NASA Technical Memorandum 86802

Development of a Knowledge Acquisition Tool for an Expert System Flight Status Monitor

James D. Disbrow, Eugene L. Duke, and Victoria A. Regenie

January 1986

National Aeronautics and
Space Administration

NASA Technical Memorandum 86802

Development of a Knowledge Acquisition Tool for an Expert System Flight Status Monitor

James D Disbrow
Systems Control Technology, Inc , Palo Alto, California
Eugene L Duke and Victoria A Regenie
Ames Research Center, Dryden Flight Research Facility, Edwards, California

1986

National Aeronautics and
Space Administration
Ames Research Center
Dryden Flight Research Facility
Edwards, California 93523

DEVELOPMENT OF A KNOWLEDGE ACQUISITION TOOL FOR AN EXPERT SYSTEM FLIGHT STATUS MONITOR

James D. Disbrow*
Systems Control Technology, Inc.
Palo Alto, California

and

Eugene L. Duke** and Victoria A. Regenie**
NASA Ames Research Center
Dryden Flight Research Facility
Edwards, California

Abstract

Two of the main issues in artificial intelligence today are knowledge acquisition and knowledge representation. The Dryden Flight Research Facility of NASA's Ames Research Center is presently involved in the design and implementation of an expert system flight status monitor that will provide expertise and knowledge to aid the flight systems engineer in monitoring today's advanced high-performance aircraft. The flight status monitor can be divided into two sections the expert system itself and the knowledge acquisition tool. This paper discusses the knowledge acquisition tool, the means it uses to extract knowledge from the domain expert, and how that knowledge is represented for computer use. An actual aircraft system has been codified by this tool with great success. Future real-time use of the expert system has been facilitated by using the knowledge acquisition tool to easily generate a logically consistent and complete knowledge base.

Introduction

A major concern during flight testing of advanced high-performance aircraft systems is the timely and efficient monitoring of advanced avionics and digital flight control systems. These complex systems are crucial to flight safety and require engineering specialists on the ground for analysis and monitoring of system performance. Modern aircraft systems are diverse, with applications ranging from new and unusual aircraft,[1] such as the X-29 forward-swept wing, through advanced avionics and flight control systems concepts, as on the advanced fighter technology integration (AFTI) F-16, or advanced wing design and control, as on the AFTI/F-111 or the F-8 oblique wing (Fig. 1). Figure 2 illustrates present flight monitoring capabilities and the goals for the expert system flight status monitor being developed at the Dryden Flight Research Facility of NASA's Ames Research Center. Level 3, our immediate goal, portrays a system that interprets the data, provides information to the systems engineer, and allows the systems engineer real-time access to the knowledge on the system.

Techniques currently available for the engineers to monitor the flights are strip chart recorders and CRT displays for analog parameters and CRT displays and light boards for discrete information such as system status and failure indications. In the mission control center in high-stress situations, it is difficult for any individual or group of individuals to always correctly identify problems and devise corrective strategy in the short time available (for examples, see Regenie and Duke[1]). As advanced systems become more essential, monitoring them becomes more critical. Fast and informative displays of the system status can save a flight, a mission, or the aircraft itself.

The expert system flight status monitor (Fig. 3) will process the telemetry downlink failure and status words using a ground-based symbolic processor. The failure words will be processed through a rule-based model of the aircraft failure management system to arrive at an independent assessment of the state of the vehicle flight control system. If the expert system detects any failures, a second level of rules will be invoked to produce high-level evaluation of the overall health and status of the aircraft. Any detected failures will be compared to the status indicator words being output by the aircraft failure management system. Rules will also be developed to resolve discrepancies between the onboard system and the expert system. Safety-of-flight conditions will result in cautions and warnings being issued to the systems engineer, who will then be able to query the expert system for an explanation or request a more detailed description of the aircraft state.[2] The expert system will also be able to display messages or emergency procedures when necessary.

The expert system flight status monitor can be separated into two sections: the expert system itself and the knowledge acquisition tool (or "semi-intelligent" editor). The knowledge acquisition tool is an editor of sorts that provides a structured, yet flexible, method of acquiring the expert's knowledge. It also provides a means of creating a centralized data base that specifies the aircraft rules.

This paper discusses the knowledge acquisition tool, the means it uses to extract knowledge from the domain expert, and how that knowledge is represented for computer use. This is an interactive program written entirely in Common LISP. It is presently implemented on a VAX 11-750 and is being rehosted to a Texas Instruments Explorer.

*Aerospace Engineer.
**Aerospace Engineer. Member AIAA.

