NRC Publications Archive
Archives des publications du CNRC

IDS: Improving Aircraft Fleet Maintenance

Wylie, A.; Orchard, Robert; Halasz, Micheal; Dubé, F.

This publication could be one of several versions: author's original, accepted manuscript or the publisher's version. / La version de cette publication peut être l'une des suivantes : la version prépublication de l'auteur, la version acceptée du manuscrit ou la version de l'éditeur.

NRC Publications Record / Notice d'Archives des publications de CNRC:
https://nrc-publications.canada.ca/eng/view/object/?id=04d870fe-7043-4a5b-95fa-4201c2187c82
https://publications-cnrc.canada.ca/fra/voir/objet/?id=04d870fe-7043-4a5b-95fa-4201c2187c82

Access and use of this website and the material on it are subject to the Terms and Conditions set forth at
https://nrc-publications.canada.ca/eng/copyright
READ THESE TERMS AND CONDITIONS CAREFULLY BEFORE USING THIS WEBSITE.

L'accès à ce site Web et l'utilisation de son contenu sont assujettis aux conditions présentées dans le site
https://publications-cnrc.canada.ca/fra/droits
LISEZ CES CONDITIONS ATTENTIVEMENT AVANT D'UTILISER CE SITE WEB.

Questions? Contact the NRC Publications Archive team at
PublicationsArchive-ArchivesPublications@nrc-cnrc.gc.ca. If you wish to email the authors directly, please see the first page of the publication for their contact information.

Vous avez des questions? Nous pouvons vous aider. Pour communiquer directement avec un auteur, consultez la première page de la revue dans laquelle son article a été publié afin de trouver ses coordonnées. Si vous n'arrivez pas à les repérer, communiquez avec nous à PublicationsArchive-ArchivesPublications@nrc-cnrc.gc.ca.

National Research Council Canada   Conseil national de recherches Canada

National Research Council Canada | Conseil national de recherches Canada
Institute for Information Technology | Institut de Technologie de l'information

NRC·CNRC

IDS: Improving Aircraft Fleet Maintenance*

R. Wylie, R. Orchard, M. Halasz,
and F. Dubé

*Published in "Proceedings of the 14th National Conference on Artificial Intelligence and Innovative Applications of AI (IAAI-97), Providence, RI. July 27-31, 1997."

Copyright 1999 by
National Research Council of Canada

Permission is granted to quote short excerpts and to reproduce figures and tables from this report, provided that the source of such material is fully acknowledged.

Canada

NRC 40181

IDS: Improving Aircraft Fleet Maintenance

Rob Wylie, Robert Orchard, Michael Halasz, François Dubé

Integrated Reasoning Group
National Research Council of Canada
Institute for Information Technology
M50 Montreal Road
Ottawa, Ontario, Canada K1A 0R6
[rob, bob, mike, francois]@ai.iit.nrc.ca

Abstract

This paper describes the Integrated Diagnostic System (IDS), an applied AI project concerned with the development of hybrid information systems to diagnose problems and help manage repair processes of commercial aircraft fleets. A study at one major airline indicated that significant benefits could accrue (approximately 2% of overall maintenance budget) through the use of innovative information technology. The IDS prototype (currently in extended field trial) takes as input a stream of messages representing maintenance and diagnostic events. These are filtered and aggregated in order to yield information in an appropriate form for various decision making tasks (and in particular for the maintenance staff while performing fault isolation and repair procedures). IDS was built using ART**Enterprise*® and makes extensive use of its rule-based and case-based reasoning facilities in order to apply various sources of knowledge (manuals, heuristics, historical data) to this problem. As well as technical issues, this paper discusses the motivation for and methodology followed in this project.

Introduction

The IDS project was initiated in 1992 by the National Research Council of Canada (NRC)[1] after examining the economic importance of diagnosing complex equipment problems correctly. Considering just 10 Canadian industrial sectors, one finds that for every dollar spent on new machinery, 51 cents are also spent on maintaining existing equipment [Statistics Canada 1990]. This amounts to repair costs of approximately $15.3 Billion per year.

Given the opportunity this indicated, the challenge was to determine the potential benefits obtainable through better use of data and information. Studies with a selected number of fleet operators in the mining, aviation, and road transportation industries showed that problems encountered by these operators had many similarities. A strategic decision was made to start with the commercial aviation industry. We approached a number of manufacturers and operators of aircraft engines with a proposal to co-develop a hybrid diagnostic system and formed a team with Air Canada, GE Canada, and General Electric (GE Aircraft Engines USA, GE Corporate Research and Development).

1. The Integrated Reasoning group does applied AI R&D on time critical decision making processes in information rich environments. NRC's mandate is to help develop new technology for Canadian Industry. Our group carries out this mandate by building functional prototypes, which explore market opportunities and identify/prove new products and services, with lead users & software developers.

