IN-63-CR
166961
32P.

JPL Publication 88-7

Knowledge-Based Diagnosis for Aerospace Systems

David J. Atkinson

(NASA-CR-182641) KNOWLEDGE-BASED DIAGNOSIS FOR AEROSPACE SYSTEMS (Jet Propulsion Lab.) 32 p CSCL 09B

N89-11441

Unclas
G3/63 0166961

March 15, 1988

NASA
National Aeronautics and Space Administration

Jet Propulsion Laboratory
California Institute of Technology
Pasadena, California

JPL Publication 88-7

Knowledge-Based Diagnosis for Aerospace Systems

David J. Atkinson

March 15, 1988

National Aeronautics and
Space Administration

Jet Propulsion Laboratory
California Institute of Technology
Pasadena, California

The research described in this publication was carried out by the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration.

Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise, does not constitute or imply its endorsement by the United States Government or the Jet Propulsion Laboratory, California Institute of Technology.

Abstract

This document discusses autonomous diagnosis of aerospace systems. The need for automated diagnosis in aerospace and the approach of using knowledge-based systems are examined. Research issues in knowledge-based diagnosis which are important for aerospace applications are treated along with a review of recent relevant research developments in Artificial Intelligence.The design and operation of some existing knowledge-based diagnosis systems are described. The systems described and compared include the LES expert system for liquid oxygen loading at NASA Kennedy Space Center, the FAITH diagnosis system developed at the Jet Propulsion Laboratory, the PES procedural expert system developed at SRI International, the CSRL approach developed at Ohio State University, the StarPlan system developed by Ford Aerospace, the IDM integrated diagnostic model, and the DRAPhys diagnostic system developed at NASA Langley Research Center.

Preface

This technical report is a preprint of a chapter which will be published as part of a book titled, "Machine Intelligence and Autonomy for Aerospace Systems." The book will be a volume of the AIAA Series Progress in Astronautics and Aeronautics. However, the earliest estimate of the date of publication of this book is some time in the spring of 1988. I have decided to issue this technical report in the belief that the technology of knowledge-based systems for automated diagnostic applications is of substantial immediate interest to engineers, technical staff, and aerospace managers who are currently developing, plan to develop, or are simply curious about the area.

The reader will note that I have to a large extent avoided discussion of much of the background, rationale, or basic technology behind knowledge-based systems. Many other high quality references for this already exist. I have also chosen not to describe simple rule-based systems for diagnosis in much detail. This particular implementation approach was one of the first to be applied to the problem of diagnosis (as part of the MYCIN system developed by Shortliffe at Stanford in the late 1970's) and its success instigated the development of many commercial products for developing rule-based systems. Instead, the chapter primarily discusses the overall architecture of knowledge-based approaches to diagnosis which may or may not be suitable for implementation using rule-based systems.

David J. Atkinson
Jet Propulsion Laboratory

4 September 1987

PRECEDING PAGE BLANK NOT FILMED

PAGE iv INTENTIONALLY BLANK

TABLE OF CONTENTS

Figures

PRECEDING PAGE BLANK NOT FILMED

PAGE vii INTENTIONALLY BLANK

1.0 Introduction

This document will discuss autonomous diagnosis of aerospace systems. In particular, we will examine the need for automated diagnosis in aerospace and the approach of using knowledge-based systems for this purpose. We will also discuss some of the research issues in knowledge-based diagnosis which are important for aerospace applications. Finally, we will review some recent research developments in this area of artificial intelligence and describe the design and operation of some existing knowledge-based diagnosis systems.

A critical aspect of the process of evaluating the performance of spacecraft and ground support systems and determining their ability to meet mission objectives is in assessing the state-of-health of these systems. The traditional approach to diagnosis of aerospace systems is to anticipate all one-point and two-point failure modes. Elaborate checklists are constructed which, it is hoped, will serve to identify all of these failure modes and corrective actions. The problem which arises is that as the complexity of spacecraft and ground systems increases, the resources required to anticipate failure modes and construct exhaustive checklists becomes combinatorially explosive. Furthermore, as a diagnostic tool, checklists seldom embody the rationale for the procedures which are being followed. This can make it tedious and difficult for humans who are performing checklist actions to focus on the immediate problem at hand. Finally, an important consideration in fault diagnosis is that quick response to failures may be critical. The ability of human ground operators or astronauts to compensate for a failure during diagnosis, determine a diagnosis with incomplete or partial information, and quickly institute a recovery procedure diminishes as system complexity increases. These considerations make the process of diagnosis a desirable candidate for automation.

Conventional automation techniques are insufficient in many monitoring, diagnostic, or maintenance applications (Richardson, 1984). Control and diagnostic mechanisms in these approaches cannot be dynamically matched to the exigencies of the situation. They are typically inflexible and cannot easily accommodate the reconfiguration or modification of a device under test. Such systems are usually unresponsive to the varying degrees of skill of the different technicians who use them. Poor real-time performance is also symptomatic of conventional automation approaches to diagnosis.

2.0 Knowledge For Diagnosis

Knowledge-based systems for diagnosis provide an approach to automating much of the process of assessing the state-of-health of spacecraft and ground systems. The demonstrated potential is great for overcoming many if not all of the limitations of conventional automation approaches mentioned in the previous section.

One way to characterize the various Artificial Intelligence approaches to diagnosis is by the type of knowledge which is encoded and used in the diagnostic reasoning process. This knowledge may be divided into two predominant areas: Heuristic or experiential knowledge, and model-based knowledge. Heuristic knowledge generally attempts to capture the inferential techniques of humans who have expertise in diagnosing faults in the given problem domain. Model-based knowledge attempts to represent the structural, functional, and causal information about the system being diagnosed. Some diagnostic expert systems utilize knowledge of both types.

Heuristics which are the result of a human expert's experience with problem-solving in a domain are sometimes called *expert knowledge* and the programs which use this knowledge *expert systems.* While the scope of high-performance knowledge-based systems certainly includes more techniques than this particular approach, the heuristic knowledge expert systems approach to diagnosis has a long history in artificial intelligence research, particularly in the domain of medicine. (For examples of knowledge-based medical diagnosis systems and related issues, see Szolovits, 1982.)

Heuristic diagnostic knowledge has the advantage of being relatively easy to represent or encode in a knowledge base. Human experts often find it convenient to describe their diagnostic inferences in terms of assertions and conclusions which may be derived in certain situations. This type of knowledge is usually represented by production rules which encode inferential knowledge as a set of *situation - action pairs.* The selection and invocation of these rules typically forms a deductive chain of reasoning when the rules are applied in a particular order. Since the order of rule invocation is largely dependent on the problem at hand, these systems can be very sensitive to the data being processed (or *data driven)* in a way that strictly procedural approaches cannot. Heuristic inferences implemented as production rules encode a high level view of the expert's decision making processes. For this reason, diagnosis using heuristic knowledge is frequently called *shallow reasoning.*

When the diagnostic problem is to automate the largely routine diagnoses of human experts, the heuristic approach to diagnosis is most appropriate. Heuristic knowledge has also been frequently employed when inferences are

PRECEDING PAGE BLANK NOT FILMED PAGE 2 INTENTIONALLY BLANK

uncertain due to missing or erroneous knowledge or symptomatic data, or in other situations when the deductive search space is combinatorially large. Heuristic knowledge derived from experience is occasionally the only type of knowledge which may be used in an expert system when the domain for problem solving is not well understood.

In diagnosis of aerospace systems, heuristic knowledge must frequently be combined with *model-based knowledge*. Model-based knowledge breaks free of many of the limitations of human expertise by representing the structure and function of the individual components of the system being diagnosed, often at many different levels of abstraction. Causal interactions between components and allowable diagnostic inferences are often explicitly stated in model-based knowledge, whereas they may be only implicitly represented in heuristic knowledge. As such, model-based knowledge provides a more complete representation of the system and its behavior than heuristic knowledge. This property can lead to better problem-solving near the periphery of the system's knowledge where heuristic knowledge might be applied inappropriately. Ultimately, in the case of novel failures much if not all of the heuristic knowledge derived from experience will fail, and more fundamental knowledge may need to be used exclusively to locate a failure.