Knowledge Acquisition and Representation

One of the main issues in artificial intelligence (AI) today is knowledge acquisition, or getting the expert's knowledge into the system. This can be done in various ways, the most obvious being that the domain expert (in this case, the systems engineer) directly inputs the knowledge into the system. Another option is that the systems engineer instructs a knowledge engineer who then inputs the information into the system.

Edward A. Feigenbaum defined the activity of knowledge engineering as the "art of bringing the principles and tools of AI research to bear on difficult applications problems requiring experts' knowledge for their solution."[3] A knowledge engineer, acting as an intermediary between the domain expert and the expert system, is seen by some as an essential element of building an expert system (given the technophobia of many experts) and by others as an unnecessary link in the chain (given that a knowledge engineer cannot read between the lines and see the gaps and tends to have a view of the subject that is rigid, formalized, and more restricted than that of the domain expert).[4] This discussion will probably go on for years. However, one lesson learned by implementing an earlier system (a feasability demonstration) was that systems engineers could implement an expert system without the aid of a knowledge engineer. Knowledge was described to the system in a form that was understandable to the systems engineers and that coincided with their conception of the system, minimizing the scope of misunderstanding and errors. Since the systems engineer is most familiar with the system and is often involved in deciding what words will be put on the telemetry stream, it seems appropriate that the systems engineer be the one to communicate with the knowledge acquisition tool.

Knowledge Base Partitions

For efficiency, it is necessary to organize the knowledge into compact, manageable units. We refer to these units as rules. The knowledge acquisition tool developed at Ames-Dryden allows several different representations of rules (Fig. 4). Some of these representations are in the form of traditional if-then (production) rules. However, some rules are also defined in unusual formats to facilitate definition of the knowledge base and to increase execution speed of the inference mechanisms in the expert system.

The rule representations were established to eliminate, wherever possible, the production rules. The relationship between the execution time of the production rules and the number of rules applied has almost exponential characteristics.[2] It was recognized that the power and computational expense of production rules were inappropriate in some cases. The partial elimination of production rules has been accomplished by partitioning the total system knowledge base into multiple knowledge bases that can be processed sequentially. Some of these multiple knowledge bases are processed continuously until no more new facts are generated, while others are processed only once per time frame. These multiple knowledge bases can be thought of as separate knowledge bases that the expert system uses.

Knowledge Acquisition Tool

One of the lessons learned from previous AI work on programs such as DENDRAL, MYCIN, and other knowlege-based systems (including our own feasibility demonstration) is that domain-specific knowledge must not be hard wired into the system if that knowledge needs to be changed frequently.[5] It was therefore decided to build an editor that would allow easy modification of the knowledge base. What was needed was a means of forcing consistency in the rule base while deliberately keeping the representation simple and uniform enough to facilitate reading and manipulating the knowledge base. During the development of a prototype expert system flight status monitor, a definite need was found for a method to provide consistency in rule entry. For example, two rules could be entered as follows

1. If <AC Power> is failed,
   Then <Analog Reversion Mode> is failed.

2. If <the primary flight control system> has failed and <the backup flight control system> has failed,
   Then <the procedure> is ejection.

For a particular aircraft, the phrases "analog reversion mode" and "backup flight control system" may be synonymous. However, unless the computer knows this relationship, if the ac power fails and the primary flight control system fails, rule 1 would fire, but rule 2 would not. There are instances where this could be disastrous. It was decided that to alleviate this problem the knowledge base should be built using certain basic words and previously defined clauses, thus limiting the vocabulary.

Initialization

The knowledge acquisition tool assumes that the aircraft flight control system has a channelized architecture (Fig. 5) with multiple redundant digital channels. To create the data structures required to monitor the aircraft system, the expert system must know the number of channels the aircraft has. Further, it is assumed that the data associated with each channel are not all available at one time and that multiple frames may be required to complete the data transfer, it is assumed that all channels require the same number of frames, and it is assumed that each channel provides an assessment of the overall health and status of all other channels but does not contain overall self-assessment information. The knowledge acquisition tool provides mechanisms for accomodating the data and hence queries the user for this information before any of the rules discussed in the following sections are created.

Basic Words

To keep the knowledge base uniform, it was necessary to restrict the vocabulary used. This method has been used in many knowledge-based systems. For example, in applying voice recognizers to the cockpit it was found that conversations involved highly stylized syntax and a restricted vocabulary that could be reduced to merely 133 words.[6] Work is also progressing in areas where it is said that anything can be expressed using a vocabulary of only 800 English words. For the kernel of our data base, it was realized that nearly all the rules depended upon certain basic words.

