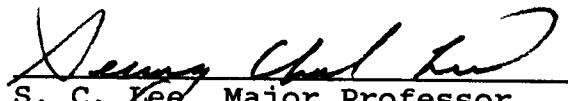


To the Graduate Council:

I am submitting a thesis written by Michelle Weisheit Ratliff entitled "Real Time State Assessment and Fault Diagnosis KBS for Space Power Systems." I have examined the final copy of this thesis for form and content and recommend that it be accepted in partial fulfillment of the requirements for the degree of Master of Science, with a major in Computer Science.


S. C. Lee, Major Professor

We have read this thesis
and recommend its acceptance:





Accepted for the Council:


Vice Provost and
Dean of The Graduate School

STATEMENT OF PERMISSION TO USE

In presenting this thesis in partial fulfillment of the requirements for a Master's degree at The University of Tennessee, Knoxville, I agree that the Library shall make it available to borrowers under rules of the Library. Brief quotations from this thesis are allowable without special permission, provided that accurate acknowledgment of the source is made.

Request for permission for extensive quotation from or reproduction of this thesis in whole or in parts may be granted by the copyright holder.

Signature Michelle Rathiff
Date July 22, 1991

Real Time State Assessment and
Fault Diagnosis KBS for Space Power Systems

A Thesis
Presented for the
Master of Science
Degree
The University of Tennessee, Knoxville

Michelle Weisheit Ratliff

August 1991

Copyright © Michelle Weisheit Ratliff, 1991

All rights reserved

DEDICATION

I would like to dedicate this thesis to my husband, Clifford L. Ratliff, who has given his love, understanding, and support during the many hours that has gone into this thesis, and to my parents, the late Lt. Col. Arthur Joseph Weisheit and Mariette Weisheit, and my twin brother, Daniel Weisheit, for their encouragement and support for anything I tried to accomplish.

ACKNOWLEDGEMENTS

I would like to thank my thesis advisor, Dr. Seung C. Lee for the patience and knowledge he continuously provided during the preparation of this thesis. I am also grateful to Dr. Lee for the opportunity to work on this research project and to co-author a paper resulting from this work. This research was funded in part by the NASA Marshall Space Flight Center from the NASA Grant NAG8-721 and by the Center for the Commercial Development of Space Power funds from NASA Grant NAGW-1192-CCDS-AL, Auburn University, and the Centers Industrial Partners.

I would also like to thank Dr. Moonis Ali and Dr. Bruce Whitehead for serving on my thesis committee and for their suggestions during the preparation of this thesis.

My thanks also goes to Calspan Corporation/ AEDC Division for financially supporting my thesis studies and allowing the time to further my education.

ABSTRACT

Research was conducted to develop a real time fault monitoring and diagnosis knowledge-base system (KBS) for space power systems. The fault monitoring and diagnosis knowledge-based system has been developed using the AMPS (Autonomously Managed Power System) test facility currently installed at NASA Marshall Space Flight Center (MSFC), however the basic approach taken for this research is rather generic and should be applicable for other space power systems. The developed KBS is entitled "AMPERES" (Autonomously Managed Power-System Extendible Real-time Expert System). A knowledge based control strategy and its supporting knowledge representation scheme were developed to effectively monitor and diagnose the operating state of AMPS which is monitored by various sensors. A sensor value validation method explores the embedded redundancies of the sensors in AMPS and maximizes their utilization in validating sensor values. A data structure called the Disturbance Interrelation Analysis Graph (DIAG) formalizes the inference scheme of the fault diagnosis task and explicitly shows the diagnosis reasoning paths. The design of AMPERES is presented with an emphasis on the DIAG formalism and implementation.

TABLE OF CONTENTS

CHAPTER	PAGE
I. INTRODUCTION	1
1.1 Objective of the Study	1
1.2 Previous Work	5
1.3 Overall Approach	9
II. ORGANIZATION OF AMPERES	14
2.1 Data Acquisition	14
2.2 Concurrent Task Handling	19
2.3 Knowledge Base Representation	20
2.4 Fault Diagnosis Module	22
III. DEVELOPMENT OF THE FAULT DIAGNOSIS SCHEME	25
3.1 Sensor Value Validation Scheme	25
3.2 DIAG Formulation	33
3.3 Fault Diagnosis Process Using DIAG	38
IV. RESULTS AND ANALYSIS	49
V. CONCLUSIONS	63
LIST OF REFERENCES	65
APPENDIX Fault Descriptions	68
VITA	72

LIST OF FIGURES

FIGURE	PAGE
1. Autonomously Managed Power System (AMPS)	3
2. FILSD Connected with Four Resistive Loads	11
3. Concurrent Task Management in AMPERES	13
4. Data Structure for Data Acquisition	15
5. Critical and Non-Critical Data Buffer Layouts ..	16
6. Critical Data Acquisition Process State Transition Diagram	18
7. Component Representation and Its Environment ...	21
8. Simplified One Line Diagram for AMPS	26
9. Causal Relation listing for AMPS	28
10. The Sensor Redundancy Graph (SRG) for Ammeters and Voltmeters Installed on AMPS	29
11. Validity Levels of the Sensors for AMPS	30
12. Part of DIAG (Disturbance Interrelation Analysis Graph) for Battery System Fault Diagnosis	35
13. Part of Set Current Context Routine	39
14. Semantic Primitive Functions in AMPERES	44
15. An example of Battery Cell Open Fault Rules	46
16. An example of Over Current Fault Rules	47
17. Display of AMPERES	52

FIGURE	PAGE
18. DIAG for the Load Connect, Load Disconnect, and Load Over Current Fault Diagnosis	55
19. DIAG for the Battery Ground Fault Diagnosis	57
20. DIAG for the Battery Short Fault Diagnosis	58
21. DIAG for the Battery Open Fault Diagnosis	59
22. DIAG for the Solar Array System Failure Fault Diagnosis	61

LIST OF TABLES

TABLE	PAGE
1. Test Cases Applied to AMPERES	50

CHAPTER I

INTRODUCTION

1.1 Objective of the Study

Electric power is a precious resource in space because of its extreme usefulness and strictly limited availability. It is essential to the operation of the crew members' life-support system as well as directly affects the successful achievement of the mission objective. For these reasons, electric power must be available on a continuous basis. Some of the specific problems that create difficulties in achieving highly reliable operation of the space power systems include the following:

i. Expensive crew members' manpower:

Space power systems must be continuously monitored throughout the mission period.

Maintaining this manpower in space, or even on the ground as an alternative, is very expensive.

ii. Difficulties in accumulating expertise in operating the space power system:

The crew members' periodic rotation from space duty makes it difficult to accumulate enough experience and expertise in space power system operation, especially for an emergency or an abnormal operating state.

iii. Possibility of misoperation:

If a fault in the space power system occurs, several alarms are initiated simultaneously because of the cascading effect of a fault. Remedial action should be taken within an appropriate amount of time. The numerous alarms and the time pressure for remedial action may induce misoperation and further aggravate the operating state.

The objective of this study is to develop a real time fault monitoring and diagnosis knowledge-based system (KBS) for space power systems to solve and/or alleviate the problems stated above and to achieve more reliable space power system operation. The system was developed using the AMPS (Autonomously Managed Power System) test facility currently installed at NASA Marshall Space Flight Center [Figure 1]. The developed knowledge based system is entitled AMPERES (Autonomously Managed Power-System Extendible Real-time Expert System).

A real time KBS introduces many complex and dynamic problems. These problems include the continuous operation of the KBS, the handling of asynchronous and concurrent events, supporting data structures for data acquisition, nonmonotonic reasoning, response time constraints, and the validation of sensor values.

Most of the conventional approaches taken for fault diagnosis and monitoring KBS's are to look at a "snapshot" picture of the current state when a fault or an abnormal condition in the power system occurs. Then fault patterns are generated by mapping the sensor values into various qualitative terms. Matching fault patterns are then identified from fault data patterns stored in the Knowledge Base (KB). The major disadvantage of this approach is that it can only handle faults whose patterns are explicitly stored in the database. Instead of pattern matching, AMPERES looks at the sensor value causing the fault as well as other sensor values in the power system and actively searches with expectation through the KB for specific faults. Starting from any sensor showing an indication of a fault, the diagnosis is performed by propagating through the sensors logical environment with the expectation for the relevant faults and providing justifications about each sensor value along the logical path.

The KB in AMPERES includes the generic method descriptions for fault diagnoses instead of enumerating all the possible specific fault cases of the faulty data patterns. This keeps the size of the KB small and coherent. Each fault possibility is diagnosed by a specialized knowledge group associated with each fault kind. Each knowledge group is composed of rules which can

assert or validate the associated fault. Once a fault is asserted, different abstraction level descriptions of the fault is possible if necessary. Before a knowledge group is invoked, a context is set around a sensor to pass a default sensor associated with that knowledge group. This context switching together with the development of semantic primitive functions enables the rule representation to be more natural and clear. Combined with the inference method described above, this inference and knowledge representation scheme provides a powerful reasoning tool for fault monitoring and diagnosis tasks.

1.2 Previous Work

Most of the early approaches taken for fault diagnosis and monitoring KBS's use fault models (Poole, 1989; Struss et al., 1989). These fault models identify failure modes of faulty components. The fault model approach depends on the assumption that all fault models are known a priori. This assumption is normally not quite true. The fact that all failure modes are not known beforehand has led researchers to examine other diagnostic approaches.

To compensate the deficiency of the fault model based approach, researchers proposed the model-based diagnostic approach (Davis, 1985; Chen et al., 1989; Console et al., 1989; Gallanti et al., 1989; de Kleer et

al., 1990; Obreja, 1990; Rafea, 1990). The model-based approach diagnoses systems, mainly electronic devices or logic circuits, from their correct behavior only. The General Diagnostic Engine (GDE) of de Kleer and Williams represents a diagnosis system of electronic circuitry using this approach. GDE does not exploit knowledge about the faulty behavior of a device, but uses knowledge only about the correct behavior of a device. One disadvantage of using the model-based approach is the additional effort necessary to acquire the correct behavior of various devices and to represent this knowledge in the computer.

Chen et al. (1989) also implemented a model-based diagnostic system called VMES which assists technicians in troubleshooting electronic circuits. Candidate ordering, reordering, and elimination are the strong points in this system. Their research emphasized the importance of choosing the most likely candidate to focus on first so that the faulty parts can be located sooner.

Gallanti et al. (1989) presented a model-based diagnostic method which makes use of models at different levels of abstraction, qualitative and quantitative. A qualitative causal model is used for candidate generation whereas a quantitative real-valued model is used for candidate validation and rejection. Their implementation, however, was designed for diagnosis in process control

rather than diagnosis in electrical circuitry. The difference is that measurements are available before the diagnostic process begins in diagnosis of process control whereas measurements are constantly changing in the diagnosis of electrical circuitry.

Since the early 80's, the causal modeling approach has been widely adopted, especially in medical domains. Incompleteness both in the general causal knowledge (missing or abstracted knowledge) and in the data describing a specific case under examination must be dealt with when working with causal diagnostic systems. Console, Dupre', and Torasso used nonmonotonic reasoning (hypothetical and circumscriptive reasoning) to draw conclusions from incomplete knowledge (Console et al., 1989). The conclusions that are drawn rely on the criterion that the confirmation of assumptions is based on the nonmonotonic hypothesis that all relevant knowledge is modeled. In other words, all the possible causes for a given state are present. One disadvantage of the causal modeling approach is this assumption.

Struss and Dressler used both the fault model and the model-based diagnostic approaches and incorporated fault models in GDE and presented GDE+ (Struss et al., 1989). The fault models and the models describing the correct behavior of a device otherwise known as normal

models in GDE+ are encoded by constraints which deduce values of system parameters from known values. GDE+ uses an extended version of the Assumption-Based Truth-Maintenance System (ATMS) as its diagnostic reasoning mechanism. This diagnostic strategy continuously gathers more information to help decide which assumptions should be removed in order to eliminate existing contradictions. In other words, a fault is identified by comparing the simulated normal device behavior with the actual one. Then the fault models enable the diagnostic system to analyze whether the faultiness of components would really explain the observations. Single fault assumption is used to generate small candidate sets. For example, if each fault model of a component contradicts the correctness assumption of at least one other component, then the faultiness of that component cannot explain the observations alone and is therefore not a candidate for a single fault. This cuts down on the potential danger of creating a large number of combinations of correct models and fault models. Constraint propagation is used for candidate generation in GDE+. This mechanism constructs sets of correctness assumptions that account for the known conflicts. In other words, their conjunctive retraction would remove the contradictions. The completeness of fault models is an essential condition for the inference mechanism in GDE+. If a faulty component

shows behavior that isn't captured by one of the fault models, invalid conclusions may be drawn.

