

AD-A242 025



AGARD-CP-499

①

AGARD-CP-499

AGARD

ADVISORY GROUP FOR AEROSPACE RESEARCH & DEVELOPMENT

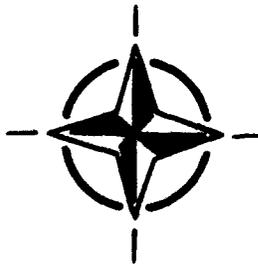
7 RUE ANCELLE 92200 NEUILLY SUR SEINE FRANCE

AGARD CONFERENCE PROCEEDINGS 499

DTIC
S
C
D

Machine Intelligence for Aerospace Electronic Systems

(L'Intelligence Artificielle dans les
Systèmes Electroniques Aérospatiaux)



NORTH ATLANTIC TREATY ORGANIZATION

CLASSIFICATION

Approved for public release;

Distribution Unlimited

Distribution and Availability on Back Cover

AGARD

ADVISORY GROUP FOR AEROSPACE RESEARCH & DEVELOPMENT

7 RUE ANCELLE 92200 NEUILLY SUR SEINE FRANCE

AGARD CONFERENCE PROCEEDINGS 499

Machine Intelligence for Aerospace Electronic Systems

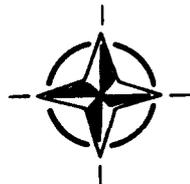
(L'Intelligence Artificielle dans les
Systèmes Electroniques Aérospatiaux)

Accession For	
DTIC	<input checked="" type="checkbox"/>
DTIC TAB	<input type="checkbox"/>
Unannounced	<input type="checkbox"/>
Justification	
Distribution	
Availability Codes	
Avail. and/or	
Dist	Special
A-1	

91-15507



Papers presented at the Avionics Panel Symposium held in
Lisbon, Portugal 13th—16th May 1991.



North Atlantic Treaty Organization
Organisation du Traité de l'Atlantique Nord

91 1113 002

The Mission of AGARD

According to its Charter, the mission of AGARD is to bring together the leading personalities of the NATO nations in the fields of science and technology relating to aerospace for the following purposes:

- Recommending effective ways for the member nations to use their research and development capabilities for the common benefit of the NATO community;
- Providing scientific and technical advice and assistance to the Military Committee in the field of aerospace research and development (with particular regard to its military application);
- Continuously stimulating advances in the aerospace sciences relevant to strengthening the common defence posture;
- Improving the co-operation among member nations in aerospace research and development;
- Exchange of scientific and technical information;
- Providing assistance to member nations for the purpose of increasing their scientific and technical potential;
- Rendering scientific and technical assistance, as requested, to other NATO bodies and to member nations in connection with research and development problems in the aerospace field.

The highest authority within AGARD is the National Delegates Board consisting of officially appointed senior representatives from each member nation. The mission of AGARD is carried out through the Panels which are composed of experts appointed by the National Delegates, the Consultant and Exchange Programme and the Aerospace Applications Studies Programme. The results of AGARD work are reported to the member nations and the NATO Authorities through the AGARD series of publications of which this is one.

Participation in AGARD activities is by invitation only and is normally limited to citizens of the NATO nations.

The content of this publication has been reproduced directly from material supplied by AGARD or the authors.

Published September 1991

Copyright © AGARD 1991
All Rights Reserved

ISBN 92-835-0628-6



Printed by Specialised Printing Services Limited
40 Chigwell Lane, Loughton, Essex IG10 3TZ

Theme

A large amount of research is being conducted to develop and apply Machine Intelligence (MI) technology to aerospace applications. Machine Intelligence research covers the technical areas under the headings of Artificial Intelligence, Expert Systems, Knowledge Representation, Neural Networks and Machine Learning. This list is not all inclusive. It has been suggested that this research will dramatically alter the design of aerospace electronics systems because MI technology enables automatic or semi-automatic operation and control. Some of the application areas where MI is being considered include sensor cueing, data and information fusion, command/control/communications/intelligence, navigation and guidance, pilot aiding, spacecraft and launch operations, and logistics support for aerospace electronics. For many routine jobs, it appears that MI systems could totally displace human operators. In other situations, MI systems would provide screened and processed data as well as recommended courses of action to human operators. MI technology will enable electronic systems or subsystems which adapt or correct for errors and many of the paradigms have parallel implementation or use intelligent algorithms to increase the speed of response to near real time.

This symposium presents the results of efforts applying MI technology to aerospace electronics applications. The symposium focuses on applications research and development to determine the types of MI paradigms which are best suited to the wide variety of aerospace electronics applications.

Thème

Des efforts importants sont actuellement consacrés au développement et à l'application des technologies de l'intelligence artificielle (IA) dans le domaine aérospatial. La recherche en intelligence artificielle couvre: l'intelligence artificielle, les systèmes experts, la représentation des connaissances, les réseaux neuronaux et l'apprentissage automatique.

La liste n'est point exhaustive. Pour certains, les résultats de ces recherches auront pour effet la transformation radicale de la conception des systèmes électroniques aérospatiaux puisque les technologies de l'IA permettent la commande et le contrôle automatiques et semi-automatiques. Parmi les domaines d'application où l'IA est à l'étude on distingue: l'alignement des capteurs, le fusionnement des données et des informations, le commandement, le contrôle, les communications et le renseignement, la navigation et le guidage, l'aide au pilote, les opérations des véhicules spatiaux et les techniques de lancement, ainsi que le soutien logistique de l'électronique aérospatiale.

Ainsi, des systèmes IA remplaceront totalement l'opérateur humain pour de nombreuses tâches de routine. Dans d'autres situations encore, les systèmes IA fourniront à l'opérateur humain des données sélectionnées et traitées ainsi que des recommandations concernant les actions à prendre. Les technologies de l'IA permettront de réaliser des systèmes et des sous-systèmes électroniques qui s'adaptent aux erreurs ou qui les corrigent. Bon nombre des paradigmes peuvent être mis en œuvre en parallèle et font appel à des algorithmes intelligents qui permettent d'atteindre des temps de réponse en quasi-temps réel.

Ce symposium a présenté les efforts qui ont été consacrés à l'application des technologies de l'IA aux problèmes de l'électronique aérospatiale. Ce symposium a porté sur les travaux de recherche et de développement en cours, du point de vue des applications, afin d'identifier les paradigmes IA les mieux adaptés à l'éventail d'applications qui se présente dans le domaine de l'électronique aérospatiale.

Avionics Panel

Chairman: Dr Richard Klemm
FFM—FGAN
Neuenahrer Str.20
D-5307 Wachtberg 7
Germany

Deputy Chairman: Eng. Jose M.B.G. Mascarenhas
C-924
c/o Cinciberlant Hq
2780 Oeiras
Portugal

TECHNICAL PROGRAMME COMMITTEE

Chairman: Dr Charles H. Krueger, Jr
Director, Systems Avionics Division
Wright Research and Development Center (AFSC), Attn: AAA
Wright Patterson Air Force Base
Dayton, OH 45433
United States

Mr John J. Bart
Technical Director, Directorate
of Reliability & Compatibility
Rome Air Development Center (AFSC)
Griffiss AFB, NY 13441
United States

Prof. Dr A. Nejat Ince
Marmara Scientific and Industrial Research Center
P.O. Box 21
41470 Gebze-Kocaeli
Turkey

Mr J.M. Brice
Directeur Technique
Thomson TMS
B.P. 123
38521 Saint Egreve Cedex
France

Mr Edward M. Lassiter
Vice President
Space Flight Ops Program Group
The Aerospace Corporation
P.O. Box 92957
Los Angeles, CA 90009-2957
United States

Mr L.L. Dopping-Hepenstal
Head of Systems Development
British Aerospace plc
Military Aircraft Limited
Warton Aerodrome
Preston, Lancs PR4 1AX
United Kingdom

Eng. Jose M.B.G. Mascarenhas
C-924
c/o Cinciberlant Hq
2780 Oeiras
Portugal

Mr J. Dorey
Directeur Technique Adjoint
Thomson-CSF Branche Equipements
Aéronautiques
Cedex 67
92045 Paris la Défense
France

Mr Dale Nelson
Wright Research & Development Center
Attn: AAAT
Wright Patterson AFB
Dayton, OH 45433
United States

Mr David V. Gaggin
Director
United States Army Avionics R&D Activity
Attn: SAVAA-D
Ft Monmouth, NJ 07703-5401
United States

Ir H.A.T. Timmers
Head, Electronics Department
National Aerospace Laboratory
P.O. Box 90502
1006 BM Amsterdam
The Netherlands

AVIONICS PANEL EXECUTIVE

Lt Colonel James E. Clay

Mail from Europe:
AGARD—OTAN
Attn: AVP Executive
7, rue Ancelle
92200 Neuilly sur Seine
France

Mail from US and Canada:
AGARD—NATO
Attn: AVP Executive
APO AE 09777

Tel: 33(1) 47 38 57 65
Telex: 610176 (France)
Telefax: 33(1) 47 38 57 99

Contents

	Page
Theme/Thème	iii
Avionics Panel and Technical Programme Committee	iv
Technical Evaluation Report	TER
Keynote Address by P.O. Bouchard	K
	Reference
SESSION I – COMMAND, CONTROL, COMMUNICATIONS, AND INFORMATION (C³I)	
↳ Machine Intelligence for Survivable Communications Network Management by N.P. Kowalchuk	1
↳ A Distributed Environment for Testing Cooperating Expert Systems by J.D. Grimshaw and C.S. Anken	2
↳ Heuristic Route Optimization: A Model for Force Level Route Planning by J.L. Barboza	3
SESSION II – SPACE OPERATIONS	
↳ Advanced Satellite Workstation (ASW) by T.E. Bleier, S. Hollander and S. Sutton	4
↳ A Synergistic Approach to Reasoning for Autonomous Satellites by J.M. Skinner and G.F. Luger	5
↳ Spacecraft Electrical Power System Fault Detection/Diagnosis and Resource Management by P.J. Adamovits and E. Jackson	6
SESSION III – OFFENSIVE/DEFENSIVE SYSTEMS ELECTRONICS	
↳ TACAID – A Knowledge Based System for Tactical Decision Making by K. Roberts	7
↳ Automated Threat Response Recommendation in Environments of High Data Uncertainty Using the Countermeasure Association Technique (CMAT) by G.B. Chapman, G.E. Johnson and R. Burdick	8
↳ Future ESM Systems and the Potential for Neural Processing by A.G. Self and G.P. Bourassa	9
↳ Neural Network Solutions to Mathematical Models of Parallel Search for Optimal Trajectory Generation by L.A. Reibling	10
↳ Application des Méthodes "Réseaux de Neurones" à la Classification Automatique de Cibles par J.-L. Regef et J. Quignon	11
↳ Threat Localization: A Major Breakthrough for Intelligent E.W. Systems by C. Maillard	12

SESSION IV – NAVIGATION SYSTEM ELECTRONICS

- > **A NASA/RAE Cooperation in the Development of a Real-Time Knowledge Based Autopilot** 13
 by C. Daysh et al.
- > **Adaptive Tactical Navigation Program** 14
 by S.L. Berning and D.P. Glasson
- > **Locally Linear Neural Networks for Aerospace Navigation Systems** 15
 by S.C. Gustafson and G.R. Little

SESSION V – CORE ELECTRONICS

- > **Pilot's Associate: Evolution of a Functional Prototype** 16
 by C.S. Lizza, S.B. Banks and M.A. Whelan
- > **Development of Tactical Decision Aids** 17
 by W.G. Semple
- ~~> **Systeme Expert Embarquable sur Avion d'Armes pour Evaluation des Performances Temps Reel** 18~~
~~par D. Servel et A. Havre~~
- > **Integrating Machine Intelligence into the Cockpit to Aid the Pilot** 19
 by E.J. Lovesey and R.I. Davis
- > **AI for RVPs, Sensor Driven Airborne Replanner (SDAR), for a Robotic Aircraft Sensor Platform (RASP)** 19A
 by R.M. Williams and J.J. Davidson
- > **A Threat Management System** 20
 by K. Holla and B. Benninghofen

SESSION VI – DIAGNOSTICS

- ~~> **Intelligence Artificielle Appliquée au Diagnostic de Pannes** 21~~
~~par M. Courtois, G. Champigneux et B. Dugas~~
- > **Expert System for the TORNADO Ground-Based Check-Out System** 22
 by J. Fey, J. Marangos, M. Merx and W. Mansel
- > **Using AITEST to Troubleshoot a Radar Modulator (RM) Unit: A Case Study in the Application of Expert Systems to Intermediate-Level Testing** 23
 by M. Ben-Bassat et al.
- > **A Knowledge-Based Assistant for Diagnosis in Aircraft Maintenance** 24
 by M.A. Piers and J.C. Donker
- > **Integrated Communications, Navigation, Identification, Avionics (ICNIA) Expert System for Fault Tolerant Avionics** 25
 by M.E. Minges
- > **A Development-Memory Approach for Enhancing Avionics Software Logistics** 26
 by M.J. Pitarys

SESSION VII – GENERIC TECHNOLOGY

- > **C³I Graphical Applications** 27
 by E.C. LaBatt, Jr
- > **Engineering Graphical Analysis Tool (EGAT) Development Program** 28
 by V.R. Clark and J.R. Diemunsch

	Reference
↳ A Knowledge-Based Intelligent Tutoring System: ACQUIRE™-ITS by B.A. Schaefer, R. Side, R. Wagstaff, O. Magusin and I.R. Morrison	29
↳ Reasoning with Uncertain and Incomplete Information in Aerospace Applications by J.C. Donker	30

TECHNICAL EVALUATION REPORT

Edward L. Gliatti
 Information Processing Technology Branch
 Systems Avionics Division, Avionics Directorate
 Wright Laboratories
 Wright-Patterson Air Force Base, Ohio, USA, 45433-6728

SUMMARY

The overall quality of the papers and presentations at this symposium was excellent. The location, facilities, interpreters, and arrangements were outstanding. The call for papers produced a moderate number of submissions, and the papers that were accepted and presented possessed a cross section of activities targeted at aerospace applications. The Machine Intelligence technologies included within the 31 applications papers were primarily Expert Systems, Knowledge Based Systems, and five Neural Network papers. The applications presented covered a variety of aerospace areas, including mission planning, navigation, communications, electronic warfare, resource management, diagnostics, target recognition, and pilot aiding. All countries had broad interests in the Machine Intelligence technologies and the application areas presented.

Obviously, a four-day conference cannot review all the MI work being accomplished throughout all the member nations. The emphasis of the symposium was on MI applications for Aerospace Electronics Systems; however, many of the papers described applied research results or laboratory simulation status. There were the exceptions, primarily in the diagnostic and navigation arena. Also, a USA Navy program for a remotely piloted surveillance mission that contained an expert system for route planning is to be test flown within the next year. The limited number of real applications presented, which are ready to transition to operational status, is an indication that the technology is still immature.

In response to the four themes of the symposium, listed completely in the next section of this report, the following can be concluded:

Response to Theme (1). Within the near future, MI will probably not have a drastic impact on the design of aerospace electronics. There are some exceptions, such as the USA Navy RPV program previously mentioned. In the far term, no assessment can be made until more MI programs move out of the laboratory into flight test and pilot comments can be reviewed. Programs that include thorough verification and validation will lay the groundwork for decisions on

transitioning the technology to aerospace vehicles. The diagnostic efforts, such as that presented by Israel for ground maintenance, will likely revolutionize the future of supporting aerospace operations, and some of these are in place and ready for the Logistics Centers. The US Adaptive Tactical Navigation program showed that MI could provide near term aid to the pilot. The maturity of the US ATN and the Navy RPV program can be attributed to the relative small size of these efforts and limited complexity of the systems.

Response to Theme (2). Within the aerospace vehicle, human operators control will not be eliminated by the implementation of MI in aerospace electronics. Routine functions can be automatically monitored, as discussed in the US Pilot's Associate program, where the health of aircraft system is continuously presented to the pilot. Also, for ground support, the use of MI can, to some extent, replace some human operators. An example of this capability is presented in the paper on Advanced Satellite Workstation, where a simple rule based system has provided the capability of removing some specialists from the support activity.

Response to Theme (3). The MI programs of the future will provide choices for the pilot from which he makes decisions. Many of the papers support this capability, most notably the US Pilot's Associate, the British TACAID, and the French Combat Aircraft Embedded Expert System for Real Time Performance Assessment programs. The papers within the Core Avionics session were among those papers that attest to MI capability to provide pilot aiding in the future. These systems are still quite complex, are only simulated in laboratory environments, and lack real time embedded electronics that are necessary before implementations in operational aerospace vehicles can take place. Advances in MI systems for pilot aiding will have to be introduced gradually to win pilot's confidence. For ground support operations, the use of MI will lessen the training needs for support personnel and could eliminate many support personnel. The papers in the space and diagnostics sessions referred directly to this capability.

Response to Theme (4). MI will enable electronic systems to correct some faults and adapt to a changing environment. Several papers addressed this area and have shown this capability in simulations. Some examples include The Integrated Communications Network Program automatically changes the communication nodes to maintain mission capabilities, the Spacecraft Electrical Power System Fault Detection/Diagnosis Program and the Resource Management System Program automatically redistributes the power in a spacecraft based on needs and failures occurring in the system, and the Integrated Communication, Navigation, and Identification Avionics (ICINA) System has detected 96% of all faults and isolates 90% of faults to a single Line Replaceable Module. There were other papers supporting this capability of finding faults. However, if implemented on-board aircraft, real time corrections of faults is presently limited by the speed of on board processors and architectures. Ground fault detection and isolation systems, where embedded processing systems aren't required, can obtain speeds required for real time operation.

Since expert system and knowledge based system paradigms were essentially the only MI technology paradigms presented at the symposium, the answer to the question of determining the best type of MI paradigms that are suitable for a wide variety of aerospace electronics applications cannot be made. Certainly the mature paradigms of expert system and knowledge based systems should be continued to be applied because of their success in most application areas addressed in this symposium. The applied research papers presented give much hope for neural networks.

The symposium clearly emphasized various important aspects of machine intelligence in aerospace electronic systems:

1. There are high expectations for machine intelligence in future aerospace electronic systems in all of the areas addressed by this symposium. The use of MI should solve current problems in activities like real time satellite control, correcting faults, providing increased pilot's situation awareness, and automating ground support. Those with respect to diagnostics and navigation can be implemented soon. Other more complex systems, like in the pilot aiding arena (as demonstrated in the Pilot's Associates program), require more improvements in the MI technology and in supporting technologies, such as real time processors.

2. There are a few areas where machine intelligence is presently implemented, but most activity is in the laboratory. Some of the present implementations are in the following areas:

- a. There are operational diagnostic programs that contain MI that are providing excellent results, decreasing manpower needs, reducing training needs, and determining faults faster than by other means. An example is in the Israel's diagnostic program for supporting aerospace electronics.

- b. Applications in the navigation area appear almost ready for operational implementation. There is a US Navy program for RPV which will fly an MI system in the near future.

- c. Progress is being made in mission planning; however, implementation is somewhat in the future due to requirements for higher embedded processor speeds and a need for an improved verification and validation capability.

3. Real time applications require improvements in processors, hardware, and theory.

4. Verification and validation methods need to be improved so that the technologies can be proven to work and sold to users.

To accelerate the implementations of these technologies for aerospace electronic applications, the kind of interactions begun by this symposium between the AGARD countries need to be continued and increased. This technology is very complex and all countries need to build on the achievements of one another to reduce the development time and costs.

THEME AND OBJECTIVES

The title of this symposium was "Machine Intelligence for Aerospace Electronics Systems." A revolution in increased complexity in airborne electronic systems has evolved to counter the sophistication of threats to the aircraft and to locate and attack targets in real time. This added complexity has increased the already heavy workload of the modern pilots that has been created for new aircraft design that have ever increasing aerodynamic, sensor, and weapon capabilities. The outgrowth of this is the desire to lower the pilot's burden and to increase his situation awareness by using increased automation, systems that automatically respond to non-essential actions, and to provide him with decision aids. Unless this autonomous capability is developed, we can expect reduced mission effectiveness. In addition, weapon systems readiness and maintenance are problems that limit availability of avionics and therefore, aircraft. To alleviate these problems, smart systems are needed that have the capabilities to adapt and reason like humans and to repair themselves or mitigate

failure through fault tolerant techniques. This has generated the current wave of activity in Machine Intelligence (MI) to provide these needed capabilities for the weapon systems of the future.

The Avionics Panel developed four main themes to consider for this symposium to determine the maturity and uses of MI for future aerospace electronic systems:

1. MI will dramatically alter the design of aerospace electronics system because MI enables automatic or semi-automatic operation and control.
2. For many routine jobs, MI will totally displace human operators.
3. For other situations, MI will provide screened or processed recommendations for courses of actions to human operators.
4. MI will enable electronic systems that adapt or correct for errors, and many of the paradigms promise to increase the speed of response to near real time through parallel implementations or use of intelligent algorithms.

The symposium was to concentrate on presenting the results of applying MI technology to aerospace electronics applications. This includes focusing on determining the types of MI paradigms which are best suited to the wide variety of aerospace electronics applications. The panel, in the call for papers, did not find an overabundance of application activities (since most of the activities are within the research arena) from which to choose; however, there was a sufficient amount of activity to report from each member nation that was it was determined worthwhile to take this initial look at the technology. Another problem in assessing papers for this symposium was in determining which papers reported truly MI activities versus those that reported work that could be considered "clever programming." This is a problem that the MI community in general still has not resolved.

For this symposium, the goals were to accurately determine the current state-of-the art of MI applications, to determine the system areas where near-term payoffs can be achieved, and to define where the member nations should put emphasis in the future.

TECHNICAL CONTENT

The symposium began with a keynote address by Philippe O. Bouchard, Brigadier General, USAF (Retired), who is the Director of the Center for Artificial Intelligence Applications, in Dayton, Ohio, USA. He set the tone for the meeting by strongly

emphasizing that the "hype" for MI systems is over, and that "opportunities are boundless for the use of MI in avionics systems, and potential applications exist in systems diagnostics, planning, command and control, intelligence, flight and fire control, and electronic warfare." He stated that events of the Persian Gulf have changed his impression of the state of advanced technology. In Operation Desert Storm, Artificial Intelligence has shown it's usefulness, and that member nation Air Forces need to maintain the technology edge presently enjoyed. Fusion of all available data will be a critical driver for future systems if air-crew situation awareness is to be improved, as required for the complex environments envisioned. Machine Intelligence efforts should concentrate on meeting the following objectives: improve training, use in unmanned flying vehicles, automate diagnostics and replace reliance on technical manuals, and develop systems that are aimed not so much at enhancing technology but being user friendly. His prediction was that future systems will routinely have MI incorporated in avionics systems; however, these system must be cautiously sold to the users.

He acknowledged that US DOD and NASA have invested funds in many applications relevant to avionics. The biggest and most complicated program and one that was reported in this symposium is the DARPA sponsored, Pilot's Associate Program. Many other less complex programs are included in numerous US programs, and some are included within the presentations here. Much of the MI technology has advanced to the point that it is considered a software engineering tool.

Session I Command, Control, Communications and Information (C3I)

The papers in this session cover the important areas of mission theater control of forces and surveillance of enemy actions. All the efforts are quite complex, large, and are being evaluated in the laboratory. Simulation results have shown more accurate assessment of the battlefield can be accomplished with the aid of MI.

In Paper No. 1, the author described an Integrated Communications Network (ICN) Management System (IMS) to allow integration of C3I functions that are adaptive and survivable in a dynamic environment, to support strategic and tactical operations. This IMS provides reactive capability despite enemy efforts and, in addition, provides corrective responses to some internal systems failures. The IMS system automatically initiates surviving resources and determines the optimal choice of end-to-end connections by weighting several

predetermined decision criteria. It evaluates conflicts and mode changes to allow for ever-changing user needs and operational conditions while allowing reduced operator intervention. The IMS is an intelligent, system-wide manager that automatically keeps the communication networks available for satisfying mission requirements. The author states that to obtain the integrated network, the resource manager requires implementation of MI techniques. The system will be prototyped in a laboratory in the fall of 1991. At present, the system automatically gives the optimum solution; however, there exists a need to shorten the response time and increase the bandwidth of the system.

Paper No. 2 described an effort to implement an in-house testbed to provide a simulation of an operational environment to validate new decision-making command and control (C2) decision aids. Most in-laboratory validations of these aids lack operational realism and, therefore, give results that are difficult to substantiate. Past attempts at a decentralized system caused bottlenecks. This led to a modified blackboard design that centralizes data and provides easy incorporation of new data and decision aids as they become available. The current system uses an Oracle relational data base, an intelligent controller for activating decision aids when needed, and an object-oriented simulation to allow realism for verification and validation. Future efforts will use a distributed object environment that allows decision aids to receive and send information. This system allows implementation of loosely coupled designed decision aids in a realistic simulation environment. Interoperability and evaluation is required before this system could be fielded in a composite operational system.

Paper No. 3 discussed a method of producing more realistic route planning. Effective route planning, to counter threats, is a requirement for future mission accomplishments and survivability, but present methods do not consider all the elements necessary to ensure success. A program called "Heuristic Route Optimization" (HERO) uses Object Oriented Programming to produce route planning that takes into account many factors that other programs do not. These variables include specific airframes, tactics, multiple sorties, threat connectivity, and alert status. The HERO program collects threat information, determines and stores costs of encountering targets, and performs an A* search to obtain lowest cost. The program can be modified dynamically as required. The system produces a mission path plan, that although may not be optimum, nevertheless, produces a good route with low lethality and cost. This tool is capable of planning multiple sorties responding to

dynamic changes while giving reasonable plans.

Session II—Space Operations

In this section, the papers are aimed at lowering the cost and improving the control of spacecraft from a ground and spaceborne perspective. Since the spacecraft is usually without a human aboard, the use of MI is vital for adaptable operations required to accomplish the mission. In addition, a key objective is for MI systems to perform the highly technical ground control function, for more accurate control and the reduction of the need for as many highly trained operators.

In Paper No. 4, the Advanced Satellite Workstation (ASW) is investigating the utility of expert systems in a simple rule-based system for the attitude control of a military satellite. Three goals were established for the program: (1) Reduce the cost of ground control, especially manpower; (2) Reduce the number of specialists needed; and (3) Automate operator tasks. Satellite information was gathered from factory experts and from flight data to make the knowledge base for the system. The expert system detects problems and offers solutions and/or automatically recalls appropriate factory information to better understand the problem. Linking the system with a main Command and Control System provided a means to evaluate the systems performance. A demonstration and report have been accomplished for this low-cost program.

Paper No. 5 discussed human reasoning and an attempt to develop a system capable of human-like reasoning for autonomous satellite control. At the present time, there are 4000 individuals supporting 80 satellite systems. By the year 2000, there will be 135 satellites. This program has as its intent the reduction of the number of personnel required while improving the accuracy of control. Humans rely on many different reasoning techniques to solve problems, and this program is integrating many of them. The system uses a blackboard architecture for blending the various kinds of reasoning through numerous reasoning modules. The modular partitioning of the system is based on reasoning methodologies rather than the usual knowledge categories. The system produces a synergistic effect through the blackboard that solves problems not solvable by any individual modules. The addition of learning feedback allows updates. Future extensions will add more reasoning modules and provide a feedback system to the modules.

Paper No. 6 described a fault detection/diagnosis and resource management system for spacecraft or space system electrical power control. Ground based monitoring and

control of space vehicles has become more demanding due to low intersect times. Thus, a requirement for a more automated system was needed. The author discussed the various costs and equipment capabilities of automation in space as compared to on the ground. One particularly critical system is the control and monitoring of the electrical power system (EPS). An AI system was developed to better distribute the power after determining faults in either the space vehicle or the ground system. The critical element of this system is the model based fault module that determines emergency responses in real time and provides input to the model based diagnostic module. The system is capable of isolating failed components, and provides input to an adaptive planner that keeps a file on successful operations. At the present time, an integrated Fault/Planning system is implemented and will be connected to a large space-breadboard EPS for Space Based Radar operation. The next phase will demonstrate the resulting radar power control system.

Session III.—Offensive/Defensive Systems Electronics

The papers of this section covered a broad range of complexity and applications including mission planning, automatic target recognition, and electronic countermeasures. Although these programs are mostly in laboratory evaluation, they nevertheless demonstrate the power of MI to help solve critical aerospace electronic application problems.

Paper No. 7 described an effort by British Aerospace, called "TACAID," to embed AI into future fighter aircraft. Earlier, work was reported to AGARD on the use of AI in mission planning in a one-on-many, air-to-air, beyond-visual-range system. Presently, they are working on the TACAID prototype to explore the use of high level heuristic reasoning to produce optimum steering and firing cues for tactical situations for single seat aircraft engaging targets beyond visual range. This system includes artificial intelligence along with conventional algorithms. The TACAID utilizes heuristics within an AI tool called "MUSE" and has been implemented on a Sun work station. Results show that this knowledge representation is flexible and can be updated easily. Future efforts dealing with uncertainty and real time may require neural networks.

Paper No. 8 discussed an artificial intelligence system approach that automatically recommends countermeasures against threats by performing analysis of signature intercepts without direct identification of the threat. The objective of the system is to recommend countermeasure response while accommodating imperfect data and errors in situation assessment. The

system uses a stored knowledge base, selects relevant experiences, and extrapolates them to the current situation to produce the best recommendation for countermeasures response. Two types of data, signal and kinematics, form the basis for the analysis and are used to calculate survivability. New knowledge can easily be added. The system makes use of various machine intelligence and processing technologies such as parallel processing, fuzzy sets, neural networks, and optimization.

Paper No. 9 provided an overview of neural processing techniques and their potential applications in ESM functions such as: tracking, deinterleaving, emitter identification, and tactical situation analysis. As radar parameters become higher in frequency and more complex in general, a radically new method is required to accomplish the ESM functions. Therefore, a neural network system was developed and evaluated. Neural networks are useful to solve the emitter identification problem due to increased density and complexity of emitters both friendly and enemy, and new war mode parameters. Results were shown from a program using probabilistic neural networks to discriminate targets in elevations with multipath present. The system is very promising; however, a few problems remain such as growth of the neural network, excessively long training, and no link to history of responses. Hardware is being developed to make implementation possible.

Paper No. 10 discussed a research activity looking at computing optimal aircraft trajectory in real time that increases survivability and mission effectiveness and minimizes radar exposure when encountering enemy threats. The normal methods of solving this problem require airborne processors with greater than 30 MIPS capability, which are not available at present. An analogous program is solved using electromagnetic fields looking at scalar field theory. A massively parallel neural network architecture is defined that solves the resulting second order partial differential equations while providing real time operation. The experimental neural network system, based on the electromagnetic analogy, has produced good path solutions.

Paper No. 11 presented results from using neural networks for automatic classification of air targets in infrared imagery. The system has two modes; one includes an operator and the other is automated with neural networks. The four layer back propagating neural network is trained using real data. This network allows good recognition, high speed, and rapidly learns new data. Hardware implementation of the vector-matrix

multiplication classification system is easily accomplished. The neural network system obtained 90% correct classification on all types of targets or buildings, whatever their orientation or distance, on any background. Real time hardware implementation of the neural network is being attempted at the present time.

Paper No. 12 stated that in hostile environments, the Angle of Arrival (AOA) is the key sorting parameter for Radar Warning Receivers due to its stability over time. A novel phase interferometer that measures AOA more than ten times better than standard systems was described. The AOA allows threat localization with increase performance by the aircraft through threat avoidance. Since the EW problem is a matter of decisions over time, smart reasoning is required. That requirement led to the inclusion of the "SEISME" expert system. This system provides threat modeling, reasoning, ambiguity determination, and heuristics for threat avoidance. Future systems will include neural networks for prefiltering. The author's view is that using neural networks for time integration would be difficult.

Session IV—Navigation System Electronics

Applications of MI to aerospace navigation electronic systems is close to being available for implementation. This was verified in the papers of this session. Results shown were most encouraging. The reason for this maturity is largely due to the smaller size and the lack of complexity of these applications.

In Paper No. 13, the author discussed the attempt to implement a real time Knowledge Based System for an autopilot for a generic high-performance aircraft. The first try using CLIPS programming language and was too slow and therefore, unsuccessful. A new implementation using a Fortran Library for Expert System Development (FLEX) provided an order of magnitude improvement in speed and satisfactory operation. Preliminary results suggest that this system is capable of real time pilot aiding. The efforts include a heavy emphasis on the verification and validation of a real time Knowledge Based autopilot.

Paper No. 14 reviewed the results of inserting Artificial Intelligence into a navigation system designed for tactical aircraft. Future aircraft require high accuracy navigation data for stand-off weapon delivery, terrain following/terrain avoidance, and night-in-weather operation. The system developed was named the Adaptive Tactical Navigation system with the purpose of selecting the best source of navigation data from nine different possible combinations of sensor inputs. The knowledge included was gathered by

interviewing many pilots. A demonstration in an F-16 dome simulation facility was successfully accomplished. The results showed that the use of an ATN system could reduce navigation errors, and the crews preferred the interaction provided. The results and experiences during this project indicated that such an intelligent system manager could produce a measurable benefit in the real world. With the inclusion of more knowledge and better sensors, the benefits of an ATN system should improve.

In Paper No. 15, the author has developed an algorithm to improve the reliability of aircraft inertial navigation systems (INS) using neural network software simulations to model systematic errors in attitude outputs. This neural network system, after training, produces the correct attitude output under almost all circumstances. The high point of this system is the use of a "jackknife" training algorithm. With this algorithm, training does not increase exponentially with the number of training examples. This training method produces a nearest neighbor neural network that is "well behaved" doesn't deviate from the training data, and therefore, provides the needed reliability. The simulations were evaluated using flight test data from sampled INS data. Locally linear and back propagation neural networks were used. The results using the flight test data provided two milliradians accuracy with five inputs from heading, pitch and roll sensors.

Session V—Core Electronics

The papers of this session exclusively covered the application of MI to pilot aiding and cockpit situation awareness in a single seat aircraft performing a tactical mission. The DARPA sponsored "Pilot's Associate Program", the largest and most complex US MI program targeted for aircraft application, was reviewed in this session. All of the efforts described in the papers are laboratory simulations that show vast improvements for the cockpit. However, real time operation and verification and validation are problems that must be resolved before serious implementation can take place.

Paper No. 16 described the USAF's attempt at developing a system to support a single-seat fighter pilot aiding capability using a knowledge-based decision-aiding system called the "Pilot's Associates Program." Two contractors, with different flight scenarios and mission requirements, were included. The program would not replace the pilot but would aid him. A video was shown that explained the system, demonstrated how the 5 expert systems might interact, and depicted the displays possible for pilot viewing. The paper discussed the four phases of the program,

getting to the real time capability to be shown in their Demonstration 4 during 1992. Present activities are trying to get real time performance and implementation on multi-chip modules with co-processors. The program had breakthroughs in the following areas: evaluation of large complex AI system, proven rapid prototyping as an effective tool in system development, and the translation of LISP into ADA.

In Paper No. 17, this British program is focused on aiding the single seat pilot by making tactical decision aids available. These aids are required to increase performance and decrease workload. This work addressed both the offensive and defensive the air-to-air combat role of modern fighter aircraft. Situation assessment algorithms were shown, along with associated cockpit displays. This paper outlined the work and achievements to date and provided informed opinion regarding the further development of Tactical Decision Aids. It discussed difficulties with the implementations including the nonexistence of a small air-worthy computer system, difficulty of obtaining a practical solution, and for a generic problem what is the right or optimum solution. Future work must address real time operation and achieving maximum performance with minimum workload.

Paper No. 18 also discussed the complexities caused by single seat aircraft in the sophisticated environments existing or expected. The author discussed the development of a Mission Management Aid using AI that fuses many of the data sources available to the pilot to make decisions, from taking care of "house-keeping" to determining optimum flight plans using the fusion of sensors and data available. Problems exist in understanding the pilot's information requirements, how to best present the data, and putting expert systems into conventional computers. Their next step will be an attempt to develop a dedicated "translation" of this expert system into a "conventional" real time language on a conventional processor. Finally, he discussed the serious problem of validating the system.

Paper No. 19 stated that due to the complexities of future aircraft and single seat aircraft, advanced aids are required for achieving mission success. This program provides mission management aids by fusing many assorted data inputs for an air interdiction mission. These aids are required due to increased fighter speeds, increasing number of cockpit systems and displays, and increases in the number of sensors. This system generates an optimum flight plan. In addition, the program automatically takes care of general housekeeping functions. Great care is taken to understand the pilot's information

requirements and displays for ease of comprehension. Finally, the system must be validated. The current status of the system is that there will be a simulation run in the laboratory in the near future. A flyable system will not be available until the year 2000.

Paper No. 19a presented a program for implementing an AI system for the control of an Remotely Piloted Vehicle (RPV). The targeted RPV program does not have man-aided ground control and does not carry weapons. The system combines AI expert systems with real time control to perform route planning for ocean surveillance. The system called "Sensor Driven Airborne Replanner" (SDAR) uses the camera system to determine for current azimuth and elevation, and the focal length of the system to determine latitude, longitude, and size of the ship under observation. Based on the response of the SDAR, autopilot commands are generated for navigation of the RPV to the best position for the mission. A laboratory simulation proved that the system could accomplish the mission. Overall, the program has developed the inference engine, rules, sensor input requirements, smart autopilot, and successfully integrated all component parts.

In Paper No. 20, a Threat Management System (TMS) was described that addresses the planning phase of a penetrating bomber mission. The prototype developed provides in-flight situation awareness, terrain and threat avoidance, and tactical advice. Preflight mission plans are developed from electronic order of battle and ground order of battle information. The results of this preflight plan is to automatically compute optimal flight paths. The system, when airborne, monitors the actual situation and replaces the mission plan as required due to changes in the environment, using knowledge about own capabilities and those of the enemy. The threat scenario is determined by using the sensor data received and the stored knowledge base. In the future, a digital map capability will be included along with real hardware.

Session VI—Diagnostics

The following papers dealing with diagnostics revealed a most mature area of MI application. The most successful uses of MI in the expert system area outside the military are already well defined for diagnosis in medicine, cataloging, ordering, and inventory. The success shown in these papers using MI, therefore, comes as no surprise.

Paper No 21 discussed the fact that one half of the costs of operating combat aircraft is due to maintenance problems.

To lower this cost requires that the maintenance be accomplished in the future as a two-level system (field, depot levels). The objective is to automate the trouble shooting and to provide non-ambiguous fault detection. The use of AI greatly aids this two level concept. However, the first studies have shown that AI by itself does not solve the problem, but must be combined with a more global philosophy. The procedure is to investigate the system from an integrated maintenance concept by checking the integrity of hardware and software components of functional chains. Intelligent machines may be used at various stages; real time during mission and fault detection at the intermediate shop and at the depot. An MI system is expected to be fielded as a full-up system by 1997. Results to date show that MI is well suited for diagnostics. The system combined techniques for best results yielding a system that detected 98% of all faults within 60 minutes.

Paper No. 22 described an Expert System called "TORRES" that identifies and isolates faults that occurred during previous flights. This is determined during post flight ground checkout. The present system checks the Tornado nose radar system. With the increased identification of faults by this system, the time and cost of performing maintenance has been greatly reduced. This has been accomplished despite the increasing complexity of this aircraft. This system also provides accurate debriefs on the cause of failure, allowing increased learning by the personnel in charge of maintenance. There is a need to include all of the electronic systems on-board the Tornado and not just the radar. The present system runs on either a PC or a Symbolics processor. A three to four times speed improvement is achieved when using the Symbolics processor.

In Paper No. 23, an expert system decision aid for assisting in aircraft maintenance was described. The system handles large scale units under test (UUT) that contain both analog and digital electronics, hydraulic, and mechanical modules. The system identifies faulty modules, determines the optimum test sequence, and learns from experiences. The diagnostic and test sequencing algorithm proves the potential of MI to perform equipment troubleshooting. The system is being used operationally at the present. The old method of doing this type of maintenance was to have numerous diagnostic charts with step-by-step instructions that led to numerous possible faults. This method, using AI, eliminates the diagnostic charts. The system displays pictures, schematics, timing tables, and color-coded data showing states of performance. This system allows quick testing, doesn't require highly trained personnel, and also includes various

support tools.

Paper No. 24 presented a knowledge-based assistant for diagnosis in aircraft maintenance. Often in the maintenance of aircraft, the diagnosis is the bottleneck in achieving timely and inexpensive repairs. Lessons learned are easily added to the knowledge base. At the present, the program is aimed at identifying problems in the air-conditioning system of transport aircraft. Since the air-conditioning system is distributed over the aircraft and since components are not directly accessible, there is no single test procedure. The AI system identifies problems using synthesis to determine cause and using past experiences and observations, hypothesizes, and symptoms to recommend solutions. Some of the elements that cause diagnosis bottlenecks include incomplete complaint information, poor documentation, lack of complaint history, and lack of feedback. The KBS will specify tests and various maintenance activities that should solve the problems. So far, the system has been validated in the laboratory. The next stage is to build a prototype system. What has been shown is that a KBS maintenance aiding system can perform diagnosis and transform results into specific tests. Past experiences are identifiable and formalized within the KBS, and within a limited scope, the system was successfully demonstrated. Lack of adequate complaint information is still a problem.

In Paper No. 25 a system was described to isolate faults to one or more modules in an Integrated Communication, Navigation, and Identification Avionics (ICNIA) system. The internal expert system provides a more reliable and fault-tolerant system. The system uses built-in tests and analysis of the airborne configuration to isolate faults. The author showed pictures of the advanced development model of the system. The software is in-flight reconfigurable to work around faults. The system allows detection of 98% of all faults, isolates 90% of the faults to an Line Replacable Module, LRM, or isolates 95% of the faults to two LRMs. The system uses Built-in-Test as the data collection mechanism for the expert system. The payoff is increased operational availability and improved supportability.

Paper No. 26 dealt with the use of MI for software supportability. Advanced avionics systems execute vast amounts of Ada software on multiple parallel computers. Avionics software support occurs after the software is fielded at the Logistics Centers. Maintenance of complex software involves many people. The majority of the cost to avionics software systems is attributed to the software support area. Correction of deficiencies and the addition of enhancements require much time for

searching and inferencing of information. The real problem is that much of the development knowledge is not available. The system described uses MI to transfer this knowledge to the user. The support system developed assists in timely placement and organization of changes to the software. It uses a rule based system and semantic networks to communicate to the user.

Session VII—Generic Technology

The papers in this session cover more general areas of possible aerospace system applications. The specific MI technologies areas of concern are quite diverse. These papers provided a most interesting close to the symposium.

Paper No. 27 detailed the results of a cognitive study looking at AI techniques to improve the man-machine interface for C3I systems. This study was required because most of the AI research for C3I focused on improvements in the basic concepts of information transmission and algorithms to improve the identification process but, did not consider the equally important part of interface to the operator through a graphics display. Two decision aids have been developed: the Tactical Expert Mission Planner (TEMPLAR) and the identification of Command and Control Operations Nodes (ICON). The TEMPLAR system is an expert system using frame-based reasoning. The ICON system is a blackboard embedded system. The techniques tried were adopted from the entertainment and advertising community. The study assumed that the required data were available and that processing power of the future will handle these future demands. The graphical representations apply to the tactical C3I domain and include mission scheduling enhancements to animate a situation assessment map over time. These presentations are understood at a glance and give the operator a better understanding of the situation. The system proved to be very user friendly and the graphics enhanced the effectiveness of using MI techniques.