In 1993, a four month study of Air Canada's maintenance operations was carried out to define the scope and breadth of IDS. Resulting highlights were:

- Diagnosing problems was seen as only part of the solution. To be truly effective, the system had to help predict incipient failures and make recommendations about the appropriate repair action given the broader context of the situation within which the problem occurred.
- The scope of the project had to extend to *all* aircraft systems on *all* fleets (Air Canada's current fleet size is 134 aircraft and growing). This operator's perspective is at odds with the perspective of equipment manufacturers. The initial intention was to tackle only the engines.
- The benefits were conservatively estimated to be in the vicinity of 2% of the entire maintenance budget.
- Decision making is highly distributed. Timely access to the right data, knowledge, and expertise is necessary to effectively carry out the maintenance mission.
- Newer generation equipment produces increasing amounts of potentially useful data. Innovative information technology was required to aid in the integration and interpretation of data.

Thus, the IDS concept had to embody the following benefits and capabilities:

- enhance maintenance performance levels at all sites,
- reduce ambiguity in fault isolation,
- advise on real-time repair action,
- provide clues to incipient failures, and
- access and display *relevant* maintenance information.

In parallel with this study, an airline market assessment was carried out. It revealed a potential world market exceeding $1B/year if similar benefits are obtained at other airlines (note: only fleet sizes >30 aircraft considered).

Decision Making Environment

This section provides a brief description of the maintenance decision making environment at Air Canada. Some aspects are specific, but the general idea applies to all airlines.

Figure 1 depicts the world within which IDS operates. Aircraft are continually on the move and turn-times at the gate continue to be shortened to maximize utilization. This creates new challenges. A number of functional groups within the airline can be involved in maintenance decisions. They are briefly described below:

- The *line technician* repairs aircraft. This happens when aircraft are met at the gate or on overnight layover. Their prime objective is to safely turn aircraft around with minimum disruption.

- *Maintenance Operations Control* views the entire fleet from a maintenance perspective. They react to any problems reported by the pilot or on-board systems to minimize disruptions. They also monitor fleet status, identify trends, deal with persistent and foreseeable problems, and determine maintenance policy.
- *Engineering* looks at specific performance indicators of the equipment and will only become involved with difficult immediate concerns, on an as-required basis. They typically have the longest decision horizon.
- The *manufacturers representative* gets involved in certain difficult problems.
- The personnel in *parts stores* must ensure that an adequate supply of spares exists from the various production sources both within and external to the airline.
- The goal of *System Operations Control* is to keep the entire fleet flying on schedule. They make system wide decisions on factors such as, disruptions due to weather or equipment failure, and flight crew readiness.

Modern aircraft, such as the Airbus A320 or Boeing B767, have systems on board which can transmit data to ground stations.[1] These data consist of routine performance snapshots (e.g. altitude, temperature, pressures, engine temperatures, valve positions, etc.), pilot messages, aircraft generated fault messages, and special purpose reports which are generated when prescribed limits are exceeded.

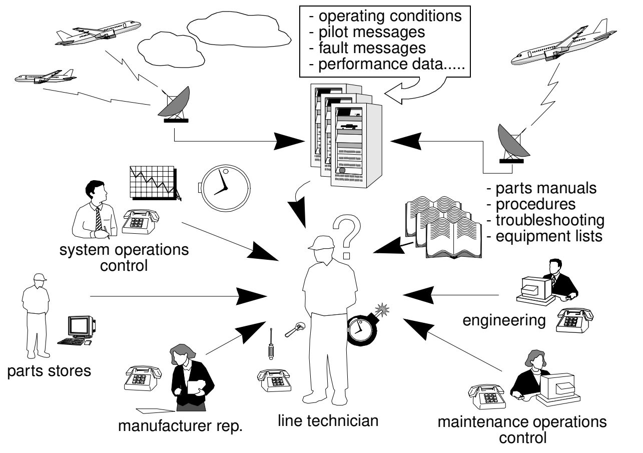

Figure 1. Current situation - airline

There are many additional databases which support maintenance. They contain descriptions of symptoms and associated maintenance actions (free form text), deferred problems, flight schedules, weather, component reliability, and parts location. There is also a wealth of useful information held at the manufacturer, and by people and information systems in the engineering and maintenance operations control departments. This is not widely distributed and thus not available to the line technician in a timely manner.

When aircraft problems occur, technicians rely heavily on professional judgement and knowledge. Additional support tools at their disposal are the:

- aircraft Built-In Test Equipment (BITE),
- Minimum Equipment List (MEL) (document governing minimum equipment needs for aircraft dispatch safety),
- aircraft log book which provides free form text descriptions of the problems encountered,
- post flight reports (a summary of the messages generated by the aircraft for a given flight leg), and
- troubleshooting manuals (currently on microfiche, but migrating to electronic formats)

The decision environment is characterized by distributed data, information, and knowledge sources. Making timely and correct decisions is important and requires access to various combinations of these sources. Given the significance of the benefits and the nature of this environment, it represents a perfect example of the type of problems the Integrated Reasoning group is interested in tackling.

Approach

Design Considerations and Prototype Scope

The scope of the IDS prototype was defined within the context of Air Canada's maintenance operation. Since a primary objective was to explore the design and implementation of *integrated* diagnostic systems, the principal challenge was to define a manageable sized prototype which convinced Air Canada and GE that the technology was relevant, and convinced NRC that the techniques were extensible/scalable to a full implementation and maintainable.