For these reasons, diagnosis using model-based knowledge is frequently called *deep reasoning*, or *reasoning from first principles.* Model-based diagnosis is applicable when human expertise is unavailable or incomplete, or when detailed explanations for failures must be provided beyond attribution to particular expert rules. This situation is usually the case in diagnosis for aerospace systems, where humans have not had sufficient experience with current or future systems to build a substantial body of diagnostic knowledge.

Whether they include heuristic or model-based knowledge, some combination of both, or other new approaches, knowledge-based systems for diagnosis of aerospace systems must confront a number of theoretical and practical issues. The following sections review a selection of the current research and developments in knowledge-based diagnosis which have applicability to aerospace diagnosis domains.

2.1 Deep Model Diagnostic Systems

Diagnosis with deep knowledge about the structure and function of systems is a relatively new and promising area in knowledge-based systems. Theoretical developments and approaches are varied (see for example (Davis, 1985)(Genesereth, 1982)(de Kleer, 1986)). However, relatively few systems have matured into actual prototype applications or operational systems. One example of a diagnostic system using deep models which has

been developed for a space system application is discussed in the following paragraphs.

The problem of detecting and isolating faults in the LOX portion of the Space Shuttle Launch Processing System (LPS) has led to the development of the LES system (*Liquid Oxygen (LOX) Expert System*) at the Kennedy Space Center (Scarl, 1985). LOX loading of the shuttle's external tank begins six to eight hours prior to launch and continues to within seconds of the launch (See Figure 1).

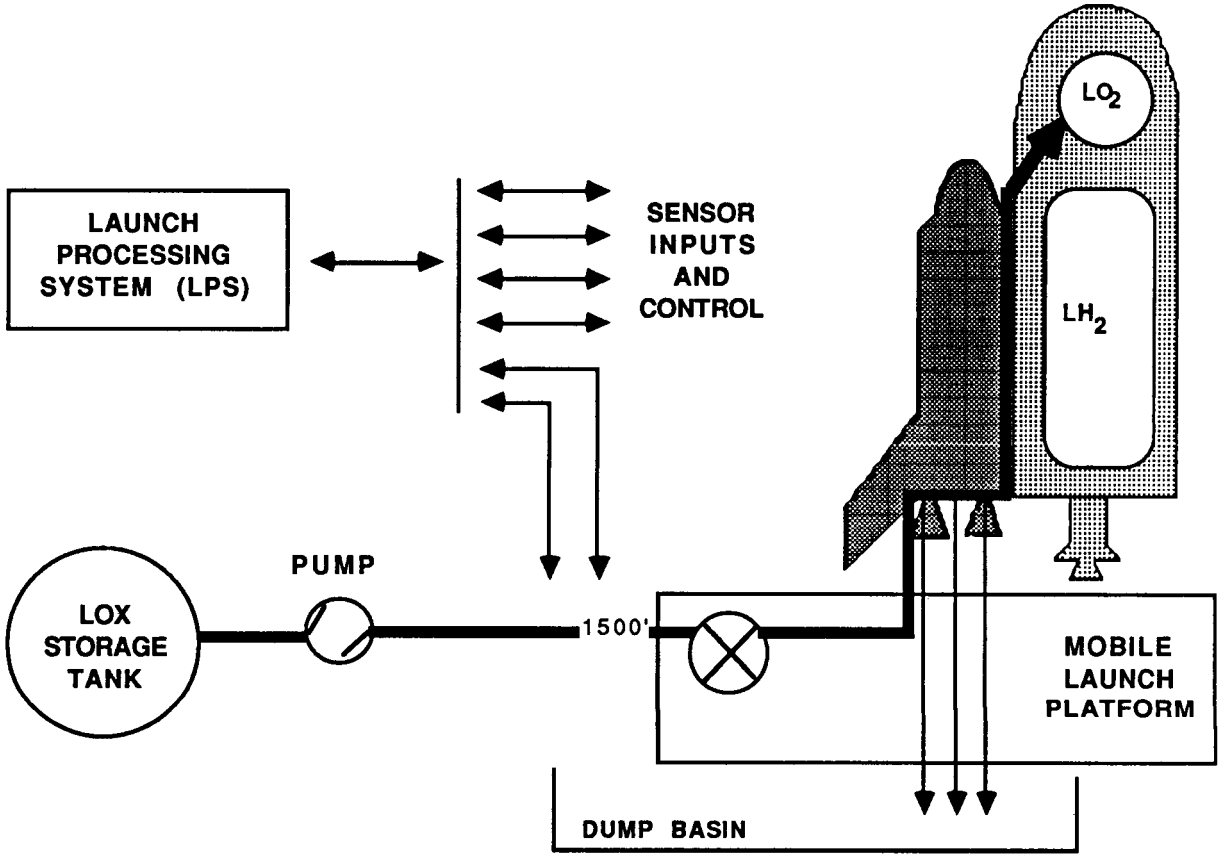

FIGURE 1: LOX Transfer at KSC

The LOX loading process is controlled by KSC's automated LPS which monitors critical sensor information on the LOX process such as temperature and pressure. When a sensor reading is outside of a prespecified range, troubleshooting must occur to determine whether the anomaly is an actual problem or a sensor malfunction. Delays in troubleshooting can easily result in a mission abort. The goal of LES is to troubleshoot the sensor system and determine if a reported fault requires termination of the LOX loading process.

LES monitors multiple sensors, including sensors on electrical, pneumatic, hydraulic, and mechanical systems. The system has been tested on data tapes of Shuttle missions and will be demonstrated during actual Shuttle launches. In typical example situations, LES determines a diagnosis in about ten seconds compared to up to twenty minutes for a human expert.

The knowledge base in LES includes a functional representation of most of the LOX portion of the Launch Processing System at KSC. This includes analog and discrete commands and sensors; other objects such as transducers, relays, solenoids, valves, LOX pressures and temperatures; and other objects, quantities, and relationships whose state the system is designed to control or sense. Functional relationships between subsystems are encoded in the knowledge base as connectivity between knowledge base objects which represent LPS subsystems. The particular representation technique used to represent knowledge about subsystems is FRL (*Frame Representation Language*) (Roberts, 1977).

When sensor data is received, LES uses information about the status of the system and functionally connected systems to determine if the reading is discrepant or nominal. This may involve checking the status of a large number of connected subsystems. Expected sensor values are derived from the functional representation of the system. Thus, when a received piece of sensor data is determined to be discrepant, LES immediately has a handle on potential sources of the problem based on the derivation of the expectation. This means that a diagnostic procedure need not exhaustively consider all of the subsystems and can concentrate on a few likely candidates. A strictly heuristic approach to diagnosis of this system would not have this benefit.

LES begins diagnosis when a discrepancy is noted. Using the process of tracing the functional connections between subsystems, the system collects suspected faulty subsystems, relevant commands, and sensor data which are related to the original discrepancy. Suspected faulty components are proved to be operating properly through a series of reasoning steps. For example, when there are intervening objects in the functional connection to the noted discrepancy which are known to be operating nominally, then the suspected component could not be malfunctioning or the effect would be propagated to the intervening subsystems. Components which are suspected of faults may also be proved to be operating correctly when there is no difference between the expected nominal output of the component and a hypothetical faulty output. The expected output is derived from the hypothesized inputs from other functionally connected components and a model of the suspect component. The hypothetical faulty output is correspondingly inferred from the behavior of other components which rely on the suspect component for inputs. Clearly, if an output value produced by nominal operation is identical to an output which is faulty when viewed by functionally "downstream" components, then the detected fault has probably

propagated from a fault in a functionally "upstream" component from the suspect component and the noted discrepancy. A third method which may also prove that a suspect component is operating properly is when the hypothetical faulty output from the suspect component results in a contradiction to the current discrepancy. In this case, a fault in the suspect component could not logically produce the observed faulty behavior in the system.