De Kleer and Williams developed the system, Sherlock, which also identifies correct and faulty models in their electronic circuitry diagnostic system (De Kleer et al., 1989). However, by incorporating fault models, these systems potentially introduce significant computational overhead for the diagnostic algorithms. Thus they seem not quite suitable for monitoring real-time processes.

Rafea (1990) combined the fault model rule based system with the causal model to develop an expert system for electronics fault diagnosis. The fault model rules capture the expertise of the domain experts while the causal model better explains the device behavior since the knowledge is described in terms of the device structure and behavior. The introduction of the causal model allows better explanations when this information is passed to the operator however increases the computational overhead for the diagnostic algorithms. This expert system was not implemented for real-time fault diagnosis.

1.3 Overall Approach

AMPERES is a research project that was developed and implemented by a group of graduate students. The research presented in this thesis briefly discusses the overall

implementation and layout of AMPERES to facilitate in the readers' comprehension of AMPERES. The main emphasis of the thesis discusses the development and implementation of the main fault monitoring and diagnosis task and its supporting knowledge representation scheme which were the accomplishments of the author. The implementation of the data acquisition task, the user interface task, and the sensor value validation method was accomplished by fellow graduate students.

AMPERES is able to collect real time operational data and to assess the power system operating status. The operational data includes the measurement data of the various sensors in AMPS such as voltmeter and ammeter measurements. Since AMPERES was developed at the University of Tennessee Space Institute and the AMPS test facility is installed at NASA Marshall Space Flight Center, data simulation systems have been developed.

For this purpose, a fault injection and load simulation device known as the FILSD is designed and assembled. [Figure 2]. In the load simulation mode, two resistive loads can be connected to the FILSD using batteries as a power source. The voltage and the currents across the loads are measured through signal conditioning circuits and the data acquisition board to create an operational environment as close to the AMPS test facility

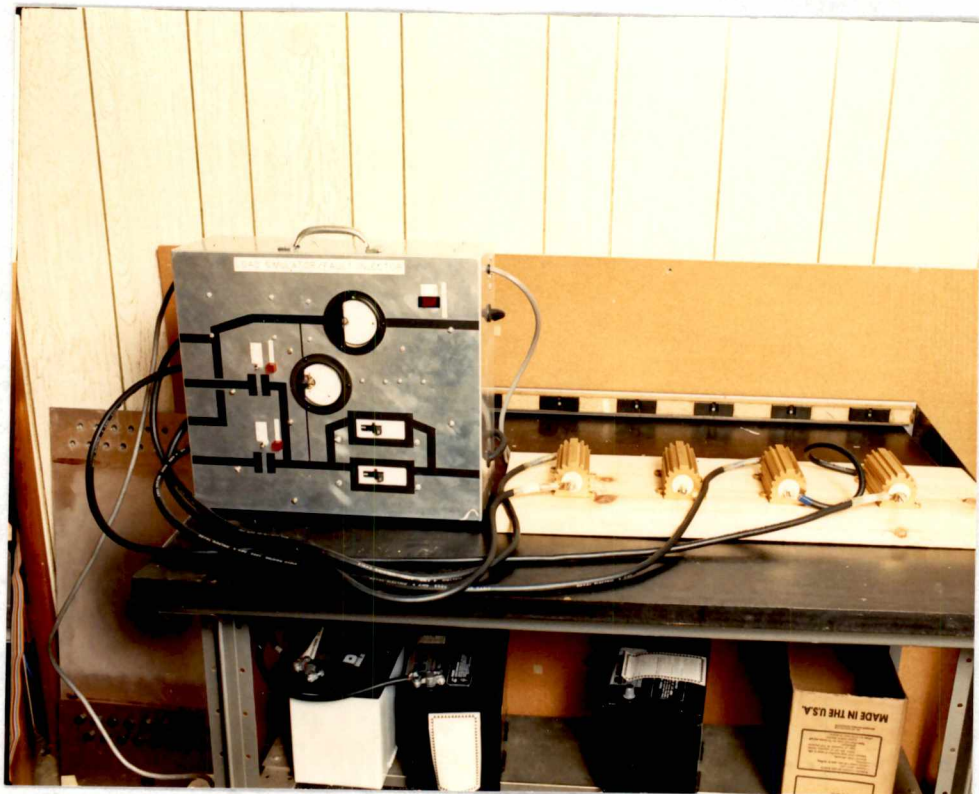


Figure 2. FILSD Connected with Four Resistive Loads

as possible. In the fault injection mode, a time delayed fault can be injected under computer control.

The data simulation program, running on a separate personal computer, can replace any one of ten simulated load data with actual data from the FILSD. The operational data is then transferred through an Ethernet to the host computer Sun 386i.

Faults and disturbances such as overloads, ground faults, battery cell open, solar array system failure, load connect and disconnect, etc, can be generated. AMPERES detects these faults and disturbances and provides a report to the operator including times of occurrence, severities, detecting sensors, locations, and fault types.

Three concurrent processes are created in AMPERES to handle the data acquisition task, the user interface task, and the main fault monitoring and diagnosis task [Figure 3]. Interprocess communication is performed through shared memory.

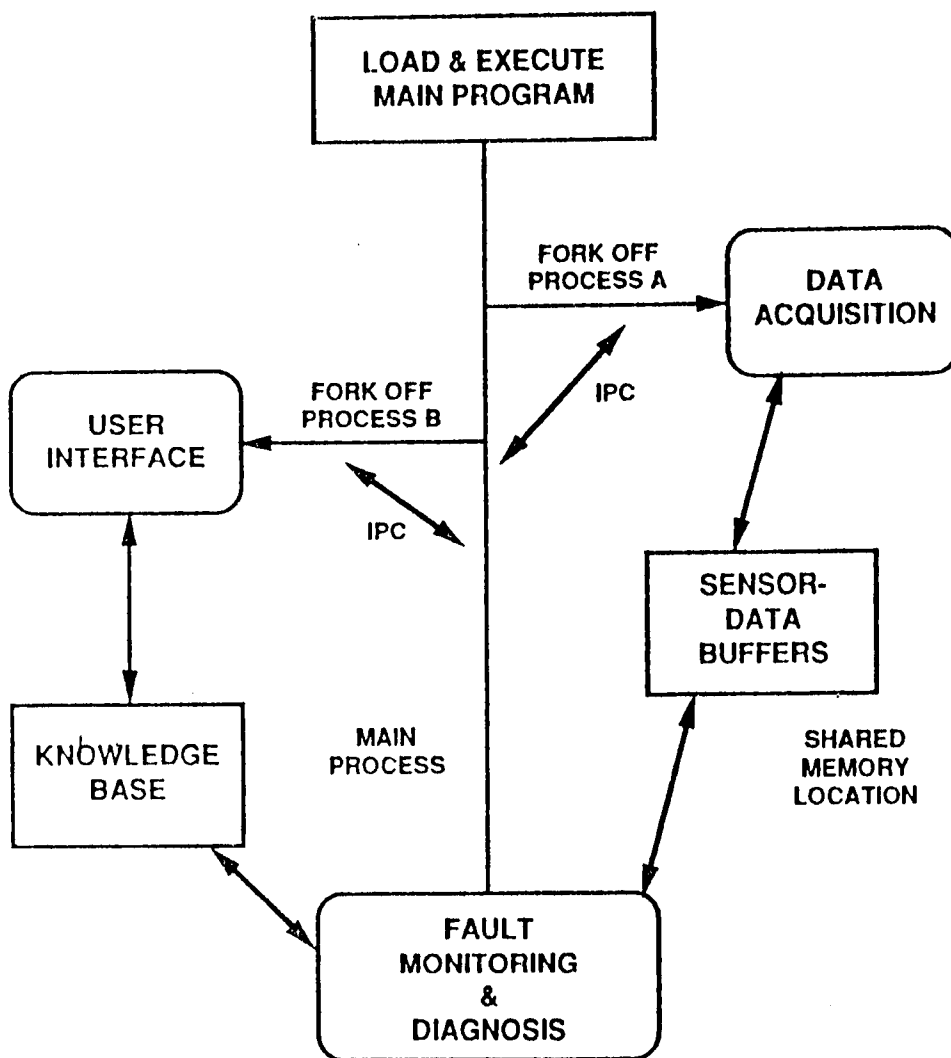


Figure 3. Concurrent Task Management in AMPERES

CHAPTER II

ORGANIZATION OF THE AMPERES

2.1 Data Acquisition

The Data Acquisition Module acquires sensor data in AMPERES. The sensor data is stored in one of three data buffers. This sensor data is categorized as either critical or non-critical data based on the possible speed changes in their measurement values and on the time resolution requirements for diagnosis [Figure 4]. Two buffers are needed for the critical data set and one buffer is needed for the non-critical data set. The data buffers are circular buffers with global pointers used to mark the present incoming data location, the present analysis data location, and the present buffer upper bound.

The critical and non-critical data buffer layouts as well as the sensor names used throughout the thesis are shown in Figure 5. The critical sensor data set currently contains 56 analog values and 42 digital values and is acquired every 10 ~ 100 milliseconds. The critical data buffers can hold up to 300 data sets. The breakdown of the analog values are listed in Figure 5. The digital values consist of 4 mounting switches for each of the 9 loads (refer to Figure 1 on page 3 showing the 10 loads inside the load center) and 3 mounting switches for the 10th load

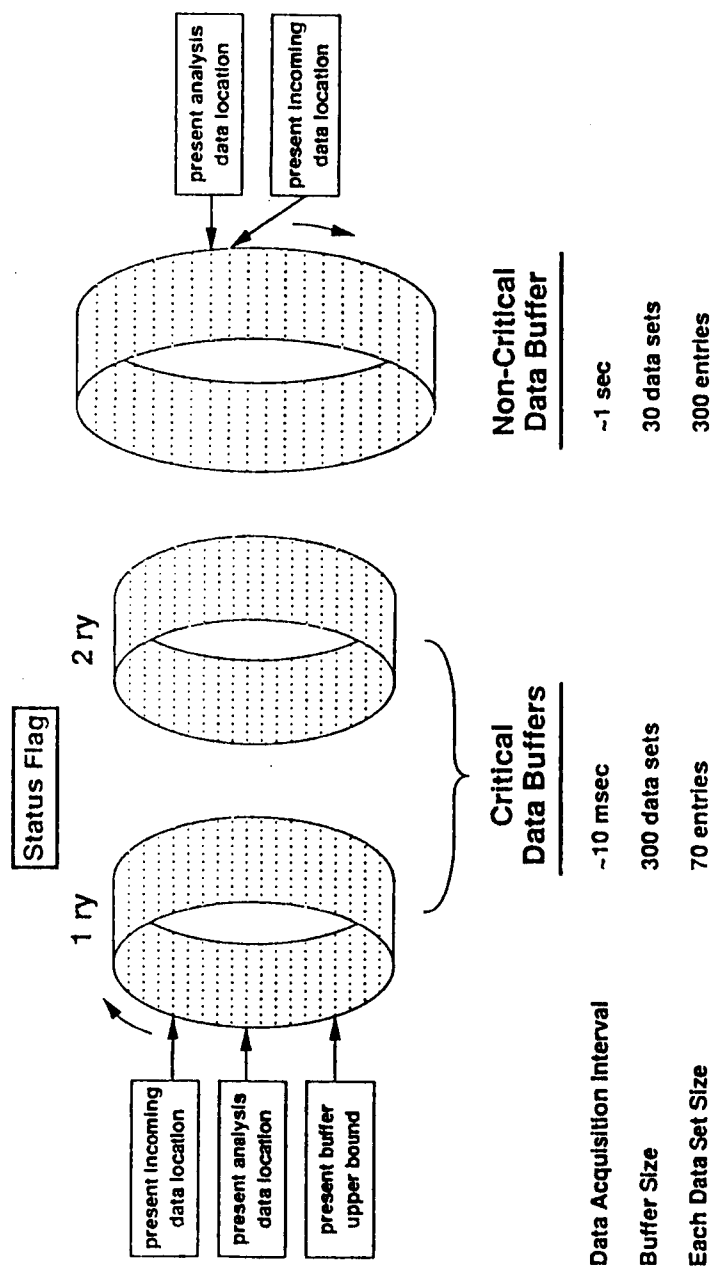


Figure 4. Data Structure for Data Acquisition

Critical Data Buffer Layout

<u>Number</u>	<u>Sensor(s)</u>	<u>Sensor Name</u>
1	main voltmeter	V_m
28	battery module voltmeters	$V_{b1}-V_{b28}$
10	load voltmeters	$V_{l1}-V_{l10}$
10	load ammeters	$A_{l1}-A_{l10}$
3	feeder ammeters	$A_{f1}-A_{f3}$
1	main ammeter	A_m
1	solar array ammeter	A_s
1	battery ammeter	A_b
+ 1	main battery voltmeter	V_b

56	analog values	
4	16-bit words for time of day bits and mounting switch position bits (digital values)	
+10	spare buffer slots	

70	entries	

Non-Critical Data Buffer Layout

<u>Number</u>	<u>Sensor(s)</u>	<u>Sensor Name</u>
168	battery cell voltmeters	$V_{c1}-V_{c168}$
31	power diode temperature meters	$T_{p1}-T_{p31}$
+14	battery module temperature meters	$T_{b1}-T_{b14}$

213	analog values	
70	same as critical data buffer	
+17	spare buffer slots	

300	entries	

Figure 5. Critical and Non-Critical Data Buffer Layouts

for a subtotal of 39 digital values. There are also 3 mounting switches for each of the channels leading into the load center for a total of 42 digital values. These digital values indicate whether the mounting switch is open or closed. The non-critical sensor data set currently contains 213 analog values and is acquired every second. The data buffer for the non-critical sensors can hold up to 30 data sets. The breakdown of the analog values for the non-critical data set is also listed in Figure 5.