With these issues in mind, it was decided that:

- *at least* two distinct user types from within the maintenance organization should be supported,
- *entire* aircraft maintenance should be supported (rather than focus on a particular subsystem),
- as far as possible, low level diagnostic reasoning should be avoided (the trend is towards embedded diagnostics which is best left to the equipment manufacturer), and
- wherever possible, existing practices and sources of knowledge should be supported/exploited.

The two categories of users selected were line technicians and maintenance operations control staff. Their activities differ with respect to decision making time horizon, information needs, work environment, and requisite skills. However, communication between them is critical to effective fleet management. The line technicians contribute to maintenance operations control's understanding of fleet status and maintenance practices. The expertise of maintenance operations control, if easily accessible to technicians, can significantly improve diagnostic performance. Finally, the two groups must be able to interact closely in order to solve difficult maintenance problems. IDS is designed to facilitate these forms of communication.

IDS exploits a range of programming techniques. Of particular interest are the convenience of:

- rules for encoding large quantities of pre-existing knowledge (in our case, from the troubleshooting manuals) and for capturing small, complex nets of heuristic knowledge acquired from human experts, and

1. Air Canada has real-time communication infrastructure to take advantage of this.

• case-based reasoning (CBR) tools (i.e. indexing) as a means of retrieval of relevant knowledge from bodies of noisy, poorly structured, and incomplete historical data.

The fact that IDS' knowledge is in part derived automatically from troubleshooting documents meant that, with modest additional effort, we could provide these manuals on-line (along with others, totalling some 60,000 pages). This was achieved by linking Netscape and IDS (with DDE) and managing the manuals using NCSAs HTTPD. Fortunately this was easy, since without on-line manuals, IDS would have had limited success. Important notes are that it is critical to use off-the-shelf technology wherever possible and one often must include functionality with little research merit to gain prototype acceptance.

Getting Started

Choosing the tools. In mid-1994, important decisions were made. One concerned the development environment. Historically we worked with UNIX and had experience building and using a range of AI and conventional development tools. We expected that IDS would use rules, case-bases, objects, and databases. Developing in-house tools would have been too time-consuming. Fully supported products would likely be more acceptable to our partners and to a developer taking over after the prototype stage.

We evaluated a number of options including ART**Enterprise*® (A**E*)[1], G2®[2], RTWorks®[3], GEN-X®[4], JETA[5], and CLIPS[6] and chose A**E*. Major benefits were inclusion of an object system, a database integration tool, a rule inference engine, an integrated GUI development tool, and a case-base engine (at the time, it was the same as Inference's CBR Express®). It was also multi-platform compatible. So far we have migrated from OS/2 to Windows 3.1 to Windows NT (first move due to lack of continued support for A**E*; second due to inadequacy of Windows 3.1 for real-time, multi-tasking applications). This led to delays, however, A**E* code ported without too much effort both times.

Development Methodology. IDS was developed using a methodology that closely matches the Evolutionary Prototyping lifecycle model [Gilb 1988, McConnell 1996]. This approach is useful when requirements are likely to change, there is no committed set of requirements, when neither the developer or client is sure of the application area, and when the developer does not know the optimal architecture or algorithms to use. This approach helped us to move through several phases, ensuring the understanding and support of our partners/clients at each stage and allowed us to modify and extend the prototype as our own understanding grew.

Following our analysis of the sector and Air Canada's maintenance operations, we had developed a high level functional description of the IDS concept[7]. These ideas were presented in various ways (data flow diagrams, GUI mock ups, user needs analyses) to the various project participants. Though this piqued interest and generated interesting discussions, it was clear that a more concrete presentation of these ideas (i.e. a system that worked with operational data and covered enough of the fully envisioned system) had to be provided in order to get real commitment.

Operating Principles

In broad terms, IDS refines an asynchronous stream of messages of atomic symptom and repair action events into descriptions of complete fault-repair episodes. The process exploits many knowledge sources, some allowing messages to aggregate, others allowing messages (or clusters) to merge, be modified or discarded. The ideal result is clear, concise, complete descriptions of fault events which unambiguously associate symptoms and correct repair actions.

In this section we provide an overview of IDS' operation and then speak in detail about some of the more interesting subsystems. The major processing blocks, information stores (object sets, databases, case-bases, rulesets), and information flows in IDS are shown in Figure 2. Reading down from the top, centre of the diagram you see the message stream (from aircraft embedded diagnostic computers and from the maintenance databases) entering IDS.