This apparently simple, yet very effective process of logically deducing first the suspected faulty components based on functional connections and then verifying a fault using a few rules reveals some of the potential for application of model-based reasoning.

2.2 Shallow Model and Heuristic Diagnostic Systems

Several systems have been built which combine elements of heuristic rule-based diagnosis with diagnosis from first principles such as that found in deep model-based reasoners. These systems are sometimes termed *shallow-model systems.* Shallow-model systems sacrifice some of the detailed knowledge found in deep models to gain efficiency and provide a mechanism for utilizing the intuitive reasoning found in heuristic diagnostic systems. Deep model systems typically are computationally expensive and do not easily provide methods for integrating heuristic knowledge.

One such system is the FAITH diagnosis system developed at the Caltech Jet Propulsion Laboratory (Friedman, 1983, 1985). (Another is the ARBY system developed by McDermott and Brooks (McDermott, 1982) which will not be discussed here.) FAITH (for *Forming and Intelligently Testing Hypotheses)* is principally a rule-based system. While production rules in the system may encode heuristic knowledge derived from an expert's experience, they also encode several standard strategies of diagnosis which experts commonly employ in troubleshooting different systems. These strategies, discussed in more detail below, rely on shallow models of the system being diagnosed. Model knowledge is implemented in the system declaratively as statements in the predicate calculus and represents facts about circuit diagrams, system block functional diagrams, subsystem types and hierarchies, and problem hierarchies. The declarative information provided to the system is called a *relational map.*

The inference engine in FAITH utilizes predicate logic in the selection and instantiation of rules and alternates between two basic phases of a diagnostic "cycle". These phases are *Explanation* (using backward chaining) and *Confirmation/Denial* (using forward chaining). In the Explanation phase, hypotheses about the location or characteristics of a fault are proposed. In the corresponding Confirmation/Denial phase, evidence is sought which will confirm the hypothesis or provide a rationale for further refinement. When such

evidence is available, the hypothesis may be refined in consequent Explanation cycles. When the hypothesis is denied, FAITH attempts to make alternative hypotheses. The basic search paradigm in FAITH is a depth first search with chronological backtracking, although methods for heuristically guiding the search towards the "best" hypotheses are provided, as discussed below.

The selection of rules in each phase is governed by the facts which are true in the current context (as in most production rule systems) as well as the context of the *diagnostic strategy* the system is currently using.

We have experimented with several different diagnostic strategies suggested to us by our experts. Strategies are called *modes* in FAITH and include *Trace mode, Focus mode, Fault Propagation mode,* and *Simulation Mode* among others. New modes are easily added to the system using the specialized language FCL (*FAITH Control Language*). FCL is also used for expressing production rules and the other declarative knowledge in the system. Trace mode involves following connections between objects in the system being diagnosed. For example, objects could be integrated circuits (IC) and connections could be electrical connections between different ICs. In general, however, objects and connections in FAITH can represent any form of *adjacency* (Davis, 1985) including functional adjacency (as in a block diagram), physical adjacency (as in a blueprint), electrical adjacency (as in a circuit diagram), thermal adjacency, etc. Focus mode involves consideration of arbitrary groups of objects (such as subsystems) one at a time. If a subsystem hierarchy is represented in the relational map, then Focus mode would look for a fault in a system by attempting to localize a fault in each of the subsystems in the system, and in the subsystems of those subsystems, and so on until a fault could be identified. In general, the groups of systems considered in Focus mode can be arbitrarily selected by an expert using heuristic knowledge or can correspond to some actual organization of the system. This flexibility of representation accorded by the FAITH relational map and associated diagnostic modes is one of the most powerful characteristics of shallow model diagnostic systems.

From our experience with knowledge engineering for the FAITH system, experts will typically change their diagnostic strategy based on the particular anomaly at hand, the subsystems being considered for faults, and the state of diagnosis. FCL provides mechanisms for implementing these tactical changes in diagnostic strategy in an orderly way during diagnosis. The other modes and additional features of FCL are discussed in detail in (Friedman, 1985).

The FAITH system was created primarily as a vehicle for studying knowledge-based automated diagnosis. Development since 1982 has included consideration of several different domains, including the Deep Space Network at JPL, digital logic circuits, and many simple examples. The most

complex application domain implemented thus far has been diagnosis of the Photo-Polarimeter Sensor (PPS) on board the Voyager II spacecraft, now en route to the planet Neptune. The PPS instrument is subject to a variety of non-predictable, recurrent faults which consume the valuable time of expert analysts. Automated diagnosis would speed recovery from these failures and free the expert to concentrate on other matters. In this application, the relational map constructed for FAITH included knowledge about the functional block diagram of the system, circuit diagrams, structure and functional knowledge about individual integrated circuits in the system, and heuristic knowledge used by the expert to quickly employ the different FAITH modes to localize a failure. While the scope of this chapter precludes an extended discussion of this application, interested readers are invited to contact the author for more details. In summary, however, FAITH could identify roughly seventy-six different failures in the systems with approximately two hundred and fifty rules and a relational map with several hundred declarative FCL statements. The application was demonstrated as a prototype, but an operational system was never constructed for budgetary reasons.

2.3 Procedural Knowledge

Malfunction handling frequently involves the use of elaborate procedures, especially in aerospace applications. The Space Shuttle, or Space Transportation System (STS) is an excellent example. The invocation of STS operational procedures requires considerable technical skill on the part of astronauts and mission controllers and attention to myriad constraints, including flight rules and avoidance of harmful subsystem interactions. Procedures such as these represent a wealth of "compiled" knowledge which should be taken advantage of by an intelligent diagnostic system.

However, in constrast to heuristic and model-based reasoning, the task of execution of preformed plans or procedures is an area which has seen little effort in Artificial Intelligence research and development. An exception is the research being conducted by Michael Georgeff and Amy Lansky at SRI International (Georgeff, 1986). They have developed a system for reasoning about and performing complex tasks in dynamic environments. The system is called PES, for *Procedural Expert System.* The knowledge representation which they have designed is capable of describing the effects of arbitrary procedures. The system's inference mechanism is capable of utilizing this knowledge to choose among alternative courses of action and accomplish operational goals. In seeking a domain in which to develop and test their system and ideas about procedural reasoning, Georgeff and Lansky chose the problem of malfunction handling in the Space Shuttle reaction control system (RCS).

Figure 2 shows a portion of an RCS malfunction procedure. This example illustrates some of the characteristics of diagnostic procedures which demand considerable technical skill from human operators and make conventional automation approaches inadequate. For example, the context or time in which a test is made can determine the interpretation placed on the results of an action or test. While this is sometimes reflected in the ordering of actions within a procedure, *context* or *time dependence* can also be a property of the overall goal of the procedure as a whole. In the illustration, a successful hot fire of the RCS indicates that a processor or other input parameter has failed. The identical test and observations in the context of another RCS procedure (not shown) can indicate a trickle current circuit failure. Knowledge of the context of the test is crucial in obtaining the correct interpretation of the results. Several other characteristics of malfunction handling procedures introduce complexity. For example, there may be tests or actions which may be sensible to perform (e.g., to obtain redundant measurements) but which are not strictly necessary to a successful diagnosis. Complexity can also be introduced by the need for special constraints to guard system safety during testing. Complexity in malfunction handling procedures is also high when there are sub-procedures which are dependent on global factors such as the state of other subsystems or the phase of flight.

Faced with this tremendous implicit knowledge embodied in procedures, and the concomitant diagnostic power, Georgeff and Lansky argue that it is neither convenient nor sensible to "deproceduralize" the knowledge required to perform malfunction handling. Most expert system knowledge representation techniques, such as frames or production rules like those found in commercial expert system shells, are not capable of easily representing procedural knowledge. Effective use of this knowledge, they contend, requires elaborate control techniques in knowledge base construction which reduce modularity. Complex control strategies may also confuse interpretation of the rules. This would make later modification extremely difficult.

