REPORT T-97 JULY, 1980

CSL COORDINATED SCIENCE LABORATORY

ARTIFICIAL INTELLIGENCE AND HUMAN ERROR PREVENTION: A COMPUTER AIDED DECISION MAKING APPROACH

FINAL REPORT

BY
R.T. CHIEN
W. BREW
D. CHEN
Y.C. PAN

THIS WORK WAS SUPPORTED BY
THE FEDERAL AVIATION ADMINISTRATION
UNDER CONTRACT DOT FA79-WA-4360

UNIVERSITY OF ILLINOIS - URBANA, ILLINOIS

I INTRODUCTION

The work on "artificial intelligence and human error prevention" continued at a robust pace with emphasis on both theory and applications.

The work related mostly to the demonstration system is summarized in Technical Report #3 "An Intelligent System for Monitoring and Diagnosis in Cockpit Environment" which is attached here as Appendix A.

The theoretical work that we have pursued is summarized in Sections II and III. We are confident that the extensive effort we spent in understanding the literature is going to pay very high dividends in the near future where several papers are being planned.

II THE INTELLIGENT MONITOR

The function of the intelligent onboard monitor is to continuously detect errors during flight. The monitor essentially acts as a passive crew member, scanning the instruments and sensors, integrating the weather updates, and staying ahead of the aircraft. The monitor does not actively influence the state of the aircraft, but it can notify the crew of perceived present and future difficulties.

The construction of such a monitor is a formidable task. In order to monitor the flight, the monitor must have a wide range of flight knowledge such as navigation, engineering, FAA regulations, and flying skills. The large amount and the wide diversity of the necessary knowledge are not the only issues to be met, but rather, the organization and the interaction of the different kinds of knowledge are also important issues in constructing the monitor. A well-known phenomenon in computer science is that one can not get more out of the computer than what one puts in. In addition to the techniques of knowledge base construction, the function and form of the knowledge base require careful consideration.

2.1 The Abstraction Levels Architecture

After studying the different kinds of knowledge required to fly an aircraft, we have organized the domain knowledge into four levels as illustrated in Figure 1. The levels are organized from the specific to the abstract in a hierarchical fashion.

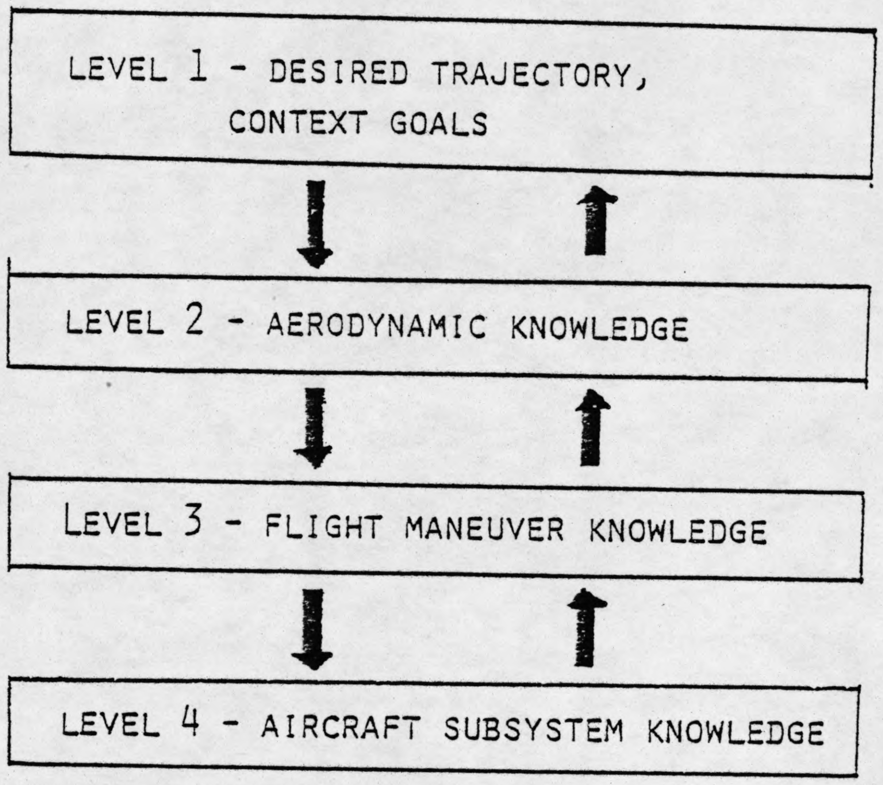

LEVEL 1 - DESIRED TRAJECTORY, CONTEXT GOALS
LEVEL 2 - AERODYNAMIC KNOWLEDGE
LEVEL 3 - FLIGHT MANEUVER KNOWLEDGE
LEVEL 4 - AIRCRAFT SUBSYSTEM KNOWLEDGE

Figure 1

Level one is the highest level and the most abstract. It describes the contextual goals of a flight phase and the desired trajectory shape. For example, the goals for the cruise phase might be to maintain an altitude of 30,000 ft and to maintain a velocity of 340 kt. Level two is the next level in the abstraction hierarchy. It contains aerodynamic knowledge, or the forces that affect the aircraft during flight. The forces are described by the force vectors such as thrust, drag, lift, and weight. The next lower level is the flight maneuver knowledge. This level contains the techniques of maneuvering the aircraft through air. An example would be that one increases thrust and raises the nose to climb. The last level contains knowledge of the aircraft subsystems. For example, methods of maintaining hydraulic pressure belong at this level.

The four levels give organizational structure and efficiency. The abstraction level hierarchy is designed to provide perspective and allow the focusing of attention. The hierarchy also gives organizational structure and efficiency. Each level contains knowledge of similar type and influences the manipulation of knowledge at other levels. Presently we are examining artificial inteligence concepts and techniques to see how they would fit into our architecture. The works examined can be classified under the headings of problem solving and knowledge representation.

2.2 Problem Solving Issues

A common approach to solving a difficult problem is to 1) divide the problem into several sub-problems, 2) solve the sub-problems separately, and 3) integrate these lower-level solutions into a coherent whole. The STRIPS planning system [1] uses Means-ends analysis to break a goal into subgoals.

STRIPS then uses resolution theorem proving to generate the plan steps to achieve the subgoals. Unfortunately, STRIPS can only handle linear problems where the subgoals do not interact, i.e. a solution exists for a permutation of the top-level goals.

ABSTRIP [2] is essentially STRIPS except the precondition literals are ranked. A higher ranked literal intuitively means a more difficult goal. The rank is lowered and the solution steps are added at each rank. When the rank is lowered to the lowest rank, the solution is complete. The central idea of ABSTRIP is that more difficult subgoals should be achieved first. However, it is not clear how to determine whether one subgoal is more difficult than another.

NOAH [3] contributed ideas for breaking goal into subgoals and integrating the subplans into a coherent whole plan. In NOAH, the expansion of a goal into subgoals is encoded in procedural code. The constructive critics detect interaction between the parallel plan steps and reorder the plan steps to avoid negative interaction. The plan steps are kept in a partial order, and total ordering is done only when necessary.

INTERPLAN [4] is a planning system that investigates the interactions between parallel plan steps. INTERPLAN does not assume linear solution as did STRIPS and ABSTRIP. If interaction can not be avoided by a permutation of top-level goals, subgoals are promoted, i.e. a subgoal is moved forward in the plan. Promotion of subgoals allows all permutations of top-level goals and their subgoals. INTERPLAN examines a more complete set of permutations between goals and subgoals than NOAH's critics.

Maintaining a consistent world model is a necessary part of problem solving. Traditional methods of ensuring data base consistency use demons that watch over the data base. TMS [5] is a truth maintenance system that records in the data base the justifications for the assertions as well as the assertions. The justifications together with dependency-directed backtracking algorithms maintain a consistent data base with increased efficiency and can give a trace of the reasoning used to arrive at a conclusion. Justifications and dependencies can also be used to model the effects of a plan. [6]

2.3 Knowledge Representation Issues

Knowledge must be represented in the computer before it can be manipulated. How the knowledge is represented should reflect how the knowledge is to be used. The characteristic quality of a certain knowledge that makes the knowledge a desirable part of the knowledge base should match naturally the characteristic quality of the representation structure.

Assertions and patterns are natural expressions of facts and data. Semantic net is a natural structure for expressing relationships between concepts and facts; a node is a concept or a fact, and the link between two nodes represent the relationship between them. While it is relatively easy to write down the relationship between concepts, one must define clearly the meaning of the concepts and relationships. [7] People are very powerful processors: missing or implied meaning are filled in, and fuzzy meaning is clarified by context and experience. Computer, on the other hand, is very literal. While people have no trouble with intrinsic and extrinsic meanings, the computer is misled easily by inexactly defined symbols in the knowledge structure.

Semantic net is also a natural representation structure for set membership and class covering relationship. The class hierarchy can classify the world into an abstraction hierarchy. [8] For example, Tom is a man who is a person who is a mammal who is an animate thing, etc. The abstraction hierarchy imposes organization and provides inheritance. A node in the tree inherits the properties of the nodes above it. Inheritance increases representational efficiency since only one copy of the property is kept for all the members of a class.

CSA [9] is another representational system that uses nodes and links. CSA is used to represent the causal relations between actions and states. A state is a description of the world and an action transforms the world in some way. A node can be either a state or an action. The link between two nodes expresses the relationship between the nodes. CSA differs from other semantic net in that the nodes are restricted to only two types of entities: action and state. The set of links is prespecified and expresses the semantic relationships that can occur between actions and states.

As higher performance is expected from knowldge-based systems, knowledge representation structure will develop to meet this need. But before the domain knowledge is packaged in a representational structure, one must precisely define the knowledge needed, the characteristics of the knowledge, and how the different kinds of knowledge are expected to interact. The intrinsic and extrinsic properties of a concept should be delineated and separated. The classification and organization of domain knowledge are important issues of knowledge representation. The power of knowledge-based system grows with the development of representational structures that match naturally the domain knowledge that is precisely defined and delineated.

III DIAGNOSIS OF THE AIRCRAFT MECHANICAL SYSTEMS

In the last report, we described the development of the Diagnosis System (DGS) as a preliminary step toward applying the Artifical Intelligence techniques in solving the aircraft diagnosis problem. This report deals with some new concepts we introduced into the more advanced system underdeveloping thereafter.

3.1 Multi-level modeling for the mechanical subsystems.

In the previous DGS system, the functional relationship among the aircraft subsystems is modeled as a single level network. i.e. all the functional components under consideration are treated equally at the same level - there is no "grouping" concept. The problem with such structure is that many of the useful diagnostic heuristics are naturally associated with a group of components working collectively as a single functional unit. The so called "subsystems" are the high level conceoptual groupings in such functional hierarchy. The following examples are to show the importance of grouping concept: (1) A particular failure mode, called "OVERLOAD", can only be associated with the electrical system as a whole, taking all the generators and the electric loads into account. It occurs when the sum of loadings exceeds the limits of generator capacities. Single-level modeling fails to express such relationship. (2) The "fuel conservation" law is very useful in detecting leakage within the fuel system. Again, the usefulness of the conservation law is manifest only if it is associated with a section of components along the fuel path.

The above discussion leads to the conclusion that there is a need for multi-level modeling, with the "expert" diagnostic knowledge attached to each functional block at every level. The knowledge attched to the functional block at the higher level serves to provide the "global view" for its subfunctions. This approach of utilizing the functional hierarchy as the backbone of the knowledge base systematically solves the traditional problems encountered by other AI researchers [10,11] who faced the difficulties of using only the local knowledge to proceed with their problem-solving, and tried to patch up by providing some demon-like global critics.

As in Figure 2, the aircraft subsystems (only partially shown in the figure) is modeled as FUEL system and POWER PLANT at the top level. The expert diagnostic knowledge attached to this level provides the clue for further investigation on its subsystems. As in this example, the typical expert knowledge looks like the followings: If there is a droping in the fuel pressure measured at the interface between fuel system and power plant, focus next on the FUEL system. If the fuel flowrate measured at the same point drops, it could be either one's fault. The FUEL system is in turn detailed by three functional blocks: the SOURCE, the TRANSFER, and the METERING. Again, a trunk of domain-specific diagnostic knowledge describing the ways to associate subfunctional symptoms with possible failures is attached to each subfunctional block.

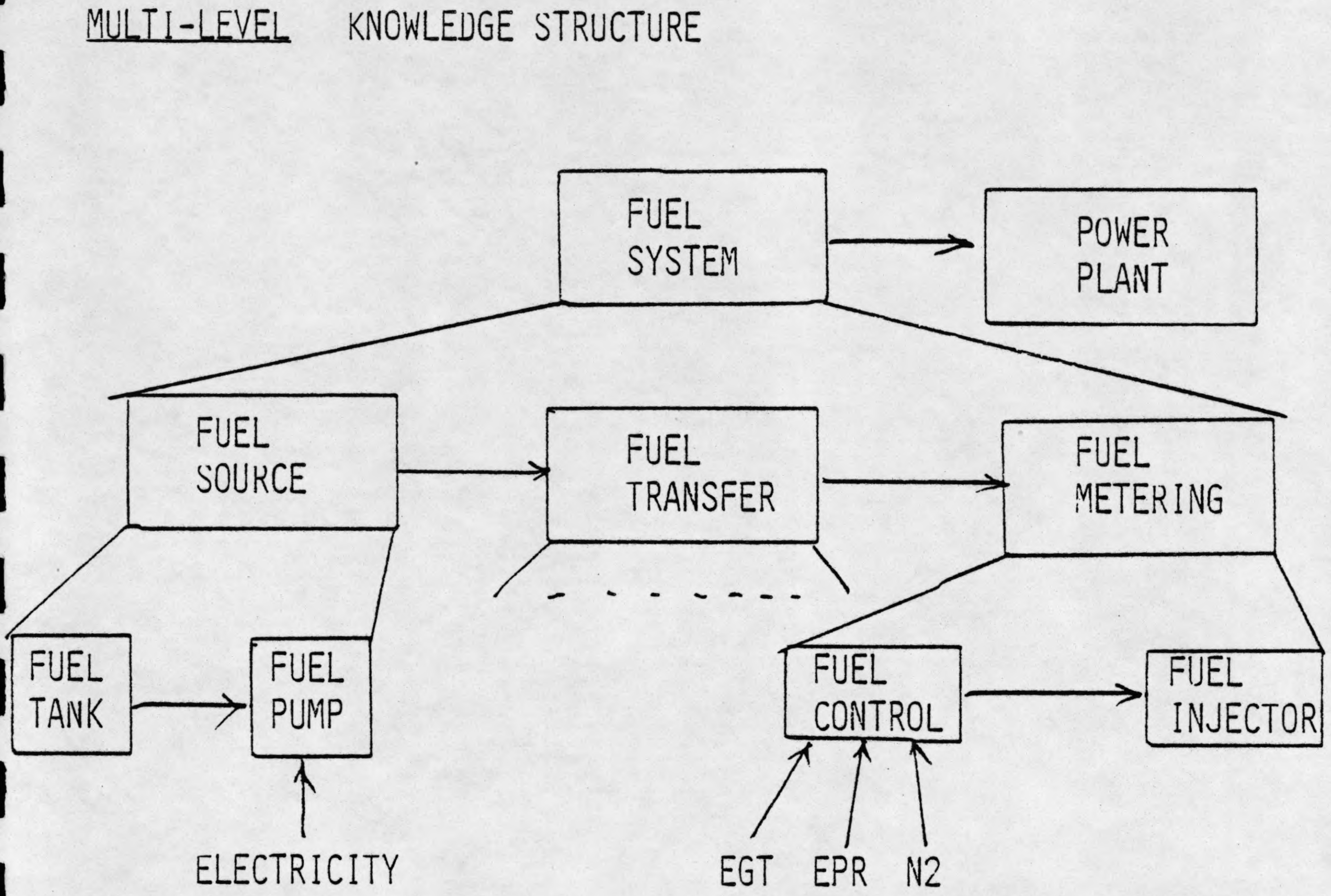

MULTI-LEVEL KNOWLEDGE STRUCTURE
FUEL SYSTEM
POWER PLANT
FUEL SOURCE
FUEL TRANSFER
FUEL METERING
FUEL TANK
FUEL PUMP
FUEL CONTROL
FUEL INJECTOR
ELECTRICITY
EGT
EPR
N2

Figure 2

3.2 Heuristical problem-solving for the subsystem diagnosis.

The traditional problem-solving has been dealing with the finding of solution that is logically supported by the known facts encoded in the knowledge base. However, in the case of expert problem-solving, reasoning based on the precise logic is sometimes impractical for the following two reasons: (1) Most of the expert knowledge are heuristcal in nature (rather than definite "laws"). The expert heuristcs are summerized from statistics, or gathered from past experience. They are applicable to general cases, but failed possibly on some special cases. As a result, problem-solving based on "heuristcs" requires a rather different control structure. Some existent techniques have been developed for other expert problem-solving systems, such as MYCIN [12] and PROSPECTOR [13]. These techniques are called "inexact reasoning" [14] or "plausible inference" [15] in the literatures.