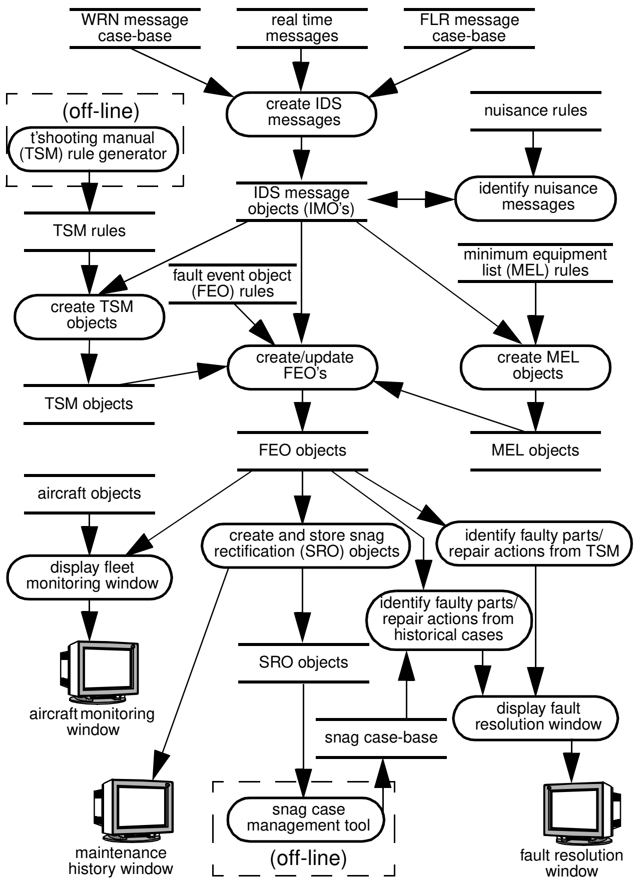
WRN message case-base
real time messages
FLR message case-base
(off-line)
t'shooting manual (TSM) rule generator
create IDS messages
nuisance rules
TSM rules
IDS message objects (IMO's)
identify nuisance messages
minimum equipment list (MEL) rules
fault event object (FEO) rules
create TSM objects
create/update FEO's
create MEL objects
TSM objects
FEO objects
MEL objects
aircraft objects
create and store snag rectification (SRO) objects
identify faulty parts/ repair actions from TSM
display fleet monitoring window
identify faulty parts/ repair actions from historical cases
SRO objects
aircraft monitoring window
display fault resolution window
snag case-base
snag case management tool
(off-line)
maintenance history window
fault resolution window

Figure 2. IDS prototype data flow diagram

1. have used ART off and on since 1985 (Inference and Brightware)
2. evaluated G2 for various projects over last 5 years & took training (Gensym)
3. have used RTWorks and in two R&D projects (Talarian)
4. GE in-house expert system development tool
5. in-house NRC development [Halasz et al. 1992]
6. see Paper Maker's Assistant [Amyot et al. 1995] & FuzzyCLIPS [Orchard 1994]
7. Air Canada's 34 Airbus A320s are initially targeted. They represent a modern aircraft and Air Canada has a lot of operational experience.

- The first processing step classifies and cleans up these messages to produce IDS Message Objects (IMOs). Classification is performed using CBR.
- The second processing step clusters these IMOs into Fault Event Objects (FEOs). This is implemented as a small set of quite complex rules which were derived through conventional knowledge acquisition sessions with engineers and maintenance technicians.

  FEO management takes not only an IMO as input but also Troubleshooting Manual (TSM) objects and Minimum Equipment List (MEL) objects.

  The TSM objects represent clusters of IMOs which are identified in the TSM as indicative of particular faults. The TSM knowledge resides in several thousand rules generated automatically from the written document.

  The MEL objects represent clusters of IMOs which are identified in the MEL manual as indicating that for safety, the operation of the aircraft is restricted in some way. The MEL knowledge comprises a small set of rules and several large lookup tables.
- The third stage of refinement associates the symptoms (i.e. message clusters in the Fault Event Objects) with appropriate repair actions. The resulting Snag (aviation term for an equipment "problem") Rectification Objects (SROs) are stored. This matching process exploits a combination of rule-based and case-based reasoning.
- Finally, suggested repair actions (bottom right of figure) are composed and presented to the user. These are derived from historical maintenance events similar to a current FEO (using CBR) and from the Troubleshooting Manual (if the FEO contains a TSM object).

Case-bases for Classifying FLR and WRN Messages. Two types of messages are generated by diagnostic routines on-board the A320 aircraft: failures (FLR) and warnings (WRN). There are about 3400 FLR and 560 WRN messages generated by the A320. A typical flight leg will generate about 6 such messages. The messages consist of text and an ATA number. ATA numbers describe the aircraft components in a hierarchical manner. For example, the first two digits represent a major subsystem. Some examples are shown below (32 indicates landing gear):

Type	Message	ATA
FLR	BRAKE TEMP SENSOR 4GW OR MONIT UNIT 2GW	324715
FLR	L L/G DOOR CLOSED PROX SNSR 27GA	323173
WRN	L/G SHOCK ABSORBER FAULT	32xxxx

It was not possible to recognize any given message received from the aircraft using a simple string match because the messages sometimes are distorted by transmission problems. They also have slightly different versions in the manual and between aircraft. The inexact matching facilities of A**E*'s CBR module (trigrams) proved useful in overcoming these difficulties (for example, '2GW' in the message may be coded as '2 GW' in the manual). Because of this we created a case-base for all the FLR and WRN messages and processed the messages, assigning each a unique identifier. Then as FLR and WRN messages are received we match the strings and ATA values against the appropriate case-base and associate the unique identifier with the message object that is created. A threshold for matching allows us to flag poor matches and make an entry in a log. Most often the poor matches are badly transmitted messages that can not be used. Occasionally we get a message that is not in our case-base and we investigate to see if this is a valid message that needs to be added to the list.