To address these concerns, the knowledge representation developed by Georgeff and Lansky describes procedures by specifying sequences of goals which the system should try to achieve in executing a given procedure. Goals are represented using a rich formalism which allows such goals as "achieve p while maintain q true" to be easily described. This goal language captures much of the important control knowledge which is lacking in other representations.

Procedures are represented by *recursive transition networks* (RTN) which resemble flowcharts. Arcs in the RTN are labelled with goal descriptions. Different paths through the various nodes in the network constitute the different ways a procedure may be executed. Each procedure also has an associated *effect* which will be realized if the procedure is successfully executed.

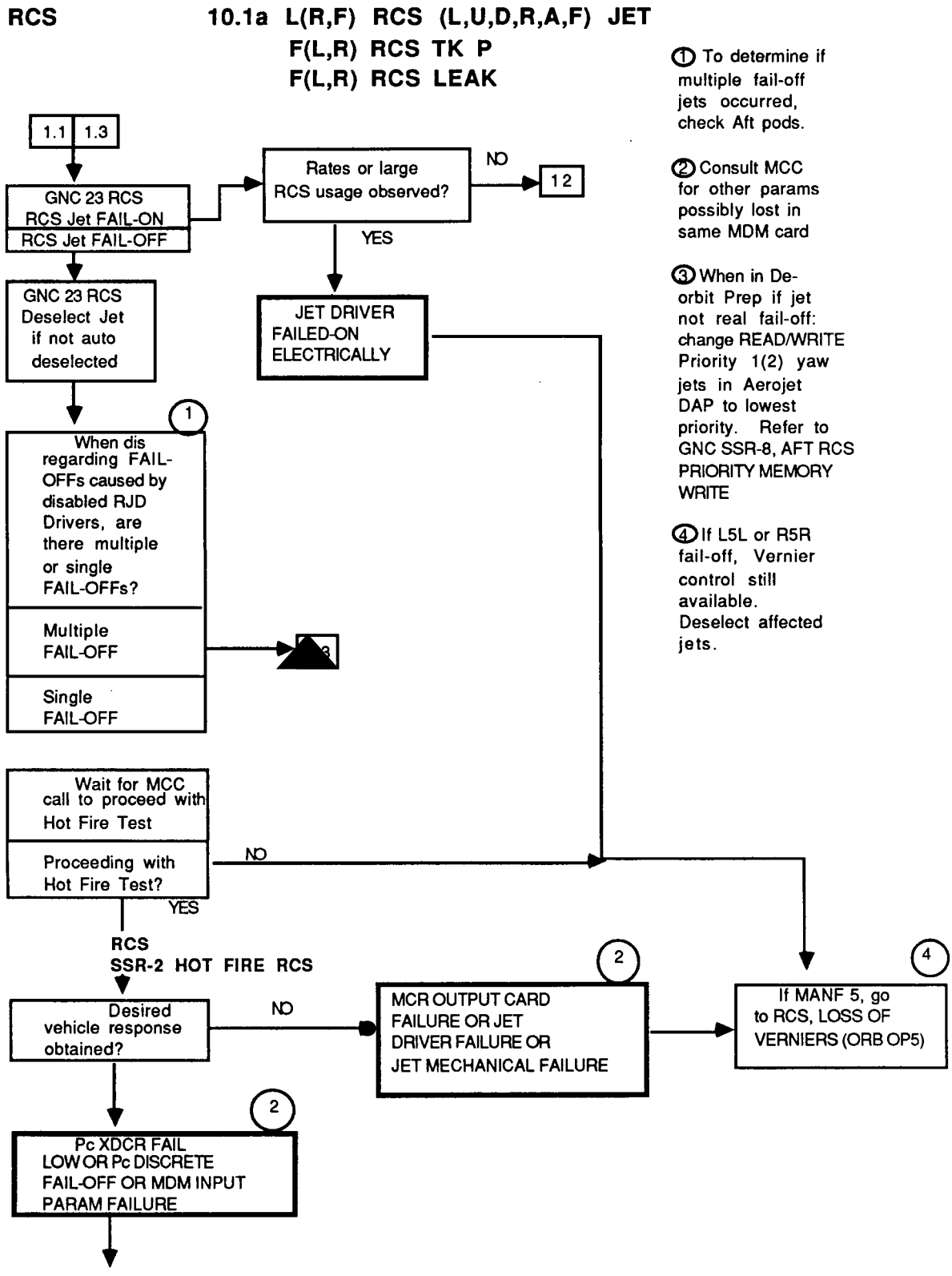

FIGURE 2: Portion of RCS Malfunction Procedure

Figure 3 shows an example procedure from the system. Much of the power of the representation comes from its ability to declare facts about procedures independently of their use. This permits inferencing mechanisms to reason about composite goals and promotes explanation capabilities, verifiability of procedures, and evolvability of the knowledge base. The procedure graphically depicted in the figure may be factually stated as

> *If* a fuel cell has a voltage drop and it can subsequently be determined that the pattern is uniform, and *if* it can thereupon be established that the humidity is high, and *if* finally it is possible to achieve a lower humidity, *then it follows that* the fuel cell will be rendered operable.

Associated with procedures are *invocation conditions* which are statements of the proper circumstances in which procedure invocation can occur. Invocation conditions are arbitrary logical expressions which may include constraints on known facts or active goals. The combination of procedure and invocation condition is called a *knowledge area* (KA). KAs are executed by the inference engine in Georgeff and Lansky's system only when invocation conditions are evaluated and found to be "true".

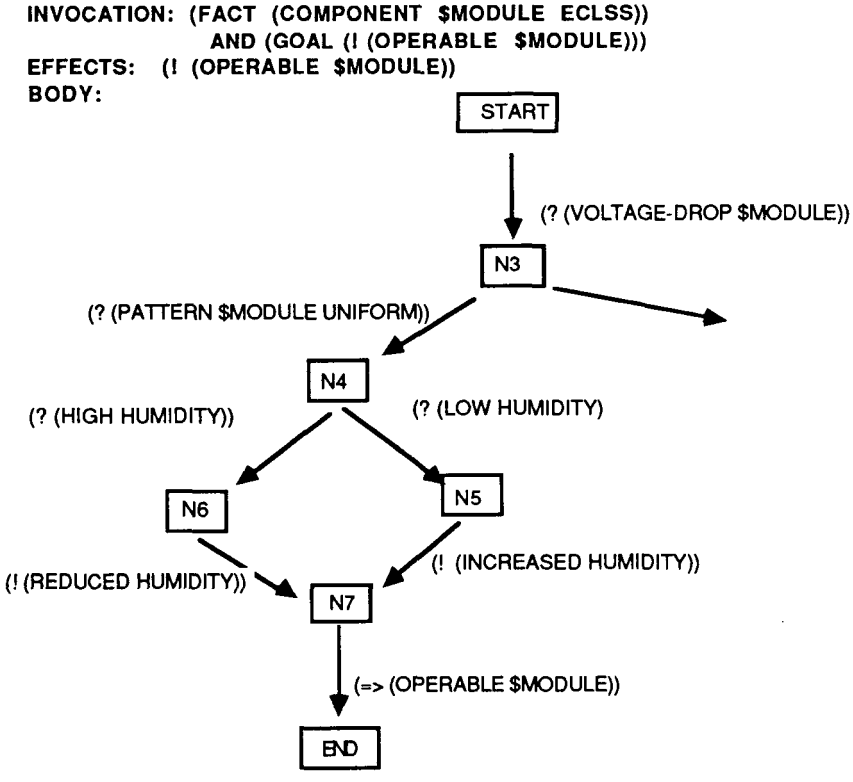

FIGURE 3: Portion of KA for Fuel Cell Malfunction

KAs achieve goals by causing the inference engine to achieve each of the goals along some path in the body of the KA. For example, in the example fuel cell malfunction KA, the system would first try to establish whether a

given fuel cell suffered from a voltage drop. This might involve a simple test directly, *or invocation of some other KA,* thus allowing KAs to chain together in the diagnostic reasoning process. To continue with the example, if a voltage drop was detected, the KA would then test the pattern of voltage loss, then humidity, etc. The combination of an explicit representation of the sequence of goals to achieve along with flexibility in the choice of procedures to achieve those goals is a main source of PES's diagnostic power.