These basic words, or indicators, are simply names used to identify bits in the telemetry stream or flight system time history. (As stated earlier, we use the term "rule" to represent "chunks" or units of knowledge. In keeping with that definition, and because basic words are internally structured the same as rules, basic words are also considered to be rules.) Three distinct types of basic words are used: failure indicators, status indicators, and cross-channel assessment indicators.

Failure indicators represent knowledge of the failed state of aircraft subsystems. For example, in a telemetry stream there may be a bit that represents an input sensor to the flight control system, such as a pitch rate gyro. If this bit is on, it could indicate that the subsystem has failed. The name of this failure indicator word might be "pitch rate gyro fail." Similarly, the names "roll rate gyro fail," "lateral stick fail," and "longitudinal stick fail" are other examples.

Status indicators are similar to failure indicators in nature, except they represent the status, not a failure. For example, status indicators may represent weight-on-wheels or normal mode, or air-to-air gun mode. They merely indicate the state of the system.

In modern redundant flight control systems, it is not uncommon for each computer to contain an assessment of the health of itself and the other computers. These are the cross-channel assessment indicators. Cross-channel assessment indicators are different from the failure and status indicators in that they are not entered into the system in the same manner. Cross-channel assessment indicators, in general, are generated by the system automatically. At system startup, the knowledge acquisition tool already knows how many channels there are in the system. Given the number of channels, the knowledge acquisition tool queries the user as to what the different channels will be called. For example, suppose the aircraft has a triply redundant flight control system. The knowledge acquisition tool knows there are three channels in the flight control system A, B, and C. It then queries the user as to what the different channels will be called. Given a channel, the user is required to name the channels in a manner understood by the user. The tool then builds a table equating the channels and their new given names. A typical exchange might be the following:

```
If the processor is A, what would you
  call processor B?
The user may enter   Right

If the processor is A, what would you
  call processor C?
The user may enter   Left

If the processor is A, what would you
  call processor A?
The user may enter   Self
```

The system then builds the following table:

Processor A	Processor B	Processor C
A — Self	B — Self	C — Self
B — Right	C — Right	A — Right
C — Left	A — Left	B — Left

The knowledge acquisition tool then asks for the name of a cross-channel assessment indicator as follows

```
The computer will display:  Enter name of
  cross-channel assessment indicator
The user may enter   Pitch Rate Gyro

The computer will respond with:  Which
  channels are assessed?
The user may respond with:  Self, Right
```

Cross-channel assessment indicators are then built automatically. In our example the list would look like the following

```
A assesses A's Pitch Rate Gyro
A assesses B's Pitch Rate Gyro
B assesses B's Pitch Rate Gyro
B assesses C's Pitch Rate Gyro
C assesses C's Pitch Rate Gyro
C assesses A's Pitch Rate Gyro
```

The basic words, or indicators, make up the nouns or noun phrases for the antecedents (if clauses or hypotheses) and consequents (then clauses or conclusions) of the production rules (if-then rules). When running the knowledge acquisition tool program, some of the first things that must be entered are these basic words, which are certainly aircraft dependent. The names of these indicators are also used when the data structure of the input frames is defined and, of course, in the inference mechanisms of the expert system.

When the basic words are being entered, the tool also asks for an explanation (except in the case of cross-channel bits, which are generated automatically). This allows us to enter an actual sentence describing the indicator, which is helpful since so much is based on abbreviations or acronyms. This explanation is available so that the end user will remember what a bit of knowledge represents, it also helps in training, making it

easier for knowledge to be transmitted to newcomers. The basic idea is that the tool is the expert, or at least contains the expert's knowledge.

The user has the choice of adding, deleting, or modifying these words. However, an indicator may not be deleted if it exists in a rule, the rule itself must first be modified or deleted. This prevents the existence of a rule that does not contain telemetry information. There are exceptions, however, which will be discussed later.

Cross-Channel Assessment Rules

Cross-channel assessment rules are production (if-then) rules that contain information about the assessment of one computer or subsystem aboard the aircraft by another. An example of a cross-channel assessment rule is as follows

If computer A says computer B has failed
and computer C says computer B has failed,
Then computer B has failed.

The major difference between cross-channel assessment rules and the other production rules described later is that cross-channel assessment rules are built automatically by the knowledge acquisition tool from the knowledge it contains in the cross-channel assessment indicators about the number of channels and the system's provisions for self-assessment.