Two data buffers are created in shared memory locations for the critical sensor data and are referred to as the primary buffer and the secondary buffer respectively. The secondary buffer provides a back-up buffer in case a fault leaves the present data buffer thereby providing pre and post fault data for use by the fault diagnosis module. A system status flag is created in a shared memory location to facilitate the buffer switching process between these two critical data buffers [Figure 6]. This flag indicates which buffer is currently acquiring data. The fault monitoring and diagnosis task and the data acquisition task modify the value of this flag thus providing a means for continuously acquiring data. This also allows the fault monitoring and diagnosis task to look at previous data in the buffer not currently acquiring data without the possibility of this data being updated.

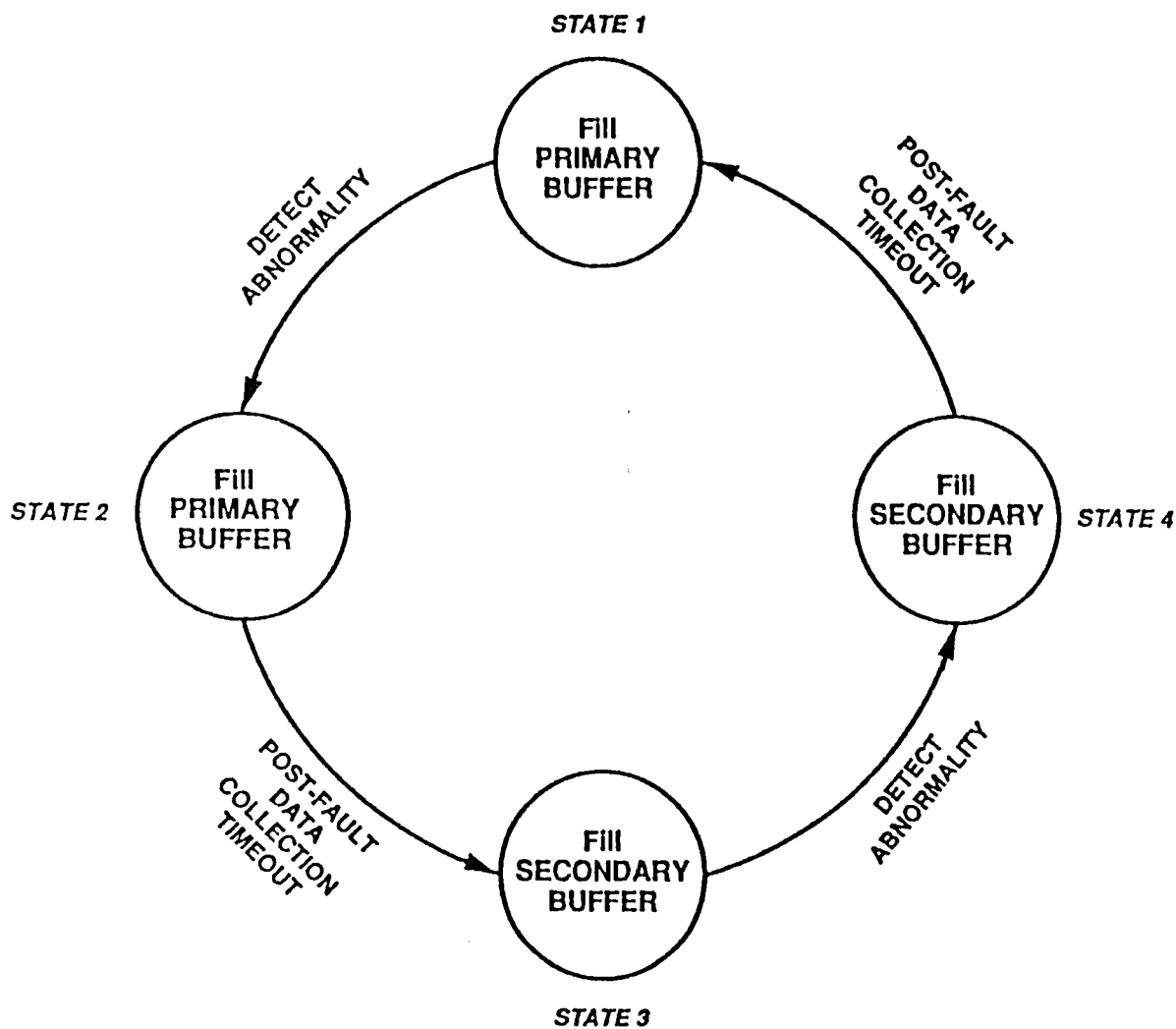


Figure 6. Critical Data Acquisition Process
State Transition Diagram

The status flag has four states. When the data acquisition process begins, the system status flag is set to state 1 and the primary buffer is used to acquire data. Upon detection of an abnormality, the fault monitoring process sets the status flag to state 2. When the data acquisition process notices that the status flag is set to 2, the secondary buffer will be used after approximately one second and the status flag will be set to 3. Now the fault monitoring process examines the latest available data in the secondary buffer marked by a global pointer. Upon detection of another abnormality, the fault monitoring process increments the status flag to 4. Upon detecting state 4, the data acquisition process fills the secondary buffer for another second, resets the status flag to 1, and begins acquiring data in the primary buffer again.

2.2 Concurrent Task Handling

To perform several concurrent tasks, the main process in AMPERES, called the Main Controller, creates several processes upon initialization. The concurrent tasks required are the data acquisition task, the operator interface task, and the fault monitoring and diagnosis task. The interprocess communication is managed through shared memory.

2.3 Knowledge Base Representation

Each component in the AMPS is represented as an object in the AMPERES using the structure definition in Common Lisp. An example of an ammeter representation is shown in Figure 7. The information contained in each object can be categorized in the following three groups:

i. The information about the component itself includes the design data, the expected operational value, the present operational value, graphics information for display, etc. For example, from Figure 7, component-id, one_of, present_expected_range, normal_expected_range, present_value, trend, faulty_state, and graphics are the slots representing the information about an ammeter.

ii. The Functional Environment of the component includes the components which are functionally or logically related to the current component. This information is essential in collecting the supporting evidences and checking the cascading effects of a fault. From Figure 7, assoc_cb, i.e. functionally associated circuit breaker, assoc_ms, connect_load, parent_ammeter, children_ammeter,

```

;-----Ammeter Frame-----
(defstruct ammeter
  (component_id)
  (connect_terminal)
  (location)
  (one_of 'load_ammeter)
  (assoc_cb)
  (assoc_ms)
  (connect_load)
  (parent_ammeter)
  (children_ammeter)
  (assoc_voltmeter)
  (present_expected_range)
  (normal_expected_range)
  (present_value)
  (member_CR)
  (trend 'steady)
  (faulty_state 'no)
  (signal_ckt)
  (graphics))

```

Figure 7. Component Representation and Its Environment

assoc_voltmeter are the slots describing the functional environment of an ammeter.

iii. The Physical Environment of the component includes the information about those components which are physically connected or attached to the current component. This information is necessary in identifying the extension of a faulty location and for graphical display of the system. From Figure 7, connect_terminal and location slots are the slots describing the physical environment of an ammeter.

iv. The Membership in Causal Relations is necessary to implement the sensor value validation procedure. It identifies the causal relations that this sensor belongs to. From Figure 7, member_CR describes the causal relations this component belongs to.

2.4 Fault Diagnosis Module

The fault monitoring process begins checking for new data in the circular data buffers by looking at the present buffer location pointers. When new data is available, the data is read from the buffer and placed into the appropriate slot values for each sensor frame. Then the

current operating state is compared with the expected operating state.

Any sensor value that has changed significantly is added to a State Change List (SCL) by the Status Monitor. The SCL is an association list which is a list of pairs and each pair is composed of one major sensor and one minor sensor. The major sensor name is the classification name of a sensor such as "voltmeter" or "ammeter". The minor sensor name is the unique name assigned to each sensor such as "V₁₃" or "A_{f1}". For example, if the SCL, denoted as L_s , were $L_s = ((\text{Voltmeter } V_m) (\text{Ammeter } A_{11}))$, L_s would indicate significant changes in the values of the voltmeter named V_m and the ammeter named A_{11} . The SCL enables the diagnosis of multiple faults since it contains pairs of abnormal sensors. The SCL is then passed to the fault diagnosis process.

The fault diagnosis process examines each pair in the SCL. First, a context is set around the sensor that is going to be the center of the universe in examining its logically related sensor values. The KB refers to variable names associated with a particular sensor's environment. For example, if the first pair of the SCL is (Ammeter A_{11}), then the context is set around the ammeter environment. This context switching allows the rule expressions in the KB to be simple and natural.

Next, the KB is invoked with the appropriate rule group. Rules are grouped such that each rule group is specialized in resolving a specific situation or a fault. The first rule group invoked is one of the major sensor rule groups, such as ammeter rules, voltmeter rules, circuit breaker rules, etc. Other rule groups can then be invoked which specialize in diagnosing a particular fault associated with a sensor such as load disconnect or battery cell open.

Once a fault is found, a fault object is created, which includes the information related to the faults. This fault object is added to a fault assertion list. All the rules that were traversed while trying to find the fault are stored in an explanation list. The fault object is passed to the user interface task which displays the fault description to the operator. The fault diagnosis process repeats the above process until all pairs in the SCL are checked. When the diagnosed fault with the present pair is the same as one of the faults found with a previous pair, the diagnosed fault is not added to the fault assertion list and the result is ignored.

CHAPTER III

DEVELOPMENT OF THE FAULT DIAGNOSIS SCHEME

3.1 Sensor Value Validation Scheme

In a fault monitoring and diagnosis process of a physical system whose operating state is monitored by numerous remote sensors, as in the AMPS test facility, one of the crucial steps involved is to validate the incoming sensor values. In realistic situations, there is a high probability that a sensor may give a false value that can be either temporary or permanent. Trying to assess the operating state based on false sensor values can be detrimental to the monitored system if any control action is taken because of these false sensor values. As a result, a sensor value validation method has been developed in corporation with the fault monitoring and diagnosis knowledge based system and was implemented by another graduate student.

The validation method utilizes the Causal Relations (CR) existing among the sensor values and the Sensor Redundancy Graph (SRG) formulated from the CR's. The definitions of the CR and the SRG as well as the detailed sensor validation procedure appear in Lee (1990).