A rule-based representation of the same procedure in our example would be significantly more complex and require multiple rules along with knowledge about how to "chain" them together. Davis and King (Davis, 1977) have provided an analysis of control problems with production rule systems which is still relevant. However, when such control can be conveniently specified, such as in the FAITH system described earlier, production rules can also be an effective representation.

The basic structure of PES from the user's point of view consists of three essential components: (1) a system data base which represents given and derived facts during diagnosis; (2) the domain procedural knowledge encoded as KAs; and (3) the current set of goals which the system is trying to achieve. Finally, the system also includes a sophisticated user interface which allows a system developer to graphically create and manipulate KAs and run an application system. With the widespread availability of checklists for many systems (such as the Shuttle) which require automation of diagnosis processes, systems such as PES which can easily take advantage of that source of knowledge may have substantial utility.

2.4 Problem Hierarchy Specialists

A group of researchers headed by B. Chandrasekaran at Ohio State University has been investigating another promising approach to diagnosis (Chandrasekaran, 1983). Their approach is based on the paradigm of *cooperating specialists.* The central problem-solving view of diagnosis, they contend, is classificatory activity. Strongly associated with this type of problem solving are specialized knowledge and system organizations, and special strategies for performing diagnosis.

Classificatory diagnosis is the act of identifying the current problem description in terms of a specific node in a predetermined diagnostic hierarchy, where nodes correspond to diagnostic hypotheses. General hypotheses are at the top of the hierarchy, and more specific hypotheses are towards the bottom. Such a hierarchy could (but need not) represent disease classes and specific diseases in those classes.

The Ohio State approach (called CSRL (Chandrasekaran, 1983) and exemplified in several systems, e.g., MDX, which diagnoses cholestatic diseases, and Auto-Mech, which troubleshoots automobile fuel systems) is

unique in that associated with each node in the classification hierarchy is a *specialist* which contains the specific diagnostic knowledge necessary to evaluate the presence or absence of the problem in the current case description. Knowledge is thus highly localized in systems built using CSRL rather than highly distributed as would be the case in a conventional rule-based system.

The basic strategy of diagnosis in the system is one of *hypothesis refinement*. In this strategy, specialists seek to establish the hypotheses they represent. At some level of confidence, a specialist may conclude that the current hypothesis is valid or that more specialized hypotheses are worth pursuing. In this case, the specialist invokes its *sub-specialists*, i.e., those specialists which represent more specific diagnoses in the hierarchy. Diagnosis thus proceeds down the classification hierarchy towards more specialized diagnoses and halts when a specialist with no sub-specialists has been established. (The problem of when problem-solving like this should terminate in general is of significant theoretical interest.)

Figure 4 shows an example classification hierarchy for an application of this approach in the domain of auto repair. The hierarchy is for diagnosis of the fuel system in 1980 model era automobiles.

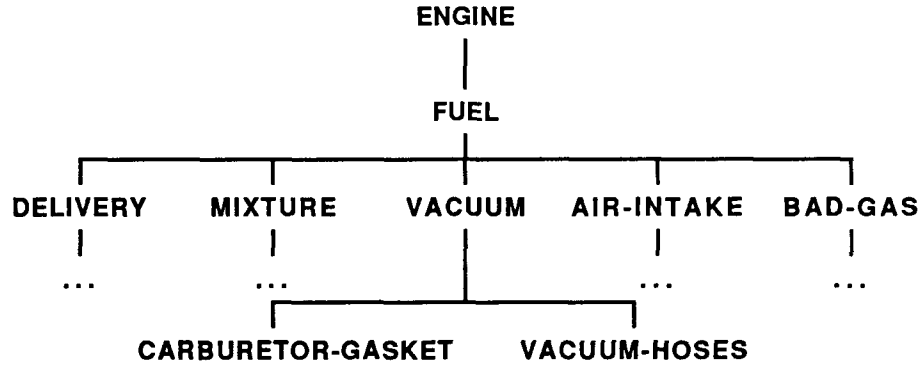

FIGURE 4: Partial Diagnostic Hierarchy for CSRL Auto-Mech

The approach of having the location and application of diagnostic knowledge tightly constrained yields some valuable benefits for the development of diagnostic systems. Since knowledge is localized within specialists and interaction among specialists is well-defined, concerns about global interaction are minimized in this approach. This supports the process of incremental specification and refinement of the knowledge bases commonly required in the development of knowledge-based systems. In the CSRL system, portions of the knowledge base may be developed in relative isolation and then executed. Real performance such as this provides more meaningful feedback to system developers than comments on a knowledge base representation implemented only on paper. However, the developers of

the system point out that significant problems remain to be solved, including how a variety of types of knowledge and problem-solving procedures can be integrated with the classificatory strategy.

A similar approach has been adopted by Siemens, Golden, and Ferguson at Ford Aerospace & Communications Corporation in their development of an expert system for satellite monitoring and anomaly management (Golden, 1985), (Siemens, 1986). While the objectives of their system are more comprehensive than the Ohio State project (including monitoring, situation assessment, goal determination and planning as well as diagnosis), their approach to diagnosis has many of the same functional characteristics of the CSRL approach.

Like the CSRL approach, diagnostic knowledge in the Ford system (called StarPlan) is organized hierarchically into anomaly classes. In StarPlan, specialized knowledge and procedures for determining specific anomalies are called *monitors.* Associated with each monitor is a *guardian* module which scans incoming telemetry for data symptomatic of failures which may be diagnosed by the monitor. A significant amount of control knowledge is incorporated to handle multiple alarms and other conditions arising from the interaction of the two types of modules. Each monitor contains goal-driven rule-sets which are specialized towards identification and resolution of the specific anomaly class, as in the CSRL specialist approach.

As Siemens et al. discuss, there are significant problems in managing the interaction of the multiple "mini" expert systems embodied in a distributed approach like their own and CSRL's. For example, the control of diagnosis suffers in both systems when a single fault anomaly introduces multiple symptoms, thereby activating multiple specialists in the CSRL system and multiple monitors in the Ford system. These individual expert systems then work on independent hypotheses which may utilize conflicting and sometimes contradictory diagnostic procedures. In StarPlan, a level of *meta-monitors* has been described (but not implemented) to handle this situation. Meta-monitors determine which hypothesis within a group of monitors is most urgent and then provide control of diagnosis to that hypothesis.

Meta-monitors in StarPlan are usually centered around individual satellite subsystems. This is one way in which object-oriented classes and problem-oriented classes can interact in systems of this type. They point out, however, that sometimes the structure of the partitions between different classes and the object and anomaly classes is counter-intuitive and therefore may inappropriately constrain descriptions by domain experts. Since control strategies based on Meta-monitors defeat StarPlan's orientation towards highly modularized knowledge bases and introduce significant control structure issues of their own, recent designs for a successor to StarPlan (called StarPlan II) have abandoned the Meta-monitor approach in favor of a model-based representation based on individual satellite objects and their

functional behavior. Unfortunately, details of the new approach are discussed only in proprietary Ford Co. documentation and are not open to general scrutiny.

3.0 Coordinating Hybrid Knowledge in Diagnostic Systems

Earlier sections have described knowledge-based systems for diagnosis which utilize different kinds of knowledge, knowledge representations, and inference mechanisms. However, some types of diagnostic reasoning must make use of multiple types of knowledge. One expert troubleshooting technique is to use heuristic, experiential knowledge to quickly isolate candidate faults and then rely on deeper causal or model-based knowledge to analyze the problem in detail and eliminate incorrect hypotheses.