The use of a case-base to identify messages and assign unique identifiers although not a traditional use of a case-base has proven to be quite robust and has an excellent record of matching messages with the types of minor distortions that occur in the FLR and WRN messages.

Case-bases to Operationalize Experience. The snag case-base is used to retrieve descriptions of historical situations which appear similar to the current situation. This serves both as a readily usable corporate memory and as a means by which maintenance operations control can feed their knowledge back to line maintenance.

As can be seen in Figure 2, the snag case-base is managed through an off-line facility (the Case-base Management Tool). This application allows one to browse the SRO database, manually clean up the contents of a SRO, convert SROs into cases, test a new case against the existing case-base (for redundancy and consistency) and, if appropriate, add it to the case-base.

On-line, the snag case-base is searched each time a Fault Event Object is selected by the user. If a case is found which has similar symptoms (clusters of IMOs) then it is retrieved. From the retrieved cases, repair actions are extracted and used to suggest courses of action to the user.

The design of this module is evolving. Initially it was implemented as an index on the full database of SROs. At present, snag cases are managed separately from SROs, though pointers are maintained referring back to the SROs from which a case was derived. Snag cases contain match, success, and failure frequency measures based upon tests against the SRO database. As well, the repair action information in snag cases is more structured than in SROs.

Features used in case retrieval are currently restricted to individual messages and clusters. There is one feature for each type of aggregation: *textual similarity*, *time proximity*, *TSM reference*, and *Human Association Grouping* (HAG). Some effort has gone into designing these features so that search of relatively large case-bases is conducted efficiently with the tools provided by A**E*. Although both speed and quality of retrieval is adequate, our knowledge acquisition sessions with Air Canada personnel indicate that some richer features should be considered (specifically, features representing temporal relations between messages and sensor data produced by the aircraft). This may require construction of specialized CBR indexing and retrieval tools.

Automatically Generated Rules

Troubleshooting Manual (TSM) Rules: The TSM contains diagnostic information in the form of FLR and WRN message patterns that occur together and suggest probable causes for a fault (as well as procedures to assess which cause is the correct one, if any). This can be translated into a set of rules to detect (as FLR & WRN messages arrive) a match to these diagnostic situations. The firing of one of these rules will trigger the creation of a TSM object that records the list of probable causes for the situation as well as the place in the TSM where the detailed diagnostic procedures are found. We can then guide the user directly to the place in the manual where the needed information is. A sample rule is shown below:

```
(define-rule tsm:TSM-RULE-00044
  "From TSM ID 21310001 00007001"    ;;; ref to text manual
   (declare (ruleset tsm-ruleset))
    (object ?obj1
      (duplicate-in-leg ORIGINAL)
      (nuisance? ~YES)
      (fin ?fin)
      (instance-of WRN-IMO)
      (flight-number ?fltnum)
      (departing-station ?dept)
      (message-id 107)) ;;; (1.000000) CAB PR SYS 1X2 FAULT
    (object ?obj2
      (duplicate-in-leg ORIGINAL)
      (nuisance? ~YES)
      (fin ?fin)
      (instance-of FLR-IMO)
      (flight-number ?fltnum)
      (departing-station ?dept)
      (sources ?s2&:(member$ "CPC1" (idsmsg:clean-up-message-
string ?s2 nil)))
      (message-id 2079)) ;;; (1.000000) OUTFLOW VALVE ELEC1
    =>
    (bind ?imos (create$ ?obj1 ?obj2))
    (bind ?con (create$ "CABIN PRESS panel 25VU: MODE SEL
p/bsw FAULT lt on"))
    (make-instance TSM
      IMO-list ?imos
      fin ?fin
      tsm-ata 213100
      tsm-task 810802 ;; allows us to locate the 'rule' in TSM
      tsm-confirmation-list ?con)
    (free$ ?con)
    (free$ ?imos))
```

This example, in a simple form, says for a given aircraft and during a specific flight leg: ***IF*** we see the WRN message 'CAB PR SYS 1X2 FAULT' (unique identifier 107) and the FLR message 'OUTFLOW VALVE ELEC1' (unique identifier 2079, with a source string 'CPC1') ***THEN*** create a TSM object that records (among other things) a reference to the place in the TSM where the procedures for this problem are found (tsm-task 810802).

This is an ideal use of a rule-base. The rules are encoded and maintained in a well structured manual and translated into A**E* rules. When the manual is modified the rules can easily be regenerated.

Nuisance Message Rules: This is a second example of rules that are (almost) automatically generated from information in documents. These rules determine if a FLR or WRN message should be considered or ignored (i.e. a nuisance). In the TSM rule above, one of the requirements for a message is that it is not labelled as a nuisance. The document that describes nuisance message conditions was deciphered by hand and a set of rules was generated. We still need to determine a way to automate this (there are about 280 of these rules) but the documents do not easily lend themselves to this. Nuisance status depends on some conditions not detectable with the data available (hence use of a MAYBE nuisance value as well as YES and NO values) and there are other Air Canada considerations for determining nuisance status that we have yet to deal with. Future versions will look more closely at this determination.