Part of the one line diagram for AMPS showing various voltmeters and ammeters is shown in Figure 8. Some

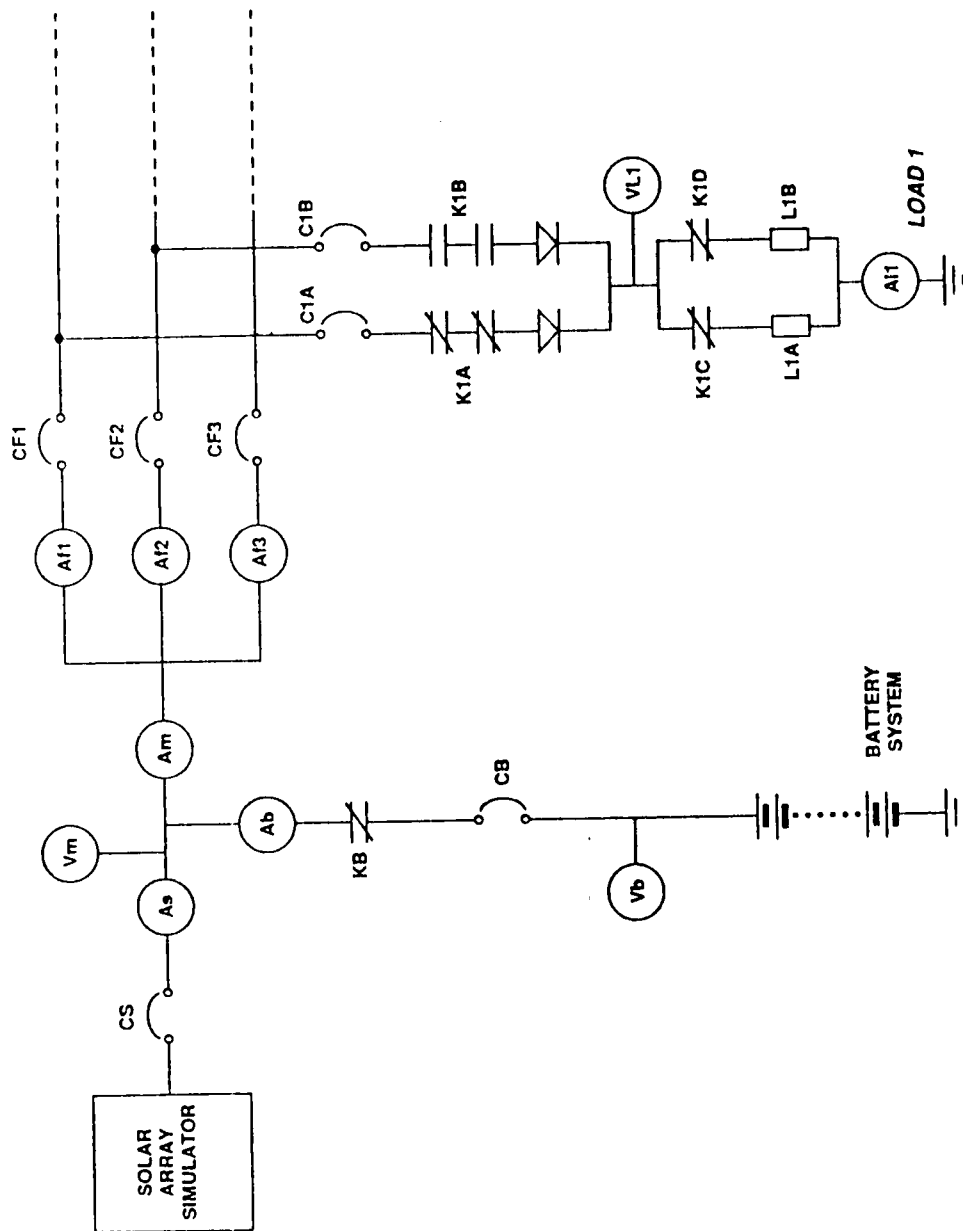


Figure 8. Simplified One Line Diagram for AMPS

of the sensors used in examples in this section are represented with the sensor names inside of circles (i.e. $V_b, A_s, V_m, A_m, A_{f1}, A_{f2}, A_{f3}, V_{l1}, A_{l1}$). The other named objects in this figure correspond to various mounting switches and circuit breakers.

Relationships existing among various sensor values in AMPS are referred to as CR's. For example, the sum of the values of the feeder ammeters, $A_{f1}-A_{f3}$, is equal to the value of the main ammeter, A_m . Existing CR's, identified as R_1 through R_{55} , among the sensor values in AMPS are identified in Figure 9. The lower case r in causal relations R_6 through R_{15} are constant resistor values. The associated SRG is shown in Figure 10. The SRG graphically shows the causal relations a sensor belongs to. It also clearly shows the excessive redundancy existing for the main voltmeter (V_m) since the main voltmeter value is supposed to be almost equal to all the load voltmeter values.

When all the CR's are consistent, the validity levels of all the voltmeters and ammeters are as shown in Case 1 of Figure 11. Initially, the validity levels of all the sensors are set to zero. The validity level of a sensor is incremented by 1 each time the sensor belongs to a consistent CR. As a result of the initial consistency test of all the CR's, the validity level of V_m is set to 11

$$\begin{aligned}
R_1: \quad A_s &= A_m + A_b \\
R_2: \quad A_m &= A_{f1} + A_{f2} + A_{f3} \\
R_3: \quad A_{f1} &= A_{11} + A_{13} + A_{15} + A_{10} \\
R_4: \quad A_{f2} &= A_{12} + A_{17} + A_{19} \\
R_5: \quad A_{f3} &= A_{14} + A_{16} + A_{18} \\
R_6: \quad A_{11} &= V_{11} / r_1 \\
R_7: \quad A_{12} &= V_{12} / r_2 \\
&\vdots \\
R_{15}: \quad A_{110} &= V_{110} / r_{10} \\
R_{16}: \quad V_{11} &= V_m \\
R_{17}: \quad V_{12} &= V_m \\
&\vdots \\
R_{25}: \quad V_{110} &= V_m \\
R_{26}: \quad V_m &= V_b \\
R_{27}: \quad V_b &= V_{b1} + V_{b2} + \dots + V_{b28} \\
R_{28}: \quad V_{b1} &= V_{c1} + V_{c2} + \dots + V_{c6} \\
R_{29}: \quad V_{b2} &= V_{c7} + V_{c8} + \dots + V_{c12} \\
&\vdots \\
R_{55}: \quad V_{b28} &= V_{c163} + V_{c164} + \dots + V_{c168}
\end{aligned}$$

Figure 9. Causal Relation listing for AMPS

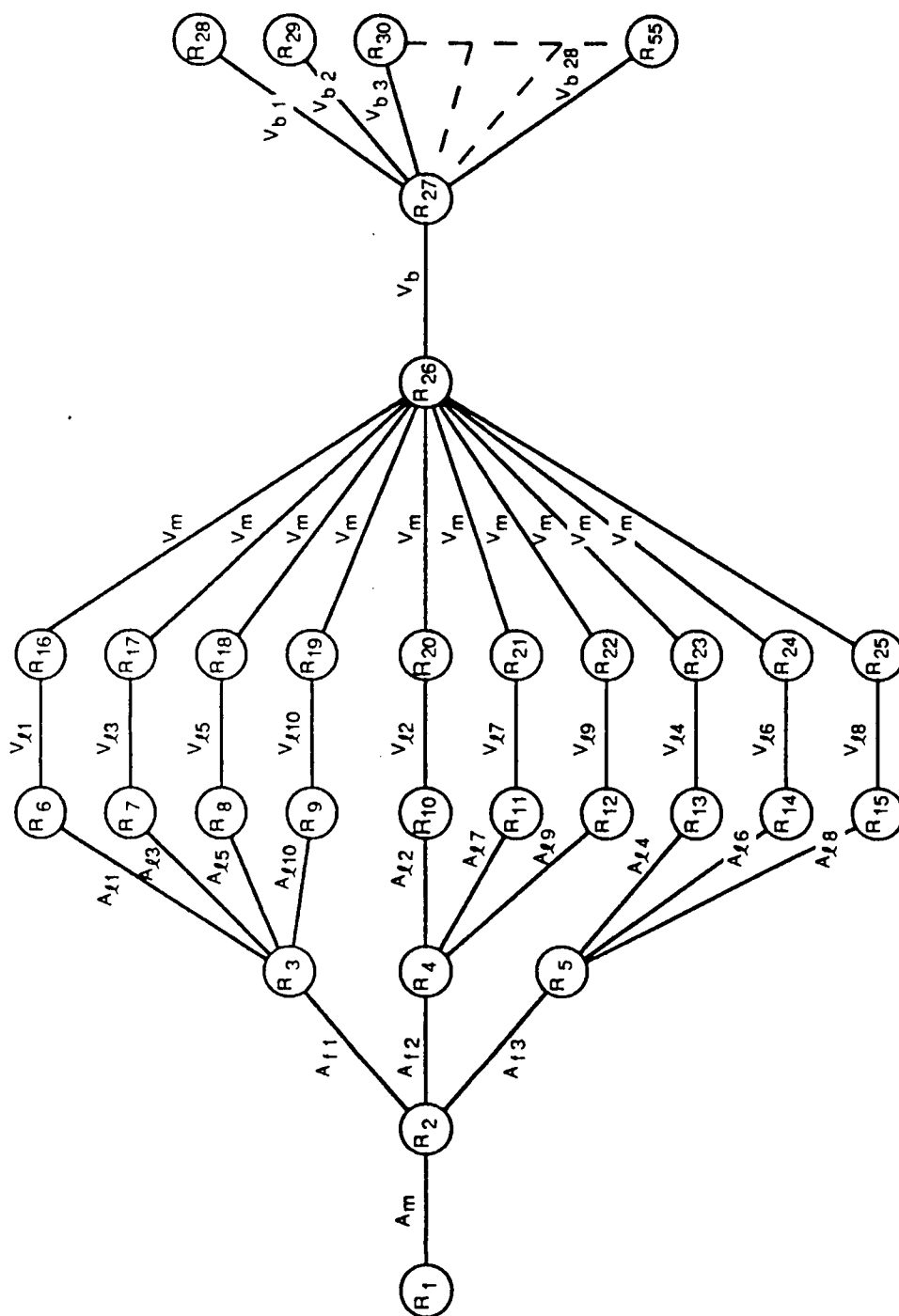


Figure 10. The Sensor Redundancy Graph (SRG) for Ammeters and Voltmeters Installed on AMPs

	$\underline{A_s}$	$\underline{A_m}$	$\underline{A_h}$	$\underline{A_{f1}}$	$\underline{A_{f2}}$	$\underline{A_{f3}}$	$\underline{A_{f4}}$	$\underline{A_{f10}}$	$\underline{V_h}$	$\underline{V_{f1}}$	$\underline{V_{f10}}$	$\underline{V_{h1}}$	$\underline{V_{f28}}$	$\underline{V_{c1}}$	$\underline{V_{c168}}$
		$\textcircled{11}$		$\textcircled{11}$	$\textcircled{11}$	$\textcircled{11}$	$\textcircled{11}$	$\textcircled{11}$	$\textcircled{11}$	$\textcircled{11}$	$\textcircled{11}$				
Case 1	1	2	1	2	2	2	2	2	11	2	2	2	2	1	1
Case 2	1	2	1	<u>1</u>	2	2	2	2	11	11	11	2	2	1	1

* The validity levels incremented using the Interdependency Lemma are circled.

** The validity levels changed from the original validity levels, i.e. the validity levels when all the CRs are consistent, are underlined.

Figure 11. Validity Levels of the Sensors for AMPS

and the validity levels of battery voltmeter V_b and all the load voltmeters $V_{li}, i=1, \dots, 10$, are set to 2. But based on the Interdependency Lemma which states that the number of sensors with the minimum validity level in a CR should be at least two, the validity levels of V_b and V_{li} for $i=1, \dots, 10$, are incremented to 11. This is due to CR R_{26} where V_b is the only sensor with the minimum validity level, which is 2, and the next lowest validity level of the rest of the sensors in R_{26} is V_m or 11. For the load voltmeters in CR's R_{16} through R_{25} , the validity levels are also incremented to that of V_m for similar reasons.

The validity level is used to express the confidence of the sensor value. With the validation results expressed as simple integer values, the system operators or the KBS can quickly grasp the overall state of the sensor system, confidently assess the system state, and initiate necessary control actions based on the validated sensor values.

To demonstrate how the validation method works, suppose the feeder ammeter A_{f1} suddenly shows a significantly increased value, and as a result, CR's R_2 and R_3 become inconsistent while the rest of the CRs remain consistent. If A_m and A_{f1} are the representative sensors in CRs R_2 and R_3 respectively, then from the expression of R_2 and R_3 in Figure 9

$$\langle A_m \rangle \neq \langle A_{f1} \rangle + \langle A_{f2} \rangle + \langle A_{f3} \rangle$$

$$\langle A_{f1} \rangle \neq \langle A_{11} \rangle + \langle A_{13} \rangle + \langle A_{15} \rangle + \langle A_{110} \rangle$$

where $\neq$ is the causal inconsistency relation.