A knowledge-based diagnostic system which seeks to exploit this type of technique needs a control structure which can switch easily between the different types of reasoning strategies. Such a system also requires multiple knowledge representations suited to each type of knowledge employed (sometimes called "hybrid" knowledge systems) and may need mechanisms to easily translate or refer knowledge from one representation to another.

Fink (Fink, 1985) has addressed the problem of coordinating two very different diagnostic techniques utilizing different knowledge representations and reasoning strategies. Her system, called IDM (*Integrated Diagnostic Model*) has the purpose of diagnosis and repair of electro-mechanical systems. It consists of three main modules: an *Experiential expert*, a *Physical expert,* and an *Executer* [sic]. The role of the Executer is to direct the problem-solving process utilizing each of the other modules as appropriate to the problem at hand and the context of the diagnosis.

Each of the Experiential and Physical expert modules has its own knowledge base. The Experiential expert knowledge base encodes the initial experiential, heuristic causal reasoning for familiar problems. This includes low-level facts and observables, such as voltage measurements, and easily deducible quantities from those facts based on experience, e.g., physical states of the device such as a generator not recharging a battery. Finally, the Experiential knowledge base encodes solutions in the form of quick fixes for well recognized problems. The general structure of the knowledge base resembles that of Casnet (Weiss, 1978).

The Physical expert knowledge base encodes deeper knowledge of the system to be diagnosed, in particular, a "physical/functional" model of the device. This knowledge, represented in object-oriented style using a semantic network based on frames, supports qualitative simulation of the device (Kuipers,1984).

The Executer module has unique knowledge which represents how the functional connections between individual objects in the Physical expert knowledge base are correlated with the heuristic knowledge in the Experiential expert knowledge base. This knowledge base also includes information about how to transfer knowledge back and forth during problem-solving.

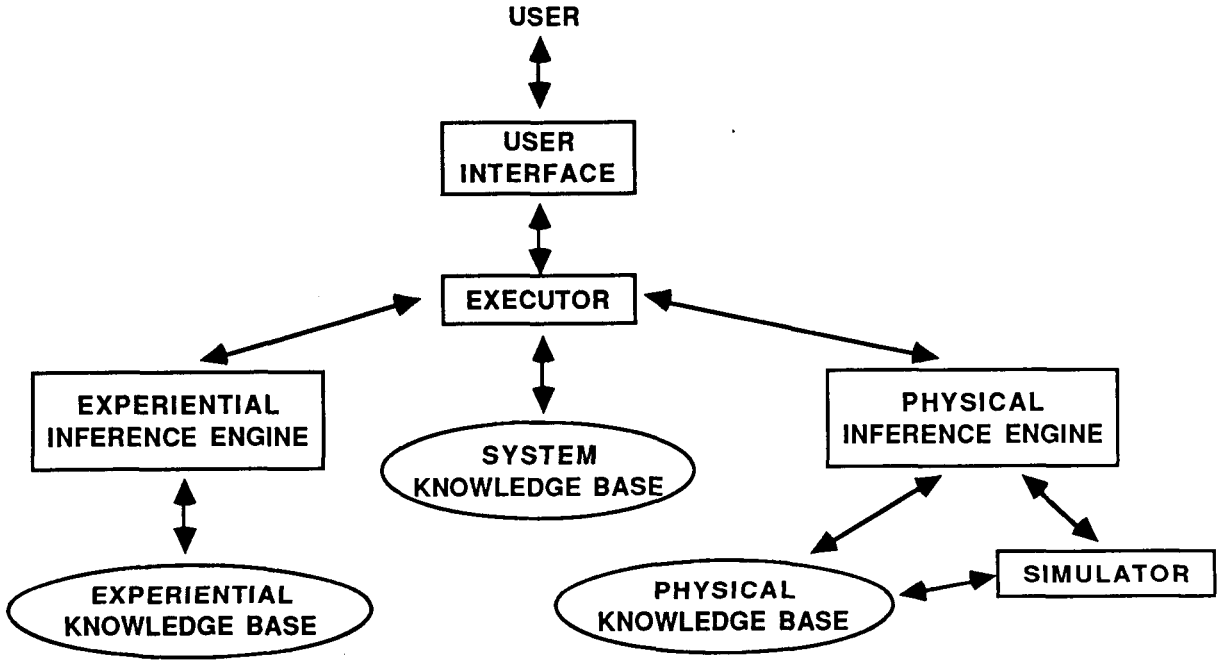

FIGURE 5: Overview of the Integrated Diagnostic Model (IDM)

While an open research topic is to clarify the nature of the knowledge required by the Executor and to design suitable representations and inference mechanisms, the other modules utilize more conventional control mechanisms. The Experiential expert utilizes a best-first search based on an evaluation of a rating of each alternative hypothesis. Ratings are established based on probability, cost, or difficulty of achieving a diagnosis based on the hypothesis. The Physical expert utilizes a version of discrepancy detection methodology originated by Davis (Davis, 1985) and followed by other systems (such as the LES system, described earlier).

One unique contribution of Fink's IDM system is that it does not require a consistent view of the problem between the two expert modules. The Executer module arbitrates any disagreements between modules and exerts control over the propagation of information between the two "loosely coupled" expert systems. The techniques being developed as part of this research will be important for hybrid diagnostic systems which must utilize the knowledge from a variety of different human experts who employ troubleshooting strategies based on their own experience as well as more fundamental knowledge about the structure and function of the system they are diagnosing.

A diagnosis system being developed by Kathy Abbott at NASA Langley Research Center to assist aircraft flight crews also includes multiple knowledge-based modules with disparate knowledge representations and reasoning strategies (Abbott, 1985a, 1985b, 1986). Unlike Fink's IDM system described above, Abbott's program, called DRAPhys (for *Diagnostic Reasoning About Physical Systems*), includes knowledge-based modules which are tightly coupled to a specific strategy of diagnostic reasoning. In addition, DRAPhys also includes a knowledge-based module which performs qualitative interpretation of sensor data as a pre-process to diagnosis. An explanation system which presents the result of the diagnosis to the flight crew is currently being developed.

The input to the fault monitoring and diagnosis process in DRAPhys is quantitative sensor data. The fault monitor compares the sensor data with the output of a quantitative model that simulates the normally functioning physical system. A fault is signaled by the monitor when the expected system state derived from the system model differs significantly from the actual system state. When a fault is detected, the monitor provides the diagnostic process with a set of the abnormal sensor values in qualitative form (e.g., "fuel flow is high") along with time tags to show the sequence of symptoms.

DRAPhys' diagnostic process is divided into several discrete stages as shown in Figure 6. Each stage has a unique knowledge representation and diagnosis strategy. The first stage utilizes heuristic, experiential knowledge compiled from the expertise of aircraft flight crews to compare fault symptoms with known fault types and failure modes. The most commonly occurring faults are detected and diagnosed in this stage. However, the first stage will be unable to identify the cause of failures which are unusual or difficult to diagnose from the qualitative sensor information provided.