With the idea of multi-level knowledge base in mind, our recent efforts have been to develop an problem-solving algorithm, based on the concept of heuristical reasoning, to proceed with the subsystem diagnosis. The key to the process is to derive the heuristics from the expert knowledge encoded in the functional model. At each functional block, the problem-solver selects those rules applicable to the current context, and evaluates the relative failure possiblities for each of its subfunctions. Subsequently, the problem-solver focuses on the subfunctional block that is probabistically most likely to fail. The result is a list of possible failures, prioritized by their relative possibilities.

REFERENCES

1. Fikes, R.E. and Nilsson, N.J., "STRIPS: A New Approach to the Application of Theorem Proving to Problem Solving," Artificial Intelligence, Vol. 2, No. 3-4, pp. 189-208 (Winter 1971).

2. Sacerdoti, E.D., "Planning in a Hierarchy of Abstraction Spaces," Artificial Intelligence, Vol. 5, No. 2, pp. 115-135 (Summer 1974).

3. Sacerdoti, E.d., A Structure for Plans and Behavior, (Elsevier North-Holland, New York, 1977).

4. Tate, A., "Generating Project Networks," Proc. Fifth International Joint Conference on Artificial Intelligence, pp. 888-893, Cambridge, Massachusetts (August 1977).

5. Doyle, J., Truth Maintenance Systems, Ms. Thesis, MIT, Artificial Intelligence Lab., Report AI-TR-419, Cambridge, MA. (January 1978).

6. London,P.E., Dependency Networks as a Representation for Modelling in General Problem Solvers, Ph.D. Thesis, Department of Computer Science, University of Maryland, College Park, Maryland, (September 1978).

7. Brachman,R.J., "On the Epistemological Status of Semantic Networks," pp. 3-50, in Associative Networks, Ed. Nicholas V. Findler, Academic Press, New York, New York, 1979.

8. Roberts, B. and Goldstein, I., "The FRL Manual", MIT AI Memo 409, 1977.

9. Rieger, C., "The Commonsense Algorithm as a Basis for Computer Models of Human Memory, Inference, Belief, and Contextual Language Understanding," Proc. of Workshop on Theoretical Issues of Natural Language Processing. Cambridge, Mass. 1975.

10. de Kleer, J. "Local Methods for Localizing Faults in Electronic Circuits", Memo No. 394, Artificial Intelligence Laboratory, M.I.T. (Nov. 1976)

11. Stallman, R.M. and Sussman, G.J. "Forward Reasoning and Dependency-Directed Backtracking in a System for Computer-Aided Circuit Analysis", Artifiicial Intelligence, Vol. 9, No. 2, pp. 135-196. (1977)

12. Shortliffe, E.H. "Computer-Based Medical Consultations: MYCIN", Elsevier, N.Y. (!976)

13. Duda, R.O. et al. "Development of the PROSPECTOR Consultation System for Mineral Exploration", Final Report for the period Oct. 1, 1976 to Sept. 30, 1978, SRI International, AI Center, Menlo Park, A. (1978)

14. Shortliffe, E.H. and Buchanan, B.G. "A Model of Inexact Reasoning in Medicine", Math. Biosci., Vol. 23, pp. 351-379 (1975)

15. Friedman, L. "Plausible Inference: A Multi-Valued Logic for Problem Solving", JPL Publication, (March, 1979)

A P P E N D I X   A

ARTIFICIAL INTELLIGENCE
AND HUMAN ERROR PREVENTION:
A COMPUTER AIDED
DECISION MAKING APPROACH

TECHNICAL REPORT #3

AN INTELLIGENT SYSTEM FOR MONITORING AND DIAGNOSIS IN COCKPIT ENVIRONMENT

by

R. T. Chien
W. Brew
D. Chen
Y. C. Pan

TABLE OF CONTENTS

1 INTRODUCTION

A research project to investigate and demonstrate the feasibility of an onboard intelligent computer system for a commercial aircraft is currently under progress at the Coordinated Science Laboratory at the University of Illinois. The projected setting of the research is in the long term future when powerful and significant amounts of computation power will be available inside the cockpit. The project aims at mapping out the architecture of a software computer system that interacts with and assists the flight crew at a high-level. The feasibility of such an architecture will also be demonstrated.

The project was started in late 1977 by a grant from the Department of Transportation to study the feasibility of an advanced onboard computer system to enhance the safety of commercial airline flights. Since computation power is expected to increase while the physical size of the computer is expected to decrease, it is timely to study the conceptual design of an onboard computer system that can assist the crew in a high-level fashion in anticipation of technological progress. The computation power available is not considered a limitation in this project, however the high-level assistance to the crew and the ease of interaction with the crew are considered important. Since the computer system concerned provides Safety Enhancement by CompUter REasoning it has been named the SECURE system.

1.1 Preliminary Investigation

The initial effort of the project was devoted to identifying the best means for effectively using an airborne computer system to enhance flight safety and to familiarization with the commercial aviation environment. Interviews with pilots were conducted to learn what they want, like or dislike, and need in the way of automated systems and computer assistance. A National Guard pilot and a flight engineer with American Airlines were initially interviewed on campus.[1] A third interview was conducted at the Flight Operations Center of United Airlines at O'Hare International Airport.[2] The whole day visit was spent in discussion with six senior captains, touring the dispatch area, and viewing the cockpit of a DC-10. The flight personnel responded positively to assistance by an advanced computer system. It was generally agreed that the computer system should aid the crew with the monitoring tasks, it should verify the correctness of instrument readings, it should propose failure recovery procedures, and it should inform the pilot of difficulties at appropriate times.

In addition to the interviews, flight operation manuals for the DC-10 also provided the necessary background information. United Airlines furnished The DC-10 Flight Manual Training and Reference and The DC-10 Flight Handbook manuals. McDonnell Douglas provided The Systems Description and The Reference and Performance DC-10 Flight Crew Operating Manuals. Reference manuals for the CF-6 jet engine were also obtained from General Electric.

In order to gain first hand information about what pilots actually do under high workload and stress situations, the United Airlines Flight Training Center in Denver was visited and the flight crew proficiency checks and training sessions were observed. In these sessions flight crews in Boeing 727 and DC-8 simulators were presented with system malfunctions such as hydraulic failures, generator failure, and engine flameout, and the crew's recovery procedures were observed. In the discussion with the flight crew afterward it was pointed out that the crew's ability to respond to warning signals in the cockpit is reaching the saturation point. Due to the complexity of the visual, aural, and tactile warning signals utilized today, some confusion usually occurs and in fact, the credibility of the signal is sometimes questioned.

These activities contributed greatly to the understanding of the flight operations and flight crew activities. They also contributed to the understanding of how the onboard computer system can assist the flight crew.

1.2 Considerations for an Intelligent System

The essential attribute of an intelligent onboard computer system is its ability to provide information which can help the pilot to stay ahead of the aircraft. The better the pilot can anticipate the future state of the aircraft, the better he can make good, safe decisions concerning the operation of the aircraft. The safe operation of a modern aircraft depends primarily upon the pilot's performance, and the pilot's performance can be improved and flight safety can be enhanced by an advanced computer system that provides relevant information to the pilot on a real-time basis.[3]

The primary objective is to prevent errors that are caused by the pilot taking an improper course of action because of inadequate information, too much or misleading imformation, or late information. The aviation domain is a time critical domain. Time margin is short and the correct action must be executed on time. The lack of information can leave the crew unaware or without direction. Too much information or misleading information would force the crew to spend valuable time sifting the information. Late information is as bad as no information as documented by many actual cases. To be useful the computer system must deliver the precise, necessary information on time, hence it must be "intelligent". Also to be compatible with the pilot, the computer system should not increase the workload of the pilot. The computer system should act like an additional highly-trained crewmember whose primary task is to assist the pilot. The value of any kind of assistance is diminished if it increases the pilot's workload by requiring pilot inputs or displaying extraneous information. The computer system must not inundate the crew with too much information. Actual experience has shown that too much information is as bad as no information at all. To present the relevant information without unnecessary embellishment requires awareness of the context of the aircraft and its environment. In short, the system must be intelligent.

The intelligent computer system should aid the crew in all phases of the flight from preflight planning to inflight information processing. It should also be knowledgeable about the aircraft and its context so that it can provide relevant information fitting to the current context. To do this, an extensive knowledge base of the aircraft, its systems, and its flight environment is necessary. More details on knowledge bases and other knowledge-based systems in the field of Artificial Intelligence are provided in section two.

1.3 A Scenario

The following is a scenario of an application of the intelligent airborne computer system. The aircraft has executed the takeoff and is entering the climb phase. The throttles are reset to climb power and the airframe is cleaned up. The computer examines the sensor inputs and notices that the aircraft is somewhat sluggish for its current configuration. The possible causes are deduced to be either low thrust or high drag. It then proceeds to check out the engines and the flight control elements in detail. After detailed examination it detects that the bleed valve in engine one has failed and the flap has not retracted fully. The computer then informs the crew of the faults and recommends shutting down engine one and reducing the aircraft velocity to prevent flap damage. It then draws upon its knowledge of the flight goals, the aircraft capability, and the aircraft aerodynamic behavior and recommends a new flight profile and the corresponding control configuration to the crew.

This scenario illustrates the high-level functions of aircraft monitoring, system diagnosis, and recovery procedure planning. Much knowledge about the flight goals, the aircraft systems, the aircraft aerodynamic performance capability, and the aircraft system capabilities are necessary to realize these functions. The later sections of this report cover different aspects of a computer system that can perform these high-leveled tasks.

1.4 The Software System

Work was performed in the first year to develop models of the aircraft and its environment.[4] A software aircraft simulator was developed to provide the necessary computer environment. In order to make the notion of the intelligent computer system more concrete, work in two flight functions, the instrument verification system and the flight monitor, were also initiated. The instrument verification system determines the consistency of an output of an instrument with respect to the outputs from other instruments. The flight monitor is aware of the flight context and notifies the crew when an instrument reading is interpreted to be an error in the present context. Preliminary results were presented to Department of Transportation and Federal Aviation Administration personnel in Washington DC. A computer demonstration of the concepts developed was included in the briefing.

After the end of the first year we worked directly with the FAA, and the present research effort is aimed in three areas, aircraft modeling and simulation, knowledge-based monitoring, and system fault diagnosis. Aircraft modeling and simulation deals with the qualitative and quantitative representation of knowledge and is discussed in section three. Research in monitoring continues and is developing a powerful second generation knowledge-based monitor. This work is described in section four. Research in instrument verification has generalized to aircraft system diagnosis and is described in section five.

2 INTELLIGENT SYSTEMS

Artificial intelligence is the study of ideas which enable the computer to perform those tasks that make people seem intelligent. The goals of artificial intelligence are to make the computer more useful and to understand the principles which make intelligence possible. The computer, essential in areas such as banking and inventory control, is already a prevalent force today, however reasearchers in Artificial Intelligence are investigating methodologies of giving the computer more knowledge to make it more useful and capable of performing more complex tasks.

The field of artificial intelligence has made impressive progress in its short history. Several programs have demonstrated learning ability. One program learns new concepts from a sequence of closely related concepts.[5] Another, given primitive concepts like multiplication, factorization, and prime number, can invent mathematics that even some professional mathematicians find interesting and exciting.[6] The computer can also understand english. Given a short passage of text in a constrained domain, a program can, based on its understanding of the text, answer questions intelligently.[7] The computer has also demonstrated impressive problem solving skills. In performing symbolic integration the computer has no human peers.[8] One system of programs, DENDRAL, interprets mass spectrogram data with the skill of graduate students in organic chemistry.[9] Another system, MYCIN, helps physicians diagnose and treat certain bacterial infections at a high level of skill.[10]

DENDRAL and MYCIN are probably the most well known artificial intelligence programs. Both operate in complex domains where much technical knowledge is required. DENDRAL determines the chemical structure of a substance from its knowledge of chemical bonds and the mass spectrogram of the substance. The substance is heated until it breaks into changed fragments. The fragments are then weighed and tabulated. Using knowledge of the composition of the fragments and its knowledge of the likelihood of bonding between atomic structures, DENDRAL is able to deduce the molecular structure of the substance. It's knowledge of chemical bonds is stored in a library of production rules. A production rule is a premise-deduction pair. If the conditions in the premise section are satisfied the fact in the deduction section is deduced and added to the pool of known facts. An example of one of DENDRAL's production rules looks like the following:

```
If there is a high peak at atomic-number/charge point 71,
   there is a high peak at atomic-number/charge point 43,
   there is a high peak at atomic-number/charge point 86,
   and there is any peak at atomic-number/charge point 58,

then there must be an N-PROPYL-KETONE3 substructure.
```

Using a collection of about 100 productions of distilled bonding knowledge DENDRAL is able to perform at an expert level.

MYCIN is also an expert problem solving system except the domain is now bacterial infection. MYCIN is acquainted with particular cases by requesting information about the patient's symptoms, general condition, history, and laboratory results. At each point, the line of questioning is determined by the ongoing analysis of all previous questions. When MYCIN is satisfied with its understanding of the case, it issues its diagnosis and recommendations. The reasoning it used to arrive at its conclusions can also be examined upon request.

Like DENDRAL, MYCIN's knowledge base consists of a library of over 300 production rules. The form of the production rules is still the same, however the content is now oriented toward infectious disease. The following is a typical MYCIN production rule:

```
If the infection type is primary-bacteremia,
   the suspected entry point is the gastrointestinal tract,
   and the site of the culture is one of the sterile sites,

then there is evidence that the organism is bacteroides.
```

Unlike DENDRAL, MYCIN uses the production rules in the backward direction. For each of the plausible diagnoses, MYCIN attempts to work toward primitive facts known from laboratory results and clinical observations.

The knowledge base is an essential component of the DENDRAL and MYCIN system. In order to perform at a competent level, much knowledge about the domain must be known to the program. A production rule captures a small quantum of the necessary knowledge. The form of the production rule is also ideal for the domain of molecular structure and infectious disease. The production rule is well suited to describing a large collection of facts when the underlying organization is either not available or not obvious. In another domain where the knowledge takes on a more organized form another knowledge structure would be more suitable.