Each cross-channel assessment rule also contains an explanation about the rule. The explanation is also written automatically from previously acquired knowledge about the system. The reason these rules are written automatically is that the possible conditions are so numerous. If there are m channels and n assessments per channel, then there are 2^{mn} possible assessments. So if there are three channels and each channel assesses the other two but not itself, then there are 2^6 possible assessments. In the case where each of the three channels also assesses itself, there are $2^9 = 512$ possibilities.

Multiple-Element Indicator Rules

Multiple-element indicator rules are lists of indicators that are similar in function. The primary purpose of these rules is to easily accomodate redundant elements. When these rules are applied, a fact that identifies the number of failures of the type defined by the multiple-element indicator rule is added to the main system status repository. There are two types of multiple-element rules Intrachannel rules are used to identify failures of redundant elements within a single channel of the flight control system, interchannel rules are used to identify failures in redundant elements within the flight control system as a whole. For example, consider a three-channel system and an intrachannel multiple-element rule named "pitch rate gyro." A level-1 failure would mean that one of the pitch rate gyros had failed. Similarly a level-2 failure would mean that two of the pitch rate gyros had failed.

Flight System Definition Rules

Traditional if-then production rules are used to model the vehicle's failure management system. These rules are also used to model the interconnections and dependencies within the flight system. Again, two types of these rules are used within the expert system flight status monitor intrachannel and interchannel system rules. These rules are the facts derived from the indicators, cross-channel assessment rules, and multiple-element rules to deduce information about the vehicle's system state. The results of these rules are used to detect flight system failures that might not be included in the vehicle's failure management system itself. These rules can also be used to generate messages identifying conditions of interest or concern. An example of a typical flight system definition rule is

If DC Power is on,
Then deduced AC Power is on.

Another more easily understood flight system definition rule might be

If Shields are up,
Then deduced phasers are off.

Since the expert system emulates portions of the failure management system of the flight control system, a method to differentiate between the flight control system's evaluation and the expert system's evaluation of the flight control system's health and status is needed. Therefore, the word "deduced" is added to the beginning of all expert system deduced clauses.

When flight system definition rules are entered into the system, the user is queried for antecedents and consequents. The antecedents and consequents are made up of clauses or lists of clauses. The clauses contain nouns or noun phrases for the subject, a verb or verb phrase, and an adjective. The nouns are chosen from the list of basic words (status, failures, or cross-channel assessment). The verb or verb phrase is either "is" or "is not." The adjective is either "on" or "off." For example, if "pitch rate gyro" is the noun phrase, then the following are all possibilities for the clauses

1. Pitch Rate Gyro is on,
2. Pitch Rate Gyro is not on,
3. Pitch Rate Gyro is off,
4. Pitch Rate Gyro is not off.

Notice that clauses 1 and 4 are logically equivalent, as are 2 and 3. This was allowed to facilitate knowledge engineering because some rules are logically thought of as not on rather than off. Once parsed, however, they are treated as equivalent statements. The program is totally menu driven, which significantly restricts what can be entered. We use only the words "on" or "off" for adjectives because we are concerned only with

telemetry data (bits are asserted high or low, that is, on or off).

While editing (adding, deleting, or modifying) production rules, it is also possible to enter a clause that is not built from the basic words. There are many cases when an established basic word is not appropriate. While relaying messages and warnings, for example, it may be inappropriate to use basic words. The knowledge acquisition tool therefore allows the user to write a clause (antecedent and/or consequent) to be used as a message, warning, or caution. Messages, warnings, and cautions are relayed immediately to the systems engineer who is monitoring the flight.

Conflict Detection Rules

Conflict detection rules are rules or facts that identify discrepancies in the vehicle's evaluation of its own health or discrepancies between the vehicle's evaluation and the expert system's evaluation. These rules compare the system health and status indicators provided by the vehicle's failure management system with the facts deduced by applying the system rules. The intrachannel conflict rules are used to identify conflicts within a channel and consist of pairs of indicator-like names. For example,

<Longitudinal Stick LVDT fail> is on and
<Deduced Longitudinal Stick LVDT fail> is off.

The intrachannel conflict detection rules are divided into two priority groups. If the aircraft has not detected a failure but the expert system determines a failure should have occurred, a high priority is assigned, and the systems engineer is informed that a high-priority conflict has occurred. On the other hand, if the expert system has not detected a failure but the aircraft has, a high priority is not assigned, and processing is done on a time-available basis. This is not to say that if the expert system does not identify a problem and the failure management system does, we should not worry about it. However, we do expect that if the failure management system identifies a problem, then it will reconfigure the control system to a less hazardous state. The reason the expert system might not identify the problem could be that all the data available to the onboard failure management system might not always be available to the expert system because of telemetry restrictions, or it could be merely that we have created a rule that is wrong.