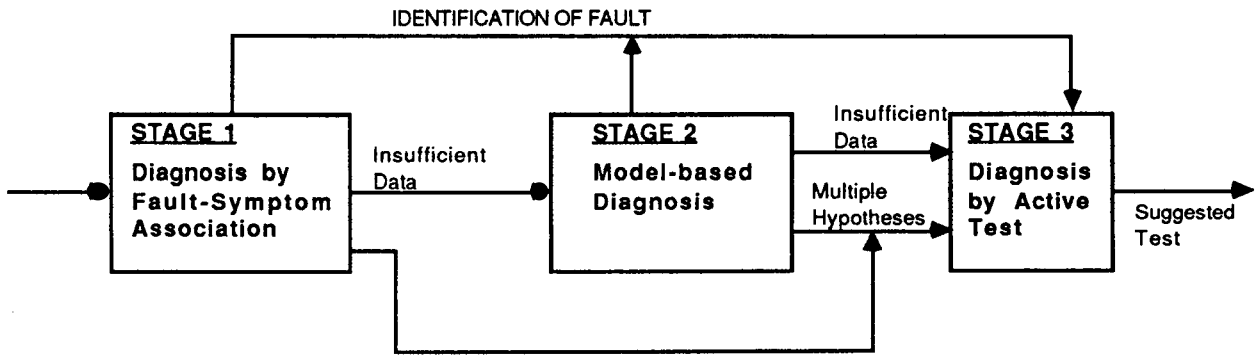

FIGURE 6: DRAPhys Multi-Stage Diagnostic Process

The second stage of DRAPhys' diagnostic process is a second knowledge-based system which is based on a functional model of the underlying system.

The objective of this stage is to localize a fault by devising hypotheses based on how the effects of a fault would propagate through the aircraft's systems. When this process (described in more detail below) fails to identify a unique failed aircraft subsystem, the third stage of diagnosis is entered.

The purpose of the third and final stage of diagnosis is to suggest "active" tests which can be executed by the pilot to provide additional, useful information for diagnosis. The results of this test are interpreted by various stages to eliminate alternative hypotheses and provide a unique diagnosis to the pilot. When a unique diagnosis cannot be identified, the diagnostic modules provide information to the explanation mechanism to inform the pilot of the potentially failed functions and reduced capability of the aircraft.

Abbott's DRAPhys system is specifically directed towards real-time fault diagnosis. As she points out (Abbott, 1985a), real-time fault diagnostic systems differ substantially from other systems which can afford to perform fault diagnosis off-line. In particular, the purposes which those systems serve are substantially different. In real-time fault diagnosis, the objective is less to identify the specific *cause* of a failure and more to discover the *effects* of a failure on the functionality of the system as a whole and what remedial actions are appropriate. The need to identify all affected subsystems with reduced performance is important.

Real-time performance affects the information available for diagnosis and the particular reasoning strategies which may be employed. One consideration is that a physical system's behavior may change as time progresses while performing fault diagnosis. During diagnosis, failure effects may propagate to other functionally or physically connected subsystems. This dynamically changes the failure symptoms with which the diagnosis system must reason. Abbott's program is designed to make unique use of this dynamically changing information about the system to identify the specific physical cause(s) of a failure, the fault type, responsible and affected system components, and the fault propagation history.

Each of the stages of the diagnostic process is able to utilize the sequence of changing fault symptoms to focus the reasoning process and eliminate false hypotheses. The first stage includes a rule-based system which was extended to permit temporal reasoning functions (Allen, 1984). This helps capture pilots' knowledge about changing symptoms associated with specific failures, e.g., when a foreign object is ingested by the turbofan engine, a pilot described the symptoms saying, "First, performance values will fluctuate, then EGT and EPR will decrease ...". Using dynamic information early in the diagnostic process helps to distinguish among faults which may have the same initial symptoms but diverge in subsequent behavior.

The functional and physical models used by the second stage of diagnosis can be thought of as a directed graph. A functional interaction or physical adjacency between two subsystems is represented as an arc in the graph.

The diagnostic process in stage two attempts to map failure symptoms to specific components in the models, and then determine the additional affected components by tracing through the graph. The time-order of symptoms benefits this process by suggesting or confirming a sequence of components affected by a failure. The first component in a functionally connected sequence of components exhibiting failure symptoms is deduced to be the component responsible for the failure. A model of physical adjacency is used to resolve ambiguity, such as when a fault propagates physically between subsystems which are not functionally connected.

Abbott points out a characteristic of the model-based approach to diagnosis which should be considered in the design of such systems. Typically, diagnoses are made in this approach by comparison of actual behavior against the nominal behavior predicted in a functional model of the system. However, modelling a faulted system can be computationally complex because of the number and variety of failures which are possible. One solution to this problem is to provide a functional model which is very *under-constrained,* i.e., the model represents a large set of behaviors rather than the small subset of behavior classified as "nominal". A simple model of functional adjacency between components, such as the one used in DRAPhys, is capable of representing many faulted configurations. Qualitative causal models also have this advantage.

4.0 Conclusion

In this document, we discussed several different approaches to automated diagnosis using knowledge-based systems. One way of characterizing these different approaches is by the type of knowledge which they employ, how the knowledge is represented, and the inference mechanisms required to perform the diagnostic reasoning. Several examples of systems were presented to illustrate the wide variety of types of knowledge and reasoning mechanisms, and also to expose some of the theoretical and practical issues in automated diagnosis. A significant issue for the future is how knowledge-based diagnostic systems can make use of multiple types of knowledge. Several approaches to this problem were also discussed.

Very few knowledge-based systems for diagnosis have completed the developmental process and entered into actual day-to-day operations. Several, such as the Kennedy Space Center LES system, are close. With the promise and need for higher performance automated diagnosis, more of the approaches discussed above will mature and find application in the aerospace domain.

PRECEDING PAGE BLANK NOT FILMED

PAGE 22 INTENTIONALLY BLANK

BIBLIOGRAPHY

Abbott, Kathy H. *Strategies and Representations for Onboard Aircraft Fault Diagnosis.* SIGART Newsletter. Association for Computing Machinery. No. 92. April 1985a.

Abbott, Kathy H. *Exploration of Expert Systems Concepts for Onboard Diagnosis of Faults in a Turbofan Aircraft Engine.* Proceedings of the American Control Conference. Boston, MA. 19-21 June 1985b.

Abbott, Kathy H. *Using Dynamic Behavior of Physical Systems for Real-time Fault Diagnosis: An AI Approach.* IEEE Transactions on Systems, Man, and Cybernetics: Special Issue on Diagnostic Strategies. (Draft copy). 1986.

Allen, J. *Towards a General Theory of Action and Time.* In Artificial Intelligence. Vol. 23. 1984. pp. 123-154.

Chandrasekaran, B. *Distributed Knowledge-based Systems for Diagnosis and Information Retrieval.* Report 763180/714659. Department of Computer and Information Sciences. The Ohio State University. Columbus, Ohio. Also AFOSR-TR-84-0039. Air Force Office of Scientific Research. November 1983.

Davis, R. and King, J. *An Overview of Production Systems.* Machine Intelligence 8. Ed. Elcock and Michie. John Wiley. 1977.

Davis, R. et al. *Diagnosis Based on Description of Structure and Function.* Proceedings of the National Conference on Artificial Intelligence. American Association for Artificial Intelligence. Pittsburgh, PA. 18-20 August 1982.

Davis, R. *Diagnostic Reasoning Based on Structure and Behavior.* In Qualitative Reasoning about Physical Systems. Ed. Bobrow, D. MIT Press. Cambridge, MA. 1985.

de Kleer, J. *Reasoning About Multiple Faults.* Proceedings of the Fifth National Conference on Artificial Intelligence. American Association for Artificial Intelligence. Philadelphia, PA. 11-15 August 1986. pp. 132-139.

Dietrich, E.S., Imamura, M.S. *An Expert System Concept For Autonomous Spacecraft Energy Management.* Proceedings of the 18th Intersociety Energy Conversion Engineering Conference. IEEE, AICHE, ANS, SAE, ACS, AIAA, ASME. Orlando, FL. 21-26 August 1983.

Fink, Pamela K. *Control and Integration of Diverse Knowledge in a Diagnostic Expert System.* Proceedings of the Ninth International Joint Conference on Artificial Intelligence. American Association for Artificial Intelligence. Los Angeles, CA.18-23 August 1985.

Friedman, L. *Controlling Production Firing: The FCL Language. Proceedings of the Ninth International Joint Conference on Artificial Intelligence.* American Association for Artificial Intelligence. Los Angeles, CA. 18-23 August 1985.