The planning system is also a major interest in artificial intelligence. The planning system generates a plan, a sequence of actions that transforms the world from the initial state to the desired goal state. NOAH is a system that plans in the blocks world.[11] It can transform the initial blocks structure into the desired blocks structure by moving one block at a time using a robot arm. NOAH uses a knowledge structure called procedural net to keep track of the plan while it is being developed and bugs are being worked out. The procedural net is a network of nodes where each node represents a particular action at some level of detail. The nodes are linked to form hierarchical descriptions of operations, and to form plans of action. The procedural net is a powerful knowledge structure that facilitates heirarchical planning.

A number of research efforts have been conducted at the Coordinated Science Laboratory to study the methodologies of planning. CADM is a computer program that helps the flight crew to make decision when the aircraft is in degraded mode operation.[12] The aircraft's internal systems are described by a general data structure called semantic net. Recovery procedures are represented by procedural nets. Through these knowledge structures CADM is able to detect system faults and recommend a recovery procedure.

Concurrent with the CADM project, Weissman studied planning in incompletely specified environments.[13] In an incompletely specified environment a system must be able to recognize when certain relevant information is missing and must be able to continue planning. A plan outline is developed and missing details are filled in when available.

Planning in an uncertain environment and planning with default plans were investigated by Davis.[14] A combination of local and global strategies can fill out the default plan when additional details are necessary. The planner can largely overcome the detriment of an inaccurate world model if it can experiment in the real world and if it has some idea of how to ignore some troublesome information.

Intelligent systems perform tasks that are considered to require intelligence. High level performance requires much knowledge. The required knowledge can be divided into two kinds. The first kind describes the domain and how to affect the domain. The second kind describes how, when, and where to apply the first kind of knowledge. These kinds of knowledge must be known to the system whether the knowledge is procedurally encoded to be executed or declaratively encoded to be examined and manipulated. The problem solving knowledge and the domain knowledge must be stored in a knowledge base accessible to the intelligent system. The knowledge base can take on the form of procedures, production rules, semantic nets, procedural nets, or other representations. A particular representation may be more suited for some domains than others. Certain kinds of knowledge are more naturally expressed in one kind of representation than another. In the case of the SECURE system, since its task of aiding the flight crew is multi-faceted, a multi-leveled knowledge base is being developed to meet the task.

The knowledge base, though required, does not make an intelligent system by itself. The knowledge base is to be used by other programs. The production rules are interpreted by the rule interpreter. The procedural net is adminstered by an executive. The semantic net is accessed by expert procedures. The knowledge base is an information source to be drawn upon by other program modules of the system.

The intelligent system is often a highly skilled expert designed to help people at some complex task. Communication between the program and people is necessary because the program receives instructions from people, the program issues results to people, or the program explains its actions to people. For the intelligent system to be of maximum utility the interface between people

3 AIRCRAFT MODELING

Part of our project has been the investigation of various kinds of aircraft models. Two aspects of this modeling work will be discussed here. First, an overview of the mock aircraft model we have developed for testing our monitoring strategies will be given. Second, considerations for constructing an aircraft model as part of a knowledge base as well as an example of what form this type of model might take will be discussed.

3.1 The Mock Aircraft Model

As stated previously, the purpose of the mock aircraft model is to provide a dynamic environment for testing the monitoring strategies we are developing. The model is used to provide the input to and to drive the rest of the monitoring system. The outputs of the model are the values of a subset of the sensors which are provided in a modern jet aircraft. Thus the mock aircraft model provides quantitative information and may be viewed as a rough simulation of certain features of an airplane which are relevent to out work.

The mock aircraft model roughly resembles a three engine commercial jet although some redundancy has been eliminated from some of the systems. The activities of the model fall into two major areas: maintaining the airplane's position, speed and heading via a simple aerodynamic and navigation model and modeling the actions of the major subsystems of the aircraft. Figure 3-1 shows the layout of the major subsystems of the mock airplane. The systems currently included in the model and the variables which are simulated are summarized below.

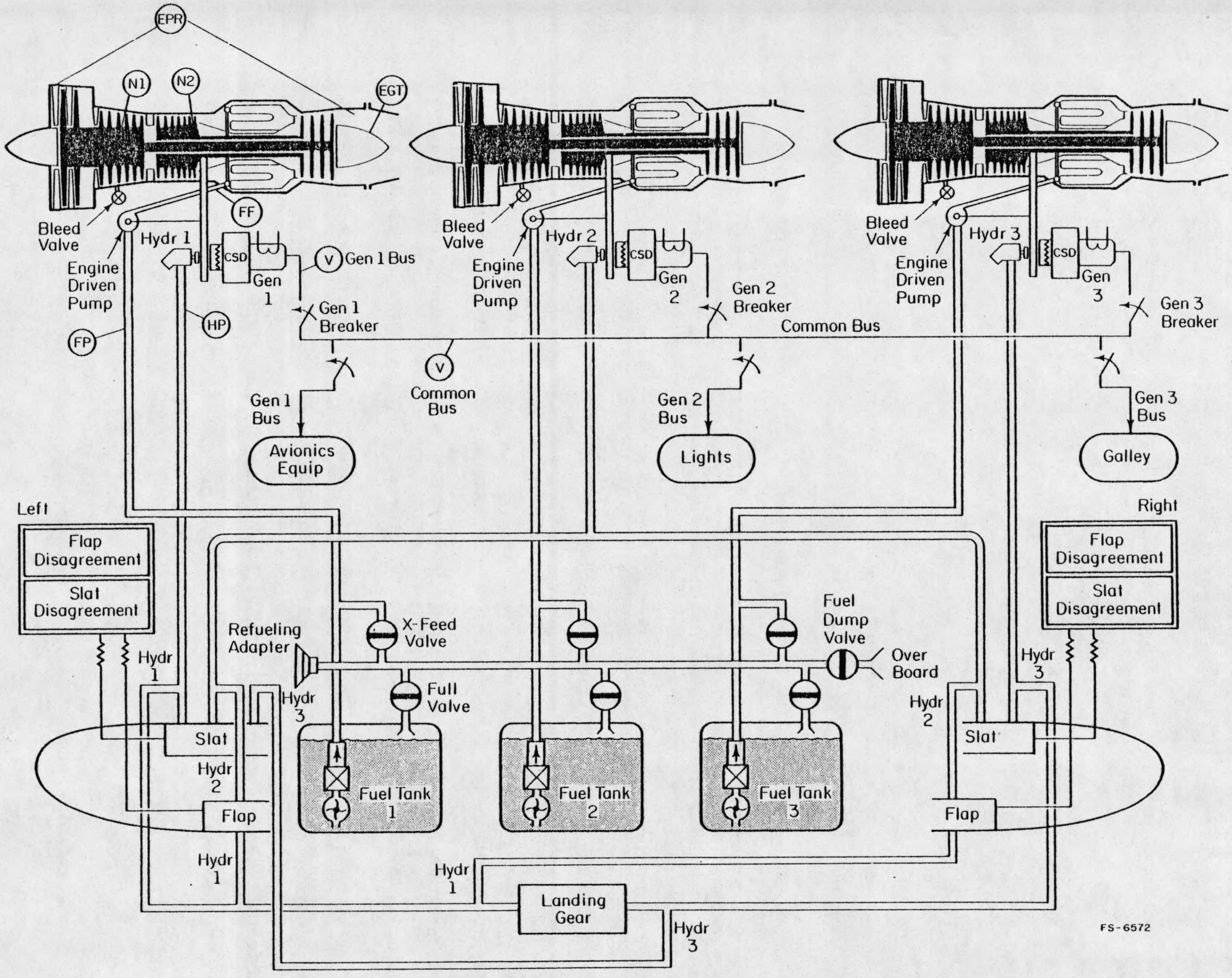

EPR
N1
N2
EGT
FF
Bleed Valve
Engine Driven Pump
Hydr 1
CSD
Gen 1
V
Gen 1 Bus
Gen 1 Breaker
HP
FP
Hydr 2
Gen 2
Gen 2 Breaker
Hydr 3
Gen 3
Gen 3 Breaker
Common Bus
V
Common Bus
Gen 1 Bus
Avionics Equip
Gen 2 Bus
Lights
Gen 3 Bus
Galley
Left
Flap Disagreement
Slat Disagreement
Right
Flap Disagreement
Slat Disagreement
Refueling Adapter
X-Feed Valve
Full Valve
Fuel Dump Valve
Over Board
Hydr 1
Hydr 3
Hydr 2
Hydr 3
Slat
Hydr 2
Flap
Hydr 1
Fuel Tank 1
Fuel Tank 2
Fuel Tank 3
Slat
Flap
Hydr 1
Landing Gear
Hydr 3
FS-6572

FIGURE 3-1

1. Flight
   elevator, ailerons, rudder, throttles, flaps, slats, speed brake, landing gear, TAS, IAS, rate of climb, pitch, bank

2. Navigation
   DME, VOR, ADF, inertial navigation, magnetic compass, barometric altimiter, radio altimeter

3. Engines
   throttles, starters, EPR, EGT, N1, N2, fuel flow, bleed valve

4. Hydraulic
   flaps, slats, landing gear, hydraulic pressure, hydraulic fluid quantity

5. Electrical
   generator bus ties, avionics breaker, lights breaker, galley breaker, generator voltage, generator frequency, generator load, bus voltage

6. Fuel
   throttles, boost bump, x-feed valves, fuel valves, fuel quantity, fuel flow, fuel pressure, fuel filter pressure

One of the interests of our project is dealing with aircraft which are malfunctioning. To this end, the mock aircraft model allows faults to be introduced into the aircraft. These faults are of two types. Simple faults usually involve breaking a sensor in the aircraft so that the flight crew receives erroneous information. A complex fault generally involves the failure of some component of the aircraft. This failure then propagates to other parts of the aircraft. For example, consider the failure of the boost pump on engine #1. This leads to loss of fuel to engine #1, loss of engine #1, loss of generator #1, loss of hydraulic system #1 (and possibly loss of some controls), and increased load on generators #2 and #3. The faults which may be introduced into the mock aircraft model are summarized below.

1. Flight
   controls, sensors, engines

2. Navigation
   controls, sensors, INS, DME, VOR, ADF

3. Engines
   controls, sensors, flameout, bleed valve stuck, fuel starvation

4. Hydraulic
   controls, sensors, engine drive, hydraulic pump, hydraulic line, hydraulic resevoir, hydraulic cylinders

5. Electrical
   controls, sensors, engine drive, generator, avionics, lights, galley

6. Fuel
   controls, sensors, engine drive, electrical power, boost pump, fuel line, fuel filter, mechanical pump

3.2 Modeling as Part of the Knowledge Base

Another aspect of modeling we are working on is the developement of an aircraft model which will be part of the computer's knowledge base. This type of model differs from the mock aircraft model in that it is designed to allow the computer to reason and draw logical conclusions about the operation of the airplane. As such, this type of model emphasizes the qualitative relationships inherent in the aircraft. The model provides knowledge of the underlying reasons why the various systems of the aircraft act and interact the way they do rather than providing the results of a numerical simulation of these systems. The knowledge base model abstracts a functional description of the aircraft rather than a physical description.

It is important to keep several ideas in mind when constructing an aircraft model. The point of view taken when designing the model is important. Because we are working in the flight domain, two points of view are apparent i.e. the pilot and the flight engineer. The pilot is concerned with the general behavior of the aircraft and how the flight controls affect this behavior. The flight engineer on the other hand is more concerned with the proper functioning and trouble shooting of the various systems of the airplane. These two points of view suggest modeling at different levels of detail and that a hierarchical approach is appropriate.

The different levels of modeling are characterized by the level of detail that they express and their breadth of applicability. The top level model which is suitable for expressing activities similar to that of the pilot has the advantage of ease of understanding but has difficulty in situations where the aircraft is malfunctioning. The more detailed level of model would allow for checking and trouble shooting the airplane but encompasses too much detail to be convenient for the top level reasoning processes.

A representation for the knowledge base aircraft model is a Common Sense Algorithm (CSA).[15] A CSA is a sematic net which has been adapted for describing the cause and effect relationships inherent in physical mechanisms. The basic nodes in the net are actions applied to the mechanism and states exhibited by the mechanism. The two major link types are causality and enablement i.e. actions cause state changes and states enable actions. Figure 3-2 is a summary of the nodes and links which may be used to construct a CSA network. A model of a subsystem in the aircraft is constructed by determining the actions and states which characterize the system and then mapping this characterization onto a CSA network with the appropriate structure of links and nodes.

Figure 3-3 is a top level CSA description of an engine and fuel system. It shows that the engine may be started by the action of pushing restart provided the electrical bus is active. Once the engine is started, it will sustain itself provided there is fuel going into the engine. Figure 3-4 is an expansion of the state node S: FUEL INTO ENGINE of Figure 3-3. This CSA