The expected values of A_{f1} , not the actual value, derived from CR's R_2 and R_3 respectively and denoted as $\langle A_{f1} \rangle_2$ and $\langle A_{f1} \rangle_3$, become

$$\langle A_{f1} \rangle_2 = \langle A_m \rangle - \langle A_{f2} \rangle - \langle A_{f3} \rangle$$

$$\langle A_{f1} \rangle_3 = \langle A_{11} \rangle + \langle A_{13} \rangle + \langle A_{15} \rangle + \langle A_{110} \rangle$$

If R_2 and R_3 form a consistency pair, then

$$\langle A_{f1} \rangle_2 \approx \langle A_{f1} \rangle_3$$

where $\approx$ is the causal consistency relation. Therefore, the validity level of $\langle A_{f1} \rangle$, denoted as $[A_{f1}]$, is decremented by one. The correct value of $\langle A_{f1} \rangle$, which is called the repaired $\langle A_{f1} \rangle$ and denoted as $\langle \tilde{A}_{f1} \rangle$, can be computed from $\langle A_{f1} \rangle_2$ and $\langle A_{f1} \rangle_3$. In this case, both CR's R_2 and R_3 can be made consistent using $\langle \tilde{A}_{f1} \rangle$, and R_2 and R_3 are said to be repaired with $\langle \tilde{A}_{f1} \rangle$. Using these repaired CRs, denoted as R'_2 and R'_3 , the validity levels of all the sensors in $\{R_2\} - \{A_{f1}\}$ and $\{R_3\} - \{A_{f1}\}$ are incremented to one, where $\{R_i\}$ denotes the set of all the sensors involved in R_i .

Since all other CRs are consistent, validity levels of A_m , A_{f2} , and A_{f3} are incremented by R_2 , and A_{11} , A_{13} , A_{15} , A_{110} by R_6 , R_8 , R_{10} and R_{15} respectively. As a result, the validity levels of A_m , A_{f2} , A_{f3} , A_{11} , A_{13} , A_{15} , and A_{110} remain the same as when all the CR's are consistent.

The invalid sensor value $\langle A_{f1} \rangle$ has thus been identified. The validity levels for other sensors are restored through the repairment of the failed sensor value and the inconsistent causal relations. Only the validity level of A_{f1} has changed to -1 as shown in Case 2 of Figure 11.

The results demonstrate how the sensor value validation results can also yield indications about the potential disturbances and their locations.

3.2 DIAG Formulation

A data structure called the Disturbance Interrelation Analysis Graph (DIAG) is developed to formalize the inference scheme of the fault diagnosis task and to explicitly show the diagnosis reasoning paths. The DIAG, $G_d = (V_d, E_d)$ is a multi-level AND/OR graph developed to characterize real-time fault diagnosis problems. DIAG is composed of three levels for a description of the faults and their interrelations: high abstraction level, low abstraction level, and the sensor level with finite non-empty supersets of nodes and labeled edges such as

$$V_d = \{V_h, V_l, V_s\}$$

and

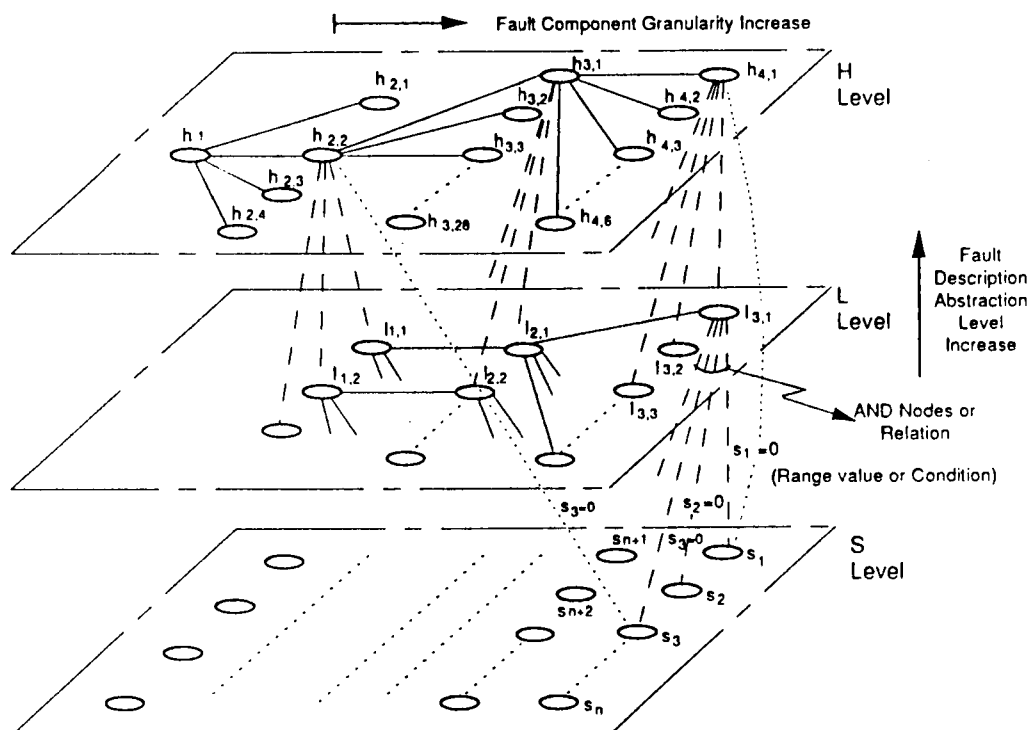
$$E_d = \{E_h, E_l, E_{hl}, E_{hs}, E_{ls}\}$$

where V_h, V_l, V_s are a set of nodes in H-level, L-level, and S-level, respectively and E_h, E_l are a set of intra-level

edges connecting H-level, L-level nodes respectively and E_{hl}, E_{hs}, E_{ls} are a set of inter-level edges connecting H-level to L-level nodes, H-level to S-level nodes, and L-level to S-level nodes, respectively.

Starting from any sensor value indicating a fault, fault diagnosis is performed by tracing the appropriate edges in DIAG. Each component or subsystem in AMPERES, including sensors, is an attribute of a node in DIAG and represented as an object to facilitate the DIAG implementation. The component representation includes information about its functional or logical environment as well as its physical environment. Each fault is also an attribute of a node in DIAG and can be diagnosed by a specialized knowledge group associated with that node. A portion of the DIAG representing the battery system fault diagnosis is shown in Figure 12.

The high abstraction level which is also called the system decomposition level and represented as the H-level, is a tree such that each node contains attributes of a component or a part of the system. To represent any system in DIAG, proper boundaries should be selected between components of the system which separates and clarifies distinct portions of the system. These portions of the system are represented as nodes in the H-level of DIAG and are connected to the root node or the system faulty node.



h_1 : System faulty	$h_{2,1}$: SAS faulty	$h_{2,2}$: Battery Sys. faulty
$h_{2,3}$: Main Feeder faulty	$h_{2,4}$: Load Feeder faulty	$h_{3,1}$: Bat. Mod. 1 faulty
$h_{3,3}$: Bat. Mod. 2 faulty	$h_{3,3}$: Bat. Mod. 3 faulty	$h_{3,28}$: Bat Mod. 28 faulty
$h_{4,1}$: Bat. Cell 1 faulty	$h_{4,3}$: Bat. Cell 2 faulty	$h_{4,6}$: Bat. Cell 6 faulty
$l_{1,1}$: Bat. Sys. open	$l_{1,2}$: Bat. Sys. short	$l_{2,1}$: Bat. Mod. 1 open
$l_{2,2}$: Bat. Mod. 1 short	$l_{3,1}$: Bat. Cell 1 open	$l_{3,2}$: Bat. Cell 1 short
$l_{3,3}$: Bat. Cell 1 ground	s_1 : Bat. Cell 1 voltage V_{e1} :	s_2 : Bat. Mod. 1 voltage V_{e1}
s_3 : Bat. Sys. voltage V_0		

Figure 12. Part of DIAG (Disturbance Interrelation Analysis Graph) for Battery System Fault Diagnosis

In this way, the nodes form a hierarchical tree structure. The system should then be decomposed until sufficient granularity of the physical components can be obtained in describing the faults. Most importantly, this decomposition should be performed such that it does not leave any component uncovered. For each node, the node along with all its siblings should completely represent the subsystem of its parent node. This assures the ability to diagnose a fault associated with any component in the system. In AMPERES, the root node has four children nodes, which implies that the system can be decomposed into four major subsystems: the Solar Array System, the Battery System, the Main Feeder System, and the Load Feeder System. Each of these nodes represents that each subsystem is simply abnormal or faulty.

The intermediate abstraction level, which is also called the disturbance instantiation level and represented as the L-level, is composed of nodes whose attributes describe a fault of each component with a specific fault instance name. These nodes form disjoint fault trees. Each node is connected to the parent node which describes the same fault but with larger component granularity. For example, in Figure 12, the node representing the battery cell 1 open ($l_{3,1}$) is connected to the parent node representing battery module 1 open ($l_{2,1}$) since battery

cell 1 is a part of battery module 1. Again battery module 1 ($l_{2,1}$) is connected to its parent node representing battery system open ($l_{1,1}$). As each parent node is divided into its children nodes, the fault component granularity increases.

The low level, denoted as the S-level, contains nodes whose attributes represent the sensors through which the system operating state is observed. These nodes are connected to the nodes in the L-level and sometimes are directly connected to the H-level when the only conclusion that can be drawn from the sensor value is that the component is faulty or abnormal. The arcs connecting the S-level and the L-level nodes are labeled with the sensor value ranges or the conditions that the sensor value should satisfy. In examining the sensor values, all the arcs whose conditions are satisfied become the candidate diagnosis reasoning paths. One sensor node can be connected to several L-level nodes with the same or different label values. If there is more than one arc connecting the L-level to the S-level, these arc's may have an "AND" relation or an "OR" relation. For example, if battery cell 1 open represented an L-level node, then battery cell voltage 1, battery module voltage 1 and battery system voltage 1 must all have values of 0. Therefore, the arcs connecting this particular L-level node

to the above three sensors in the S-level would have an "AND" relation.

The DIAG data structure provides a formalized method for relating the "eyes" of a system to various fault abstraction levels. In the case of AMPERES, the sensors through which we "see" the components of this system are clearly mapped to fault descriptions.

3.3 Fault Diagnosis Process Using DIAG

Part of the DIAG formalism has been implemented in the fault diagnosis process of AMPERES using Common Lisp on the SUN 386i. The realization of DIAG in AMPERES is presented in this section. All the components and subsystems, including sensors, are represented as structures. Slot names in the structures shown in Figure 7 such as "faulty-state", "present_expected_range" and "normal_expected_range" are for the realization of part of the DIAG.

The first task performed by the fault diagnosis process sets the environment around the first abnormality pair from the SCL. As each pair (abnormality pair) of the SCL is evaluated, the set_current_context routine assigns the variable names used throughout the KB to values associated with the abnormality pair being resolved. Part of this routine showing the ammeter environment definition is shown in Figure 13. For example, if the environment is

```

(defun set_current_context (major minor)

  (setq major_sensor major)
  (setq minor_sensor minor)
  (setq present_sensor minor_sensor)
  (setq mkey minor_sensor)
  (setq key (eval mkey))
  (cond ((equal 'ammeter major_sensor) (ammeter_env
    minor_sensor))
    ((equal 'voltmeter major_sensor) (voltmeter_env
    minor_sensor)))

(defun ammeter_env (minor_sensor)
  (setq component_id (ammeter-component_id key))
  (setq connect_terminal (ammeter-connect_terminal key))
  (setq location (ammeter-location key))
  (setq one_of (ammeter-one_of key))
  (setq associated_cb (ammeter-assoc_cb key))
  (setq associated_ms (ammeter-assoc_ms key))
  (setq connect_load (ammeter-connect_load key))
  (setq parent_ammeter (ammeter-parent_ammeter key))
  (setq children_ammeter (ammeter-children_ammeter key))
  (setq associated_voltmeter (ammeter-assoc_voltmeter
key))
  (setq
    present_expected_range
(ammeter-present_expected_range key))
  (setq
    normal_expected_range
(ammeter-normal_expected_range key))
  (setq present_value (ammeter-present_value key))
  (setq trend (ammeter-trend key))
  (setq faulty_state (ammeter-faulty_state key))
  (setq signal_ckt (ammeter-signal_ckt key))
  (setq graphics (ammeter-graphics key)))

```

Figure 13. Part of Set Current Context Routine

being set around the pair (ammeter A_{11}), the global variables in the 'ammeter_env' definition would be assigned the current values from the A_{11} ammeter structure.

The number of lists received by the fault diagnosis task could potentially be quite large if consequences of various faults were also added to the SCL. Therefore, the number of lists in the SCL is minimized by the "present_expected_range" slot of the sensor structures. This allows the fault diagnosis process to concentrate on the initial source of the disturbance without being distracted by consequences of the disturbance. For example, a natural consequence of the system voltmeter value going to zero would be the load voltmeter and ammeter values also going to zero. However, we only want the system voltmeter to be included in the SCL. With the system voltmeter value being zero, the "present_expected_range" of the load voltmeter and ammeter would also be zero. Therefore, these two sensors would not be included in the SCL since their current values are not outside of the expected range.