The interchannel conflict rules are simply words or facts that are compared across channels. An example of an interchannel conflict rule is

<Backup Mode-A> is on and
<Backup Mode-B> is off and
<Backup Mode-C> is off.

Note that in the interchannel case, the conflict is within the airplane, not between the airplane and the expert system.

Each of the conflict rules has an associated definition of severity that is used to determine the appropriate action if a given conflict is detected.

Conflict Resolution Rules

Conflict resolution rules are if-then production rules used for fault isolation or procedure initiation when conflicting information is detected. These rules are used to detect specific failures within the vehicle's failure management system or within the onboard failure detection system. These rules have the entire system status information repository available to them. Conflict resolution rules can also be used to initiate queries to the user that will provide information about the vehicle system. These rules can add facts to the system status information repository or initiate procedures to help isolate faults.

Procedural Rules

Procedural rules are production rules whose primary purpose is to mechanize the emergency procedures associated with failures in the flight system. However, procedural rules may also be used to define any nonemergency procedure that might be needed. These rules also contain information associated with each antecedent clause that identifies where a specific fact should be sought, either in the information repository or from the user. For example, consider the case in which the expert system detects decreasing hydraulic pressure. To deduce that there is a hydraulic failure, the system needs more information, such as,

Is FLT HYD light illuminated on annunciator panel light?

The user or systems engineer could then ask the pilot this question. If the answer is yes, the expert system would check other information, such as whether or not the emergency power unit (EPU) is running. If not, it would display a message instructing the user to tell the pilot to turn on the EPU. It may then issue warnings, such as,

1. minimize control inputs,
2. deselect emergency generator, or
3. maintain as high an engine rpm as possible.

System Operability Rules

System operability rules are used in general to provide high-level information not only on the health and status of the vehicle flight system but also on the particular control system mode being used. These rules are meant to provide the user with only the most useful information (such as, "the flight system is fully operational" or "longitudinal rate damping mode is not operational"). These rules are structured as traditional if-then production rules and are arranged in a hierarchical manner. The expert system evaluates each of the system operability rules until one is satisfied. It then displays the information to the systems engineer. This is the only backward-chaining mechanism in the expert system flight status monitor. The con-

sequents of these rules are used to establish a hierarchical set of hypotheses for determining the next worst failure condition.

Frame Entry

After all the previously described information is entered into the system, the knowledge acquisition tool allows the user to enter which indicators (including failure, status, and cross-channel assessment indicators) are in which frames as well as allowing the user to name "unused bits." (Unused bits are the bits in the telemetry stream that are not used by the expert system.) The knowledge acquisition tool also allows the user to change the persistence. This means that if a bit of data changes, that change must occur for a specified number of frames before it is recognized as a change by the expert system. This persistence is specified to accomodate the fact that telemetry "dropouts" and noise can occur. To be useful in a dynamic test environment, the user is also allowed to modify this information.

Rule Checking

The development of a knowledge-based system is an iterative process in which knowledge is encoded, added, changed, and deleted. It is possible in this iterative process to leave gaps in the knowledge base, gaps that the expert may have overlooked during the knowledge acquisition process. This is particularly true in a large knowledge base like the flight status monitor where we are concerned with several hundred rules. Although it has not been implemented yet, it is planned that the consistency and completeness of the rule base will be checked. There will be checks for redundant rules, conflicting rules, subsumed rules, circular rules, unreachable clauses, dead-end clauses, and missing rules.[7] This checking will need to be done in any complicated system.

Restrictions

The application of the knowledge acquisition tool was restricted to digital flight control systems because, as a first attempt, it was important to bound the problem of designing the program and not make it too generic. A digital flight control system can be considered a stand-alone entity — something that can be bounded. If an aircraft loses a weapon or a fire-control computer, while it is perhaps serious, the situation is not critical, at least in the test environment. However, if the flight control system is lost, the airplane could be lost. The same reasoning applies for not using analog data from the airplane. It was necessary to reduce the scope while still having a realistic program.

Concluding Remarks

The knowledge acquisition tool provides a means of entering aircraft rules in an orderly manner, and it provides consistency and partitioning in rule entry. This tool provides help screens and on-line documentation. At each step in the creation of the knowledge base, it provides aid and direction in creating, modifying, and maintaining the knowledge base. As a side benefit, the tool provides automatic text generation of aircraft system rules. Thus, it provides consistent and complete systems-level documentation for all aircraft. This tool is presently being used on an advanced high-performance research aircraft at NASA's Ames-Dryden Flight Research Facility. To date, only a portion of the flight system definition rules have been implemented using this tool. Since the knowledge acquisition tool is designed to be generic and capable of accomodating a broad class of flight control systems, future use of the tool is expected on other programs.