NODES

A:	ACTION
T:	TENDENCY
S:	STATE
SC:	STATE CHANGE

LINKS

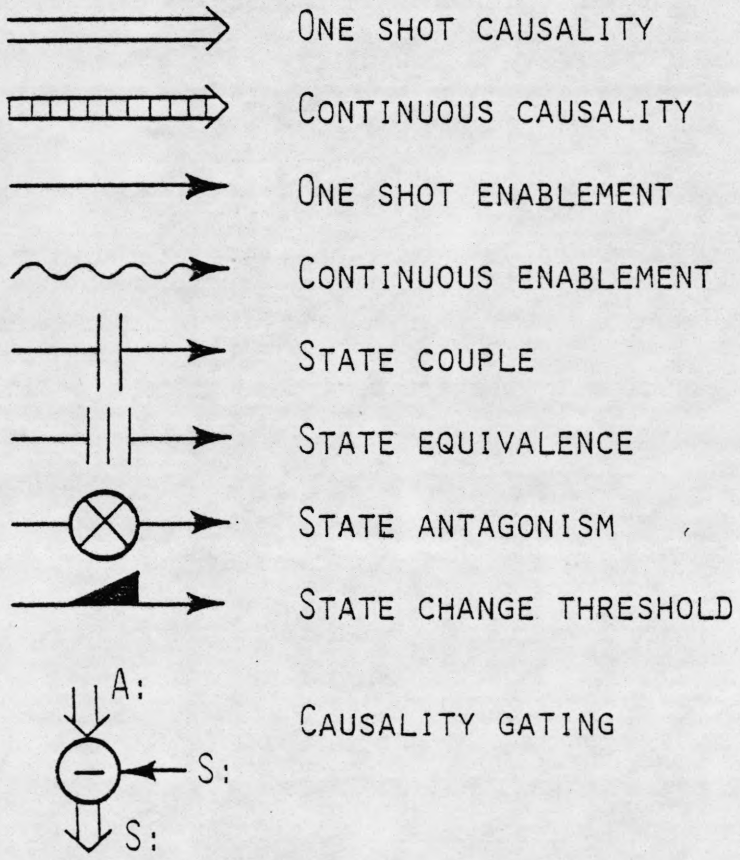
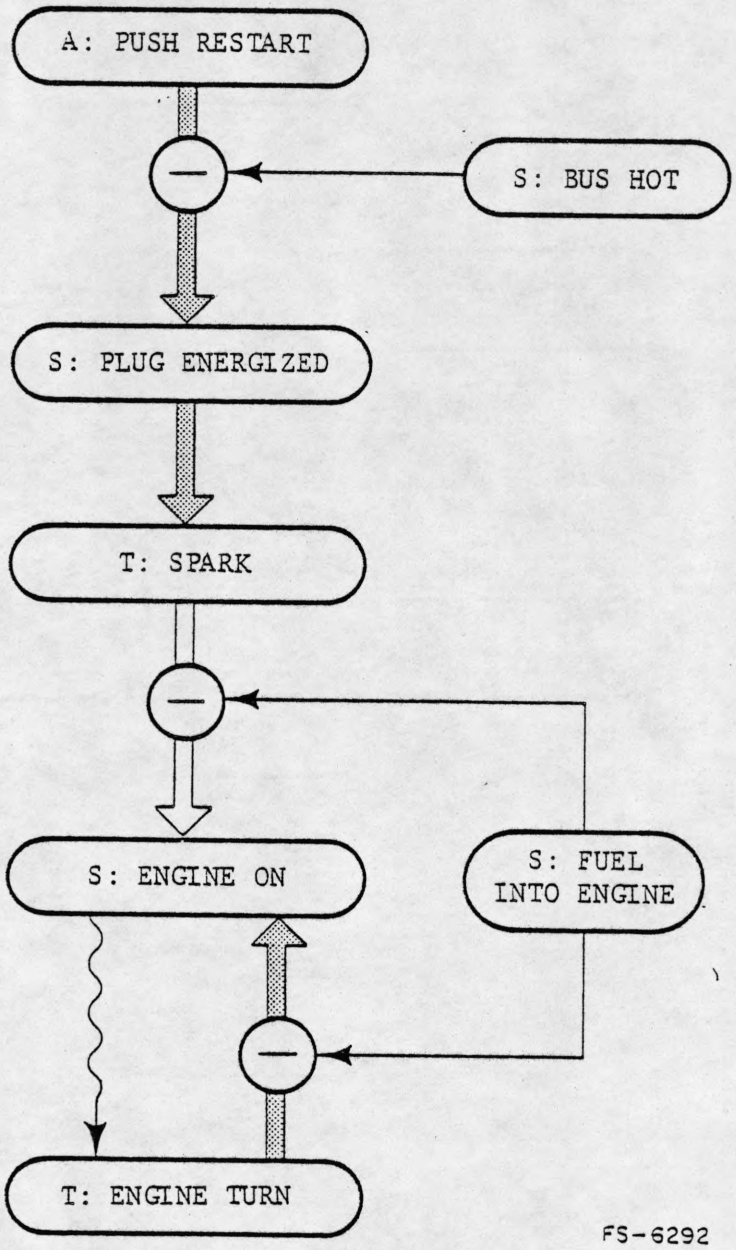
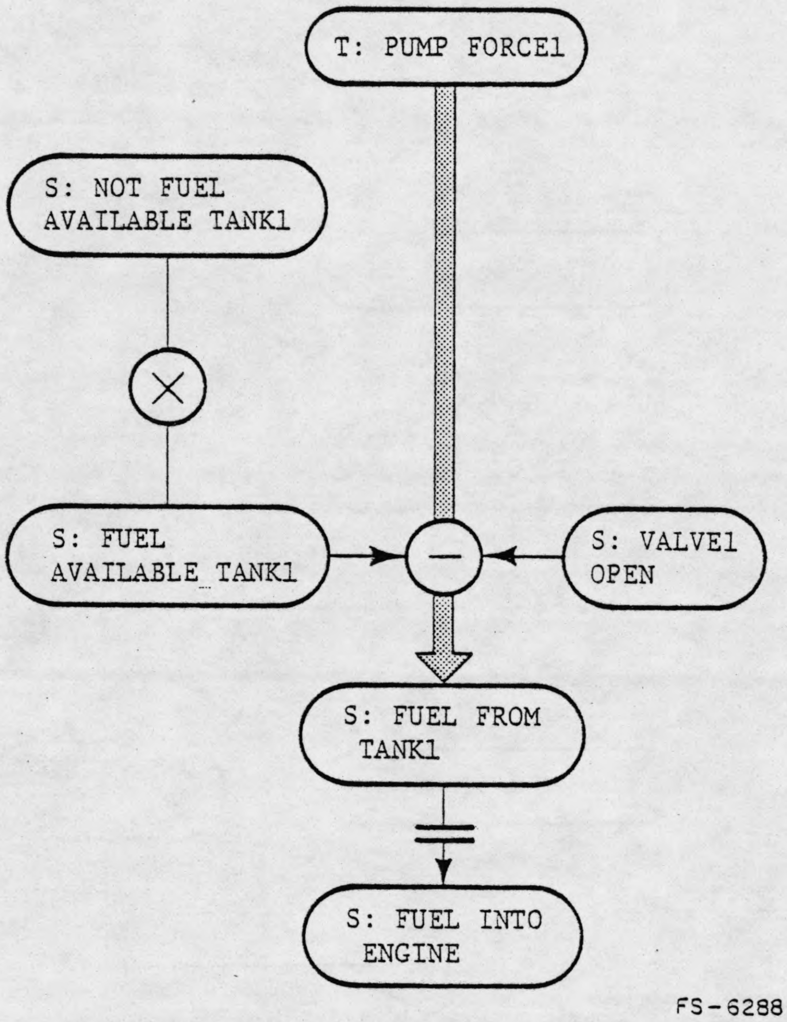


FIGURE 3-2



FS-6292

FIGURE 3-3



FS-6288

FIGURE 4

shows that in order to get fuel into the engine we must satisfy three conditions: fuel available in a tank, pump force and fuel valve open.

CSA's have sereral properties which make then attractive for the knowledge base model. CSA's are a very modular representation which makes it easy to construct and modify the knowledge base. The network structure of a CSA model makes the cause and effect relationships describing the aircraft more explicit than if they were embedded in numerical formulas. This structure facilitates the use of the model for reasoning in that the computer may more easily trace the explicit cause and effect links in order to answer questions about the aircraft. For example, suppose we ask why an engine isn't running. Tracing the CSA links of Figure 3-3 we see that this could be caused by failure to start the engine or failure to deliver fuel to the engine. These causes can be further traced via Figures 3-3 and 3-4 to possible problems in the engine ignition or fuel systems. Also it may be possible to adapt CSA's to a hierarchical structure to allow reasoning at differert levels of detail.

There are several potential uses of the knowledge base model in our monitoring system. Consequence analysis would use the model to predict the consequences of state changes in the aircraft. Trouble shooting would use the model to determine possible causes for faults which may be discovered in the aircraft. A planner will need a model with which to reason while constructing a recovery plan to get around any difficulties caused by a fault. Once a plan is constructed, the model may be used to check for unwanted side effects. This approach has the advantage over more convetional ones using preprogrammed recovery procedures in that if the model is properly constructed, situations which the programmer had not anticipated may be handled. Other sections of this report will elaborate on these uses of the aircraft model.

4 FLIGHT MONITORING

The onboard monitor is a module of the intelligent onboard computer system. The intelligent onboard computer system provides high-level assistance to the flight crew in all phases of the flight, and the monitor aids the crew in the detection of errors and faults that may occur in flight. The monitor receives inputs from the aircraft instrument sensors and advises the crew of the problems perceived and recommends corrective procedures.

4.1 Motivation

The concept of an intelligent monitor is the result of a series of interviews conducted with various flight personnel.[2] When questioned as to how can an intelligent onboard computer system aid the flight crew, the interviewed flight crews invariably included monitoring as a desirable task for the computer. The flight crew devoted a significant portion of their time to monitoring the aircraft and the flight. The crew repeatedly scan the instruments to verify that the support subsystems are functioning correctly and that the aircraft trajectory is correct. Having the computer monitor share this task of providing continuous detection of errors and faults would aid the crew in staying ahead of the aircraft.

A high-level intelligent monitor aids the crew in several ways. First, the monitor eases the crew's high workload. During takeoff and landing, the workload is especially high; many procedures must be executed on a tight schedule. The situation is aggravated further in the case of the degraded capability aircraft since operation procedure is changed due to the reduced aircraft capability. The monitor can share in the error detection task, thus enabling the crew to devote more time to manage other aspects of the flight. Second, the monitor is able to store large amount of data about the aircraft. Physical and functional information about the aircraft subsystems and components can be retrieved from the monitor's knowledge base. When a fault is detected, from its knowledge of the aircraft subsystems, the monitor can provide corrective procedures for the crew to choose from. Third, the monitor provides early error detection. The computer operates at a higher speed than people, and its performance is not effected by the repetitive scanning of the instrument sensors, thus ensuring error detection at the earliest possible moment and giving the crew more time to correct the error.

4.2 The Monitoring Function

Error detection is the primary function of the monitor. Error detection requires knowing the correct state of the aircraft. When the correct state of the aircraft is known, a comparison to the actual state of the aircraft would reveal the errors and faults that would require further attention. The difficulty of error detection lies in obtaining the correct state of the aircraft. The sophistication of monitoring is directly related to the degree of difficulty in obtaining the correct state of the aircraft, or the correct references, used in the comparison.

4.2.1 Static Constraint Monitoring

Simple monitoring involves static constraint where the constraint or reference remains fixed for long period of time and large number of situations. For example, to prevent engine overheating, the engine temperature should be below 800 degrees centigrade. To ensure proper engine lubrication, the oil pressure should be normal. These constraints are static in nature; they hold true except in unusual circumstances. The present warning systems aboard commercial aircraft today primarily utilize static constraints. Engine fire detection uses static engine temperature reference. Stall warning measures the angle of attack against the stall angle. At a higher level, the ground proximity warning system uses a combination of static constraints. The present static constraint monitoring systems have proved to be very useful, however the static constraint technique can not be applied to the more dynamic and fluid contexts as shown by the relatively frequent false alarms of the ground proximity system.

4.2.2 Dynamic Constraint Monitoring

More sophisticated monitoring involves quantities where the references are more difficult to obtain. The correct reference for quantities such as velocity, climb rate, altitude, throttle setting, flap setting is difficult to obtain because the correct value is a function of the phase of the flight, the equipment malfunctions, the control settings, and the environmental factors. The difficulty is due to the complex and dynamic nature of the flight domain.

Consider the normal takeoff profile of a three engined commercial jet aircraft as illustrated by figure 4-1. At the beginning of the takeoff roll, the flap and the slat are set at 50 percent extension, and the throttles are set at takeoff power. The aircraft rotates at velocity VR and climbs at ten knots over V2, the climb velocity. The gears are raised on positive rate of climb. At 1,000 ft, the climb rate is reduced to roughly 1,500 ft/min by lowering the pitch, and the flap and the slat are retracted at their

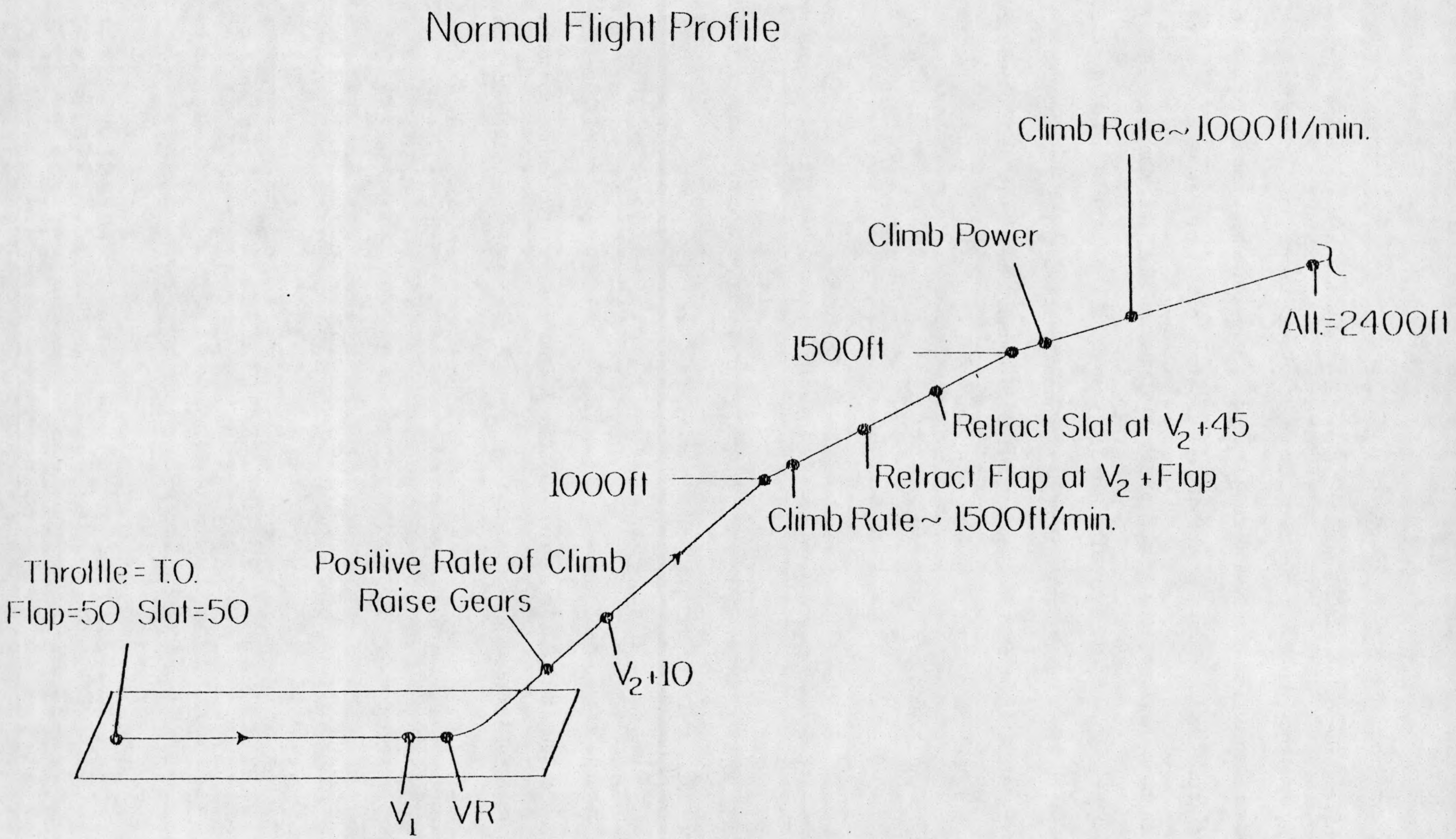

Normal Flight Profile
Climb Rate~1000ft/min.
Climb Power
Alt.=2400ft
1500ft
Retract Slat at V_2+45
1000ft
Retract Flap at V_2+Flap
Climb Rate~1500ft/min.
Positive Rate of Climb
Raise Gears
Throttle=T.O.
Flap=50 Slat=50
V_2+10
V_1
VR

FIGURE 4-1

retraction speed. At 1,500 ft, the throttles are reduced to climb power and the aircraft climbs at roughly 1,000 ft/min. Now consider the single engine failure takeoff profile shown in figure 4-2. Engine 1 failed immediately after the aircraft is airborn. Now the aircraft climbs at velocity V2 instead of V2+10 knots. Throttle 1 is pulled back to idle, and at 1,000 ft the aircraft is leveled to gain velocity and to retract the flap and the slat. When the flap and slat are retracted, the other two throttles are pulled back to climb power, and the aircraft climbs at 210 knots. As shown by the above examples, the correct throttle, flap, and slat setting varies depending on the flight phase and the aircraft integrity. The flight profile for a short duration of the flight, the takeoff phase, varies drastically depending on the aircraft capability.

A flight is a sequence of contexts, and what is correct for one context of the flight is not necessarily correct for another context. Thus to perform monitoring in a dynamic environment, the monitor must obtain the correct references dynamically from its awareness of the current environment and its knowledge base.