Next, the appropriate rules in the rule groups associated with each sensor type are invoked by the routine `get_minor_rb`. The bottom level in DIAG is the S-level, also called the sensor level, and is used to facilitate the rule group invocation associated with each sensor. The KB is

composed of rules and structures representing objects. All the sensor level rules referred to in the KB are grouped by sensor type and are stored in separate rule group structures. For example, the rules associated with the voltmeter sensors are defined in the voltmeter rule structure. This provide a faster search when the rules associated with a particular sensor are being located in the KB. The fault diagnosis module first examines the KB by looking in the corresponding sensor type rule group of the major sensor from the '(major sensor minor sensor') pair picked up from the SCL. The KB contains rules for all the sensor types through which the system operating state of AMPERES is observed.

The "sensor rules" currently defined in AMPERES are ammeter rules, voltmeter rules, tempmeter rules , circuit breaker (cb) rules and mounting switch (ms) rules. Inside these sensor rules, rules are grouped according to functional similarities. For example, in the ammeter rules, there are "load" ammeter rules applicable to load ammeters A_{11} , A_{12} , ..., A_{110} , as well as "feeder" ammeter rules applicable to feeder ammeters A_{f1} , A_{f2} , and A_{f3} . This applicable class of rules can be identified by looking at the "one_of" slot value in each sensor structure. The grouping of rules in this manner keeps the number of required rules small.

Once the appropriate sensor rules in the KB are accessed, sensor value ranges or various conditions of the sensor value are examined to try to establish candidate diagnosis reasoning paths. This is the realization of the edge traversal in DIAG. As shown in Figure 12, there exists some L-level nodes whose edges connecting lower level nodes or children nodes are ANDed and some node whose edges are ORed. When a sensor nodes' edge reaches a fault description node that is not ANDed with other sensor nodes, that fault description associated with the L-level node can be asserted or is validated immediately. When edges are ANDed, all the conditions associated with each ANDed edge for other sensor nodes must be satisfied before the fault description may be asserted. Sometimes conditions may be collectively defined among the ANDed edges instead of individually associated with each edge. Consequently, all the fault descriptions associated with the parent nodes on the same or higher level of the asserted fault node are validated.

Evaluating an "if" clause in a rule for a certain sensor is equivalent to checking the label of an edge connected to that sensor node. One sensor node can be connected to several L-level nodes or H-level nodes with the same or different edge label values. When a condition on an edge connected to a sensor node is satisfied, the

appropriate fault node connected to that edge is visited and associated fault rules are invoked through "check". "Check" is referred to as a semantic primitive function [Figure 14] and permits the KBS to probe the occurrence of a fault. These rule groups invoked through "check" are known as fault description rule groups. Fault description rule groups correspond to both the H-level or the system decomposition level and the L-level or the disturbance instantiation level. Rule groups that describe the faultiness of a part of the system are part of the H-level (i.e. battery system failure rules, no system voltage rules, solar array system failure rules). When an H-level node is asserted, it means that the component represented by that node is "abnormal" or "faulty". Rule groups that describe a specific fault instance name correspond to the L-level (i.e. battery short rules, load connect rules, battery open rules). Each L-level node is associated with a fault and rules asserting that fault are invoked by traversing the edge connected to the sensor nodes.

Finally, when a fault can be confirmed from the observed sensor values, another semantic primitive function called "assert" validates the confirmed fault. The "check" function appears in the "then" part of rule groups and are considered to be mutually disjunctive. That is, if a check results in an assertion of a fault, then follow that

```

;
; semantic primitives available in the if part of the rules
;
(setq *if_language_dictionary* (sort '(is
                                     is_not
                                     exceeds
                                     below
                                     changed
                                     increased
                                     decreased
                                     indicates
                                     contents_of
                                     all_of
                                     none_of
                                     one_of
                                     same_value
                                     value_of) #'string-lessp) )

;
; semantic primitives available in the then part of the
; rules
;
(setq *then_language_dictionary* (sort '(assert
                                     check) #'string-lessp) )

```

Figure 14. Semantic Primitive Functions in AMPERES

diagnostic path and abandon other checks that might appear in the same "then" part. However, once a fault is asserted, fault descriptions with deeper (or higher) abstraction levels are automatically asserted through DIAG.

An example of rules associated with a node representing the "battery cell open" fault is shown in Figure 15. In the rule, significant changes in the sensor values are used rather than number values to enhance the capability and accuracy of the diagnosis. The variable "normal" is used rather than explicit values since normal value ranges may vary depending on the current operational context. This greatly simplifies the composition of rules.

The terms "one_of", "changed", "stay", and "assert" all appear as the first keyword in the "if" and "then" clauses and are included in the group of semantic primitive functions. The variable "present sensor" is a variable receiving the minor sensor, which is the second element in the pairs from the SCL.

An example of rules associated with the "over current" fault nodes in L-level is shown in Figure 16. Rules can be recursively invoked along the distribution feeders to find out the source location of the over current. These rules may be invoked by traversing the edge connected from the present minor sensor node whose sensor value has suddenly increased or changed from normal to some

```

(setq battery_cell_open_rules '(
  (rule 1
    (if (one_of present_sensor is_a
              battery_cell_voltmeter)
        (changed present_sensor from normal to zero)
        (stay (all_of battery_cell_voltmeter in
                     battery_module (present_sensor)
                     except present_sensor) normal)))

    (then (assert component (present_sensor) open_fault)))

  (rule 2
    .
    .
    .
  ))

```

Figure 15. An Example of Battery Cell Open Fault Rules

```

(setq over_current_rules '(
  (rule1
    (if (is_present_sensor one_of 'load_ammeter))
    (then
      (assert present_sensor 'over_current) ))
  (rule2
    (if (is_not_present_sensor one_of 'load_ammeter))
    (then
      (check children_ammeters 'over_current )
      (check present_sensor 'ground_fault)
      (check present_sensor 'short_circuit) ))
)
)

```

Figure 16. An Example of Over Current Fault Rules

new value. For example, suppose the main ammeter A_m is showing a sudden value increase and the over current rules are invoked. Since A_m is not a load ammeter and rule 1 is not satisfied, the conditions associated with other ANDed edges connected to the load ammeter nodes, $A_{11}..A_{110}$, are checked. Then over current rules are invoked again for each of the load ammeters if the labeled condition is satisfied.

CHAPTER IV

RESULTS AND ANALYSIS

AMPERES was tested to demonstrate its ability to diagnose faults accurately and within an acceptable period of time. These results only analyze the implementation of the fault diagnosis module and not the sensor value validation method. Therefore, it is assumed that all sensor values are valid. The test cases used to validate AMPERES consisted of simulated data sets with known faults.

One important question when analyzing the results from the various test cases is whether or not a precise measurement can determine the level of success in this KBS. Recall that the goals of this KBS are to provide continuous monitoring of the space power system, to alleviate the problem of accumulating experience and expertise in monitoring space power systems, and to reduce the possibility of misoperation due to the cascading effect of a fault. Using the above goals, the measurement of success can be defined as the ability of AMPERES to diagnose faults accurately and within an appropriate response time.

Based on past experience of terrestrial power systems, the anticipated faults for AMPERES appear in Table 1. A description of these faults can be found in the Appendix. These faults were injected into AMPERES using

Table 1. Test Cases Applied To AMPERES

Test Case Number	Test Case Description	Elapsed Real Time To Detect Fault (sec.)	Total CPU Time To Detect Fault (sec.)
1	Normal Data		
2	Load Connect	1.00	.30
3	Load Over Current	.48	.04
4	Load Disconnect	.62	.26
5	Battery Ground	1.38	.34
6	Battery Short	.84	.22
7	Battery Open	.52	.10
8	Solar Array Failure	.58	.38

the FILSD simulation device and the data simulation program. Response time results for the processing of the fault diagnosis module to detect a fault appear in Table 1. The test cases used to validate this system included a normal data simulation, load connect and disconnect, load over current, battery open, battery ground, battery short, and solar array system failure. These test cases demonstrate AMPERES ability to continuously and consistently diagnose faults as well as concentrate on the initial source of the disturbance without being distracted by consequences of the disturbance.

All the known faults in the test cases were diagnosed accurately and within an acceptable period of time. AMPERES was able to continuously acquire operational data while the fault diagnosis module diagnosed the various faults. Due to limitations in the RAM on the SUN 386i, memory swapping was noticed which was a definite factor in the response time.

A sample display of AMPERES in execution showing various faults that were injected are shown in Figure 17. The four windows in this display are used for operating status displays and consist of the System Operating Status Window, the Major Operational Parameter Window, the Operator Interface Window, and the Fault Record Window. The System Operating Status Window shows the one line

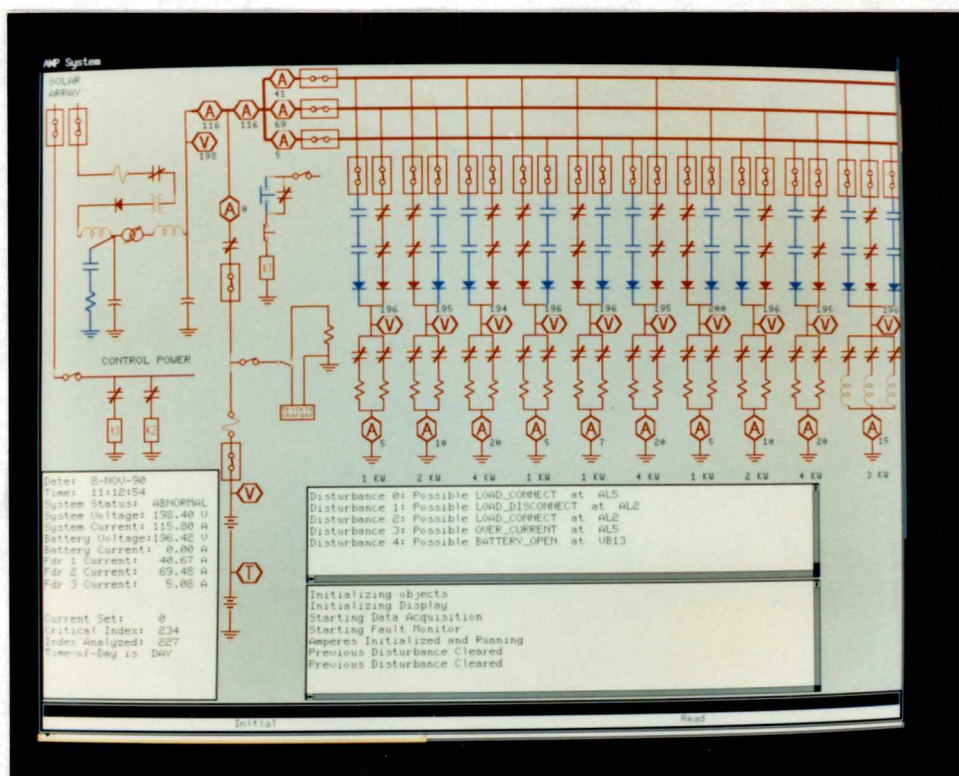


Figure 17. Display of AMPERES

diagram of the system and the current positions of all the circuit breakers and the magnetic switches. It also displays the major meter readings such as load currents, feeder currents, main current, main voltage, and battery voltage. The Major Operational Parameter Window displays major operational information of the power system including the present time, main system voltage, main system current, present data buffer locations, and overall system state. The Operator Interface Window displays explanations about the current operating state. The Fault Record Window displays a brief description of the disturbances detected.

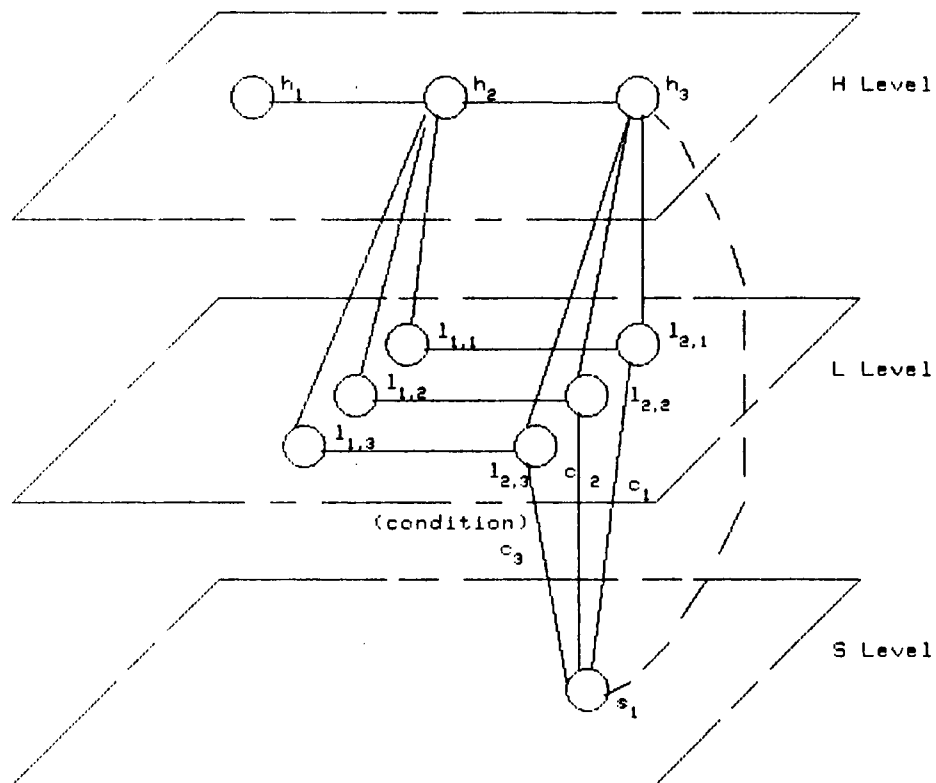
To demonstrate AMPERES ability to execute under normal conditions, test case number one consisted of a normal data simulation. No faults were detected using test case one and the system status window in the display of AMPERES indicated a normal status.