References

[1]Regenie, Victoria A., and Duke, Eugene L., "Design of an Expert-System Flight Status Monitor," NASA TM-86739, 1985.

[2]Duke, Eugene L., and Regenie, Victoria A., "Expert Systems Developments and Application," NASA TM-86746, 1985.

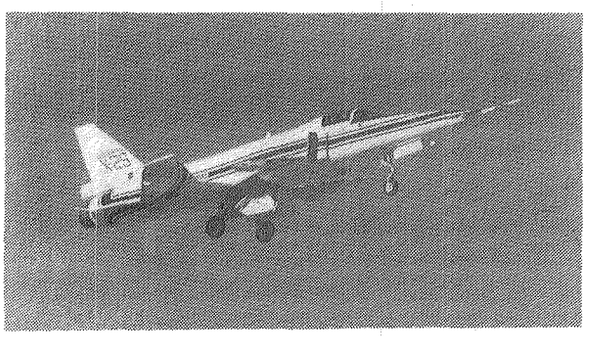
[3]Feigenbaum, E.A., "The Art of Artificial Intelligence Themes and Case Studies of Knowledge Engineering," IJCAI-77 Proceedings of the Fifth International Joint Conference on Artifical Intelligence, vol. 2, 1977, pp. 1014-1029 (p. 1014).

[4]"Survey — Expert System Shells," Expert Systems User, vol. 1, no. 2, May 1985, pp. 16-19.

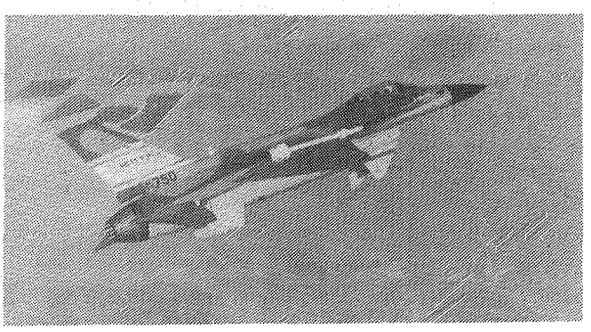
[5]Davis, R., and Lenat, D.B., "Knowledge Based Systems in Artificial Intelligence," McGraw-Hill, New York, 1982 (p. xix).

[6]Ives, Russel B., "An Approach to the Reduction of Classification Logic Complexity in an Expert System for Voice Recognition," Proceedings of the First Annual Artificial Intelligence and Advanced Computer Conference, 1985, pp. 177-181.

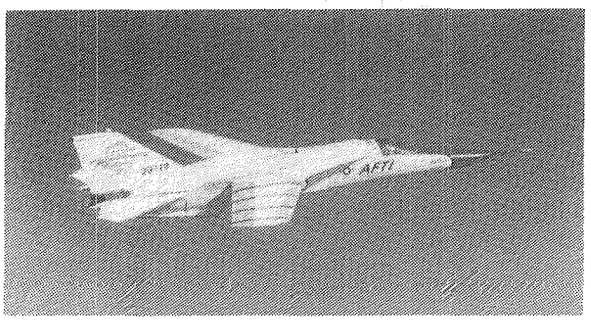
[7]Nguyen, T.A., Perkins, W.A., Laffey, T.J., Pecora, D., "Checking an Expert Systems Knowledge Base for Consistency and Completeness," IJCAI-85 Proceedings of the Ninth International Joint Conference on Artificial Intelligence, vol. 1, 1985, pp. 375-378.



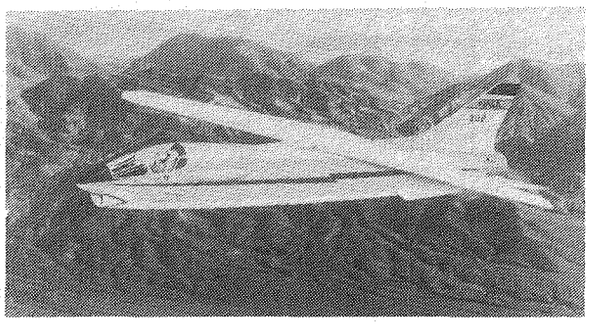
X-29A forward-swept wing advanced aircraft design



AFTI/F-16 advanced avionics and flight control system



AFTI/F-111 advanced wing control



F-8 oblique-wing research aircraft advanced wing control

Fig. 1 Typical Ames-Dryden aircraft with advanced control systems.

