fault as existing within a physical element. Alternatively, deductive reasoning may be used to fault isolate if a functional failure is detected, and all other physical components measure healthy.

### 2.3.9 Testability in an Integrated Product Team Environment

Many development programs follow a philosophy of Integrated Product Teams, in which all personnel associated with the program, including the military, government, prime contractor, and subcontractor, are all associated with a particular product. Sometimes identified as an aspect of Total Quality Management or

concurrent engineering, this practice is used on programs such as the F-22 Advanced Tactical Fighter.

In such an environment, it is imperative that the different teams follow consistent approaches to testability implementation. A working group committee should be chartered to establish best practices for fault coverage analysis, and these practices should be coordinated through working group meetings. Guidelines, standards, and practices may all be published for consistent communication among the teams. Existing tools should be updated for fault coverage analysis and BIT/TPS implementation, and then disseminated and taught.

One of the difficulties in implementing a testability program is that many program terms may be interpreted differently, which results in inconsistent implementations. The need for common practices becomes evident when different vendors implement their analysis differently. If standards are not followed, there is no way to verify or validate vendor claims, no way to correlate different vendor analysis packages, and no way to apply the vendor data to different viewpoints. The result is that no valid tradeoffs can be applied, and suboptimal implementations may be captured.

### **2.3.9.1 R&M Functions in a CAD/CAE Environment**

Figure 8 depicts functions that should be performed automatically or semi-automatically in a CAD/CAE environment.

Currently, there is no software to implement the functions shaded in the diagram. For example, Mentor Graphics V8.1 implements hierarchical design and schematic capture, there are associated parts lists and analysis tools such as Analogy's Saber. There are also reliability databases, such as Rome Laboratory's ORACLE, which are available for component reliability analysis, and may be more complete if the thermal analysis has been performed.

The problem is a classic dilemma. Testability analysis requires some amount of reliability analysis to determine failure modes on a failure-rate weighted basis. However, full MIL-HDBK-217 analysis cannot be performed until the piece part design is complete. By this time in the product development process, it is too late to modify the design to implement the proper testability characteristics.

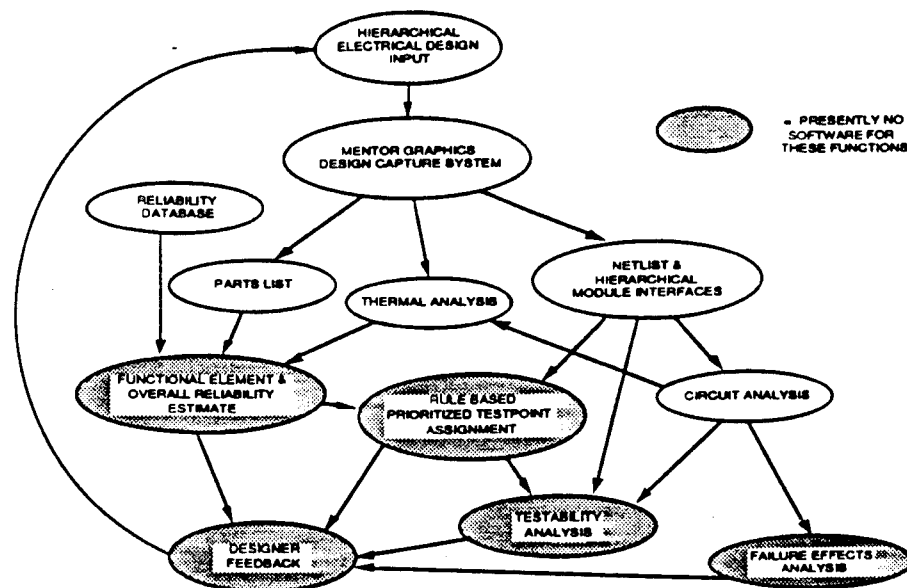


Figure 8. R&M and design CAD/CAE functions.

Currently, the type of analysis depends upon the program phase. Early in the design cycle, the approach is functional and top-down, while a physical bottoms-up analysis is performed during the detailed design. Unfortunately, there are no tools that support both phases.

The ultimate solution to the dilemma lies in methodology as well as tool improvements. If a tool assists in performing functional reliability estimates when the design exists only at the functional hierarchical level, then an initial allocation and characterization of failures may be obtained. Rule-based, prioritized test point assignment can be executed, and aspects of the testability analysis can be automated. The specific task that can be automated through simulation is failure effects analysis. The most important aspect of this capability, however, is the real-time designer feedback. As the designer conceives the design and enters it into the system, testability figures of merit can be attained, and this can improve the product before it is even breadboarded.

### 2.3.10 Summary

The testability process pervades all product phases. There are adverse impacts on life cycle cost and operational availability if a structured process is not followed. There are relevant advances in the CAD/CAE technology that facilitate



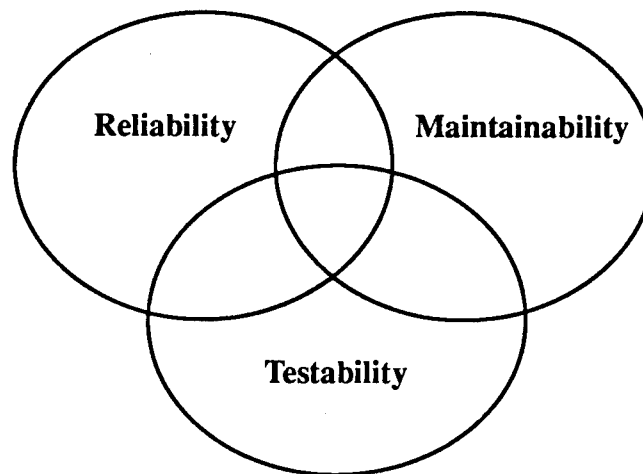
testability in a concurrent engineering environment, but succinct definition of both the process and the technology is required to successfully develop tools and implement methodology.

### 3. System Concepts and Methodologies

The R&M Design Expert system combines testability, reliability, and maintainability (TR&M) into a single working environment. During the first year of the program we developed a methodology to combine these three design areas. The Supportability Analysis section examines each of the areas in the Venn diagrams. The Concept: The R&M Design Expert System section details our approach to integrating these ideas into the system.

#### 3.1 Supportability Analysis

During phase I of the R&M Design Expert System development, we developed a unique methodology to combine TR&M. Figure 9 is are presentation of this methodology.



*Figure 9. Design methodology for testability, reliability, and maintainability.*

The basic data needed for logistics metrics are reliability values for components, knowing how to test components, and knowing how to access and maintain parts and assemblies. Knowing the time required to test a component is also valuable. If all this data is obtained and integrated into the system database, the various metrics can be calculated automatically.

When the data is collected and the metrics generated the information, along with the design data, is exported to specialized databases. The system design database can use data from lower level designs to toll up metrics to become system metrics. The logistics database can use the design data for initial calculations of how to stock replacement parts. Also, the design data can be imported into the maintenance database to begin prognostic calculations that determine when preventative maintenance should be performed, as well as the most likely causes of a failure.

Automatically determining the desired metrics begins by obtaining raw data for TR&M. This data may be extracted from databases, input for specific instances, or generated from rules.

To generateilities metrics requires data about the particular metric at the lowest level of assembly possible. Also needed is data about how the system is constructed, both in a physical and functional sense.

System construction begins at the component level. Each component belongs to a leaf function and a leaf assembly. These are the lowest levels of functions and assemblies of interest to the designers. The leaf functions of a system are of sets of components that have no common elements. The same is true of leaf assemblies. Leaf functions belong to one or more branch functions. Leaf assemblies belong to a single branch assembly. Branch function may also belong to one or more branch functions, while branch assemblies may belong to another branch assembly. Alternatively, branch functions may belong to one or more top-level functions; while branch assemblies may belong to a top-level assembly. These are the system level functions and assemblies. It is possible for leaf functions and leaf assemblies to directly belong to top-level functions and top-level assemblies.

The pressures of short development schedules, low-cost development, and concurrent engineering combine so that designs must work when first implemented. The requirements of known life cycle costs and customer satisfaction require that reliability and maintainability be design into a system. In addition, military procurements often specify supportability metrics that must be met.

Designers must accurately determine the logistical cost of an initial design. This can be accomplished throughout the generation of metrics that grade a design. Combining individual supportability metrics will provide designers with information on how each part, assembly, or function affects the entire system. These metrics, applied at the sub-assembly and sub-function level, allow logisticians better insight into how to support the system once it has been built.

At the moment, designers can use drawing and simulation tools to aid design. However, these supply no information as to how the system will perform over its lifespan with regard to cost or availability. Designers can calculate the Mean Time Between Failure (MTBF) for a system using reliability values from data sheets supplied by component manufacturers, but the process is long and repetitive. There are no standard tools to determine the how well a system may be tested or how easily it may be maintained. Both these processes involve examining a system or assembly after it has been designed, using a variety of techniques that do not lend themselves to using common data.

Tools must be developed that determine a system's metrics using a common set of data. This ensures that different metrics are not determined using people's conflicting ideas. It also allows the full implications of a design change to be known by generating all the metrics changes.

### 3.1.1 Reliability

Generating the desired metrics automatically, during the design cycle, is both possible and desirable. However, the design process must be disciplined. The board designers, system designers, test software designer, and ILS designers must examine the same data, and must adjust their design quickly to respond to changes.

An example of this hierarchy is the rotor on an internal combustion engine. The rotor is the sole component of the rotor leaf assembly. This leaf assembly is a member of the distributor branch assembly. The distributor is a member of the engine branch assembly, which is a member of the chassis top-level assembly of a vehicle. The rotor component is also a member of the spark generation leaf function. This leaf function is a member of the ignition branch function, which is a member of the electrical system top-level function of the vehicle.

The criticality of functions must be specified to determine metrics on critical items.

Obtaining the initial data to establish reliability metrics is simple, though time consuming. The reliability metrics for each component must be entered into a database for the system being designed. This data can be obtained from the manufacturers data sheets, data entered in other databases, computer models of the component that generates equations, and experience. In the case of models, the metrics must be generated for selected environments. This data will be in the form of particular values for each component, assuming normal operating conditions. Reliability values for cables, connectors, printed circuit boards, etc.

also need to be obtained. These values will either be the MTBF or Failures Per Million Hours (FPMH) for the component. MTBF values will eventually be converted to FPMH.

### 3.1.2 Testability

The data needed to obtain a testability metric of functions and assemblies requires the engineer to know all the failure modes (malfunction) of the system, and the failure that each test will detect.

Each leaf function is examined to determine all the ways it may fail. The probability that the failure of a leaf function is caused by each individual malfunction is then associated with thermal function.

The programs that will test the system are specified and the set of malfunctions indicated or each test result are listed. This data provides information about how well a malfunction can be isolated to a single assembly, or set of assemblies; as well as which test must be run to isolate to a given numerical ambiguity group. At initial stages of development, this involves creating an architecture of the test programs that will be developed. As the design progresses, programs and sub-programs can be added to, and removed from, the architecture based upon the metrics produced for testability. The time it takes for each program to execute to the points where a result is obtained is listed.

### 3.1.3 Maintainability Metric

The maintainability metric, as defined here, describes the system's ability to be repaired. Since maintainability metrics that incorporate test times will be discussed later, the metric data described here deals only with assemblies, parts, and repair times.

The ability to repair an assembly can be based upon many things. The most useful is experience with a fielded assembly. Another way to determine repair times is reading a handbook that tells the time it should take to perform all the mechanical operations necessary to repair an assembly.

Other information required to generate maintainability metrics is an assembly hierarchy, and the tools, facilities, test equipment, and abilities to repair the assembly. The times associated with repairing an assembly will be different when given different mixes from the described categories. Furthermore, the type of

maintenance performed upon an assembly must be noted as well (i.e. preventative maintenance, corrective maintenance etc.).

### 3.1.4 Reliability Metric

The easiest metric that may be determined during design is reliability. These values can be determined for both functions and assemblies. By generating values at a low level, designers can determine where reliability is a problem, given the mission of the system; and where reliability may be sacrificed to lower the initial system cost.

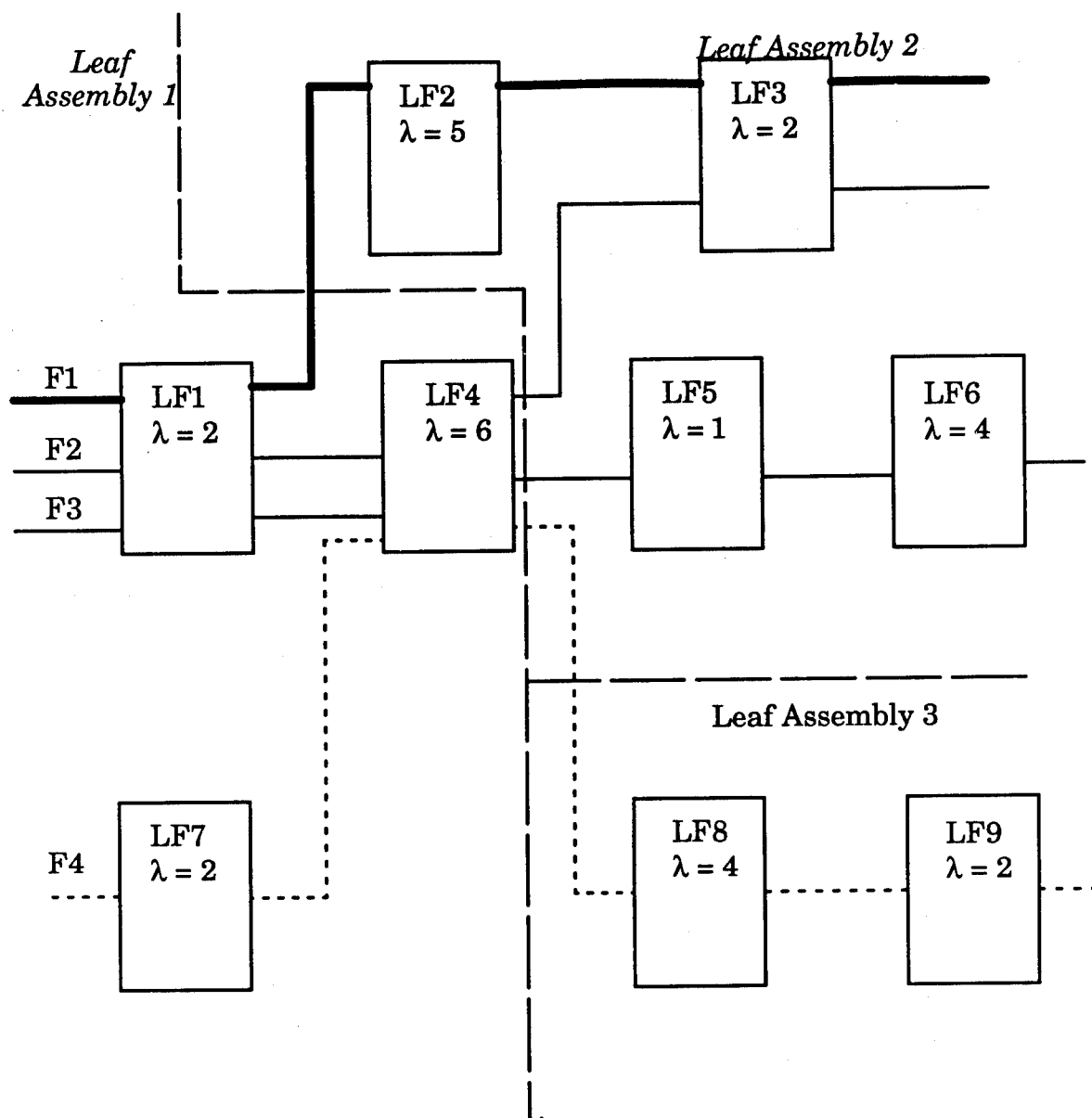
The reliability metric is generated by adding the FPMH of all the components within each lowest level function and assembly to determine the FPMH of the particular function or assembly. These values can be added to determine the FPMH for any level function or assembly, as well as for critical functions.

An example of a set of functions, sub-functions, and sub-assemblies is shown in Figure 10 to illustrate the process of determining the reliability of a system or its parts. The module shown is made up of three sub-assemblies containing nine sub-functions, and it performs four functions. The reliability of the components of each sub-function is shown as 1 (FPMH). The reliability of the entire module is 29 failures per million hours or a MTBF of 34,483 hours. The  $\lambda$  of assembly 1 is 11, assembly 2 is 12, and assembly 3 is 6. The  $\lambda$  of function 1 is 11, function 2 is 14, function 3 is 10 and function 4 is 14.

### 3.1.5 Testability Metric

The testability of an assembly during production depends upon the test equipment and software used. Given the current state of technology, production testing techniques allow almost 100 percent testing of individual assemblies. However, we will address the testing performed after production.