Below is an example of a rule that determines if the WRN message 'CAB PR LO DIFF PR' is a nuisance. In this case it is, if it occurs during flight phase 6 (cruise) or 7 (approach) — information that is available.

```
(define-rule nuisance:WRN-rule-00003
  (declare (ruleset imo-ruleset))
  (object ?message      (instance-of WRN-IMO)
    (message-id 104) ;;; (1.000000) CAB PR LO DIFF PR
    (ata-chapter 21)
    (flight-phase 6 | 7))
  =>
   (set-attribute-value ?message nuisance? YES))
```

Hand Crafted Rules. Some rules were hand crafted using conventional knowledge elicitation. The expertise was derived from Air Canada personnel and from experience we gained during prototype development. We describe two such sets of rules.

Minimum Equipment List (MEL) Impact Rules: The MEL manual describes conditions under which an aircraft is allowed to fly. For example if a cockpit loudspeaker is not working (and the other is ok) the code requires that it be fixed within 10 days. On the other hand, if the two aircraft loudspeakers are inoperative, then a NOGO (i.e. "no go") situation exists. Each WRN message has been assigned a MEL impact code (GO, NOGO, GOIF, UNKNOWN, or NOIMPACT). This is used along with snag (SNG) messages to indicate that there is a potential problem with the aircraft that has MEL impact. The SNG message is a text message generated by the pilot to confirm that a WRN message was valid (they may have done some tests to confirm the condition as per on-board instructions). We identified several conditions that should raise a warning indication (either *red* - most serious, or *yellow* - potentially serious). An example is:

```
When a WRN IMO with mel-impact slot of NOGO
    and an associated SNG message
 Then create MEL object that identifies the WRN and SNG
     and set status for appropriate system for aircraft to red
```

These rules are straightforward and easily maintained, however, it is still to be determined if they are adequate to capture the essence of all situations that should be flagged and brought to the attention of the users.

Fault Event Object (FEO) Rules: A FEO represents the third level of aggregation of fault evidence. The first level of evidence contains all the individual IMOs (WRN, FLR messages). The second level contains all the TSM and MEL objects. The third level are the FEOs which attempt to collect these clusters of messages (plus other messages related through temporal proximity or textual similarity) into sets of symptoms indicating a particular fault.

A FEO may evolve during the course of its flight leg. This corresponds to the process of refining a fault hypothesis as more evidence is accumulated. A simple example of this involves addition of individual IMOs to a FEO because they originated around the same time as the important IMOs in the FEO. More complex examples are: substitution of a more specific TSM object for a less specific one; substitution of a MEL *nogo* object for a MEL *goif* object; collapsing of two similar FEOs; and deletion of one FEO if it is a strict subset of another FEO.

The FEO rules are the ones which caused the most difficulty in the prototype. They are not a set of predetermined operating procedures or well known knowledge. They represent our best efforts to aggregate information into useful and meaningful clusters. These rules were reworked at least 3 times during development, either to control the number of FEOs generated or to correct inconsistent and unwanted behaviour. It can be difficult to understand the rules, yet we have not determined that some other technique would be more appropriate for this problem.

Phases of Development

During the course of development, it was of prime importance to periodically validate design specifications against evolving user requirements. Design review sessions were held at the beginning of each phase to ensure proper communication between the various parties involved in development of the application. This allowed design validation before implementation and also helped the clients better understand the functions being developed. A prototype evaluation was done at the end of each phase with development and improvement being effected between each.

Basic Database/User Interface Development. The activities that took place in this phase were: training on the use of the development tools, creation of databases with Air Canada's operational data, and development of a GUI to demonstrate capabilities of a major part of IDS.

We had restricted access to internal Air Canada information systems. Rather than receiving AIMS[1] and MAS[2] asynchronously, we were provided with data dumps every 90 days[3]. The creation of the databases was of great value since it allowed us to gain a thorough understanding of aircraft generated information and the formal snag reporting procedures. It provided the foundation for the next development phase and data for a parallel NRC/GE/University of Ottawa machine learning research project.

The GUI was composed of four basic screens (using static data) that had limited functionality but were very successful in motivating discussion about IDS' functionality. Two of these screens are shown in Figure 3. The *Fault Monitoring* screen shows the fleet fault status. The aircraft have identification numbers and current flight status is shown (i.e. origin, en-route, or destination). Beneath each identification number are icons for the aircraft's four major subsystems: *air frame* (AF), *avionics* (AV), *engines* (ENG), and *auxiliary power unit* (APU). These icons and those for the aircraft change colour depending upon the nature of the problem. The *Fault Resolution* screen provides the technician with a recommendation as to the probable cause of the fault. It is divided into three sections: a grouping of symptoms, the ambiguity group identified by TSM rules, and the most probable causes based on past experience (case-base). It should also be noted that IDS will process all the faults generated by the aircraft and will generate recommendations for each one, provided they are distinct. From this screen the technician is provided automatic access to relevant pages of the MEL, electronic parts, and TSM. If the technicians wish to do some of their own troubleshooting they can access other screens (not shown) providing relevant snag and maintenance histories to the current problem.