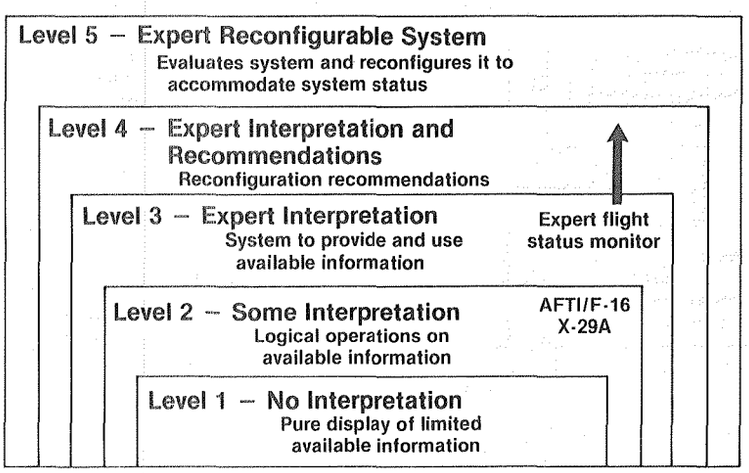


Fig. 2 Levels of flight monitoring automation.

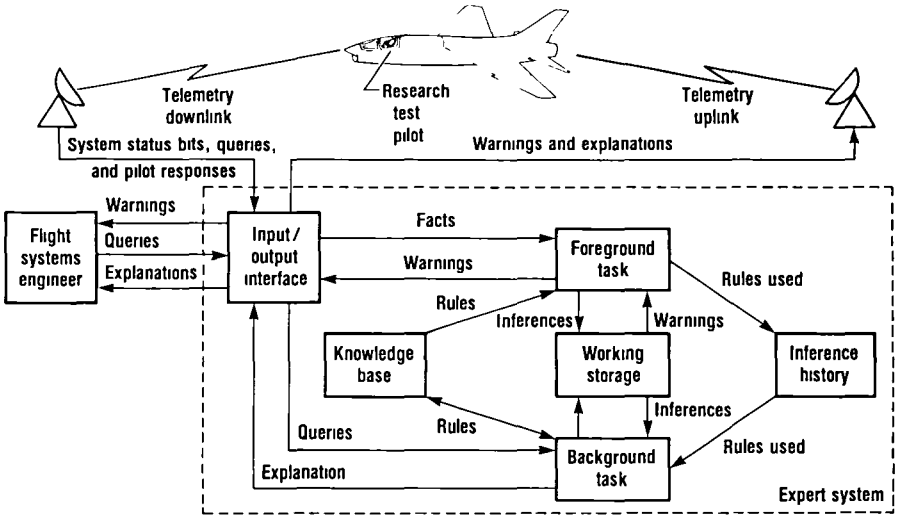


Fig. 3 Overview of expert system flight status monitor.

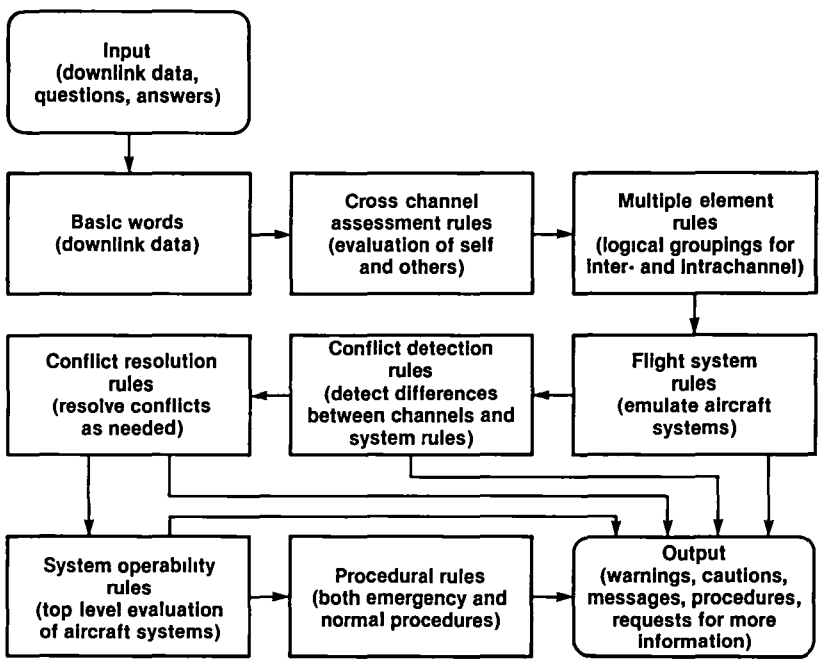


Fig. 4 Knowledge acquisition tool.