In Paper No. 28, a graphics tool titled "Engineering Graphical Analysis Tool" (EGAT) was described that provides the user with a single-step capability to identify bottlenecks or incorrect parameters in an activation frame architecture. From the Advanced Tactical Navigation System program described earlier in this session, it was found that the activation framework architecture had problems setting up the control settings and interacting between the AI and conventional systems. It was extremely difficult to perform verification and validation of the knowledge base. EGAT was the tool developed to deal with these problems arising in the activation framework architecture. The graphics interface displays the interactions among all levels

of the various activation frame groups. The user can dynamically modify parameters to determine their effect on the system. This greatly reduces the effort and increases the accuracy of manual interpretation. Future work is expected to translate the system into ADA language, incorporate parallel processing, and fully develop the knowledge base for verification and validation.

Paper No. 29 discussed a knowledge-based acquisition system developed for the Canadian Space Agency using the ACQUIRE expert system shell. The system allows the users to rapidly develop their own expert system knowledge bases and convert them for use in task-oriented training. The system has a learning module that matches the user's knowledge data base with an expert's. Another module evaluates the user's accomplishments. The program, written in C, is applicable to crew training problems. Results have confirmed that the tools provide a practical means of utilizing existing industrial personnel to construct complex, automated training courses. Experiences gained should meet a broad range of training needs.

In Paper No. 30, problems with reasoning with incomplete and uncertain information for application to aerospace electronic systems were discussed. Many of the decisions to be made in an aerospace vehicle suffer from not having complete data and uncertainties within the available data. The author stated that MI was harder than researchers were led to believe! Some applications are being accomplished; however, further progress is hindered by lack of verification and validation of performance of MI implementations. For reasoning with incomplete and uncertain data, the Dempster-Schaefer and Bayesian methods were described. It is tricky setting the probability limits for these uncertainties.

RECOMMENDATIONS

There should be another symposium in two or three years, which should include presenters from this symposium reporting on their progress as well as new papers and presenters. A lecture series should be conducted on MI technology and applications to aerospace electronic systems to disseminate MI research findings more rapidly.

This symposium concluded that future MI applications should be accelerated for smaller and simpler aerospace electronic systems. The relative high degree of maturity of this type of applications reported here attests to this conclusion. Certainly, programs aimed towards the most successful applications, such as navigation

and diagnostics systems, should be accelerated to implementation. If this is done, the continued success with these less risky applications would lay the groundwork for the approval of larger system applications in the future. Also, in these days of declining defense budgets, these types of simpler applications are more affordable. This approach should not exclude work on the larger and more complex programs at a reasonable level so that the full benefits of this MI technology can be realized in the future.

One of the fundamental limitations of MI for aerospace electronics systems, as defined by most of the speakers at this symposium, is the need for real time embedded hardware. Before implementations can take place, considerable progress needs to be accomplished in the development and availability of airborne architectures and processors that provide real time operation. In addition, another limitation of MI for aerospace electronics system is in the area of verification and validation of MI solutions. Without good methods of doing verification and validations, it is difficult to assess the maturity and benefits of the technology and the developed systems will not gain the acceptance necessary for major system implementation. Therefore, it is recommended that an increasing share of

funds for MI applications for aerospace electronic systems be spent on the development of real time embedded hardware and in the verification and validation areas.

General Bouchard made interesting comments about the acceptance of MI by pilots that should be considered by MI planners and developers. In the process of trying to sell MI systems, he has found that most military pilots are not ready to accept claims that pilots need additional aid for flying their aircraft or achieving mission objectives. Pilots will not accept systems which remove control or make decisions pertaining to critical flight or mission parameters. Future systems should be sold, therefore, on the basis of their support to the pilot with little attention given to the MI included.

Many MI technologies were discussed at this symposium; however, no single approach has shown a capability of solving, in an optimum way, all of the problem areas discussed. Therefore, for applications of the future, it is recommended that hybrids consisting of mixes of MI technologies be used since they offer the best solution to many aerospace electronics system problems.

KEYNOTE ADDRESS

by

Philippe O. Bouchard

Director

Center for Artificial Intelligence Applications

3155 Research Blvd, Suite 200

Dayton, OH 45420-4066

United States

Preliminary Remarks

Thank you for the kind introduction, and thank you to our host nation, Portugal, for the excellent conference accommodations. Since submitting my remarks for clearance, a war has been fought and major United States Air Force reorganizations have resulted in name changes for many of the R&D organizations represented here today.

Therefore, I am going to depart slightly from the speech written last year, which has been overcome by many events.

Moreover, I trust my former compatriots at the United States Air Force's newly named Rome Laboratory, and those at the newly named Wright Laboratory will excuse my language if I occasionally lapse into terminology which signals out either avionics or C³I. There is no intention of enhancing the importance of one versus the other...both are equally important electronic system technologies...developed in the United States Air Force by separate organizations.

Also, please excuse my use of the term Artificial Intelligence and its abbreviation AI, the term commonly used in the United States. I will have more to say about this unfortunate wording later.

Introduction

It is opportune that AGARD should be addressing the topic "Machine Intelligence for Aerospace Electronic Systems" during a time of United States defense spending cuts and economic recession. The United States defense market will, for the next few years, best be characterized by a curtailment of major new program starts. The few new systems which survive the cuts will include major advances in electronics integration and new technologies, such as Artificial Intelligence. The Advanced Tactical Fighter (ATF) immediately comes to mind, wherein its avionics functional integration architecture coordinates sensor fusion, synthesizes cockpit displays, monitors systems' states, and provides decision aids. However new systems such as the ATF will be the exception during the decade of the 1990s.

Upgrades of existing aerospace systems will most likely be the norm, and avionics and C³I upgrades may be the method most employed for improving total system performance because of high operational payback. Since defense avionics technology has often been the source of advances in commercial aircraft, and orders for commercial jet airliners are at a record high in the United States, the requirement for innovative avionics technology should remain strong despite tight budgets. Although United States research and development outlays will slow in the early 1990s, many companies have already stated they will increase R&D spending in electronics, communications, and sensor technologies.

During my experience as Director for the Center for Artificial Intelligence Applications (CAIA), I have observed that the largest percentage of successfully implemented knowledge-based systems occur in commercial applications to improve decision making, productivity, product quality, and diagnostics. The higher risk, higher pay-off intelligent systems applications have been the domain of the military. In fact, I have observed a propensity for the United States Defense Department to invest its largest percentage of AI dollars in highly visible weapon systems applications versus the acquisition management techniques which are used to procure these same weapon systems. Nevertheless, as the AI technology for these higher risk military system applications mature, commercial spinoffs should be plentiful. Thus, the market for intelligent electronics should remain strong, and that is why this conference on "Machine Intelligence for Aerospace Electronic Systems" is timely.

This paper briefly describes one of the initiatives taken by the United States Air Force to ensure continued growth in the application of AI to pertinent problems, many in electronics. Also discussed are likely applications for future knowledge-based systems based on the Persian Gulf War experience.

The Center for Artificial Intelligence Applications

The Center for Artificial Intelligence Applications (CAIA) began its operations in October, 1987. The United States Air Force at Wright Patterson AFB in Ohio, established the CAIA through a \$10 Million contract. Its mission is to adapt AI technologies for Air Force applications and to expand the Dayton, Ohio, area AI talent base through education and training. The Center is managed and funded by the Aeronautical Systems Division, AI Technology Office, located at Wright Patterson AFB, Ohio.

The CAIA consists of a small staff and AI researchers from one commercial company and seven Ohio universities. This consortium consists of Central State University, The University of Dayton, Case Western Reserve University, The University of

Cincinnati, The Ohio State University, Sinclair Community College, Wright State University and Teknowledge Federal Systems. The CAIA staff's function is mainly administrative, although some are involved in AI training and prototyping of expert systems in the Center's well-equipped prototyping facility. However, the goal is to have the majority of the prototyping and applied research accomplished by the university and commercial subcontractors at their local facilities.

The Center's mission is to satisfy United States Air Force requirements; therefore work is focused on **AI applications** rather than basic research. Five major services are provided. *Application Screenings* are the initial risk/value assessment of a problem which might lend itself to an AI solution. Typically a panel of six to ten consortium members (engineers, scientists, and knowledge engineers) interview an expert for some two to three hours to assess AI and systems engineering risks and value to the user. Those applications which survive the initial screening are recommended for the next step, *AI Application Assessments* which are similar in format to Screenings. Again, a panel of consortium members interview an expert, this time for some two or three days to reaffirm the screening's risk assessment and to gather sufficient information to write a prototyping or applied research plan.

If the application lends itself to a knowledge-based technology solution, the next step in the series of services is *Rapid Prototyping of an Expert System*. Depending on the complexity of the problem, this could take six to nine months. The result is a proof-of-concept system with which the user can decide whether to proceed to a fully fielded system.

If the assessment indicates high risks, but also indicates high value to the user, the Center typically will start what is termed an *Applied Research and Development (R&D)* project to reduce risks before starting a prototyping effort. All of the Center's Neural Network R&D falls in this category.

The last service provided by the Center is *AI Training* to raise the AI literacy both in the Air Force and the local civilian sector. Short courses are tailored for various levels of the Aeronautical Systems Division 11,000 person work force. For instance, a three-hour AI overview is taught for executives and managers, while a four-day course on fundamentals, a five-day hands on course in expert system development, and a two-day hands on neural network workshop are attended by practicing scientists, engineers, computer professionals and first-level supervisors.

In summary, the five services offered by the Center are Applications Screenings, Applications Assessments, Rapid Prototyping, Applied Research & Development, and Training.

The Center's work generally falls in the following application areas of primary interest to the Air Force:

- Manufacturing Technology
- Data Management
- Program Management
- Design Assistance
- Diagnostics

The first two areas, Manufacturing Technology, and Data Management, have been determined, through a survey, to be of primary interest to the Dayton area industry. As the Center expands its services to serve the private sector, these application areas are emphasized.

During the first six months of the Center's existence, seventy applications were screened and approximately half were recommended for assessment or applied R&D in a prioritized order. Currently, the Center is working on fourteen projects, several of which are avionics related.

Examples of some of the avionics work currently being conducted are as follows:

Polynomial Neural Networks for Airborne Applications

This is an applied R&D project to determine the feasibility of using neural networks in a missile evasion scenario. Researchers at Wright State University have determined that a polynomial neural network can out-perform, in speed and accuracy, the conventional look-up table technique in permitting a fighter aircraft to evade hostile air to air missiles. An additional finding associated with this work is that large look-up tables, when needed can be constructed more efficiently by using a neural network approach.

Automated Development of Test Program Sets

This is an applied R&D project to compare and merge the strengths of two different knowledge-based diagnostic systems for avionics. The Ohio State University and Central State University are investigating the feasibility of closely integrating compiled knowledge and deep causal knowledge based approaches to achieve a fast, accurate, robust technique for diagnosing avionics malfunctions.

Neural Networks for Processing Inertial Navigation Unit Data

This multiple phase project conducted by the University of Dayton assessed neural network simulations for maintaining the pointing and tracking accuracy of an optical device detection system whose performance was limited by insufficient Inertial Navigation Unit (INU) data. Neural network simulations were evaluated using F-16 flight test data that sampled INU outputs at higher-than-standard rates. Results indicated that neural network simulations could predict INU data at the required rates and accuracy. Follow-on work will address the development of neural network software simulations that adaptively smooth INU outputs. Dr Stephen Gustafson will describe in more detail this project in his presentation.

Other avionics related projects which are being contemplated include the design of a knowledge-based architecture for an autonomous electrical spacecraft on-board power system, automatic target recognition using hierarchical neural system, and neural networks in conjunction with an analytic model to form an air to air radar target identification system.

There are many other on-going and prospective projects in areas such as manufacturing, program management, and design assistance. Each year new applications are screened and assessed to assure a continuing supply of relevant high priority projects.

The Center has been successful in achieving its mission in that expert systems and neural networks have been applied to a variety of Air Force applications, and the Dayton, Ohio, AI talent base has certainly been expanded. Over 400 people have attended the many short courses offered by the Center and numerous graduate students have participated on the prototyping and applied R&D projects.

Lessons Learned

I would like to give you my observations, call them lessons learned if you will, concerning the use of AI in aerospace electronic systems. I would also like to postulate some lessons learned from the recent war in the Persian Gulf region where AI was used for the first time in a combat situation.

You'll note from my background that I am an aeronautical engineer by education and a technology generalist by practice. AI is not my expertise. I suspect one of the major reasons I was selected for my current position was to ensure the work accomplished by our AI Center remains focused on useful Air Force applications. My people often refer to me as the AI Center's "resident skeptic", which gives me the feeling that I must be doing something right.

And certainly having spent over 30 years in the United States Air Force, one of which was flying combat missions in the Vietnam War, I have formed opinions about our recent experience in the Persian Gulf, especially as it relates to high-tech weaponry.

Given that background, let me make a few observations about AI and its future potential in military aerospace electronics.

The senior leadership of the United States Air Force has been quoted in various publications with lessons learned from the Gulf War. For instance, the Secretary of the United States Air Force has taken the post-war position that we must maintain our technological edge, despite budget cuts. Pertinent to this conference's agenda, he has signaled out special needs such as weapons guidance in bad weather, advanced electronic warfare techniques to disrupt enemy C³, rapid and accurate bomb damage assessments, and mobile missile detection. The allied force air component commander, Lt General Horner wants a faster preparation for the daily air tasking order, which at the peak of the war, amounted to an average of 3,000 sorties per day. Lt General Fornell, the Air Force Systems Command Electronics Systems Division commander states that future high priority programs will address better fusion of data... and the presentation of that information in an easy-to-use form.

Certainly these are important future systems needs, many of which could incorporate AI technology.

Let me add a few to this list. One of the most important, non-perishable lessons from the Gulf War is the importance of excellent training. Future conflicts more than likely will not afford us five months of force build up and in theater training. I believe there is a need for intelligent training techniques to obviate the high operations and maintenance cost of training in aircraft. Although today's state-of-the-art simulators are good, I agree they cannot replace flying time for effective training. These training systems must be improved with "intelligence", to be better alternatives to increasingly scarce flying time.

With the cutting back of aircraft and flying time, we may have to reconcile additional use of unmanned flying vehicles, particularly in reconnaissance. There is much potential for AI in this application, principally in adaptive modification of vehicle performance. Capabilities needed are multi-sensor fusion, automated sensor management, dynamic mission replanning, and automatic target recognition.

In future conflicts, we cannot rely on the large deployment of civilian contractors to support our high-tech weaponry...needlessly putting these civilians in harm's way. Therefore, we must increasingly automate our diagnostics and technical manuals, embedding knowledge-based systems where it makes sense. We must do the latter without creating a need to deploy knowledge engineers to support such systems. In other words, AI systems of the future, especially deployable diagnostic systems, must allow the user to modify or maintain the knowledge base.

And finally, let me next address a pet peeve of mine...system user friendliness. For those of you who have not yet read the 29 April issue of Business Week, I commend it to you. The artificial intelligence technology we are discussing at this conference, despite increasingly successful use in commercial and military operations, still has two strikes against it. First of all, its name, Artificial Intelligence, evokes skepticism in any potential user. I compliment you for using the term Machine Intelligence, but you are fighting a difficult battle in trying to get the term adopted in the United States. I tried unsuccessfully to inject that same term in the vocabulary of the Air Force Systems Command in 1983-1984. The term Artificial Intelligence is firmly embedded in the United States commercial vocabulary. The second obstacle which must be overcome is the false expectations associated with this technology. Of the many technologies I have worked with during my career, this one has been "hyped" the most. The user community has been stung with high development costs and false expectations, although that is changing with respect to expert systems. Given those obstacles, the last thing a developer would want to do is to encumber an AI system with a user unfriendly interface. Americans are starting to rebel against the devil of user unfriendly designs. The Business Week article

states that consumers are being made to feel like technological illiterates...and they don't like it. Simplicity...not complexity...is the true mark of any well designed product. My AI Center learned its lesson in this area when we prototyped an expert system which advises engineering project managers on which of over 1500 possible data items should be requested on research and development procurement actions. Initial user reaction was best characterized as "moderate apathy" because we emphasized the AI portion, which amounted to less than 10% of the total programming effort. It was only after we added a government form printing capability to this prototype that the system was enthusiastically embraced. I suspect that some of the more successful AI systems deployed to the Persian Gulf War will prove to be heavy on user friendliness and light on AI content.

Summary and Conclusion

In summary, the United States Air Force's Center for AI Applications in Dayton, Ohio, is applying machine intelligence to a variety of military aerospace electronics problems, and simultaneously is expanding the Dayton Area AI talent pool, from which Wright Patterson Air Force Base can recruit. The recent war in the Persian Gulf has highlighted many potential applications of AI, and when lessons-learned are published on those AI systems which were deployed, I suspect system user-friendliness will play as important a role as knowledge-based sophistication. Despite United States Defense research and development budget cutbacks, I believe investment will remain robust for intelligent aerospace electronics systems, not only for its potential in preserving the free world's defense aerospace technological edge, but also for its spinoff potential in commercial aerospace applications.

Again, thank you for your invitation to join you at this conference. I am looking forward to hearing your presentations.

COMPONENT PART NOTICE

THIS PAPER IS A COMPONENT PART OF THE FOLLOWING COMPILATION REPORT:

(TITLE): Conference Proceedings of machine Intelligence
for Aerospace Electronic Systems Held in Lisbon, Portugal on
13-16 May 1991 (L'Intelligence Artificielle dans les Systemes
Electroniques Aeroespaciaux)

(SOURCE): Advisory Group for
Aeronautical Research and Development, Neuilly-sur-Seine
(France)

TO ORDER THE COMPLETE COMPILATION REPORT USE AD-N242025

THE COMPONENT PART IS PROVIDED HERE TO ALLOW USERS ACCESS TO INDIVIDUALLY AUTHORED SECTIONS OF PROCEEDINGS, ANNALS, SYMPOSIA, ETC. HOWEVER, THE COMPONENT SHOULD BE CONSIDERED WITHIN THE CONTEXT OF THE OVERALL COMPILATION REPORT AND NOT AS A STAND-ALONE TECHNICAL REPORT.

THE FOLLOWING COMPONENT PART NUMBERS COMPRISE THE COMPILATION REPORT:

AD#: _____ TITLE: _____

AD-P006 3 2 6

DTIC
S D
NOV 18 1991

Not to be distributed.

Accession
NTIS
DTIC
Unannounced
Justification

By _____
Distribution /

Availability

Dist A-1 21

AD-P006 326



**MACHINE INTELLIGENCE FOR
SURVIVABLE COMMUNICATIONS NETWORK MANAGEMENT**

by
Nick P. Kowalchuk
Rome Laboratory (AFSC)
RL/DCLD
Griffiss AFB NY 13441-5700
USA

1. SUMMARY

The development of communications networking technologies that increase the survivability of services provided to military users has been an important goal in Air Force programs over the past several years. This enhanced survivability is a result of work that has focused on two areas: the development of more robust communication equipment, and the development of a management system that coordinates the use of that equipment. This paper will discuss the role of machine intelligence in the design of such a network management system. Emphasis will be placed on the intelligent network management decision making capabilities that are required in a military environment, and the design tradeoffs which must be made in developing a system that can optimize the system-wide use of resources without the need for human intervention.

A current Air Force research effort titled "The Integrated Communications Network Management System", or IMS, will be used as a case study for an overview of the critical issues that comprise this paper.

2. LIST OF SYMBOLS

ICN - Integrated Communications Network
IMS - ICN Management System
ISDN - Integrated Services Digital Network

LAN - Local Area Network
LOS - Line Of Site
PBX - Private Branch Exchange

3. BACKGROUND

The military networking environment imposes a number of constraints upon the way in which specific machine intelligent technologies can be implemented. These constraints include short response times, processing power limitations, limited transmission bandwidth, and the potential loss and destruction of assets. To better understand the context in which machine intelligence is applied to the design of a survivable network management system, it will be helpful to discuss the communications requirements which characterize the Air Force networking environment.

3.1 Requirements

The Air Force is designing its future communications systems to meet the following six fundamental requirements:

- 1) Interoperability - Primarily through international standards
- 2) Integrated Services - From a single control interface
- 3) Security - User and access authentication, and secure services
- 4) Survivability - Intelligent networking algorithms

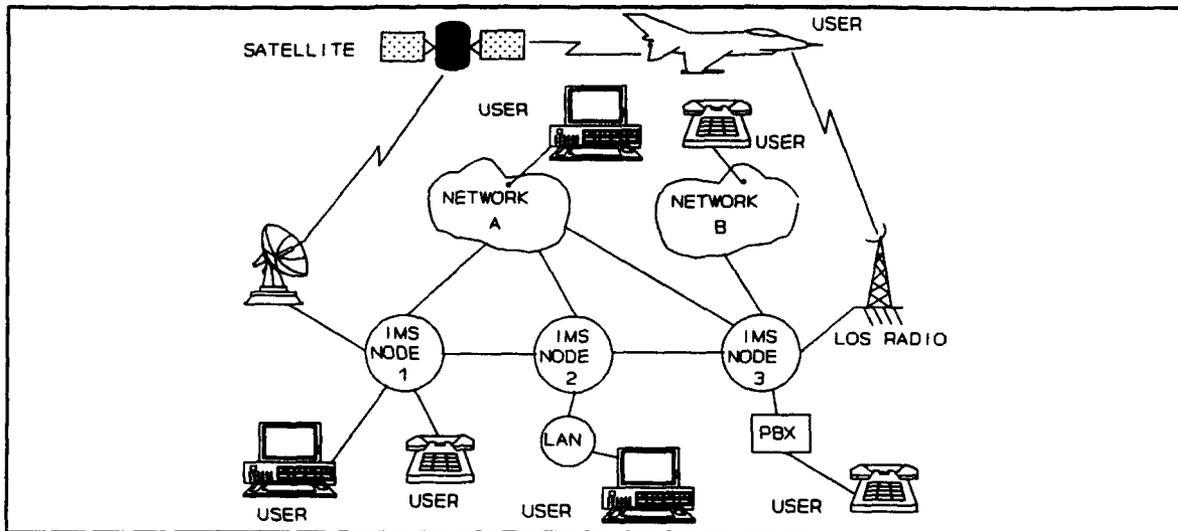


Figure 1 - THE ICN ENVIRONMENT

5) Flexibility - In defining and providing services

6) Capacity - Load balancing of resources

Each of the above requirements has been identified as a deficiency of past or present networks. Rome Laboratory has initiated the IMS program to develop a management system architecture which addresses these deficiencies. Machine intelligence technologies have been targeted to the survivability requirement for the IMS.

3.2 The ICN

The combination of all the transmission resources (communications media, switches, multiplexers, networks, etc.), and all non-IMS management systems that can be monitored and/or controlled by the IMS shall be referred to as the Integrated Communications Network, or ICN. Some of the resources and management systems within the ICN may not be under the control of the IMS, in which case the IMS must treat that resource as a "black box" from which it can request a connection. A simple conceptual illustration of the ICN is provided in Figure 1. The ICN

concept includes a distributed set of nodes that host network management processes for the ICN. These network management processes form the ICN Management System, or IMS. Note that Figure 1 reflects the distributed nature of the IMS. Machine intelligence technologies, in the form of network management decision making algorithms, form the core of the IMS. The remainder of this paper will discuss in detail the nature of these technologies.

4.0 THE IMS

The performance objective of the IMS is to provide the communications network management functions that optimize the use of all surviving transmission resources on a system-wide or "global" level, while providing users with an increased survivability and availability of services during all phases of an attack. The IMS thus provides the reactive capabilities for a communications system to "fight back" against enemy attacks.

4.1 IMS Machine Intelligent Capabilities

Machine intelligence technology in the IMS will enable the automatic

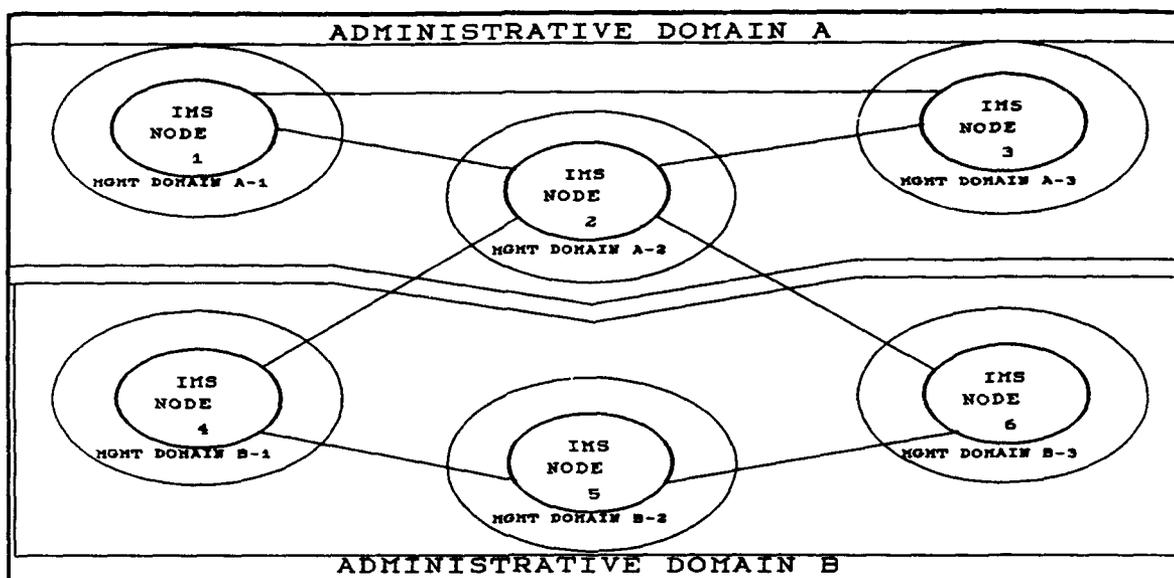


Figure 2 - THE IMS DOMAIN CONCEPT

operation and control of its three major functions; service provisioning, resource provisioning, and service fault resolution. The IMS employs intelligent algorithms to implement these functions, which will be discussed in detail in later sections. These intelligent management algorithms must take courses of action to adapt the provisioning status of the IMS subsystems to dynamically meet the communications needs of its users, particularly in a hostile environment where its resources are subject to damage and destruction. The human administrator/network manager is only required for initial network configuration, establishment of administrative domain, and input of the "master communications plan", which keeps track of subscribers and their accounts and the administrative status of resources in that domain. Keep in mind the fact that the IMS is a distributed management/control architecture, as depicted in Figure 2. Dispersed IMS nodes must communicate with each other and other management entities, possibly across multiple administrative domains, to develop a concurrent knowledge of the "global state" of the network. Information fusion is therefore a key requirement for each of the three IMS

functions at the system level, with a goal of making an "intelligent" management decision that is based upon pieces of information that are sometimes incomplete or inaccurate.

4.1.1 Service Provisioning

The IMS automates the process of satisfying the service requests of communications users in a three stage service provisioning process (See Figure 3). Two of these stages, service mapping and service assignment, rely upon machine intelligence in their design and will be discussed in more detail. The service provisioning function of the IMS is to optimally map all communications service requests to the available transmission resources. An IMS "service" is defined in a manner similar to the ISDN method for characterizing a service. The IMS defines a service in terms of its associated access, information transfer, and general attributes (See Table I).

4.1.1.1 Service Mapping

The first stage of service provisioning is service mapping, where user service requests must first be translated into specific

01 1113 003

91-15508



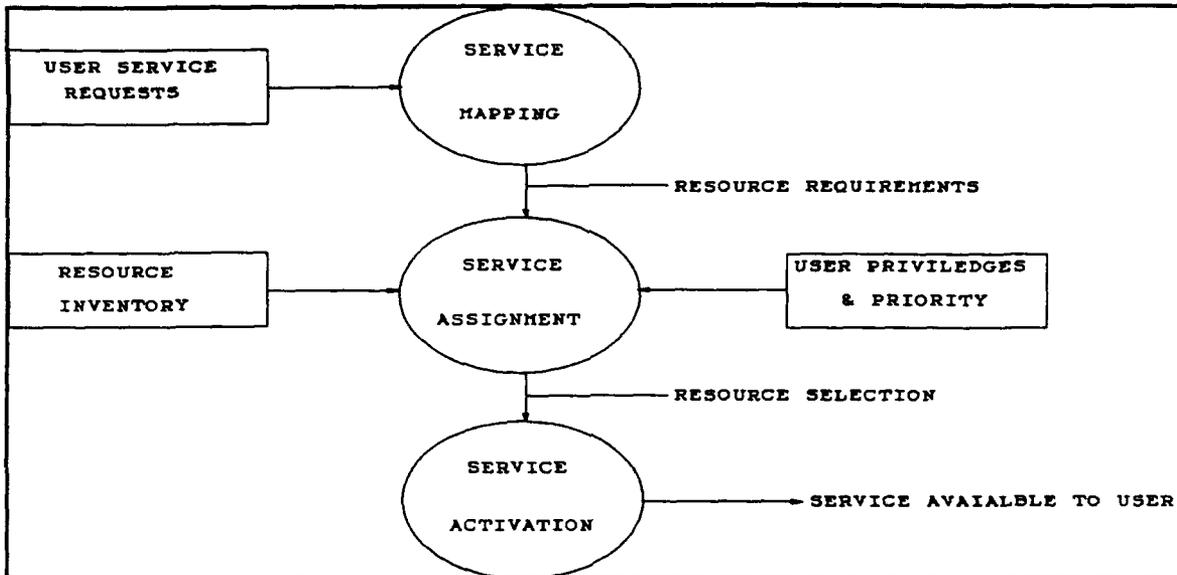


Figure 3 - THE SERVICE PROVISIONING PROCESS

Table I - IMS SERVICE DEFINITION

ACCESS ATTRIBUTES
- User Descriptors (name, password, privileges, priority)
- Service Establishment Mode (demand, reserved, permanent)
- Symmetry (simplex, half duplex, duplex)
- Protocol (TCP/IP, TP-4, X.25, etc.)
INFORMATION TRANSFER ATTRIBUTES
- Information Transfer Rate (bit rate or throughput)
- Information Transfer Type (voice, data, video)
- Configuration (point-to-point, multipoint, broadcast)
GENERAL ATTRIBUTES
Quality Of Service (survivability, availability, reliability)
- Performance (delay, error rate)
- Security Level (Unclassified, confidential, secret, top secret)

Table II - RESOURCE REQUIREMENTS

Service ID
Source User Name
Destination User Name(s)
Priority
Type
Protocol
Status
Start Time
End Time
Survivability Requirements (reconstitution, A.J, LPI)
Symmetry
Maximum Delay
Security Level
Data Rate
Maximum Allowable Error Rate

resource requirements which are needed to support that service. Table II illustrates the type of information that constitutes a resource requirement. The attributes that characterize a service dictate which transmission resources are acceptable. It is the task of the service mapping phase to insure that there is no ambiguity of the nature of the resources that are required to satisfy a given request. These resource requirements are then passed to the service assignment phase.

4.1.1.2 Service Assignment

The next stage of service provisioning is service assignment, where every resource requirement has to be matched with a suitable subset

of the resources that are actually available at that time. The types of resources that can be allocated are connections, links, ports and equipment. Connections in the ICN are defined as end-to-end communications pipelines. Links are the point-to-point segments through which connections are routed. Ports are the access points to connections. Equipment covers the physical communication boxes (transmit and receive) that are allocated to connections, such as modems and multiplexers.

The service assignment process must allocate the connection(s) that are required to satisfy a service request. In the case where no existing connection can satisfy a service requirement, the service assignment process must check existing links to see if resources can be combined to create a new connection that can support that service. This decision is based on topology information which identifies the links and equipment that can be used to provide a service. This resource topology information is provided by the resource provisioning process, which will be discussed in the next section.

Another application for intelligent algorithms in the service assignment process is when there is more than one path in the IMS that can satisfy a service request. In this case, a procedure is invoked that will select the "closest to optimal" set of resources. This selection is based upon the chosen performance metrics for the IMS, from a local versus system-wide perspective (e.g., Do we minimize local congestion, or perform global flow control, or something in-between?). The choice of resources for service assignment is therefore determined using the relative weights of decision criteria such as: maximize system-wide throughput and connectivity, minimize blocking, minimize congestion in regional sectors and over scarce resources at the system level, and perform access

control on unauthorized users. Note that there may be conflicts at the system level which the IMS must resolve when trying to optimize each of the above parameters. The mapping of specific system goals to network metrics and the assignment of relative weights to each metric in a flexible and dynamic manner in response to constantly changing user needs and operational conditions is a key function of the IMS. With this capability, the IMS helps to reduce the degree to which users and system administrators are required to be in the loop to execute the allocation of available resources to user service requests.

4.1.2 Resource Provisioning

The resource provisioning function of the IMS involves the determination of the status and configuration of the transmission assets. After the resources are made available by the resource provisioning process, the service provisioning process controls the utilization of those resources by making decisions in response to each service request. The assignments made by the service provisioning process affect resource utilization and traffic load balance, which demonstrates the strong interdependency between service and resource provisioning.

The resource provisioning process performs two major functions, which are the determination of resource availability and the management of link and equipment configuration to support the establishment of links and connections for service provisioning.

4.1.2.1 Monitor System Topology

The IMS system topology is comprised of the collection of available IMS resources, their current configuration, and information regarding their end-to-end service availability. The system topology information is supplied to the service provisioning process to

facilitate the optimal matching of services to resources. The topology information can be represented by a directed multigraph, where IMS nodes are vertices and communication links are edges. The following paragraph discusses the means by which the resource provisioning process can use this topology information to assist in the decision making processes associated with topology management.

4.1.2.2 Manage Resource Configuration

The resource connectivity and status information gathered by system topology algorithms is used to manage the configuration of resources in support of the resource provisioning function of the IMS. The IMS thus automates the process of reconfiguring and reallocating surviving assets in the cases where existing services have failed, so that the users are shielded from the actions that are required to perform this task. For an added level of protection from potential loss of service, the IMS also monitors cut-set and congestion problems and recommends possible solutions. A cut-set represents a split in the connectivity of a network that creates separate islands of connectivity among the remaining assets.

4.1.3 Performance Analysis For Fault Resolution

Performance analysis is an integral function of the IMS which assists the service and resource provisioning processes, especially when assets are degraded and lost in the course of providing required services. Data concerning link and equipment failures is used to perform trend analysis and localization of fault scenarios, and current or predicted performance problems. These performance problems include the identification of cut-sets and congestion areas discussed in 4.1.2.2.

In executing performance analysis, performance data and resource status information are compared with expected values so that service faults can be identified. Cut-set algorithms are run to detect current and potential future cut-sets. Traffic analysis algorithms are also run, so that transmission congestion points can be located. Finally, all of this analysis data is used to manipulate the service provisioning process so that services can be reassigned to correct or avoid the problems that were identified.

5.0 CONCLUSION

In summary, an IMS node provides a communication user with a survivable, integrated control interface to the services of various networks and transmission resources, while screening that user from the idiosyncracies of accessing the individual systems and from the reconstitution actions that the IMS initiates to restore failed services. The IMS can thus be considered an intelligent "manager of managers", which provides dispersed system-wide network management decision making capabilities to automate the processes of service and resource provisioning and network reconstitution.

The role of machine intelligence technologies in the development of the survivable communications network management system discussed in this paper is centered on the need to reduce the degree to which users and system administrators are required to be in the loop to allocate available global resources to user service requirements, throughout all phases of an attack.

This paper has used the IMS program as a case study to outline some of the potential applications for machine intelligence in the design of survivable network management systems. Work is still continuing in the identification of areas in network management where machine

intelligent functions can be integrated to enhance the survivability of the systems being developed.

6.0 ACKNOWLEDGEMENTS

I would like to thank all the people at Stanford Telecommunications, Inc. who were involved in the performance of the IMS program, for their outstanding contributions to the system design and prototype development for the IMS, and whose work is reflected in this paper.

Discussion

1. Prof. A.N. Ince, Turkey

You have addressed in your talk only the requirements that users may have for a survivable communication network. Can you please comment on what measures you are considering which can benefit from machine intelligence techniques to meet the needs of the users for reliable/continuous/timely service?

Author:

The discussion of requirements at the beginning and end of this presentation was made to emphasize what requirements formed the basis of the IMS program. The incorporation of machine intelligence techniques such as expert systems for trend analysis and fault resolution; the use of information fusion concepts to collect incomplete pieces of management information and make an intelligent decision as to the current "global" state of the network; are good examples of how MI is being applied to this effort. The published paper gives a better description of applications.

2. Dr G.H. Hunt, United Kingdom

I am interested to hear more about the security aspects of the Network Management System. Are there different levels of security protection in the network links and different security levels in the message contents, and does this complicate the network management?

Author:

The IMS handles security in two ways. First, from a user standpoint, only authorized users are allowed to access the IMS at the appropriate level of clearance. The IMS then guarantees the security of that service by selecting only the links which are capable of meeting the security requirements of that service. *Very importantly*, the IMS *does not* itself handle the user data or message contents (other than headers), but rather it communicates with the transmission fabric (switches, multiplexers, etc.) and establishes the links that are capable of supporting the security requirements of that data. Internally, the Management Information Base (MIB) contents of the IMS must be a trusted system because this is where the information regarding where the network resources are, that can support that service.



COMPONENT PART NOTICE

THIS PAPER IS A COMPONENT PART OF THE FOLLOWING COMPILATION REPORT:

(TITLE): Conference Proceedings of machine Intelligence
for Aerospace Electronic Systems Held in Lisbon, Portugal on
13-16 May 1991 (L'Intelligence Artificielle dans les Systemes
Electroniques Aeroespaciaux)

(SOURCE): Advisory Group for
Aeronautical Research and Development, Neuilly-sur-Seine,
(France)

TO ORDER THE COMPLETE COMPILATION REPORT USE AD-A242025

THE COMPONENT PART IS PROVIDED HERE TO ALLOW USERS ACCESS TO INDIVIDUALLY AUTHORED SECTIONS OF PROCEEDINGS, ANNALS, SYMPOSIA, ETC. HOWEVER, THE COMPONENT SHOULD BE CONSIDERED WITHIN THE CONTEXT OF THE OVERALL COMPILATION REPORT AND NOT AS A STAND-ALONE TECHNICAL REPORT.

THE FOLLOWING COMPONENT PART NUMBERS COMPRISE THE COMPILATION REPORT:

ADM: _____ TITLE: _____

AD-P006 327

DTIC
ELECTE
NOV 13 1991
S D D

Accession For	
NBS CR&I	<input checked="" type="checkbox"/>
DTIC TAB	<input type="checkbox"/>
Unannounced	<input type="checkbox"/>
Justification	
By _____	
Distribution/	
Availability Codes	
Dist	Avail and/or Special
A-1	21

AD-P006 327

A DISTRIBUTED ENVIRONMENT FOR TESTING COOPERATING EXPERT SYSTEMS

Capt Jeffrey D. Grimshaw
 Mr. Craig S. Anken
 Air Force Systems Command
 Rome Laboratory/C3CA
 Griffiss AFB, NY USA
 13441-5700

1. SUMMARY

This paper discusses the Advanced Artificial Intelligence Technology Testbed (AAITT) being sponsored by Rome Laboratory. The purpose of this testbed is to provide a powerful environment for integrating dissimilar software systems including expert systems, conventional software, databases, and simulations distributed over a network. In addition, this testbed will provide measurement and analysis tools for evaluating the results of the various user-supplied software components.)

Section 2 of this paper discusses the need for operational software support systems to cooperate with one another and for these systems to be thoroughly tested. Section 3 covers previous attempts at getting decision aids to work together. Section 4 discusses the AAITT in more detail, including its goals and high level design.

The AAITT will be based on the ABE (A Better Environment) module-oriented system and on the Cronus distributed computing environment. This will allow the various components to communicate transparently over a heterogeneous network. Generic database and simulation capabilities will be provided to support the development, integration, and testing of new software components.

2. PREFACE

An intelligent command and control (C2) decision aid often begins its life in a "sterile" laboratory environment. Each decision aid is developed to solve a relatively narrow problem within the overall C2 decision-making/ management problem. To

provide the user guidance, this aid has sources of data and knowledge stored in local format(s), a problem solving methodology which uses the knowledge to reason over the data, and usually the ability to interact with the user. Testing may be performed by presenting the aid with predefined scenarios and comparing the results to some standard. While this approach may be adequate in the laboratory setting, what happens when this tool is brought into the operational realm? That is, when it is brought into an environment where information may be stored in several databases and where multiple decision aids and/or conventional programs are working at various portions of the overall problem. Finally, how reliable will this system be, having been developed in isolation thousands of miles from the battlefield? To answer some of these questions, the Rome Laboratory has embarked on the development of the AAITT, a distributed environment that will support the testing of and cooperation among multiple decision aids. The testbed is being developed by a team of contractors lead by General Electric Aerospace Advanced Technology Laboratories.

Before going further, several terms need to be defined. A component is a stand-alone software program designed to solve part of an overall problem (see Figure 1). User-supplied components (USCs) refer to those components which the application developer supplies and wants connected to the testbed. A module is a component with some type of software "wrapper" that allows it to communicate with other modules. An AAITT application is a collection of modules configured to work interactively to solve an overall problem.

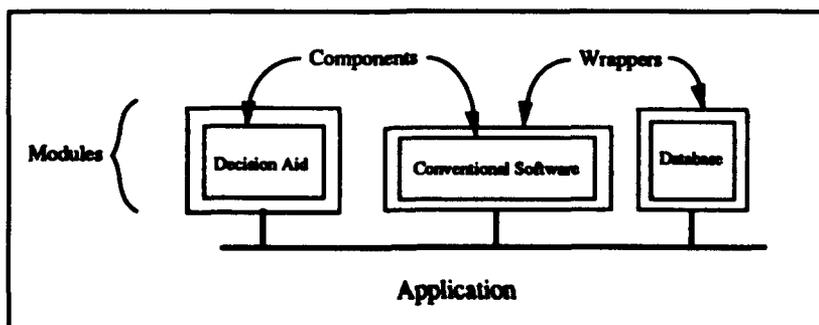


Figure 1. Testbed terminology

2.1 Cooperation between Decision Aids

Many decision aids have been developed in various military planning domains. These include mission schedulers (e.g., KRS [1], TEMPLAR [2], AMPS [3]), route planners (e.g., RPDW [4], HERO [5]), and a variety of databases and simulations, see Figure 2. These systems are usually developed independently with no plan for interacting

with other systems. However, in "real" usage these systems will need to work together to solve much broader planning problems and need to access data located in remote databases. Researchers have often emphasized the investigation of new reasoning techniques applied to simplistic problems over these interoperability issues.

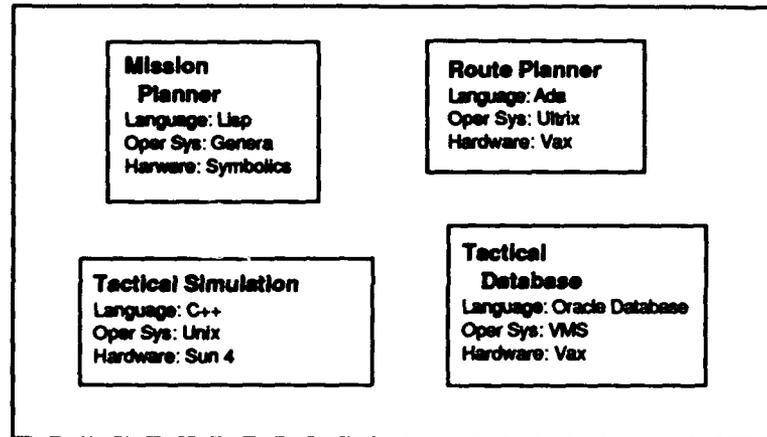


Figure 2. Examples of related software applications developed on dissimilar systems

2.2 Testing/Verification/Validation

Software systems are usually developed by first trying to understand the needs of the end user and then mapping these needs to requirements/specifications. This is the "what the software should do" portion. The

second phase is developing the system by mapping the requirements/specifications to the design/implementation. This is the "how the software will do it" portion. These mapping processes, shown in Figure 3, are usually informal and are thus sources of error.