1. Aircraft Information Monitoring System: *all* aircraft/pilot generated messages
2. Maintenance Automated System: free text accounts of symptoms and repairs
3. Data (~1GB) has been provided continuously since October 1994

Figure 3. Sample interface screens

The evaluation consisted of an introduction to IDS concepts and operation through formal presentation, followed by demonstration of the prototype to Air Canada focus groups. The prime goal was to provide a general understanding of IDS to a wide selection of personnel involved in subsequent evaluation. This resulted in a much clearer definition of the IDS concept and improved visibility to Air Canada's upper management.

Operational Data driving Off-line Prototype. The databases from the first phase allowed playback of historical data at any speed (up to limits imposed by the prototype). This had the benefits of allowing us to pass large amounts of data through the system for testing purposes, and providing some feel for the processing requirements of the system (in fact, on a 90 MHz Pentium we could run at 30 to 40 times real-time indicating that the expected scaling up of the prototype could be accommodated).

Due to electronic manual unavailability, this version of the prototype dealt only with selected aircraft subsystems. Selected TSM portions were transcribed from microfiche to text files from which rules (~2000) were derived automatically. Access to on-line manuals was also severely restricted in this phase for the same reason (only the MEL in Wordperfect and converted to HTML was available).

The configuration used in this evaluation is shown in Figure 4. It allowed us to simulate any time period for which we had data (approximately one year) and to observe the interaction between line maintenance and maintenance operations control personnel. It also allowed validation of the IDS design and assessment of its potential to meet requirements. Two-hour sessions, consisting of three parts were carried out with twelve users: introduction to IDS and basic training on operating procedure, accomplishment of assigned tasks, and open discussion. Each user was teamed with two evaluators, one guiding the evaluation process by asking the user to carry out specific tasks and providing assistance when required, and the other taking notes of the user's comments and monitoring the process.

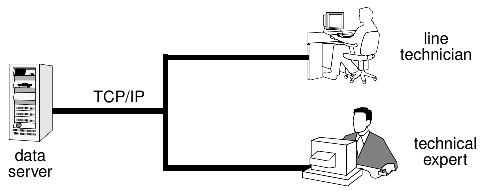

Figure 4. Set up for second evaluation

Results were positive and confirmed that the overall concept and functions implemented were meeting client requirements. Using actual data to drive IDS gave a feeling of reality which was important to maintaining a high interest level of the evaluators. Having the users operating the application under close supervision allowed us to limit any damage arising from operation of fragile software and to maintain client confidence that IDS might actually become a robust application. It was clear that the system had value but the next iteration had to provide on-line manuals.

On-line Prototype with Real-time Data. Critical to this phase was the extension to the entire aircraft, the inclusion of full on-line manuals, the use of real-time data and at least a modest case-base of historical problem resolutions.

We were expecting access to manuals in SGML, to use these directly and, in the case of the TSM, to extract the rules contained in them. This did not come to pass and we were ultimately forced to make use of assorted manuals (from various sources and in different formats). They were converted to HTML and rules extracted (now about 9000).

For on-line evaluation, IDS has been installed at two sites: airport line maintenance (Toronto), and maintenance operations control (Montreal) as shown in Figure 5. Other elements of the system include:

- a data server that preprocesses the data stream, archives it, and serves it to any IDS client nodes that are running,
- a HTTP server that provides access to electronic manuals and manages a comment/bug report facility, and
- a captive IDS client node at NRC for monitoring.

The systems access live A320 MAS and AIMS data. To collect user comments and to react promptly to bug reports, the functionalities provided by internet browsers (Netscape version 3.0) were exploited. HTML forms were created and linked to the IDS application which allowing users to enter comments while using the system.

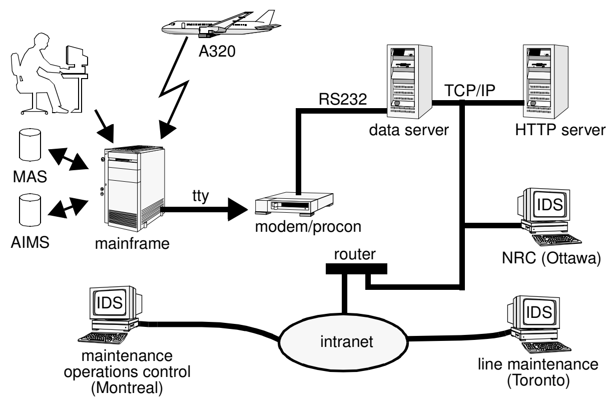

Figure 5. Set up for third evaluation

This evaluation (minimum 4 month) determined the usefulness of the application in the operational environment. Equally important are the ability to assess user acceptance, and how IDS meets the prime objective of improving overall aircraft operation and maintenance efficiency.

Feedback indicates that an "intelligent" application automatically collecting, grouping and assessing sets of fault symptoms, and automatically alerting maintenance personnel is an asset. Also the ability to automatically index to the pertinent maintenance manual pages to support the fault resolution process is extremely useful. Overall recommendations are for a system incorporating more intelligence and integrated with most of the information systems related to the maintenance operation.