To determine the testability of a function or assembly, the ability to test each way the function or assembly may fail is examined. Testability is calculated by assuming a certain set of test and calculating what percentage of the possible faults of a function or assembly can be detected by the tests. Since the number of possible faults is different for each function or assembly, this value must be calculated separately for the testability of any selected group of circuits, such as the critical functions of the system; otherwise a sub-function or sub-assembly with a low number of possible faults would skew the results.



*Figure 10 Reliability Example*

The metrics describe may be generated for the ability to isolate a fault to an ambiguity group where the number of assemblies/sub-assemblies is determined by the person generating the metric.



Further metrics that may be determined are the Mean Time To Test by determining the time required to diagnose each fault that may be detected and determining the mean. Graphs may be produced relating test accuracy (number of ambiguity groups) to test time.

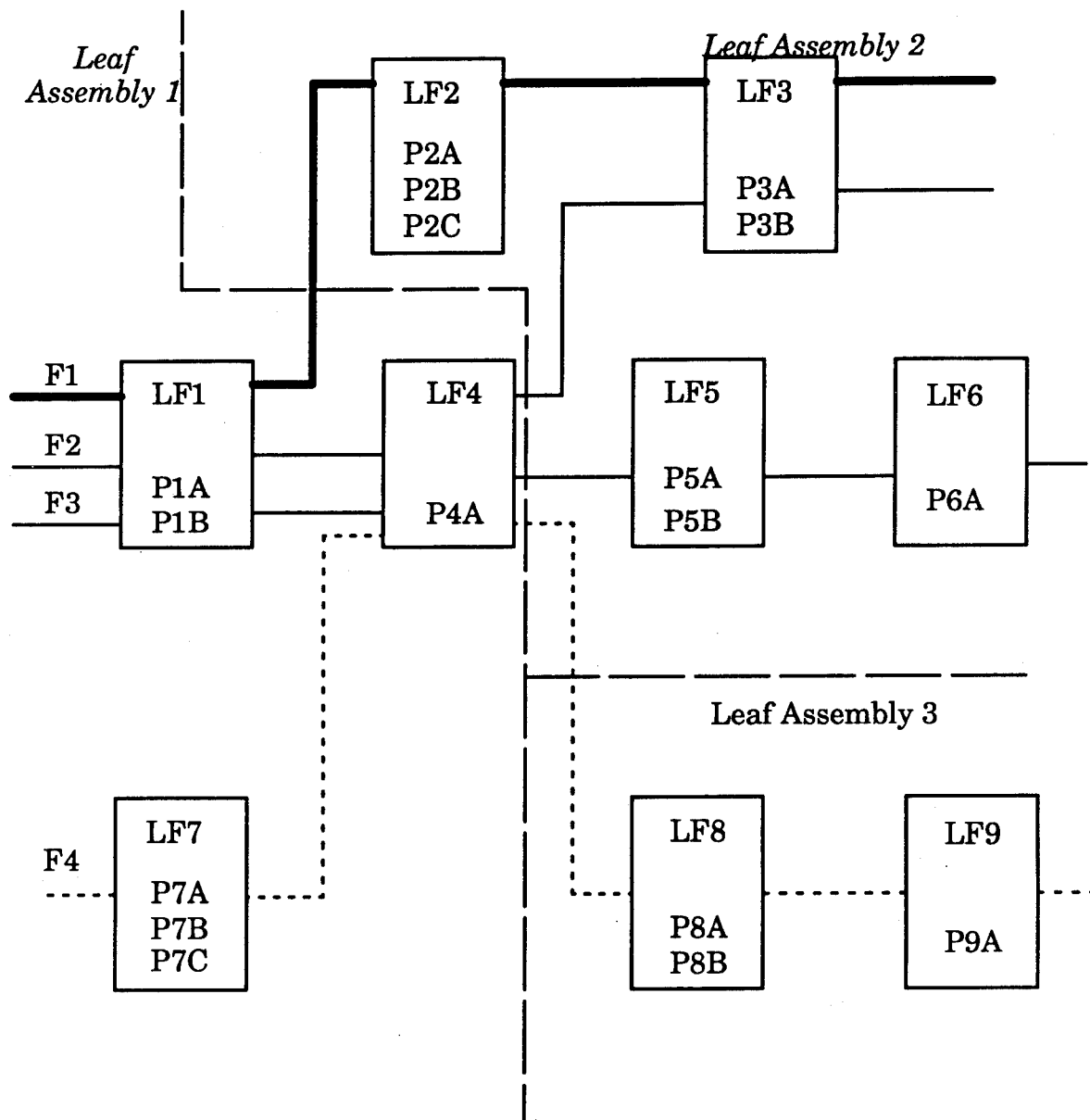


Figure 11. Testability Example

Figure 11 shows the same module as Figure 10 ,along with the complete set of problems (faults) that could occur in each sub-function. Four tests are described in Table 1., along with the problems that each test will find.

Test	T1	T2	T3	T4
P1A	X	X	X	
P1B	X			
P2A			X	
P2B			X	
P2C			X	
P3A			X	
P3B	X			
P4A	X	X		X
P5A			X	
P5B			X	
P6A			X	
P7A				X
P7B				X
P7C				X
P8A				X
P8B				X
P9A				X

*Table 1. Test coverage.*

This table shows the test coverage for different sets of tests. Test T2 and T4 cover 11 out of 17 problems (65%); tests T1,T2, and T4 provide 82% coverage, tests T1 – T3 provide 65% coverage, and a set of all these tests provides full coverage.

### 3.1.6 Maintainability Metric

The primary maintainability metric is Mean Time To Repair (MTTR). This is calculated by determining the time to repair each replaceable system assembly and dividing by the number of replaceable assemblies. The MTTR can also be generated for selected parts of a system, such as replaceable assemblies or specific functions. The time to repair/replace each assembly is determined by adding the

time required to perform each step when removing/replacing the assembly. This data must come from the mechanical engineers' system design.

Using the previous examples, and assuming that the module takes no time to access, we can invent the following procedures for maintaining the sub-assemblies:

#### Sub-assembly 1

Loosen and open door	5.0 minutes	Total 7 minutes
Remove three screws	1.5 minutes	
Extract board	0.5 minutes	

#### Sub-assembly 2

Remove 4 screws and cover	2.0 minutes	Total 3.5 minutes
Remove 2 screws	1.0 minute	
Extract board	0.5 minute	

#### Sub-assembly 3

Open hatch	5.0 minutes	Total 6 minutes
Extract board	1.0 minute	

To replace the sub-assemblies, the total times must be doubled to 14, 7, and 12 minutes, respectively. These values lead to a MTTR of 11 minutes for the module. The MTTR of functions 1 – 2 (sub-assemblies 1 and 2) is 10.5 minutes. The MTTR of function 4 (sub-assemblies 1 and 3) is 9.5 minutes.

Combinations of the data obtained for testability, reliability, and maintainability will produce metrics that provide more important information than those describe above. These metrics include reliability weighted metrics, which make parts with a higher likelihood of failure more important when calculating metrics. Relationships between the 'ilities may be exploited to determine the optimum way to proceed with the system design or the logistics planning.

### 3.1.7 Reliability/Testability Combined Metric

One of the most important metrics that can be determined by examining the relationship 'ilities is the failure rate weighted testability metric. This metric provides designers with a measure of how important tests are for particular parts of a system. The metric highlights the areas of the system that have a high likelihood of failure, where more effort would be justified in creating tests or adding hardware to make testing easier. The metric also highlights where addi-

tional work should not be expended because of the high reliability of the hardware. This metric helps select postulated tests to be implemented.

Calculating the failure rate weighted testability is done by determining the FPMH of each possible fault in a function or assembly that is tested, adding them up, and dividing by the FPMH of the entire function or assembly.

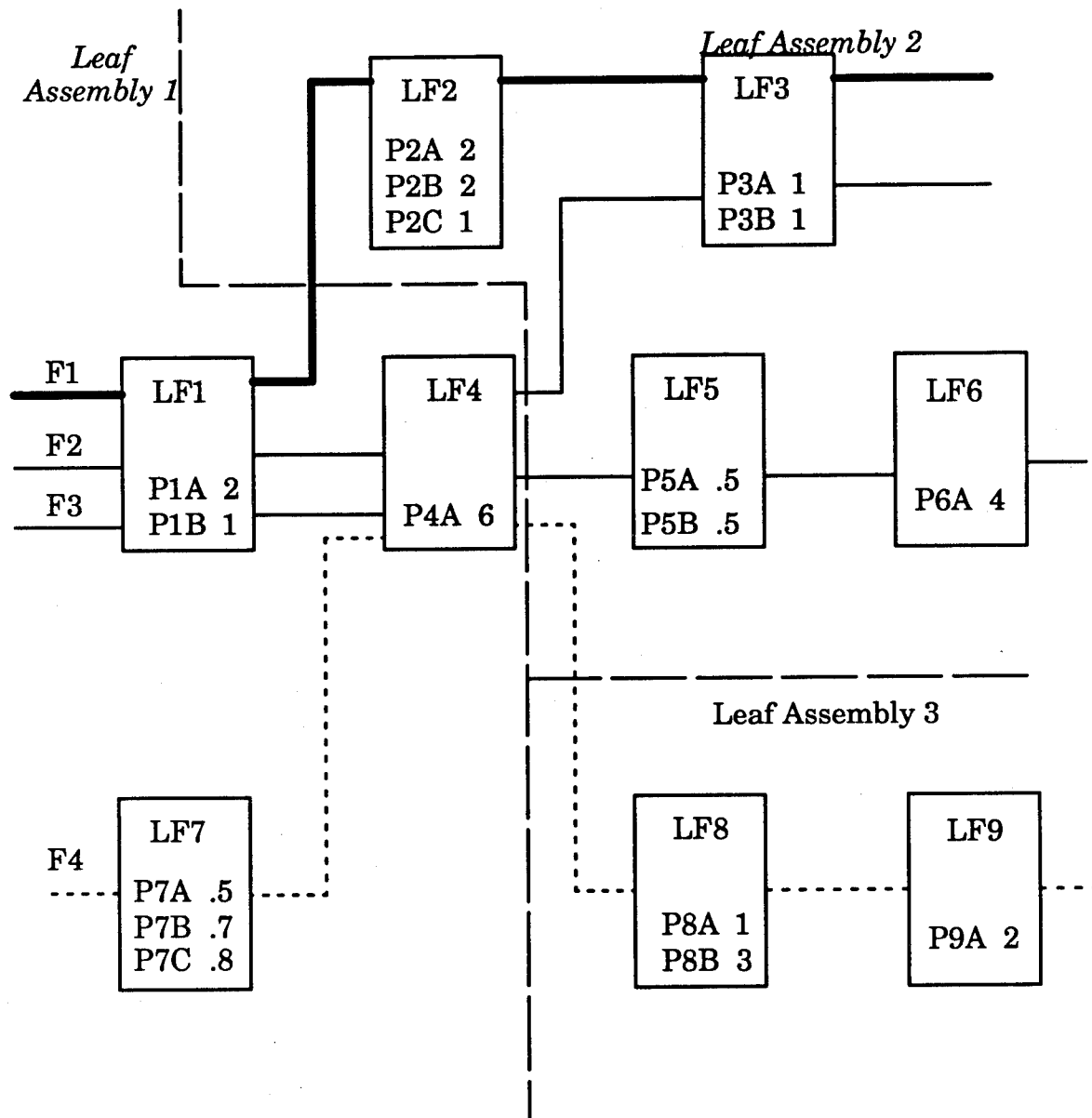


Figure 12 Failure rate weighted testability example.

This method is shown in Figure 12. The problems(faults) in each sub-function are listed along with the FPMH of each possible problem. Using the test described previously, we can substitute the  $\lambda$  value for each indication of coverage in the test coverage table. Tests T2 and T4 provide 72% coverage (21/29); tests T1, T2 and T4 provide 62% coverage; tests T1, T2 and T3 provide 72% coverage, and tests T1, T2, T3 and T4 provide 100% coverage. Values correspond with values of 65%, 82%, 65%, and 100% calculated without relying on the failure rate of the faults. Table 2. illustrate this.

Test	T1	T2	T3	T4
P1A	2	2	2	
P1B	1			
P2A			2	
P2B			2	
P2C			1	
P3A			1	
P3B	1			
P4A	6	6		6
P5A		.5		
P5B		.5		
P6A		4		
P7A				.5
P7B				.7
P7C				.8
P8A				1
P8B				3
P9A				2

Table 2.. Test coverage with  $\lambda$  values.

### 3.1.8 Reliability/Maintainability Metric

The metric formed by combining reliability and maintainability data allows decisions to be made regarding maintainability using a better metric than just the reliability metric. Adding the reliability component permits designers to design a system where the assemblies that need the most maintenance will be easier to maintain. It also provides designers of the logistics plan a better insight into

how to place supplies, maintenance equipment, and personnel to enable maximum efficiency in supporting the system.

The failure rate weighted maintainability metrics calculated by determining the FPMH of all faults in a replaceable assembly and multiplying by time to repair the assembly. Table 3. illustrates this using data from the reliability and maintainability examples.

Assembly	FPMH	Mean Time to Repair	Failure Rate Weighted Maintainability
Leaf Assembly 1	11	7 min.	77 min./million hrs
Leaf Assembly 2	12	3.5 min.	42 min./million hrs
Leaf Assembly 3	6	6 min.	36 min./million hrs

*Table 3. Failure rate weighted maintainability.*

### 3.1.9 Testability/Maintainability Metric

The metric formed combining testability and maintainability data provides the MTTR metric.

When testability and maintainability capabilities are combined, the results is an analysis of the relative ease with which a diagnosable element may be replaced. For example, such a metric can evaluate the certainty with which an easily-replaceable unit may be determined to be faulty. Such an analysis is also the basis of the "repair vs. dispose" analysis performed during module development.

These analyses support both design and logistics functions and are useful in concurrent engineering development. The information used to perform these analyses is the same as described previously.

### 3.1.10 Testability/Reliability/Maintainability Metric

An overall metric may be determined for any given circuit, which combines all of the relevant issues examined above. This metric is a representation of the overall supportability of the system under analysis, since it represents how often then system will fail, the relative cost (time and material) to diagnose, and the cost to replace once properly diagnosed. This metric, if further refined, could represent a basis by which competing proposers of a system may be evaluated,

as well as a basis for incentives during development. It is important, however, that such a metric be correlated to the ultimate Operation and Support cost of the weapon system development.

An analysis incorporating the data gathered for testability, reliability, and maintainability (TR&M) allows the determination of units that have low values for each 'ility. If a unit has a low reliability and a low testability, it should be easily maintained so that it may be exchanged to determine if it is not operating.

This analysis pinpoints system units that have a great need to be modified so that system faults that are very difficult to discover do not keep the system off line for a long time.

## **3.2 The Concept: The R&M Design Expert System**

The R&M Design Expert System was designed as a framework to address TR&M issues in the same environment. Though we did not address each area of the Venn diagram, we did integrate the three fields into a single environment with a strong focus on testability. The following sections detail the initial concept of the approach taken for each of the sections. During the implementation phase, we found that not all aspects of the concepts could be developed within the scope of the contract.

### **3.2.1 Scope: Focus on Test Selection and Coverage**

After we reviewed the technical areas of design and tools available to aid the TR&M process, we decided that the greatest impact from applying knowledge-based system techniques was in testability. While there are numerous reliability tools available to the electronic system designer, there are few, if any, tools that aid the design with functional testability. The testability process is pervasive throughout all product phases. For purposes of scope, the concept addressed only the diagnosability subset of maintainability, specifically test coverage (e.g. can the system detect and isolate the fault(s) given the testpoints and associated measures?).

The functional block diagram of the R&M Design Expert System is shown in Figure 13; it consists of four main system modules. For clarity, the concept is described in a serial, sequential process; however a more enlightened view is shown in Figure 14. The system can be thought of as a body of shared, interacting knowledge about TR&M, wrapped around a set of core tools that uses infor-



mation available from design data.

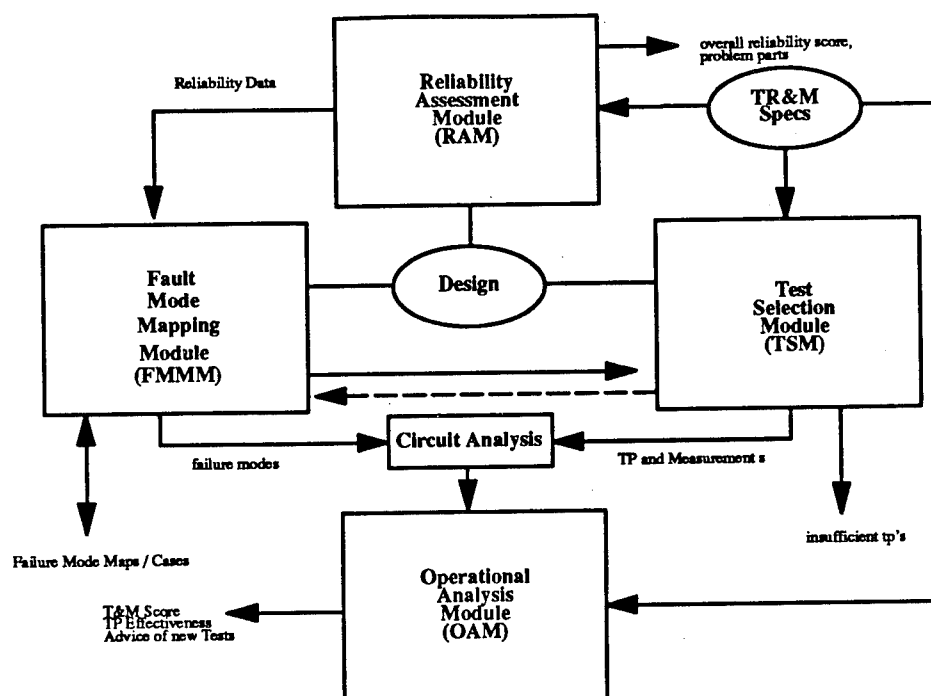


Figure 13 System concept.