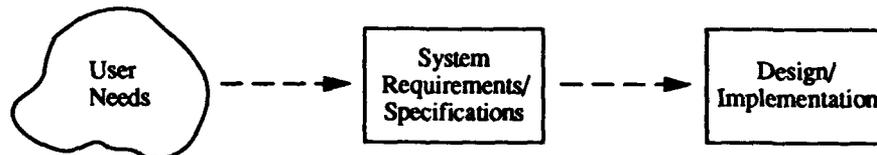


Figure 3. Software Development Process

As systems are developed, the need exists for them to be tested before entering into the operational domain. This is necessary in order to quantify and qualify the errors introduced in the development process. Exhaustive testing is impractical for software systems with even moderate complexity. Theoretical proof testing is likewise

difficult and must be rechecked for even minor modifications. Still, there needs to be some level of confidence that the software works as advertised and as expected. This testing includes verification and validation, shown in Figure 4.

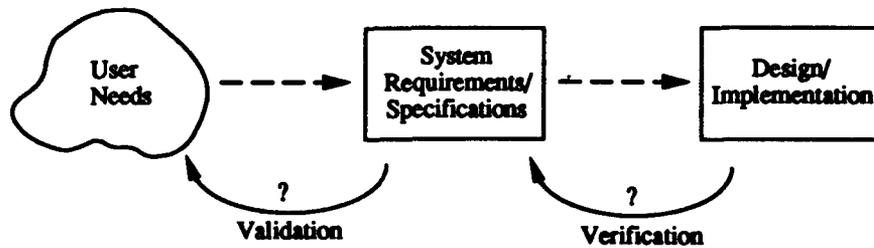


Figure 4. Software Verification and Validation

Verification is concerned with the correctness of the algorithms or reasoning process of the software. For example, "does a sorting algorithm work for negative numbers?" or "does the mission planner select the best available aircraft?" A system that provides a commander with the wrong answer could lead to disaster. Another important part of verification is the need for decision aids to work on large size problems. Often, promising techniques do not scale-up appropriately. For instance the mission planner may work fine for planning X missions on a given computer, but may require too much time or actually crash the machine when attempting to plan 2X missions. Part of verification should be the determination of what size problems can be handled and how gracefully the system degrades.

Validation is concerned with showing that the software satisfies the needs of the user. For example, "does the user really require the absolute best flight path through the threats?" or "did the user want to consider other types of displays for the requested information?" In support of this, there has been more and more interest in the use of rapid prototyping to support iterative development. Often operational users have only a limited understanding of their own needs. When asked to describe their requirements, they must rely on past systems as points of reference. While this is a start, it does not provide the user with an understanding of what could be. With rapid prototyping, a user gets to "play" with the system early in the development process. By actually sitting down and trying different features, a user can begin to describe more accurately his or her actual needs. This is an iterative process as the user discovers what is really needed in a system. This then becomes part of the validation process.

3. PREFACE

While there have been many attempts at decision aid (and conventional software) cooperation, the underlying models used have often been a variation on two main cooperation techniques. These techniques are based on either a decentralized or centralized theory of control and are characterized by the amount of knowledge each decision aid has about the other decision aids versus the degree of central control present in the system.

The decentralized approach provides a more efficient form of distributed processing, however important characteristics such as global data sharing and reconfigurability are

not easily obtained. The centralized approach provides for the sharing of global data through a common data source. While this aids in system reconfigurability and extensibility, resulting bottlenecks can become a problem.

Architectures that can be considered decentralized include the hardwired and broadcast configurations while centralized approaches include the blackboard, information broker, and meta-level controller architectures [6]. The following section looks at these architectures and presents some examples of previous work that have been done in the cooperating expert systems domain. Section 4 describes the software controlled architecture (soft architecture) being used in the Advanced Artificial Intelligence Technology Testbed (AAITT). This approach will allow developers to graphically define the architecture most appropriate for their application.

3.1 Decentralized Techniques

A decentralized system architecture can be viewed as one where the information flow and data exchange does not travel through a common agent (or number of agents). The most popular examples of this type of interaction include the hardwired and broadcast architectures.

3.1.1 Hardwired Architecture

A "hardwired" architecture consists of a number of decision aids cooperating through direct calls to one another. Because of this, each aid on the network is required to know a lot about the other decision aids it will interact with. Typically interaction among the aids is through direct function calls which are embedded into each aid at development time. Since these function calls are "hardwired" into the aids, the architecture is fixed and difficult to modify. If a decision aid is not available on the network its task will not be completed regardless of whether another aid can handle the job. Sharing common data can also present problems in this type of architecture. Issues such as storage management and updates can be very entailed in large hardwired systems where common information is needed by many of the decision aids. Figure 5 shows an example of a simple hardwired configuration where decision aid 1 provides direct input to decision aid 2 and decision aid 3. These two aids then interact in working towards a solution.

91 1113 004

91-15509

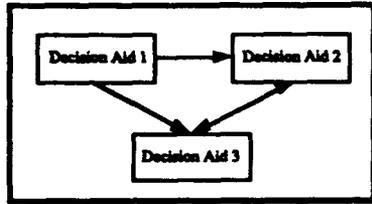



Figure 5. Hardwired Architecture

3.1.1.1 ALLIES

The MITRE ALLIES [7] project attempted to get two existing expert systems to cooperate in constructing a military plan (Figure 6) using a "hardwired" approach. The system included three components: a tactical intelligence aid, ANALYST, which reconstructed enemy force dispositions on a map, an AI planning system, OPLANNER, which handled a large portion of the command and staff responsibilities, and an object-oriented simulation, Battlefield Environment Model (BEM), which modeled and simulated the OPLANNER results.

Because the OPLANNER and ANALYST were written in the same language and ran on the same machine it was relatively easy to get these systems to cooperate. Network communication was not necessary since the two systems were hosted on the same machine. Because communication was established through extensive changes to both the OPLANNER and ANALYST software, eventually the ANALYST became more or less a subprocess of the OPLANNER. Communication to BEM was obtained through simple file transfer among machines. Overall, the above techniques did produce efficient data sharing and a minimum of communication, however extensive knowledge of the other systems had to be "hardwired" into each system. While this guarantees a system will have the data it requires to accomplish its task, problems could arise when the system becomes very complex or more decision aids must be added to the architecture [6].

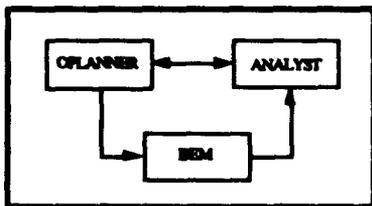


Figure 6. ALLIES

3.1.1.2 Cooperating Expert Systems (COPEs)

The COPEs project [6] studied the problem of coordinating cooperation between existing, independently developed systems. After doing an analysis of past research, Advanced Decision Systems proposed a software solution which would combine aspects of other approaches to provide an architecture for the integration of three existing decision aids. Minimal changes on existing decision aids

would be required. The decision aids to be used included the Automated C3CM Battle Management Decision Aid (ACBMDA), the Tactical Expert Mission Planner (TEMPLAR), and the FORCES simulation [6].

The main concept behind the COPEs design was to provide each decision aid with its own customized, knowledge-based "wrapper." This wrapper would serve as an interface to the other decision aids and would possess knowledge about its own inputs, outputs, goals, and other requirements. The initial plan was to also include knowledge about other specific decision aids and how to cooperate with them using these wrappers. Some meta-level knowledge would also be used for cooperation of decision aids that were not customized explicitly. In the long term, specific decision aid knowledge was to be extracted from the wrappers and inserted into a shared COPEs knowledge base. The wrapper concept underlying COPEs provides a more modular design and less modification of existing decision aids than the "hardwired" design of ALLIES. Still, information about other decision aids must be coded into each new decision aid wrapper. Since each wrapper is "customized", a new one must be developed for each aid, regardless of whether it's of the same type (e.g., planner) as a previous aid in the system. Since each wrapper has a lot of information about other decision aids, it may be difficult to add or replace decision aids. The long term design plans of using meta-level knowledge and a COPEs knowledge base could help ease this problem. Figure 7 provides an overview of the COPEs wrapper concept.

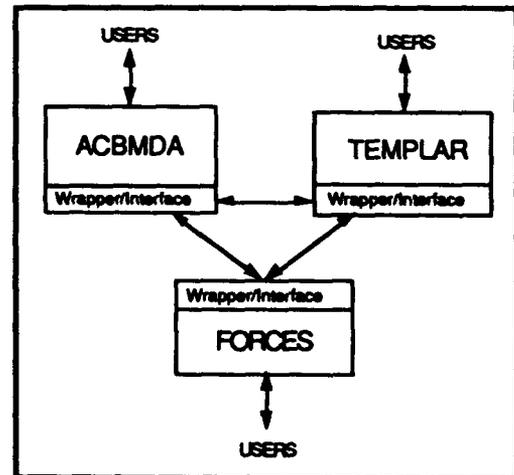


Figure 7. COPEs Architecture

3.1.2 Broadcast Architecture

The broadcast architecture differs from the hardwired model in the way data is represented and decision aid communication takes place. In the broadcast model all decision aids must understand a common data representation. The aids must distinguish what type of information requests they can

handle since all decision aid output is broadcast over the entire network of systems. Any decision aid that can service a particular request will do so in this type of architecture. Figure 8 shows a typical broadcast architecture where three decision aids interact through broadcasted message traffic.

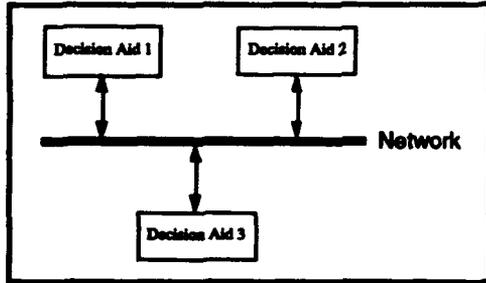


Figure 8. Broadcast Architecture

3.2 Centralized Techniques

A centralized architecture is based on the concept of some type of controlling or common agent through which all information and data exchange takes place. Centralized techniques include the blackboard, information broker, and meta-level controller architectures.

3.2.1 Blackboard Architecture

The blackboard architecture [8] is based on the idea of a common workplace (or blackboard) where a problem can be incrementally solved. Each decision aid is responsible for a specific portion of an overall problem and is allowed access to the "blackboard" when the appropriate partial solution (or data) is made available by the other agents. Some sort of controlling mechanism is needed to monitor the blackboard and determine which decision aid should execute next. In this type of configuration, decision aids do not need to know information about the other knowledge sources involved in the process (see Figure 9).

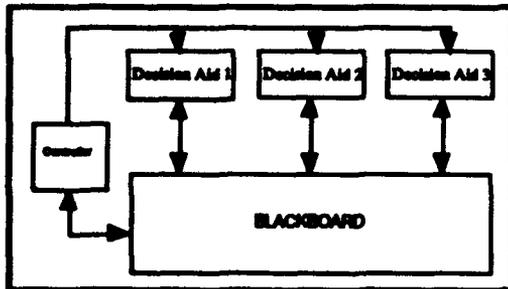


Figure 9. Standard Blackboard Architecture

3.2.2 Information Broker Architecture

The idea behind the information broker [6] concept is to have various "broker" agents responsible for specific information exchange. Each broker is responsible for a specific type of message request or exchange that a decision aid may make. The decision aids interact directly with the brokers that handle their input and output needs and do not require information about the other decision aids on the network. They do however need to have knowledge about the brokers that can handle their requests. Representation translations must also be done between the brokers and the various decision aids. Figure 10 shows a basic information broker configuration where all the decision aids have message interaction with two "brokers."

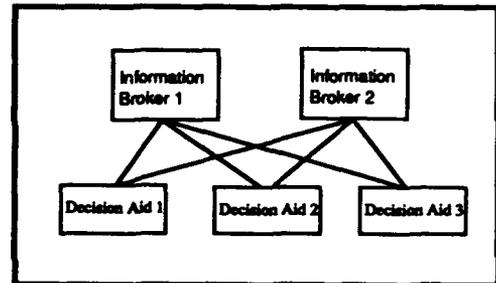


Figure 10. Information Broker

A variation on this architecture would be to broadcast decision aid requests to all of the brokers, having the appropriate broker handle each message (see Figure 11). This would replace the "hardwired" connections from decision aid to broker, becoming somewhat similar to the broadcast architecture described in Section 3.1.2. The difference would be the existence of a (or possible many) centralized controllers rather than direct contact between the decision aids.

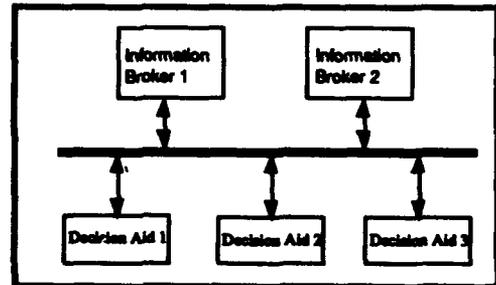


Figure 11. Broadcast Information Broker Architecture

3.2.3 Meta-Level Controller Architecture

The meta-level controller architecture (Figure 12) is based on controller objects which are responsible for message traffic among the decision aids in the system. A controller

object (sometimes called a wrapper) is usually responsible for one decision aid, but sometimes handles several of them. This differs from the broker paradigm in that a controller would handle a specific type of decision aid whereas brokers handle specific types of information, regardless of what aid it is generated from. The long term plans for

the COPES project described in Section 3.1.5 was to provide meta-level controllers to each decision aid in the system. In the COPES project these controllers (wrappers) would each be responsible for one decision aid only and would possess knowledge concerning cooperation rules among each other.

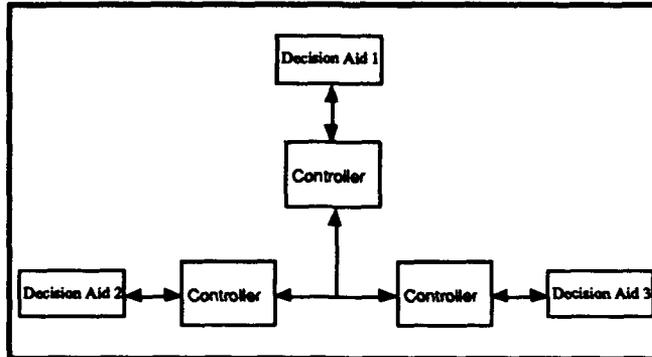


Figure 12. Meta-Level Controller Architecture

3.2.4 TAC-2: A Testbed for Integrating Decision Aids

TAC-2 [9] was designed as a demonstration of the concept of integrating AI-based Command and Control decision aids. The work was based on TAC-1 [10], an effort to develop a framework for knowledge-based system cooperation, and leveraged off of the lessons learned in the ALLIES project described above. The framework developed through the TAC-1 work was called the Knowledge-Based Battle Management Shell (KB-BATMAN shell) and was demonstrated in the TAC-2 effort through its use in the integration of three knowledge-based decision aids. The integrated aids included an Air Force mission planner, an intelligence analyst, and a simulator. The KB-BATMAN shell provided a common database and "intelligent" message router object through which the decision aids could repetitively cycle through the C2 process of intelligence analysis, planning, and simulated execution. The Figure 13 below shows the KB-BATMAN architecture. All information and data exchange is passed through a common database (blackboard) with a router component serving as the intelligence behind database access and control.

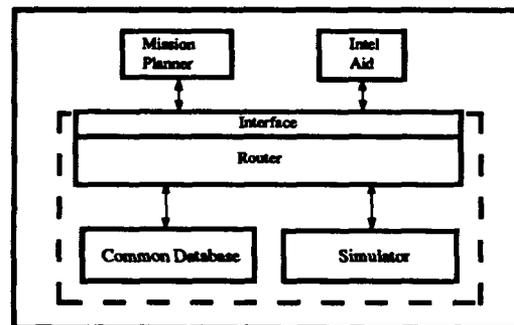


Figure 13. KB BATMAN Architecture

3.3 Common Problems

In some form, the efforts described above all contain concepts and ideas from the different architectures mentioned throughout Section 3. For each project, the architecture was chosen to provide the most efficient and economical solution to the specific problem at hand. While all have their advantages, there are also disadvantages inherent in each design. Problems such as extensibility, modularity, scalability, and bottlenecks are areas where each of these models, at some point, begins to fail. In addition, the problems of testing and evaluation of decision aids (discussed in Section 2.2) are not addressed by these architectures. The following section describes work being done to develop a "soft architecture" which will allow an application to be configured and reconfigured until a desired architecture is found. The system will also provide metrics for evaluation of the various aids, assisting the user in areas such as design, analysis, and integration as well as providing a flexible medium for cooperation.

4. AAITT

The goal of the Rome Laboratory Advanced AI Technology Testbed is to develop a decision aid/conventional software integration environment which will allow the user to:

1. Easily configure various application suites by providing tools to add or remove user-supplied components, or to modify the communication paths of the various problem solving modules,
2. Provide generic database and simulation modules that can be tailored to the needs of the current application,
3. Observe and trace these modules' actions and interactions,
4. Select, gather, and review metrics and statistics on run-time performance, and

5. Rapidly change the flavor of the interactions among the suite's components based upon the results of previous runs.

4.1 Soft Architecture

The components of the AAITT are shown in Figure 14. They consist of the testbed manager (later described as the MCM Workstation), a database, a simulator, and various USCs. The grayed sections represent interface software. A software controlled architecture (soft architecture) has been chosen for the AAITT. A soft architecture is one that can be graphically configured and modified by a testbed manager, while minimizing the recoding of the interface software by hand. An example configuration is shown in Figure 15. To try a different communication architecture (e.g., a blackboard approach instead of data-flow), the user would make the necessary changes using the graphical interface provided by the testbed manager.

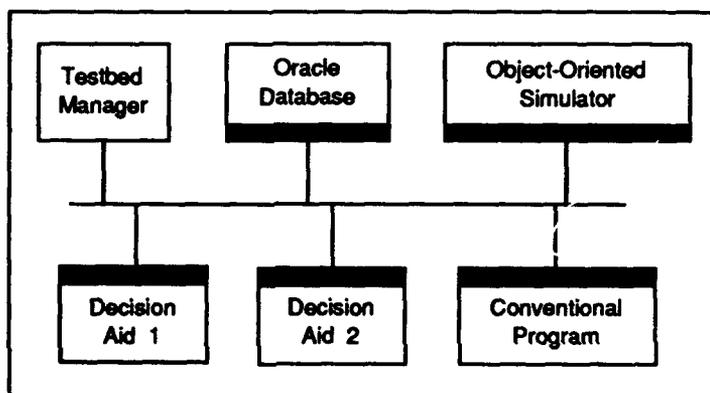


Figure 14. Network View of an AAITT Application

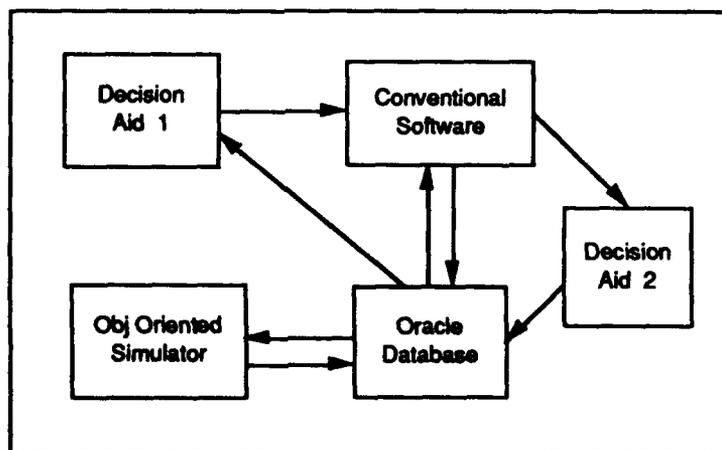


Figure 15. An Experimental Architecture of an AAITT Application

To support the soft architecture envisioned by the AAITT, three major sub-systems are being developed. These are the Distributed

Processing Substrate (DPS), the Module Framework, and the Modeling, Control, and Monitoring (MCM) Workstation.

4.1.1 Distributed Processing Substrate

The DPS will allow dissimilar software systems (e.g., databases, simulations, expert-systems, and conventional software) running on heterogeneous hardware platforms (e.g., VAX, SUN, and Symbolics) to interact with one another. An important feature of the DPS is that it will translate data representations between the various hardware systems and programming languages. Several candidates were considered by the contractor team for providing the foundation of the testbed and are shown in Table 1. The criteria for selection was:

1. Runs on Sun 3/4 and Symbolics 36xx.
2. Runs on Vaxs running VMS and ULTRIX.
3. Runs with applications written in C/C++ and Common Lisp.
4. Runs with applications written in Ada.
5. Provides interface specification and stub code generators.

6. Provides application building tools.
7. Stable, mature product.

The resulting evaluation of each candidate is shown in Table 2. Based upon this analysis, the Cronus distributed computing environment [11] was selected. Cronus provides heterogeneous host support for distributed application development. It gives the user an object-oriented view of resources on a network. In addition to this object-oriented view provided by Cronus, the DPS will use ABE [12], providing a module-oriented programming capability. This will allow the structuring of a distributed testbed application at a higher level than Cronus. The ABE environment supports modules which are independent entities that communicate data and control among themselves through very well-defined interfaces. The goal is to use the higher-level, computational and architectural model provided by ABE with the lower-level, distributed system environment support provided by Cronus.

Table 1. Candidate Testbed Message Passing Environments

Product	Developer
Alpha	Carnegie Mellon Univ.
Cronus	BBN Corp.
ISIS/Meta	Cornell University
Mach	Carnegie Mellon Univ.
MetaCourier	Symbiotics, Inc.
Star O/S	ADS Corp.

[13]

Table 2. Candidate Environment Evaluation

Product	Sun3/4 Symbolics	Vax VMS Ulrix	C/C++ Common Lisp	Ada Interface	Interface Spec & Stub Code Generators	Application Building Tools	Stability
Alpha						None	Poor
Cronus	YES	YES	YES	future	YES	Excellent	Excellent
ISIS/Meta						Good	Excellent
Mach				YES	YES	Good	Excellent
MetaCourier	YES	*	YES		YES	Good	Fair
Star O/S	YES		YES			None	Excellent

* unknown [13]

4.1.2 Module Framework

The Module Framework will allow the user to easily embed a new component in the AAITT. This framework is used to create Component Interface Managers (CIMs). A CIM is a wrapper that provides the interface between the distributed testbed and each component. The

Module Framework will facilitate the creation of the CIM by guiding the user through the creation process and by providing a library of generic CIMs. The CIM is very similar to the wrapper concept used in COPES. One difference is that the AAITT provides a semi-automated generator for easily creating and modifying CIMs, whereas the COPES wrappers are coded by

hand. Another difference is that CIMs allow the testbed to control module loading, initialization, resetting, etc.

There are several types of communication that will occur at a given CIM. These types include simple reception and transmission of data, and may include more complicated transmissions which then wait for a response. When transmitting, the CIM must know which module to send the message to. The name of the destination module will be set when the user completes the modeling phase at the MCM Workstation. The recipient can easily be changed by making the appropriate changes at the workstation. (Note: the actual location on the network of the receiving module will be maintained, transparently to the user, by the DPS.) CIM to CIM message types include ASCII text, integer, real, and database query response.

4.1.3 Modeling, Control, and Monitoring Workstation

The MCM workstation will provide a central user console for building and configuring applications, and for the display and analysis of performance metrics gathered during application runs. The modeling functions will allow a user to specify graphically how the modules are to interact with one another during execution. This will define the architecture of the application. The control functions will allow the user to load, initialize, execute, and reset the components distributed over the testbed's network. Break point capabilities will aid in the debugging process. The monitoring functions will allow the user to specify measurements, monitors, and instrumentation. Measurements refer to the quantifiable features that help the user understand system performance. Monitors are the procedures through which measurements are captured. Instrumentation allows the user to process the resulting measurements and present them in an appropriate manner. The monitoring capabilities of the MCM Workstation will be very important in testing out the USCs and set the AAITT apart from testbeds that simply try to connect existing systems.

4.2 Testing

As stated in Section 2.2, it is important to understand the strengths and weaknesses of the USCs before providing them to operational units. These weaknesses may include actual errors, problem size limitations, excessive run times, incompatibilities, and so on. The AAITT will have features to address both the verification and validation problems addressed in Section 2.2

4.2.1 Verification - Is the software doing the job right?

The end users must have some confidence that the USCs will work as advertised. To verify the results of the USCs, the AAITT will provide metrics for evaluation and heuristics.

4.2.1.1 Metrics

Metrics will be provided by the MCM Workstation as discussed in Section 4.1.3. Timing and memory limitations will be addressed by these metrics. In addition,

bottlenecks will be identified, possibly indicating that a new application architecture should be selected or that faster computers/networks should be used. Higher level metrics will also be addressed, for instance the number of mission routes planned per minute, or the speed at which the simulation simulates one hour of time.

4.2.1.2 Evaluating a solution heuristically

The quality of a solution cannot be determined by the speed of the inference engine or the number of queries handled by the database per minute. Quality can sometimes be assessed by using some type of heuristic. To support this, the AAITT will need some type of generic heuristic evaluation module that can be tailored to evaluate a given class of USCs. For instance, a route planner heuristic evaluation module may take the planned routes and determine for each route if the specified aircraft can fly at the chosen altitudes, if the aircraft has enough range, given the available on-board fuel, and whether the legs of the route are too short (difficult to follow) or too long (vulnerable to enemy attack). This evaluation module could be used regardless of which specific route planner was being tested.

4.2.2 Validation - Is the software doing the right job?

The end users need to make sure the USCs really address their particular problems. To validate whether or not the USCs are meeting their needs, the testbed will provide rapid prototyping facilities and the benefit of graphical simulations.

4.2.2.1 Prototyping environment

The testbed will allow component developers the ability to quickly demonstrate their systems in a realistic environment. By providing both database and simulation capabilities, developers can concentrate on software algorithms and user interface issues, instead of developing sets of problems and ways to evaluate results. This will mean that USC users can be brought in earlier in the development process, allowing them to direct or redirect the developers before design decisions are made. This process will help developers better understand the users' needs, not just the users' stated requirements.

4.2.2.2 Graphical simulation

The simulation component will provide a simulation language and various tools for developing graphical simulations. This will allow users to monitor the execution of plans developed by the USCs. The simulation language is designed to be interactive so that user developed simulations can be stopped, queried, modified, and continued at any point. This type of interactive simulation should help in determining whether the USC developer has really addressed the user's needs.

4.3 Components

The AAITT will have generic components to facilitate the development and testing of USCs. These generic components have been used

to create demonstration components to plan missions in the tactical air combat domain.

4.3.1 Generic Components

Conventional and "expert" software work by understanding the problem at hand, automatically or semi-automatically attempting to solve the problem, and then posting the results (usually to a screen or file). Databases and simulations can provide a supporting environment for evaluating these types of software systems. A database can be used to present the USCs with specific problem scenarios. Once the USCs finish processing, the results of the components can be stored in the database for later evaluation. Heuristics can be helpful in evaluating the quality of a result, for instance the routes chosen by a route planner could be evaluated on the percentage of time that allied aircraft are exposed to enemy air defenses. If a heuristic is not available, a simulation could be helpful, allowing a human evaluator to see the results of following the recommendations of a particular USC. Generic components will be provided to develop these types of databases and simulations.

4.3.1.1 Database

The testbed will have a generic database for storing and retrieving information needed by the USCs. Results of the USCs can be stored in the database for later evaluation. The Oracle relational database has been chosen as the testbed's database shell. Using the DPS, the database will appear as a module in the testbed that can respond to structured query language (SQL) statements. As additional databases are created, they can then be used by other USCs.

4.3.1.2 Simulation

The testbed will have a generic simulation capability to show the consequences of following the results of the other modules. The ERIC object-oriented simulation language has been chosen as the testbed's simulation language [14]. ERIC is based on the Common Lisp Object System (CLOS) and allows the simulation developer to develop a hierarchy of object classes with associated behaviors and attributes. ERIC allows objects to be created dynamically and for the simulation to be halted, modified, and continued without recompiling. The message parser provided by ERIC allows for expressive message passing. Results from the simulation can be posted to the database or sent on to USCs.

4.3.2 Demonstration Components

Demonstration components will be used during the AAITT demonstrations to show the feasibility of the testbed.

4.3.2.1 TAC-DB

The Tactical Database (TAC-DB) provides a realistic, though unclassified, laydown of tactical units and equipment in the central European theater [15]. This database was developed by Knowledge Systems Corporation (KSC) in 1989 and reflects the NATO and Warsaw Pact capabilities at that time. In addition to identifying where units are located, the

database contains information on individual weapon systems. As new USCs are added to the testbed with new requirements for domain information, TAC-DB will be extended to provide the necessary support.

TAC-DB provides the "blue-view" of the world for the USCs. This means that information in the database is only as accurate as blue intelligence is. This makes sense since USCs should not be able to access more information than the blue side knows. (Note: there may be some "red-view" information stored in the database used only by the simulator to create red units, etc., that are currently unknown to the blue side.)

4.3.2.2 LACE

The Land Air Combat in ERIC (LACE) simulation simulates the tactical engagement of NATO and Warsaw Pact forces in the central European theater [16]. The simulation reads in friendly and enemy unit information from TAC-DB, creates these objects, and then executes air tasking orders (ATOs) also stored in the database. Friendly air missions must penetrate through enemy air defenses in order to attack targets. As missions return, their results are posted to TAC-DB for use by the USCs.

LACE contains the ground truth for the scenario. The USCs do not have direct access to this ground truth data, only to the TAC-DB. The USCs could, however, be used to schedule LACE reconnaissance missions which will post their intelligence reports to TAC-DB upon completion. This information then becomes available to the other modules of the testbed.

4.3.2.3 Decision Aids

Two decision aids will be included as part of early demonstrations. These include the Air Force Mission Planning System (AMPS) developed by The MITRE Corporation and portions of the Route Planner Development Workstation (RPDW) developed by the Jet Propulsion Laboratory. AMPS [3] was developed to support the planning and replanning of ATOs. ATOs are the plans used by NATO to conduct air missions. AMPS contains domain knowledge concerning aircraft and weapon capabilities, weapon system availability, and scheduling constraints. In addition to planning these missions, AMPS was developed to intelligently support replanning, the process of fixing a plan which has problems, without replanning the whole ATO. The RPDW [4] was designed to facilitate the development and evaluation of route planning algorithms. Our route planner is one of the algorithms provided by the RPDW. To use this planner, the user must first develop a threat-contour map based on the location and line of sight of the various air defense systems. The planner then determines a route through the map while attempting to minimize aircraft vulnerability. The results generated by AMPS and the route planner will be sent to TAC-DB and then on to the simulation. The idea is to provide feedback to the decision aid developers concerning their systems.

5.0 Conclusion

C2 decision aids and conventional programs are being developed to support operational commanders. An environment is needed to allow these dissimilar systems to easily work together. In addition, realistic testing of these components is required to make sure they perform as expected. The AAITT is designed to address these concerns by providing a loosely coupled, distributed support environment, allowing independently developed aids to cooperate. The generic database and simulation capabilities will allow for easy development and evaluation of C2 decision support systems. If the database is large enough and the simulation can provide updates at fast enough rates, the testbed can present the decision aids with realistic wartime environments. This is the kind of interoperability and evaluation that is needed before fielding composite operational systems.

6.0 Bibliography

1. Walter, S., Benner, K., Anken, C., "Knowledge-Based Replanning Applied to Coordinated-Service Mission Planning," RADC-TR-87-173, September 1987.
2. TRW Defense Systems Group, "Tactical Expert Mission Planner (TEMPLAR)," RADC-TR-89-328, January 1990.
3. Dawson, B., Day, D., Mulvehill, A., 90 "The AMPS," RADC-TR-90-131, July 1990.
4. Cameron, J., Cooper, B., Mishkin, A., "Route Planner Development Workstation, Volume 2 - Software User's Guide," JPL D-2733 Vol 2, U.S. Army Engineer Topographic Laboratories, Jet Propulsion Laboratories, November 1985.
5. PAR Government Systems Corporation, "Draft Functional Description of Heuristic Route Optimization (HERO)," F30602-88-C-0118, May 1990.
6. Grover, M., Kinion, P., Cromarty, A., Edwards, T., "Cooperating Expert Systems (COPEs)," RADC-TR-90-104, June 1990.
7. Benoit, J., Bonasso, P., Davidson, J., Ellenbogen, J., Laskowski, S., and Wong, R., "Airland Loosely Integrated Expert Systems: The ALLIES Project," J. Technical Report MTR-86W00041, MITRE, April 1986.
8. Nil, P., "Blackboard Systems: The Blackboard Model of Problem Solving and the Evolution of Blackboard Architectures," AI Magazine, Summer 1986.
9. Nugent, R., Skeen, D., Tucker, R., "TAC-2: A Testbed For Integrating Cooperative Knowledge-Based Decision Aids," RADC-TR-89-74, July 1989.

10. Morawski, P., Nugent, R., Tucker, R., "TAC-1: A Knowledge-Based Air Force Tactical Battle Management Testbed," Technical Report MTR-87W00171, MITRE, September 1987.
11. BBN Systems and Technologies Corp., "Cronus Advanced Development Model," RADC-TR-89-151 Vol. 1, September 1989.
12. Erman, L., Davidson, J., Lark, J., Hayes-Roth, F., "ABE: Final Technical Report," TTR-ISE-90-104, May 1990.
13. GE/ATL, "Advanced AI Technology Testbed, Technical Information Report (DPS) Analysis Draft," F30602-90-C-0079, February 1991.
14. Hilton, M. and Grimshaw, J. "ERIC Manual," RADC-TR-90-84, April 1990.
15. Kearney, H., Lazzara, A., Pendergast, J., Richer, A., Erich, R., "Cooperative Red and Blue Database System," RADC-TR-90-98, June 1990.
16. Anken, C.S., "LACE: Land Air Combat in ERIC," RADC-TR-89-219, October 1989.

Discussion

1. J. Bart, United States

Explain what you meant by working the "transportation problem?" What is your organization doing in this area?

Author:

DART, Resource planning, etc. The Rome Laboratory Knowledge Engineering Branch (RL/COES, soon to be RL/C3CA) is working with United States TRANSCOM to apply AI technology to the cargo/transportation palling problem. To address this problem we are investigating case based, constraint-based, and fuzzy-logic reasoning, among others. DART is an early prototype to address part of this problem. It is a somewhat intelligent front end to a database containing a planning scenario. Since DART uses Cronus and an Oracle database, it should be easy to integrate into the AAITT.

2. Charles Krueger, United States

Do system tests "prove" system compliance under all conditions?

Author:

No. Proof must come from analysis of component parts of system.



COMPONENT PART NOTICE

THIS PAPER IS A COMPONENT PART OF THE FOLLOWING COMPILATION REPORT:

(TITLE): Conference Proceedings of Machine Intelligence
for Aerospace Electronic Systems Held in Lisbon, Portugal on
13-16 May 1991 (L'Intelligence Artificielle dans les Systemes
Electroniques Aero spatiaux)

(SOURCE): Advisory Group for
Aeronautical Research and Development, Neuilly-sur-Seine
(France)

TO ORDER THE COMPLETE COMPILATION REPORT USE AD-A242025

THE COMPONENT PART IS PROVIDED HERE TO ALLOW USERS ACCESS TO INDIVIDUALLY AUTHORED SECTIONS OF PROCEEDINGS, ANNALS, SYMPOSIA, ETC. HOWEVER, THE COMPONENT SHOULD BE CONSIDERED WITHIN THE CONTEXT OF THE OVERALL COMPILATION REPORT AND NOT AS A STAND-ALONE TECHNICAL REPORT.

THE FOLLOWING COMPONENT PART NUMBERS COMPRISE THE COMPILATION REPORT:

AD#: _____ TITLE: _____

AD-P006 328

DTIC
ELECTE
NOV 13 1991
S D

Accession For	
NTIS GRA&I	<input checked="" type="checkbox"/>
DAC TAB	<input type="checkbox"/>
Unannounced	<input type="checkbox"/>
Justification	
By _____	
Distribution /	
Availability Codes	
Dist	Avail and/or Special
A-1	21

This document has been approved for public release and its distribution is unlimited.

AD-P006 328

HEURISTIC ROUTE OPTIMIZATION
A Model for Force Level Route Planning

Lt Janet L. Barboza
Rome Laboratory, COAA
Griffiss Air Force Base, NY 13441-5700
USA

1. SUMMARY

A major shortcoming of tools and methods currently employed for route planning is that they do not incorporate force-level factors, or, if they do, the representation is inadequate. Many important factors necessary for approaching the best possible route are ignored. Heuristic Route Optimization, or HERO, an exploratory development project of the Advanced Concepts Branch of Rome Laboratory, is a model for automated route generation for force-level mission planning. Object Oriented techniques and a dynamic threat representation allow detailed analysis of multiple planning variables in producing effective, survivable mission plans.

2. INTRODUCTION

The threat environment is the greatest obstacle to mission accomplishment; the likelihood of survival is diminished by enemy defenses. Threat avoidance is often impossible due to such basic constraints as fuel supply and range limitations. Thus, effective mission planning, which takes into account the threat environment through which missions will be directed, is essential to mission accomplishment.

Current planning methods plan single sortie paths that fail to incorporate much of the complexity that affects the quality, and therefore survivability, of the mission or missions flown. The human planner most likely will be unable to effectively analyze many variables without the aid of automated tools. Not only is he constrained by time and the limitations of his own knowledge, many parameters may not even be at his disposal for evaluation.

Automation increases the number of variables that can be analyzed for mission planning. Yet automated tools, including those that use dynamic programming to minimize the sum of the lethalties along a proposed path, do not adequately represent the threat environment. They are not flexible enough to do so, and therefore are limited as well.

Object-oriented design and implementation provides a vehicle to further enhance the automation of mission planning functions. By using object-oriented techniques, the parameters essential for optimal mission planning, such as airframe configuration and the threat environment, can be represented as "objects." The interaction of objects allows the program to be flexible and dynamic. The HERO program demonstrates the advantage of using object-oriented techniques in automated route planning.

3. FORCE LEVEL APPROACH

Existing tools and current methods for route planning are narrow in their scope, usually planning for a single penetrator alone in a static threat environment. In reality, however, it is very unlikely that a single penetrator will fly over sterile airspace. Others will have gone before, or will come after. The threats will not always react in exactly the same way, depending instead on conditions at a given time. To bring the flight of a single penetrator to the reality of the air battle requires a deeper and more expansive evaluation of the interaction of penetrators and threats.

Ideally, all the parameters affecting the route would be evaluated in mission planning. Type and number of penetrators, connectivity of the threat network, availability of countermeasures, environment variables, speed and altitude are just a few of the many planning variables necessary to employ force-level tactics and strategy to produce a more efficient, more effective, overall plan, in addition to creating plans for individual sorties. An object-oriented approach to the representation of penetrators, threats, and the environment allows detailed analysis and manipulation of such variables in order to not only produce realistic routes, but to suggest the best employment of tactics such as electronic warfare and saturation.

4. THREAT ENVIRONMENT

Whether planning a single sortie or multiple sorties, a planning tool must adequately represent the threat. Speed, altitude, electronic warfare, terrain, and on-board countermeasures are factors that can be used to lessen vulnerability to threats. An automated tool should represent the threat behavior, which can be tremendously complex, according to such factors.

In representing threat behavior, the connectivity, or dependency, of the threats is an important issue, even if the only connectivity is that the penetrator will have to fly over the defended area once again during egress. Even when planning a single sortie, the element of surprise is removed as soon as the first radar detects the presence of the penetrator. When threats are networked together, threat lethality becomes a function of threats already encountered. When a penetrator is detected by a threat, that threat is alerted to other penetrators that may come along. Connected threats will also be alerted. The result is increased lethality as the penetrator encounters alerted threats. The state of alert of the threat affects the

lethality of a discontinuous path, where exposure to a particular threat or threat network is intermittent, as well. Because the exposure is intermittent, a discontinuous flight path may actually have a lower lethality than continuous exposure over the same area of lethality. The second and successive exposures will encounter an increased state of alert, but the threat's readiness may have changed. For instance, there is a time required to acquire and fire at the penetrator. A continuous exposure may give the threat enough time to react, while intermittent exposure may not afford the threat the opportunity. A realistic representation of the threat environment, then, should take into account the history of penetrator exposure to the threat environment.

5. FLAPS THREAT ENVIRONMENT

The Force Level Automated Planning System (FLAPS) is a mission planning tool built for use in European theaters of operation as an aid for the generation of the Air Tasking Order (ATO)*. Although FLAPS is not used extensively by USAFE (United States Air Forces in Europe), for which it was intended, it has been used in the creation of exercise ATOs and is well known in both the Research and Development and the user community. It contains algorithms and techniques for incorporating minimum lethality routes in the construction of candidate missions, and is one of the five decision aids that serve as the basis for the development of the Advanced Planning System (APS), a Rome Laboratory effort which will be utilized in all Tactical Air Force (TAF) theaters for force-level planning.

The paths generated by the FLAPS dynamic programming algorithm are for a single sortie of a generic aircraft type exposed to the threat environment at a constant altitude. Threat modeling in FLAPS consists of an array of cells, each cell containing a lethality value. The lethality value is arrived at by taking the negative logarithm of the probability of survival when exposed to a threat in a particular location, at a constant altitude. Terrain masking, or using the terrain in the defended area as a block to detection by enemy radar, increases the probability of survival, and is considered in the calculation of the lethality value. Dynamic programming is used to minimize the sum of lethalties along a path, but each cell is necessarily considered to be a separate independent event. This simplistic representation of lethality ignores threat connectivity, whether it be by communication, shared radar, etc., and cannot take into account the history of penetrator exposure to the threat or threats.

While FLAPS is a force planner, its routing algorithms are insufficient for routing force level assets for the creation of an overall plan. It makes no provision for specific

* The Air Tasking Order is the plan for the allocation of force resources for the next day's plan. It translates the Joint Force Commander's apportionment into sorties, by aircraft type, for each mission.

aircraft characteristics, such as speed, weaponeering, and capability, nor does it account for multiple penetrators. The threat environment is static and cannot account for the interaction of penetrators within the threat environment, over time.

6. HERO APPROACH

Automated force planning, represented realistically, would at the very best be difficult, if not impossible, to accomplish using traditional programming techniques. The use of such techniques, which do not offer the flexibility of object-oriented techniques, cannot easily accommodate the level of detailed analysis of the many variables that enter into "optimization" of mission planning. HERO exploits object-oriented design, using it as a platform for the dynamic representation of the planning process and of the variables that affect mission accomplishment, as well as serving as a good technique for organizing, and providing for extension to, a complex software system.

The objective of the HERO effort was to prove the feasibility of using object-oriented programming techniques in automated route generation for theater level mission planning, utilizing heuristics where possible to reduce the number of options that must be evaluated. Object-oriented programming provides the means to incorporate those mission planning parameters that are impossible or impractical to include using other techniques, factors that are inextricably associated with flying a route through a defended area with the best chance of survival. It allows the creation of a dynamic environment model within which planning takes place.