One scheme of dynamically obtaining the monitoring references takes advantage of the sequential nature of a flight.[4] A flight can be divided into the takeoff phase, the cruise phase, and the landing phase. These three phases can be divided into finer sub-phases. For example, the takeoff phase can be subdivided into the start engine, the taxiing, the takeoff roll, the initial climb, and the second stage climb sub-phases. This heirarchical description of the flight is shown in figure 4-3. Each of the sub-phases specifies a context, and the correct references for the context can be determined and stored in the monitor's knowledge structure.

Engine Failure After V_1 Flight Profile

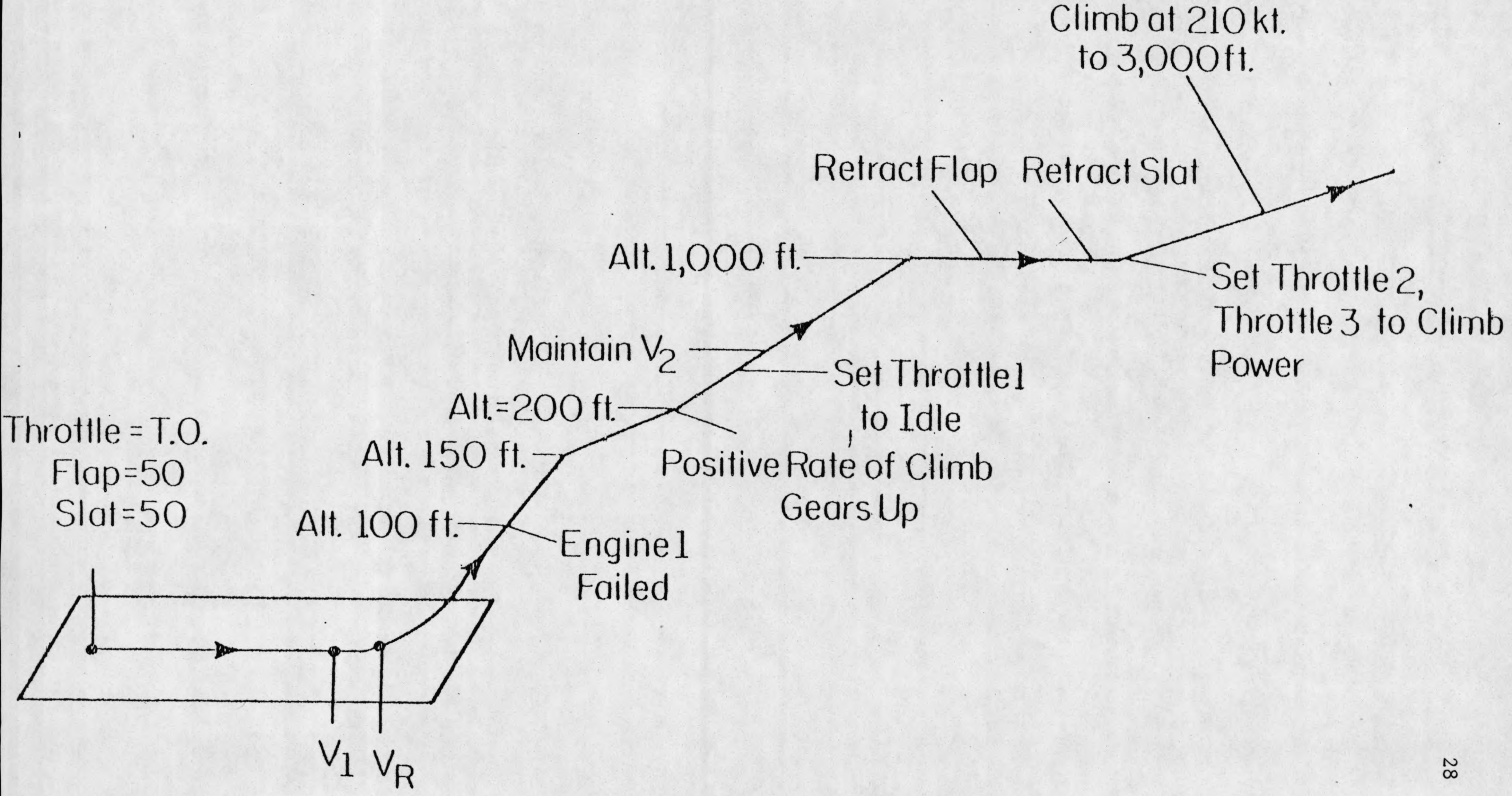


FIGURE 4-2

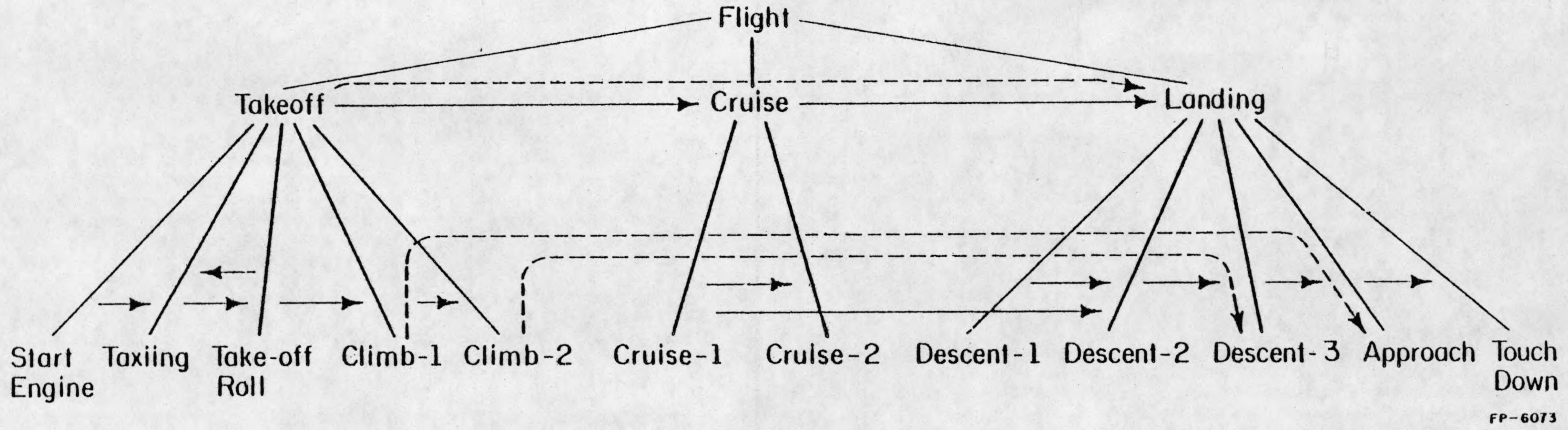

Flight
Takeoff
Cruise
Landing
Start Engine
Taxiing
Take-off Roll
Climb-1
Climb-2
Cruise-1
Cruise-2
Descent-1
Descent-2
Descent-3
Approach
Touch Down
FP-6073

FIGURE 4-3

A data structure that contains the correct references for the takeoff roll phase appears in figure 4-4. The data structure contains a header section that specifies the entry conditions or the characteristic features of the context. The header section is used to determine the applicability of the data structure. The rest of the data structure specifies the variables of concern and their references. When the monitor determines that the conditions specified in the header section are satisfied, it uses the references in that data structure to perform monitoring. A library of such data structures can cover the entire flight, thus providing the monitor with the necessary references.

Such a scheme of obtaining the monitoring references suffers from several drawbacks. The first problem is that the library can not cover all the situations that may arise during a flight. Even though the library can cover the regular phases of the flight and the more common failures such as the loss of an engine, it is not possible to anticipate all the fault and combinations of faults that can occur. The second problem is that it is possible for the monitor to apply the wrong set of references to a given situation. The header section that specifies the characteristic features of a context is limited in discrimination precision for otherwise the size of the library would be enormous. As a result of the limited discrimination precision of a context, it is possible for the monitor to apply an inappropriate set of references to certain unusual circumstances.

A flight monitor using the stored references technique has been implemented. It is used to verify our ideas about the sequence of contexts in flight and to provide the monitor module for the other modules in the intelligent onboard computer system to interact with. Due to the drawbacks

TAKEOFF-ROLL-S

ENTRY CONDITIONS:

ALTITUDE: ZERO

ENGINE THRUST: CRUISE → TAKEOFF

FUEL: MAX

AIRSPEED: ZERO → ONE HUNDRED

POSITION: TO RUNWAY

NORMAL STATE:

GEAR: DOWN

BRAKE: ZERO

AIRSPEED: ~ ONE HUNDRED

FUEL: MAX

THRUST: CRUISE → TAKEOFF

EGT: NORMAL

RPM: NORMAL

ALTITUDE: ZERO

FLAP: EXTENDED

SLAT: EXTENDED

SPEEDBRAKE: RETRACTED

CLIMB RATE: ZERO → UP-2

OTHERS: DON'T CARE

FIGURE 4-4

cited above, a more sophisticated monitor that performs correctly in a wider variety of situations is desired.

4.3 Multi-Level Knowledge Base

Since it is not feasible to anticipate all the faults that may occur during flight, the monitor must have the knowledge to handle unexpected situations as they occur. The knowledge required span all levels of flying including knowledge of the general goals of a phase of flight, the aircraft aerodynamic performance capability, the flight maneuver techniques, and the aircraft subsystems. The knowledge required are organized into a structure called partitioned abstraction levels. A level is an abstraction of the knowledge of aircraft in flight because it contains only certain relevant portion of the total flight domain knowledge, and it is separated from other levels so that the relationship between the different levels or the relationship between knowledges of the different aspects of flight can be made explicit.

The knowledge base is divided into four levels. Level one specifies the flight phase goals, level two contains aerodynamic knowledge, level three consists of the aircraft maneuver knowledge, and level four holds the aircraft subsystem knowledge. The levels are also related heirarchically as shown in figure 4-5 and are ranked by their distance from the overall goals of a flight phase.

Level one knowledge base describes the overall characteristics of a flight. A flight can be described by a sequence of flight phases or contexts as shown in fugure 4-3. The expected behavior of the aircraft in a given

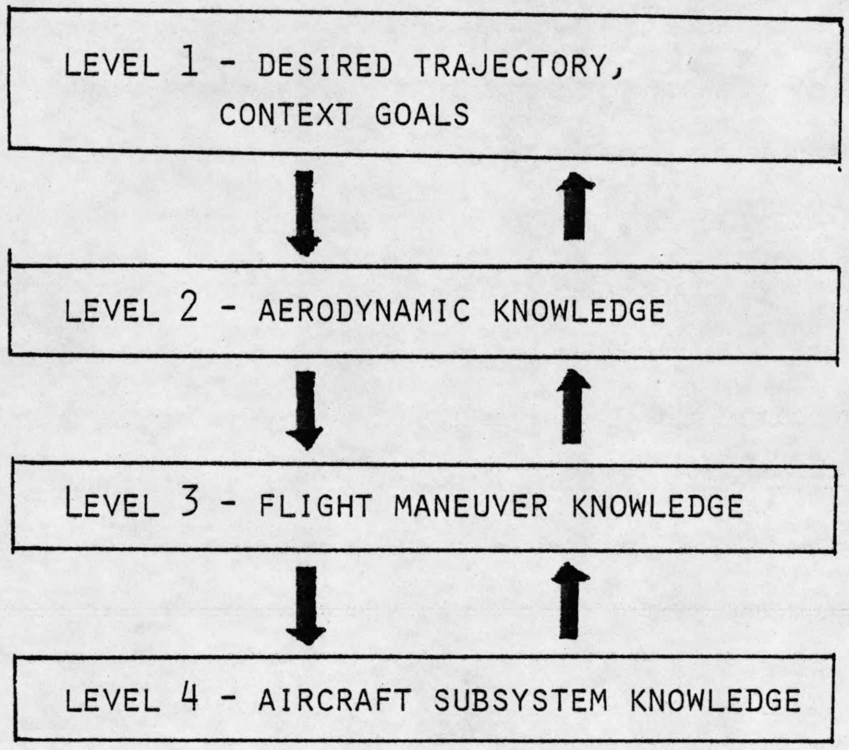

LEVEL 1 - DESIRED TRAJECTORY,
CONTEXT GOALS
LEVEL 2 - AERODYNAMIC KNOWLEDGE
LEVEL 3 - FLIGHT MANEUVER KNOWLEDGE
LEVEL 4 - AIRCRAFT SUBSYSTEM KNOWLEDGE

FIGURE 4-5

context is stored in a data structure associated with that context. Figure 4-6 shows such a data structure that is associated with the climb phase of a typical flight. The aerodynamic goals of the aircraft in the climb phase is to accelerate to a velocity of 340 kt. and to climb to an altitude of 31,000 ft. In addition to the aerodynamic goals, the data structure also contains the prioritized constraints for a context. The prioritized constraints are the ranked constraints on the aerodynamic quantities used to describe the desired behavior of the aircraft when it is in degraded operation. For example, in the climb phase, the prioritized constraints specifies that is most important to protect the engine thrust, then maintain altitude, then keep the aircraft velocity greater than two hundred knots, then minimize the gear, flap, and slat extensions, etc. The aerodynamic goals and the prioritized constraints of level one are given in aerodynamic quantities since they will be passed to the aerodynamic experts of level two to be achieved.

Level two deals with the aircraft trajectory and the forces that influence it. The aircraft moving through air is viewed as a point mass pushed by forces such as lift, thrust, weight, and drag. Figure 4-7 illustrates the force vectors that affect the aircraft in flight. The level two knowledge base consists of the force vector model of the aircraft and methods of obtaining quantities such as velocity and climb rate by manipulating the model. The model consists of force equations as shown by examples in figure 4-8. The equations are stored instead of hard coded, thus can be retrieved, examined, and manipulated. It can be seen in figure 4-8 that the lift is a function of the aircraft body lift coefficient, the angle of attack(alpha), and the velocity. It can be determined that the velocity has the greatest effect on lift, but to maintain lift at low velocity, the

CLIMB PHASE

LEVEL 1. GOALS

(ACCELERATE-TO VELOCITY 340 KT) AND

(CLIMB-TO ALTITUDE 31,000 FT)

LEVEL 1. PRIORITIZED CONSTRAINTS

1. (PROTECT ENGINE-THRUST)
2. (CLIMB-RATE $\geqq$ 0)
3. (VELOCITY $\geqq$ 200 KT)
4. (MINIMIZE AIRFRAME-DRAG)
5. (CLIMB-RATE $\geqq$ LOW)
6. (VELOCITY $\geqq$ 250 KT)
7. (CLIMB-RATE $\geqq$ 500 FT/MIN)