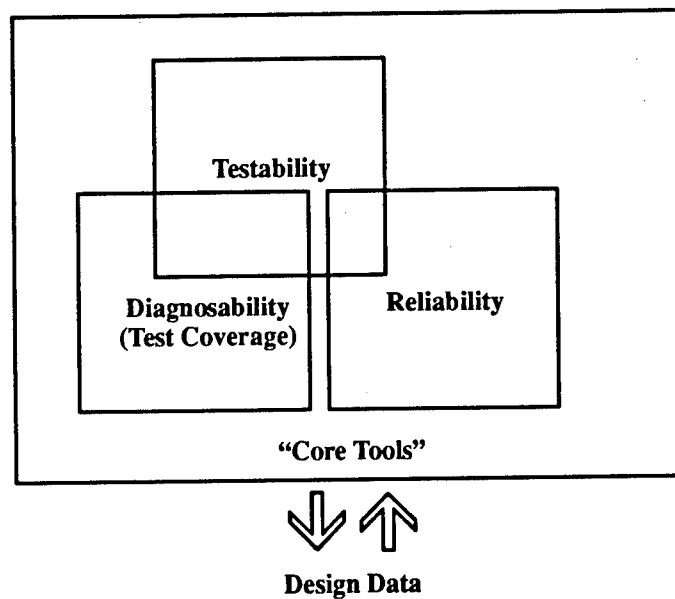


Figure 14. System concept with different view / same functionality - "Shared Data."

### **3.2.2 Concept Design Drivers: Tools Used in the System**

Before detailing the specific features of the system concept, a brief description of the tools used is necessary to understand their impact on the system concept. The tools to represent and analyze electronic system designs include Mentor Graphics' Falcon Framework™ with Design Architect™, and Board Designer™. Analog circuit simulation, Saber™ from Analogy, Inc. was used to collect data. The Mentor and Analogy tools used for this program are part of the standard utilities being used at all Martin Marietta facilities. Reliability data was determined by RL-Oracle. The knowledge-based system tool was ART-IM™ from Inference Corp.

#### **3.2.2.1 Mentor Tool Suite: The Base CAD/CAE Environment**

Mentor offers a development environment/tool suite ranging from schematic capture to analog and digital simulations. The tools adhere to a standard behavior, look, and feel in the Falcon Framework and are offered as either applications or more comprehensive station software. Design Architect is the base schematic capture tool. It offers a fairly complete set of available digital and analog parts libraries that optionally can contain information for use in other Mentor applications (e.g. simulation models, footprints). The Mentor tools use a common control language, called AMPLE™, to communicate to a tool, and effect changes in the person-machine interface directly. Among non-interactive features, Mentor offers Design File Interface™, which consists of a small number of C language routines that allow data access to the design, including back annotation. Changes, additions to tools, or complete user applications can be developed using Design File Interface, AMPLE and C. Tools are loosely linked via the Design Manager™ tool, which provides a window-based environment to manage designs, and tool invocation, as all the Mentor applications can be started in Design Manager via a click on an icon. Electronic system designs can easily be renamed, moved, etc. via Design Manager. User applications can be embedded in Design Manager (or other tools) via Registrar™. Mentor tools are not currently CALS compliant.

#### **3.2.2.2 Saber Analog Simulation**

Saber is the analog simulation used to develop the R&M Design Expert System. Saber was integrated into the Mentor tool suite, via an optional product called Frameway™. Frameway allows designers to access the simulation, with options, directly from Design Architect. Once invoked, designers can choose from many options, including DC operating point analysis, AC analysis, transient analysis, signal plotting, etc. Frameway also comes with a large number of analog and

digital schematic parts and devices for use within Design Architect. These objects contain pointers to the correct Saber simulation model and can be customized using the Design Architect utilities. Users can write their own simulation constructs with the MAST™ modelling language. The simulation is fairly expensive and requires a large amount of RAM to run effectively. It is also necessary to purchase an extra cost digital simulation option, as some analog functionality is hidden in this option. The tool is easy to install and maintain. Future releases will allow Saber to be started directly from Design Manager.

### **3.2.2.3 RL-Oracle**

RL-Oracle was the reliability tool for the system. RL-Oracle is the automated version of MIL-HDBK-217E. For purposes of system development, RL-Oracle was considered an off-line, manual process, as it only runs on a PC or VAX environment. There are similar tools available that run on the proper SUN platform, but the cost was not warranted. RL-Oracle was run using SoftPC, a PC emulation on a SUN workstation, which enabled access to the Unix file system that the other tools operated under.

### **3.2.2.4 ART-IM: The Knowledge-Based System Tool**

ART-IM (Automated Reasoning Tool for Information Management) is a knowledge-based system development environment from Inference Corporation. The system offers five major components, including an object-oriented knowledge representation which includes schemata, an inference engine which includes both forward and backward chaining, a machine-specific formalism compiler, a graphical user interface (GUI) that allows the developer to browse, edit, and view the application in operation, and a module which allows Case-Based Reasoning (CBR). Future releases of ART-IM will also include the dynamic control of application GUI components.

## **3.2.3 The Application of Knowledge-Based System Tools in Electronic System Design**

Traditional development of expert systems used either forward or backward chaining production (rule-based) systems. In forward chaining systems (e.g. OPS5, CLIPS), data about the world is examined to see what can be concluded; backward chaining systems make an assertion and seek data to support same. CBR attempts to capture specific knowledge as a series of experiences or cases.

CBR systems are emerging in help-desk applications, such as customer service troubleshooting. Where a traditional chaining system needed to have a fairly precise match on necessary conditions or data, the CBR system can match against similar, but not exact, cases. Electronic system design is not always an exact science; there always seems to be a trade-off of one very important feature with an equally important one. A traditional chaining system will not allow the effective application of trade-offs, both at the system and user interaction level. The use of CBR in the R&M Design Expert System allowed the capture of these trade-offs, enabling potential users to make choices among the various cases, particularly in selecting failure modes and tests. The system also used a forward chaining approach for test coverage evaluation.

### **3.2.4 Required System Inputs**

#### **3.2.4.1 A Hierarchical Design to Analyze**

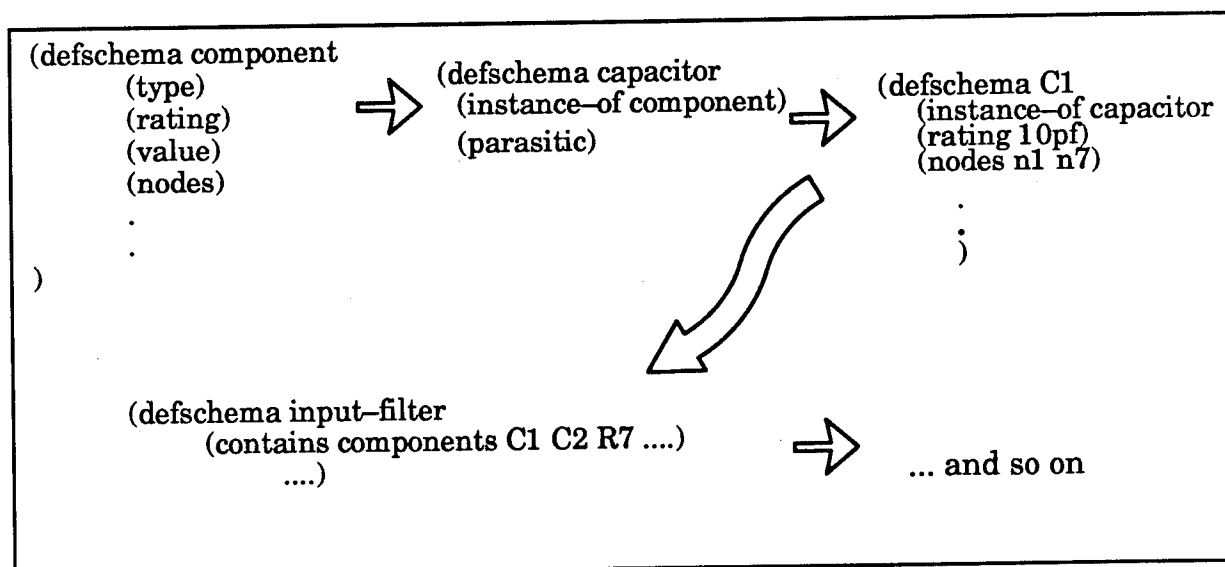
The system concept assumes that design data is available in a hierarchical representation. From the CAD/CAE perspective, the hierarchical design methodology is analogous to top-down programming. Each symbol on a design sheet can represent a lower functionality. With this method, each section of the design can be built and tested separately. When a particular section has been correctly implemented, it can be held constant while other sections of the design can be changed. Through this method it is possible to reduce the amount of complexity on the higher level schematics, which makes the design more manageable.

All designs in the R&M Design are represented hierarchically. This allows the system to view and reason about TR&M from a functional perspective. Without a hierarchical representation in the CAD/CAE system, the knowledge-based system would have absolutely no knowledge, for example, that "capacitor C1, between nodes N1 and N7" is in the input filter.

Mentor supports the design hierarchy representation methodology. Lower level designs can be implemented and attached to a higher-level component. The lower levels may be represented as schematics, symbols or VHDL modules. Each of these representations is interchangeable in the design hierarchy. Using the design hierarchy also allows changing portions of the design later (using a new implementation of a component, for example), so long as the interfaces to the other portions remain constant.

Mentor also allows for different viewpoints for designs. This enables viewing each level of the design hierarchy separately. A netlist, for example, can be

created with all the design levels, one level or a multiple of levels using different design viewpoints. These viewpoints can also specify what characteristics about each component or functional representation can be seen and hence support the informational needs of the knowledge-based system representation. There is a one-for-one mapping of design levels and objects between the CAD/CAE and knowledge-based system representation. The ART-IM representation of the design hierarchy includes a set of prototype schemata that describe generic component types (e.g. capacitors, diodes) and modules which then are instantiated to describe the complete design. An example of this is shown in Figure 15.



*Figure 15 An example ART-IM representation.*

### 3.2.5 Additional Design Representation

We found that electronic system designers often use the component reference designation for additional information. An example of this is the use of ranges of numbers to signify a component's main functional area (e.g., all component numbers 1-999 are in the input filter). The ability for designers to communicate this type of information, if available, was thought to be a useful supplement of the hierarchical design data.

### **3.2.5.1 TR&M Specifications**

In addition to a design to analyze, potential users were asked to provide design specification information that has not been extracted or synthesized from the design itself. Specification information could include MTBF, component derating data, operating temperature requirements, etc., that will act as metrics for system analysis. Most of this information should be available to users from the product or design specification as provided by or for the procuring entity. We reviewed several specification for analog power assembly designs, and provided a base template as part of the system.

### **3.2.6 System Modules**

The R&M Design Expert System concept contains four functional modules: Reliability Assessment Module (RAM), Fault Mode Mapping Module (FMMM), Test Selection Module (TSM) and the Operational Analysis Module (OAM). These modules, plus the associated CAD/CAE and knowledge-based system tools, make up the total system.

#### **3.2.6.1 Reliability Assessment Module (RAM)**

Reliability assessment has two purposes in the system concept: 1) to provide a base MIL-HDBK-217E analysis, and 2) and, perhaps more importantly, to provide the electronic system designer and potential system users with an initial functional reliability assessment that can be refined in further analysis, and guide the selection of focus for testability/test coverage analysis. A system module that is 99.99% reliable is less interesting to analyze for testability than one that is at the fringe of passing specification, particularly with limited analysis resources.

The R&M Design Expert System did not attempt to duplicate existing functionality in MIL-HDBK-217E analysis; we used the USAF Rome Laboratory-developed RL-ORACLE tool to assess module level reliability. RL-ORACLE was considered an off-line, manual process, as it only runs in a PC or VAX environment. There are other 217E tools available that run on the SUN platform. RL-ORACLE was run using SoftPC, a PC emulation on a SUN workstation, which allowed access to the Unix file system that the other tools operated under. Using data collected from the functional design representation (hierarchy) and system-level specification, the RAM created the required input file, system.dat, for the reliability tool. This file was partitioned by modules of the design as determined by the hierarchy to produce reliability estimates for each functional module, as



well as the total system. RAM allows users to review and edit the input and the summation output file of RL-ORACLE. RAM is shown in Figure 16.

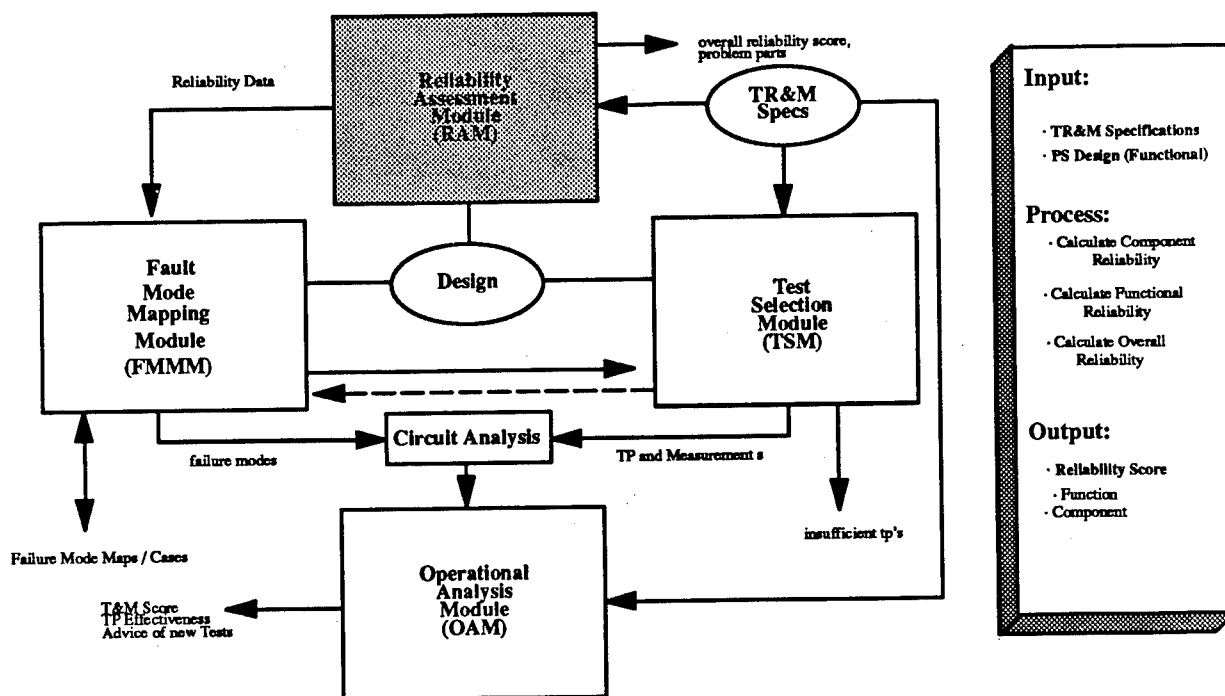


Figure 16 Reliability Assessment Module(RAM).

### 3.2.6.2 Fault Mode Mapping Module (FMMM)

One way to begin to evaluate a candidate electronic system design for testability and fault coverage is through fault injection, or a Failure Modes Effects Analysis (FMEA). The R&M Design Expert System concept's Fault Mode Mapping Module (FMMM), shown in Figure 17, addressed a portion of this type of analysis. The FMMM provided potential system users the ability to select defined faults to inject in a selected electronic system design module (e.g. the input filter) for testability analysis. FMMM is the system's repository of previously defined fault-to-failure-mode mappings, and some information of the failure mode for simulation purposes. Once users select, edit or create a new mapping, the system then performs a simple, yet necessary, consistency check on this mapping.

Failure modes were defined in one of two ways; 1) simulation component model alteration or 2) behavioral component insertion (e.g., if leaky capacitance is desired, add a capacitor to the component netlist). While the model alteration method of simulating the failure is most desirable (i.e., no netlist alteration, or



bookkeeping of multiple netlists for schematic versus simulation), most simulation vendor's models are proprietary, and hence not viewable or alterable.