Actual missions consist of the interaction of real-world objects, therefore, mission planning should be concerned with such interaction as well. The HERO concept focuses on this interaction of objects. Within the HERO scenario, penetrators interact dynamically with threats and the environment, including terrain, as route options are developed. Penetrators are specific aircraft with particular characteristics and capabilities. HERO can plan multiple missions simultaneously, at a specified altitude, for single or multiple sortie missions, taking into account the command, control, and communication relationship among threats, and the potential effects of saturation and electronic warfare capabilities in traversing enemy airspace. The real-world objects are translated into the "objects" of object-oriented programming. Each high level grouping, or class, of objects contains all the shared characteristics of that type of object. The characteristics of a class in object-oriented programming are defined as "instance variables," and subsets of a class are "instances" of that class. New members are added by creating new instances of the class

* For the purpose of this paper, "optimization" means finding the best possible route under the given conditions and limitations.

object, and each instance either uses the default value, if any, for each instance variable, or it defines its own values. The creation of instances makes increasing the number of subsets an easy thing to do. Each instance can "inherit" the characteristics of the class, and have characteristics of its own, without a lot of additional coding. For example, although there are many types of penetrators, they all share the common characteristics of an airplane. Each penetrator instance that is created has the characteristics of an aircraft, such as weight, drag, and fuel consumption, but the actual values may be different. Figure 1 shows the definition of the penetrator class. (HERO was developed with Symbolics Flavors, a Symbolics extension to Common Lisp, where the top-level objects are called FLAVORS.) An F-15E is an instance of a penetrator, as is an F-16A, but each has its own unique characteristics. This type of flexibility and ease of coding makes it possible to create many instances of a class, as well as to easily add to those that already exist in the program.

```

(defflavor penetrator
  ((tail-number 0)
   (mission)
   (mission-role)
   (airframe)
   (fuel)
   (weapons)
   (ecm-assets)
   (total drag 0)
   (max-speed 0)
   (climb-rate 0)
   (service-ceiling 0)
   (range 0)
  ()
  :initable-instance-variables
  :writable-instance-variables)

```

aircraft. This value is derived from DELPHI data. Determination of lethality becomes increasingly sophisticated as more variables are factored in as a result of the interaction of an aircraft or group of aircraft with the threat environment. The routing algorithm gauges the cost of encountering a threat based on the threat characteristics, doctrine, and timing that is extracted from the state transition diagrams.

The threat model assumes threats to be in a ready state initially. From the ready state, the diagrams show how the threat behaves, based on time factors. The threat system requires a certain amount of time to perform some function, or change its state, and these possible transitions are represented by arcs between the nodes that represent possible states of the threat system. A state transition diagram for an SA-8 battery is shown as figure 2. The transition information extracted from the diagrams is used to construct the threat objects in HERO.

91-15510


```

; AS CONFIGURED
; IN KPH, AS CONFIGURED
; IN FT/SEC, AS CONFIGURED
; IN FT, AS CONFIGURED
; IN KM, AS CONFIGURED

```

Figure 1. PENETRATOR FLAVOR

7. THREAT ENVIRONMENT

Enemy Command, Control, and Communications (C³) are accounted for in the HERO threat models. The behavior of the threats is represented as transitions from one state to another, in some amount of time. Data for each threat is assembled on state transition diagrams. The diagrams reflect the generalization of behavior when individual TELS, RADAR, and Command and Control (C²) components are considered. The representation is simplified by forgoing consideration of mobile threats, assuming such movement would not lessen the overall threat because of overlapping threat coverage. The effects of saturation, the network of threats, and Electronic Counter-Measures (ECM) are extracted from the state transition diagrams and built into the route planning algorithm, thereby gauging the cost of encountering a threat.

The initial lethality value for an aircraft over a particular threat is based on altitude blocks and range of the threat to the

8. INTERACTION OF OBJECTS IN THE THREAT ENVIRONMENT

The objects hold "knowledge" of their states, stored as instance variables. The threat and radar objects contain information on what state they are in and how they will be affected as the environment changes, e.g., a penetrator is detected by radar. The penetrator object keeps track of the changes in weight, drag, and on-board EC (Electronic Combat) assets. The object which is the mission is then able to "collect" information on the state of the threats it passes (i.e. alerted by another penetrator, out of missiles, etc.) and store the cost of encountering the particular threat, given the state of threat and penetrator at that particular time. The lethality value that the threat (object) supplies to the mission (object) is based on the expected behavior of the particular type of threat (e.g. SA-6), with its particular alert history and current state of readiness, when it is alerted to a certain penetrator type with a specific weapons load. The partial routes created by

91 11 13 005

the mission object are evaluated by the A* ("A-star") search algorithm, using a best-first search technique. The best completed route is chosen based on lowest heuristic cost.

9. ROUTE PLANNING

The HERO planning space is modeled with a region-based approach. The airspace is represented geometrically. The airspace is divided into AGL (Above Ground Level) planes, defined as the airspace a fixed distance above the ground projected onto a plane. Each AGL plane is then divided into polygonal regions. Overlapping polygons are combined by computational geometry such that each polygon represents an area of homogenous characteristics. Within a polygon, any route extended will encounter the same lethality cost, thus avoiding the dilemma posed by search through adjacent cells of equal lethality in the kind of array of fixed lethality described in paragraph 5. To accomplish region-based route planning, the polygons are divided into triangular areas (triangulated) which serve as a basic area of traversal in generating candidate legs. At a given AGL, the range of a threat or radar is a cross-section of its actual three-dimensional range, therefore representing the danger to an aircraft flying a route through the defended airspace at a given AGL (figure 3).

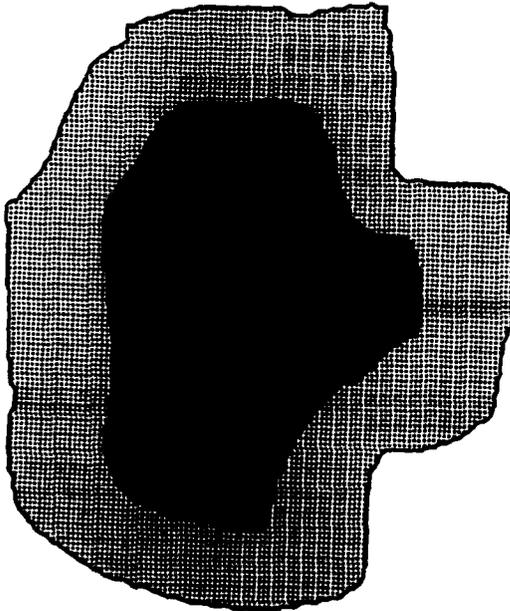


FIGURE 3. Threat Lethality as a cross-section in AGL plane.

In evaluating partial routes extended by the mission, HERO uses the technique of "possible worlds." The search tree is the tree of possible worlds in the planning space, and consists of many "plan worlds." A plan world represents the status of a mission simulation at a point in time, breaking the simulation down into steps or actions which are evaluated according to the cost of executing that step. Such costs, which are extracted from the scenario objects, include fuel used,

lethality encountered, etc., and form the basis for evaluating the possible missions against each other to find the best, or lowest cost, of the possible worlds. [3:4-6 - 4-7]

10. TIME AND SPACE

The processing time for route planning in HERO is significant. The HERO effort is a development model, and as the prototype is resident on a Symbolics 3650 machine, which is a better development environment than a run-time environment, speed was sacrificed for ease of development. The search through the tree of possible worlds becomes larger and requires more processing power and memory space as more variables are factored into the analysis. During demonstration of the system, planning ingress and egress for one mission of 4 F-16's through twelve threats (SA-6s and SA-8s) took approximately 1 hour. Although this is very discouraging, the dynamic threat representation holds far too much promise to be thrown away because of time considerations.

The geometric representation is a major factor in the time required for mission planning in HERO. Because branches are generated from triangle to triangle, too many branches may be created. Legs that are too short to be flyable may be extended as well. The trade-off of speed for local homogeneity became more and more obvious as the effort progressed.

11. SCENARIO

HERO uses the Soviet system of surface-to-air missiles and deployment doctrine. The scenario consists of 2 Fronts, 6 Armies, 19 Divisions, and 1 OMG (Operational Maneuver Group). The Front and Army units consist of air defense brigades and battalions. The divisions are tank and motorized rifle regiments, and air defense regiments with associated SAM (Surface-to-Air Missile) batteries. The OMG consists of 5 Brigades and includes combined arms regiments and air defense battalions with their SAM batteries. In order to preserve the unclassified status of the HERO project, older SAM system (SA-4, SA-6, SA-8, SA-9, with ZSU-23-4) are used. HERO currently uses central Europe (Germany) for its scenario, but any scenario that uses DTED (Digital Terrain Elevation Data) can be employed.

12. STATUS

The HERO prototype is installed at Rome Laboratory. It clearly demonstrates the complexity of the mission planning process as well as the benefits of object-oriented design and implementation. As the emphasis of the HERO effort was the exploration of route planning alternatives, some features of the user interface have either not been implemented or have been implemented without flexibility. One of the key issues that HERO was to address, overlapping threat envelopes, was not fully developed at the completion of the effort and stands as partially implemented.

13. CONCLUSION

One of the most important ideas to be gleaned from the HERO effort is the complexity of the task at hand. If the goal is to include all factors in mission planning and to test every possible path for optimality, the run-time would far outweigh the benefits of such an endeavor. It is well known that there is not much to be gained without risk. So, while we may risk missing the optimal route by the application of a well-placed heuristic, the outcome will be a very good, survivable option that may not have been obvious had the automated tool not been applied at all. Clearly, it is better to have a very good route with low lethality and other costs, arrived at within a reasonable amount of time, than to have no route at all because the optimum could not be arrived at in time, or one that is not as good because all the parameters cannot be considered by available automated or manual means.

The key factor that makes HERO a model for future development of route planning aids is the representation of the threat environment. HERO is capable of planning individual or multiple sorties, and the object-oriented approach has proven to be a practical way to insure the dynamic representation of real world objects in the automated environment, as well as to insure the flexibility required to encompass additional planning parameters as they are deemed necessary by the user. Although a system like HERO could be useful in planning individual sorties at the unit level, its use in theater-level planning shows much promise. Application of the principles demonstrated by HERO would allow the planner the opportunity to not only put the best possible resources on a target, but to more clearly see where and how tactics and strategy can be incorporated into the ATO.

REFERENCES

1. Clark, Thomas A. Analysis of the Force Level Automated Planning System (FLAPS). In-House Report, RADC-TR-89-300, October 1989.
2. Kruchten, Robert J. "Force-Level Route Planning: An Extendable Route Planning Concept." Proceedings, Automated Mission Planning Technical Symposium, 1989.
3. PAR Government Systems Corporation. "Functional Description for Heuristic Route Optimization (HERO)," RADC Contract F30602-88-C-0118, Data Item A003, 19 October 1989.
4. ----- "Threat State Notes for Heuristic Route Optimization (HERO)," RADC Contract F30602-88-C-0118, 6 April, 1990.

Discussion

1. Glen Johnson, United States

Does it frighten you to use independent threat models when this conference is on using MI to achieve networking?

Author:

There are interactions in the system. Comments made in the talk were an introduction to inadequacies of present models.

2. George Chapman, United States

Can your system adapt its heuristic search algorithm for pilot preference, mission, or mission phase (i.e. can the method you use to evaluate the lethality for each path change)?

Author:

No. There has not been an interface or protocol for allowing such changes.

3. Edward Gliatti, United States

What is the status of your program?

Author:

The fifth prototype is installed in the laboratory.

4. Dr G.H. Hunt, United Kingdom

Could you give some more detail about your measurements of the system performance and the results you obtained?

Author:

We are not as concerned with performance for this laboratory demonstration as we would be for a system being transitioned to the field. The processing time is really very slow. For a very simple scenario of four F-16s flying through SA-6s and SA-8s environment, I believe 12 of them, it can take from 1-4 hours, depending on configuration of the planner. We are considering many variables, but the representation of the state space is a major factor in the long processing time.

5. Lyle Reibling, United States

Since using A*, what information goes into the heuristic estimate, and is it an underestimate?

Author:

Not sure if it is an underestimate. Fuel and time are basic information for the heuristic estimate.

6. Dr M.C. Donker, Netherlands

You mentioned the A* algorithm to determine optimal routes. Is there any room for pilot preference? And secondly, would you not be too predictable if you use the same heuristics all the time?

Author:

Many of the parameters can be set by the user. As for us applying heuristic rules, based on preference, in order to limit search, my experience has shown that one pilot's

preference is not another's, but each believes his to be the correct one. There are actually very few heuristics in the program. They are basically constraints that can't be violated.

7. T. Schang, France

What is the depth of the search using A* or what is the partial leg length?

Author:

I'm really not sure if I can say what the depth is. We have not put a limit on depth of search. Minimum leg lengths is normally configured by the human planner. We have a value hardwired in for our use but I can't say off-hand what the value is.

8. Prof. A.N. Ince, Turkey

What aspects of your program can qualify for being "intelligent"?

Author:

Object oriented programming has its roots in artificial, or machine intelligence. HERO uses objects to represent the real world and the knowledge contained in those objects can be transferred to other objects.

9. T. Schang, France

As mission objects collect all threat state information, how do you manage combinatorial explosion?

Author:

While processing time is slow, due in part to the geometric representation of the state space and hardware limitations, the object-oriented approach does not have the problem of combinatorial explosion that other methods may encounter.

10. W.E. Howell, United States

Do you account for uncertainty in hostile aircraft state vector in your system?

Author:

Most of the development has been done with clean data. Uncertainty is being addressed at present at the situation assessment stage rather than tactics, which will follow later. *However*, we have done some checks, to ensure that the plans do not fall apart, by injecting noise as data mapped from simulator to tactical database area. This will be pursued *more fully as we go from initial development to evaluation and further development.*

11. J. Driessche

Please elaborate on the reasons why MUSE was preferred over other commercial systems?

Author:

Primarily the features offered, including windowing, downloading, as well as AI features, met our requirements and close working contacts with supplier gave promise of excellent support if modifications were required. Secondly, MUSE well supported on Sun workstations.



AD-P006 329

ADVANCED SATELLITE WORKSTATION (ASW)

T. BLEIER, S. HOLLANDER, S. SUTTON
 The Aerospace Corporation
 P. O. Box 3430
 Sunnyvale, CA 94088-3430
 USA

1. SUMMARY

The Advanced Satellite Workstation (ASW) was proposed and started by the author in 1986 as a low level independent research and development effort. Its goal was to evaluate the utility of expert systems in military satellite testing operations. It quickly became apparent, however, that the rule base was too limited. This project's goal was then modified to capture as much of the factory and flight experience as possible in an electronic, updateable medium. The resulting electronic library would then permit much deeper insight into the satellite's operation and more confidence in understanding and expanding the expert system rule base. The first on-line demonstration was done during June 1990 in one of the Air Force's Mission Control Centers, and evaluations of ASW are currently underway. Preliminary results are very encouraging.

2. GOALS OF THE ADVANCED SATELLITE WORKSTATION (ASW)

The Department of Defense supports an active constellation of 60 to 70 satellites today. The emergence of "LIGHTSAT" technology and PEGASUS air-launched boosters may reduce the cost of booster and satellites significantly. This is expected to dramatically increase the number of satellites which need support during the 1990-2000+ timeframe. One might observe that there should be a way of significantly reducing the corresponding cost of ground control for these less expensive LIGHTSATS. The "old" way of doing business using dedicated Mission Control Centers (MCC), large support staffs, on-site contractor support and highly specialized job functions may change if the defense budgets continue to decline while the appetite for satellite support to military forces continues to increase. This situation has led to the following ASW goals:

- Reduce the cost of ground control, especially the manpower requirements; e.g., the cost of training and maintaining a large staff of satellite controllers.
- Provide an integrated decision support environment for satellite operators to reduce the number of "specialists," i.e., use technically competent "generalists." The goal would be to provide these "generalists" with the right type of background data, in a form they could easily understand—a "decision support environment."
- Use advanced technology to automate operator tasks when it is appropriate; e.g., let the machine do the repetitive work and let the engineer do the judgmental work.
- Offload the primary command and control system processing workload; i.e., perform computational intensive telemetry processing and display functions with an offline computer.

The last goal reflects a current loading problem in the Air Force Satellite Control Network's "Command and Control System (CCS)" in which only 3-4 simultaneous satellite contacts can be supported from one Mission Control Center. ASW could offload the CCS and increase the overall system capacity by performing more telemetry processing in a separate processor.

3. ASW DESIGN

Today, satellite specialists are used to periodically support mission control team personnel who have significant experience in operating many satellites in a large control center but lack expertise in individual satellites and their subsystems.

The ASW design started as a concept in which an MCC operator, with some previous satellite control experience, was envisioned to have a much wider variety of information at his or her disposal. The ability to understand a satellite's inner workings was needed, and to do this, the ASW concept was envisioned to include elements such as logic diagrams, simulations, expert systems, and pictures of the "as built" satellite to provide this indepth knowledge (Figure 1). The computer aided design details (left), expert systems and waveform analyzers (center) and "as built" pictures from factory or on orbit photographs (right) represent the new types of data needed by the satellite operator. The functional goals described above and this broad concept were translated into a 3 screen workstation design illustrated in Figure 2.

4. ASW FEATURES

The following features of ASW are described with corresponding background illustrating why each feature was included in the system. During the development of ASW, the entire system's data base was generated for testing the joint Air Force/NASA satellite called Combined Release and Radiation Effects Satellite (CRRES), which is a fairly complex scientific payload with more than 40 instruments. The satellite was launched on July 25, 1990, and most importantly for the ASW project it represented an unclassified source of R&D data which was significantly easier to handle than classified data from operational DOD satellites. It allowed easy input to data bases and less restrictive configuration control over software changes for this prototype environment. This is very important if quick progress is desired. By allowing the developer an easy method of changing the system to fit the operator's desire during this type of prototyping, a clear statement of the requirements is an *output* of the process rather than an input to a classical design-to-spec activity.

Telemetry Display System: The telemetry display was built with more flexibility than the somewhat rigid displays of the existing Command and Control System (CCS). Telemetry point selections, configuration of one or multiple data plots, scaling, and windows were basic features desired. Selected color displays of both minor alarms and critical limits, forward and backward scrolling of data, and multiple plots using different colors were additional features included in the system. It was built around the commercial product DATAVIEW with "C" code enhancements. Figure 3 illustrates one of the screen displays. The key feature of this system is the ability to change and customize the displays in realtime. This was an essential feature that the technical advisors wanted to augment the more rigid display in the primary CCS system. Software "buttons" are placed on the screen so the analyst can select features and options by clicking a mouse. This display also allows the operator to ask for more detailed expert system advice as well as supporting information contained in "hypercard" displays.

Expert Systems: A typical satellite, contacted through a remote tracking station, may generate between 3 and 30 minutes of

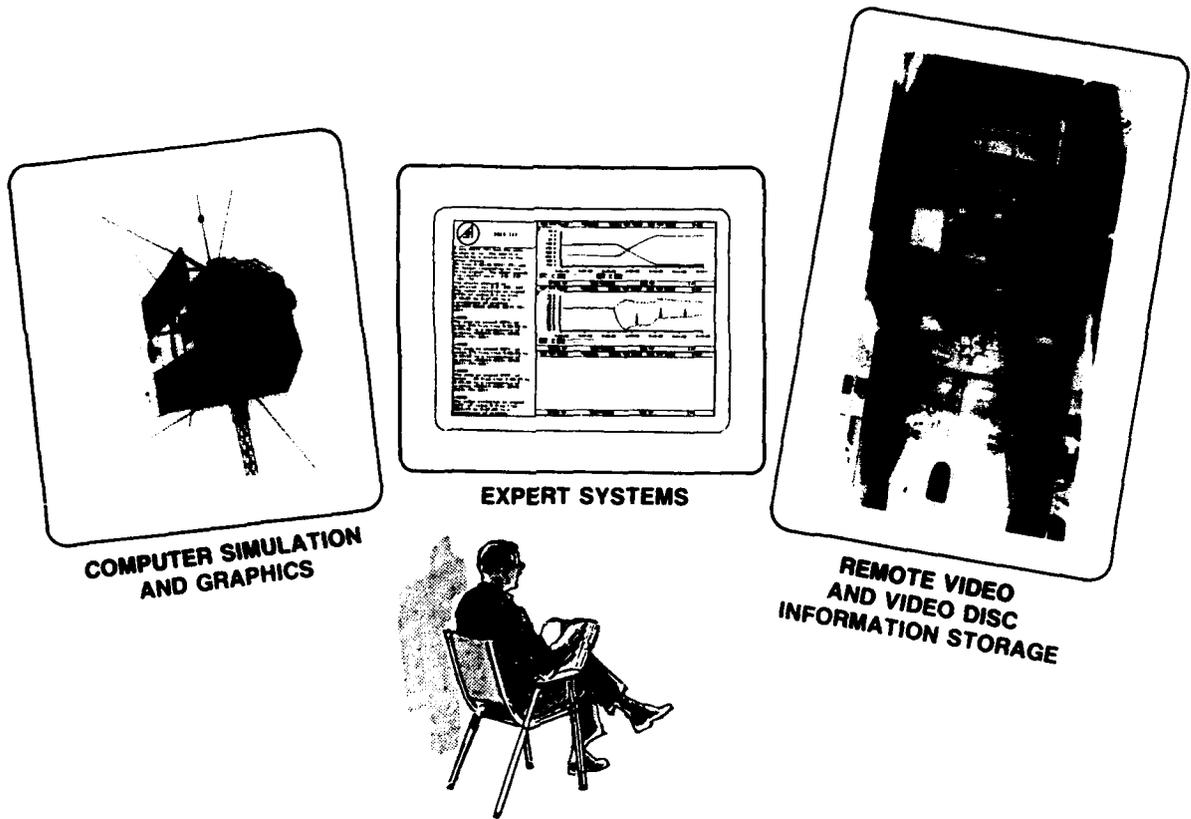


Figure 1. Advanced Satellite Workstation

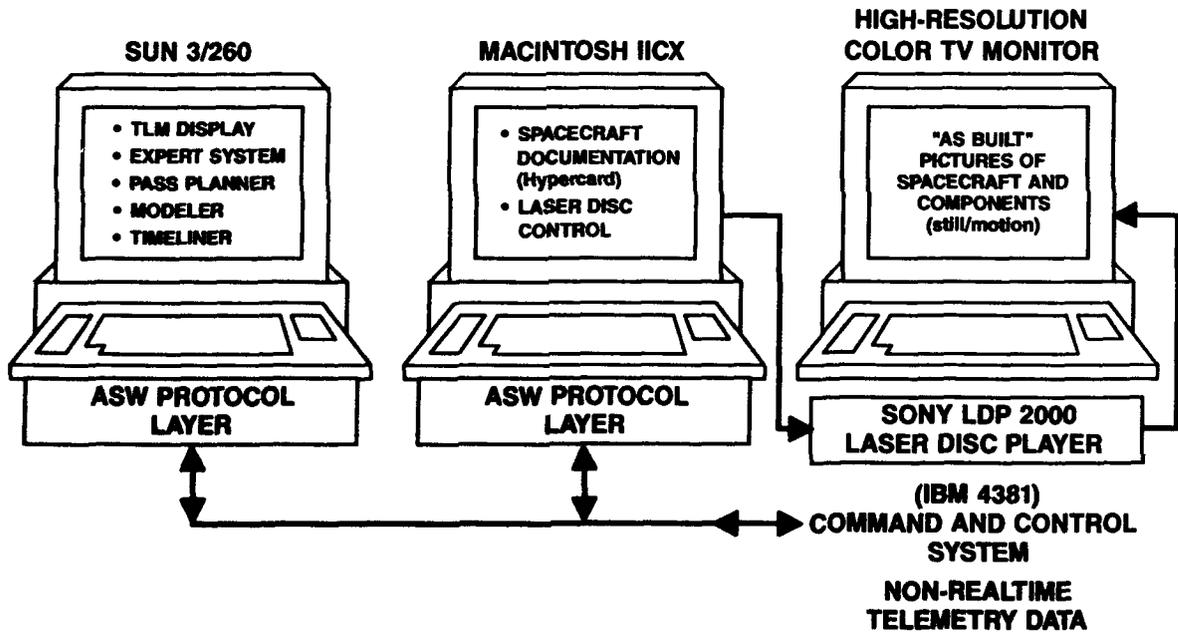


Figure 2. Three Screen Workstation Designs

ASW Post Pass Expert System

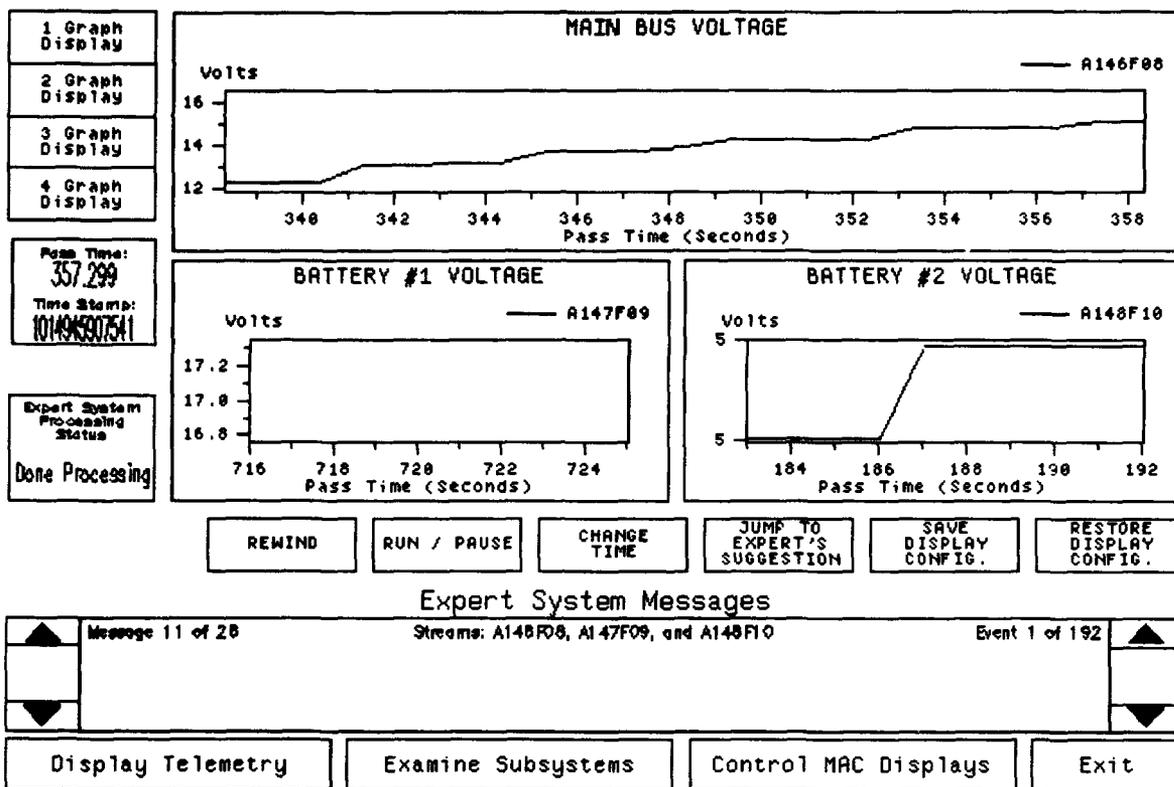


Figure 3. ASW Telemetry Analysis

telemetry data for each realtime contact or "pass". This data may be merged with tens to hundreds of minutes of telemetry recorded onboard the satellite and played back on a separate telemetry channel along with the realtime data. The result is long sequences of telemetry data which must be laboriously scanned by human analysts. ASW was set up to have an expert system (currently with 90-100 rules) for the electrical power system scan all of this data in a few minutes. The rules include simple maximum/minimum limits, rate of change, and combinations of rules based on both design and operational experience. Any violations of the predetermined rules causes the expert system to select the offending telemetry waveform and present the anomalous data to the operator. The expert system automatically calls the hypermedia system and selects the proper section of the Orbital Operations Handbook - OOH (an "encyclopedia" of the satellite's subsystem information). We found that many satellite controllers do not trust expert systems. ASW tries to develop their trust by augmenting the expert system's terse recommendations by displaying the raw data which triggered the recommendation, and electronically "pointing" to the section of the satellite's OOH which explains how the subsystem should operate. Figure 3 illustrates the types of messages displayed by the expert system (at the bottom of the figure). It also states how many instances of the anomalous behavior were detected. ASW's expert system is configured now for only the electrical power subsystem of CRRES, but other subsystems can be added. The commercial expert system NEXPERT was used to build this application, and it executes in a Sun/Unix environment. The developers created the first expert system example, including the linkage to the telemetry display and hypermedia.

Subsequent subsystems were intended to be generated by the satellite technical analysts. To date, several "rules" have been created by these analysts with relative ease.

Hypermedia: Discussions with experienced satellite analysts disclosed that many types of information are used to support an "expert's" knowledge base. These include the logic diagrams from each electronic controller, sample waveforms from previous tests, written text describing correct operational behavior of the subsystems, "as built" pictures of various boards, assemblies, subsystems and systems, simulations, performance curves, and finally, simple operator experiences recorded as notebook entries. This information existed in many different forms and was difficult to organize. Mr. Stewart Sutton analyzed the situation and decided to try various scanning, storage, and display techniques to capture the data in a technique called multimedia. Macintosh scanners were used for text and graphics data, video tape pictures were converted to a 12-inch laser disk for a Write Once Read Many (WORM) drive, and logic diagrams were converted to a modeling system written in C code. The entire system was organized around a "point and fetch" type data base system called Hypercard (see Figures 4-7). The multimedia type information under the controlled access of Apple Macintosh Hypercard, is called hypermedia. The application of this Hypercard technique for ASW was termed the Information Navigator, and selected parts of three volumes of the Orbital Operations Handbook were scanned or hand typed into the system. An operator can now "navigate" through the information stacks at his or her own pace, and can browse in the order which makes sense for that particular investigation. The Telemetry Display System, the

91 1113 006

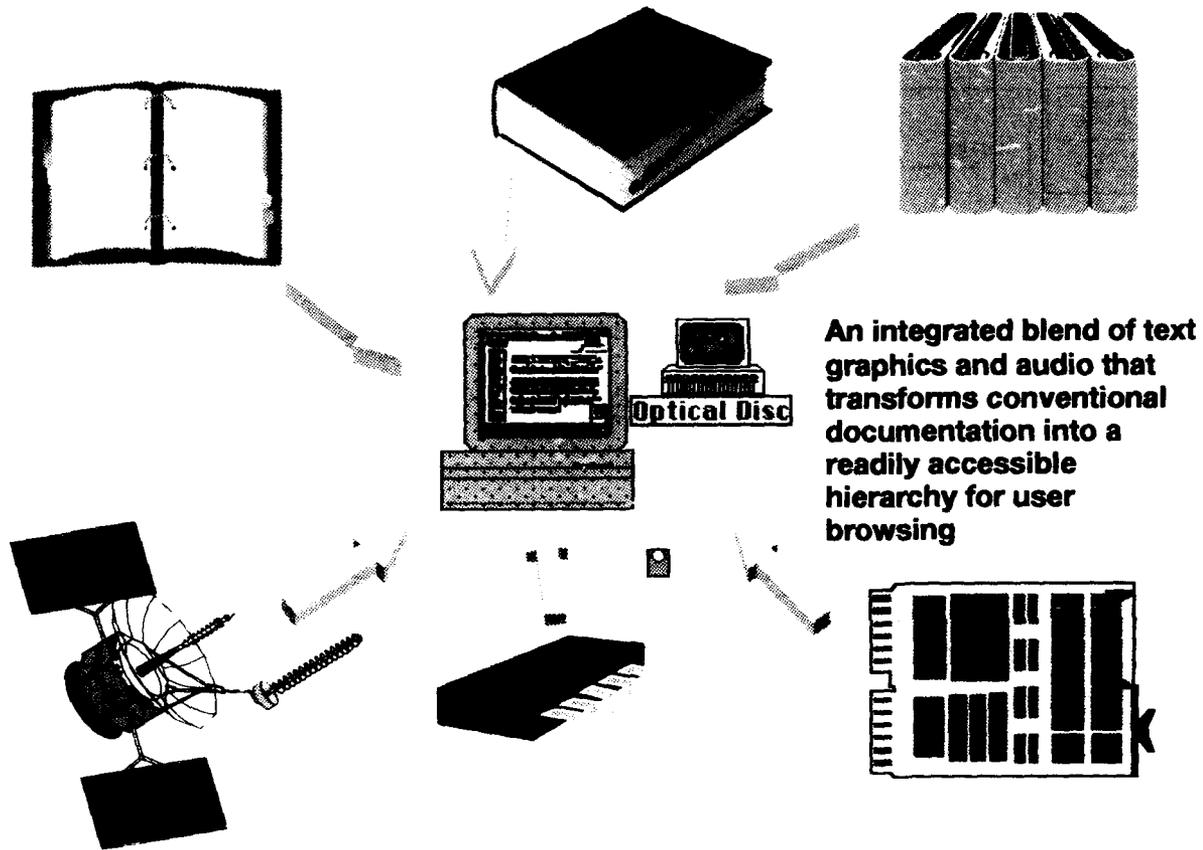


Figure 4. Multimedia Information Systems.

INDEH

CRRES EPDS OVERVIEW

R
?

Electrical Power and Distribution System Overview OOH(4.6)

Power Management

Power System Adjustment

EPDS Initialization

The overall purpose of the CRRES EPDS is to provide for the generation, storage, distribution, and control of power required to operate the spacecraft and experiments. It does not require the use of ground-based power management computer programs. While sunlit, power is normally supplied by the solar arrays. Batteries supply power while in the earth's shadow, or during specific functions requiring CRRES to be pointed away from the sun.

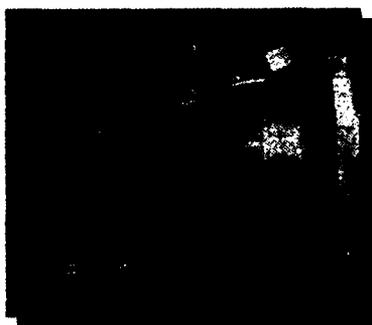
Command Mnemonics

Telemetry Mnemonics

1. overview, (OOH, Vol. 1, pp. 4-92)

ENTRY SCREEN FROM THE OOH EPDS PROTOTYPE

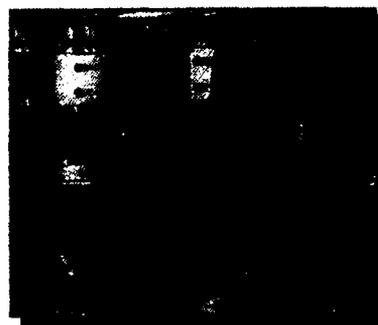
Figure 5. Information Navigator OOH.



**CANISTER CHEMICAL
RELEASE**

- Provide quick and easy access to spacecraft subsystem hardware in graphical form

- Animation of subsystem components by randomly accessed video clips and stills of as-built hardware



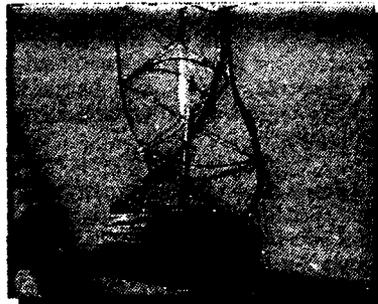
SEPARATION BOLTS



**SOLAR PANEL
AND BULKHEAD**

- 3D visualization of subsystem components by randomly accessed video clips of subsystem models

- Correlation of video, graphics, and audio with supplementary text information



MAGBOOM ANTENNA

Figure 6. As Built Pictures.

✳
INDEX

CRRES Laser Disk Video

View #1

SOLAR ARRAY
Solar Zoom

SOLAR ARRAY
Solar Bracket

Fasten Panel

SOLAR ARRAY
Solar Side

Cable Away!

Set Cntl

View #2

GPS Antenna

SOLAR ARRAY
Pan Array

View #6

SOLAR ARRAY
Solar Pan

Thrusters

QA Inspection

View #7

View #4

view #8

< >
<< >>
|< >|
search

Audio ch. 1 Display frm *s
 Audio ch. 2 Picture on

Set Player

Figure 7. As Built Pictures (Laser Video).

Expert System, the Hypermedia, and the non-machine interfaces combine to form the basis for the "Decision Support Environment."

Modeler: Logic diagrams, performance curves, and text descriptions frequently do not clearly describe the electronic control logic which exists in each of many satellite subsystems. Mr. Peter Homeier generated an easy-to-use logic description which starts at a high level "box/input/output" view, and proceeds in a hierarchical fashion to successively lower and lower elements within the subsystem. The approach is to lead the user from simple descriptions to more detailed descriptions in successively more detailed representations; unfortunately, time did not allow for a specific application of Modeler within this version of ASW for the CRRES satellite.

Automated Contact Support Plan Generator: The ASW is a type of electronic "tool box," which includes a collection of support programs; e.g., the telemetry scroller, expert systems hypermedia, and laser disk storage of still and motion pictures of the spacecraft. It is based on the goal of reducing workload and increasing the effectiveness (knowledge) of the mission control staff.

The staff was shown the first prototype of ASW and was asked to identify additional functions which were labor intensive and could be streamlined. A primary candidate was the routine task of manually generating a "Contact Support Plan;" i.e., a time ordered list of commands and actions which must be executed during a routine satellite contact. The pass plan is done dozens of times per day, thousands of times per year. The automation of these functions now includes a windows-based text editor with "fill-in-the-blank" capabilities, utilizing both automatic and user-selected information. The output is a laser printer version of the pass plan in a format that everyone is accustomed

to seeing, but which is faster and more accurate than the manual methods. Figures 8 and 9 illustrate two screens of this system, where relevant data is being electronically transferred and accepted by the operator to generate both the header and body of the pass plan. Data relevant to the individual pass (e.g., station name, rise time, elevation, fade time, etc.) are automatically retrieved and inserted within the plan. Templates for standard command plans (e.g., "VCR read out") are retrieved and inserted into the command list file.

Timeliner/Scheduler: Planning and operation of an R&D satellite includes the time phased planning of many experiments throughout the satellite's life. These experiments must be correctly interlaced with realtime contacts, other experiments orbit constraints (light, dark, apogee, perigee, etc.), and tape recorder limitations. A graphical experiment planner was created called Timeliner which can illustrate these constraints and options on one page. The data can be added, deleted, moved, or modified via mouse actions, and the output sent to a laser printer for review by various team members. Figure 10 illustrates a sample output for several days of the CRRES mission.

5. ETHERNET CONNECTION TO THE COMMAND AND CONTROL SEGMENT (CCS)

Any type of workstation which analyzes satellite telemetry needs decommutation and engineering unit conversion functions. ASW uses the services of the Air Force Consolidated Space Test Center's (CSTC) facility to perform the functions in the IBM 4381 computer system called Command and Control System (CCS). Telemetry data is received at a remote tracking station, relayed to Sunnyvale, California, routed to a Mission Control Center, and processed/stored by the CCS. ASW terminals are connected to an Ethernet which is fed by a file

Pass Plan

REV 12.0		SUPPORT PLAN - RCN - 4643					PAGE ___ OF ___ PAGES DATE 31 MAY 90				
STATION	START	STOP	RISE	MD	FADE	DURATION	T/W/S	R/T MODE/DOZNEY	TAPE ANNOTATION		
ERANT	0739	0748	27505	27819	28044	F-R 100		33/13	PB		
			RCO AZ 201.08	MO AZ 147.28	FOE AZ 92.90	SFC-R 0	YCD 123	OFFSET 0	SHIPMENT <input checked="" type="checkbox"/> YES <input type="checkbox"/> NO		
ANALOG VCR R/O			MAX EL	FOY	S/C STATUS AT ACQUISITION						
			0.64	07	TRN	RECORDER STATUS					
					CRMS		VCR1	VCR2	VCR3	DIGITAL	
					MEMCMA(CMS) 28/88008	POS	300	300	300	BCT	
					ALPHA 43-180-144	STATUS	0	0	0	0	
			TRANSMITTER STATUS								
				TR1	TR2	TR3	TR4	TR5	TR6	TR7	
				1	2	3	4	1	2	SP1 <input type="checkbox"/>	
CCS			EXPERIMENT STATUS								
MASTER	RLM ID	LSET/YES	OFFSET								
HL 101/7	01	123	0	L1 <input type="checkbox"/>	L2 <input type="checkbox"/>	L3 <input type="checkbox"/>	L4 <input type="checkbox"/>	L5 <input type="checkbox"/>	L6 <input type="checkbox"/>	L7 <input type="checkbox"/>	

Header

Body

Full View

Load

Save

Print

Restart

Exit

Figure 8. Automated Contact Support Plan Generator

Pass Plan

	NON TIME	ACT TIME	<input checked="" type="checkbox"/>	ON/OFF/DAYS	REMARKS (SUN/EDL/SUN)
CUT				→	CTS ENAB/DSBL
	27476			ACTIVE	
COPY					
PASTE	27595			RSE/AOS	0.0 OBS
				A/TRK	SS17-
EDIT				UPLINK MOD 0	
				RNLOCK	

Header	Body	Full View	Load	Save	Print	Restart	Exit
--------	------	-----------	------	------	-------	---------	------

Figure 9. Automated Contact Support Plan Generator (Body/Editor)

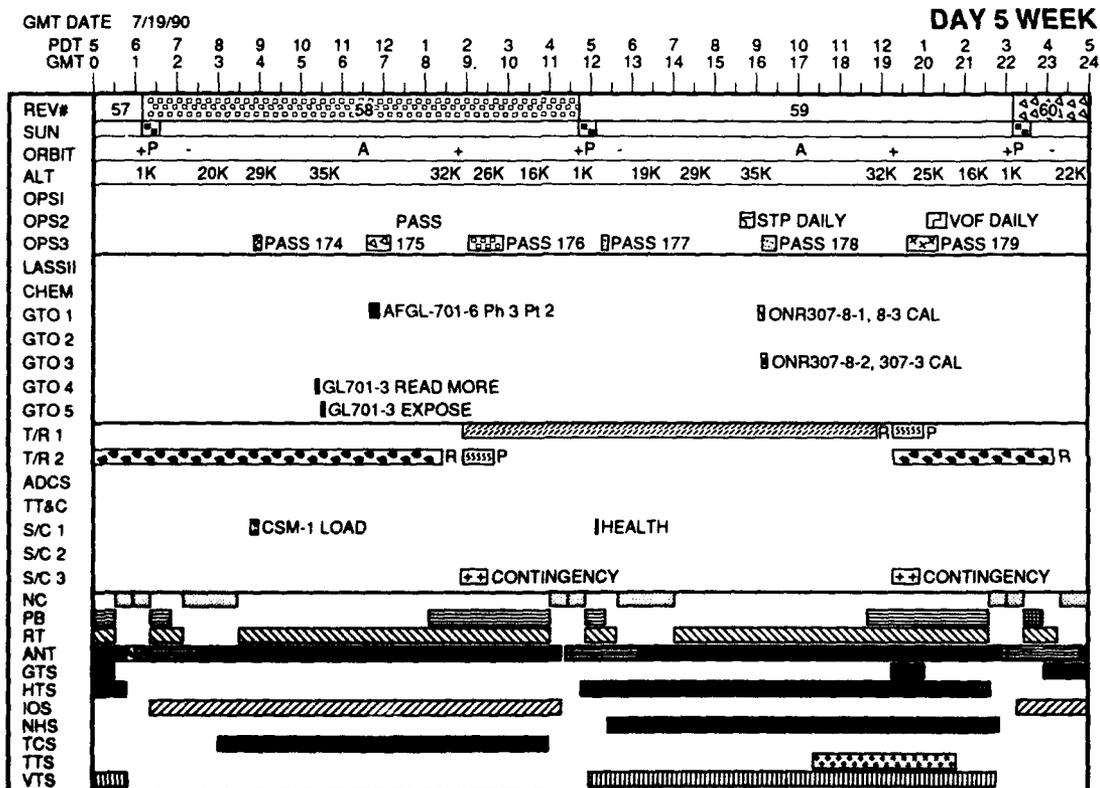


Figure 10. CRRES Mission Data

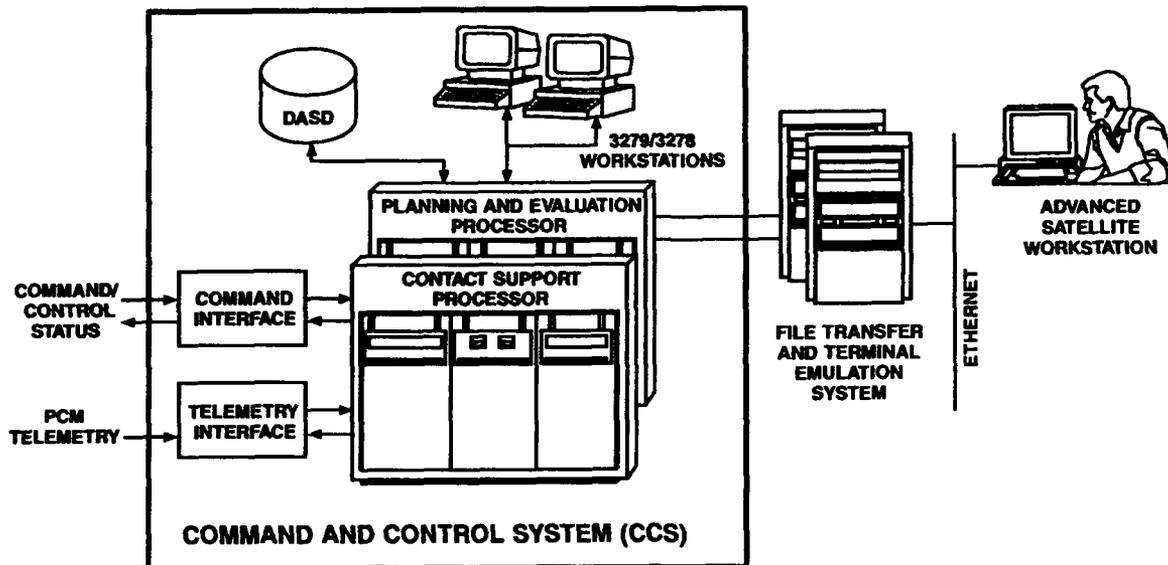


Figure 11. System Configuration

server attached to CCS (see Figure 11). The data is retrieved from an archive disc and sent to the Ethernet in a file transfer process so that ASW can access the data repeatedly without loading the CCS with many requests for service.