FIGURE 4-6

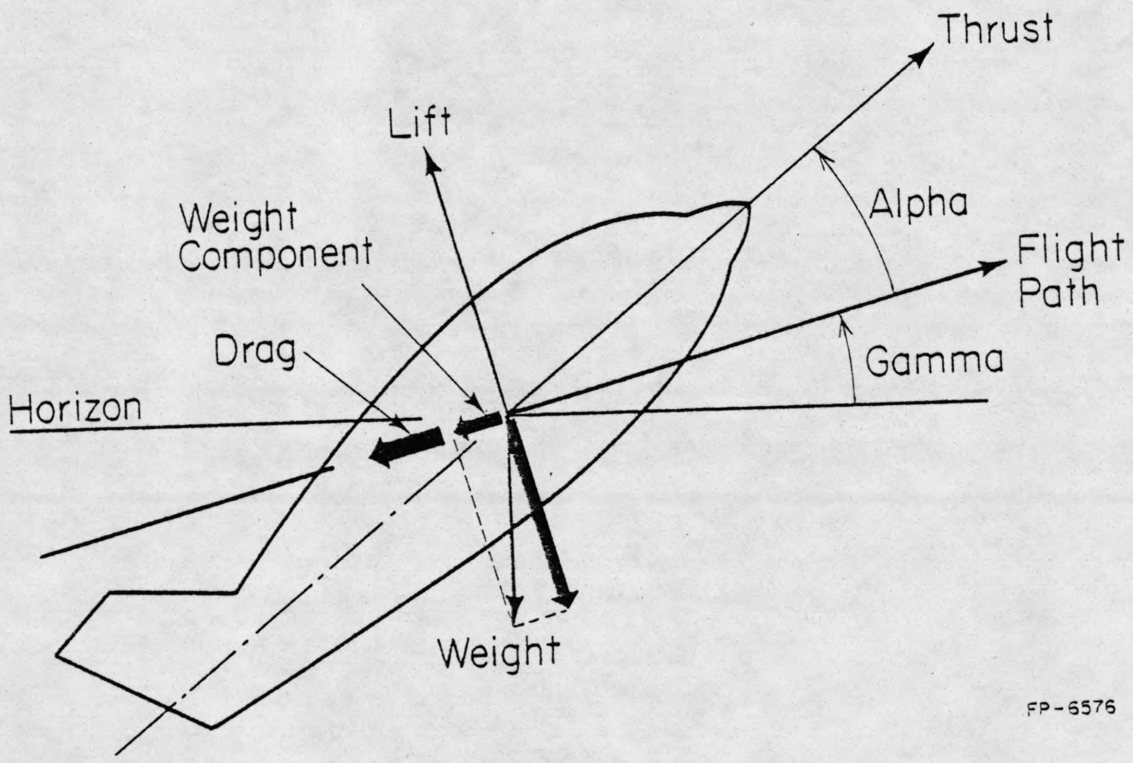

Thrust
Lift
Alpha
Weight Component
Flight Path
Drag
Gamma
Horizon
Weight
FP-6576

FIGURE 4-7

LEVEL 2. FORCE VECTOR MODEL
DESCRIBED BY EQUATIONS

THRUST = ENGINE-THRUST + GRAVITY-COMPONENT
- REVERSE-THRUST

DRAG = FLUID-DRAG + INDUCED-DRAG + BRAKE-DRAG

LIFT = LIFT-COEFFICIENT * ALPHA * $(\text{VELOCITY})^2$

FLUID-DRAG = DRAG-COEFFICIENT * ALPHA * $(\text{VELOCITY})^2$

FIGURE 4-8

angle of attack and the lift coefficient should be increased, thus translating to elevator movement and flap and slat extensions. In addition to providing qualitative knowledge about aircraft aerodynamics, the equations can also be executed in a quantitative fashion thus providing a simulator for the aircraft. The quantitative model can be used to estimate the effect of an aircraft configuration.

Level three holds the knowledge about the flight controls used to directly affect the aircraft trajectory. Level three contains the relationships between the aerodynamic quantities of level two and the physical aircraft control settings. Figure 4-9 illustrates such relationships. The intangible quantities of engine thrust is associated with the throttle setting, and the abstract lift coefficient is translated into flap setting. Level three relates the flight techniques available to the crew to the aerodynamic quantities of level two. The abstract goals and variables of level one and level two are now translated to the physical aircraft configuration. However, the controls specified by level three are those directly affecting the aircraft trajectory such as flap, slat, elevator, spoiler, landing gear, and the throttle setting.

The aircraft controls that do not directly affect the aircraft trajectory belong to level four. Level four contains knowledge about the aircraft subsystems that enable the aircraft to fly. As shown in figure 4-10, examples of level four knowledge are that engine integrity and the proper fuel flow are necessary to support engine thrust and that hydraulic pressure is necessary to enable flap movement. The aircraft subsystems dealt with in level four are the engine system, the fuel system, the electrical system, and the hydraulic system. The aircraft controls known in level four include electrical breaker, LEVEL 3 FLIGHT TECHNIQUES

TO OBTAIN ENGINE THRUST OF 40 KLB
SET THROTTLE AT T. O. POWER.

TO OBTAIN HIGHER LIFT COEFFICIENT
SET FLAPS AT 22^{o}.

TO OBTAIN MORE BRAKING FORCE
DECREASE LIFT AND
APPLY BRAKE PRESSURE.

FIGURE 4-9

Level 4 Support Subsystems

To support engine thrust $\Rightarrow$

MAINTAIN FUEL FLOW AND PROTECT AGAINST ENGINE OVERLOAD.

To support flap movement $\Rightarrow$

MAINTAIN HYDRAULIC PRESSURE.

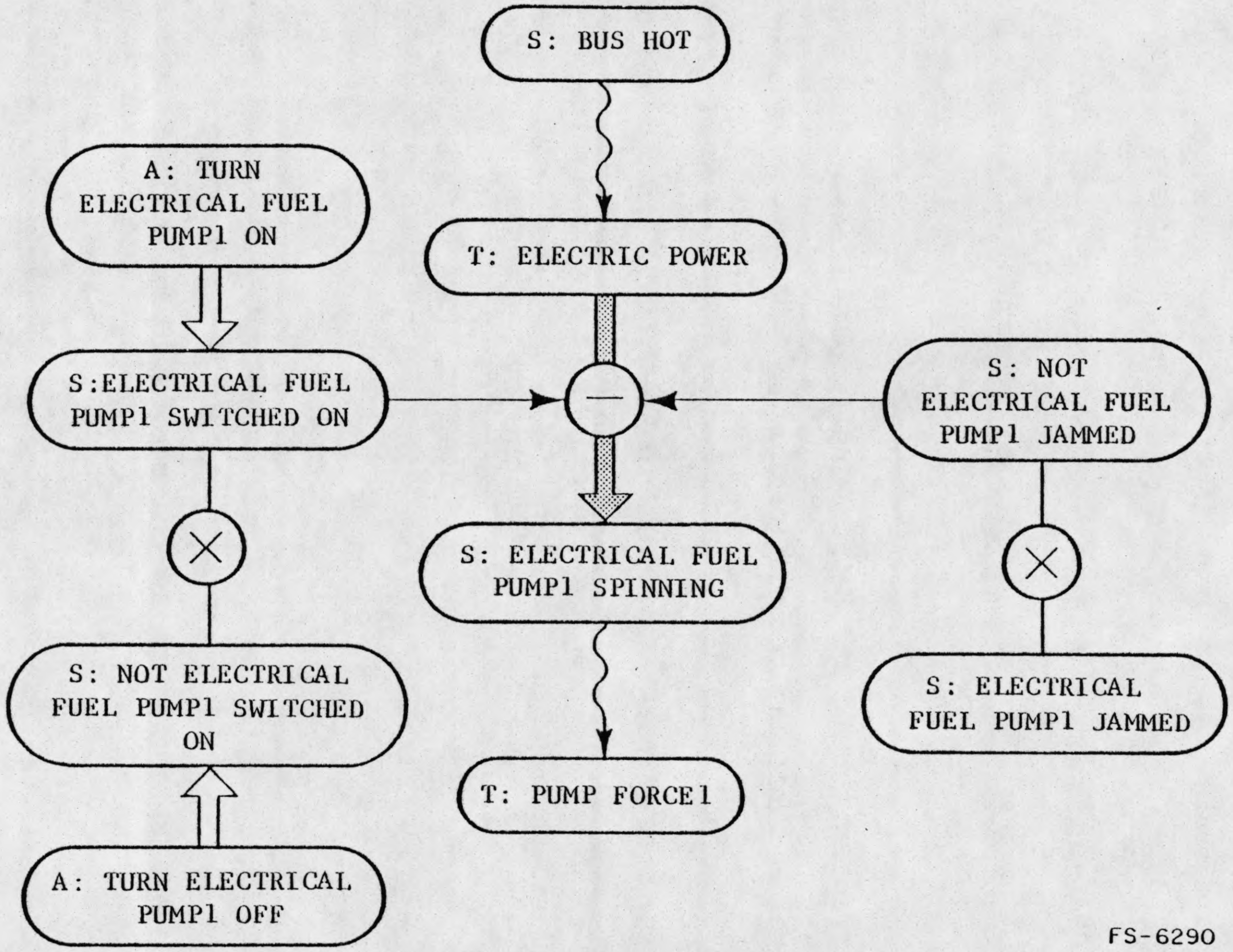
Figure 4-10

bus-tie relay, hydraulic pump, fuel pump, cross-feed valve, etc.

Level four contains both physical and functional knowledge about the subsystems. Physical knowledge specify the location of the components and the connection between the components. Functional knowledge specify the causal relationship between the states of components. For example, fuel will flow into the engine only if there is fuel in the tank, if the boost pump is pumping, if the fuel valve is open, if there is no obstruction in the fuel line, and if the fuel pump is pumping also. Functional relationships can be represented by cause-effect graph structures such as shown in figure 4-11. The cause-effect graph structure can also be organized in a heirarchical fashion.

The abstraction levels are heirarchically organized by its distance to the high-level definition of the flight and by the direct dependence relationship between levels. For example, the velocity goal of level one is achieved by the proper relationship between the thrust, lift, drag, and weight force vectors and the angle of attack of level two. The desired thrust and angle of attack are passed to level three and translated to the correct throttle setting and elevator setting. The level four experts then make sure the proper fuel flow is delivered to the engine and the hydraulic system functions correctly to enable elevator movement.

The heirarchical levels facilitate two types of monitoring: top-down and bottom-up. Top-down monitoring is the mapping from level one goals to the level four physical system states, and it can detect errors through indirect evidence and generate corrective procedure recommendations. Bottom-up monitoring is the mapping from the level four physical states to the level one goals. The bottom-up mapping is also called consequence interpretation. The


S: BUS HOT
A: TURN ELECTRICAL FUEL PUMP1 ON
T: ELECTRIC POWER
S:ELECTRICAL FUEL PUMP1 SWITCHED ON
S: NOT ELECTRICAL FUEL PUMP1 JAMMED
S: ELECTRICAL FUEL PUMP1 SPINNING
S: NOT ELECTRICAL FUEL PUMP1 SWITCHED ON
S: ELECTRICAL FUEL PUMP1 JAMMED
T: PUMP FORCE1
A: TURN ELECTRICAL PUMP1 OFF

FS-6290

FIGURE 4-11

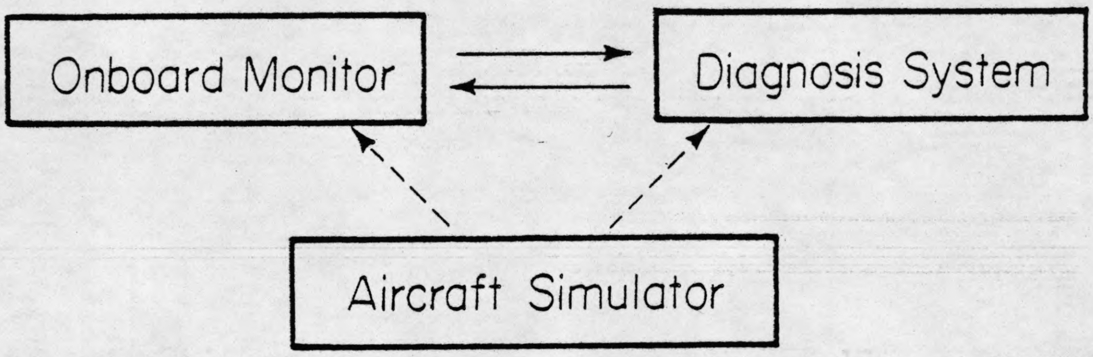
significance of an event in a level is interpreted at the next higher level. For example, the significance of an overheating engine is interpreted at level three to be the shutting down of the engine. Level two experts interpret the shutting down of the engine as a reduction of velocity and climb rate. Level one experts note the expected reduction of velocity and climb rate due to the drgradation of the engine system and refer to the prioritized constraints to determine the desired velocity and climb rate schedule. Then planning can proceed from level one to level four, and when the planning is done, the monitor can issue the recommendations to the crew.

The partitioned abstraction level is an architecture for a knowledge-based monitor. The levels are organized for conceptual clarity in the division of the knowledge about the aircraft and flying. The levels also allow the specification of the relationships between levels. The implementation of the architecture is underway, however it is not yet completed. Most of the effort spent thus far is concentrated on level one and level two. In order to test the architecture and to demonstrate the system, portions of level three and level four have also been implemented. Due to the time and manpower limitations, only a limited number of scenarios can be demonstrated, however, the completed scenarios happily confirm our confidence in the architecture.

The monitor is a part of the intelligent onboard computer system and interacts with the other modules of the system. In particular, it calls on the diagnosis system for help when necessary. The diagnosis system localizes faults in the sybsystem using diagnostic and subsystem knowledge and is discussed in more detail in section five. The monitor and the diagnostic system can call on each other and pass relevant information to each other as shown in figure 4-12. The monitor calls on the diagnosis system when it detects indirect evidences indicating that there may be a fault in the system.

The following is a walkthrough scenario that illustrates top-down monitoring, or monitoring through indirect evidence. The scenario can also be demonstrated on the computer. The environment for the scenario is provided by the aircraft simulator. The aircraft simulator provides aerodynamic and internal subsystem simulation of the aircraft. The aircraft is modeled after the popular present day three engined commercial jet aircrafts. The relevant internal subsystems are shown in figure 4-13. The monitor, the diagnosis program, and the aircraft simulator runs simultaneously on the computer under time-sharing. The simulator passes the state of the aircraft to the monitor and the diagnosis program upon request, and this is shown is figure 4-12. As discussed in section three, the experimenter can alter the aircraft control settings via a terminal keyboard thus playing the role of the flight crew. The experimenter can also set faults in the simulated aircraft, providing a realistic environment for the monitor and the diagnosis program.