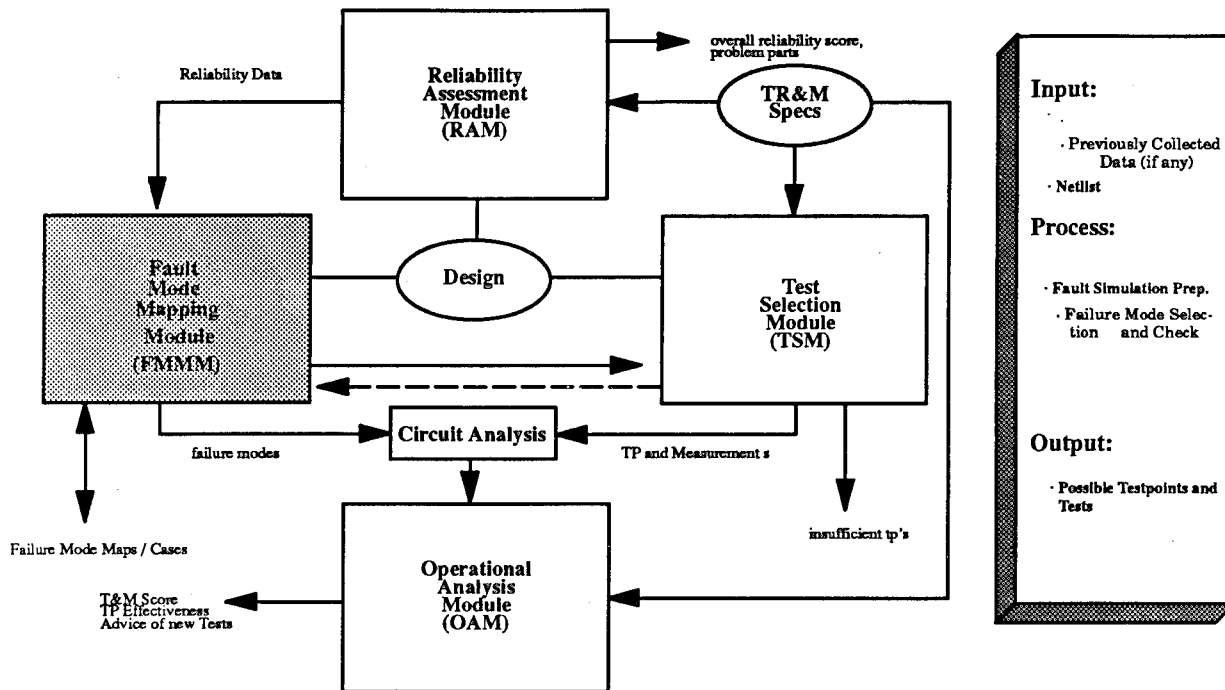


Figure 17 Fault Mode Mapping Module (FMMM).

### 3.2.6.3 Test Selection Module (TSM)

The Test Selection Module (TSM), shown in Figure 18, is a knowledge-system module for testability analysis. The TSM verifies any testpoints and test selections that users defined, and suggests additional tests and test points for the failure modes defined in FSSM. The TSM's knowledge about failure modes and their associated test points and tests or measures was organized as a set of cases. Users select among the the best cases presented by the system for particular failure modes or define new cases. These cases include test points and a measurement apparatus (e.g., a scope). The system did not have a comprehensive set of cases for all failure modes defined in the system; it had a few working templates that users built upon or customized. This enabled the system to grow in knowledge over application. The TSM is a pre-simulation testability analysis. It defines and verifies (via the knowledge-base) tests and testpoints for particular failure modes. The next step was to collect data to see how effective the tests and

test points actually were. This is the focus of the Operational Analysis Module (OAM) described in section 3.2.6.4.

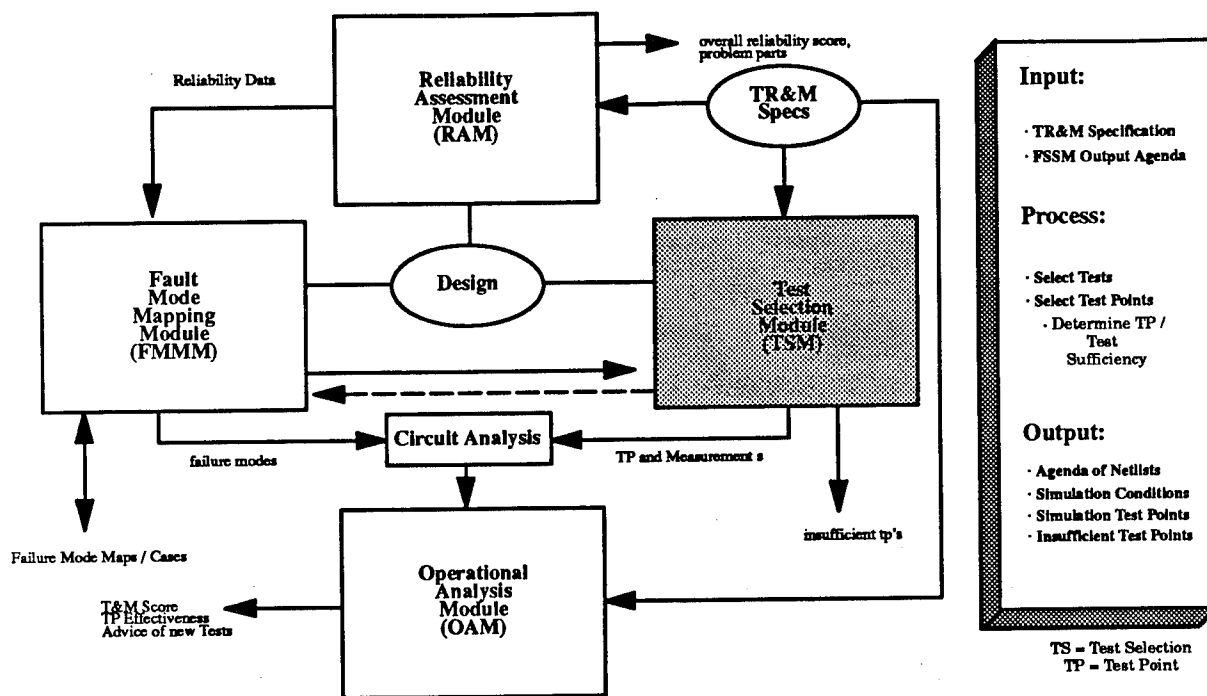


Figure 18 Test Selection Module (TSM).

### 3.2.6.4 Operational Analysis Module (OAM)

The Operational Analysis Module (OAM) is a knowledge-based fault detection/diagnosis module. OAM, shown in Figure 19, reads output data as a result of the system users simulation run(s). Using this data, the system attempts to detect and possibly isolate the faults defined by the FMMM with the test points defined by the user and/or TSM to evaluate test coverage. Detection/diagnosis occurs in the OAM using a traditional forward chaining rule-based technique that matches specific signature behaviors and observations to particular faults. Especially challenging to the OAM is its ability to detect and diagnose multiple, masking faults. If a fault cannot be isolated, or even detected, the system defines new testpoints, if possible, and upon another simulation run invoked by the user, attempts to detect/diagnose. The testability cases defined by TSM, or the user, are then updated to reflect the outcome.

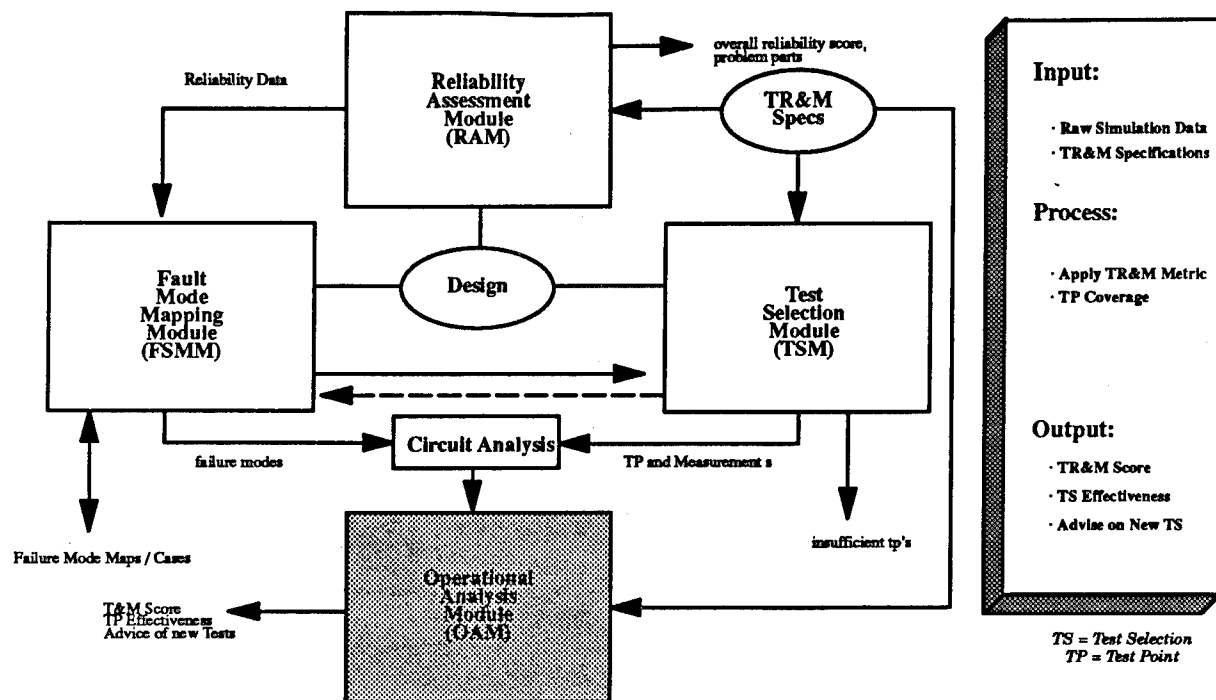


Figure 19 Operational Analysis Module(OAM).

### 3.2.7 Assumptions

The concept makes several assumptions for system development and operation. A hierarchical design is probably the single most important requirement for system input. A flat schematic/design can be evaluated in a limited sense, without any consideration of module level functionality. All modules or components of the R&M Design Expert System (e.g. TSM), could be operated individually without depending on a previous module; however, in certain cases it was much more effective to have run a previous module. For example, executing the OAM functionality without defining faults was not very useful.

## 4. Implementation

### 4.1 Development Overview

Phase I development concentrated on building the initial interface for the R&M Design Expert System and building a concept demonstration that illustrated the future functions of the system.

During the first months of Phase II, we put significant effort into tool integration and data extraction. This included designing and implementing an interface between the R&M Design Expert System and Oracle for reliability analysis; coupling ART-IM and Mentor so that decisions on the current design could be made; and developing a methodology to communicate and extract information to Saber, the circuit simulator. Developing the interface became an ongoing practice as Mentor continued to upgrade the tool over the life of the contract - often implementing major changes between versions, which caused effort to be placed into coming up to speed on the new tool, and verifying and changing existing work.

A critical activity for Phase II was creating the Fault-to-Failure-Mode Map(Failure Mode Map). The Failure Mode Map was designed to be used in both the Test Selection Module (TSM) and the Operational Analysis Module (OAM). The Failure Mode Map represents the experiences of the system and the designer modeling how faults affect the operation of the design.

The model used for test selection functions and the OAM were similar, but operated from opposite directions. Work in both the TSM and the OAM built upon the Failure Mode Map to implement their functions.

During Phase III, we concentrated on expanding the TSM, the OAM, the Fault Mode Mapping Module, and the Fault-to-Failure-Mode-Mapping Casebase.

We expanded the Fault-to-Failure-Mode Map to cover more functional blocks, and different faults and failure modes. The new mappings conformed with the previous Map developed in Phase II. We did not extend the map to include faults other than open and shorted components due to lack of time. All the extensions were added to the casebase. The representation and weights of the casebase were also validated.

The OAM underwent the most alterations. We added functionality to analyze more than one simulation regarding a fault, a generalization function to go from a specific testpoint to the general, and increased fault isolation techniques. Figure 20 shows the R & M Design Expert System.

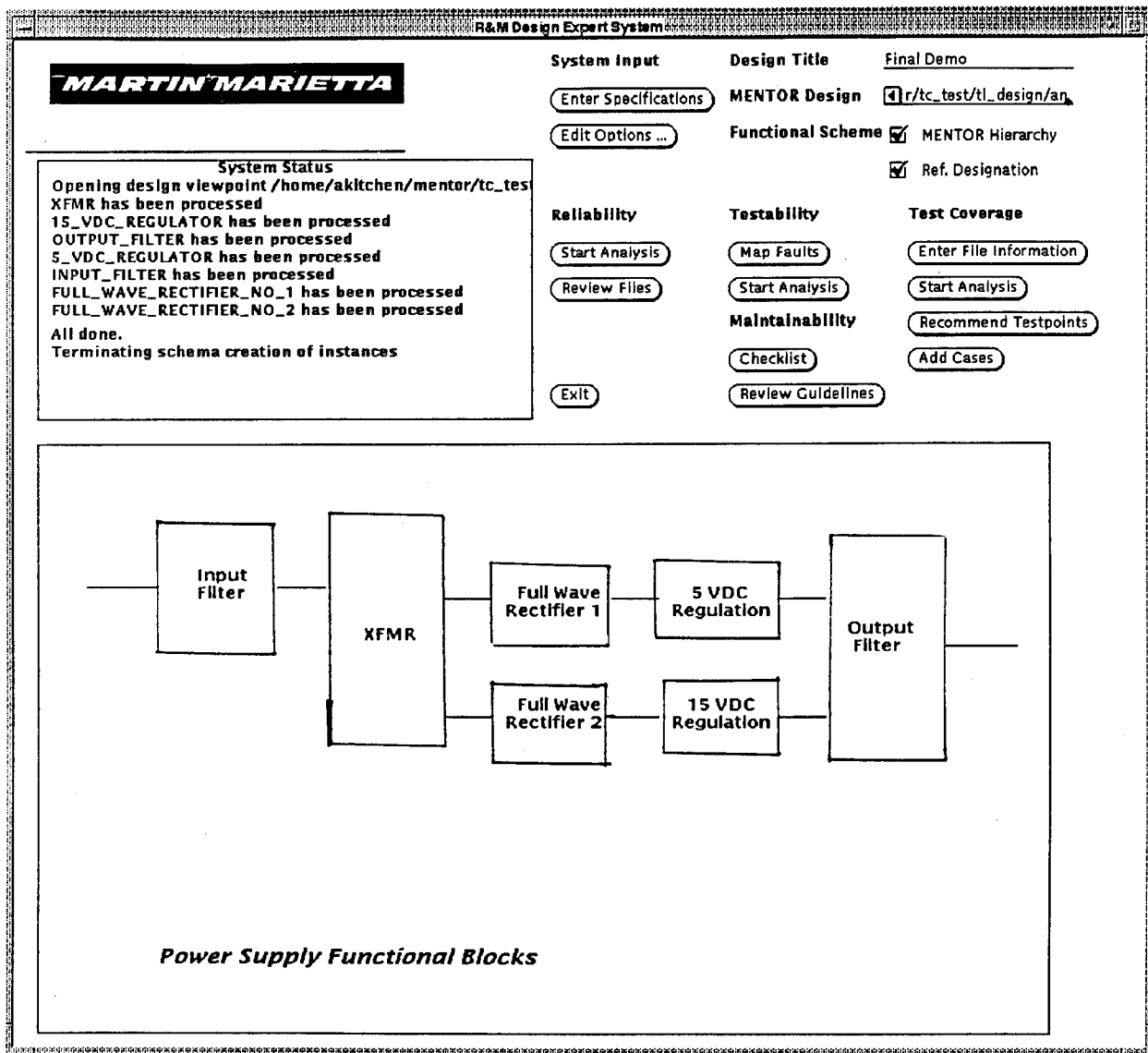


Figure 20 The R&M Design Expert System.

## 4.2 Interfacing with Mentor Graphics

Implementing an efficient, correct, and easily maintainable interface with Mentor was important to easily expand functionality.

The Mentor tools used in the R&M Expert System are Design Architect™, a schematic capture tool; Design Viewpoint Editor™ (DVE), used for registering information about the design; and DFI™, an interface for accessing information about the design. Design Architect, DFI, and DVE are the main tools to access information about the hierarchical design. The relationships between these tools is shown in Figure 21.