6. TEST CASE IN MISSION CONTROL CENTER VI

The process of building ASW reached a milestone in July 1990 when the laboratory version was moved into an operating environment—Mission Control Complex VI at the CSTC in Sunnyvale. The functions of ASW currently being evaluated in the operational environment are:

Training—Using Hypercard/Hypermedia services of the Orbital Operations Handbook (OOH) to train Planner/Analysts (PA).

Data Browsing—PA analysis of electrical power system performance by using the expert system to apply 90 to 100 rules to the recorded data identifying unusual waveforms. The telemetry display system then allows the user to judge for himself if the waveform is benign or anomalous.

Anomaly Support—Using the telemetry system, hypercard, video picture, and modeler to analyze various subsystems (Modeler is not equipped with a CRRES data base as of this writing).

Pass Planning—PA using the automated Pass Planner to improve the speed and accuracy of the pass plan generation process.

Experiment Planning—Contractor Technical Advisors using the Timeliner to plan many of the thousands of experiments and support requirements.

Several very useful suggestions were received from the support team, logged by an Aerospace Corporation engineering intern, and relayed to the design team. The updated version of ASW is now available for use as an engineering tool. The long term goal is to generate a final report, to highlight the successful features, and include these features in a Technical Requirement Document (preliminary A-Spec) for future

procurement of a fully documented and supported satellite workstation section for the Air Force CSTC.

7. FUTURE

The limited time and funding of ASW did not allow all of the ideas to be developed and implemented. The types of problems encountered resulted in the following enhancements that should be included:

1. The use of new, high quality video cameras for taking the "as built" pictures.
2. A technique for transferring the video pictures directly from a camcorder to a read/write laser disk. (The process now involves "mastering" on a 1-inch master tape, then employing a \$2000 transfer procedure to "burn" a 12-inch laser disk.)
3. Addition of a data base manager to organize and maintain telemetry files on the SUN workstation.
4. A better laser disk to increase storage capacity in the Sun Workstation for telemetry files (e.g., 1-2 gigabyte capacity).
5. Improve the ease of acquiring OOH (satellite technical data); e.g., contractually task contractors to deliver these OOH documents in hypercard-compatible formats/media.
6. Establish interfaces so any expert system and telemetry processor can access each other's data.
7. Determine the cost (in man months) to deliver a similar system for the next satellite mission. Given ASW experience and the enhancements above, how long would it take to acquire the satellite data and input to a fully populated ASW database for each satellite subsystem.

The ASW designers and users encourage the collective community of spacecraft builders and operators to share similar accounts of success and problems with experiments such as ASW. This type of cooperation will help reduce the support costs associated with all DOD, NASA and civilian spacecraft.



COMPONENT PART NOTICE

THIS PAPER IS A COMPONENT PART OF THE FOLLOWING COMPILATION REPORT:

(TITLE): Conference Proceedings of machine Intelligence for Aerospace Electronic Systems Held in Lisbon, Portugal on 13-16 May 1991 (L'Intelligence Artificielle dans les Systemes Electroniques Aero spatiaux)

(SOURCE): Advisory Group for Aeronautical Research and Development, Neuilly-sur-Seine (France)

TO ORDER THE COMPLETE COMPILATION REPORT USE AD-A242025

THE COMPONENT PART IS PROVIDED HERE TO ALLOW USERS ACCESS TO INDIVIDUALLY AUTHORED SECTIONS OF PROCEEDINGS, ANNALS, SYMPOSIA, ETC. HOWEVER, THE COMPONENT SHOULD BE CONSIDERED WITHIN THE CONTEXT OF THE OVERALL COMPILATION REPORT AND NOT AS A STAND-ALONE TECHNICAL REPORT.

THE FOLLOWING COMPONENT PART NUMBERS COMPRISE THE COMPILATION REPORT:

AD#: _____ TITLE: _____

AD-P006 330

DTIC
ELECTE
NOV 13 1991
S D D

Accession For	
NTIS (GPO&I)	<input checked="" type="checkbox"/>
DTIC (DA)	<input type="checkbox"/>
Unannounced	<input type="checkbox"/>
Justification	
By _____	
Distribution _____	
Availability Codes	
Dist	Avail and/or Special
A1 21	

[Faint stamp: This document contains information which is exempt from automatic downgrading and declassification]

A SYNERGISTIC APPROACH TO REASONING FOR AUTONOMOUS SATELLITES

Captain James M. Skinner
Phillips Laboratory
Kirtland AFB, NM 87117

Professor George F. Luger
University of New Mexico
Albuquerque, NM 87131

SUMMARY

Based on our earlier research [1,2], we are convinced that the best way to approach problem-solving tasks is not through any single method of reasoning, but by a method that will allow several reasoning methods to be blended together. The thrust of this effort is to develop a synergistic approach to reasoning that will allow a system to rely on multiple reasoning methodologies, thus benefiting from the strengths of each of the reasoning methods, while minimizing their respective weaknesses. This paper discusses the steps we have taken towards developing such a system, which are: (1) the categorization of reasoning methods, (2) the selection of reasoning approaches to blend, (3) design of a framework to blend the systems, and (4) proposed tasks to investigate the result.

Human Reasoning

Research in Artificial Intelligence (AI) is sometimes partitioned into two approaches: a psychological approach which employs AI programs as useful tools for studying the mind, and an engineering approach which uses AI programs to solve problems, regardless of their similarities to human reasoning. While this effort is concerned with the latter approach, it is useful to examine the reasoning methods used by humans to guide us in modeling a reasoning approach for machines.

When humans reason they rely on many different reasoning techniques, including: (1) relying on memory of past cases when solving problems which are similar to these past cases.

(2) using heuristics, or rules of thumb, when confronted with a familiar situation.

(3) following a procedure to solve a problem.

(4) developing a mental model of the problem, or referring to a schematic to diagnose problems in a complex system.

(5) comparing an unfamiliar problem to a past problem (and solution) from a different domain.

(6) using a formal logical method to prove a theorem is true.

The methods outlined above are by no means an exhaustive list of human reasoning methods. They do, however, represent a sample of methods that humans use. We might wish to ask if a computer could reason in a similar method.

First, we must address the question of whether a machine is capable of reasoning. While entire books have been written on related subjects [3,4,5], we will sidestep the issue by defining reasoning as "the drawing of inferences or conclusions from a

set of facts or suppositions." When confined to this definition, few people will disagree that not only is it possible for computers to reason, but that many already possess this capability.

With this in mind, we can see that each of these reasoning methods have been automated to some degree as (1) case-based reasoning, (2) rule-based reasoning, (3) conventional reasoning, (4) model-based reasoning, (5) analogical reasoning, and (6) automated reasoning respectively. While it is doubtful that we will ever be able to automate the human reasoning process, a good facsimile might come from developing a system that is able to combine some or all of these reasoning techniques. Before we discuss how that might be done, it would be useful to develop a scheme to categorize the different methods of reasoning.

Several schemes are possible for classifying reasoning methodologies. Past approaches have included classification based on the degree of precision (e.g., sound versus fuzzy reasoning) or degree of generality (general purpose versus special purpose reasoning). The approach taken in this paper is a classification based on what we call the "depth" of reasoning.

Shallow Versus Deep Reasoning

Reasoning methods can be categorized as "shallow" or "deep," depending on the type of knowledge used in the system. An approach taken by Harmon (Figure 1) is to classify knowledge into three levels: (1) heuristic or shallow knowledge, (2) domain knowledge (includes procedural models) and (3) deep or theoretical knowledge [6].

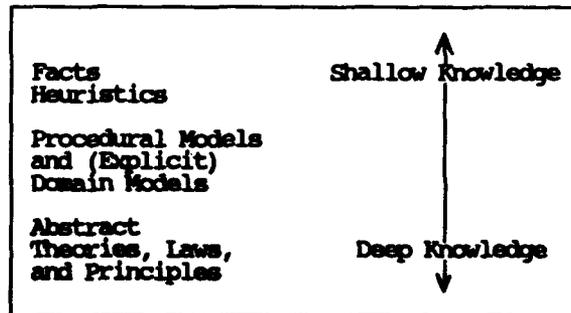


Figure 1. Three Levels of Knowledge [6].

To illustrate this knowledge classification scheme, Harmon considers how a non-technical person would react if their television set did not work. First, he might check to see if it was plugged in.

If so, he might shake the cord and switch all of the knobs on and off several times. If all of this failed, the person would decide he needed to take the TV to the shop or purchase another one. This person has only Level 1 knowledge of television sets. It consists of a few heuristics that he applies to televisions, toasters, and other electrical devices.

If the person takes the TV to a repair shop, he expects the repair person to have much more knowledge about televisions. The repair person has Level 2 knowledge of television sets; she probably does not have an advanced degree in electronics or understand the physics of television, but she has domain or procedural knowledge of televisions and TV repair. The domain model includes many heuristics about how a malfunction in one component might induce symptoms elsewhere; the procedural model provides her with a step-by-step approach to troubleshooting the TV. The repair person can rely on both models to guide her in the systematic diagnosis of the TV set, and in selecting an appropriate repair strategy.

If you wanted to design a new television, you could probably find someone in an engineering department of a university who understands how televisions worked. This person has Level 3 knowledge of television sets. He probably can't fix your television set, but he has the underlying or deep knowledge of physics and electronics that would enable him to create a device that would function as a TV.

We can extend this knowledge classification scheme to reasoning systems. As might be expected, traditional expert systems (i.e., rule-based system) are associated with shallow knowledge; they rely on heuristic knowledge to solve a problem. Appropriately, we will refer to these systems as shallow reasoning systems. Conventional systems, which normally encode procedural knowledge, will be considered as the midpoint between shallow and deep reasoning systems. Automated reasoning systems (i.e., theorem provers) are associated with deep knowledge, and will be called deep reasoning systems.

When extending this classification scheme to reasoning systems in general a problem arises; theories can be encoded in rules, and heuristics can be represented as models. To account for this, we will modify the definition slightly.

To classify a reasoning system as shallow or deep, we will consider whether the system uses an implicit or explicit model of the domain when reasoning. The domain model is considered implicit if there is no distinction between different types of knowledge. Rule-based systems are an example; they use heuristic knowledge arising from many different sources (e.g., empirical knowledge, structural knowledge, causal knowledge) and will often combine different types of knowledge into a single rule without distinction. A system using an implicit domain model will be considered a shallow reasoning system.

The domain model is considered explicit if a distinction is made between the different sources of knowledge. The types of knowledge and the role that each plays in the problem-solving process is made clear. As an example, a system that models the subcomponents of an electronic component and describes the relationship between the components is an example of a system employing an explicit domain model, and would be considered a deep reasoning system.

While there is a tendency to attach a bad connotation to the term shallow, this is not justified with reasoning systems (consider that MYCIN, a very successful expert system used to diagnose meningitis infections, contained only shallow knowledge). Each form of reasoning has strengths and weaknesses. The application should dictate which form of reasoning is used.

Applications that require the system to capture experiential knowledge are well suited for shallow reasoning systems. This type of knowledge is easily encoded into empirical associations. Consider developing an expert system to capture the knowledge of a retiring technician. The technician's knowledge has been gathered over the years from many different sources and is best represented as an implicit model.

However, the lack of explicit representation of more fundamental knowledge can cause some serious problems. These problems include: (1) rapid degradation outside the narrow domain of expertise of an expert system, possibly leading to incomplete or incorrect conclusions; and (2) the inability to transfer knowledge to other tasks. This inability stems from the fact that heuristics are usually task specific; rules written for the diagnosis of a component will not normally be useful in an expert system developed to design the component, even though the underlying mechanism and physical principles are the same for both tasks [7].

Diagnosis of man-made devices is a good application area for a deep reasoning system. Especially if the device is a complex state of the art component with little or no expertise available. The explicit model of the deep reasoning system allows the system to respond to situations that could not have been predicted. But development of the explicit domain model requires the fundamental principles of the domain to be well understood. This makes deep reasoning inappropriate for areas which are not well understood. In medicine, for example, most of the knowledge used for diagnosing and treating diseases is empirical, not based on a model of the relevant biological and chemical mechanisms.

While reasoning methodologies do not normally rely purely on shallow or deep reasoning techniques, we propose that they can be broadly categorized as one or the other. Returning to the three levels of

knowledge of Figure 1, we claim the reasoning methodologies can be placed in correspondence with the levels of knowledge as shown in Figure 2. In this paper we will discuss the three shallow reasoning methodologies (case-based, rule-based, and conventional) and one of the deep reasoning methodologies (model-based).

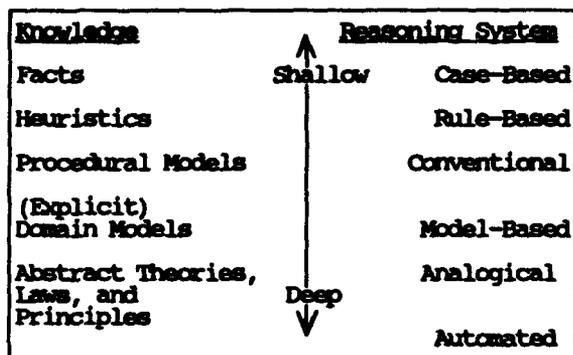


Figure 2. The Depth of Selected Reasoning Systems.

Case-Based Reasoning

The mounting evidence that human experts rely heavily on memory of past cases when solving problems has led to an increase in the research of case-based reasoning (CBR) [8]. In CBR, past cases are used to solve a new problem case. This can increase both the quality and the efficiency of the reasoning by deriving shortcuts and anticipating problems in new situations based on past experience with similar cases.

There are two main types of CBR: classification and problem-solving. Classification CBR argues that a new situation should or should not be treated like a past one based on similarities or differences with the past case. Problem-solving CBR formulates a solution suited to the new case by modifying or adapting past solutions. Classification CBR is usually used for strategic planning or legal reasoning while problem solving CBR is typically used for design or diagnosis.

Proponents of CBR claim several advantages. First, the shortcuts in reasoning and the capability of avoiding past errors enhance performance. Second, no causal model or deep knowledge of the structure is necessary, although their existence will improve performance. A third advantage is the scalability of CBR. While there is a bottleneck in choosing the base cases to reason with, researchers have managed this problem by indexing the cases. Researchers believe that indexing will allow them to scale CBR up to the point where they can tackle real problems in real time. Finally, knowledge acquisition in case-based reasoning is much easier than for other reasoning methods. This is because much of the knowledge required for CBR is in the form of cases. Furthermore,

many domains have existing case bases (e.g., medicine, law) that could be used as a seed.

Primary concerns with CBR are: (1) organizing cases in memory, (2) selecting the relevant cases, and (3) modifying existing cases to fit new problems. All of these areas are current topics of research.

Case-based reasoning is suitable for domains involving classification (medical, legal, planning) and problem solving (design, mathematics, diagnosis). CBR is best suited to domains in which many training cases are available, and where it is difficult to specify appropriate behavior using abstract rules.

Rule-Based Reasoning

Rules are the most commonly used knowledge representation technique in artificial intelligence. In a rule-based expert system, the domain knowledge is represented as sets of rules that are checked against a collection of facts or knowledge about the current situation. The rules are expressed as IF-THEN statements. When the If portion of the rule (the premise) is satisfied by the facts, the action specified by the THEN portion (the conclusion) is performed. This action may result in the addition of new facts, continuing the cycle until the goal is achieved.

The strength of rule-based systems lies in the simplicity of their construction and maintenance. Rules can be easily constructed because experts tend to express most of their problem-solving techniques in terms of situation-action rules which can be readily coded. Maintenance of the system is somewhat simplified because each rule approximates an independent chunk of knowledge, so that existing knowledge can be refined and new knowledge can be added in a modular fashion.

The weakness of rule-based system is their lack of robustness. This arises from the fact that only an implicit model exists, not an explicit one. The information contained in the model is a collection of empirical associations drawn from an expert. It is impossible for the system to respond to a situation unforeseen by the expert (or not coded by the knowledge engineer).

Rule-based systems have been successful in a wide variety of applications including diagnosis, configuration, and control. The classic example of a rule-based system is MYCIN, an expert system designed to solve the problem of diagnosing and recommending treatment for meningitis and bacteremia. While tools exist capable of unambiguously diagnosing meningitis, these tools require on the order of 48 hours to return a diagnosis. Unfortunately, treatment for meningitis patients must begin immediately. The goal of MYCIN was to emulate the doctor's expertise of forming a diagnosis that covers the actual infecting organisms based on initial symptoms and test results [2].

91 1113 007

91-15511



Conventional Reasoning

Contrary to popular belief, AI advocates do not propose that all problems should be solved with AI tools. Conventional programming methods will suffice in many instances. Table 1 lists the differences between conventional systems and AI systems.

Table 1. Comparison of Conventional and AI Systems [9].

Conventional	AI System
Representation and use of data	Representation and use of knowledge
Algorithmic	Heuristic
Repetitive Process	Inferential process
Effective manipulation of large data bases	Effective manipulation of knowledge bases

Many problems are best solved by conventional programming [9]. For example, problems which have tractable mathematical solutions such as solving differential equations with numerical analysis techniques are not appropriate for AI, while problems requiring algebraic simplification lend themselves quite readily to symbolic reasoning. Problems that can be solved with algorithms (formal procedures that guarantee the correct solution every time) are better left to conventional systems. For example, it is more cost effective to sort lists with a conventional system than with an AI program.

Model-Based Reasoning

The first of the deep reasoning systems to be discussed is model-based reasoning. Model-based reasoning uses an explicit model of a system to describe the components of a physical system, the connections between the components, and the behavior of each of the components [10]. The system is able to reason about physical laws which apply to the system, and the effect the laws may have on the system. This form of reasoning allows for a more robust response to a previously unencountered set of circumstances.

Model-based systems commonly use causal reasoning, reasoning from first principles, and reasoning from the principle of locality in solving a problem. Causal reasoning relies on knowledge concerning how the behavior (or misbehavior) of one component affects the behavior of another component. Reasoning from first principles relies on the laws of physics or mathematics to predict or explain behavior of a system. The principle of locality considers how components are connected (mechanically, electrically, physically) in determining how behavior of one component can be influenced by another component.

The strength of model-based systems

lies in their fundamental knowledge of the domain. This allows the system to reason about situations previously unencountered, and for which the system has not been explicitly programmed. The type of knowledge representation also allows the knowledge to be transferred to other tasks; that is, a model-based system originally developed for diagnostics would be of significant value when developing a design expert system for the same domain whereas knowledge from a diagnostic rule-based system is of questionable value.

The weaknesses of a model-based system become apparent only when it is a pure model-based system; that is, when no heuristics are used. Diagnostic searches through components of systems based on the principle of locality make sense only if all components are equally fallible and equally accessible. Otherwise, rules are essential. In fact, many cases can be shown where model-based reasoning requires more time than rule-based systems. For this reason, most model-based systems will include at least a few rules to improve performance.

In a subsequent section we argue for blending these four reasoning methods to produce a synergistic effect. We claim a modified blackboard is a suitable framework for merging the individual reasoning methodologies. The traditional blackboard architecture is discussed next to provide a foundation for these arguments.

Blackboards

The blackboard architecture was developed for a speech understanding system developed in the Seventies known as Herasay-II [11]. Since its development, the blackboard model has been proven a robust model of problem solving on applications from ocean surveillance (HASP) to air traffic monitoring (TRICERO).

The traditional blackboard model contains three major components as shown in Figure 3 [12]:

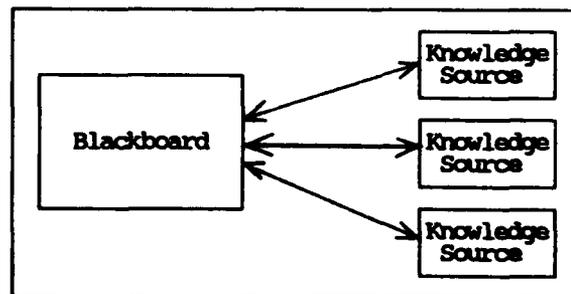


Figure 3. The Traditional Blackboard Architecture [12].

(1) The knowledge sources - The knowledge needed to solve the problem is partitioned into knowledge sources, which are kept separate and independent.

(2) The blackboard data structure - The problem solving state data are kept in a global data store, the blackboard.

Knowledge sources produce changes to the blackboard, which lead incrementally to a solution to the problem. Communication and interaction take place solely through the blackboard.

(3) Control- The knowledge sources respond opportunistically to changes in the blackboard. The structure of the control is left open. It can be in the knowledge sources, on the blackboard, or a separate module.

Although implementations vary, knowledge source activity is usually event driven. Each change to the blackboard constitutes an event that in the presence of specific other information on the blackboard can trigger one or more knowledge sources. The control mechanism selects a single knowledge source to execute its action on each problem-solving cycle. The control mechanism may use a variety of criteria such as the credibility of the knowledge source's triggering information, the reliability of the knowledge source, or the importance of the solution it would generate. When a knowledge source is triggered, it will typically produce new blackboard events. These events may in turn trigger other knowledge sources [11].

Blackboard systems construct solutions incrementally. On each problem-solving cycle a single knowledge source executes, generating or modifying a small number of solution elements. Some elements are assembled into growing partial solutions; others may be abandoned. Ideally, elements are eventually assembled into a complete solution.

As an illustration of how a blackboard uses cooperating knowledge sources, we can compare its operation to an aircraft accident investigation board. Whenever an Air Force aircraft crashes, an investigation is performed to determine the facts surrounding the case. After the investigation is complete, a board is held. The board consists of the board chairman and specialists who may be able to help determine the cause of the accident (e.g., pilot, navigator, aircraft mechanic). The chairman lists the facts on a blackboard in the front of the room. He then asks for comments from the group.

If someone in the room has information to add, they raise their hand. The chairman calls on one of the people with their hand raised, and that person adds new information (facts or hunches) to the board, then returns to their seat. Based on this new information more hands may go up, and some of the hands that were up may go down. The chairperson again selects someone. This person may add new facts or hunches, or may refute earlier hunches by other members of the board. The session continues until the cause of the accident is determined, or until the group can contribute no new information.

The parallel to a blackboard is obvious. The chairman is the control associated with the blackboard. The panel members are the knowledge sources, each

with a particular domain. The blackboard in the front of the room is the blackboard structure of the architecture.

The Synergistic Reasoning Approach

Each of the reasoning methods previously discussed has associated strengths and weaknesses. Merging the methods in the proper fashion could create a system that would benefit from the strengths of each of the methodologies while minimizing their weaknesses. It is postulated that a such a blend would result in a synergistic effect, allowing a system to solve problems that could not be solved by any of the individual reasoning methodologies.

Such a synergistic blend is possible by making a fundamental modification to the blackboard problem-solving approach. The essence of the modification is to partition the system based on reasoning methodologies rather than knowledge modules. Indeed, it may be more appropriate to partition the system based on methods of reasoning. Consider a person taking a (closed book) test. All knowledge to be used during the test is self contained. There is no reason to believe that the knowledge is partitioned into different modules within the person. However, the person may use several different methods of reasoning about the problem. These methods of reasoning can be modeled as "reasoning modules."

This approach will produce a synergistic effect by allowing the modules to focus their individual strengths on a problem. By cooperating through the blackboard and posting partial solutions, a problem that could not be solved by any of the systems individually could be solved by the system as a whole. That is, one reasoning module could post a partial solution not obtainable by any of the other modules, and while it might not be able to generate the desired solution, one of the remaining modules, which was also unable to generate the desired solution from the original problem, might be able to do so based on this new result. A second synergistic effect is expected from the ability of one reasoning system to refute conclusions of another reasoning module. A diagram of the synergistic reasoning system is shown in Figure 4.

Satellite Autonomy

Satellite Autonomy is a suitable application for the proof of concept of the SRB. Satellite control is a complicated, tedious, and labor intensive process. According to a 1989 GAO study, over 4,000 government and contract staff are required to operate the Air Force Satellite Control Network consisting of fixed ground-based tracking stations, central control facilities, and communication links [13,14]. This network currently controls the operations of approximately 80 on-orbit satellites. Predictions are that 135 satellites will be on-orbit by the year 2000, and 150 will be on-orbit by 2015.

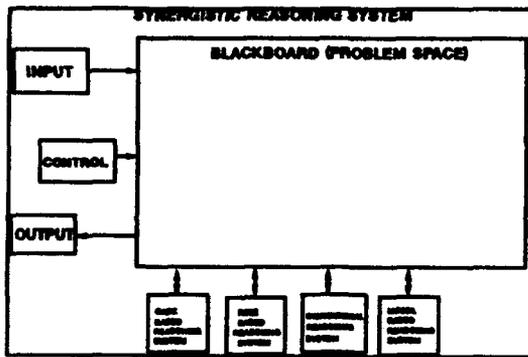


Figure 4. The Synergistic Reasoning System

However, the number of controllers supporting the network is likely to remain constant while the level of expertise decreases due to retirements [15].

As early as 1985, AI technology was identified as capable of supporting high levels of satellite autonomy. A Jet Propulsion Laboratory study identified specific applications that could benefit from one or more AI techniques [16]. As a method of meeting the goals of satellite autonomy, it is possible to develop autonomous subsystems for the satellite. Autonomy for each subsystem would be valuable in its own right, and could serve as components of a fully autonomous satellite. Subsystems on-board that were identified as well-suited for autonomous operations include: (1) guidance, navigation, and control, (2) power systems, (3) thermal control, and (4) payload management.

The same 1985 report defined eleven levels of satellite autonomy which are shown in Appendix A. Current satellites operate at about Level 3. The goal of an application associated with the synergistic reasoning system would be to provide a satellite with the salient characteristics associated with Level 5 operation, specifically, autonomous fault tolerance for operations in the presence of faults specified a priori. This capability will employ spare system resources, if available, or will maximize mission performance based upon available capability and/or available expendables without ground intervention.

Satellite autonomy is well suited as a domain for testing the Synergistic Reasoning System. Current control of satellites is performed through the sole use of conventional programs and frequent human intervention. A significant increase in effectiveness in satellite operations is expected from implementation of a synergistic reasoning system since performance improvements are documented for rule-based and model-based systems on similar domains. While there is less

experience with case-based systems, the unpredictable behavior of spacecraft often leads to problems which are currently solved by relying on past cases.

Figure 5 is a diagram of a Synergistic Reasoning System for satellite autonomy. Note that human intervention has been modeled as a separate reasoning module. The blackboard lists the satellite subsystems. The individual expert systems are shown from left to right according to the depth of their reasoning methodology. An implicit control line runs from the control module to each of the components of the system. A proof of concept in this domain would be limited to a single subsystem, such as the fault diagnosis and recovery of the autonomous navigation system.

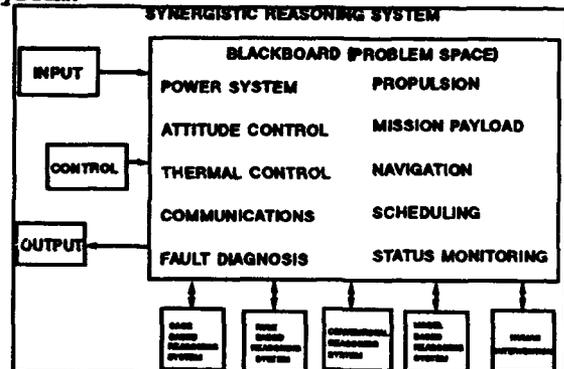


Figure 5. SRS for Satellite Autonomy

Satellite Autonomy Sample Operation

As an example of how the reasoning system might produce a synergistic effect from the proposed architecture consider the following scenario:

- I. During normal operation, the conventional program performs routine operations.
- II. The signal from the ground station falls below a predetermined level.
- III. A message is posted to the blackboard by the conventional system that a problem exists.

"signal weak from ground station"

IV. The executive module determines that the problem is diagnostic in nature and does not require immediate human intervention. The rule-based reasoner is activated.

V. The rule-base tries a series of data driven quick fixes.

IF <Signal Weak>
THEN Increase Gain.

IF <Gain Increased> and
<Signal Weak>
THEN Calibrate Pointing.

IF <Gain Increased> and
<Pointing Calibrated> and
<Signal Weak>
THEN Check Components <Sensor 1>.

VI. The executive module acknowledges the inability of the rule-based module to correct the problem. The model-based reasoner is activated.

VII. A model of the Sensor system is constructed:

Component1: Antenna
Component2: Amplifier
Component3: Phase Shifter
Component4: Power Supply
Component5: Ground System Output

VIII. The suspected components are posted on the blackboard. At this time the rule-based module is used again to determine the order of search. The rationale is that the blackboard may contain additional information on the current situation which will modify the corrective actions of the system. For example, if multiple subsystems were experiencing difficulty, there would be reason to suspect the power supply. In this instance, the rule-based reasoner adds no additional information and the model-based reasoner is reactivated.

IX. The model-based reasoner then examines each of the components and determines that all components are sound. The results are posted to the Blackboard.

Signal Weak From Ground Station.
Rule-base Quick Fix Tactics Unsuccessful.
Components of Communication System
Verified Sound.

X. The executive module activates the case-based reasoner to determine if a similar event has previously occurred. The case-based system finds a match with a previous event:

Problem:
Ground Station Reports Weak Signal.
Rule-base Quick Fix Tactics Unsuccessful.
Components of Communication System
Verified Sound.

Repair:
Fault in Attitude Control System.

XI. Check Attitude Control System is posted to the Blackboard. The model-based system builds the following components:

Component1: Attitude Control Electronics
Component2: Earth Sensor
Component3: Sun Sensor Assembly
Component4: Rate Gyro
Component5: Reaction Wheel
Component6: Solar Array Switch

XII. Components of the model are diagnosed and it is determined that the rotation speed of one of the four reaction wheels is low. The faulty rotation wheel is turned off and the blackboard is sent the message:

Fault detected in momentum wheel #2.

XIII. The Rule-based system would use data driven tactics in an attempt to solve the problem, such as recycling power to the wheel. If these attempts are unsuccessful, a message stating that it will be necessary to develop new algorithms to work around the fault would be sent to the mission controller at the ground station (modeled as "Human Intervention").

Future Improvements

Two extensions to the Synergistic Reasoning System are proposed: (1) the addition of more reasoning modules and (2) a feedback system to allow controlled modification of the modules based on past sessions. This extended system is shown in Figure 6.

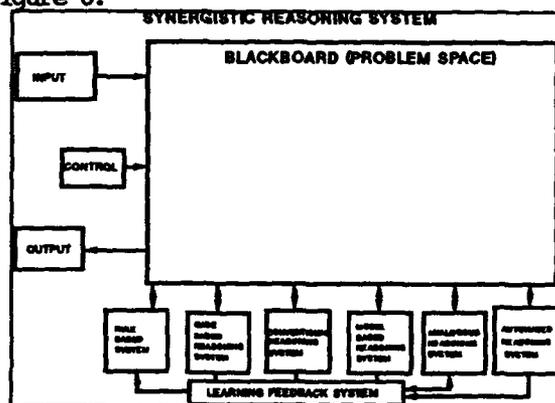


Figure 6. The Extended Synergistic Reasoning System.

The four reasoning methodologies used in this research can cover most common problem instances. To be successful, however, they require that either rules, past cases, algorithms, or models exist for the problem domain. To be successful in domains for which such information is not available more powerful reasoning methodologies are required. Two such reasoning methods are analogical reasoning and automated reasoning. While it is not expected that analogical and automated reasoning systems would be used as often as the other forms of reasoning, it is conceivable that they could prove valuable. For example, they could provide considerable value in deep space missions in which previously unencountered situations occur, and human intervention is hampered by the amount of time required by long distance communications.

The second extension is to modify the traditional architecture to allow the

controlled modification of one reasoning module based on results from one or more of the other reasoning modules. This violates one of the defining features of the blackboard architecture which is that the knowledge sources are totally independent of each other, and cannot influence each other directly [11].

The controlled modification would be in the form of a feedback system that allow the contents of a module to be modified based on the results of previous sessions that involved other reasoning modules. For example, the system would allow a solution resulting from use of the model-based system to be added as a rule to the rule-based system. In addition, the results of any session (including unsuccessful sessions which required human intervention) would be stored as a new case in the case-based reasoning system.

This feedback system could also be used to improve the performance of the analogical reasoning system. When analogical reasoning is employed, the result is positive mappings (those attributes that are confirmed to correspond between source and target), negative mappings (those attributes that are confirmed to not correspond between target and source), and neutral mappings (those attributes whose correspondence has yet to be confirmed). The feedback system could postulate a theorem that a neutral attribute has a positive (or negative) mapping. It could then use the automated reasoner to prove or disprove the theorem, leading to an increase in confidence in, or denial of, our analogy.

Bibliography

1. Skinner, James M. A Diagnostic System Blending Deep and Shallow Reasoning, Proceedings of the Ninth International Workshop on Expert Systems and Their Applications. Avignon, France, 1989.
2. Luger, George and William A. Stubblefield. Artificial Intelligence and the Design of Expert Systems. Benjamin/Cummings. Redwood City, CA. 1989.
3. Searle, John. Minds, Brains, and Science. Harvard University Press, Cambridge, Mass, 1984.
4. Torrance, Steve. The Mind and the Machine - Philosophical Aspects of Artificial Intelligence. John Wiley & Sons, Chichester, 1984.
5. Weizenbaum, J. Computer Power and Human Reason San Francisco: Freeman, 1976.
6. Harmon, Paul and Brian Sawyer. Creating Expert Systems for business and Industry. John Wiley & Sons, Inc. 1990.
7. Iwasaki, Yumi. The Handbook of Artificial Intelligence Volume IV. Edited by Avron Barr et al. Addison-Wesley Publishing Company, INC. Reading, Mass. 1989.
8. DARPA. "Case-Based Reasoning from DARPA: Machine Learning Program Plan", Proceedings: Case-Based Reasoning Workshop. Morgan Kaufman Publishers, San Mateo Ca 1989.
9. Waterman, Donald, A Guide to Expert Systems. Reading MA: Addison Wesley Publishing Company, 1986.
10. Davis, Randall, et al., "Diagnosis Based on Description of Structure and Function" Proceedings of the National Conference on Artificial Intelligence, pp. 137-142, 1982.
11. Hayes-Roth, Barbara. "Blackboard Systems," The Encyclopedia of Artificial Intelligence. Edited by Shapiro. 1989.
12. Nil, Penny. The Handbook of Artificial Intelligence Volume IV. Edited by Avron Barr et al. Addison-Wesley Publishing Company, INC. Reading, Mass. 1989.
13. GAO Report to the Chairman, Subcommittee on Defense, Committee on Appropriations, House of Representatives, Military Space Operations - Shuttle and Satellite Computer Systems Do Not Meet Performance Objectives. August 1988.
14. GAO Report to the Chairman, Subcommittee on Defense, Committee on Appropriations, House of Representatives, Military Space Operations - Operational Problems Continue With the Satellite Control Computer System. August 1988.
15. Skinner, James M. and Richard Higgins Jr. Employing Artificial Intelligence to Meet Space Requirements, Proceedings of the Twenty-seventh Space Congress. Canaveral Council of Technical Societies. Cocoa Beach, Florida, 1990.
16. Jet Propulsion Laboratory. Autonomous Spacecraft Program - Technology Development Tasks for Spacecraft High-Level Autonomy. Jet Propulsion Laboratory, California Institute of Technology, Pasadena, California, 1985.



COMPONENT PART NOTICE

THIS PAPER IS A COMPONENT PART OF THE FOLLOWING COMPILATION REPORT:

(TITLE): Conference Proceedings of machine Intelligence
for Aerospace Electronic Systems Held in Lisbon, Portugal on
13-16 May 1991 (L'Intelligence Artificielle dans les Systemes
Electroniques Aero spatiaux)

(SOURCE): Advisory Group for
Aeronautical Research and Development, Neuilly-sur-Seine
(France)

TO ORDER THE COMPLETE COMPILATION REPORT USE AD-N242025

THE COMPONENT PART IS PROVIDED HERE TO ALLOW USERS ACCESS TO INDIVIDUALLY AUTHORED SECTIONS OF PROCEEDINGS, ANNALS, SYMPOSIA, ETC. HOWEVER, THE COMPONENT SHOULD BE CONSIDERED WITHIN THE CONTEXT OF THE OVERALL COMPILATION REPORT AND NOT AS A STAND-ALONE TECHNICAL REPORT.

THE FOLLOWING COMPONENT PART NUMBERS COMPRISE THE COMPILATION REPORT:

AD#: _____ TITLE: _____

- AD-P006 331
- AD-P006 334
- AD-P006 336
- AD-P006 344
- AD-P006 345
- AD-P006 346
- AD-P006 352



Information on this report is available for public use and should be distributed to all users.

Accession for	
NTIS	DTIC
DTIC TAB	Unannounced
Justification	
By _____	
Distribution /	
Availability Codes	
Dist	Avail and/or Special
A-1	

SPACECRAFT ELECTRICAL POWER SYSTEM
 FAULT DETECTION/DIAGNOSIS AND RESOURCE MANAGEMENT

Peter Adamovits
 Canadian Space Agency,
 Computer and Intelligent Systems Group,
 3701 Carling Avenue,
 Ottawa, Ontario, K2H 8S2,
 Canada,

Eric Jackson
 International Submarine Engineering Limited,
 1734 Broadway Street
 Port Coquitlam, British Columbia, V3H 1X1
 Canada

Artificial Intelligence

Abstract

The paper discusses the domain of Electrical Power Systems for Radar Spacecraft and its effect on the design of AI-based systems for health monitoring and resource management. The paper presents work on Electrical Power Systems (EPS) Fault Handling and Resource Planning. The Fault Handling System relies on model-based reasoning techniques while the Resource Planning System relies on case-based reasoning techniques. The two systems perform their functions in a cooperative way and are distributed over both ground-based and space-based processing elements.

1 Introduction

As spacecraft and space systems become larger and more complex, the demands on a traditionally structured ground-based operation to monitor and control the various subsystems increase dramatically. The high operational management requirements become readily apparent in the radar spacecraft application domain discussed. Canada is building a civilian radar spacecraft 'Radarsat' and is participating in a research and development program for an advanced military surveillance spacecraft 'Space Based Radar' (SBR).

The radar payload of this class of spacecraft places demands on the electrical power system significantly different than those of civilian communication relay spacecraft. The Electrical Power System (EPS) for radar spacecraft must deal with a highly inclined low earth orbit radar payload having tens of thousands of RF transmit / receive elements demanding between 5 and 30 kW of electrical power [Eatock 90]. In the example

presented, communications opportunities may be infrequent and of short duration; an eclipse may occur each orbit; the system is expected to operate with failed or degraded components; and short term power demands will exceed the nominal power output of the solar panels.

The remaining sections of the paper present an overview of the domain of radar spacecraft and the electrical power system is described in sections 2 and 3; Sections 4 and 5 describe the Ground Segment and Spacecraft Operations; Section 6 reviews the needs for automation and spacecraft autonomy; Section 7 presents an Artificial Intelligence based approach to planning and diagnostics; Sections 8, 9 and 10 describe previous developments and the present status on diagnosis and planning work and; the paper is summarized with section 11.

2 Radar Spacecraft Overview

2.1 Mission and Orbit Mechanics

The purpose of a radar spacecraft is to survey predetermined areas of the earth as it passes over them. The major constraints on the mission are the orbital coverage of the earth, the payload and housekeeping energy requirements, and the available energy on the spacecraft. Inevitably, there will be times when more radar exposures are desired than can be achieved.

The requirements for the selection of an optimal orbit for a radar spacecraft are significantly different from those for communications or meteorological spacecraft. A radar spacecraft is typically required to scan the complete earth over some period, while communications spacecraft are

AD-P006 331



typically geostationary. In order to map out a large percentage of the earth, the orbit plane of a radar spacecraft is highly inclined to the earth's equatorial plane. Retrograde orbits (i.e. inclination greater than 90 degrees) are common. With any inclined orbit, the orbit plane precesses about the earth at a rate dependent on: the angle of inclination (decreasing with inclination), the radius and ellipticity of the orbit. Also, as the spacecraft orbits the earth, the earth spins on its axis beneath the spacecraft. These factors define the payload coverage, the frequency and duration of contact possible (spacecraft to ground stations), and the frequency and duration of eclipse operation.

2.2 Payload and Housekeeping Energy Requirements

Once the orbital mechanics have been determined from the mission requirements, many constraints defining the spacecraft system, including the ground support and the EPS requirements, will be fixed. A typical spacecraft of this class is the Canadian Radarsat. In this case, the following preliminary estimates apply:

- Orbit altitude: approximately 800 km,
- Orbit period: approximately 101 minutes,
- Eclipse: maximum 18 minutes, 73 days long season centred on summer solstice,
- Orbit is retrograde: spacecraft is nominally passing in a direction opposite the earth's spin.
- Single ground station contact: 12 minutes contact maximum (at 5 degrees elevation above horizon); typically 3 orbits of contact totalling 29 minutes duration (or less) followed by 4 orbits of no contact. Cycle repeats twice every day.