Test cases number two and four verified the load connect and disconnect conditions. Load ammeters A_{12} and A_{15} were connected using a simulated data set from the data simulation program on the personal computer. Load ammeter A_{15} was then disconnected. All three disturbances were accurately diagnosed by AMPERES and appeared in the user interface window of the AMPERES' display. By having specialized knowledge groups for sensor types and faults and generic descriptions of the various faults, only four

rules in the knowledge base were traversed for the load connect condition. This test case demonstrates the advantage/strength of having a small, coherent, natural, and clear rule representation in the KB.

Test case number three illustrates AMPERES ability to focus in on the initial source of a disturbance rather than being distracted by a consequence. This test case also verifies the recursive invocation of the over current rules. The load ammeter A_{15} was the sensor passed to the fault diagnosis task in AMPERES with the over current fault condition. As a consequence of A_{15} 's present value showing an increase, the main ammeter A_m 's present value also shows a sudden increase. AMPERES accurately diagnosed the over current condition on the load ammeter A_{15} since the "present_expected_range" of A_m includes the sudden increase. As a result, the main ammeter, A_m , is not included in the SCL and not sent to the fault diagnosis task.

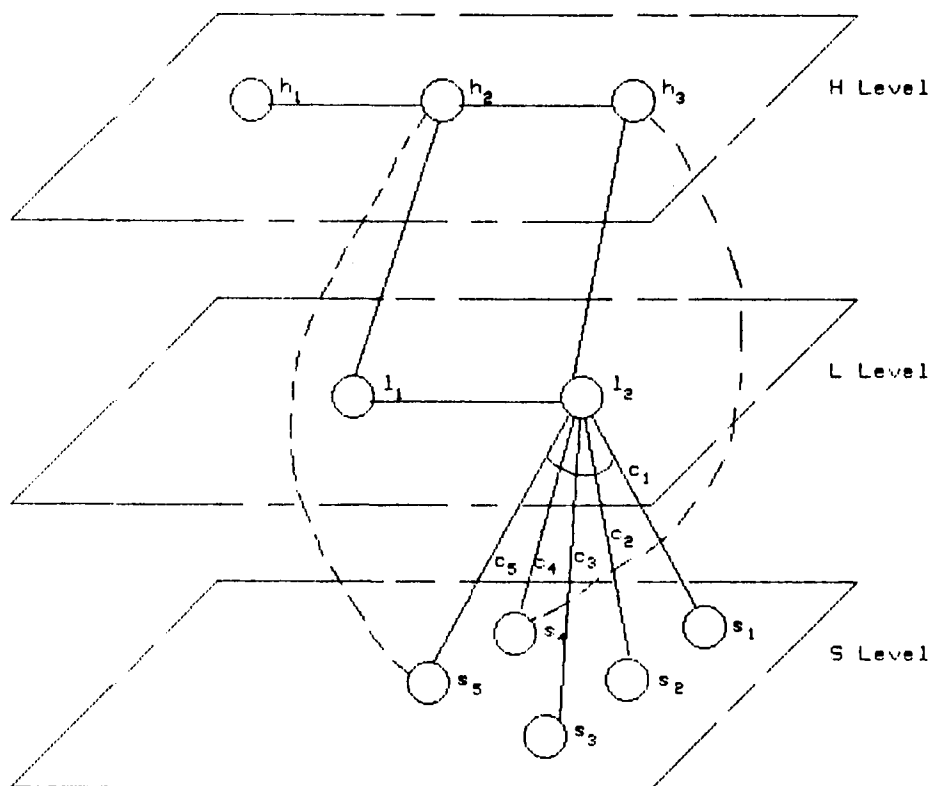
The DIAG for the load connect, load disconnect, and the load over current test cases is shown in Figure 18. DIAG graphically shows the diagnosis reasoning paths used by the fault diagnosis task to diagnose these three test cases. When the load connect, load disconnect, or the load over current conditions are asserted, the faults associated with the nodes to the left and above are also validated.



- | | |
|--------------------------------------|---|
| h_1 : System Faulty | $l_{2,1}$: Load Ammeter 5 Over Current |
| h_2 : Load Feeder Faulty | $l_{2,2}$: Load Ammeter 5 Disconnect |
| h_3 : Load Ammeter 5 Faulty | $l_{2,3}$: Load Ammeter 5 Connect |
| $l_{1,1}$: Load Feeder Over Current | s_1 : Load Ammeter 5 A ₁₅ |
| $l_{1,2}$: Load Feeder Disconnect | c_1 : $s_1 >$ present expected range |
| $l_{1,3}$: Load Feeder Connect | c_2 : s_1 changed from normal to zero |
| | c_3 : s_1 changed from zero to normal |

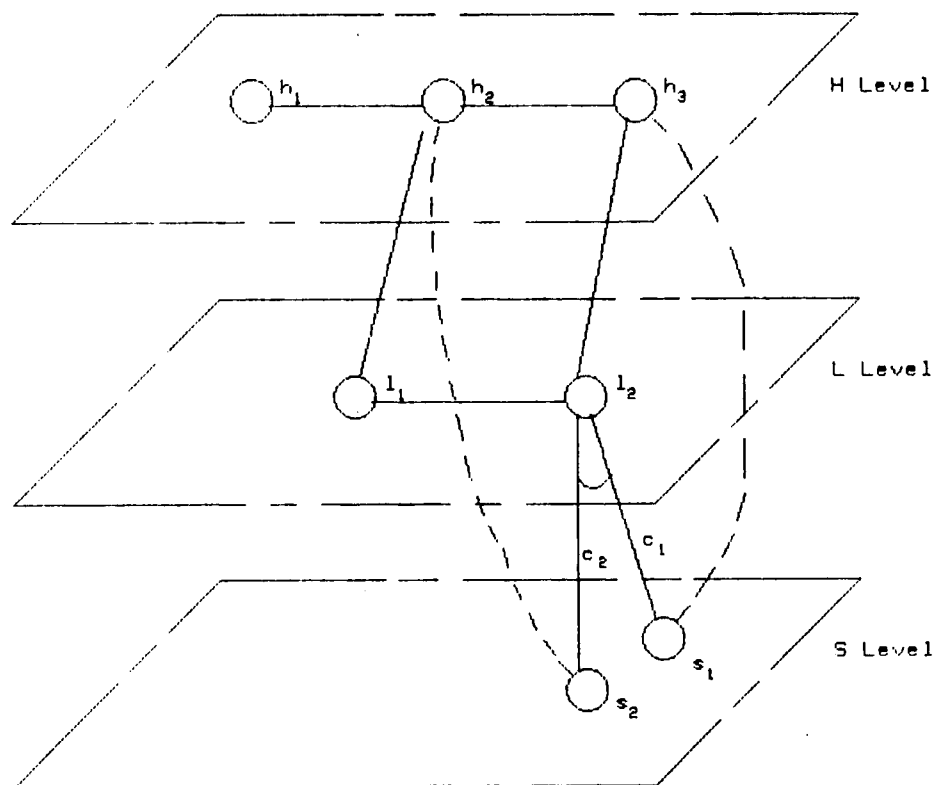
Figure 18. DIAG for the Load Connect, Load Disconnect, and Load Over Current Fault Diagnosis

The battery ground, short, and open faults were verified in test cases number five, six and seven. Different battery module voltmeters were used in each of these test cases to illustrate AMPERES ability to diagnose faults on different sensors. Battery module V_{b4} was accurately diagnosed for the battery ground fault. Battery module V_{b13} was used to verify the battery open fault and battery module V_{b20} was used in the battery short fault. The DIAG's for the battery ground, battery short, and battery open fault diagnosis are shown in Figure 19, Figure 20, and Figure 21 respectively. The battery modules used in the test cases are used in these graphs, however any of the battery modules can be replaced in these graphs. These test cases illustrates the powerful reasoning mechanism in the DIAG approach. There are twenty-eight battery modules in the battery system as well as one hundred sixty-eight battery cells. A human monitoring the space power system might take a considerable amount of time to determine which battery module is shorted or open. AMPERES is able to diagnose these three test cases accurately and within approximately ten seconds. In the battery open test case, the explanations or justifications of how the battery open fault was determined can be found in an explanation list. These explanations are the faults that can be asserted while propagating through the sensors logical environment.



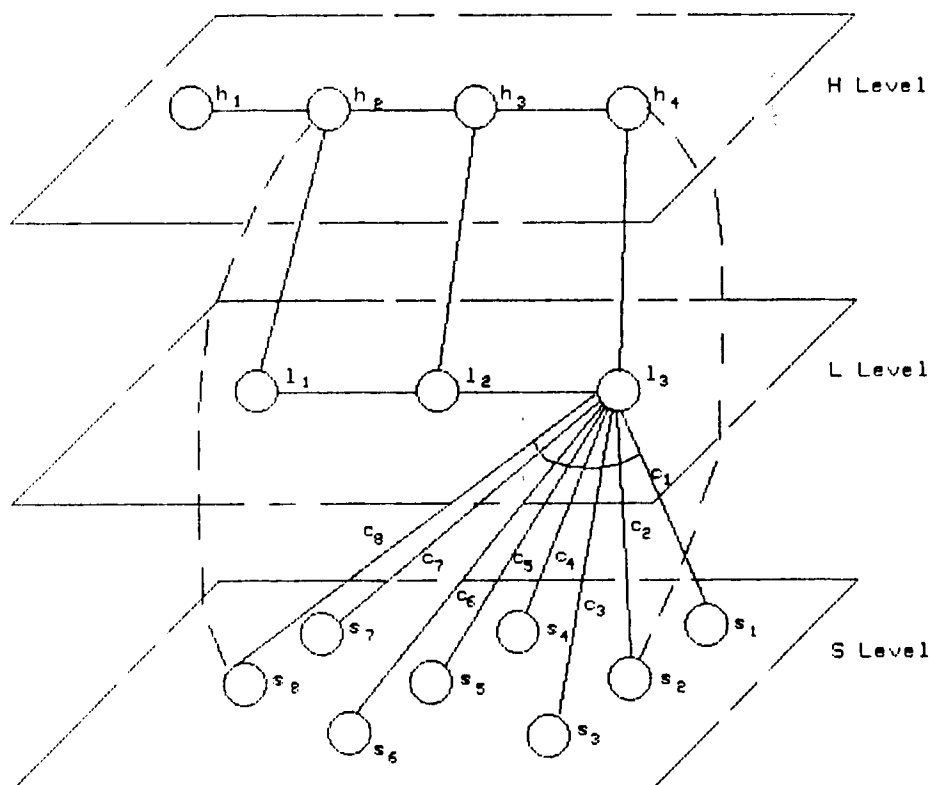
h_1 : System Faulty	s_4 : Battery Module 4 Voltage U_{b4}
h_2 : Battery System Faulty	s_5 : Battery System Voltage U_b
h_3 : Battery Module 4 Faulty	c_1 : $s_1 = 0$
l_1 : Battery System Ground	c_2 : $s_2 = 0$
l_2 : Battery Module 4 Ground	c_3 : $s_3 = 0$
s_1 : Battery Module 1 Voltage U_{b1}	c_4 : $s_4 = 0$
s_2 : Battery Module 2 Voltage U_{b2}	c_5 : s_5 will have some reduced value
s_3 : Battery Module 3 Voltage U_{b3}	

Figure 19. DIAG for the Battery Ground Fault Diagnosis



h_1 : System Faulty	s_1 : Battery Module 20 Voltage U_{b20}
h_2 : Battery System Faulty	s_2 : Battery System Voltage U_b
h_3 : Battery Module 20 Faulty	c_1 : $s_1 = 0$
l_1 : Battery System Short	c_2 : s_2 will have a reduced value
l_2 : Battery Module 20 Short	