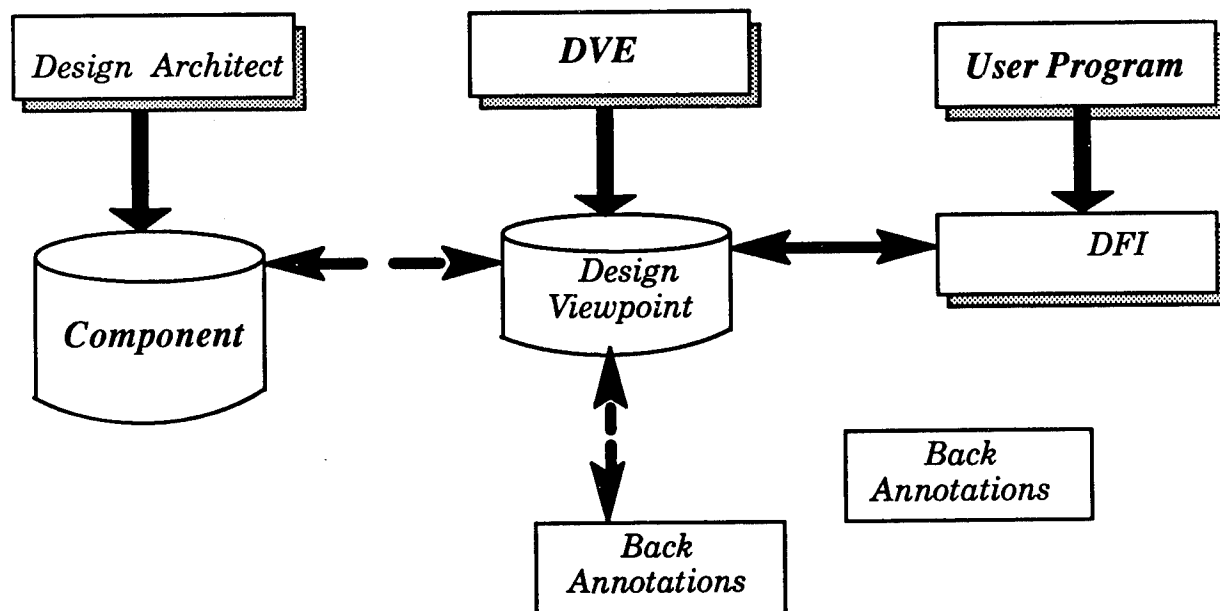


Figure 21. Relationships used by the R&M Expert System in the Mentor Graphics hierarchy.

## 4.3 Interfacing with the Design

The design viewpoint is the key to extracting information from the design. If the design viewpoint is not correctly set up, the needed information can not be extracted. From a technical stand-point the design viewpoint is critical. The viewpoint can be considered the window through which the design is viewed. A design may have multiple viewpoints, which can be attached and unattached according to need. Viewpoints provide a read-write channel to the data in the design, but may only read the information that is registered as visible in the viewpoint. Users must register the data visible with DVE. For example, a component may have a part number associated with it, if it is registered with DVE.

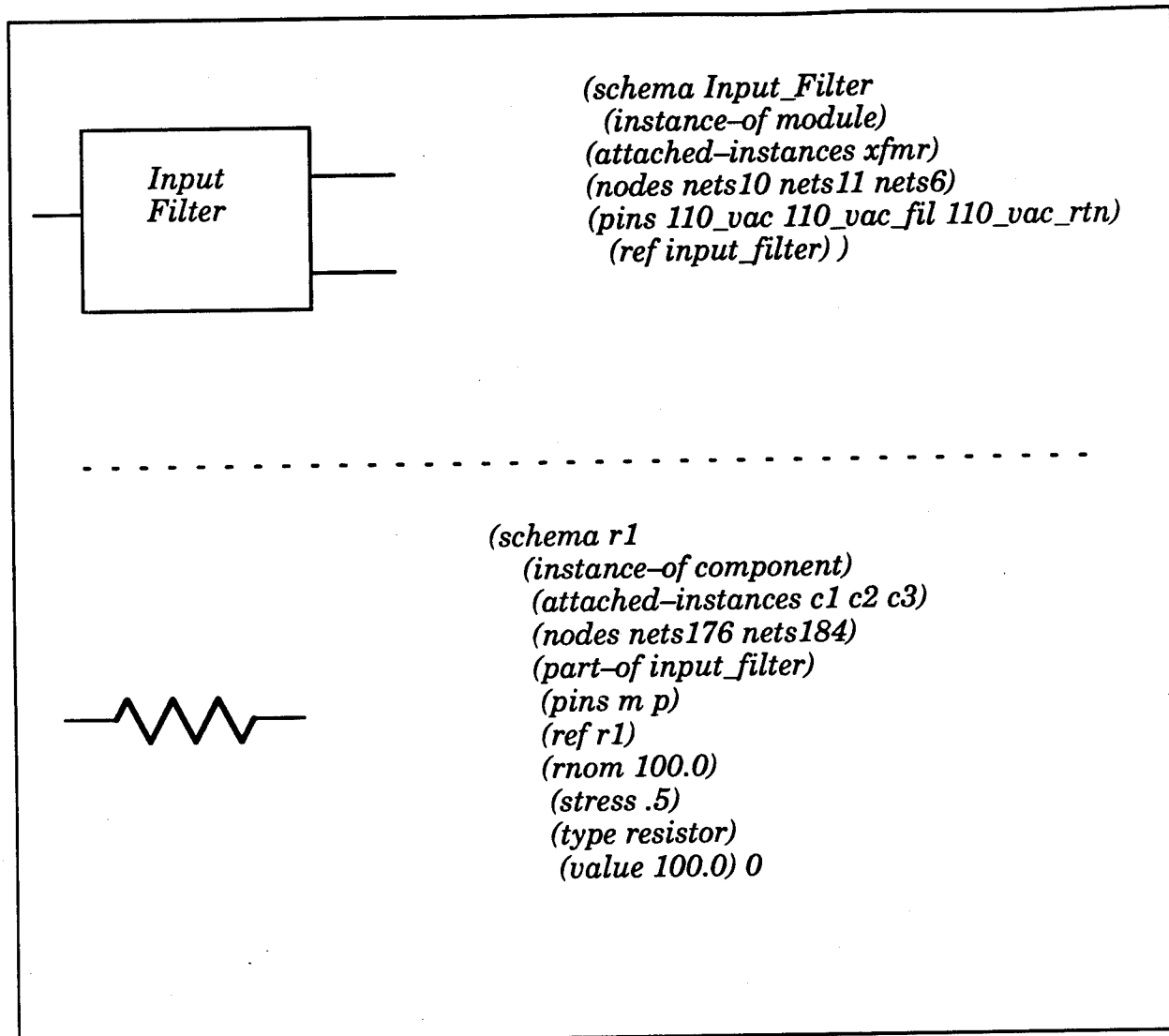
Back annotations were used to add new properties to the designs. Back annotations also enable editing of existing properties. Since they are stored in a separate file, many different back annotations can be available for each design.

DFI, a C library containing design access functions, uses the viewpoint channel to access the properties of the objects and to write information back to the design (in the form of back annotations). The interface is accessed through C programs, which allowed us to insert this information into ART-IM™ where the knowledge-based tools can work on it. DFI has a number of limitations, including:

- Graphical information can not be accessed from a design, such as placement of components on the sheet, or the thickness of connections which may represent important connections
- Information must be accessed through a design viewpoint, which may or may not be set-up correctly, thus altering the view of the design
- Modifications are limited to back annotations, and are not permanently connected with the design
- The data is formatted consistent with the pre-V8 database even though the underlying model has changed significantly
- Back annotations must be registered before they can be viewed in Design Architect.

We designed a module to gather all the pertinent information from the design and insert it into the ART-IM knowledge base. In the knowledge base, we modelled both low-level components and functional blocks. (See Figure 22 for example). This information was used in the Testability Module.





*Figure 22 Examples of the models of functional blocks and components used in the knowledge base.*

#### 4.4 Fault-to-Failure-Mode Map

The Fault-to-Failure-Mode Map provides a set of behaviors corresponding to component faults. The more complete the map is, the more useful it is to the design and the more accurate the results of its use. The map is designed to help users select a minimal set of test points, to develop test plans based on the availability of these testpoints, and to help isolate a fault to a particular component in a functional block. An example of the Fault-to-Failure-Mode Map is shown in Figure 23.

Functional Block	Fault	Failure Mode	Measurement Point	Measurement Parameter	Signature Diagnostic Behavior
Rectifier	Open Diode	Increased Ripple under load	Rectifier Output	peak to peak voltage	Increased peak to peak voltage
Rectifier	Open Diode	Decreased Frequency	Rectifier Output	Frequency Spectral Analysis	Frequency will be 1/2 of working circuit
Transformer	Short Source Drain	Reduced Output Voltage	Transformer Output	AC Voltage	Reduced peak to peak voltage
Transformer	Open Capacitor	Decreased Output Voltage	Transformer Output	AC Voltage	Peak to peak voltage is zero
Output Filter	Open Inductor	Decreased output Voltage	System Output	DC Voltage	DC Voltage reduced

*Figure 23 Example of a Fault-to-Failure-Mode Mapping used for establishing the casebase.*

#### 4.4.1 Creating the Fault-to-Failure-Mode Map

The Fault-to-Failure-Mode Map was based on two designs: a simple power supply design, and a more complicated power supply design obtained from Martin Marietta's AstroSpace Group. Each design had functional blocks which included an input filter, a transformer, a rectifier, a regulator, and an output filter. (The AstroSpace design contained additional control, reset and shutdown functions, which were not modelled.)

The Fault-to-Failure-Mode Map table was generated by simulating faulted circuits and a baseline circuit. Faulted circuits are simulated by making minor changes to the baseline netlist file. For example, an open component is modelled by simply removing the component from the netlist. A shorted component is modelled by replacing the component with an extremely small resistor. Other faults can be created by adding or altering components in the netlist file. Five faults were selected from each of the functional blocks.

#### 4.4.2 Using the Mode Map for Testability

The mode map is the basis of testability analysis for the R&M Expert System. The R&M Expert System uses the mode map to suggest tests and test points for

any given failure and to analyze the designer's provided tests and testpoints. The theory behind test point selection illustrates why a mode map is useful in test point selection.

Test points help to uniquely identify fault components; however, too many test points lead to inefficient or impossible design. A minimal set of test points is desired that meets a system's particular fault isolation requirements. Faults are not required to be isolated down to the device level, but to the Line-Replaceable Unit (LRU) level.

With respect to the mode map, additional test points make the resulting signal behaviors more unique. For example, if faults in either of two serial devices lead to the same behavior, the faults are indistinguishable and we won't know which device failed. If a test point is inserted between the two devices, then we may be able to tell which device failed. Test points at every node in the system would certainly support isolation down to the component level, and the map's behaviors would be unique because the behavior would be defined in terms of a large number of test points. That is, each behavior would uniquely identify the fault causing it, and the fault could be easily identified from the behavior seen under real-life circumstances.

Unfortunately, a large number of observations would be required because of the large number of test points. If we remove some test points, then we cannot distinguish some faults from other faults, but the observed behavior at least limits the possible causes to a set of faults - the degree of uniqueness of the behavior dictates the degree of completeness of the fault isolation it supports. For practicality of design, the set of test points must be minimized. The size of the LRUs drives the placement of test point. The map documents the relationship between possible or likely faults, isolatable faults, and level of isolation and a set of test points.

#### **4.4.3 Using the Mode Map for Fault Isolation**

The mode map contains a failure mode for each fault. Failure modes are behaviors that deviate from the baseline behavior of a circuit. The baseline behavior is the behavior the circuit exhibits when no faults are present. The mode map is used to identify a unique fault based on knowing the failure mode. The map is generated by inducing faults into the simulation and noting the resulting failure modes or behaviors, and it is used in the opposite direction - identifying faults based on observed failure modes.

The mode map shows that there are several parameters that must be measured to detect failure modes. The comparison of the failure-mode waveforms with baseline waveforms requires tolerance specifications so that the system may know what level of deviation is allowed by the designer or the specification for the design. These specifications for the inputs and output signals of the circuit are provided by the circuit customer, but the tolerances for test points are not. These tolerances should be derived from the available input/output tolerance specifications combined with the simulation performed to construct the map.

## 4.5 Casebase Methodologies and Views

The R&M Expert System uses case-based reasoning( CBR) for a large portion of the knowledge-based functionality. CBR is a method through which experiences or cases are captured and stored. CBR technology can find cases that are similar to the current situation. In the R & M Design Expert system we used CBR to represent the Fault-to-Failure-Mode Map. The structure of the casebase is illustrated inFigure 24.

### Elements in the Cases:

Name : Type of fault (e.g. open diode)  
 Functional Block: Functional block fault occurred  
 Failure Mode: Effect on the Circuit  
 Measurement Point: Point where measurement is taken  
 Measurement Type: Type of measurement (e.g. DC voltage)  
 Signature Behaviour: observed behaviour  
 Measurement Devices: devices used in measurements

*Figure 24. Structure of the casebase used in the testability and test coverage modules.*

## 4.6 Casebase Views

The R&M Expert System used a methodology which employs views on the casebase. Two views express the concepts in which we want to assist users. These concepts help users define tests and test points, and isolate to a type of component within a functional block. The same information is contained in each view; however, each task requires the casebase to be viewed differently. Each of these views provides users the opportunity to interpret the information that resides in

the casebase in the manner most appropriate for the situation or task. By using this approach, we can weight the information in the casebase to yield the correct information, depending on the module.

A casebase view is a specified set of weighting of elements in the casebase. Each element can contribute toward the final score of the case. The score is a measure of how well a case matches the proposed case. The proposed case is constructed to model the question that is currently being asked to compare it to those cases that already exist in the casebase. The score can range from -1.0 to 1.0. A score of 1.0 constitutes a perfect match - all the elements of the proposed case are exactly the same as the elements on a case stored in the casebase. The elements are given weights relative to each other. For example, the name element could be given a weight of 80, while the functional block is given a weight of 100. This causes the functional block feature to contribute more toward the total score. An element may be ignored in the matching process by being assigned a weight of zero.

#### 4.6.1 Testability View

The testability view of the casebase is the view which is used to recommend possible tests and test points to users based upon a functional block and a type of fault. In this view, the functional block element was given weight (57% of the total score) over the name element (43% of the total score). The other elements are not pertinent to the testability portion of the system. The name and functional block elements of the case can point to tests and testpoints that should be used. We initially considered the name to be more important, as the type of fault usually suggests what type of test and testpoint could be used rather than the functional block. But we found that this results in incorrect suggestions because the same component in different functional blocks reacts in different ways. Thus the functional block was considered more important, which results in suggestions only with related components in the functional block.

The Test Selection Module (TSM) is used to suggest to users possible tests and test points that can be used to isolate a fault. Any tests and test points that users define are checked against the casebase for validity. The casebase is not exhaustive, however. It has a representative number of cases that can be expanded upon by system users. This allows users to customize the casebase in a way that makes sense for their projects.

## 4.6.2 Test Coverage View

The test coverage view of the casebase is the view used to isolate faults in a circuit. The casebase is the last step of the fault isolation process. This view uses the measurement point, measurement type, and signature behavior fields of the casebase. Using this information, the Operational Analysis Module (OAM) attempts to isolate the fault to the type of component within a functional block.

In the test coverage view of the casebase the signature behavior (39% of total score) feature is given the most weight, while the measurement point and measurement type features each have the same weight (31% of total score). This weighting was used because the signature behavior is the most important factor, although the measurement point and measurement type are also important in isolating a fault.

## 4.7 Reliability Assessment Module

The purpose of the Reliability Assessment Module (RAM) is to provide a base MIL-HDBK-217E analysis and a functional reliability assessment. These measurements provide electronic designers with a guide to focus the testability/test coverage analysis. A system module that is 99.9% reliable is less crucial for a testability analysis than one that is near the low end of the specification.

The R&M Design Expert System uses the USAF/Rome Laboratory-developed RL-ORACLE tool to accomplish the reliability assessment. This does not preclude the ability to use another reliability tool. The RL-ORACLE analysis is an off-line process because it must run under a PC emulation on the Sun workstation (we use SunPC).

The RAM collects information regarding the parts and functional blocks from the design through a DFI interface. An input file, system.dat, is automatically generated for designers and it contains all the pertinent information needed for the RL-ORACLE analysis. Once the analysis has been completed, the RAM accesses the output of the analysis and back annotates this information to the design. This allows designers to view both the functional-level and component-level MTBF and Failure Rate, informing designers of the components or functional blocks that may fail. Any functional block that fails to meet the specification will be noted and pointed out for modification.



## 4.8 Test Selection Module

The TSM is a the knowledge-based portion of the system that assists users in testability analysis. This module suggests additional tests and test points for the failure modes which interest the user.

The main component of the TSM is the casebase. The casebase stores all previous knowledge of particular failure modes and the corresponding tests and test points. The module constructs potential cases from the functional block and fault mode entered by the user. The information is used to query the casebase on potential tests and test points that can be used to detect the desired fault. The casebase potentially returns any number of potential tests and testpoints. The TSM must deduce which case most closely matches the fault required by the user. Each case returned by the casebase is given an overall score based upon how close the case in the casebase matches the presented case. Cases that are close can be presented to the user. The selections made by users for the measurement point and test equipment are then used as inputs to the OAM.