Earth observation spacecraft tend to be sun synchronous. That is the spacecraft orbit plane is at a constant angle to the sun vector at any time of the year (ignoring earth declination). Most spacecraft of this class have day-night orbits (i.e. they observe both sunlit and dark sides of the earth in a single orbit). While also sun synchronous the Radarsat spacecraft has a dawn-dusk orbit. Space Based Radar is not expected to be sun synchronous and as a result will experience significantly different ground contact and eclipse characteristics.

The factors that affect the design of the EPS include the duration and frequency of eclipse operation, the power requirements and duty cycle of the payload and housekeeping functions, the

duration and frequency of contact with ground station(s) and/or the requirement for relay spacecraft. Contacts with a single ground station are infrequent and of short duration. Contact duty cycle can be improved significantly by using multiple ground stations or other spacecraft to relay the signal to ground.

2.3 Spacecraft Mission Planning

Planning the spacecraft mission will be considered from the payload requirements viewpoint. Following the attainment of final orbit, the mission requirements define areas of interest on the earth's surface. The radar image start times, duration and quality define the radar power profile needed to attain the images required. The ideal radar power profile will be translated to the required EPS power profile. An assessment will be made of the available electrical power to attain the EPS power profile. If the available power is not sufficient due to weak batteries, solar eclipse, component failures, or other reasons; the mission plan will be altered by dropping survey areas of lesser interest on the earth's surface. This is accomplished iteratively by ground-based mission analysts.

3 Domain Description: Spacecraft Electrical Power System

The Electrical Power System includes a prime power source (solar panels), storage elements for eclipse operation (batteries), power distribution busses, control electronics (charge controllers, discharge controllers, power conditioners) and distribution equipment (switches, relays and circuit breakers). The EPS provides power to the spacecraft loads which have a variety of power profile characteristics. Duplication of component and the ability to configure the power distribution network is used to provide a high level of operational capability. While the operational capability is increased, the total power available to the spacecraft loads may be reduced when components have failed (figure 1).

The EPS must provide electrical power through all phases of operation: launch, transfer orbit and final orbit. Typically, through launch and transfer orbit operation, the EPS must maintain only the loads needed to attain the final orbit. After launch and the attainment of final orbit, the solar panels will be deployed and the various housekeeping and

payload functions will be tested. The test and verification process typically requires a few weeks to complete.

Following attainment of the final orbit, the power requirements of the spacecraft are met both by the solar panels and batteries. During sunlit operation, the solar panel power output is used for battery charging, housekeeping and radar payload requirements. During solar eclipse, only batteries are available to meet the power requirements of the spacecraft.

The characteristics of the radar payload significantly affect the design of the EPS. Short term power surges required by the payload must be met by either batteries or other short term storage elements (capacitors). The power required by the radar during transmission may exceed the solar panel capacity by up to 100%. The resulting duty cycle averaged over the orbit is on the order of 50% [Moody 89]. Design constraints may affect this level during eclipse operation.

The secondary loads in the form of Attitude and Orbit Control, Telemetry Tracking and Command, Thermal Control, etc. are one or two orders of magnitude lower in power requirements than the radar payload. The on/off time characteristics of many of these loads are similar to a random or stochastic process. While these loads are equally important to the payload in terms of spacecraft health and effectiveness, the demands in terms of electrical power are lumped together as a single load which can be modeled as a stochastic process.

4 Overview of Ground Segment/Data Acquisition System

In the Radarsat project the mission control ground station (Ground Segment) and the radar image data ground station (Data Acquisition System) are not co-located and may in fact be located in geographically distant locations. The actions performed are:

- The Ground Segment performs health monitoring, management and control of the spacecraft. This includes attitude and orbit control, EPS control, radar payload control, control of the recorded image playback to the Data Acquisition System etc.
- The Data Acquisition System is responsible for receiving and the initial processing of the radar image data.

Through increased on-board processing, portions of each activity may be completed on the spacecraft.

Although both Ground Segment and Data Acquisition Systems are of equal importance, the following discussion addresses the needs of the Ground Segment only. As noted in section 2.1, contact with a ground station (Ground Segment or Data Acquisition System) will be a maximum of 12 minutes for an orbit where contact is possible. During this period the following events will occur:

- The spacecraft will appear at the horizon.
- The ground station antenna must locate the spacecraft (typically accomplished by the time the spacecraft reaches a point 5 degrees above the horizon).
- The spacecraft is tracked through its trajectory to a point near the opposite horizon (5 degrees above).
- When contact has been made, the spacecraft will transmit subsystem sensor readings (telemetry) indicating the status and health of the spacecraft. (Data Acquisition System will instead receive radar-image data at this point.)
- The Ground Segment will process portions of the subsystem status and health data and transmit a command stream to take corrective actions that may be necessary.
- The Ground Segment will also transmit the payload command stream (schedule) that will be used to control the radar system for the next period (typically 24 hours).

These last two actions will likely be completed on the next orbit pass (i.e. 101 minutes later) following initial processing of the status and health data. Thus two consecutive passes are needed to receive telemetry and transmit the spacecraft command stream/payload schedule to the spacecraft.

The Ground Segment and the Data Acquisition System both benefit from additional ground stations and/or relay spacecraft. Contact periods can be extended significantly with these additional communications paths. The trade-off is the required bandwidth and number of ground stations versus the cost and maintenance of a relay spacecraft network.

It must be recognized that the Data Acquisition and Ground Segment systems serve very different requirements. Here we are concerned with the health and control of the Electrical Power System

91-15513



91 1110 008

(through the Ground Segment). The primary objective of the spacecraft is however to provide radar images.

For civilian spacecraft the need to maintain a continuous radar-image data to the ground is not present. Delays on the order of days may not be critical in terms of the radar image information gathered. As a strategic defense issue, the availability to provide near continuous coverage is more critical.

5 Spacecraft and Ground Segment Operations

Normal mission operations of a radar spacecraft are as follows: on a given orbit the spacecraft can be expected to pass over an area of interest; at that time the radar will be activated and image data will be acquired. As noted in Section 3, the capacity of the solar panels is usually exceeded through these times by a significant level. The reliance on batteries and other short term storage devices causes the management of power storage to be intricately tied into the mission planning aspect of the spacecraft (see section 2.2).

Power Storage Management and Mission Planning operations are not automated in an intelligent systems way and require the attention of a large number of subsystem experts and designers [Adamovits 1990]. Traditional ground control techniques rely on subsystem operations personnel, each controlling a single subsystem of the spacecraft. With a polar orbiting spacecraft, long periods of little or no contact can be followed by short periods of intense activity. While this approach may be expected to remain for some time in spacecraft systems, the reliance on automation in the ground-based system and autonomy of the space-based system is expected to increase.

Ground-based activities during normal operations are biased toward mission planning, station keeping and health monitoring. Station keeping activities refers to orbit maintenance: inclination, altitude, eccentricity etc. Health monitoring includes: detecting component degradations and failures, observing trends in system and subsystem performance, etc. All the above activities are expected to be automated in increasing degrees in future spacecraft.

Failure recovery through all phases of operation is usually accomplished by a minimal degree of designed-in overcapacity and through the configurable nature of the system. During the launch and transfer orbit operations, the spacecraft relies on battery power and minimal amount of solar panel power. EPS faults in general and battery faults in particular can seriously jeopardize the entire mission. Following deployment of the solar panels, the EPS is more capable of dealing with failures of individual EPS components.

The requirement for automation is defined at many levels. In many cases the requirement stems from the lack of communication opportunities. In normal operation the spacecraft is expected to operate unsupervised for periods of up to 10 1/2 hours in duration. Potential problems with the Ground Segment system extend this time requirement significantly. A 'survivability' requirement of 7 to 14 days of no contact is also required. During this survival period the spacecraft is required to deal with health and normal subsystem management issues through its on-board processing elements.

6 Automated Operations: Space-Based vs Ground-Based

Two fundamental reasons to automate systems are to assist humans in handling complexity and to achieve critical (fast) time responses. Automation of mission expertise is very desirable for assisting humans in the complex spacecraft operations planning and management as well as in health monitoring. Fault detection and response is an obvious area where automation is desirable in order to achieve critical time responses. As noted previously this is especially true when contact is infrequent and communication is short in duration. In general, automation to assist humans in handling complexity can be a Ground Segment based function, and automation to achieve critical time responses should be space-based.

Space-based automation is termed autonomy in this paper. In this discussion, autonomy does not mean that the system is not controllable by humans, but that it can operate with some known level of capability in the absence of supervision for as long as is necessary (several hours through to several days). The system can be overridden or reconfigured by humans. In the present case, the level of interaction possible with Ground Segment

support systems is limited due to infrequent telemetry.

The factors that drive the time response requirement are safety concerns and mission concerns. Safety concerns are due to faults, i.e. *their expected criticality or potential of criticality*. Faults may shorten the life of the spacecraft or decrease the capability of the spacecraft and therefore must be detected, diagnosed, and dealt with quickly.

Mission concerns that drive the time response requirements are the importance of obtaining timely data. Questions that must be asked are:

- When is the data needed?
- When is it available?
- Will the same data be available in N orbits?
- Can it be relayed to earth before then?
- What is an acceptable cost to obtain the data?

If the requirement for timely data is high enough the system must be expected to operate in the *presence of faults*. On-board systems must be able to detect, diagnose, and recover from faults autonomously, and we must accept the risks associated with this approach.

The telemetry link is the major factor in determining whether processing occurs on-board or in ground based systems. Relevant telemetry link parameters include bandwidth, time delay, and availability. All of these parameters may differ for the uplink and the down-link. They will determine, for example, whether or not there is time to perform fault handling functions on the ground while maintaining an acceptable amount of data gathering.

While time response is the driver for autonomy, *other factors oppose increased autonomy*. They are: the prohibitive costs of installing processing capacity on-board the spacecraft and the degree of faith in the capability of the autonomous systems. A lesser factor is the level of instrumentation available on-board. To a limited extent more instrumentation can translate into a reduced requirement for on-board processing to detect and isolate faults. With increased instrumentation, mass and overall system complexity concerns resurface.

The goals of increasing on-board autonomy are limited by the degree of on-board computational capacity available. This is due to the fact that, installing computational capability on-board a spacecraft is very expensive in terms of mass and power consumption. In addition, the requirements

for radiation hardening, space-rated components, and redundancy greatly increase the mass and power consumption of space-based computers over equivalent terrestrial computers.

Considerable reluctance by spacecraft designers and operations personnel can be expected for the use of embedded systems for higher level cognitive aspects of problem handling. To illustrate, terrestrial expert systems for diagnosis will include an Explanation function, so that the operator can check the computer's results against his own judgement. This scenario is not practical here. A compromise is to implement a system that performs fault detection and module level (rather than component level) diagnosis autonomously, and then implements a predetermined, conservative response. This may leave the spacecraft partially operable or as a minimum in a 'safe' mode. At the next available point the data is sent to the ground for processing by ground personnel and computers, analyzed, and a recovery plan is uploaded.

A similar conservative approach can be generated for autonomous replanning. A mission which is uploaded to the spacecraft, consisting of a series of areas for imaging can contain associated "importance" factors to describe the importance of each section of data. The autonomous controller, in the absence of supervisory communications, can determine whether or not to attempt to obtain each section of data by its relative importance together with the predetermined risk associated with a partial fault diagnosis.

7 Artificial Intelligence Based Approach to EPS Diagnostics and Planning

Previous works in diagnosis and planning are being combined to form a system that addresses the Fault Detection/Diagnosis and Resource Management aspects of a radar spacecraft EPS. Real-time fault detection/model based diagnosis was explored in phase 1 of the Health Monitoring Power Management and Control System project (Section 8). Case-Based reasoning applied to resource management was investigated in the Case-Based Planning System project.

The resulting system will combine both model based and case-based techniques to address the autonomy and controllability requirements of the EPS. The composite approach relying on both

model-based and case-based techniques was taken in order to provide a system capable of dealing with each of the widely varying domain constraints described previously.

In the design described in sections 8 and 9, the computational load is distributed between on-orbit (fault detection/diagnosis and replanning) and ground-based personnel and computer facilities (diagnosis and planning). The system design thus provides increased on-board autonomous fault isolation and plan failure recovery in addition to ground-based logistics support in complex fault diagnosis and resource management.

In the fault handling system presented, a model-based fault detection/diagnosis module is connected directly to the EPS sensors and determines emergency responses in real time. It is constructed to generate emergency responses (i.e. before the diagnosis process is completed). This capability is useful for complex faults where a complete diagnosis may take a significant amount of time or may require a probabilistic analysis. In such cases, the possibility of certain types of dangerous faults may be indicated early in the diagnostic process, and an emergency response may be generated at that time. The fault detection/diagnosis system addresses a continuum of faults from single and multiple component degradations through catastrophic component failures. The output of the system is used to isolate failed components and to provide input to an adaptive planner.

The planner and dynamic replanner use case-based reasoning techniques to create, revise and maintain a case library of successful operational plans. These plans are continually updated as the operational constraints of the environment and the EPS change.

The described approach to EPS Planning and Fault Handling has been successfully demonstrated on representative electrical power circuit models and is being extended to address the entire EPS. The completed diagnostic and planning system will be demonstrated on a large scale software simulation and a breadboard hardware simulation of a representative radar spacecraft EPS [Eatock 90], [Moody 89].

8 Description of Health Monitoring Power Management and Control System (HMPMCS) Phase 1

In the previous phase of the project, a proof-of-concept real-time Power Usage Analysis and model-based Fault Handling system was demonstrated [ISE 89]. For that demonstration, a real-time Power Usage Analysis, Fault Detection and Emergency Response system and an off-line Fault Diagnosis system were coupled to a simulated spacecraft EPS. Power Usage Analysis, model-based Fault Detection, and Emergency Response generation were performed using an object-oriented control system package [Zheng 1989], while complete Fault Diagnosis was performed by an off-line system based on a commercial expert system shell. The control system package was also used to generate the EPS simulation. Two computers connected by a serial link were used in the demonstration, one for the real-time software and the other to run the expert system shell (figure 2).

A simple spacecraft electrical power system was simulated using a real-time control system package with an update rate of 200 Hz. The simulation included a power distribution bus and resistive-inductive loads controlled by switch/circuit breakers. Sensor and actuator faults were both easily injected into the simulation. Single and double fault scenarios were tested and were used by the real-time fault detection/emergency response system and by the off-line fault diagnosis system.

A Power Usage Analysis function was implemented. It was based on a schedule of events, each of which had an expected power profile. The power usage of the simulated EPS was checked against the schedule on every boolean command and data event and anomalies were detected. Future power usage and potential hazardous conditions were then predicted based on the schedule and the detected anomalies.

A Model-based Fault Detection and Emergency Response system was also implemented on the real-time system and was run at 100 Hz. It monitored the sensors of the EPS and commands for switch closures and compared the expected behaviour with the resulting sensor readings from the simulator. Anomalous behaviour was detected and categorized, and emergency responses were generated when necessary.

The Off-line Fault Diagnosis system was implemented using an expert system shell (KEE) and was run on a Unix workstation (Sun Sparc 1). Simulator inputs were received over a serial link at 1 Hz and buffered (using Unix pipes) to the application. A constraint propagation model was implemented and linked to an Assumption-based Truth Maintenance System. Single and double fault analyses were demonstrated.

8.1 Lessons Learned and Future Development Directions for the HMPMCS

The phase 1 implementation of the HMPMCS system demonstrated the suitability of the approach to spacecraft EPS fault detection and diagnosis. Specifically, the division of responsibility between a real-time fault detection system from the more heuristic based diagnostic system proved successful in diagnosing both single and multiple faults. During the project development a number of lessons were learned:

- The real-time component using only a shallow knowledge base often initiated 'safing' actions before the off-line system could detect or react to its input data. The data that the off-line system was operating on was no longer relevant to the present state of the system. The out of phase of nature of the operations may be resolved through a meta-level reasoning system or other techniques.
- The reaction time of the off-line system often hindered the overall system performance. Alternate AI development tools and techniques will be examined to improve this components run time performance.
- The application circuit that was diagnosed in phase 1 provided significantly more data than is feasible for a full spacecraft level EPS. This simplified the design of the HMPMCS components. Phase 2 will require more sophisticated techniques to offset the reduction in sensors. This is expected to be accomplished through an increased deep reasoning.

Other upgrades to the Phase I work will include:

- Implementing a more closely coupled fault detection and diagnosis system based on object-oriented techniques,
- Investigation of probabilistic fault diagnosis,
- Link fault diagnoses to Power Usage Analysis,
- More effort on trend analysis,
- Add planning/replanning segments, with the replanning function connected to fault handling and trend analysis,

- Connecting the system to a large-scale breadboard spacecraft electrical power system and,
- Identification of space based and Ground Segment based components.

9 Case-Based Planning System

The Case-Based Planning System (CBPS) project is exploring the application of case-based reasoning techniques to spacecraft electrical power system resource planning. The approach has foundations in the work of Hammond [Hammond 89] and recent developments at the Canadian Space Agency [Ulug 91a], [Ulug 91b]. The CBPS relies on both space-based and Ground Segment-based processing and case library storage elements. The CBPS performs plan selection, revision, creation, evaluation and dynamic replanning. Many of these are performed in a cooperative way in both space based and ground environments as described below.

9.1 Task Definition

The CBPS receives a list of requirements for a set period of the mission in terms of individual task power/time requirements and task to task interrelation constraints. For example:

- a) Turn on radar transmitter/receiver to scan the earth at orbit locations corresponding to time 21:36:19 through 21:43:22; 21:54:45 through 22:01:30.
- b) Recharge batteries prior to eclipse.
- c) Prepare for orbit correction: turn on thruster heaters.
- d) Perform orbit correction: at specific time turn on thruster heaters then fire thrusters, etc.

Tasks can have the following inter-task constraints:

- Tasks can occur concurrently (a, b).
- Task must follow another task (d must follow c).
- Tasks are mutually exclusive (recharge batteries and eclipse operation).

As defined in the CBPS, tasks use resources of two classes:

- Depletable resources (battery charge, hydrazine fuel etc.).
- Non-depletable resources (solar panel power, solar heating etc.). The CBPS will handle multiple resources (now 3) in the planning process

concurrently (electrical power, reaction thruster fuel, etc.).

9.2 Ground Segment Based Planning Components

The CBPS is provided a list of tasks and their corresponding constraints from the mission planner (operations personnel supporting Ground Segment activities) as identified above. Given these, the ground segment based components of the planner (see figure 3) performs the following actions:

- The selector attempts to locate an appropriate plan from a library of previously used or precalculated cases. If an identical plan is not available but a similar plan is, then this plan is modified by including additional tasks or removing extra tasks.
- If the selector cannot find or modify an appropriate case, then a planner is invoked to create a new unique plan by applying constraint satisfaction techniques on the task constraints that have previously been provided.
- Following the selection of an existing plan from the library or the generation of a new plan, the plan is verified against the model of the EPS to ensure its appropriateness in the given operational environment.
- The plan is then submitted to a plan executor (in the ground segment based system this is a simulated spacecraft EPS).
- During the simulated execution, the plan may fail due to a variety of reasons: errors in selection criteria, errors in plan creation or changed operational characteristics of the components. If this should occur then a dynamic replanner is invoked. The replanner repairs the plan from the point of plan failure and forward in time. The executor is again invoked and the cycle continues.
- Following execution, the plan is evaluated for its success in meeting the required operational requirements of the mission. The library maintenance system is then invoked to record the information for later plan selection. (Information stored with the plan includes the environmental parameters existing during plan execution and the success in operating in this environment.)
- When an appropriate plan has been selected, modified or created, this information is transmitted to the spacecraft (at the next available time) for later execution in orbit. The transmitted information may include: the name of the appropriate plan (or the location in the space-based plan library), a list of changes necessary for a specified plan to be effective in the current situation, or a complete plan to be added to the library and used later.

9.3 Space-Based Planning Components

The space-based components of the CBPS are invoked based on the time schedule dictated by the orbit position of the spacecraft as it passes over areas of interest.

- Given the selection information provided by the ground system, the space-based selector retrieves a plan from the library and submits it to the executor.
- The executor initiates the plan at the appropriate time based on the orbit position and other factors (eclipse, battery status etc.). Many plans may in fact be executing concurrently at any time. Within each executing plan, many tasks may also be executing at a time.
- If the plan should fail, a space-based replanner is invoked to dynamically repair the plan or at least minimize the negative impact of the failure. (As the recovery is required in real time, the space-based replanner is biased toward safeing operations rather than searching for optimal solutions.)
- As with the ground segment-based system, the plan is then evaluated to identify its level of success in meeting the required mission goals. This information is transmitted to the ground segment-based system through the down-link. The ground segment-based system is responsible for library maintenance on both the ground segment-based and space-based components of the CBPS.

9.4 Lessons Learned and Future Development Directions for the CBPS

The Case-Based Planning System has been implemented and tested against a software simulated spacecraft EPS. In its present form, the CBPS performs all operations identified above for the ground segment; space-based components have not been implemented.

During previous phases of development a number of lessons were learned. Some of these are:

- The CBPS was implemented in Smalltalk V/286. This version of Smalltalk has proven quite robust and provides an effective development environment on which to develop prototype systems.
- Initial versions of the planning, replanning, selection and evaluation components were coded in a Smalltalk compatible version of Prolog. While this system was reasonably well integrated with the Smalltalk environment the run-time performance

was not acceptable. Alternatives are now being examined.

- Task and plan definitions and library encoding of the same has been an area of continued development. This is expected to remain for some time as more complex planners and replanners are developed. As an example, a complete redesign may be necessary to support hierarchical planning.
- An autonomous library maintenance facility may be needed to 'garbage collect' unused or seldom used plans. This will become a necessity for systems that operate over the life of a spacecraft (several years).

Presently the CBPS is being evaluated to identify the expected performance when scaled to a fully operational radar spacecraft electrical power system. If the evaluation presently underway is successful, the CBPS will be further developed and integrated with the HMPMCS providing the planning component. Initial results look promising.

10 Present Status of HMPMCS

An integrated Fault Handling/Planning system is being implemented and will be connected to a large scale breadboard EPS for Space Based Radar. The resulting system will demonstrate autonomous power management for the SBR EPS. The system will include both space-based and ground segment-based components.

The Health Monitoring Power Management Control System (HMPMCS) will form the basis of this new system. The Case Based Planning System will be integrated into the HMPMCS providing the planning elements. The resulting system will draw from the designs described in sections 8 and 9. The two systems will work cooperatively on the diagnosis/planning aspects of spacecraft electrical power system management and control and will share a number of components (EPS model, user interface and reasoning mechanisms). The identification of component failures by the diagnostic system will assist the planning system in the plan selection and evaluation.

As noted above, phase 2 of the HMPMCS project will be demonstrated on a radar spacecraft electrical power system breadboard. The breadboard EPS will demonstrate advanced techniques in spacecraft power system design. The EPS breadboard contract is in the early stages of phase 2. The anticipated SBR spacecraft and EPS

requirements are now being defined. As the diagnosis and planning requirements are not yet available, the designs of the HMPMCS and CBPS are not yet fixed.

Preliminary designs indicate the system will be similar to those described in sections 8 and 9. It is expected that the real-time or near real-time components will be developed in another language such as C, C++, or the proprietary C/C++ based 'Control System Probe' of ISE [ISE 1989], [Zheng 1989]. The AI aspects of diagnosis and planning are expected to be prototyped using an AI development environment (possibly based on Smalltalk). It is anticipated that portions of the AI components may need to be rewritten using a more efficient language following a review of the resulting performance. It is anticipated that the HMPMCS and the CBPS can be tightly integrated sharing considerable code (EPS model, user interface, etc.).

11 Summary

The Electrical Power System and operational requirements of high-inclination low-earth-orbit radar spacecraft were described. Given these requirements, factors affecting the design of an Artificial Intelligence based fault detection/diagnosis and resource management system were described. Past accomplishments and current activities on a 'Health Monitoring Power Management Control System' and a 'Case Based Planning System' were described.

Acknowledgements

The authors would like to thank Craig Maskell of the Canadian Department of National Defence, Space Based Radar Project Office, David Andean of the Department of Communications, Radarsat Technical Program Office, and Selim Ulug of Software Kinetics, CBPS Project support, (all of Ottawa, Canada) and Breen Liblong of the Alberta Research Council, (Calgary, Alberta, Canada) for reviewing and contributing to this paper.

References:

[Adamovits 90] Peter J. Adamovits, "Multiple Fault Diagnosis of Spacecraft Electrical Power Systems",

Sixth CASI Conference on Astronautics, pages 68-76, Ottawa, November 1990.

[Eatock 90] Brian C. Eatock, "Progress in DND's Space-based Radar R&D Project", Sixth CASI Conference on Astronautics, pages 452-463, Ottawa, November 1990.

[Hammond 89] Kristian J. Hammond "Case-Based Planning-Viewing Planning as a Memory Task" Academic Press Inc. San Diego, 1989 (ISBN 0-12-322060-2).

[ISE 89] International Submarine Engineering Limited and The Alberta Research Council, "Expert System Architecture and Component Description" Contractor report for Canadian Space Agency. July 1989.

[Moody 89] M. H. Moody and C. A. Maskell, "Electrical Power Distribution on Space Based

Radar Satellites", IEEE, AES Magazine, pages 10-16, November 1989.

[Ulug 91a] S. Ulug and I. Smith, "Case Based Planning System Software Design: CBPS Version 3.1" Software Kinetics Limited, Canadian Space Agency Contractor Report Number CSA-DSM-CR-91-004, January 1991.

[Ulug 91b] S. Ulug and I. Smith "Case Based Planning System Project Design/Development Philosophy", Software Kinetics Limited, Canadian Space Agency Contractor Report Number CSA-DSM-CR-91-002, January 1991.

[Zheng 1989] Xichi Zheng and Shil Srivastava, "Events and Actions, An Object Oriented Approach to Real-Time Control Systems", IEEE Pacific Rim Conference on Communications, Computers and Signal Processing, pages 269-272, June 1989.

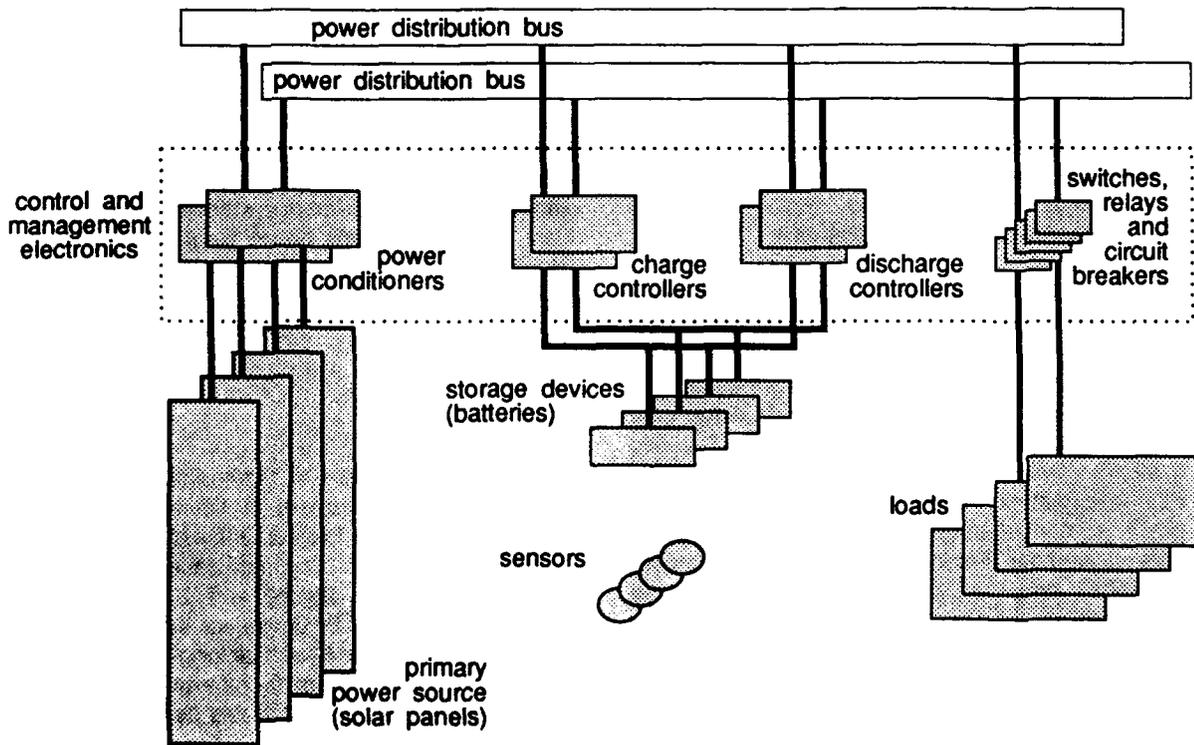


Figure 1 Electrical Power System Block Diagram

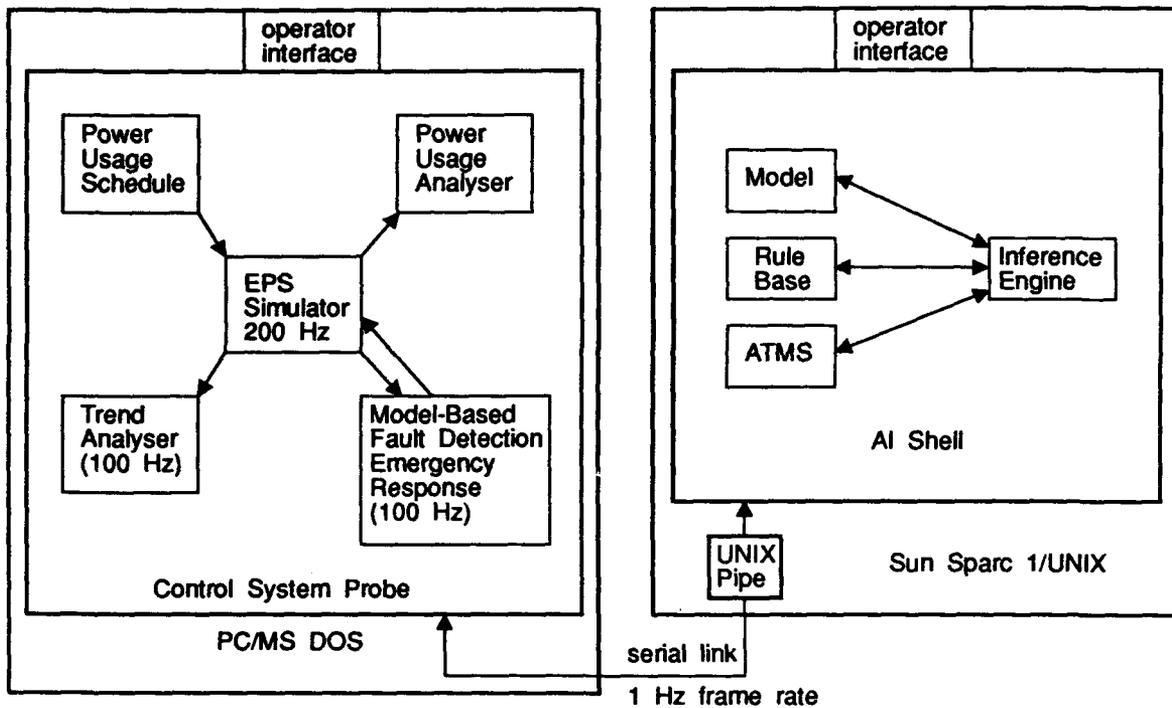


Figure 2 Health Monitoring Power Management Control System (HMPMCS) Phase 1

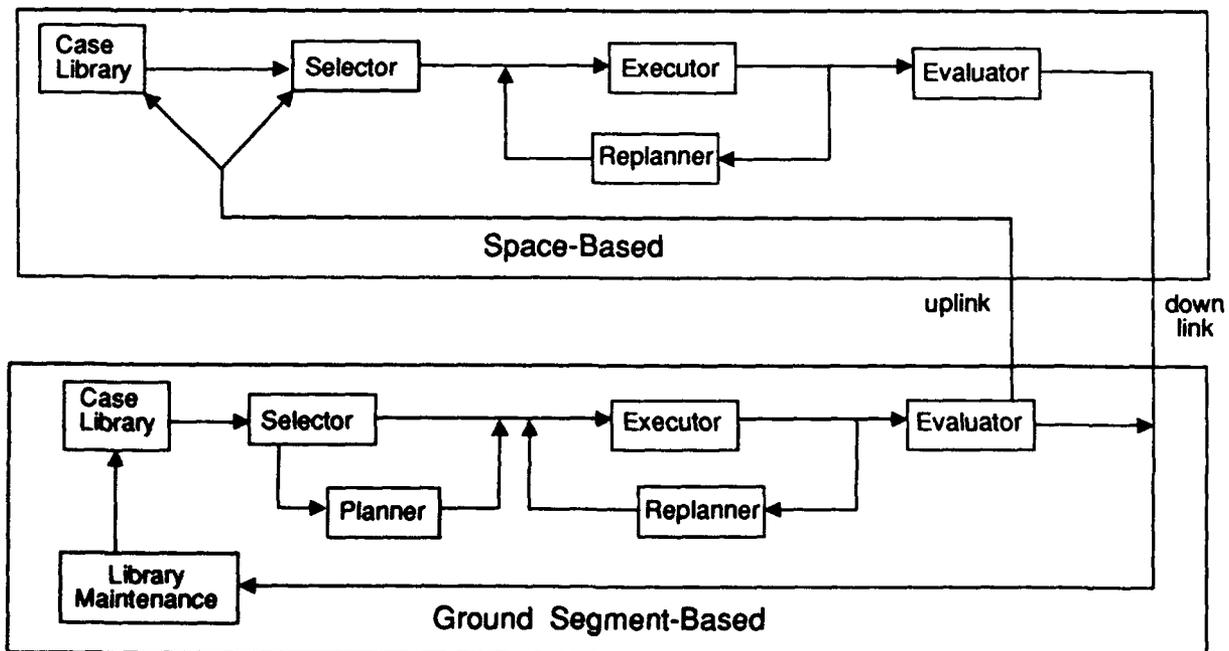


Figure 3 Case Based Planning System



COMPONENT PART NOTICE

THIS PAPER IS A COMPONENT PART OF THE FOLLOWING COMPILATION REPORT:

(TITLE): Conference Proceedings of machine Intelligence
for Aerospace Electronic Systems Held in Lisbon, Portugal on
13-16 May 1991 (L'Intelligence Artificielle dans les Systemes
Electroniques Aero spatiaux)

(SOURCE): Advisory Group for
Aeronautical Research and Development, Neuilly-sur-Seine
(France)

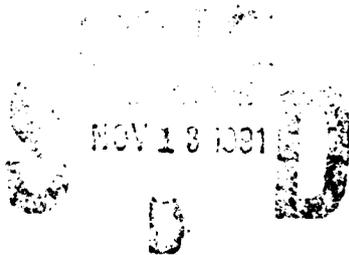
TO ORDER THE COMPLETE COMPILATION REPORT USE AD-A242025

THE COMPONENT PART IS PROVIDED HERE TO ALLOW USERS ACCESS TO INDIVIDUALLY AUTHORED SECTIONS OF PROCEEDINGS, ANNALS, SYMPOSIA, ETC. HOWEVER, THE COMPONENT SHOULD BE CONSIDERED WITHIN THE CONTEXT OF THE OVERALL COMPILATION REPORT AND NOT AS A STAND-ALONE TECHNICAL REPORT.

THE FOLLOWING COMPONENT PART NUMBERS COMPRISE THE COMPILATION REPORT:

AD#: _____ TITLE: _____

AD-P006 332



Accession For	
NTIS CRA&I	<input checked="" type="checkbox"/>
DTIC TAB	<input type="checkbox"/>
Unannounced	<input type="checkbox"/>
Justification	
By _____	
Distribution/	
Availability Codes	
Dist	Avail and/or Special
A-1	21

This document has been approved for public release and sale; its distribution is unlimited.

AD-P006 332

TACAID - A Knowledge Based System for Tactical Decision making

Dr. K. Roberts
Software Technology Dept.
British Aerospace (Military Aircraft) Ltd.
Warton Aerodrome
Preston PR4 1AX
UK

1 SUMMARY

British Aerospace has long been aware of a possible future requirement to embed Artificial Intelligence technology in its conventional airborne systems, in order to achieve the complete mission system of future fighter aircraft. We have followed a development lifecycle of prototyping and evaluation to achieve these aims. The first of these prototypes, which explored the use of AI technology in tactical mission planning in a 1 vs multiple, air-to-air, BVR combat scenario, has already been presented within the AGARD forum. The second prototype, TACAID, which is discussed in this document, uses an identical scenario to explore the coupling of high level heuristic reasoning with conventional modules that produce optimised steering and firing cues, and enemy and own kill probabilities against selected targets. Our aims were to assess the value of embedded ~~AI~~ *artificial intelligence*.

TACAID is developed using the tool *USE*, a blackboard based system which possesses many of the generic features used in AI namely:

- 1) Object oriented knowledge representation;
- 2) Object hierarchy;
- 3) Object relations;
- 4) Forward production rules;
- 5) Backward chaining goals;
- 6) Symbolic list manipulation; *AND*
- 7) Knowledge source scheduling. ←

In addition there is also the facility for linking to "C" object modules. The architecture of the system is described in terms of the knowledge representation and the planning mechanism which couples to the conventional algorithms. Its evaluation as an "intelligent" system is also reviewed along with our intended future directions for AI involvement in mission systems.

2 INTRODUCTION

The next generation of fighter aircraft will present the pilot with a high degree of information to assimilate, and electronic equipment to operate within the cockpit. In a combat situation he will be required to interpret radar signals, passive radar detection data, infra-red data and digital C¹ data from ground based or airborne systems (JTIDS) in order to form a tactical plan of action. This data may often be both conflicting and uncertain which will add to the problem. He will have at his disposal a complex array of weapons and electronic counter-measures, and will also have to perform routine monitoring of the aircraft instruments. The need for future aircraft to exhibit high agility and manoeuvrability mean that the pilot will have to perform these tasks without the aid of a co-pilot. This will place the pilot under an intense workload in an environment where the time between radar and visual contact in a combat situation is a matter of minutes at supersonic speeds. The outcome of such engagements will be often dictated by the element of surprise due to inadequate assimilation of this high data flow rate rather than determined by aircraft or weapon system characteristics.

It would be clearly advantageous for a computer system to perform some of these tasks, reducing the pilot workload without compromising the aircraft agility. Such systems are known as Mission Management Aids (MMA's) and this document

describes one such prototype, TACAID (TACTical AID), developed in the Software Technology Dept. at BAe, Warton. It is a research tool to explore the use of Artificial Intelligence (AI) ideas in MMA's and investigate the potential benefits of coupling between AI and more conventional modules.

3 MACHINE INTELLIGENCE IN THE COCKPIT

The need for increased automation on future fighter aircraft has led to a program of work within BAe to define the complete mission system. This work is led by Systems Engineering R&D group at Warton. There are several roles under consideration:

- i) Sensor Fusion - the integration of the raw sensor data available from the aircraft; radar, RWR, IRST, JTIDS and IFF to give a co-ordinated and clear picture of the outside world.
- ii) Situation Assessment - the view of the outside world derived from the sensor fusion process is organised to prioritise threats, isolate clusters and specific aircraft formations, infer possible enemy intent and allocate targets amongst friendly aircraft.
- iii) Tactical Planning - the pilot must decide on an appropriate tactical plan of action once an understanding of the situation has been realised. This plan must be flexible enough to account for the dynamic nature of the outside world due to new and changing threats, system faults, enemy missile firings at own aircraft and unpredictable enemy behaviour. Computer systems which perform this task, or at least aid in the process, are known as "planners" or TDA's (Tactical Decision Aids).
- iv) Sensor Management - both the Situation Assessment and TDA modules will require feedback to the sensors to dictate their optimal operation in the combat environment. Modern sensors have multi-mode capabilities, the use of which is dependent on the current tactical situation. Control of search volumes, search times, mode of operation, and cueing of sensors to illuminate a target, either concurrently, or sequentially, is a complex task in a modern environment.
- v) Utilities Management - All of the above are dependent on the system status of the aircraft to determine if a particular action or recommendation is feasible. The health status of the engine, hydraulics, sensor, navigation, flight and weapons systems of the aircraft will need monitoring to provide the boundary conditions for tactical planning.
- vi) Pilot Interface - all the relevant data must be presented to the pilot in a compact, unambiguous manner. The quantity of data and the limited availability of space in the modern cockpit make the "Man Machine Interface" (MMI) an equally important part of the mission system.

The research initiative at BAe is exploring all aspects of these problems. A conceptual architecture for the interaction of these roles is shown in figure 1, it should be noted that in reality the boundaries between them are not so clearly defined. As such, it should be stressed that this diagram does not reflect our design philosophy.

BAe has long realised that Artificial Intelligence technology has a strong role to play in these systems, since they are attempting to replace or emulate many of the cognitive processes that the pilot and/or the co-pilot already has to perform in the cockpit.

The actions and reasoning processes involved in situation assessment and planning involve heuristic knowledge of a symbolic nature, often involving data that has an inherent uncertainty. The interface between man and machine must also possess a high degree of intelligence. Beyond responding to the pilots requests, it must monitor the pilots control of the aircraft so it is aware of his situation and objectives so that it can organise and prioritise information effectively, and present relevant data without prompting. It should be stressed that AI will not provide the total solution to these problems and this technology will have to be integrated with conventional computing techniques. In particular the high level reasoning processes in tactical planning and situation assessment will be based on numerical algorithms that generate steering cues, missile escape zones, kill probabilities and flight parameters. Sensor and Utilities management will also have to integrate with systems that are controlled and have been implemented conventionally. It is the hope that combining these techniques will lead to a system with a high degree of synergy.

Our mission systems work programme spans the three BAe(MAL) sites. The Systems Engineering R&D Dept. at Warton is researching situation assessment and tactical planning with numerical processing techniques, Systems Engineering at Brough are responsible for the Utilities management, and Systems Engineering at Kingston are researching sensor management and pilot/machine interfaces. The Software Technology department at BAe, Warton has been responsible for the research of AI in mission systems. It has concentrated its efforts on Tactical Decision Aids through the development and evaluation of prototype systems. The first of these prototypes, KATS, has already been reported in a previous AGARD forum[1]. It is a knowledge based system written in POP-11, designed to provide tactical recommendations in a one vs. multiple air-to-air BVR (Beyond Visual Range) combat scenario. The system is interfaced to a conventional simulator, which provides a model of enemy aircraft capable of performing aggressive tactics on a single aircraft controlled from the knowledge based system.

The second prototype, TACAID, is described in this document. Its major aim was to evaluate the synergistic coupling that could be achieved between AI modules and the conventional modules already under development by the Systems Engineering R&D Dept. at Warton. The system was developed using the tool MUSE[2], a general purpose software environment for AI program development. The remaining part of this document is devoted to a description of the TACAID program and its evaluation. For a more general discussion of our mission systems programme the reader is referred to another paper presented within this forum[3].

4 DEVELOPMENT ENVIRONMENT

Our requirements for a development tool included the ability to test our software in an embedded environment, to be fully integratable with a conventional language, and possessing all the generic features of an AI language to allow rapid prototyping. A survey of currently available tools led us to MUSE[2]. Muse is a general purpose software environment for the development of AI applications in an object oriented environment. It operates a series of knowledge source objects which communicate via a set of shared database objects, within each database knowledge is stored in the form of objects. The knowledge sources perform inferencing on the databases and return their results to an appropriate database. The programmer is completely free to control the number of knowledge sources, the number of databases, their interaction, and their execution scheduling. This type of mechanism is often referred to as a blackboard architecture.