The scenario is set in the climb phase of the flight. Refer to figure 4-14 for the outline of the scenario. The aircraft has just taken off the ground. The landing gear is raised and the throttles are pulled back to climb power. The pitch is lowered and the velocity increases. When the flap and slat retraction speed approach, the flap and slat are retracted. At this point the monitor notices that the aircraft velocity and climb rate are lower than expected given its aerodynamic knowledge of the aircraft and the present aircraft configuration. From its awareness of the forces acting on the aircraft, the monitor concludes that the possible causes are either low thrust or high drag. The monitor then calls the diagnosis system asking it to check



FP-6600

FIGURE 4-12

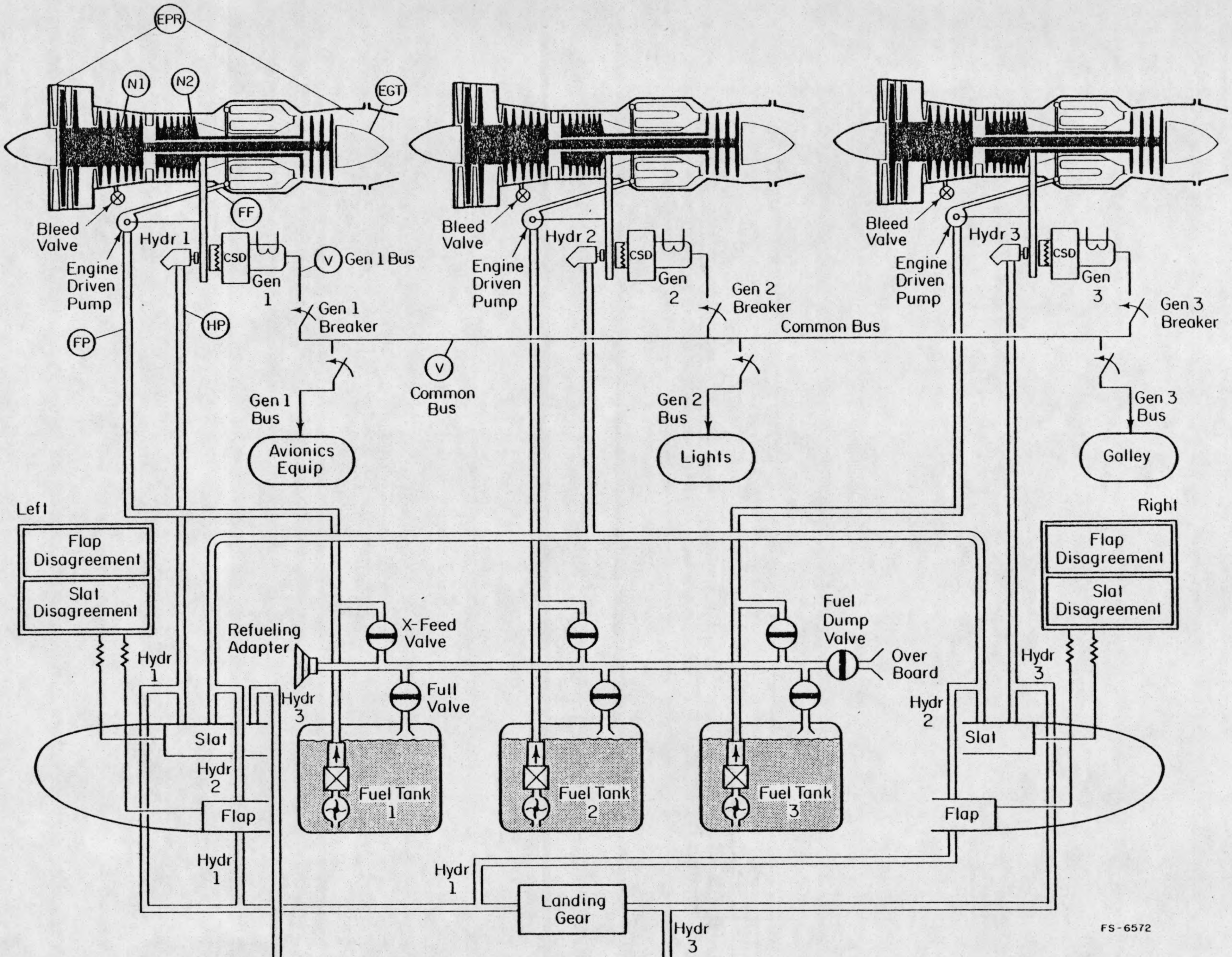

EPR
N1
N2
EGT
FF
Bleed Valve
Engine Driven Pump
Hydr 1
CSD
V
Gen 1 Bus
Gen 1
Gen 1 Breaker
HP
FP
Hydr 2
Gen 2
Gen 2 Breaker
Common Bus
Hydr 3
Gen 3
Gen 3 Breaker
V
Common Bus
Gen 1 Bus
Avionics Equip
Gen 2 Bus
Lights
Gen 3 Bus
Galley
Left
Flap Disagreement
Slat Disagreement
Right
Flap Disagreement
Slat Disagreement
X-Feed Valve
Refueling Adapter
Fuel Dump Valve
Over Board
Full Valve
Hydr 1
Hydr 3
Hydr 2
Hydr 3
Slat
Hydr 2
Flap
Hydr 1
Fuel Tank 1
Fuel Tank 2
Fuel Tank 3
Slat
Flap
Hydr 1
Landing Gear
Hydr 3
FS-6572

FIGURE 4-13

INITIAL STATE:

CLIMB PHASE

CLIMB POWER

CLEAN AIRFRAME

SYMPTOMS:

LOW VELOCITY

LOW CLIMB RATE

DIAGNOSIS:

ENGINE 1 BLEED VALVE FAILED

FLAP STUCK IN OPEN POSITION

FLAP SENSOR FAILED

CORRECTIVE ACTIONS:

THROTTLE DOWN ENGINE 1

LIMIT VELOCITY

CLIMB AT LOWER CLIMB RATE

FIGURE 4-14

out the engines, the flap, the slat, and the landing gear. The diagnosis system returns with the messge that the engine one bleed valve has failed and the flap is stuck in the open position. Once given the failures, the monitor recommends shutting down engine one since the bleed valve failure increases fuel consumption for the thrust produced and there are still two good engines left. The monitor is also aware of the velocity limitation on the aircraft when the flap is extended to avoid flap damage. The monitor then reminds the crew of the speed limitation and adjusts the climb rate goal to a lower more appropriate value. Using its aerodynamic knowledge, the monitor arrives at the thrust and angle of attack values that satisfy the velocity, climb rate, and the reduced thrust capacity constraints. The disired thrust and angle of attack values are then translated into throttle and elevator settings and communicated to the crew.

This scenario illustrates monitoring using indirect evidence. High level knowledge is used to detect the possiblity of system fault and to point the diagnosis system in the right direction. When the diagnosis system has located the faults, the constraints imposed by the faults on the flight are communicated to the crew. Planning then proceeds from the abstract to the concrete, incorporating the new constrants, and the recommended control settings are finally communicated to the crew.

5 SUBSYSTEM DIAGNOSIS AND INSTRUMENT VERIFICATION

The Diagnosis and Verification system is to have the computer monitoring aircraft from the Flight Engineer's point of view. It provides two essential functions: (1) verifying instrument readings to avoid "false alarm", and (2) locating the source of subsystem failure. With the diagnosis report, the Monitor can properly follow up the abnormal situation if failure ever occurred in the subsystem.

5.1 Review of Fault Detecting Techniques

Subsystem failures can be detected through the following ways:

1. Verification of Redundant Information:

   Airplanes are usually equipped with redundant instruments. Disagreement among the related instruments provides primary indication for a certain category of failure. For example, the engine Fuel Flow reading is supposed to be matched with the dropping rate of the tank Fuel Quantity reading. If the dropping rate is abnormally higher than the Fuel Flow, it may be because of either a gauge failure or a leaking fuel system.

   The Instrument Verification System[4], developed earlier for this project, was implemented as part of the Monitor to detect instrument failures based on the verification of the redundant sensor information.

2. Comparison with the Subsystem Performance Model:

   It is the flight engineer's duty to be familiar with the normal performance of each aircraft subsystem in response to the control parameters. In the flight manual, the engine performance chart gives the quantitative relationship among various jet engine parameters (N1, N2, EGT, EPR, FF) in response to the throttle setting at a particular environment (airspeed, air density). Normal range of the hydraulic pressure, oil pressure, fuel temperature and other subsystem parameters are also specified. Based on these same information, the Monitor can detect deviations.

3. Indirect Fault Detection through Secondary Effect:

A subsystem failure, if not caught directly by some proper sensor, can be indirectly detected by its secondary effect on other subsystems or the aerodynamic performance of the airplane. An electrical power failure in the Galley bus can be traced back to the faulty circuit breaker. A lack of climbing speed can be traced back to a faulty bleed-valve which causes engine malfunctioning.

5.2 About Subsystem Diagnosis

When fault occurs in certain subsystem, the effect usually propagates to other subsystems whose operations are depending on the output of this faulty subsystem. As a result, instead of having a dedicated indication on the exact cause(s) of failure, the pilot can be expected to look at an instrument panel with abnormal instrument indications and flashing warning lights all over. The situation is further complicated by the fact that the instruments/sensors themselves may be faulty. Facing such situation, it will be relying on the pilot's knowledge about the subsystems to properly interpret the abnormality.

What can the Computer Diagnosis system do is to provide a comprehensive report about the abnormal situation. Relying on the knowledge about the failure symptoms of each subsystem as well as the relationship among subsystem functions, the computer diagnosis system traces back the function-dependency chain from the detected secondary fault indication to pinpoint the original source of failure. Based on the same knowledge, the diagnosis system is able to verify a suspicious failure situation with its effects on other dependent subsystems and to distinguish between an instrument failure and a true subsystem failure. For example, if some of the engine parameters are reading low - an indication of a possible engine flame-out - while the related generator, hydraulic, oil systems are running normally, the diagnosis system will conclude that it is more likely an instrument failure rather than an engine failure.

5.3 Diagnosis Knowledge Base

There are two types of knowledge contained in the computer knowledge base to support the Diagnosis system. One type of knowledge describes the functional behavior of each component under abnormal situation. The other type lays out the functional relationship among various components. The representational structure of these knowledge within the computer is described below:

1. FAILURE MODES for each subsystem:

   For each of the failure situation about a particular subsystem, its associative local symptoms are stored in the knowledge base. For example [see Fig. 5-1], the symptoms associated with the "bleed-valve failure" of the engine subsystem are N1:low, N2:low, EGT:high, EPR:low, ...

   The abnormal symptoms may be caused by a local fault, or it may be inherited from other functionally related subsystems. As an example, the local failures associated with the Engine subsystem are FLAME-OUT, BLEED-VALVE FAILURE, FUEL-CONTROL FAILURE. Each is characterized by an unique set of abnormal engine parameters. Nevertheless, not all of the engine malfunctions result from the engine local problems. Fuel-starvation, which also causes engine flame out, is inherited from the Fuel subsystem problem which may in term be the consequence of a faulty Electrical subsystem.

2. FUNCTIONAL RELATIONSHIP among Dependent Subsystems:

   When a particular failure is inherited from other dependent subsystem, such "dependency" is explicitly specified in the knowledge base. For example [see Fig. 5-2], a low BOOST-PUMP output may be resorted to a "low fuel-supply" from FUEL TANK or a "electrical-power off" from the COMMON BUS.

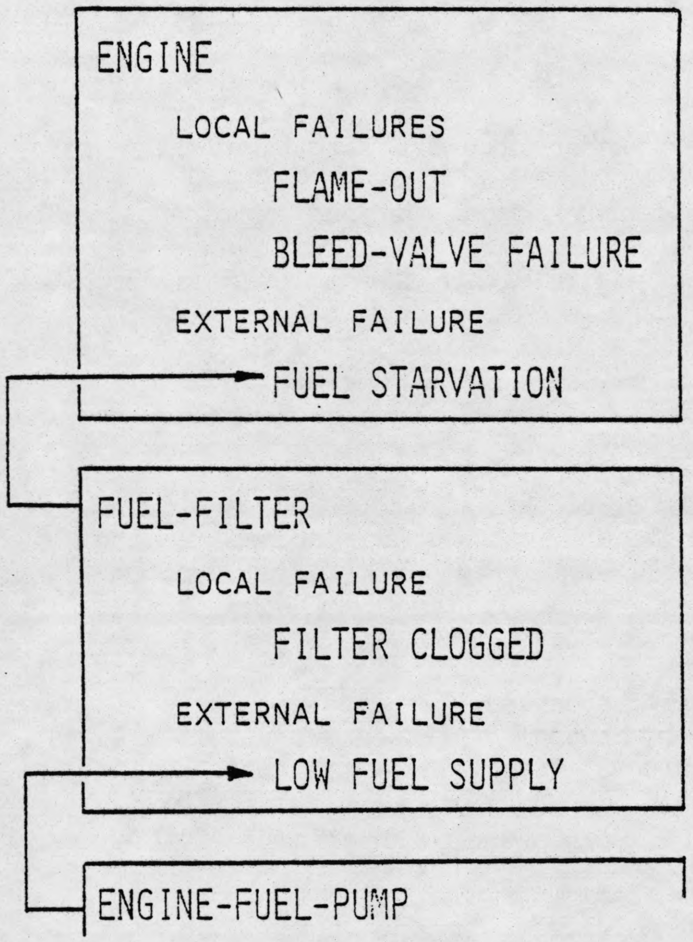


FIG. 5-1. EXAMPLE OF KNOWLEDGE BASE.

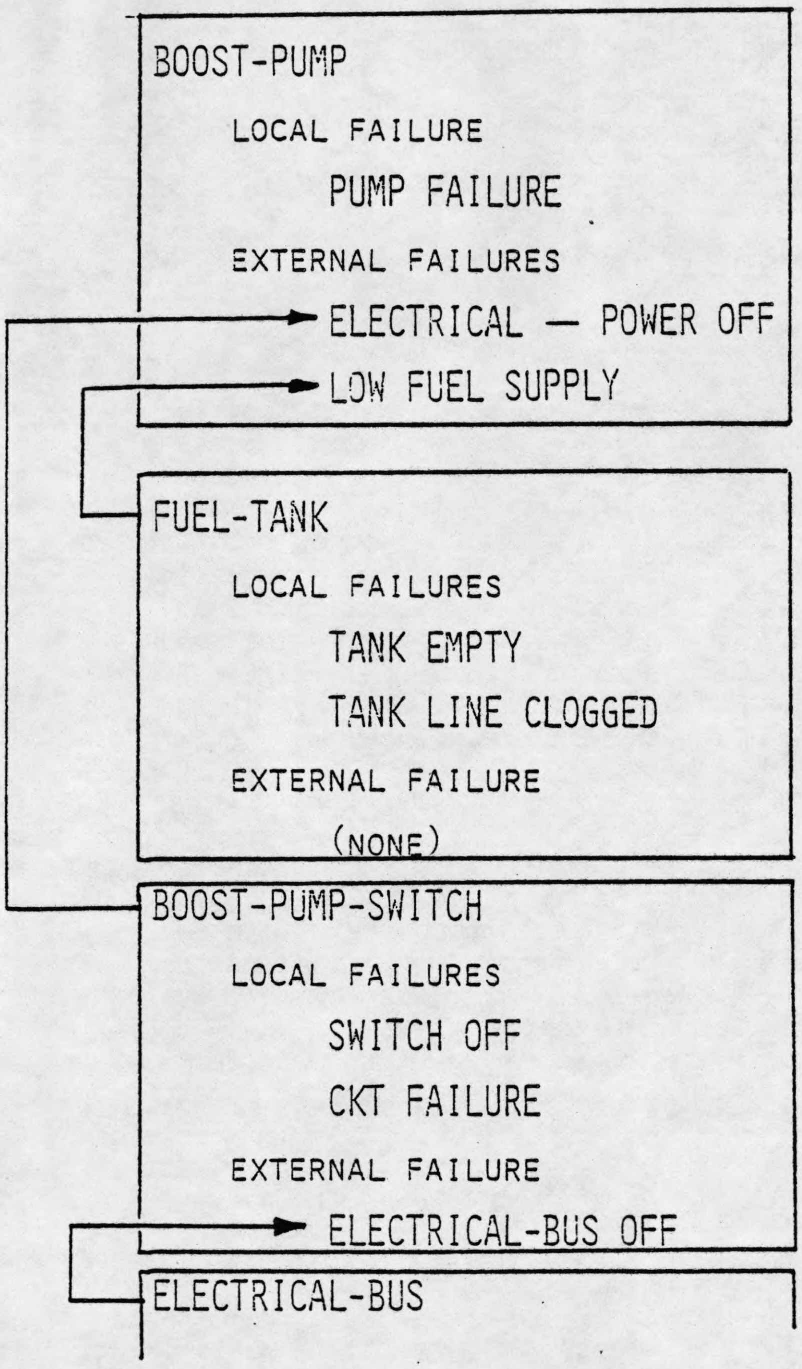


FIG. 5-2. EXAMPLE OF KNOWLEDGE BASE.

5.4 Diagnostic Inference Program

With the necessary diagnostic knowledge stored in the knowledge base, the Diagnostic Inference program is designed to make use of such knowledge to answer the question raised by the Monitor (or other potential users) about the suspicious system parameters. The diagnostic process can be described in four phases: FAULT HYPOTHESIZATION, CONSEQUENCE VERIFICATION, DEPENDENCY-DIRECTED BACKTRACKING and FAULT IDENTIFICATION [see Fig. 5-3].