## 4.9 Operational Analysis Module

The OAM is the knowledge-based portion of the system that provides feedback to designers as to the effectiveness of specific tests and test points. This module detects and diagnoses the fault and suggests additional tests and testpoints when the fault cannot be diagnosed.

The OAM reads as input the output results from Analogy's Saber simulation runs and attempts to detect and isolate the injected faults with the testpoints defined by users in the TSM. If a fault cannot be isolated, or even detected, the system defines new test points, if possible and upon another simulation run invoked by users, attempts to detect and diagnose the fault. The casebase is then updated to reflect the outcome by the addition of new cases to the casebase. The OAM consists of four main components. The components represent the four primary tasks that must be performed in an automated approach to the test coverage process. These include the Saber Parse component, which reads the simulation results into the system; the Fault Behavior Analysis component, which analyzes and detects faulty behavior; the Fault Isolation component, which is responsible for pinpointing the type and location of the fault that caused the behavior; and the Update Mapping Casebase component, which is responsible for updating the system's casebase of faults.

The basic software architecture diagram of the OAM is shown in Figure 25.



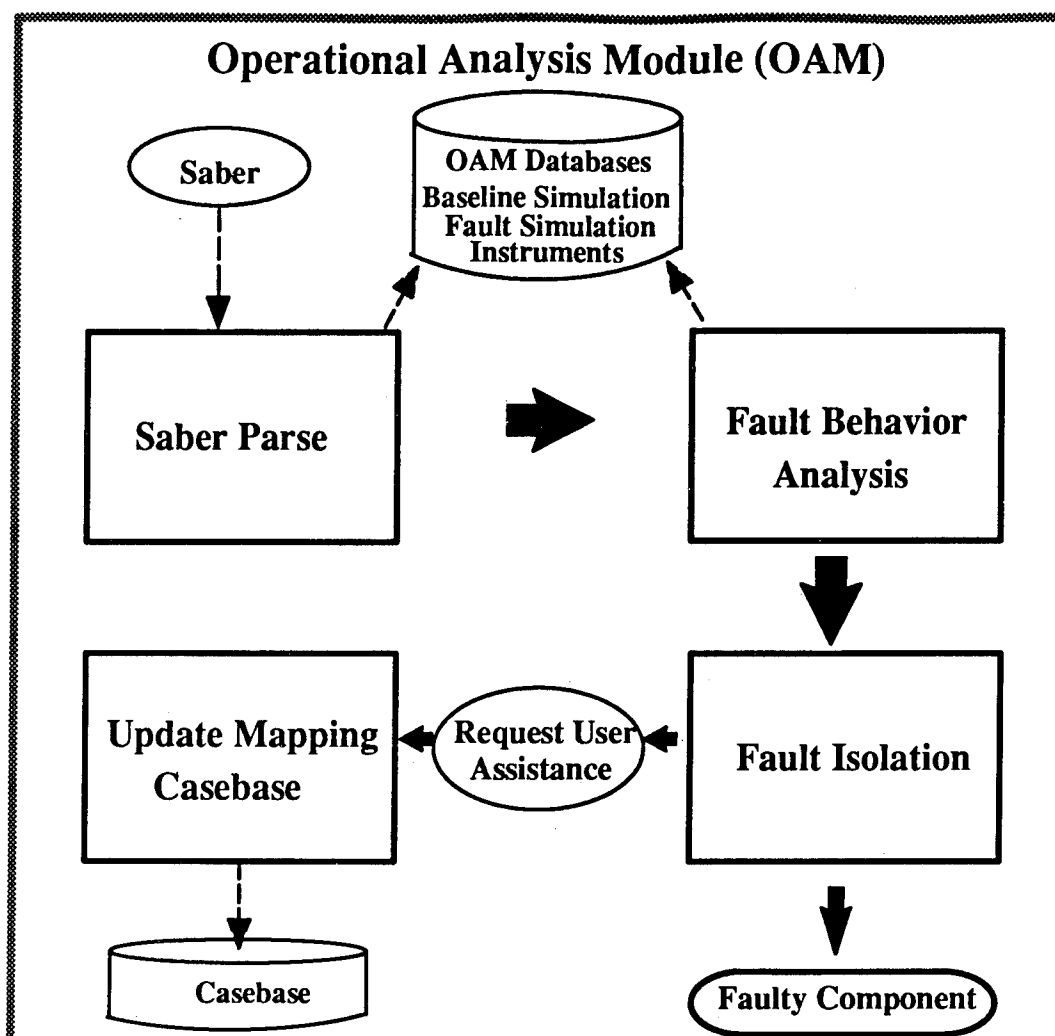


Figure 25 Architecture of the Operational Analysis Module (OAM).

#### 4.9.1 OAM Databases

Local databases required by the OAM include the Baseline Simulation Database, the Fault Simulation Database, and the Instrument (measurement device) Database. The OAM software architecture diagram, Figure 25, displays the interfaces between the databases and the software components of the OAM.

The Baseline Simulation Database is a repository of atomic information relating to baseline circuit's behavior at each of the net locations, including selected measurement points, during the baseline simulation. The baseline simulation is a fault-free simulation of the circuit model. The information contained in the Baseline Simulation Database represents the true expected behavior of the cir-

cuit model. The baseline is used by the system as a comparison base when attempting to detect faulty circuit behaviors.

The Fault Simulation Database is a repository of atomic information relating to the circuit's behavior during a fault simulation. In a fault simulation, designers inject a fault into the model and then execute a simulation to determine the circuit's behavior. The information contained in the Fault Simulation Database relates to the faulty circuit's behavior at each of the design's net locations and user-selected measurement points. The baseline and fault simulation databases are identical in structure; however, they are maintained separately to preserve the integrity of the baseline circuit behavior data.

The Instrument Database contains information related to the properties of the Saber instruments used in the baseline and fault simulations. Information contained in the database includes the designer-selected instrument name, the designer-selected test point, functional block location, and the Saber primitive instrument type (i.e., "mave," "mfreq"). The instrument properties are obtained by reading the Saber-generated netlist file.

Saber instruments perform functions such as averaging and recording peak values of waveform properties (i.e., frequency, voltage, etc.). The instruments are placed at the designer-selected test points to assist in characterizing the waveform behaviors. The primary Saber instrument types used during development and testing include:

- mave - Records the average voltage of the waveform
- mfreq - Records the average fundamental frequency of the waveform
- mpeak - Records the minimum and maximum peak voltage values of the waveform.

## 4.9.2 Parsing the Saber Output

The Saber Parse Component is responsible for processing and parsing the Saber-generated output files created during the design simulation, and subsequently initializing and populating the OAM databases with the parsed information. All information required to detect and characterize anomalous behaviors caused by the injection of a fault into the design is contained in the output files. The Saber-generated output files include the Baseline simulation file, Fault simulation file and the Netlist file.

Upon an initial analysis of the Saber simulation output files, we determined that some processing of the data would have to be performed before parsing the files. This involved normalizing certain types of expressions in the output to ensure that they were useable by the system. The process of normalizing the Saber simulation output files was accomplished using the Unix tools “awk” and “sed.” The “awk” and “sed” tools are very reliable specialized text manipulation tools that can match specific text patterns and perform global substitutions and manipulations.

The Saber netlist file contains a system-useable description of the circuit design, including a breakdown of all functional blocks and their associated nets. In addition, the netlist file also contains a listing of all Saber measurement devices (instruments) used during the simulation and the measurement points where they were placed.

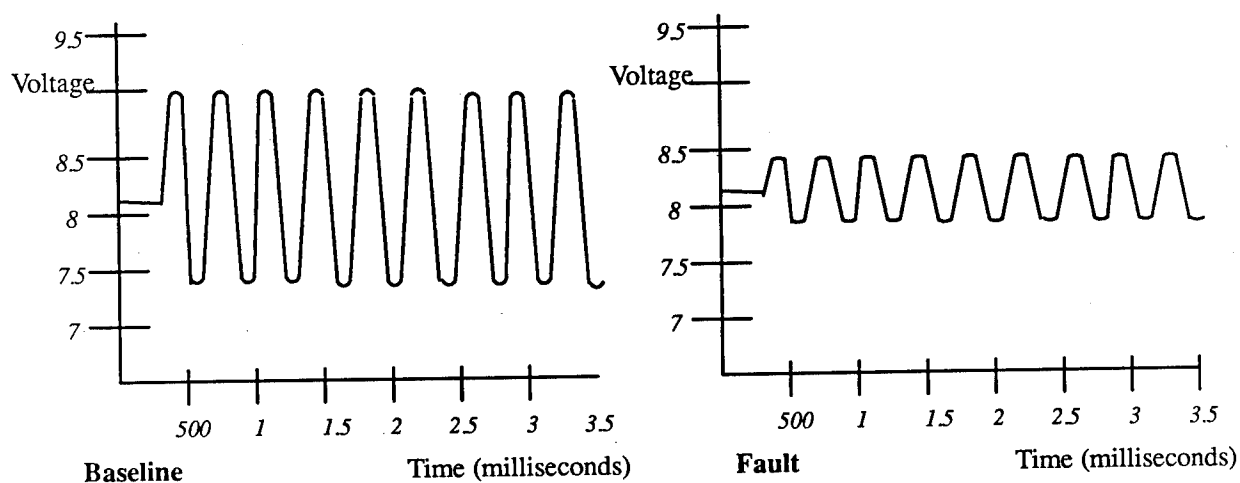
### **4.9.3 Fault Behavior Analysis**

The Fault Behavior Analysis component detects and characterizes any anomalous behaviors that may have been manifested during the simulation of the faulty circuit. In general, this is accomplished by contrasting the observed waveform samples from the simulation of the faulty circuit to those of the baseline simulation. Although the process of contrasting the faulted waveform against the baseline waveform can be fully automated, the system still requires outside direction regarding where to look in the design. This direction is provided through a loose coupling with the TSM. Tests and test points selected by designers during the testability function are first retrieved by the Fault Analysis Behavior software component. Test methods and net locations where the simulation waveform will be analyzed are subsequently determined based on this information.

From the beginning, we realized that automated fault detection would not be more efficient than manual fault detection in every case. Often, performing a manual waveform analysis of the faulty circuit behavior is quite simple. Any anomalies between the faulty and baseline circuit behaviors can be visually identified and characterized immediately by designers in an adhoc fashion. Displays provided by the simulation software (Figure 26) allow designers to view behaviors at design measurement points while the simulation is in progress. In Figure 26, it is immediately clear from the visual representation of the simulation waveform that the peak-to-peak voltage has reduced significantly from the baseline design. In an automated solution, however, the benefit of designer experience and level of visual awareness is not present in the same fashion. Depending on the test type (i.e. frequency, DC voltage) to be performed, the system must

perform a battery of evaluations on the waveform blindly until a specific anomalous waveform behavior is detected, or none are detected at all.

During the design phase of the Fault Behavior Analysis component, we carefully considered how to collectively associate a series of waveform evaluations with a particular test type while also allowing the waveform evaluations to be used interchangeably with different test types. In addition, the model had to be easily expandable to support future designs. In the end, we solved the problem using a multi-level solution consisting of an upper-level and lower-level.



*Figure 26 Time versus Voltage display for the baseline and fault circuits as viewed by the designer during the simulation.*

At the upper-level, fault tests are the top-level control functions of the Fault Behavior Analysis process for a specific test type. A fault test is responsible for knowing what anomalous waveform behaviors to look for and in what order they should be evaluated based on the specific test type in which it was designed to handle. For example, a fault test designed to handle the frequency test type (Saber "mfreq" measurement device), defines a procedure in which a series of waveform evaluations are executed to characterize the frequency of the faulty circuit's waveform.

At the lower-level is a system library of Fault Signature Identification utilities. A fault signature is a specific waveform behavior that deviates from the baseline (e.g., increased frequency, decreased voltage). A fault signature identification utility is a system library function that, given the baseline and faulty waveform data, contains the algorithm to detect a specific fault signature.

The following describes some of the FaultSignature Identification utilities implemented in the OAM's library of utilities.

**Voltage Doubled** -Determines if the voltage of the faulty circuit is twice the voltage of the baseline circuit. The Saber instrument type "mave" is used to detect this fault signature.

**Increased Value** -Determines if the faulty circuit waveform property has increased or decreased from the baseline waveform value. Frequency and voltage changes are detected using the "mfreq" and "mave" instrument types, respectively.

**One-Half Frequency** - Determines if the fundamental frequency of the faulty circuit waveform is approximately one-half the baseline waveform frequency. The Saber instrument type "mfreq" is used to detect this fault signature.

**Varying Output** -Determines if the waveform property of the faulty circuit is variable over the duration of the simulation while the same baseline waveform property remains constant. Frequency and voltage changes are detected using the "mfreq" and "mave" instrument types, respectively.

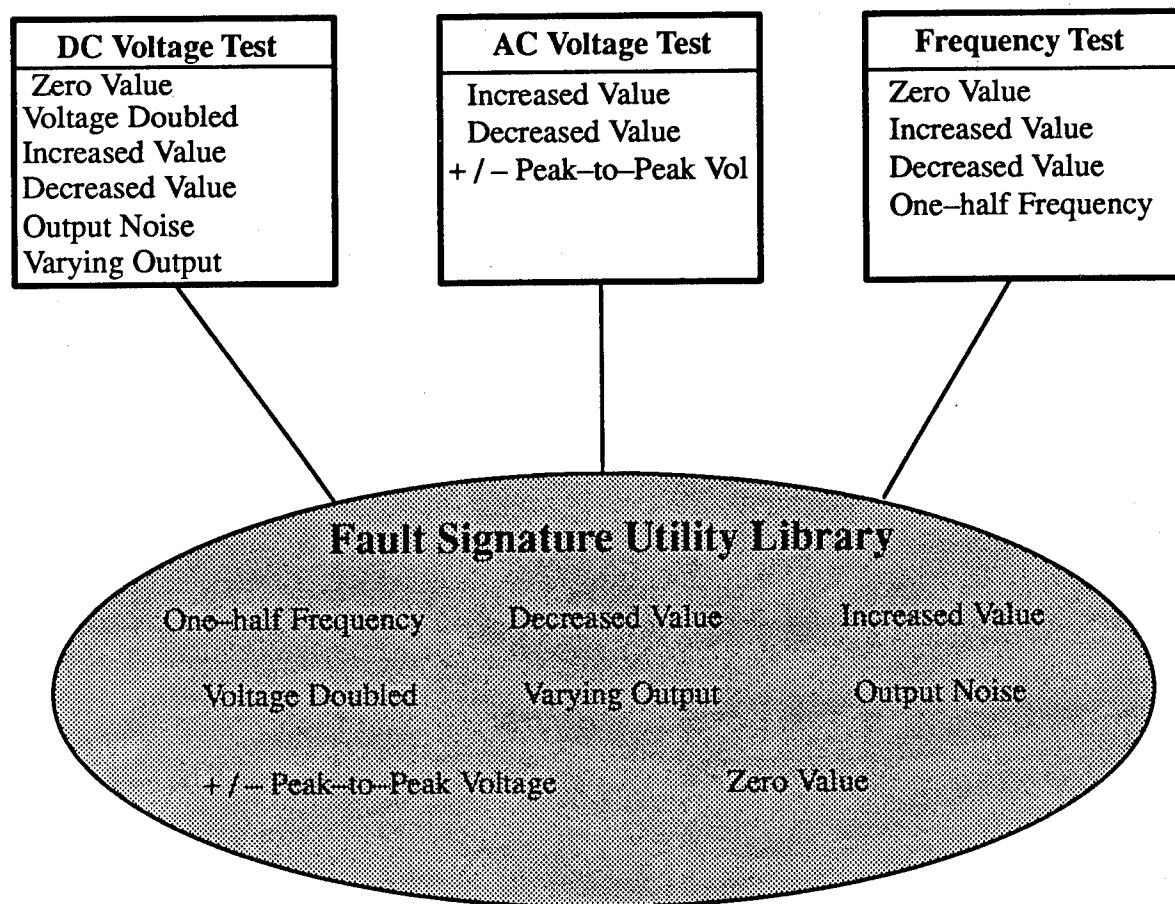
**Output Noise** - If high frequency noise was placed on the rectifier input, this utility determines if the noise was not filtered out and is still present on the rectifier output. The Saber instrument type "mave" is used to detect this fault signature.

**Increased Peak-to-Peak Voltage** - Determines if the minimum and maximum peak-to-peak voltage of the faulty circuit's waveform has increased from the baseline circuit's peak-to-peak voltage. The Saber instrument type "mpeak" is used to detect this fault signature.

The key advantage of this architecture is that it supports the easy expansion of the system's fault behavior detection capabilities. At any given time, the system's fault behavior detection capabilities are represented by the extent of the library of Fault Signature Identification utilities. The architecture allows the library of utilities to be easily expanded to reflect the fault-to-failure-mode maps of new circuit models. Upgrading the system's capability to identify a new faulty

behavior involves simply attaching a new Fault Signature Identification utility to a current fault test.