The basis of MUSE is an object oriented procedural language called POPTALK. The language allows for the creation of objects that hold data, and functions that can operate on that data. The object represents some real or conceptual entity that is a useful way to visualise a problem. The structure of an object is held in a schema, or template definition, that can be defined from other schemas in a hierarchy. Objects can be

created dynamically, related to other objects, loaded into any specified database, and manipulated in a procedural framework of lists, mathematical operations and control loops.

The knowledge source objects can manipulate the knowledge in the databases either procedurally via Poptalk code, or declaratively, by use of Forward Production Rules or Backward Chaining Rules. Their execution sequence is determined by an Agenda object, which holds a prioritised list of knowledge sources awaiting execution, further objects can be added to the list at a specific priority. The forward production rules exist as a set of rule-set objects that contain rules of the form :

IF condition_1 AND condition_2..... THEN action

The conditions test on the existence of object instances, with specific attributes, that reside in one of the blackboard databases. The action can consist of either, straight Poptalk code, the modification, creation or deletion of an object instance in one of the databases, or the action can invoke another knowledge source. These rules are implemented declaratively since the controlling mechanism is hidden from the programmer, they should be thought of as a statement of knowledge as opposed to a line of code.

The Backward chaining mechanism allows the programmer to implement an inference network of goal states that are dependent on a set of sub_goals, the sub_goals themselves can be decomposed into their dependency on further goals. At some point the goals are directly related to the existence of a particular object, with specific attributes, in one of the databases. Any knowledge source can call on the backward chaining system to test the logical state of one of these goals by performing a depth first search of the goal tree. The programmer is free from worry about the implementation of this controlling mechanism.

Given the ability of MUSE executable code to be down-loaded onto a target processor, as would be required for our software to run embedded within a conventional system, and a rudimentary C interface, it meets most of our requirements. Despite its wealth of features we have had to perform various enhancements to the tool to fully meet our requirements. We have enhanced the Muse tool to give it full compatibility with the C programming language; object instance pointers can be passed between Muse and C to allow slot updates from C and Poptalk code can be executed from C. We have also given the tool enhanced graphics capability for demonstration purposes.

5 ALGORITHMIC DEVELOPMENT

Algorithmic work on attack planning has been carried out independently from the TACAID programme. Given an air-to-air BVR scenario, 1 vs many, referred to as the α scene, the algorithms perform a reduced and prioritised sub-set known as the β scene, which are deemed to represent the greatest threat to the friendly fighter. This selection is based on numerical data such as range rate, velocity and height, and therefore is better suited to algorithmic computation.

Given the prioritised β scene list, algorithms exist that will generate flight legs against the hostile aircraft which will lead to the destruction of these targets. These flight legs are then analysed numerically using simulated AMRAAM type missile performance to calculate kill probabilities against the targets for missile firings along the flight path. This calculation performs the identical analysis to evaluate enemy firing points against self. Based on both calculations, the optimal firings points are evaluated for own aircraft, with an overall score for that option. In general the analysis results in a set of ranked options which the pilot may opt to fly against the targets. These options are regenerated at each update of the sensor data. More details on these options can be found in[4].

It is recognised that some of these tasks may be better suited to an AI approach and our programme represents a compromise between, integrating conventional computing techniques into the cockpit at an earlier date than embedded AI, and actively pursuing AI research. TACAID's aim was to place a high level heuristic architecture above these algorithms to control if and when options should be generated, to select the options for the

friendly aircraft to fly, and to recommend alternative strategies when it is no longer safe to fly these options.

6 TACAID

TACAID consists of three main components. A blackboard database to hold an internal representation of the world as seen by the aircraft sensors. A tactical knowledge base, which acts as an inventory of types of engagements, and how and when they should be conducted. And a set of reasoning elements that monitor the state of the world and control which plan should be adopted.

The blackboard database consists of aircraft and missile instances, and a special instance for own aircraft that is conceptually executing TACAID in the air. The state parameters of these objects are generated by a conventional simulator, and there is a one to one correspondence between the existence of an object in the TACAID database and those generated by the simulator. These objects hold data that specify the state of the object (eg. velocity, threat data, missile status etc.), and functions that act on this data (eg. missile firing, tactical manoeuvres, state update etc.).

The tactical database utilises the hypothesis that the state space can be quantised into a finite, and manageable, set of states relevant to tactical planning. Given that the world is in one of these states there will be a desired plan, or objective, that will be appropriate for this state. At run time, a complete set of plan templates are defined to cover this state space. A set of pre-conditions associated with each plan dictate which part of the state space applies to this particular plan, and a set of triggers dictate when the aircraft state is no longer valid for the plan.

Within each plan the state space is quantised further to allow a decomposition of the plan, or objective. This is achieved by a set of steps which must be completed in sequence to realise the objective. Steps, themselves, are represented as objects and hold pre-conditions and triggers, identical in structure to those of plan objects, which control the sequencing between steps. Pre-conditions and triggers are implemented in a declarative backward chaining network as a series of goals. Each goal can be thought of as one of the quantised states of the world. The backward chaining mechanism of MUSE will evaluate the logical state of this goal, on request.

There are four main reasoning modules within TACAID. An information manager, which controls the flow of data between the conventional simulator and TACAID, and updates, creates or destroys the objects in the TACAID blackboard database as necessary. Any recommendations for own aircraft, derived from TACAID's reasoning logic, are communicated to the simulation. The information manager also invokes a set of forward production rule-sets that perform some preliminary quantisation of the state space.

There is an Analyser which performs initial grouping of targets into specific formations. This reasoning module can be thought of as performing situation assessment tasks.

There is a Monitor reasoning object which searches the state space for world states that are of interest to the current plan. This corresponds to evaluating the triggers that are associated with the current plan. If any of these triggers evaluate true then the planner reasoning module is invoked.

The planner is invoked to cause either a re-planning action, or to cause the planner to advance to the next stage of the plan ie. execute the next step object associated with the plan. The type of action is dictated by the corresponding trigger which invoked the planner. Again, the backward chaining mechanism is invoked to search the pre-condition state space of either, the plans held in the TACAID database (in the case of a re-planning instruction), or the steps associated with the current plan (in the case of a next step instruction). This process results in a new set of recommendations for own aircraft which are loaded into the information manager.

Figure 2 depicts the interaction of these reasoning modules with the blackboard database. The analyser has been omitted for clarity since it is only executed once during run time initiation. The heavy set arrows denote interaction with the database whilst the smaller arrows represent data flow, or message passing between objects. The blackboard database can be thought of as being partitioned into subsets. The information manager takes data from the simulation and creates an object representation of this data in the blackboard. The forward production rule-sets of the information manager then analyse this data to quantise the state space and store their results on the blackboard. The information manager then executes the monitor. The monitor then searches through the state space to look for events that are of interest to the current plan. If such an event occurs, the monitor executes the planner. The planner searches the tactical database to generate an appropriate response. This will involve either changing or modifying the currently adopted plan, and informing the monitor what is now of interest in the outside world. The cycle repeats itself, after the simulation updates itself.

7 RUN-TIME BEHAVIOUR

TACAID has been implemented on a SUN-3/160 colour graphics workstation. The layout of the display is shown in figure 3. On the top right is a graphics display depicting the current state of a simulated battle. The hexagons represent enemy fighters and the squares represent enemy bombers. To each aircraft is attached a state vector showing the direction of motion, the length of this vector is proportional to the aircraft speed. TACAID is generating instructions for the aircraft at the centre of the range display, own aircraft. The battle is viewed from a reference plane fixed with respect to own aircraft motion. The circular and ellipsoidal regions represent SAM (Surface to Air Missile) site zones. It should be noted that these graphics were developed by the Systems Engineering R&D group at Warton as part of their MMA programme.

On the top left is a mouse activated control panel which allows partial control of the TACAID process. This panel also serves as a display for significant state parameters of own aircraft and the world.

On the lower right is another graphics window depicting the current plan selected for own aircraft by the TACAID process. This is displayed in a hierarchical format of the selected plan object and its associated step objects. Note that each step has an associated set of instructions that will invoke the algorithmic modules.

Finally, on the lower left is the Poptalk interaction window, to which, run-time messages are scrolled. The user can temporarily halt the execution of the process from this window and interrogate the blackboard database, change object slots, or redirect the planning process as he sees fit.

In this particular example, TACAID is performing aggressive manoeuvring on the escorted attack raid. These manoeuvres are generated by the algorithmic modules as described in section 5. The TACAID logic searches through the options database at each simulation update to select an optimal response to the tactical situation. At the same time TACAID monitors the state space to ensure that aggression is the required response. The option selected is displayed on the graphics display along with the optimised firing points against the enemy targets.

In this situation, a pilot would focus his attention on specific states of the outside world that may require a deviation from his current plan or objective. For example, a missile firing by one of the targets at own aircraft, may represent such a state. In TACAID this process is modelled by the trigger objects associated with the plan and the steps. Given such a state, a pilot would then perform a deeper reasoning process based on his knowledge of the existence of a missile. "How dangerous is it?", "How long before I must take action against it?", "Can I still behave aggressively, and how?" etc. In just such a situation TACAID would be triggered to select a new plan of action and the monitor configured to evaluate the threat of the incoming

91-15514



91 1113 009

missile. Aggressive manoeuvring would continue by selecting an appropriate attacking flight path from the options database within the constraints imposed by the missile.

The synergy we have achieved between the coupling of AI and algorithmic modules is clearly seen in these examples. AI provides the heuristic guidance of what response to make in a given situation and the algorithmic modules provide optimisation of this response.

8 EVALUATION AND ASSESSMENT

We have developed a system that has demonstrated the feasibility of automated tactics in the cockpit and the advantages that can be gained from the synergistic coupling of AI and algorithmic modules. This has been achieved with the aid of an enhanced version of the MUSE tool which has proved to be an

excellent prototyping environment for these studies.

Despite our enthusiasm, we do not believe we have in any way developed a system that could outperform a pilot. Although providing a weak model of the pilot's cognitive thought processes it does lack valuable features. At present, TACAID is incapable of dealing with uncertainty or performing predictive behaviour. Nor can it co-ordinate its activity with other friendly aircraft in the combat scene. These problems may well prove to be too complex and demanding for sequential computing techniques and the present TACAID architecture. In anticipation of these problems, the Software Technology Dept. at Warton has entered into an academic and industrial collaboration to develop a methodology for distributed AI architectures[5]. Although the aims of this project are non-military, its results will be of great benefit to our MMA programme.

References

1. Kearney, P.J., "The Representation of Tactical Knowledge" AGARD CP 440, October 1988, paper 31
2. Englemore, R., Morgan, T., "Blackboard Systems" Addison & Wesley, 1988
3. Semple, W.G., "Development of Tactical Decision Aids", AGARD AvP Sym 61, May 1991, paper 17
4. Mitchell, N., "Computer Aided Tactics in the Cockpit" AGARD CP 440, October 1988, paper 25
5. Martin, S., & Slade, A., "Methodology for Distributed AI and its Application for Data Fusion Applications", IEE Colloquium - Principles and Applications of Data Fusion, 1991

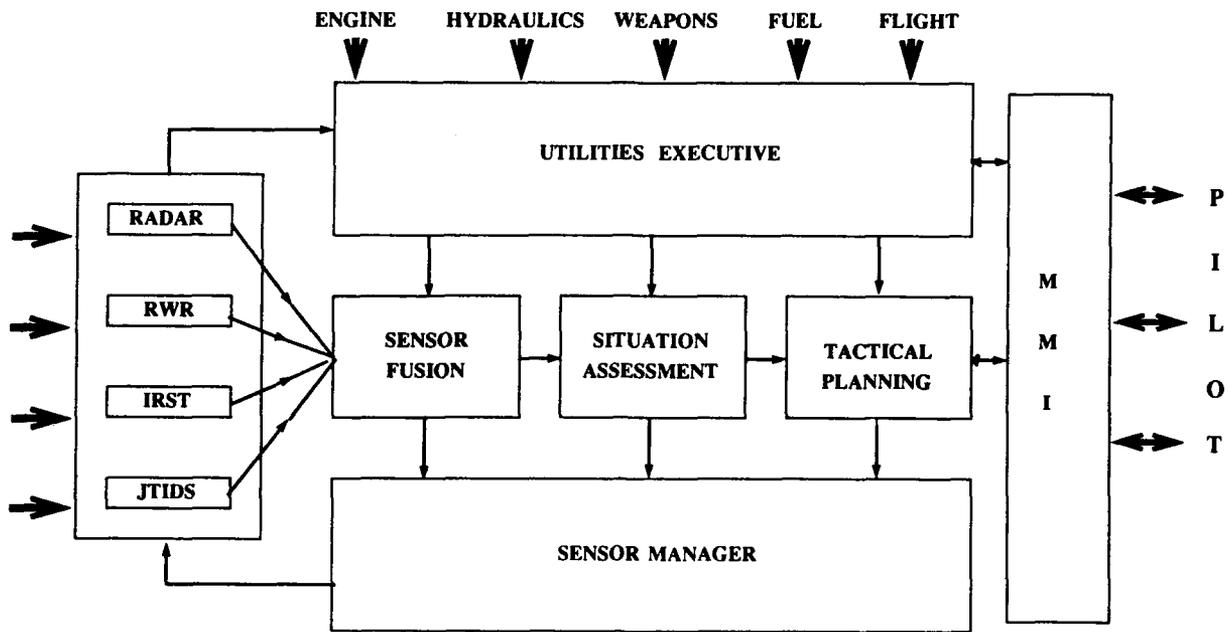


Figure 1. Possible functional architecture for a MMA

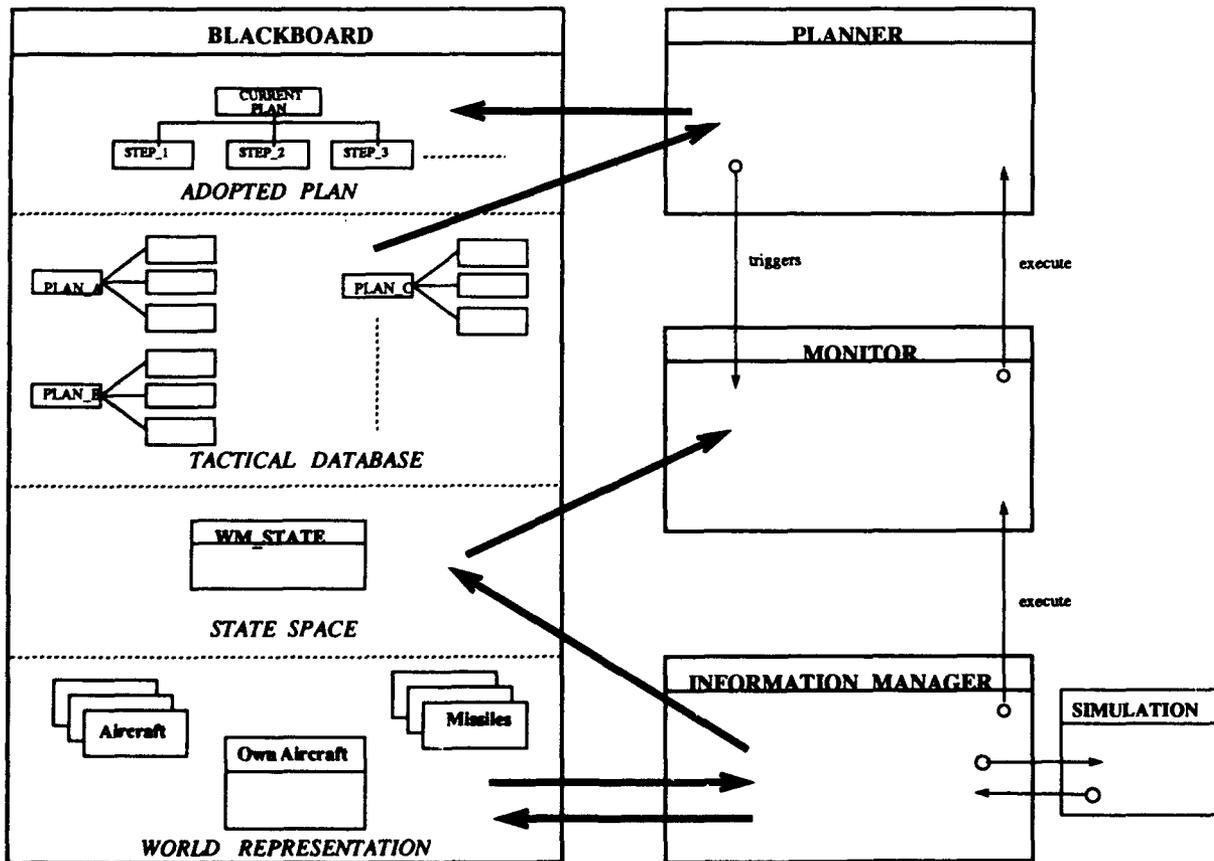


Figure 2. Interaction of TACAID's reasoning modules with the blackboard database

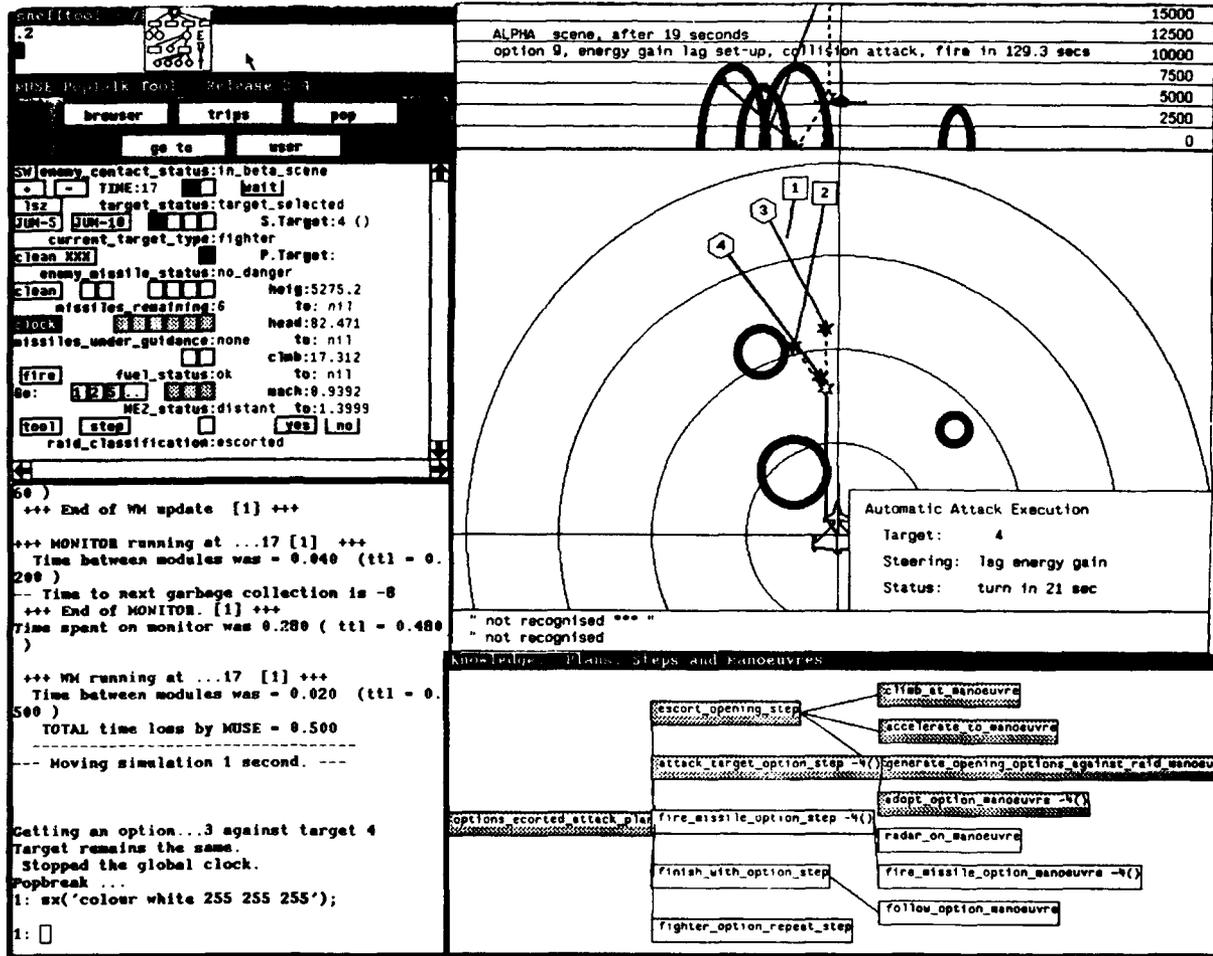
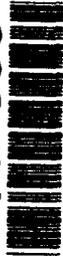


Figure 3. Layout of the TACAID display



AD-P006 333



Automated Threat Response Recommendation in Environments of High Data
Uncertainty Using the Countermeasure Association Technique (CMAT).

Mr. George B. Chapman
Dr. Glenn Johnson
Mr. Robert Burdick
Mission Research Corporation
735 State Street
Santa Barbara, California 93101
U.S.A.
(805)-963-8761

1. Summary

This paper discusses the CounterMeasure Association Technique (CMAT) system developed for the Air Force, which is used to automatically recommend countermeasure and maneuver response to a pilot while he is under missile attack. The overall system is discussed, as well as several key technical components. These components include use of fuzzy sets to specify data uncertainty, use of mimic nets to train the CMAT algorithm to make the same resource optimization tradeoffs as made in a database of library of training scenarios, and use of several data compression techniques to store the countermeasure effectiveness database.

This paper introduces the CounterMeasure Association Technique (CMAT) that provides automated threat response recommendation well suited to handle incomplete and corrupt threat identifications. By monitoring the processed intercepts and detections from the avionics, countermeasure reactions are selected and scheduled. By considering all potential threat systems represented by signal intercepts, and determining whether an anti-aircraft missile is on the way, the CMAT procedure selects a response to maximize survival prospects considering this and possible future threats to ownship, mission constraints, and aircraft status. Countermeasure selection maximizes the estimated frequency of survival without requiring a threat identification. The entire database of likely threat systems is considered in the reaction selection.

2. Introduction

With the growing sophistication of avionics and anti-aircraft systems, the workload and complexity of tasks a pilot is required to perform has increased to the point that aids are needed to satisfy military aircraft survival and kill goals. It is widely recognized that automated assistance is required to achieve the timely awareness that permits the proper survival or offensive reaction to threat systems, particularly in situations where hostile systems have numerical superiority. It is also necessary that these automations accommodate sensor errors and intelligence uncertainty, including the tactic of WArtime Reserve Mode (WARM) threats whose characteristics may differ from those of cataloged peacetime modes.

This paper presents an overview of the CMAT algorithm, the system architecture, as well as several useful artificial intelligence techniques used to manage data uncertainty, to solve and efficiently retrieve required data, and to achieve automated learning of mission resource optimization.

In section 3 we discuss the important issues which influenced our system design. In section 4 we provide a brief description of the four system modules: the sensor post processor, the chalkboard memory manager, the CMAT survivability estimation module, and the resource optimization module. We also discuss the database requirements and show several interfaces that have been developed to enable data entry into the developed system. In section 5 we discuss the natural language interface and underlying fuzzy set data representation used to specify and manage data uncertainty. Several examples are included which show how we perform fuzzy data fusion of datasets stored in chalkboard memory. In section 6 we discuss two methods used to

1. This work was performed under contract F33657-89-C-2175, Wright Patterson Air Force Base (ASD/XRS).
2. "Mimic Nets", Glenn Johnson, Mission Research Corporation, Feb 1991. Submitted for publication.

achieve data compression of the countermeasure effectiveness information databases required by the system. In section 7 we discuss the mimic net technique as applied to the problem of training our system to learn the proper optimization of countermeasure and maneuver response selection according to training libraries.

3. System Design Issues

We identified eight critical issues in the design of a countermeasure recommendation and threat identification pilot-aiding automation. The eight issues are: adaptability, uncertainty handling, data compression, clear user interfaces, data buffering, efficient search of data structures, chalkboard memory architecture, and processing speed.

1. Since tactics and deployments can change rapidly, the system design should rapidly accommodate change in the database. Threat descriptions, effectiveness data, sensor capabilities, threat densities, and mission objectives change on a regular basis. (For example, the intelligence community will obtain new information, threat system capabilities will be upgraded thus nullifying effectiveness tables, defensive and offensive strategies will be modified, and the theater of operations may change.) These changes should not require recoding of the system software. A system with prolonged utility must accommodate an environment of constantly changing information.

2. The system should accommodate data uncertainty. In radar dense environments, pulse collisions and processor loading make the extraction of clean threat signatures, (for example, Radar Frequency (RF), Pulse Width (PW), Pulse Repetition Frequency (PRF), and Group Repetition Interval (GRI)) difficult. Incorrect and incomplete data may be the only information initially available for threat reaction. Future systems may also automatically allocate and direct the aircraft sensor suite to improve the data detected in the initial intercepts. The threat reaction system must be capable of effectively dealing with this uncertain and incomplete data, and of incorporating improved data as it subsequently becomes available.

3. The system should use data compression techniques. The importance of this issue becomes evident when one considers the

quantities of data that are needed by the system. For example, threat response effectiveness depends on the threat, countermeasure, maneuver, time-to-go, launch range, azimuth, elevation, aircraft velocity, and aircraft signature. Using even a modest sampling of the variables listed above results in greater than ten billion data samples needing more than 80 Gigabytes of storage (at 8 bit precision). When compounded by the other memory requirements (inner and outer launch envelopes, threat signatures, uncertainty profiles, danger maps, mission objective tables), it is evident the size of the database would be enormous without use of data compression.

4. The system should have a user friendly database interface. A graphical data display presents the database in a simple format, which can be easily edited by users without the need for specialized training. For example, to present effectiveness data, uncertainty profiles, launch envelopes, or danger maps, editable contours are used rather than editable table displays.

5. The system requires a buffer manager to manage received datasets. This is necessary to permit the threat response system and the sensor systems to operate at different data rates. The buffer performs dataset memory management tasks which include making data fusion decisions as datasets arrive.

6. All memory management functions used by the buffer manager and by the internal system need to use efficient search strategies for processing datasets in memory. This is an important issue due to the potentially thousands of datasets that may be simultaneously active in memory. It is a problem compounded by the high number of features included in a dataset, which generally also include bounds for measurement uncertainty.

7. A chalkboard memory should be used to store the data sets and other information required by the system. By using a chalkboard, the system can economize on data storage and processing. Different modules can share interim calculations and results as they collectively process their automation tasks.

8. The system must be capable of running in real time in practical architectures.

Parallel architectures permit system growth without increases in processing time.

4. The CMAT System Design

A data driven approach is used to make the CMAT system adaptable to different situations. Data compression and fusion techniques are used to manage the large amount of expected measurement data. The CMAT system uses quantitative discrimination to associate measured data with stored data, but represents this data in terms of fuzzy sets to manage data uncertainty. Finally, the system uses an automated learning technique, the mimic net, to provide recommendation of optimal threat response.

The CMAT system concentrates on achieving three functions:

- a. quality assessment, fusion, and processing of the sensor data available on tactical aircraft.
- b. recognition and priority assignment to threat situations requiring response.
- c. recommendation of viable defensive reaction given the threat situation.

The major functional components of the CMAT procedure are explained in the subsections below and illustrated in Figure 1.

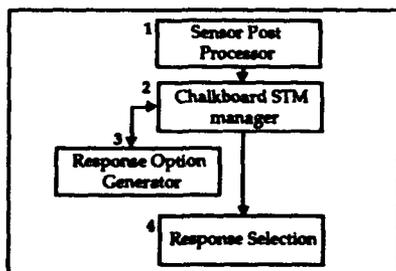


Figure 1. System Block Diagram.

4.1 Sensor Post Processor

The sensor post processor stores, fuses, and sorts measurement datasets collected by onboard sensor systems. The datasets are grouped into two types: radar signature data or missile kinematic data. Each dataset consists of measured feature values and their associated measurement variances.

Typically, there will be few, if any, missile datasets, corresponding to few

missiles in flight in the battle area. In contrast, there could be thousands of radar signature datasets, corresponding to all active fire control radars as well as spoof emitters, acquisition and tracking radars, and search radars. The radar signature datasets enable threat identification and permit appropriate threat response strategy. However, the radar signature datasets must also be associated with the missile datasets. This is especially difficult when angular resolution is low and there are great numbers of distinguishable radar datasets. The system uses two data buffers, one for each dataset type.

The two buffers are structured as 2D binary trees indexed on azimuth angle of arrival which include upper and lower measurement uncertainty bounds. The benefit of the data organization as a binary tree storage lies in the use of binary search strategies to efficiently locate and sort datasets into 'bins'. Efficient data structures are a consideration since thousands of datasets can be simultaneously active.

As new datasets are added to the buffers, a Mahalanobis distance measure is calculated between each new dataset and the datasets already contained in the corresponding buffer. The distance measure is developed from estimated measurement variances. The data is fused to minimize the expected variance of the combination. The variance estimates are derived using sensor resolution and estimated signal-to-interference ratios in information theoretic expressions for the particular discrimination procedure. These variances are corrected for each estimated sensor loading corruption error, eq (1). Datasets judged to be sim-

$$\sigma = \frac{(\Delta m)(Load)}{c_k \sqrt{SIR}} \quad (1)$$

where

Δm = Sensor Resolution

Load = function of sensor loading in pulses per second

c_k = constant term on order 1-10

SIR = Signal-to-Interference-Ratio

ilar are fused into a single dataset. This not only improves the quality of the measurements, but helps reduce the number of datasets to be stored. If a dataset is not sufficiently similar to an existing dataset, a new one is created in the appropriate buffer.

91-15515


91 1113 010

4.2 Chalkboard Memory

The purpose of the chalkboard memory module is to periodically download data from the sensor postprocessor and convert it to a format recognized by the rest of the system. The significant feature of the chalkboard memory manager is that it centralizes data storage and allows all system modules to retrieve and update data. This architecture not only provides for efficient storage of all measurement data in one place, but is also flexible to permit future modules to be integrated with minimal interface difficulty. The three primary functions of the chalkboard memory module are: data formatting, data fusion, and memory management.

Managing data uncertainty is a required attribute of the system. To achieve this, the data from the missile and radar signature dataset buffers is formatted to be compatible with fuzzy set based uncertainty management algorithms. The datasets are converted to a fuzzy set representation, as shown in Figure 2. The measured feature value converts to the center point while the feature measurement variance converts to a fuzzy set width.

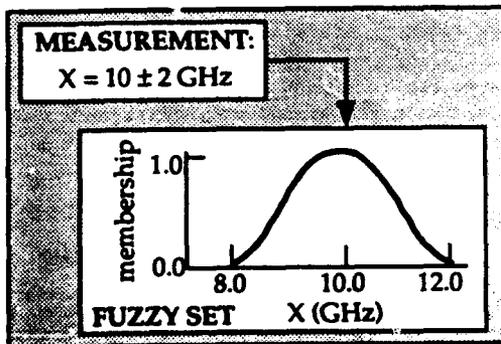


Figure 2. Converting a Measurement to a Fuzzy Set.

Two Chalkboard Short Term Memory banks (CSTM), one for missile data and one for signature data, are used. After downloading and formatting the datasets from the sensor post processor data buffers, the chalkboard memory module integrates the newly formatted datasets with those already in the CSTM. This is accomplished by comparing each newly formatted dataset to each existing dataset in the CST: using a dis-

tance measure. This measure includes a fuzzy set proximity test to compare fuzzy features (2) and a Scalar proximity measure to test Scalar features (3). Datasets that pass the distance measure criteria are declared distinct datasets. The criteria are:

$$FP(S_k, S_i, b_n) > \text{Threshold for all } b_n \quad (2)$$

$$SP(S_k, S_i, U_n) > \text{Threshold for all } U_n \quad (3)$$

where

$$FP(S_k, S_i, b_n) = \max_{b_n} \min(0(S_k, b_n), 0(S_i, b_n))$$

$$SP(S_k, S_i, U_n) = \frac{|U(S_k, U_n) - U(S_i, U_n)|}{\sigma^2(S_k, U_n) + \sigma^2(S_i, U_n)}$$

S_k = kth signature dataset

S_i = ith signature dataset from CSTM

b_n = nth fuzzy feature

U_n = nth scalar feature

V = measured value for scalar feature

σ = measured variance for scalar feature

FP = fuzzy proximity measure

SP = scalar proximity measure

$0(S_k, b_n)$ = fuzzy membership for nth feature of kth observed signature dataset

Scalar data are fused to minimize measurement variance. Fuzzy sets are fused using the fuzzy fusion algorithm discussed in section 5. After each newly formatted dataset has been processed the buffers are cleared to make way for new sensor data.

As newly formatted datasets are fused with datasets already in the CSTM, a reinforcement coefficient associated with the fused dataset is increased. Conversely, the coefficients associated with datasets that did not get fused are decreased. Thus greater emphasis is given to datasets that are seen repeatedly in successive update intervals. If a particular dataset's coefficient drops below a threshold, the dataset is deleted from memory, reflecting that the particular measurement is no longer confirmed by the sensor systems. The CSTM have exponential reinforcement and forgetting laws. The CSTM also stores data from the response option generator and threat assessment and prioritization modules about threat classification for use by the rest of the system.

4.3 Strategy Generator

The strategy generator procedure generates a ranked list of appropriate response options using the data in short-term memory.

In the CMAT procedure, three steps determine the effectiveness of the threat response options, Figure 3. In the first step, the signature datasets corresponding to each detected threat are processed through a layer of threat neurons. Each threat neuron calculates a measure of similarity between the observed threat and a threat cataloged in the threat database. In the second step, these similarity measurements are modified based on the threat density probabilities specified in the threat map. In the final step the similarity data is processed through a layer of countermeasure neurons. Each countermeasure (CM) neuron evaluates the effectiveness of a particular response option based on similarity information, engagement geometry, and the countermeasure effectiveness data stored in another database.

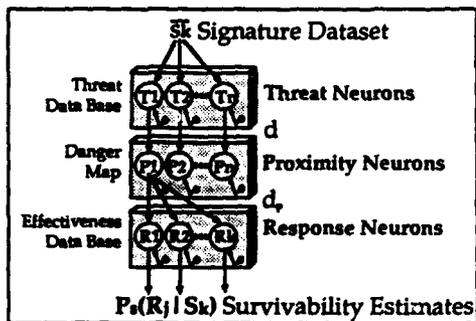


Figure 3. Response Evaluation

The similarity calculation is shown in Figure 4. The diagram illustrates how the system processes uncertainty information.

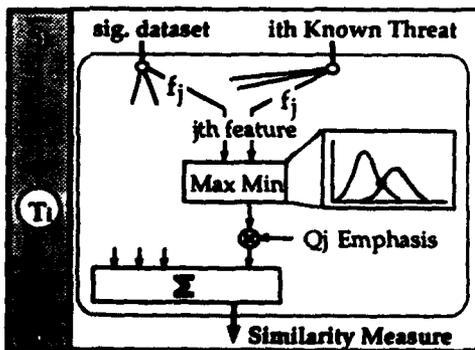


Figure 4. Threat Neurons

A fuzzy pair is formed for each feature consisting of: a) the feature membership function of the observed threat, and b) the

feature membership function of the cataloged threat. For every such pair, the fuzzy set intersection is computed. This value is then tuned according to a database confidence factor and weighted by the relative importance of the particular feature. These values are summed together to form one value, the similarity of the observed threat to the particular known threat (4).

$$R(S_k, T_i) = \sum_n Q_n \sum_n Q_n F(\max_{b_n} \min(0(S_k, b_n), K(T_i, b_n))) \quad (4)$$

where

- F(x) = confidence tuning function
- = a + (b + cx)^d
- Q_n = emphasis weight for feature n
- Q_m = emphasis weight for threat mode m
- $\sum_n Q_n = 1, \sum_m Q_m > 1$

The strategy generator procedure makes reasonable countermeasure recommendations in the case of incomplete or missing data. For instance, consider a detected threat which is operating in a mode entirely different from those cataloged in the threat database. This situation could result in a measured feature value that has no overlap with the fuzzy set descriptions of the corresponding feature of the threats in the database, Figure 5. To mitigate this problem, the fuzzy set corresponding to the particular observed feature is iteratively broadened using a feature relaxation procedure until some minimum overlap is achieved. The similarity measure for that feature is then recalculated and a response recommendation is made based on this new calculation. This approach makes a recommendation based on a closest match between observed threat and known threats.

Each feature's fuzzy set is widened until a minimum overlap criteria is reached

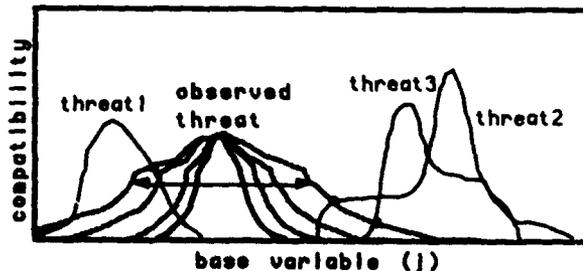


Figure 5. Feature Relaxation.

The threat neuron procedure has several key features:

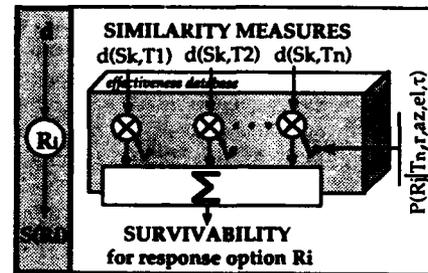
- a) It simultaneously accommodates sensor measurement uncertainty and database threat description uncertainty via the use of fuzzy sets.
- b) It uses a selectable weight to reflect variation in a feature's relative discrimination importance.
- c) It uses a parallel architecture which allows for efficient processing of the fuzzy pair data.
- d) It uses piecewise analytic functions for membership function representation which allows for fast closed form analytic solutions to the fuzzy intersection.
- e) It is data driven.

In the second step of the strategy generation procedure, the similarity measurements from the threat neurons are passed through a layer of proximity neurons. Here, data from the threat map is used. This map is constructed using intelligence information and premission briefing data that specify the probable number and locations of each of the cataloged known threats. Each similarity measure is completed by multiplying it with a proportional function of the threat probability estimate. The resulting similarity measure represents the proximity of the observed dataset to the known threats in the database with adjustments for data uncertainty and expected threat density probabilities.

Figure 6 shows the final step in which the similarity measures are passed through a response neuron layer. The survivability estimate is calculated by considering the assessment of the detected threat being each of the known threats in the database. The countermeasure effectiveness is derived for the engagement geometry and kinematic parameters of the engagement. This algorithm has the benefit that its operation does not depend on whether dissimilar signature datasets were fused together in chalkboard memory. This is because the threat response algorithm always considers the likelihood that the detected threat is each known threat when calculating survivability.

The survivability calculation can take advantage of parallel computing architectures so that processing speed is independent of the number of response options. The proce-

dures uses a data driven approach. Changes in the effectiveness database are made without requiring changes to the response neuron algorithm.



$P(R_j)$ = survivability of j th CM response
 t = time-to-go r = +/- launch range
 az = +/- azimuth T_n = n th Threat
 el = +/- elevation

Figure 6. Response Neurons

4.4 Reaction Selection

The list of generated reaction strategies, ranked by estimated effectiveness, is processed by the Reaction Selection module. This module makes the high level, tactical tradeoffs necessary for final strategy selection. Here, the CMAT system considers complex tradeoffs (mission, tactics, estimated effectiveness, and aircraft status), by directly mimicking off-line databases of expert knowledge. A library of expert knowledge supports this function. The mimic net technology used for this application is discussed in section 7.

4.5 Databases

Integral to the CMAT threat response recommendation system are the databases containing the descriptions of the known threat systems. The first such database contains all features described in terms of fuzzy sets, for example, radar frequency, pulse repetition frequency, and pulse width. A second contains descriptions of the threat inner and outer launch envelopes. The launch envelopes describe the aerodynamic performance of the antiaircraft missile.

4.6 Database Interface

Database interfaces are used to allow the user to quickly review the database entries as well as to readily change or add data. These user-friendly interfaces are necessary considering the amount of data

required by the CMAT system. The data is an integral part of the system, and only incremental changes need to be made on a regular basis. That is, there is no need for routine, voluminous data entry.

Three interfaces are shown in Figure 7. The first of these interfaces is used to enter the fuzzy feature descriptions of the known threat systems. This tool allows an analyst to express fuzzy threat feature values for each known threat system in each known operating mode using simple English sentences. A natural language translator maps these sentences into piecewise continuous analytic functions to mathematically

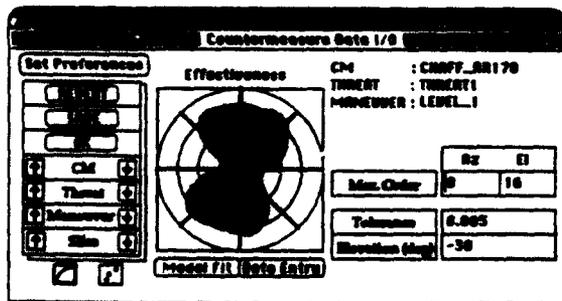
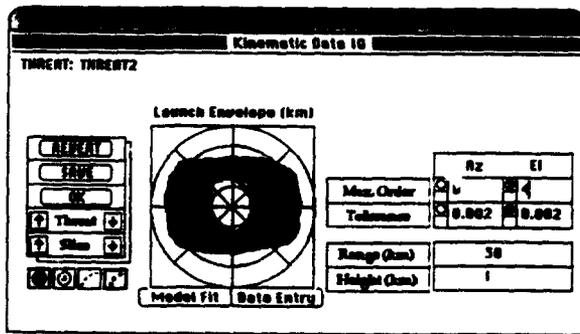
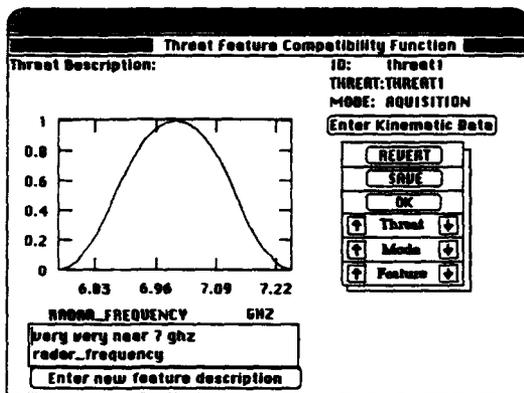


Figure 7. Database Interface.

represent the uncertainty conveyed by the original sentence. The user tunes the graphical descriptions of threat features using the interface to his satisfaction.

The second interface is used to enter data describing threat launch envelopes. Contours representing two dimensional slices of a three dimensional launch envelope contour are drawn on the screen with a mouse. Models of the user input data are saved rather than the user's original data points, resulting in a significant savings in storage requirements.

Similarly, the third interface shown is used to enter the effectiveness profiles for all available countermeasures against all known threat systems.

5. Linguistic Variables and Fuzzy Sets

Fuzzy sets are used to characterize data uncertainty in the CMAT database. For example, threat feature information (PRF, PW, GRI, RF, Infrared (IR) Color Ratios) are characterized using fuzzy sets. The natural language interface used to specify the fuzzy membership functions is based on the use of linguistic variables as described below.