1. Fault Hypothesization

   When a system parameter is questioned about its status (for example, Is the THRUST on ENGINE- 1 LOW?), the Diagnosis system looks into the knowledge base for the subsystem that is directly responsible for that parameter (in this example, the ENGINE subsystem). THis thus opens a diagnostic "window" on a particular portion of the knowledge base so that the diagnostic process can be focussed on. The description within the window lists all the possible syndromes related to the focussed subsystem. The list of syndromes (namely, FLAME-OUT, BLEED-VALVE FAILURE, FUEL-CONTROL FAILURE, FUEL-STARVATION) are the initial fault hypotheses for the diagnostic process.

2. Consequence Verification

   As a failure situation is questioned on a particular subsystem, the Diagnosis system firstly looks for possible instrument fault. Based on its knowledge about the subsystems modeled in the Diagnosis knowledge base, it verifies the corresponding symptoms of every hypothesis with its consequences. Thus, a suspicious engine flame-out will be verified with its related oil-pressure reading, hydraulic indications, and generator output parameters. An instrument failure can then be logically determined, and thus avoid issuing a "false alarm".

3. Dependency-Directed Backtracking

   On the other hand, if the failure situation is confirmed with its consequence, the Diagnosis system will proceed to locate the faulty system(s) that causes the abnormal situation. Based on the knowledge about the function-dependency among the subsystems, it traces back to all the dependent subsystems whose failures may potentially cause current abnormal situation. For example, if the Engine subsystem is questioned for lack of thrust, the Diagnosis system will initiate the Consequence Verification phase by looking into the knowledge base for those subsystems that are affected by the functions of the Engine

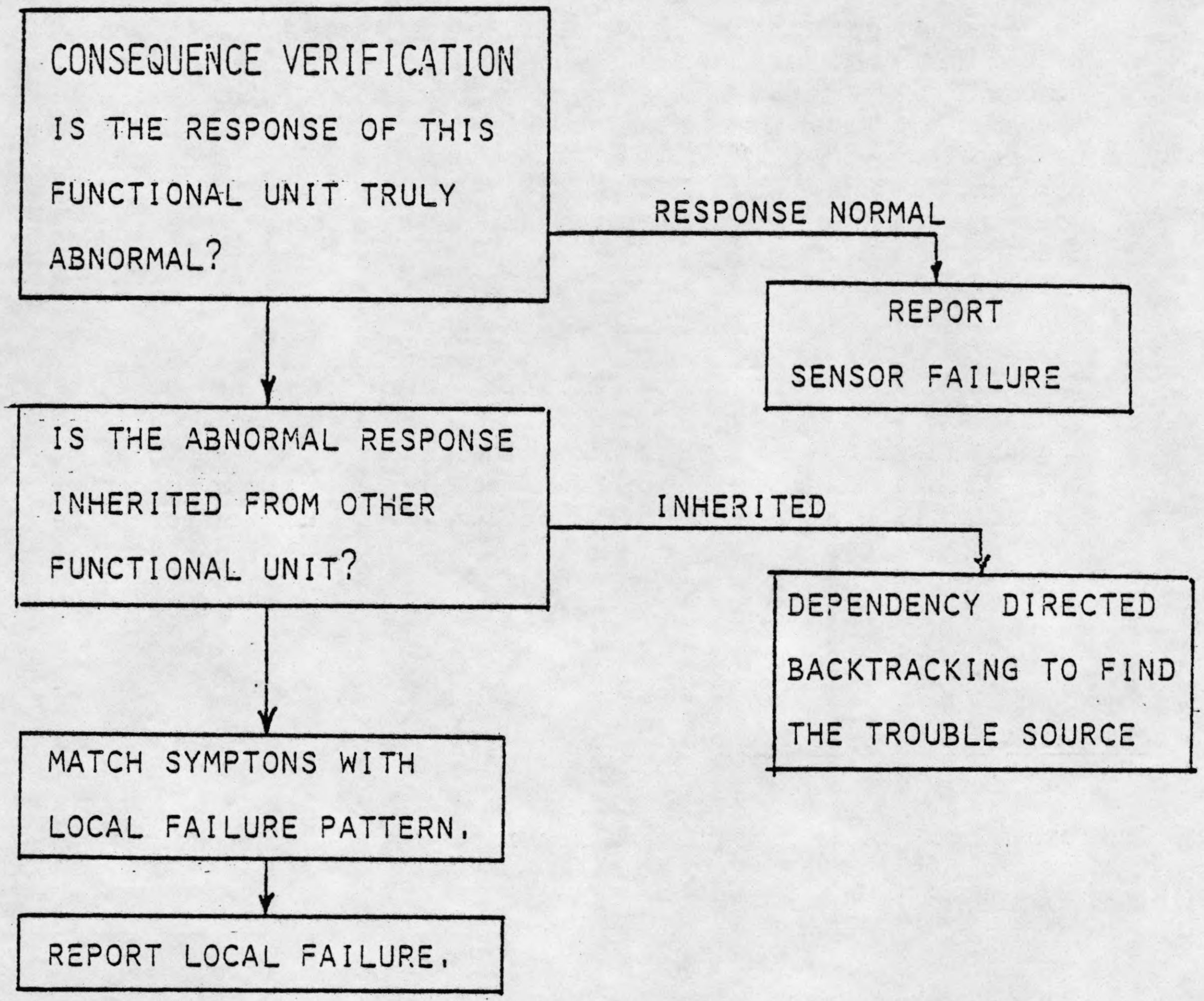

CONSEQUENCE VERIFICATION
IS THE RESPONSE OF THIS FUNCTIONAL UNIT TRULY ABNORMAL?
RESPONSE NORMAL
REPORT SENSOR FAILURE
IS THE ABNORMAL RESPONSE INHERITED FROM OTHER FUNCTIONAL UNIT?
INHERITED
DEPENDENCY DIRECTED BACKTRACKING TO FIND THE TROUBLE SOURCE
MATCH SYMPTONS WITH LOCAL FAILURE PATTERN.
REPORT LOCAL FAILURE.

FIG. 5-3. FAULT DIAGNOSIS PROCESS.

subsystem. As a result, the corresponding Oil system, Hydraulic system and Generator system are the targets to be used for the verification of the Engine system. If the abnormal engine performance is confirmed, then the Dependency-directed Troubleshooting phase begins. By refering to the "functional dependency" description in the knowledge base, the Diagnosis system finds those subsystems whose failures may cause current abnormal performance in the Engine subsystem. In this example, the Fuel subsystem is the only candidate. Now a question is to be raised on the Fuel subsystem. The process of HYPOTHESIZATION-VERIFICATION-BACKTRACKING is again repeated on the Fuel subsystem (which may finally trace back to a faulty Electrical subsystem).

4. Fault Identification

   If the results from all of the dependency-directed backtracks are negative, then an inherited failure is not likely and the Diagnosis system concludes that it is a failure local to the current subsystem. It then matches the given abnormal situation with each of the local failure pattern to pinpoint the cause of failure.

5.5 Conclusion and Perspective

The capability of a computer Diagnosis system is determined by the following factors:

1. The accuracy of the subsystem model, especially its resolution power when multiple failures have occurred.

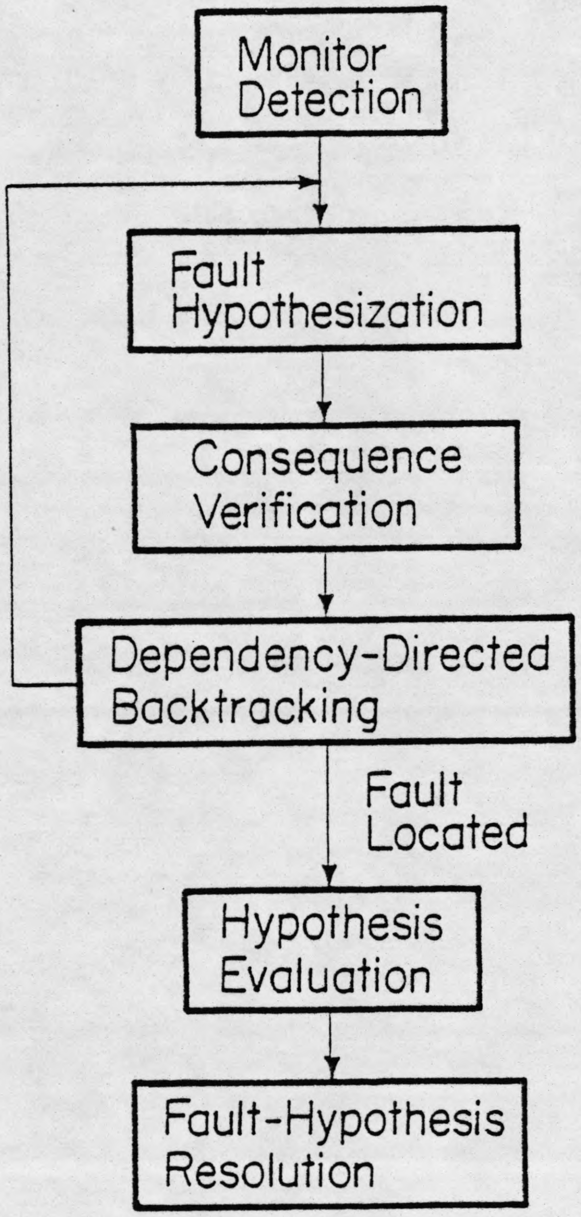
   Based on regular engine parameters (N1, N2, EGT, EPR, FF), it is difficult to recognize a mixed failure situation caused by both malfunctioning engine fuel-control and faulty bleed-valve. A solution is to add more sensors to the engine subsystem to provide detailed information within the engine.

2. The amount of sensor information available -

   The computer-based system suffers from a physical disadvantage - it does not have the visual and other sophisticated human sensory input to get the "feeling" about the flight. However, one important reason why more sensors are not added to current airplane is that excessive instruments tends to distract the pilot more than to help. The computer, with its tremendous data processing capability, can make better use of the large amount of sensory information.

3. The ability to suggest Diagnostic Experiment -

For any practical purpose, the Diagnosis system will work on a domain with only limited sensors. Thus in many cases, it is not possible to expect the Diagnosis system to come with one exact answer. Based on the information presented at the time, the Diagnosis system can pin down the possible causes of failure to a relatively smaller set. In order to determine what alternatives the pilot might have under such circumstances, a purposeful experiment can be suggested to the pilot to perform on the aircraft, and then can be properly followed up from observing the response [see Fig. 5-4]. For example, if one of the engine parameters indicates a possible flame-out while other engine indications are normal, a diagnostic experiment to reduce engine throttle setting by 50% will give a clear picture about the ambiguous situation.



FP-6522

FIG. 5-4. A MORE ELABORATE FAULT DIAGNOSIS SCHEME.

6 CONCLUSIONS AND FUTURE DIRECTIONS

In this report we outlined several concrete examples of how one could program a computer to exhibit abilities of intelligent monitoring in the cockpit environment and to automatically diagnose faulty subsystems. Such a system could be of great value in increasing the safety of a flight, as well as in lowering the workload during such busy times as landing and take-off, thus allowing the crew to focus their attention on more important tasks at hand.

For any given context or state of the aircraft, monitoring can be achieved by comparing values of variables to what they are supposed to be. The difficulty is due to the dynamic nature of the flight where references are indeed moving targets which cannot be totally put into the form of tables for static comparison.

To solve the problem in its full dynamic context we propose the approach of an intelligent system based on an internally stored knowledge base. With the help of the knowledge base the system is capable of keeping up with changing contexts and fluctuations in variables, and thereby is able to derive a set of dynamic references for verification and checking. If deviations are detected, a warning is issued to the diagnosis system which will then determine the probable cause or the faulty components, so that an alternative course of actions can be planned.

The system is intelligent because it is:

1. capable of monitoring the operation of the aircraft through indirect means and the use of dynamic references,
2. capable of making automatic diagnoses of subsystems containing single and multiple faults of components or subsystems, and

3. capable of planning, problem-solving and generating alternative plans based on the knowledge of aircraft system models.

We believe that the present system is an impressive demonstration of the potential of our approach, and this progress leads to a number of promising avenues for further investigation.

1) The multi-level structure proposed lends itself to the study of problem-solving activities that are not possible otherwise. A constant look-ahead is possible by contemplations that become necessary because of a potential problem. With the slightest sign of a possible malfunction, one could begin to anticipate a potential emergency and prepare to act in time.

2) It is also possible to consider the case of weather difficulties or any other causes that may make it necessary for the system to derive a new flight plan which optimizes in the multi-dimensional world of energy, time, facilities available, and passenger inconvenience.

3) A system can be developed to analyze the consequence of any potential action and to reason out what modifications are necessary.

4) The knowledge base can be extensively programmed to serve as an expert system onboard for the purposes of crew consultation. Automatic check-listing will then become a simple application of the broad range of possibilities of usage.

5) The modeling technique can be extended to include both qualitative and quantitative information so that predictions can be made quickly as well as logically.

These are just some of the immediate possibilities for extending the work in useful directions. It is obvious that when such systems are available, experts in the aviation industry, in both the airline and the aircraft manufacturing sector, will find a number of uses for the basic system and will adapt it to many new applications.

REFERENCES

1. Quarterly Progress Report Number 2, Artificial Intelligence and Human Error Prevention: A Computer Aided Decision Making Approach, Coordinated Science Laboratory, University of Illinois, Urbana, Illinois, February 1978.

2. Ross, C. B., "Flight Crew Interviews Concerning an Airborn Computer System", TR-2 on Artificial Intelligence and Human Error Prevention: A Computer Aided Decision Making Approach, Coordinated Science Laboratory, University of Illinois, Urbana, Illinois, October 1978.

3. Morishige, R., "Analysis of the Functional Requirements for an Intelligent Airborn Computer System", TR-1 on Artificial Intelligence and Human Error Prevention: A Computer Aided Decision Making approach, Coordinated Science Laboratory, University of Illinois, Urbana, Illinois, March 1978.

4. Final Report, Artificial Intelligence and Human Error Prevention: A Computer Aided Decision Making Approach, Coordinated Science Laboratory, University of Illinois, Urbana, Illinois, January 1979.

5. Patrick Henry Winston, "Learing Structural Descriptions from Examples," PhD thesis, in *The Psychology of Computer Vision*, edited by Patrick Henry Winston, McGraw-Hill Book Company, New York, 1975.

6. Lenat, Douglas B., "AM: An Artificial Intelligence Approach to Discovery in Mathematics as Heuristic Search", SAIL AIM-286, Artificial Intelligence Laboratory, Stanford University, July 1976.

7. Schank, R. C. and Ableson, R. P. (1977). *Goals, Plans, Scripts and Understanding: An Enquiry into Human Knowledge Structures.* Erlbaum Press, N. J.

8. Bogen, Richard, *MACSYMA Reference Manual*, Laboratory of Computer Science, Massachusetts Institute of Technology, Cambridge, massachusetts, 1975.

9. Buchanan, B., Sutherland, G., and Feigenbaum, E. A., "Heuristic DENDRAL: A Program for Generating Explanatory Hypotheses in Organic Chemistry", in *Machine Intelligence 4*, American Elsevier, New York, 1969.

10. Shortliffe, Edward, "MYCIN: A Rule-Based Computer Program For Advising Physicians Regarding Antimicrobial Therapy Selection." Doctoral dissertation, Stanford University; Memo AIM-251, Stanford Artificial Intelligence Laboratory.

11. Sacerdoti, E., "A Structure for Plans and Behavior", SRI AI Center Technical Note 109, August 1975.