Conclusions

Technically, IDS has confirmed our belief that hybrid reasoning systems (in which appropriate reasoning techniques are integrated to operationalize diverse knowledge) are practical and provide promise for further research. They can be built using currently available tools, but a fair degree of customization and innovation is required to produce acceptable results.

IDS has demonstrated that using appropriate tools is critical to success when building applied AI systems. We feel that ART**Enterprise*® reduced the effort involved in the development task though it is difficult to quantify. Using beta versions forced us to patch or work around many bugs. Also, due to its continuous, time constrained, asynchronous character, IDS is not typical of the applications for which A**E* is commonly used. This has stretched A**E*'s limits and identified some areas needing improvement.

Other examples of off-the-shelf software easing implementation include using pcANYWHERE®[1] to enhance our

1. Symantec Corporation

ability to provide support to remote IDS nodes, and embedding Netscape to provide on-line documentation and communication. Both of these are excellent examples of off-the-shelf functionality costing nearly nothing to incorporate and radically improving utility.

IDS has also confirmed our belief that, to be useful, almost any decision-making/decision-support tool must be closely coupled to the organization's underlying information flows. A corollary to this is that your development tools must provide good data integration support.

As an experience in managing applied AI, this project is teaching us a lot about development methodology and the need for flexibility: continually reviewing functional and design decisions. Interestingly, our "evolutionary prototyping" approach has forced us to address many system maintenance issues. As a side effect of the development strategy, we have developed a reasonably complete suite of maintenance and validation tools.

As a government research laboratory, we have carried many aspects of the work described in this paper as far towards an operational form as can be justified.[1] We have made a lot of progress in demonstrating both the importance and viability of this sort of system for an airline. The intent is to build IDS in stages, with each building block providing measurable utility and being added once technical issues are better understood. The existing prototype serves as the foundation for further development.

Future Work

Our role in the project is evolving towards a longer term focus. IDS has broad scope. The prototype, along with its infrastructure and data resources provides an excellent platform upon which to test new ideas. Many important emerging issues we see are to:

- explore more refined ways in which to extract meanings from free form text;
- improve upon the automated symptom clustering strategies and the indexing of the case-bases;
- determine if the diagnostic coverage of the case-based reasoning paradigm is adequate when applied to the collective "experience" of Air Canada *or* if another operator's experience should be shared *or* if it is worthwhile to explore deeper reasoning techniques;
- automate case creation and maintenance (from AIMS and MAS data);
- integrate GE Aircraft Engine performance software and experiment with model-based trend analysis;
- continue experiments with automatic feature extraction from aircraft performance data to provide richer symptom sets for case retrieval. We are currently investigating knowledge and data driven constructive induction, and automatic trend recognition;
- provide robust repair management advice by extending IDS to reason about other data sources and knowledge such as: downline station capability, component repair history/reliability, aircraft repair/maintenance history, deferred problems, parts location, schedules, and flight movement (including weather);
- extend to other Air Canada aircraft, such as Airbus 319/340, Canadair CL65, and Boeing 767/747;
- investigate and develop a suitable architecture to support large numbers of geographically dispersed users;
- develop and test Personal Communications Services (PCS) solutions for end users; and
- generalize the concepts and extend them to other fleet operation applications.

1. Effort to date (since 1994): NRC-11, GE-2, and Air Canada 1.5 person-years. This will shift more towards the partners in near future.

Acknowledgments

Many people are involved in this project, but special thanks for their continued support and patience, go to: Sylvain Pilon (GE Canada - development team), Alan Pisano (GE - Lynn, MA), and Alan Reece (GE Canada). From Air Canada: John DiDonato (line maintenance - Toronto), Jeffrey Kastner (Business Innovation), Laura Logan (Technical Operations), Joe Mallory (Information Technology), Chris Sowerby (Propulsion Engineering), Gerry Richardson, Collin Drennan, and Henry Dungen (all of Fleet Management). From NRC: George Forester (Business Development), Reg Shevel, and Marvin Zaluski (both of development team).

References

1. Amyot, R.; Gowing, J.; Wylie, R.; Henzell, R.; Futcher, J.; Reinsborough, J.; Coderre, A.; Henzell, P. and Vadas, O. 1995. Steam and Condensate Diagnostic Expert System, *81st Annual Meeting, Technical Section, Canadian Pulp and Paper Association*, Montreal, pp. A41-A44, Jan. 31 - Feb. 3.
2. Gilb, T. 1988. *Principles of Software Engineering Management*, Addison-Wesley.
3. Halasz, M.; Davidson, P.; Abu-Hakima, S.; and Phan, S. 1992. JETA: A Knowledge-based Approach to Aircraft Gas Turbine Engine Maintenance, *Journal of Applied Intelligence* 2, 25-46.
4. McConnell, S. 1996. *Rapid Development: Taming Wild Software Schedules*, Microsoft Press, pp 147-48/433-43.
5. Orchard, R. 1994. FuzzyCLIPS Version 6.02A Users's Guide, *NRC report ERB-1045.*
6. Statistics Canada, 1990. *Private and Public Investment in Canada*, Investment and Capital Stock Division, Revised Intentions, catalogue 61-206 Annual.