The basic software architecture diagram of the Fault Behavior Analysis component is presented in Figure 27.



*Figure 27 General software architecture of the Fault Behavior Analysis component.*

#### 4.9.4 Fault Isolation

The Fault Isolation software component conveys any fault signatures detected during the Fault Behavior Analysis phase to the casebase of faults. It retrieves match score information from the test coverage view of the casebase about possible matches and attempts to isolate the fault signatures to a specific fault in the casebase using the information. If the fault cannot be isolated, additional



test points are defined. Following another simulation run and Test Coverage execution, the Fault Isolation component will attempt to isolate the fault again.

To isolate a fault means to pinpoint the function and type of component that caused the overall design to function abnormally. In the simplest case, one fault signature will be unique to a single fault in the casebase. As the system's casebase grows, the probability of this occurring becomes less likely. In most cases, detections of multiple fault signatures will be required to isolate the behavior to a unique fault. These are called multiple masking faults. Often, the set of fault signatures matches multiple faults in the casebase, creating ambiguity that must somehow be managed. The majority of our design effort for this component focused on developing a scheme to manage the ambiguity caused by multiple fault matches.

The most significant requirement for this component was to define a possible set of criteria for identifying one fault match over another. The two primary criteria to accomplish this were the association of multiple fault signatures to a fault match and match score. The procedure for using the criteria is described in the following paragraphs.

First, upon retrieving all casebase matches for a set of fault signatures, the matches are analyzed to determine if all or a majority of the matches can be associated to any common faults. If so, the fault with the greatest number of observed fault signatures corresponding to casebase matches is selected as the most probable isolation candidate. Figure 28 shows a series of fault signatures, their casebase fault matches, and how associating multiple signatures to faults eliminates the less likely match candidates.

In the example, five fault signatures were detected by the Fault Behavior Analysis component and sent to the casebase. The matching results are shown in the figure.

In the example, "Open Diode" is the most likely candidate for fault isolation because the greatest number of observed fault signatures could be masked to it.



Open Diode	Open Inductor	Shorted Resister	Open Capacitor
Fault Signature 1	Fault Signature 1	Fault Signature 3	Fault Signature 4
Fault Signature 2	Fault Signature 2		Fault Signature 5
Fault Signature 3	Fault Signature 3		
Fault Signature 4			
Fault Signature 5			

*Figure 28 Example of multiple fault signature associations to a single fault.*

If all or the same number of fault signatures can be masked to more than one fault candidate determined during the first step, the next step is to try to isolate the fault based on the average match score. The average match score is simply the overall average of the scores for all detected fault signatures that correspond to a particular fault candidate. At this point, the most probable fault isolation candidate of the group is the candidate with the greatest average match score. If the average match scores for two or more remaining fault isolation candidates are equal, the system cannot isolate the fault based on the given fault signature detections at this time. If this is the case, the Fault Isolation component queries the casebase for any additional tests and test points associated with the fault isolation candidates and reports them to the user. The user can then execute additional simulations using the recommended tests and test points and re-execute the OAM, allowing the the Fault Isolation component to make another pass at the problem using any new detected fault signatures.

In addition to resolving multiple fault match ambiguity, the average match score is also provides feedback pertaining to the overall strength of the match or confidence level. If a unique fault can be successfully isolated by the system, a "Fault Confidence Level" is then accessed by the system and subsequently reported to users. The four levels of fault confidence and their associated average match score ranges are PERFECT: 1.0 (100%), HIGH: 0.8 – 0.99(80% – 99%), MED: 0.6 – 0.79 (60% – 79%) and LOW: 0.4– 0.59 (40% – 59%). If the match score received from the casebase is less the 0.4 (40%), one or more of the weighted casebase attributes is significantly different enough to justify discarding the match altogether.

### **4.9.5 Update Mapping Casebase**

If the Fault Isolation component fails to discern the failure, the Update Mapping Casebase component provides a vehicle for users to assist the system by allowing them to enter new fault cases into the casebase that can be used to further isolate the fault. The system recommends when and if user assistance is required to isolate the fault; however, activating the Update Mapping Casebase component is manually initiated by users. After the user inputs the new case, the Update Mapping component then updates the casebase to permanently reflect this new information. The Update Mapping Casebase interface can also be used at any time to augment the casebase, regardless of whether a request has been made by the system to do so. The input required by the Update Mapping Casebase component to create a new fault case is fault name, functional block, signature behavior, measurement point and measurement device.

## 5. Lessons Learned

The basis for the R&M Design System was to build a proof-of-concept system for applying knowledge-based systems to testability, reliability, and maintainability (TR&M). The major portion of this contract was to come up with the ideas, and to implement them if possible. This section details what we learned over the course of the contract.

### 5.1 Working with the Tools

The R&M Design Expert System was designed to work with Mentor Graphics' Design Architect™, DFI™, and Design Viewpoint Editor™; ART-IM™; Analogy's Saber™; RL-ORACLE and SunPC™. We expended major effort to extract or insert information into the various tools. The interaction and extraction of the information was required before the main effort could be undertaken. The interaction and extraction had to be re-checked after every new upgrade due to the differences between releases. The interface with Mentor required the most amount of work. Interfacing with the tools was difficult and very time consuming.

At the beginning of the R&M Design Expert System projects, we made assumptions that greatly affected how the interaction with the Mentor tools would be undertaken. The main assumption was that the design would be in a functional hierarchy to get the most out of the system. Correctly building a hierarchical design in Mentor requires a large amount of effort from the beginning. We found with the AstroSpace design that if the design is not in a hierarchical form, it is not easy to convert it to the hierarchical form. In fact, we were not able to convert the design in the scope of this effort.

However, the functional (or hierarchical) representation is needed for each of the sections of the program. While the reliability section could function at some level, the testability and test coverage portions are severely handicapped by the lack of hierarchical design.

Another drawback of the system design is the amount of time needed by users to set up the system to work properly. In addition to setting up the hierarchical design, each component needs to have properties added to it. This could be accomplished by setting up specialized commands to add properties to the components,

but the values still needed to be added. The viewpoint needs to be established correctly, and the properties need to be added to the viewpoint as well.

For us to automate the data extraction and analysis, we had to institute a strict naming convention. This enabled us to recognize the different functional blocks over designs, as well as recognize certain properties about the designs. This naming convention includes the functional blocks, components, and some properties contained on the components.

Our only concern with the Saber simulator is the time required to simulate the design, especially in a hierarchical format. For our simulations, we flattened the design and simulated in that manner. Simulations still took hours to run.

## 5.2 Using Casebase Reasoning Approach

The casebase reasoning (CBR) approach to this problem worked very well. We developed a methodology for the fault representation that fit very well with the representation required by the CBR tool. This methodology was the Fault-to-Failure-Mode map. This map represents the faults found in the power supply. The concept of the casebase was to make a generalized representation of each fault. Instead of having "98% increased mpeak on net 45," we represented this fault as "increased peak to peak voltage at the system output." With this representation, we were able to build up the casebase from design to design.

This generalized approach enabled us to use the casebase from design to design. Thus, what is learned from each design does not need to be discarded. If the designers follow a convention with their designs, the system may be able to "pick up" the conventions by the cases that are added by the engineers. This allows the system to "learn" over time. This allows a more experienced designer to use it in the beginning to populate the system; less experienced designers would be able to use it to improve their designs.

With our casebase system we developed the two different viewpoints needed for testability and test coverage. The viewpoints work on related data in opposite directions. In other words, the testability viewpoint is used to discover a test point and test equipment to form the functional block and fault. The test coverage portion uses the test point, test equipment, and signature behavior to discover the functional block and fault. It is theoretically possible that additional viewpoints could be conceived that would cover maintainability, or any of the other metrics as described in the "Supportability Analysis" section of this report.

One drawback of the casebase approach is that each possibility must be in the casebase or the system will not be able to make any recommendations. Although the casebase matches can match on cases that are similar, though not exact, at times this is not much help. For example, if you are looking for a test and test point for a shorted diode in the input filter, it may not help to know the tests and test points for an open diode in the input filter, or for a shorted diode in the transformer. If a fairly robust casebase existed it would solve this problem.

A better approach may combine CBR with a knowledge-based system. The casebase could use experience to recommend solutions; if that was impossible, the knowledge-based system could extrapolate and recommend. Currently, we use a knowledge-based system with the Operational Analysis Module (see discussion in the following paragraphs).

### 5.3 Testability

We initially designed the testability module of the R&M Design Expert System to both recommend test points to users and evaluate testpoints that were provided by users on the Mentor design. However, after our experiences with Mentor's Design Architect, we could not develop a good way for users to represent the test point, and tests for the system. A possibility would be to use the nets properties for this. We could not pursue this concept due to the scope of the contract.

We also designed the system to alter the design to inject the fault or to use faulty models of the components during simulation to represent the fault. We were unable to do either of these alternatives.

Two possibilities for altering the design to inject the fault are: alteration within Mentor, and alteration of the netlist generated for the design. Altering the design via Mentor is not easily achievable due to the interface required. The system would have to select a particular component from the general class (e.g. resistors), then alter the value of the component, and then insert the desired instruments at the appropriate place. Altering the component value is the easiest of these jobs. To do this requires an appropriate viewpoint on the design, back annotation, and attachment of the back annotations to the design for the alterations to take place.

Analogy planned to release faulty component models during 1994. Due to their high cost, we were not able to investigate this track. It is possible that these faulty models would be an excellent way to model the design with the various parts.

## 5.4 Degrees of Fault Isolation

Initially during this program we hoped to isolate multiple faults and single faults. This isolation proved to be extremely difficult using the casebase approach. The signature behavior of multiple faults varies dramatically from a single fault. Diagnosis of multiple faults is ten times more complicated, because one fault may mask or distort another fault. The casebase approach does not lend itself to diagnosis of multiple faults, due to the number of permutations that would be required to diagnosis the faults. In this situation, a hybrid case-base, knowledge-based system or straight knowledge-based system would work better.

It is important to note that many highly complex diagnostic algorithms have been developed in the industry, none exceedingly effective, in order to diagnose multiple faults. The lack of effectiveness is due to the tendency for a second fault to either mask a first fault, compound the signature of the first fault, or results in almost total inoperability of the system. Such multiple-fault diagnostic systems are highly dependent upon their own structure. For example, a system which test the power supply functionality first may not have to be concerned with all future possible combinations of power supply failures and other system failures. Conversely, if the system did not test the functionality of the power supply system early in the diagnostic process, then it always remains a suspect element in subsequent diagnostics.

Given that the diagnostics of a multiple fault system is so highly complex, and dependent upon its own structure, it clearly follows that the predictability of the diagnostics of a multiple fault system is even more suspect. With present technology capabilities, the most effective manner in which to avoid the cost of multiple faults is to partition functional elements into replaceable elements, and avoid wherever possible an overlap of a single function onto several physical items. If there is a high degree of fault isolation among the replaceable elements. In such an instance, it will be less likely for multiple faults to interact with each other in the adverse conditions described above.

It should also be noted that it is not necessary for a well-supportable system to be capable of testing in the presence of multiple faults. As systems become more complex, and individual components more reliable, then the functions which may fail can be more separated more readily, facilitating more efficient supportability. For example, in order to test the stabilization system of a tank turret of 1960's vintage, it was necessary to operate the system over terrain and observe its ability to maintain target designation. In a system such as the M1A2, the stabilization system is compartmentalized into several subsystems which may be readily tested.



## 5.5 Importance of Standard Naming Conventions

Perhaps one of the most critical lessons we learned during the development of the Saber Parse component was the importance of forcing the designer to adhere to standard naming convention with regard to the circuit's functional blocks and netnames. This convention enables the system to make the association between the design-specific functional blocks and the more generic functional block names used to designate measurement points in the casebase. Currently, the functional block names can vary and still be associated by performing a simple string analysis. This is accomplished by comparing the designer-assigned functional block name to a series of abbreviated terms known to be associated with the generic functional block names used in the casebase of faults. If the difference between the two is too great, the system will not be able to correlate design functional blocks to casebase functional blocks. This seriously limits the ability of the casebase to match and provide an accurate score for detected fault behaviors.

The designer should also adhere to a standard naming convention with netnames utilized during the simulation phase whenever possible. In general, it is difficult to use a standard naming convention with netnames because a net can be attached to many functional blocks, causing much confusion as to which functional block or blocks to make the name indicative of. In this implementation, we only needed to enforce the standard naming convention on netnames representing inputs and outputs for the entire circuit design because the functional blocks in which system inputs and outputs are located may vary from design to design. Therefore, the name assigned to the net by the designer must be apparent enough that the system is able to deduce if it corresponds to a system input or output.

Specifically, any naming convention used for netnames should be indicative of the major functional block in which it is located and whether it is at the input or output of the block. For example, if a system output is a positive5-volt charge and is at the regulator output, the net should be assigned a name similar to "reg\_+5\_out." It would probably be a good idea to adopt a standard naming convention for every netname simply for the benefit of anyone who must work with the design; however, in this particular case, system requirements mandate that they are only needed for netnames located at the circuit's inputs and outputs.

## 5.6 Simulation Software Version or Vendor Changes

Because the system relies on the ability to parse a specific file format to populate its internal databases, any changes in the software tools used to create those



files will most likely have a serious impact on the system itself. The type of change that could adversely effect the system could be a major change, such as a brand change, or a simple change, such as a software version upgrade. Either way, the procedures involved with automatically reading a particular software vendor's output are so specialized that any change would probably result in a major re-implementation of the system's parsing software components. In the current implementation, Analogy's Saber version 3.2 is used to generate all simulation output and the netlist file.

## **5.7 Simulation Accuracy Versus System Performance**

There is potential for a significant trade-off between the accuracy of the observed circuit behavior measured during a simulation and the performance of the expert system in both parsing the simulation output and detecting faulty behaviors. During a simulation, a more accurate representation of the circuit's behavior is obtained because the duration and sampling rate of the simulation are increased. A longer simulation time typically means that there is a greater probability that the observed behavior toward the end of the simulation represents the true circuit behavior. This helps to assure that the waveform is free of inconsistencies caused by system power-up fluctuations and other variations that may be observed prior to the waveform reaching a steady-state. A greater sample ingrate means that the waveform is being measured more frequently, thus providing a more accurate representation of waveform characteristics such as shape, frequency and voltage. As both duration and sampling rate are increased, the resulting simulation output file sizes are increased significantly. Ultimately, this will have a great effect on the expert system's performance of both the parse and fault detection process.

The most significant effect on system performance will be on the output file parse portion of the system. Increases in file sizes of up to 1 megabyte can result in the total duration of the parse process increasing by up to five standard minutes. In addition, the fault detection process may be similarly effected, but at a lesser magnitude, because all waveform observations past the power-up fluctuations should be evaluated to ensure that any faulty behaviors accessed by the system are consistent.

As simulation accuracy variables increase, there is a clear point that can be reached in which system performance is effected to the extent where it may be more efficient for the designer to perform processes manually that are currently automated by the expert system.

## 5.8 Casebase Continuity

Maintaining continuity regarding expressions used in the system's casebase is critical to the fault diagnosis process. During the fault diagnosis process, test, testpoint, and signature behavior (e.g., "increased peak to peak voltage," "increased ripple on output") information is conveyed to the casebase and a score is computed representing how closely the information matches that of a particular fault in the casebase. The match score is weighted, with the greatest amount of significance placed on the signature behavior. It is essential that all signature behaviors of the same type be named consistently among all fault cases in the casebase in which they are associated. Any lack of continuity in the signature behavior descriptions may result in erroneous faults being matched by the CBR portion of the system.

An example of discontinuity is if a designer adds a new fault to the system's casebase with the signature behavior type "DC voltage too low" and the standard expression used for other cases with a signature behavior of that type is "reduced DC voltage." The fault detection and isolation portions of the system detect signature behaviors in the design and convey them to the casebase using the standard expression. The potential problem exists when the CBR portion of the system attempts to match the standard signature behavior expression to the inconsistent one and it receives a very low match score. Most likely, due to the low score, the fault isolation portion of the system will disregard the particular casebase fault as a candidate for isolation altogether.

To avoid this potential problem, future developments of systems such as this should include a specialized graphical interface for casebase insertion that allows all case slot entries to be mouse-selectable from a list of standard expressions. This would enforce a continuity policy for casebase expressions even as designers who use the system change.

## 5.9 Designer Versus Automation

There were several clear short-term trade-offs involved with an automated approach to the problem. The most significant trade-offs were in fault detection and diagnosis. The main advantage of the automated approach is avoiding routine mistakes and oversights made by designers during the fault detection process. This can be attributed to both the system-aided test selection and automating the fault detection process itself. In the case of a less experienced designer, this would be an even greater advantage because the designer would immediately gain the experience of the system's knowledge base for test selection and fault detection and isolation.