Figure 20. DIAG for the Battery Short Fault Diagnosis



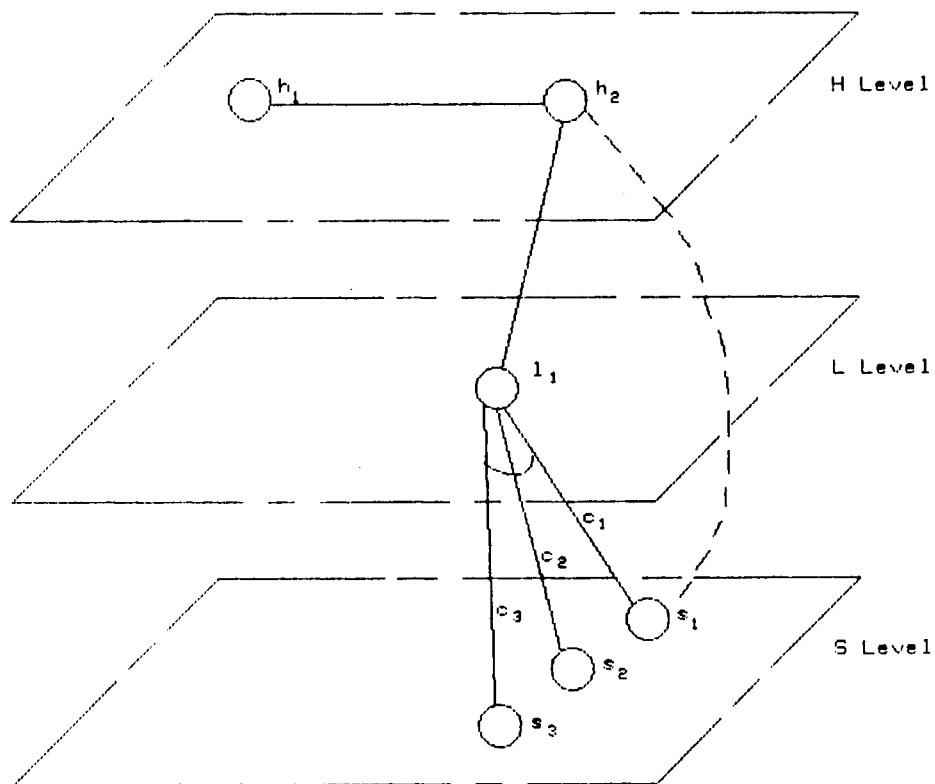
h_1 : System Faulty	s_7 : Battery Module 13 Voltage U_{b13}
h_2 : Battery System Faulty	s_8 : Battery System Voltage U_b
h_3 : Battery Module 13 Faulty	c_1, c_3, c_4, c_5, c_6 : s_1, s_3, s_4, s_5, s_6 have normal values
h_4 : Battery Cell 74 Faulty	c_2 : $s_2 = 0$
l_1 : Battery System Open	c_7 : $s_7 = 0$
l_2 : Battery Module 13 Open	c_8 : $s_8 = 0$
l_3 : Battery Cell 74 Open	
$s_1 - s_6$: Battery Cell Voltages 73-78 $U_{c73} - U_{c78}$	

Figure 21. DIAG for the Battery Open Fault Diagnosis

This important part of the DIAG approach provides operators of space power systems with justifications of how a fault was determined.

Test case eight verifies the solar array system failure fault with the solar array ammeter, A_s , and also demonstrates AMPERES' ability to examine pre and post fault data. A solar array system failure in AMPS can be caused by either the main ammeter (A_m), the main voltmeter (V_m), or the solar array ammeter (A_s). To determine which sensor caused the solar array system failure, it would be helpful to examine pre fault data as well as post fault data. For example, if the main ammeter caused the disturbance, then the battery ammeter (A_b) would have changed from "charge" to "discharge" and the main voltmeter would have changed from "normal" to some "higher value". Having access to pre fault data enables AMPERES to determine these important facts. The DIAG for the solar array system failure fault diagnosis with the solar array ammeter A_s causing the fault is shown in Figure 22.

These eight test cases illustrate AMPERES' strengths which include the ability to look at pre and post fault data, concurrently acquire data and diagnose faults within acceptable time constraints and differentiate between the source of a fault and the consequences of a fault. The inference scheme, formalized with DIAG, enables AMPERES to



h_1 : System Faulty
 h_2 : Solar Array System Faulty
 l_1 : Solar Array System Failure
 s_1 : Solar Array Ammeter A_s
 s_2 : Main Ammeter A_m
 s_3 : Battery Ammeter A_b

$c_1 : s_1 = 0$
 $c_2 : s_2 = s_3 = \text{same positive value}$
 $c_3 : s_3 = s_2 = \text{same positive value}$

Figure 22. DIAG for the Solar Array System Failure
Fault Diagnosis

propagate through the sensors logical environment with the expectation for relevant faults as well as provide justifications about each sensor value along the logical path. Another important strength of AMPERES is the KB representation which includes generic descriptions of the faults instead of specific descriptions of all the possible fault cases.

CHAPTER V

CONCLUSIONS

The overall implementation scheme for a real-time operating state assessment and fault diagnosis knowledge based system for space power systems has been described. The organization of AMPERES including the critical and non-critical buffering, the concurrent task handling, as well as the fault diagnosis scheme development of the sensor value validation method and the DIAG formalism have been discussed.

The validation results give useful information or clues about the ongoing situation to the fault diagnosis KBS. The DIAG approach facilitates the implementation of the diagnosis process and enables the AMPERES to provide a fault description in several different abstraction levels and to answer queries for such different fault descriptions. Initiating the fault diagnosis process from looking at a sensor value and checking each potential fault facilitates the diagnosis of multiple faults.

The generic descriptions of faults of various sensors keeps the knowledge base small and coherent. The "context switching" concept along with the development of semantic primitive functions enable the rule representation in the knowledge base to be more natural and clear. Combined with

the DIAG data structure, the inference and knowledge representation scheme implemented in AMPERES provides a powerful reasoning tool for fault monitoring and diagnosis tasks.

LIST OF REFERENCES

LIST OF REFERENCES

Chen, J. and Srihari, S., "Candidate Ordering and Elimination in Model-based Fault Diagnosis", Proceedings of the Eleventh International Joint Conference on Artificial Intelligence, pp. 1363-1368, August 1989.

Console, L., Dupre', D.T., and Torasso, P., "A Theory of Diagnosis for Incomplete Causal Models", Proceedings of the Eleventh International Joint Conference on Artificial Intelligence, pp. 1311-1317, August 1989.

de Kleer, J., and Williams, B., "Diagnosis with Behavioral Modes", Proceedings of the Eleventh International Joint Conference on Artificial Intelligence, pp.1324-1330, August 1989.

Gallanti, M., Roncato, M., Stefanini, A., and Tornielli, G., "A Diagnostic Algorithm based on Models at Different Level of Abstraction", Proceedings of the Eleventh International Joint Conference on Artificial Intelligence, pp. 1350-1355, August 1989.

Gupta, U.K., and Ali, M., "Leader- An Integrated Engine Behavior Real-Time Fault Diagnostic Expert System for Space Shuttle Main Engine (SSME)", Proceedings of the Second International Conference on Industrial & Engineering Applications of Artificial Intelligence & Expert Systems, June 1989.

Hayes-Roth, F., Waterman, D., Lenat, D., Building Expert Systems, Addison-Wesley Publishing Company, 1983.

Kessler, R., LISP, Objects, and Symbolic Programming, Scott, Foresman and Company, 1988.

Lee, S.C., "Sensor Value Validation Based on Systematic Exploration of the Sensor Redundancy for Fault Diagnosis KBS", Submitted to IEEE Transactions on Systems, Man, and Cybernetics, 1990.

Lee, S.C., Ratliff, M., Peterson, C., "AMPERES: A Real Time Fault Monitoring and Diagnosis KBS for Space Power Systems", Applied Artificial Intelligence an International Journal, May 1991.

Lee, S.C., Roepke, R., Ratliff, M., Wilhite, L., Sherley, Pl, and Peterson, C., "The Fault Monitoring & Diagnosis Knowledge-Based System for Space Power Systems", Project

Report, Center for Commercial Development of Space, Auburn Space Power Institute, November 1990.

Obreja, I., "Diagnosis of Power Plant Faults Using Qualitative Models and Heuristic Rules", Proceedings of the Third International Conference on Industrial & Engineering Applications of Artificial Intelligence & Expert Systems, July 1990.

Poole, D., "Normality and Faults in Logic-Based Diagnosis", Proceedings of the Eleventh International Joint Conference on Artificial Intelligence, pp. 1304-1310, August 1989.

Rafea, A., EL-Desouki, A., and EL-Moniem, S., "Combined Model Expert System for Electronics Fault Diagnosis", Proceedings of the Third International Conference on Industrial & Engineering Applications of Artificial Intelligence & Expert Systems, July 1990.

Rich, Elaine, Artificial Intelligence, McGraw-Hill Book Company, 1983.

Struss, P. and Dressler, O., "Physical Negation - Integrating Fault Models into the General Diagnostic Engine", Proceedings of the Eleventh International Joint Conference on Artificial Intelligence, pp. 1318-1323, August 1989.

Waterman, D., A Guide to Expert Systems, Addison-Wesley Publishing Company, 1986.

APPENDIX

APPENDIX

FAULT DESCRIPTIONS

LOAD CONNECT

A load connect condition occurs when the sensor value of a load ammeter changes from a previous value of zero to some new positive value. The previous value is obtained by retrieving the sensors value at $t-50\text{ms}$ (present time - 50 milliseconds). This condition can occur with the load ammeters $A_{11}..A_{110}$.

LOAD OVER CURRENT

A load over current fault occurs when the present value of a load ammeter exceeds its present expected range. This fault can occur with the load ammeters $A_{11}..A_{110}$.

LOAD DISCONNECT

A load disconnect condition occurs when the sensor value of a load ammeter decreases from a previous non-zero value to a value of zero. The previous value is obtained by retrieving the sensors value at $t-50\text{ms}$ (present time - 50 milliseconds). This condition can occur with the load ammeters $A_{11}..A_{110}$.

BATTERY GROUND

A battery ground fault occurs if the battery voltmeter V_b shows some reduced voltage value and one of the battery module voltmeters along with all the battery module voltmeters below it have a present value of zero. For example, if the battery module voltmeter V_{b14} were grounded, then the present voltage values of V_{b1} through V_{b14} would be zero and V_b would indicate a reduced value.

BATTERY SHORT

A battery short fault occurs when one of the battery module voltmeters V_{b1} through V_{b28} has a present value of zero and the remaining battery module voltages have normal values. Consequently, the battery voltmeter V_b will show some reduced voltage value.

BATTERY OPEN

The battery open fault occurs if one of the battery cell voltmeters has a value of zero. Recall that each battery module contains six battery cells. Therefore, the remaining five battery cells in the battery module indicating a battery open will have a normal value. As a result, the battery module voltmeter and the battery voltmeter will have values of zero.

SOLAR ARRAY FAILURE

The solar array failure can occur one of three ways.

The first way is if the solar array ammeter A_s has a value of zero during the day and the main ammeter A_m and battery ammeter A_b have the same positive value, then the solar array ammeter has caused the failure. The second way for a solar array failure to occur is if the main voltmeter V_m has a value of zero during the day. This results in the main voltmeter causing the solar array failure. The third way results in the main ammeter A_m causing the disturbance and occurs when the value of the battery ammeter A_b changes from "charge" to "discharge" (from a negative to a positive) and the main voltmeter value changes from normal to some higher value.

VITA

Michelle Weisheit Ratliff was born with her twin brother in Evansville, Indiana on December 27, 1962. She attended West Henderson High School in Hendersonville, N.C. where she was voted "most talented" and graduated fifth in her class in May 1981. The author received the Allen H. Sims Leadership Scholarship and attended Brevard College in Brevard, N.C. where she graduated magna cum laude with an Associate in Arts degree in May 1983. While at Brevard College, Mrs. Ratliff was a member of the Phi Theta Kappa honor society, a residents' assistant, and president of Jones Dormitory. She then attended North Carolina State University where she was treasurer of the Upsilon Pi Epsilon Computer Science Honor Society and was a co-op student with Northern Telecom in the Research Triangle Park, N.C. She was a member of the National Guild of Piano Playing for 10 years. Mrs. Ratliff graduated cum laude and received the Bachelor of Science degree in Computer Science in May 1986. She then accepted a computer science position at Calspan Corporation in Tullahoma, Tennessee where she also began her studies toward the Master of Science degree in Computer Science. She was awarded this degree in August, 1991.