PRECEDING PAGE BLANK NOT FILMED

PAGE 24 INTENTIONALLY BLANK

Friedman, L. *Diagnosis Combining Empirical and Design Knowledge.* (JPL Internal Document D-1328). Jet Propulsion Laboratory. California Institute of Technology. Pasadena, CA. 15 December 1983.

Genesereth, M. *Diagnosis Using Hierarchical Design Models.* Proceedings of the National Conference on Artificial Intelligence. American Association for Artificial Intelligence. Pittsburgh, PA. 18-20 August 1982.

Georgeff, Michael P., Lansky, Amy L. *A System For Reasoning In Dynamic Domains: Fault Diagnosis on the Space Shuttle.* Technical Note 375. Artificial Intelligence Center. SRI International. January 1986.

Golden, M. and Siemens, R. *An Expert System For Automated Satellite Anomaly Resolution.* Proceedings of the Computers in Aerospace V Conference. AIAA/ACM/NASA/IEEE. Long Beach, CA. 21-23 October 1985.

Kuipers, B. *Commonsense Reasoning about Causality: Deriving Behavior from Structure.* Artificial Intelligence. Vol. 24. Nos. 1-3. Dec. 1984.

Looney, Harry G. and Parnell, Gregory S. *Expert Systems for Space Operations.* Working paper 86-04. Working Paper Series. Department of Operational Sciences. Department of the Air Force. Air University. Air Force Institute of Technology. School of Engineering. Wright-Patterson Air Force Base, Ohio. October 1986.

McDermott, D. and Brooks, R. *ARBY: Diagnosis with Shallow Causal Models.* Proceedings of the National Conference on Artificial Intelligence. American Association for Artificial Intelligence. Pittsburgh, PA. 18-20 August 1982.

Pipitone, Frank. *FIS, An Electronics Fault Isolation System Based on Qualitative Causal Modeling.* Proceedings of the First Annual AAAI Conference on Aerospace Applications of Artificial Intelligence. Dayton Convention Center. Dayton, Ohio. 16-19 September 1985.

Proceedings of the First Annual AI Research Forum, NASA Ames Research Center, Moffett Field, CA. July 22-24, 1986.

Richardson, J. Editor. *Artificial Intelligence in Maintenance: Proceedings of the Joint Services Workshop.* Air Force Systems Command, Air Force Human Resources Laboratory, Brooks Air Force Base, Texas, 1984.

Roberts, R. and Goldstein, I. *The FRL Manual.* Memo 409. Artificial Intelligence Laboratory. Massachusetts Institute of Technology. Cambridge, MA. September 1977.

Scarl, E. et al. *A Fault Detection and Isolation Method Applied to Liquid Oxygen Loading for the Space Shuttle.* Proceedings of the Ninth International Joint Conference on Artificial Intelligence. American Association for Artificial Intelligence. Los Angeles, CA. 18-23 August 1985.

Siemens, R. W., Golden., M., and Ferguson, J. *StarPlan II: Evolution of an Expert System.* Proceedings of the Fifth National Conference on Artificial Intelligence. American Association for Artificial Intelligence. Philadelphia, PA. 11-15 August 1986.

Szolovits, P. *Artificial Intelligence in Medicine.* American Association for the Advancement of Science. Westview Press, Inc. Boulder, CO. 1982.

Tanner, M.C. and Bylander, T. *Application of the CSRL Language to the Design of Expert Diagnosis Systems: The Auto-Mech Experience.* Department of Computer and Information Sciences. The Ohio State University. Columbus, Ohio. November 12, 1983.

Weiss, S. et al. *A Model-Based Method for Computer-Aided Medical Decision-Making.* Artificial Intelligence. Vol 11. No. 2. pp. 145-172. 1978.

TECHNICAL REPORT STANDARD TITLE PAGE

1. Report No. 88-7	2. Government Accession No.	3. Recipient's Catalog No.
4. Title and Subtitle Knowledge-Based Diagnosis for Aerospace Systems		5. Report Date March 15, 1988 6. Performing Organization Code
7. Author(s) D.J. Atkinson		8. Performing Organization Report No. JPL PUB 88-7
9. Performing Organization Name and Address JET PROPULSION LABORATORY California Institute of Technology 4800 Oak Grove Drive Pasadena, California 91109		10. Work Unit No. 11. Contract or Grant No. NAS7-918 13. Type of Report and Period Covered JPL Publication
12. Sponsoring Agency Name and Address NATIONAL AERONAUTICS AND SPACE ADMINISTRATION Washington, D.C. 20546		14. Sponsoring Agency Code RE159 BK-549-03-21-01-00

15. Supplementary Notes

16. Abstract

This document discusses autonomous diagnosis of aerospace systems. The need for automated diagnosis in aerospace and the approach of using knowledge-based systems are examined. Research issues in knowledge-based diagnosis which are important for aerospace applications are treated along with a review of recent relevant research developments in Artificial Intelligence. The design and operation of some existing knowledge-based diagnosis systems are described. The systems described and compared include the LES expert system for liquid oxygen loading at NASA Kennedy Space Center, the FAITH diagnosis system developed at the Jet Propulsion Laboratory, the PES procedural expert system developed at SRI International, the CSRL approach developed at Ohio State University, the StarPlan system developed by Ford Aerospace, the IDM integrated diagnostic model, and the DRAPhys diagnostic system developed at NASA Langley Research Center.

17. Key Words (Selected by Author(s))	18. Distribution Statement
Artificial Intelligence Knowledge-based systems Fault diagnosis Automation Computer Systems	Unlimited distribution Unclassified

19. Security Classif. (of this report)	20. Security Classif. (of this page)	21. No. of Pages	22. Price
Unclassified	Unclassified		

HOW TO FILL OUT THE TECHNICAL REPORT STANDARD TITLE PAGE

Make items 1, 4, 5, 9, 12, and 13 agree with the corresponding information on the report cover. Use all capital letters for title (item 4). Leave items 2, 6, and 14 blank. Complete the remaining items as follows:

3. Recipient's Catalog No. Reserved for use by report recipients.

7. Author(s). Include corresponding information from the report cover. In addition, list the affiliation of an author if it differs from that of the performing organization.

8. Performing Organization Report No. Insert if performing organization wishes to assign this number.

10. Work Unit No. Use the agency-wide code (for example, 923-50-10-06-72), which uniquely identifies the work unit under which the work was authorized. Non-NASA performing organizations will leave this blank.

11. Insert the number of the contract or grant under which the report was prepared.

15. Supplementary Notes. Enter information not included elsewhere but useful, such as: Prepared in cooperation with... Translation of (or by)... Presented at conference of... To be published in...

16. Abstract. Include a brief (not to exceed 200 words) factual summary of the most significant information contained in the report. If possible, the abstract of a classified report should be unclassified. If the report contains a significant bibliography or literature survey, mention it here.

17. Key Words. Insert terms or short phrases selected by the author that identify the principal subjects covered in the report, and that are sufficiently specific and precise to be used for cataloging.

18. Distribution Statement. Enter one of the authorized statements used to denote releasability to the public or a limitation on dissemination for reasons other than security of defense information. Authorized statements are "Unclassified—Unlimited," "U. S. Government and Contractors only," "U. S. Government Agencies only," and "NASA and NASA Contractors only."

19. Security Classification (of report). NOTE: Reports carrying a security classification will require additional markings giving security and downgrading information as specified by the Security Requirements Checklist and the DoD Industrial Security Manual (DoD 5220.22-M).

20. Security Classification (of this page). NOTE: Because this page may be used in preparing announcements, bibliographies, and data banks, it should be unclassified if possible. If a classification is required, indicate separately the classification of the title and the abstract by following these items with either "(U)" for unclassified, or "(C)" or "(S)" as applicable for classified items.

21. No. of Pages. Insert the number of pages.

22. Price. Insert the price set by the Clearinghouse for Federal Scientific and Technical Information or the Government Printing Office, if known.