There is a disadvantage, however, for more experienced designers. More experienced designers often have knowledge and experience that may not be accounted for in the system's knowledge base. In an automated approach, the tendency by users will most likely be to rely on the system's recommendations, when in fact there are probably some cases where their own experience would better serve the problem at hand.

Another consideration when evaluating the effectiveness of an automated approach is the issue of system performance time versus designer performance time. Of considerable issue in the fault detection process is the time that it takes for the automated system to read in the simulation results and subsequently perform its series of tests until a fault signature is detected. Often, a designer could have identified the fault signature during the the simulation itself, simply by monitoring the displays of the plotted waveform that are typically provided by the simulation software. Of course, the extent to which system versus designer performance time is an issue is also proportional to the experience of the designer.

## **5.10 Simulation and Fault Diagnosis Software Integration**

During the development of the Operational Analysis portion of the system, we realized that commercial simulation software and the fault detection portion of the expert system are well suited for future integration. This would eliminate system performance drawbacks caused by having to parse large simulation output files. The live simulation waveform would be processed and analyzed, which would eliminate the need to parse the output files. In addition, eliminating the output parse requirement would also minimize the system's memory requirements. Currently, large simulation output files must be maintained in memory for the fault detection and isolation processes.

The merger of the two would also provide a higher level of feedback to the designer during the simulation phase. Expected behaviors could be detected almost instantaneously as they occur, thereby limiting the overall simulation time as well.

Another area in which the integrated software offers improvement is in the creation of the circuit's Fault-to-Failure-Mode Map. When the functionalities are combined, all information necessary to create the circuit's Fault-to-Failure-Mode Map is available in one system. Therefore, Fault-to-Failure-Mode Maps for new designs could be automatically generated.

## 5.11 Measurement Points: Specific to General

To guarantee that the system's casebase of faults is applicable to many designs, only generalized measurement points are used. All measurement points in the casebase are expressed at the functional block level of the design. Design-specific measurement points used during the simulation are represented by nets on the design schematic. On a typical design, multiple nets are located at the inputs and outputs of the circuit's functional blocks. As a result, a general measurement point in the casebase (e.g. rectifier output, regulator input) will correspond to more than one design-specific net location and vice versa. This one-to-many relationship can pose a significant problem when the designer attempts to select design test points for the simulation. The designer can either arbitrarily select a specific net, a set of nets, or all nets. During our development process, we selected all nets corresponding to the general measurement point location to ensure that any expected faulty behavior was not overlooked.

This also has the potential to cause problems in the test coverage portion of the system. There were several cases where we observed contrasting waveform behaviors among net locations at the same general measurement point. If the multiple signature behaviors are not accounted for in the casebase this will introduce additional ambiguity into the process of fault isolation.

## 5.12 Effects of Casebase Expansion

Expanding the system's casebase significantly impacts the performance of the fault isolation portion of the system. As the system's casebase of faults is augmented to reflect new faults and faulty behaviors, the level of ambiguity caused by the matching of multiple faults during the fault isolation process is increased. The results is that additional simulations using different tests and test points may be required to isolate the behavior to a single fault.

## 5.13 Manipulation of Saber Output Files

Automating the fault detection and isolation processes requires that general design makeup information and all simulation results be available in a system-extractable form. We reached the goal, but not without having to go to great lengths. In general, the most difficult part of the process was to first transform the output into a standard, system-extractable format. This required a significant amount of effort to define and implement a data normalization policy.

The greatest need for data normalization was with the observed waveform data. Throughout the Saber simulation, waveform observations are measured at regular time intervals. Depending upon the magnitude of the measured value, the observations are reported at varying units of measure and written to the output files. In addition, Saber outputs the waveform measurement and the alpha-numeric symbol representing unit of measure as one expression. While this expression would be easily understood by the designer, the concatenation of a numeric value and character symbol is unusable to an automated system. The data normalization process was required to scan the Saber-generated output files, separate all measuring unit symbols from numeric values wherever present, and insert a default value following numeric values without a measuring unit symbol to preserve data format consistency. By separating the measurement unit symbols before parsing the data, numeric values could be easily standardized to a common measurement unit upon being parsed by the system.

In general, the normalization process has a very little effect on the overall format of the file itself. It is merely concerned with normalizing individual expressions in the file that will be needed by the system.

## **5.14 System Resource Requirements**

Using a loosely coupled approach to information sharing between software modules can be quite expensive in the area of system resources. First, there is the time required with exchanging information between modules through files. When results from another software module are needed to perform a function, the files containing the information must first be formatted, in some cases, and then parsed by the system. This adds significant overhead to a software module's time in performing its particular system function.

System memory resources are also greatly impacted by a loosely coupled information sharing approach. Large simulation output files (megabytes) must be maintained in system's memory for the entire period they are needed for a particular circuit design. One possible solution to this problem in future developments is integrating commercial design simulation software with the fault detection portion of the expert system, as discussed earlier in this section.

## **5.15 Conclusions**

Simulation for analog testability analysis is years away. This is not surprising since full simulation of analog circuits is still not currently performed. At most, particular situations with specific inputs and ambient conditions are analyzed.

However, when one considers all of the possible variables (temperature effects, design margin, losses,...), and add to these variables the possibility of every stage having several failure modes, it is easy to see why the problem compounds.

It is recommended to focus, during the near-term, on automated reliability analysis, since this may be relatively easily achieved by integrating the 217 databases with parts, lists, and, at the next stage with thermal analysis.

## 6. Glossary

This glossary contains terms that you will encounter when dealing with the R & M Design Expert System, and the related software.

### ART-IM™

ART-IM™ is a knowledge-based tool by Inference Corporation.

### Back Annotation

Back Annotations are edits made to properties on the design. These edits are contained in a separate back annotation object which may be attached or unattached to the design. The values in the back annotation object will only be visible when it is attached to the design.

### Baseline

The baseline is the circuit without any faults injected into it. The term is mainly used to refer to a simulation of the circuit with no faults.

### Casebase

A casebase is similar to a database of information associated by a specific situation or condition. Casebase may be indexed by specific features.

### Component

A component is a single simple part of the design of which functional blocks are constructed of. For example, a resistor.

### Design Architect

Design Architect is Mentor Graphics Schematic and Symbol Editor. See the Mentor Graphics Design Architect User's Manual and the Design Architect Reference Manual for more information.



## Design Hierarchy

The design hierarchy is a method of arranging the components of the design into functional descriptions. Levels of description are built upon one another from the most generalized to the least. Building the design hierarchy in this manner is synonymous with the top-down design methodology.

## Design Viewpoint

A design viewpoint is an object through which we “view” the design. The design viewpoint filters the information in the design database to those pieces of information that are of interest to a particular application.

## Design Viewpoint Editor (DVE)

The Design Viewpoint Editor (DVE) is used to alter information regarding the design viewpoint. DVE allows for adding, deleting and modifying visible properties. Back Annotation objects can also be attached and unattached through DVE. For a more complete description see Mentor Graphic's Design Viewpoint Editor User's and Reference Manual.

## Fault

A fault is a component in the system that is not functioning correctly.

## Fault Mode

A fault mode is a manner in which the user may select a fault to inject into the circuit for evaluation.

## Fault Signature

A fault signature is measured behavior of the circuit at a particular test and testpoint.

### **Faulty Component**

See Fault.

### **Functional Block**

A functional block is a segment of the electronic design that comprises of a single function. For example an input filter.

### **High Level Component**

In the Mentor Graphics design environment, a high level component is a symbol that represents a functional block.

### **Low Level Component**

In the Mentor Graphics design environment, a low level component is a symbol that represents a simple part (e.g. a resistor).

### **Netlist**

A netlist is a listing of the components, the connectivity and values that they have. The netlist is used as an input for simulation programs.

### **Path**

A path is Unix terminology for the listing of the directories and subdirectories to describe the location of the file in the large hierarchy of subdirectories in the system.

### **Property Owner**

Property Owners "own" properties. The owner of a property is said to be that component to which the property is attached. See Mentor Graphics Design Architect User's Manual for more information.

## Properties

Properties are associated with components in the design. Properties hold information which describes characteristics of the design which are not apparent from the schematic alone. When a property is registered in the design viewpoint they become visible properties (see visible properties).

## Reliability

Reliability analysis in the R & M Expert Design System comprises of a functional reliability measure computed with MIL-HDBK-217E standards.

## RL-Oracle

An implementation of MIL-HDBK-217 in obtaining failure rates of electronic piece parts and reliability characteristics of electronic equipment/systems.

## Saber

Saber is a analog simulation tool which can be accessed directly from the Mentor Graphics design environment.

## Schematic

A schematic is created by the Schematic Editor in Design Architect, by default, it has one sheet. Schematic contains the components that define the design. See Mentor Graphics Design Architect User's Manual for more information.

## SunPC

SunPC is a DOS simulator for Sun Microsystems. It is used in the R & M Design System to execute RL-Oracle on the same platform as the system.

## Symbol

Symbols are create by the Symbol Editor in Design Architect. The symbol is design object which graphically represents a component. It consists of symbol body, sym-

bol pins, and symbol properties. See Mentor Graphics Design Architect User's Manual for more information.

## Test

A test is a measurement device that is used to detect anomalous readings at a point in the circuit.

## Testability

Testability analysis in the R & M Expert Design System comprises of an analysis of the tests and testpoints for a given fault in the design.

## Test Equipment

Test equipment refers directly to the physical equipment that is needed to perform a test.

## Testpoint

A testpoint is the point in the circuit at which a test is placed.

## Threshold

The threshold is a value that is utilized by the Test Coverage portion of the R & M system that is used to determine the presence of a signature behavior.

## Visible properties

Visible properties are properties in the design that have been registered in the design viewpoint. These are not to be confused with properties that are *visible* when looking at the design sheet. Visible properties are available for use by other tools in the Mentor Graphics suite of tools.

## **7. Appendix**

### **7.1 Appendix 1: Deliverables**

#### **Technical Reports:**

1. Technical Report, Phase I
2. Technical Report, Phase II
3. Final Technical Report (this document)

#### **2167A Documents:**

1. Software User's Manual
2. Software Design Document
3. Software Test Plan

Monthly Status Reports, 36 months.

Software implementation of proof-of-concept system.

## 7.2 Appendix 2 – References

### *Authored Documents*

- [1] Hartwell, Robert E., “Testability for All Seasons”, Robert E. Hartwell, General Electric Areospace, Automated Systems Department, Internal Document. 1989.
- [2] Johnson, Michael, L., “Desk–Top Computer Database Technique for Integrated R&M Analysis”, in Proceedings, Annual Reliability and Maintainability Symposium, 1990.
- [3] Kececioglu, Dimitri, “Reliability Engineering Handbook”, 1991, Prentice–Hall.
- [4] Vaccaro, James M., Caroli, Joseph A., Lyne, George W., and Richards, Dale W., “R&M Software Tools: Their Importance to Concurrent Engineering”, USAF Rome Laboratory, undated.
- [5] Various Authors, “System Engineering Practice (SYP), Part 3: Specialty Engineering, Practice 3.1: Reliability”, General Electric Areospace, Automated Systems Department, Internal Document. 1988.

### *Military Standards and Handbooks*

- [6] MIL–STD–721C, Definitions of Terms for Reliability and Maintainability. 12–June–1981.
- [7] MIL–STD–756, Reliability Modeling and Prediction
- [8] MIL–STD–785, Reliability Program for Systems and Equipment Development and Production
- [9] MIL–STD–1629, Procedures for Performing a Failure Mode Effects and Criticality Analysis
- [10] MIL–STD–2165, Testability Program for Electronic Systems and Equipment, 26 January 1985.
- [11] MIL–HDBK–338–1A, Electronic Reliability Design Handbook. 12–October–1988.
- [12] MIL–HDBK–472, Maintainability Prediction

[13] MIL-HDBK 217E, Military Handbook: Reliability Prediction of Electronic Equipment. 15-January-1982. Additionally, Notice 1 of change to MIL-HDBK 217E, 2 January 1990.

### *Product Literature*

[14] Systems Effectiveness Associates (SEA) product literature / documentation.

[15] Innovative Software Designs (ISD) product literature / documentation.

[16] Cadence product literature / documentation.

[17] Management Sciences, Inc (MSI) product literature / documentation.

### **CAD Vendor Addresses**

Analogy Inc., 9205 SW Gemini Drive, PO Box 1669, Beaverton, OR 97075. (503) 643-3361

Cadence Design Systems Inc., Systems Division, 555 River Oaks Parkway, San Jose, CA 95134. (408) 943-1234

Dynamic Soft Analysis Inc., 313 Cuyasuta Road Pittsburgh, PA 15215. (412) 781-3016

Innovative Software Designs Inc., (410) 788-9000

Management Sciences Inc. (MSI), 6022 Constitution N.E., Albuquerque, NM 87110. (505) 255-8611

Mentor Graphics Corporation (MGC), 8005 S.W. Boeckman Rd, Wilsonville OR 97070. (503) 685-7000

Systems Effectiveness Associates (SEA), 20 Vernon Street Norwood, MA 02062. (617) 762-9252

Valid Logic Systems (see Cadence Design Systems, Inc)



# Index

## A

ART-IM™, 92

## B

Back Annotation, 92

Background, 4

    Design Process, 5

    Specification, 4

    Test Point Selection, 7

Baseline, 92

BETASOFT, 18

## C

Casebase, , 68, 92

    Views, 68

        Test Coverage, 70

        Testability, 69

CHEAP, 23

Component, 92

## D

Design Architect, 92

Design Hierarchy, 93

Design Viewpoint, 93

Design Viewpoint Editor, 93

## F

Fault, 93

Fault Mode, 93

Fault Signature, 93

Fault-to-Failure-Mode Map, 65

    Creating, 66

    Fault Isolation, 67

    Testability, 66

Faulty Component, 94

Functional Block, 94

## **G**

GADDS, 22

Glossary, 92

## **H**

High Level Component, 94

## **I**

Implementation, 61

Interfacing

Design, 63

Mentor Graphics, 63

Operational Analysis Module, 71

Databases, 72

Fault Behavior Analysis, 74

Fault Isolation, 77

Reliability Assessment Module, 70

Test Selection Module, 71

Introduction, 1

Program Scope, 1

Report Overview, 3

System Concept, 3

## **L**

Lessons Learned, 81

Low Level Component, 94

## **M**

MEAP, 20

## **N**

Netlist, 94

## P

Path, 94  
PC-FMEA, 21  
PC-MAINTAINABILITY, 20  
PREDICTOR, 17  
Properties, 95  
Property Owner, 94

## R

RAMCAD, 23  
REAP, 16  
RELEX, 17  
RELEX FMECA, 21  
Reliability, 5, 95  
Risk Spectrum , 18  
RL-ORACLE, 15, 95

## S

Saber, 95  
Schematic, 95  
STAT, 22  
SunPC, 95  
Supportability Analysis, 38  
    Maintainability, 41  
    Maintainability Metric, 45  
    Reliability, 40  
    Reliability Metric, 42  
    Reliability/Maintainability Metric, 48  
    Reliability/Testability Metric, 46  
    Testability, 41  
    Testability Metric, 42  
    Testability/Maintainability Metric, 49  
    Testability/Reliability/Maintainability Metric, 49  
Survey Of The Tools , 12  
    FMECA, 20  
    Maintainability, 18  
    Reliability Analysis, 13  
    Testability, 22  
Symbol, 95

System Concepts and Methodologies, 38

System Modules, 56

Fault Mode Mapping Module, 57

Operational Analysis Module, 59

Reliability Assessment Module, 56

Test Selection Module, 58

## T

Test, 96

Test Equipment, 96

Testability , 6, 96

Automated Test Equipment, 10

Testability Process, 27

Diagnostic Development, 30

Fault Coverage, 28

Fault Coverage Database, 31

Integrated Product Development, 34

Issues, 27

Model Development, 33

Testpoint, 96

The Concept, 50

Required System Inputs, 54

THERMAX , 16

THERMOSTATS, 16

Threshold, 96

## V

VIABLE, 17

Visible properties, 96

***MISSION  
OF  
ROME LABORATORY***

**Mission.** The mission of Rome Laboratory is to advance the science and technologies of command, control, communications and intelligence and to transition them into systems to meet customer needs. To achieve this, Rome Lab:

- a. Conducts vigorous research, development and test programs in all applicable technologies;
- b. Transitions technology to current and future systems to improve operational capability, readiness, and supportability;
- c. Provides a full range of technical support to Air Force Materiel Command product centers and other Air Force organizations;
- d. Promotes transfer of technology to the private sector;
- e. Maintains leading edge technological expertise in the areas of surveillance, communications, command and control, intelligence, reliability science, electro-magnetic technology, photonics, signal processing, and computational science.

The thrust areas of technical competence include: Surveillance, Communications, Command and Control, Intelligence, Signal Processing, Computer Science and Technology, Electromagnetic Technology, Photonics and Reliability Sciences.