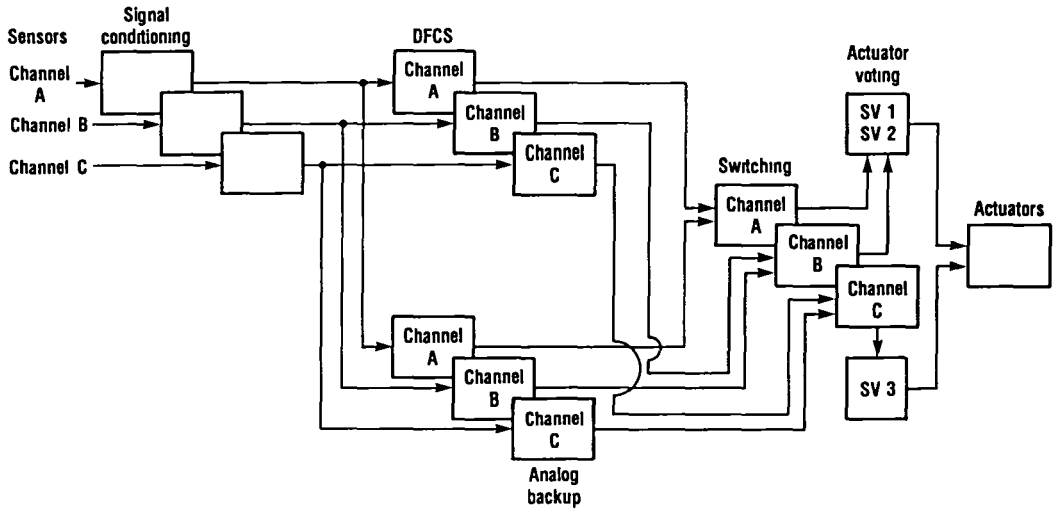


Fig. 5 Overview of digital flight control system.

1 Report No	2 Government Accession No	3 Recipient's Catalog No
NASA TM-86802		

4 Title and Subtitle	5 Report Date
DEVELOPMENT OF A KNOWLEDGE ACQUISITION TOOL FOR AN EXPERT SYSTEM FLIGHT STATUS MONITOR	January 1986
	6 Performing Organization Code

7 Author(s)	8 Performing Organization Report No
James D. Disbrow,* Eugene L. Duke, and Victoria A. Regenie	H-1332
	10 Work Unit No
	RTOP 533-02-81

9 Performing Organization Name and Address	11 Contract or Grant No
NASA Ames Research Center Dryden Flight Research Facility P.O. Box 273 Edwards, CA 93523-5000	
	13 Type of Report and Period Covered
	Technical Memorandum

12 Sponsoring Agency Name and Address	14 Sponsoring Agency Code
National Aeronautics and Space Administration Washington, D.C. 20546	

15 Supplementary Notes

Prepared as AIAA Paper 86-0240 for presentation at the AIAA 24th Aerospace Sciences Meeting, Reno, Nevada, January 6-9, 1986.

*Systems Control Technology, Inc., Palo Alto, California

16 Abstract

Two of the main issues in artificial intelligence today are knowledge acquisition and knowledge representation. The Dryden Flight Research Facility of NASA's Ames Research Center is presently involved in the design and implementation of an expert system flight status monitor that will provide expertise and knowledge to aid the flight systems engineer in monitoring today's advanced high-performance aircraft. The flight status monitor can be divided into two sections the expert system itself and the knowledge acquisition tool. This paper discusses the knowledge acquisition tool, the means it uses to extract knowledge from the domain expert, and how that knowledge is represented for computer use. An actual aircraft system has been codified by this tool with great success. Future real-time use of the expert system has been facilitated by using the knowledge acquisition tool to easily generate a logically consistent and complete knowledge base.

17 Key Words (Suggested by Author(s))	18 Distribution Statement
Knowledge-based systems Expert systems Digital flight control system Knowledge acquisition	Unclassified -- Unlimited STAR category 61

19 Security Classif (of this report)	20 Security Classif (of this page)	21 No of Pages	22 Price*
Unclassified	Unclassified	10	A02

For sale by the National Technical Information Service, Springfield, Virginia 22161.

End of Document