Linguistic variables are used to efficiently specify data uncertainty and provide clear interfaces with a numerical database. They differ from traditional variables in that their values can be represented by sentences in addition to numbers. Figure 8 shows a comparison of a set of numerical and verbal linguistic values. Use of English phrases to express the value of a variable provides interesting opportunities for characterizing data in a way strictly limited by numerical methods. For instance, linguistic variables effectively describe qualitative attributes or behavior. These qualitative descriptions pro-

NUMERICAL	LINGUISTIC
1.0	"Exactly 1.0"
(0.6, 3.2)	"Probably 3.2"
(.99, 3.2)	"Very Likely 3.2"
3.14159	3.14159
45 years	"Middle-aged"
250 lbs	"Heavy weight"
14 GHz RF	"Approximately 14 GHz RF"
1 us Pulse Width	"High Pulse Width"

Figure 8. Examples of Numerical and Linguistic Variables.

vide an effective method for specifying data uncertainty. Graphical feedback rapidly trains the user in the use of linguistic modifiers.

Linguistic variables are useful because they simplify specification of complicated data sets. It is not necessary to understand rigorous implementation details or detailed equations in order to enter data. The linguistic values are checked for proper grammar then automatically translated into piecewise analytic functions characterizing the fuzzy set membership.

Figure 9 shows a representation of the linguistic value 'Near 8 GHz RF.' The value, 'Near 8 GHz RF' is represented graphically by a compatibility function. The compatibility function (sometimes referred to as a membership function) describes the degree of compatibility each value of the base variable, radar frequency, has with the

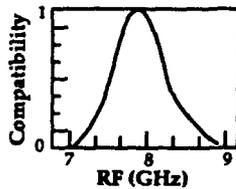


Figure 9. The Fuzzy Compatibility For "Near 8 GHz Radar Frequency".

linguistic value, "Near 8 GHz." As expected, compatibility decreases when the base variable is increased beyond 8 GHz or decreased less than 8 GHz.

Suppose a threat is detected with a RF of 2 GHz. Based on this single threat feature measurement, when comparing the observed threat's RF with a threat whose RF is represented by "Near 8 GHz", it would be con-

cluded that no compatibility exists between the two descriptions. Conversely, if the detected RF was equal to 7.7 GHz, it would be concluded that there is a high compatibility between the two descriptions.

Precise rules exist which define how linguistic values are formulated and how these formulations are interpreted. Therefore, although linguistic variables can be used to describe qualitative attributes, there exists an exact mathematical formulation which echoes the precise meaning assigned to each linguistic value. During program execution the value of each linguistic variable is translated into a fuzzy set description. These fuzzy sets are described by precise, piecewise continuous analytic functions.

All linguistic variables consist of five basic components: primary terms, connectives, negation, units, and feature name. Words like 'high', 'low', and 'moderate' are defined as primary terms. Each primary term is like an additional vocabulary word. The more words there are, the easier it is to describe the concept being presented. Generally, three to five different primary terms have been implemented for each threat feature in the CMAT database. In addition, any real number can be used as a primary term. Table 1 shows the vocabulary of linguistic terms currently available in the CMAT natural language interface.

To simplify the specification of a primary term generic functions are used. Figure 10 shows three standard functions currently available in the database. Use of these generic functions provide several advantages to program operation:

connectives	{ and, or }
negation	{ not }
hedges	{ extremely, very, near, approximately, more or less, greater, less, higher, lower, longer, shorter, larger, smaller, wider, narrower, than, much, increase, decrease }
primary terms	{ high, low, small, large, wide, narrow, long, short, moderate, middle, CW, seeker, long track, short track, ground-based, airborne, close to ground, close to sky, and real number }
unit scales	{ Milli, Centi, Kilo, Mega, Giga }
units	{ frequency, time, size, ratios }

Table 1. Linguistic modifiers currently implemented in the CMAT natural language interface database.

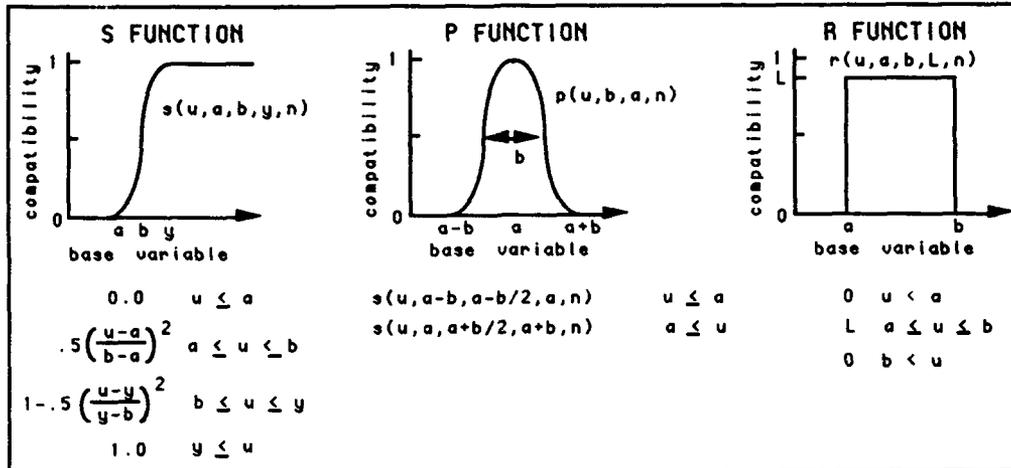


Figure 10. Primary terms used for construction of the piecewise analytic membership functions.

1. Analytic functions (instead of data samples) are used to represent data. Thus, accuracy and quick speeds associated with analytic computation methods are utilized when computing with linguistic values.

2. When primary terms are described by coefficients corresponding to standard analytic functions, linguistic modifiers can be applied by simple modification of these coefficients. As a result, the language interpreter is very fast.

When radar signature datasets are fused in the chalkboard memory it is necessary to fuse two fuzzy sets together. This fusion needs to have the property that as repeated sightings of a similar feature value are made, this value becomes more and more pronounced in the fused fuzzy set. It is also desirable to maintain the fused fuzzy set as a piecewise analytic function. This requirement is imposed to ensure the resulting fuzzy set can be processed rapidly using closed form arithmetic when the calculation of equations (2) and (4) are performed by the CMAT system.

5.1 Fusion of Fuzzy Data

To satisfy these two constraints we use an algorithm which decays the existing fuzzy set in chalkboard memory by a time constant every update period. As a new measurement

is fused with the existing dataset we scale the height of the new measurement (between 0 and 1) to a level incrementally above the height of the existing fuzzy set. To minimize distortion effects, we select the minimum required scale factor to push the new dataset up in order to achieve visible growth of the function in the fused result. Once the desired scale factor has been calculated, the new dataset is fused by use of logical 'or' operator applied to the two datasets. Figure 11 shows an example of the fusion algorithm applied to a set of 10 consecutive fuzzy feature measurements.

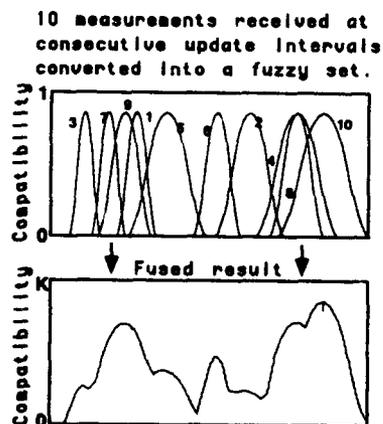


Figure 11. Fuzzy data fusion

6.0 Database Compression Techniques

The CMAT database uses several data compression schemes in order to minimize data storage requirements by the system. Two of these compression algorithms are discussed below.

6.1 Chebyshev Function Fits

One recent technical advancement enabling the compression of large tables of reference data for use in an automated system is Chebyshev functional expansion of multidimensional tables. This methodology simultaneously achieves three goals: large tables or figures of data are compressed into a limited number of expansion coefficients, the fit process is a well conditioned, efficient numerical procedure permitting an interactive graphical interface, and the average value of the function, necessary to assessing the mean value of the data over uncertainty regions, is nearly as readily evaluated as the function itself.

This technique is used in CMAT to encode countermeasure and maneuver effectiveness data, threat launch envelopes, and the aircraft signature for use in the situation awareness and response strategy system.

The Chebyshev polynomial fit is closer in method to discrete Fourier expansions than to Least Squared (LS) error polynomial fits, although the fit is in terms of polynomials. Instead of minimizing the fit error at each of the input data points, this method uses a local polynomial fit to evaluate the interpolated function at pre-selected points, and then passes a polynomial curve through those points.

The advantages of this approach are:

- a) Reduced operation count:
 $NM(\log_2(N) + \log_2(M))$ versus $(NM)^3$
 in two dimensions
- b) Fit is well-behaved numerically versus ill-conditioned matrix inversion.
- c) Approximate min-max rather than LS fit criterion.

The purposeful selection of a fit criterion provides for an efficient evaluation of Chebyshev polynomial expansion coefficients. The Chebyshev polynomial expansion coefficients, c_k , are defined by:

$$f(x) = \sum_{k=1}^N c_k T_{k-1}(x) - \frac{1}{2}c_1$$

The choice of expansion and the discrete orthogonality of the Chebyshev polynomials provide a simple inversion formula for the expansion coefficients:

$$c_j = \frac{2}{N} \sum_{k=1}^N f(x_k) \cos(\pi(j-n)(k - \frac{1}{2})/N)$$

with

$$x_k = \cos(\pi(k - \frac{1}{2})/N)$$

Thus, the evaluation of the coefficients requires an orthogonal, cosine transform of function values at selected points. This evaluation is performed efficiently using Fast Fourier Transform (FFT) methods. Defining the N point FFT of the function data:

$$F_j = \sum_{k=1}^{N/2} (f(x_{2k-1}) + i f(x_{2k})) e^{2\pi i(j-1)(k-1)/N}$$

and phasor arrays:

$$U_j = \frac{1}{N} e^{\pi i(j-1)/2N}, \quad Z_j = \frac{1}{N} e^{3\pi i(j-1)/2N}$$

The coefficients are:

$$c_j = \operatorname{Re}(U_j^*(F_j + F_{N+2-j}^*) - i Z_j^*(F_j - F_{N+2-j}^*))$$

From the Chebyshev polynomial expansion of a function, a Chebyshev polynomial expansion for the integral of the function is readily derived in terms of these same coefficients. Thus, averages of the fit function are evaluated as efficiently as the function itself.

In CMAT, the concern is to maximize the survival chances of ownship, the probabilities of survival in the range 0.5-1 are of more concern than small probabilities. Data fitting errors lead to estimated effectiveness values greater than one or less than zero. Both concerns are mitigated by performing the curve fit to a function of the actual effectiveness. The desired function has a conveniently calculated inverse function, forces the fit fidelity to be best at higher effectiveness values, and limits the effectiveness to less than unity. All these desirable features are

achieved by the choice:

$$f(x) = 1 - (1 - P(x))^a$$

where a is set to between $\frac{1}{2}$ and $\frac{1}{3}$ which is the best compromise value between fidelity and integration convenience. The model curve fit is made to this function, rather than to the actual effectiveness values, $P(x)$. If any calculated value for $f(x)$ is less than zero, it is truncated to zero with no significant loss of fidelity since effectiveness values near zero are of no interest.

6.2 Effectiveness Data Compression

Efficient compression schemes alone are not sufficient to store the countermeasure and maneuver effectiveness data which is a function of four principal variables: launch range, azimuth, elevation, and deployment time with respect to the time to impact of the missile. In addition, there is dependence on the speed of the aircraft, and on the speed of the airborne interceptor in air-to-air engagements. A further data compression scheme is used to store the effectiveness data which is a function of at least four variables.

This dilemma is solved by using a fit which is a function of only two principal variables, launch azimuth and elevation. The dependence on the remaining two kinematic variables, launch range and missile time-to-go, is attained by applying corrections justified by physical arguments.

The simplified CMAT model for countermeasure effectiveness' dependence on all four kinematic variables first breaks down into two independent models, countermeasure effectiveness and location of ownship within the threat launch envelope. The reaction is effective if either the self-protect countermeasure succeeds, or if ownship escapes the kinematic capabilities, the launch envelope, of the missile. In the CMAT model, effectiveness adjustments for range and time-to-go are applied to the curve-fit effectiveness data. No adjustments are made at the optimal countermeasure deployment timing, nor at the center of the launch envelope range. The curve-fit, tabulated effectiveness values are the countermeasure effectiveness for midrange in the threat launch envelope, and with the countermeasure deployed with the most effective timing.

Three individual effects are accounted for by the time-to-go, τ_g , effectiveness adjustment. First, if the countermeasure is deployed 'too late', it cannot affect the missile and therefore cannot be effective. The time scale for 'too late' is determined by the aerodynamic and control response time constant for the missile, τ_a . If time-to-go is less than this time constant, the effectiveness of the countermeasure is diminished. The time-to-go must also be adjusted for the countermeasure device bloom delay, τ_b . This bloom delay is the time that elapses between deployment and the onset of countermeasure effectiveness. If $\tau_g < \tau_b$, the effectiveness vanishes.

The second effect is the duration of the countermeasure effectiveness. If a countermeasure is effective for only a limited period of time, τ_d , the effectiveness of the countermeasure is reduced when it is deployed too soon. This period that the countermeasure is deceptive can be infinitely long, but typically is limited either by the properties of the device (e.g. it burns out, or decelerates) or the characteristics of the threat (for example, a transition period is expended before the transition to home-on-jam operation, or the discrimination of countermeasure from target). Thus, the time scale that determines 'too early' for the countermeasure deployment is fixed by this deception time, τ_e .

Another effect related to the duration of countermeasure effectiveness is the period, τ_{FOV} , that ownship dwells within the missile seeker Field-Of-View (FOV) after the seeker has been deceived by a false target. As long as the target dwells within the seeker FOV, the seeker can recover ownship, particularly if the seeker can determine that it has been deceived by a countermeasure or the countermeasure loses its effectiveness. For example flares burn out, and chaff decelerates to near the ambient clutter Doppler. The time period that ownship dwells within a deceived seeker's FOV can be estimated from the engagement parameters.

7. Mimic Nets

The CMAT system uses a mimic net to automatically capture knowledge from training sessions using actual electronic combat

scenarios and considered, expert selections of reaction strategy. The trained net reproduces and generalizes from the training in ways that are tailored by secondary considerations. 'Maximally intuitive', 'single cost emphasis', and 'maximal extrapolation' are three presently employed generalization techniques. For example, maximal extrapolation requires maximizing the normalized separations of correct options from rejected options.

The ability to capture and reasonably generalize knowledge is a requirement to automate complex tasks. Even if an automated system is meant only to aid a human in the accomplishment of complex tasks, proficiency in the application of accumulated knowledge is required. Since for complex tasks, human experts can not always fully articulate the 'rules' that lead to their decisions, automated systems must also be capable of capturing a model for the decision process strictly from observations of the situations and selected, expert responses.

A mimic net captures a model of a decision process by examining experts' selections. The mimic net is an algorithmic system that duplicates and generalizes from the observed situations and responses.

The mimic net is employed to automatically capture knowledge from libraries of training sessions that contain the decisions of experts and the options from which each expert decision was made. The trained net always reproduces the training data decisions when faced with sets of options contained in the training sessions. That is, the mimic net has a vanishing mean squared error (MSE) for the training data. The training data is exactly replicated by the net. Rather than employing a minimum MSE fit for the net weights to the training data, the net is trained to perfectly reproduce the training data. This is the 'mimic' quality.

A mimic net exists for every internally consistent training database. As long as the teacher or expert makes consistent choices, the mimic net will reproduce his selections and infer his rationale. More comprehensive libraries of training data produce more accurate replications of the expert's rationale. Once the rationale is accurately constructed, adding additional consistent training datasets to the library is redundant.

Although mimic nets can both classify input features and rank sets of input features, it is used to rank reaction options according to stored, expert ratings of the desirability of reaction options generated in the CMAT system. The mimic net applies the higher level tradeoffs for ultimate selection of the defensive reaction strategy. Excess fuel requirements, future value of expendable resources, increased exposure to other potential threats, are figured in together with reaction option survival estimates to select electronic countermeasure reactions.

The library of data from which to construct the mimic net is generated by presenting Electronic Warfare (EW) experts with combat scenarios. The EW experts, pilots, intelligence analysts, and tacticians, select their best estimated response to each threatening scenario. The mimic net captures these training scenarios into a set of standard linear programming objective functions, and uses these objective functions to generalize to new scenarios. The training procedure includes generation of a set of features, quantities that specify each scenario. These features include the survival frequency estimation from CMAT, fuel costs, time remaining in the mission, mission phase, and measures of increased exposure due to each reaction.



COMPONENT PART NOTICE

THIS PAPER IS A COMPONENT PART OF THE FOLLOWING COMPILATION REPORT:

(TITLE): Conference Proceedings of Machine Intelligence
for Aerospace Electronic Systems Held in Lisbon, Portugal on
13-16 May 1991 (L'Intelligence Artificielle dans les Systemes
Electroniques Aero spatiaux)

(SOURCE): Advisory Group for
Aeronautical Research and Development, Neuilly-sur-Seine,
(France)

TO ORDER THE COMPLETE COMPILATION REPORT USE AD-N242025

THE COMPONENT PART IS PROVIDED HERE TO ALLOW USERS ACCESS TO INDIVIDUALLY AUTHORED SECTIONS OF PROCEEDINGS, ANNALS, SYMPOSIA, ETC. HOWEVER, THE COMPONENT SHOULD BE CONSIDERED WITHIN THE CONTEXT OF THE OVERALL COMPILATION REPORT AND NOT AS A STAND-ALONE TECHNICAL REPORT.

THE FOLLOWING COMPONENT PART NUMBERS COMPRISE THE COMPILATION REPORT:

AD#: _____ TITLE: _____

AD-P006 331
AD-P006 334
AD-P006 336
AD-P006 344
AD-P006 345
AD-P006 346
AD-P006 352



For public release and sale to customers to whom it is

Acquisition	
NTIS	Oral
DTIC	T/B
Unannounced Justification	
By	
Distribution	
Availability Codes	
Dist	Avail and/or Special
A-1	

FUTURE ESM SYSTEMS AND THE POTENTIAL FOR NEURAL PROCESSING

By

Arthur G. Self, BSc., PhD.
Gregory Bourassa, Ba, BASc(EE)

Lockheed Canada Inc.
Iber Road Facility
Stittsville, Ontario, Canada
K2S 1E6

AD-P006 334

1.0 INTRODUCTION

The projected radar electromagnetic environments (eme) for the 1990's and beyond include: higher pulse densities (several million pps); frequencies to 40 GHz and higher; stable, jittered, staggered and pseudo-random pulse repetition intervals (PRIs) with multiple frequencies; spread spectrum techniques; multiple agile radar beams and multi-mode missile seekers. Electronic Support Measures (ESM) concerns the passive detection and identification of radar signals. Thus, an ESM system which can measure such signal characteristics will most likely flood its main processor with information to such an extent that it may not be able to cope. In addition, missing pulses and receiver/processor shadowing times may lead to degraded input data to the processor. A number of likely solutions exist ranging from special purpose hardware to new processing techniques. For the former, one could argue the speed of digital hardware technology developments is such that the newest, fastest SBC (Single Board Computer) will handle all, especially with RISC/ASIC assistance; this may be true and it is acknowledged as such. However, in this paper, a radically different processing approach is reviewed, namely that of neural networks.

In the past few years, neural networks have shifted from being primarily a research technology to active use in wide-ranging defence applications. A recent AFCEA publication (ref. 1) on DARPA-funded activities is the most up-to-date publication on such work.

This paper will indicate the likely applicability of a neural processing approach to a range of ESM functions together with results from some preliminary proof-of-concept investigations.

2.0 ESM SYSTEMS AND CONVENTIONAL PROCESSING ARCHITECTURES

2.1 ESM FUNCTIONALITY

An ESM system usually comprises a number of distinct subsystems, as shown schematically in Figure 1; these

subsystems can be realized in both hardware and software as follows:

2.1.1 Feature Extractor

This is the RF front-end of the ESM system which captures the incident RF signal energy and makes measurements such as: radio frequency, time of arrival (TOA) of each pulse, pulse width, amplitude, bearing, modulation-on-pulse (MOP). Such measurements are then assembled into a digital word (called a pulse descriptor word or pdw). Continuous wave (CW) signals are also measured and flagged within the pdw.

2.1.2 Tracker

This raw sensor output will then be pre-processed by the tracker subsystem. The tracker is typically special-purpose digital hardware for monitoring the incoming pdw data stream. It will review incoming data by comparing such measured parameters as RF, bearing and pulse width with previous data already loaded into a window addressable memory (WAM). Incoming data which matches a WAM entry(ies) will be deemed as being from an already detected emitter(s) and this will be monitored, subsequently, for any parameter changes in order to readjust the WAM values. Also, emitter update reports will be sent to the man machine interface (MMI). New incoming emitter data will not match any current WAM entries and thus will fall through to the next subsystem.

2.1.3 Deinterleaver

Previously unseen emitter data will arrive at the deinterleaver, whose function is to extract the individual emitter pulse trains; it does this by recognizing certain parameters of the pulse train such as pulse repetition interval (PRI) and then extracting all those pulses present in the input data deemed to have come from the same emitter. Additionally, characteristics such as frequency agility, PRI stagger/jitter will be derived and an emitter report compiled for onward transmission to the Identifier subsystem.

Initially, upon equipment switch-on, all the data will pass to the deinterleaving subsystem; however, the ESM

system will eventually reach a steady state where, perhaps, only 10% of the data will be new. This preprocessing concept (2.1.2 and 2.1.3) is a particularly powerful technique in ESM systems since it means that the deinterleaver is focused on new data only rather than reprocessing data it has already handled.

2.1.4 Identifier

It is the function of the identifier to indicate the emitter type using the emitter reports provided by the deinterleaver. Traditionally, this is derived by comparing the measured characteristics in the emitter report with a set of a priori data contained in a library. Also, the identifier may be able to correlate an emitter identification with a platform type (e.g., class of ship, type of aircraft, submarine). This correlation is more difficult to establish since one emitter type may be mounted on several different platforms, introducing ambiguity. The identifier information will also be passed to the operator via the MMI.

2.1.5 Tactical Situation Analyst (TSA)

From ref. (6), this will attempt to resolve emitter ambiguities (from the library searching techniques above) through various spatial and temporal correlation techniques. Those emitters that cannot be identified with high confidence will be passed to a hypothesis generator; this, in turn, will postulate one or more hypotheses on the identity of the emitter, and then assign resources to prove or disprove these hypotheses. As shown in Figure 1, the TSA will have direct links to all the subsystems since it may test its hypotheses at any or all stages of the processing chain. So, for example, if the TSA is attempting a platform identification for a particular emitter then it may produce a list of possible candidates based on such data as:

- emitter reports received to date;
- geographical area of the ESM platform; and
- state of conflict/peace, tension, hostilities.

This list may contain several candidates and the TSA wants to home in on the actual one. To refine this further, the TSA may then task the deinterleaver to see if it can detect certain specific emitters (within the new data); detection of these specific emitters would remove certain ambiguities and thus allow the TSA to pass on an unambiguous platform identification to the MMI.

Similarly, the TSA will likely use more of the library data than is used by the Identifier. The TSA would most likely use a library structured in such a way as both to show inter-relationships amongst the characteristics as well as to facilitate optimum search strategies.

2.1.6 Current Techniques for ESM Processing

Basically, these can be subdivided into the following categories, namely:

a) Dedicated preprocessing

This is essentially a sorting process, which selectively captures and (possibly) stores data based on programmable parameter windows. These parameter windows are programmed with *parameter ranges corresponding to emitters which have been previously detected* (and thus have been reported to the MMI). This preprocessing technique is a critical one in that the deinterleaving/identification functions are not overloaded with continuously sorting previously seen emitters and thus their availability is maximized for handling the new/unknown data. As mentioned above, the preprocessor is typically implemented using digital hardware (WAMS, e.g.) in order to keep up with the high input data rate. Additionally, there would reside a specific controller which autonomously adjusts the filter parameter window so as to keep centred on the data.

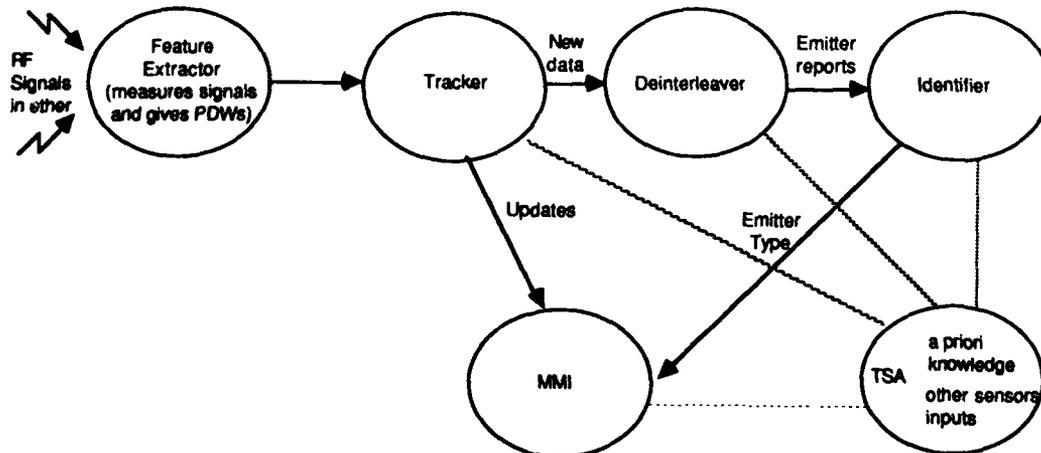


Figure 1.

b) Traditional signal processing algorithms

For new data, a range of signal processing algorithms can be invoked; these would include:

- clustering - so as to conduct a preliminary grouping of the data, typically in (RF, bearing, pulse width) space.
- deinterleaving - the extraction of repetitive PRI patterns (jitter, stagger, e.g.) using TOA information (mainly).
- flagging - from a determined PRI, to identify those pulses making up the actual pulse trains.
- scan determination - derive the scan type, and characteristics, from an amplitude-time history.
- identification - assignment of an emitter type to a deinterleaved emitter report. Currently, ESM libraries are just collections of customized records accessed by search, retrieval and update procedures. A successful match on search is generally considered to establish a detection.

Typically, these algorithms are executed in a serial fashion on any buffer of new input data (although multiple instantiations could be executed in parallel).

c) AI techniques

The use of AI/IKBS/Expert System (ES) techniques is now being actively developed in the

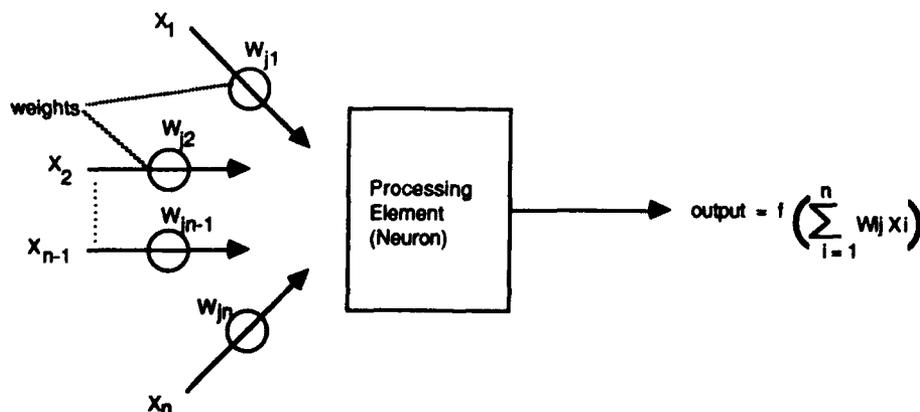
EW community and thus brief mention is included here. Areas of specific interest to ESM include:

- library structure and searching techniques. From ref. (2) Canadian work is ongoing into both an object oriented emitter library as well as an emitter classification system based on symbolic reasoning.
- hypothesis testing. As indicated in ref. (2), such testing can be invoked for identification purposes in order to determine which emitter type (out of a range of candidates) best describes the true source of the observed signal. Additionally, there may be scope for such techniques within the deinterleaving function itself.
- auto operator assistant and situation assessment. Aids the operator in determining the disposition, composition, and intent of the platforms thought to be operating within the ship's area of interest. Since both emitter and platform identifications are often ambiguous, there will usually be a number of tactical situations which could explain a given set of observations. An inferencing system based on beliefs could determine the most likely interpretations of the observed signal data.

3.0 NEURAL NETWORKS APPROACH

3.1 GENERAL

A neural network is a massively parallel, information processing architecture composed of many simple processing elements interconnected to achieve certain collective computational capabilities. Neural networks are constructed of many processing elements (or neurons), as shown below.



91-15516



01 1113 0111

Each processing element may have many input signals but is limited to only one output signal (which may go several ways). Processing elements can be feed-forward only, or have feedback loops as well. Also, processing elements can be fully connected to all the other processing elements or linked only to a few (sparsely connected). Any given neural network design will dictate the nature and number of these feedback loops and connectors. Clearly, connectivity relates to the network's parallelism whilst the design of its feedback loops has implications for the networks adaptivity/trainability.

If the sum of the inputs to a given processing element exceeds a predetermined threshold, the processing element "fires" and sends a signal to the other connected processing elements. As shown above, each input has a weighted value that determines the strength of the interconnection to the fire or non-fire decision of the following processing element. This value can be auto-, or self-, adapting through feedback loops and this is the unique and outstanding feature of a neural network.

3.2 NEURAL PROCESSING - IMPORTANT FEATURES

Conventional approaches to target detection and classification typically include explicitly formulated algorithms, software, and hardware; included will be an analysis and development of specific feature extraction and pattern recognition algorithms which then dictate design and implementation. The challenge, for any such signal processor, is to accept "raw" sensor signals as input and provide target classifications as output. The feature that most distinguishes neural network processing from conventional signal processing is the ability to internally develop, or "learn", the algorithms required for target detection and classification; this includes the extraction of specific target signatures buried in noise. Because of their adaptive nature, neural networks can adapt to changes in the data and learn the characteristics of input signals. Furthermore, because of their non-linear nature, neural networks can perform functional approximation and signal filtering operations which are beyond optimal linear techniques. The architecture and operation of a network is fundamentally different from conventional signal processors; such a processor would require minimal front-end analysis and design, no explicit algorithms, and no software. Additionally, since they are naturally massively parallel, this suggests a decision making ability which is both high speed and fault tolerant.

As indicated in Section 1.0, eme's facing future ESM systems will likely be complex and of high density. Additionally, the massive databases that exist on recorded signals (elint/comint data) probably means that adequate training sets exist with which to train such neural processors.

A list of the attributes of a neural network approach as it applies to our problem is as follows:

- 3.2.1 There exist direct theoretical relationships between other successful learning algorithms (such as Kalman filters) and neural learning algorithms. This adds an essential mathematical credibility to the whole concept of neural networks. It also means that much of existing mathematical theory for filtering and pattern recognition can be translated to the neural network domain. The back-propagation learning algorithm for layered, feed forward networks represents a major advance in effective neural network learning. Back propagation is a learning algorithm which is designed to solve the problem of choosing weight values for hidden units in a layered feed forward network which has been shown to be directly related to the extended Kalman filter.
- 3.2.2 It is necessary to find a compact set of features which can represent an emitter; it is also important that the feature set be sufficiently complete so that emitters can be appropriately discriminated (e.g., friendly/hostile, as well as between defined types). Neural network technology can contribute to the feature selection task in a number of ways, namely:
 - 3.2.2.1 By providing hardware for massively parallel implementations of traditional feature detection algorithms. Whereas special purpose parallel hardware could always, in principle, be constructed for a specific problem, the hardware would, in general, be limited to only that particular problem. One of the things that a neurocomputer offers is the possibility of not only solving one problem in parallel, but a broader variety of problems, through (e.g.) simple downloading of network parameters.
 - 3.2.2.2 Neural networks can implement optimum feature receivers for the extraction of weak features from high clutter environments.
 - 3.2.2.3 Certain networks have the capability of self-organization, so that training data can be clustered without provision of "correct" outputs. Thus, neural network technology can contribute to the feature selection task through the automatic discovery of clustered features by using neural network learning algorithms. Techniques similar to principal components analysis (e.g.) can be used, which have the ability to adapt, so as to represent the characteristics of the input signal.
 - 3.2.2.4 Given their well-documented ability to perform in the presence of noise and incomplete data, neural networks can be made much more robust than conventional algorithms when dealing with missing pulses. Thus, in wartime, the performance of neural networks should degrade gracefully rather than catastrophically.
 - 3.2.2.5 Neural networks have a capability to integrate automatically a diverse set of features. In the ESM case, parameters such as (RF, pulse width,

bearing, MOP, polarization, scan type) are important for target identification. However, there is no obvious metric available which combines such features into an effective classifier. The back-propagation algorithm, combined with an appropriate training set, could be effective here.

- 3.2.3 Although more speculative, neural networks may also provide tools for developing expert systems; they can implement propositions and constraints with the capability to backtrack for explanations. Inferencing techniques can be developed to allow the expert system to reach conclusions when only a fraction of the input values are known. [However, it should be noted that a lot of development is still required to produce a high performance neural network-based ES.]

4.0 ESM PROCESSING REQUIREMENTS

Table 1 shows, for each ESM subsystem (described earlier) the likely input data rates and a translation of functionality into a definition of the likely computer instructions per second required to give this functionality. It can be seen, even with today's environments and requirements, that the ESM processing requirements can carry up to many millions of instructions per second. Table 1 also makes the point that not all ESM functions are operating at the same levels of speed; for example, the tracker (from Fig. 1) has to keep up with the input pdw rate. However, the identifier need only work at the level of a few reports/second. This is a very important point to remember when discussing ESM processing performance requirements.

Trends in future electromagnetic environments can be categorized as follows, namely:

- a) Increasing density, and complexity, of electromagnetic environments. Signal complexity is evident in intrapulse signal modulations (such as chirp and phase coding) as well as increasing agility in both individual signal features (such as RF, pulse width, and PRF) and in combination. Signal densities are rising due to the increasing use of high duty cycle waveforms against an increasing background environment from more conventional sources. Thus new algorithms are continually required to recognize the new emitted signals.
- b) Certain "traditional" processing characteristics such as scan information will soon be unavailable due to emitter electronic scanning developments. Such future thrusts push the ESM system designers towards architectures which capture every pulse (or pulse group) and then extract the maximum amount of information from it. Intrapulse modulation measurements will call for high sampling rates, thus further accelerating the data input problem.

- c) Emitter war modes create a well known problem for conventional library searching strategies.
- d) Own/friendly emissions could also be exotic or high duty cycle, thus creating new problems for the ESM system designer.
- e) An increasing awareness for strict EMCON control until the last minute, puts increasing emphasis on an ESM function which works in real time even on the most complex emissions.
- f) Overlapping signal parameters (e.g., RF, PW bearing) between probably all combinations of hostile and friendly emissions will result in problems for a dedicated preprocessor. For example, tracking of individual emitters will be more difficult and thus its integrity will degrade; this, in turn, will result in fragmented old and new data being passed to the deinterleaver which would result in false reports. The efficiency of PRI predictive gating is perhaps questionable given the numbers, and ranges, of emitters appearing with various amounts of PRI agility. Thus, dedicated preprocessor hardware may be limited in its future performance.

The authors have not attempted to show how Table 1 might change to accommodate the above postulations. Suffice it to say, we see significant problems facing us as we design future ESM systems. We believe that both established and new techniques/technologies need to be investigated for their applicability to this problem. As we will show, a neural processing approach is gathering momentum and it may provide some clues to resolve the above scenario.

5.0 ESM SYSTEM OPPORTUNITIES

It must be stated that it is not the intent of this paper to prove that neural processing is the only technique for future ESM processing needs. Rather, it is to say that neural processing would appear to have some significant attributes which are directly applicable to the problems foreseen in the ESM domain. Neurocomputing is a fundamentally new and different information processing scheme; neural networks are able to derive an information-processing function, or algorithm, from examples of that function's operation.

From Figure 1, the following areas of specific applicability of neural networks are proposed and discussed.

5.1 DEINTERLEAVING

5.1.1 A range of possibilities exist, namely:

- a) The ability to cluster in a hyperspace of high dimensionality so as to include all possible measured, and derived, features of the emitter signal. So, for example, all single pulse data including, perhaps, representations of specific transformations (such as Fourier

I N S T R U C T I O N	Monitoring function probably 50-100 instructions based on $\frac{1}{100}$ sec sampling 150 - 300k ins/sec.	Possibly 100-500 instructions per input signal. For 5 new signals/sec. and mode changes, (500 - 2500) ins/sec.	Possibly 50 instructions/report. 250 - 500 ins/sec.	Probably very high (several million ins/sec.) since mixture of rule based, object base, database operations.
FEATURE EXTRACTOR	TRACKER	DEINTERLEAVER	IDENTIFICATION	TACTICAL SITUATION ANALYST
D A T A	Up to 1-2 million pps ⁻¹ Typically, a few 100 kpps ⁻¹ (300, max.)	10-30 kpps ⁻¹ in steady state	5-10 new emitter reports/sec. 10/sec. updates on existing emitters.	All new emitter reports. Continual review and monitoring of all processing modules (see opposite for consequent data rates).
F U N C T I O N	Measurement of single pulse characteristics, such as RF, TDM, PU, AMP, MOP, CU, Bearing characteristics).	Binning of new data (say 10% steady state) but 100% on start-up. Extraction of PRL.	Library matching of new emitter reports with a priori knowledge.	Hypothesis testing for platform identification. Also, such testing for those emitters not yet identified. Expert assistance to EM operator. EOB prediction.
MONITORING	Parallel comparisons of new pdws versus established tracks (e.g., windowing on certain pdw characteristics).	Monitoring of windows for parameter change.		

Table 1.

coefficients on RF variability within the pulse) could be included. In a trained network, all of these features which are implicit in the input data are handled automatically by the network's connection weights. Currently, systems use only a relatively small set of features are used with a high level of overlapping parameters between emitter types thus leading to ambiguous and/or erroneous results.

- b) Optimum selection of features (from a much wider range of single pulse features) for PRI extraction; for example, pulse shape features may be useful as well as intentional/unintentional modulation information. Currently, PRI is extracted using a TOA difference histogramming techniques; this approach has limited success when the input data stream is incomplete or has exotic characteristics (such as multi-level staggers, e.g.). Neural networks can perform their own feature extraction and, with sufficient data, it should be possible for a network to discover a more appropriate set of features by analyzing the pulse profiles directly.
- c) Improved PRI extraction possibilities, especially for the exotic agility patterns. Thus, a neural network may be able to pattern match rather than time window matching to extract PRI. As in (b), currently the flagging of the constituent pulses in a pulse train is conducted using time-windowing techniques. This is quite complex even for low levels of stagger. Also, non-jittered and jittered signals are difficult to extract in the presence of each other due to an overlap in their definitions (i.e., unjittered = x% spread whilst jittered = y% spread but which encompass x).
- d) The recovery of noise-corrupted or LPI signals. Currently, ESM systems invoke the deinterleaving process once they have declared an intercept; typically, this relates to having received a minimum number of pulses that are clustered. Should the measured input signal have only a few pulses, or be of such poor quality that successful clustering did not take place, then deinterleaving may not be instigated. Neural networks offer the possibility of being able to make something of such data inputs. Neural networks are much more robust than conventional algorithms when dealing with missing pulses. In fact, they can be made to yield a figure of merit for the suggested classification they produce, allowing very graceful degradation in the most difficult cases.

5.1.2 Lockheed Canada Inc. (Lockheed Canada) own research in this area has been concentrating on two major areas, namely:

- a) Deinterleaving using conventional signature parameters such as RF, bearing, pulse width and pulse repetition interval (PRI). Specifically, a range of dense and exotic emitters have been simulated which include a number of "target" emitters to be found by the neural network. The performance of the network is then monitored as the dimensionality of the input data (and thus the complexity of the network) is progressively increased.
- b) Unintentional modulations (as well as pulse shape, etc.) within a pulse are likely to offer significant processing improvements for future ESM systems as digital signal processing technology approaches the speeds required. This specific work relates to the training of a network with real intrapulse data and examining its ability to segregate emitters using such features. This is followed by incorporating such an ability into the network developed in (a) above in order to quantify any performance improvements in the neural deinterleaver.

Using conventional pattern recognition approaches, we are also establishing a baseline performance "figure of merit"; this is essential if we are to understand the relative performance of neural networks in this particular ESM function. For this task, we are using both the leave-one-out-method as well as the Kohonen-Loeve transform.

5.1.3 Preliminary Lockheed Canada results have concerned the extraction of a high level PRI staggered signal in the presence of noise. The processing required to identify the number of distinct intervals in a staggered pulse train, and the length of the sequence of these intervals (since some intervals may appear more than once in each sequence) can be very time-consuming. If the length of the sequence were known in advance, the extraction of the exact intervals from a pulse train and the subsequent characterization of the PRI of the train would be quite fast and efficient. Lockheed Canada has designed a custom neural network to identify the length of the stagger sequence directly, together with the stagger level. It consists of an input layer sparsely connected to a hidden layer, followed by a standard back-prop style connection to the output layer. The network was simulated in software, by writing a small program (14 predicates) in PDC Prolog, which allows various sized nets to be tested in an interactive session. Using real trials recorded data from single emitters, the net was found to identify stagger sequence length correctly 100% of the time, in spite of repeated intervals, and regardless of input set size and order. When sets of data with missing pulses are presented, the net recognizes this fact but does not extract the stagger level.

Follow-on investigation involves training a standard back-propagation net to recognize the detailed deviations in the output of the custom net caused by missing data, and thus extract stagger level in some percentage of these cases. Lockheed Canada now considers the existing net, even without the planned enhancements, to be a useful module for inclusion in an ESM processor. With simple neural chip support, it would allow the PRI calculations for all signals to be handled in the following sequence:

- 1) Submit the pulse train to the neural net, obtaining the length, L , of the staggered interval sequence as an output. With chipset support, and the existing net architecture, this is estimated to require less than one millisecond. Go to 2. If the net signals bad or ill-conditioned data go to 3.
- 2) Read the pulse train, calculating the time deltas, for $L+1$ pulses. Read another L pulses to confirm. Go to 4. If the confirmation fails (possibly try another confirmation and if it too fails then) go to 3. This step is estimated in the worst case to require tens of microseconds.
- 3) Go to bad data handling routines.
- 4) Return the confirmed stagger pattern.

Timings quoted here are orders of magnitude faster than current techniques.

5.2 IDENTIFICATION

Traditionally, emitter reports are compared with a priori data held in a library; typical algorithms operate on an 'if-then-else' type approach to each library entry, where hash coding typically defines the specific library entries involved. Also, such algorithms tend to compare specific parameters in a serial fashion using (usually) quite a small range of parameter values. Ref. (2) details how alternative, more powerful approaches are being examined.

A Neural network approach could be well suited for the identification function in an ESM system since large training sets of a priori knowledge exist. It is here that the robustness of a neural network approach could be well tested against postulations of likely emitter war modes.

Once specific emitters have been identified, the resultant network weights could be noted and then stored in the library. Then, upon switch-on of this future neural-based ESM system, a much wider range of library data would be downloaded into the network for immediate operation.

5.3 SITUATION ANALYSIS

This function does not, in general, exist in current ESM systems. However, much work is ongoing into sensor

(and weapon) fusion systems for all three Services and thus it is only a matter of time before such capabilities appear in an ESM system; Ref. (2) describes one such Canadian initiative in this area. As indicated above, neural networks offer the promise of providing tools for developing expert systems; one of the major limitations of AI/IKBS/expert system work is both the limited real human expertise which is to be captured as well as the actual adequate representation of the rules and human knowledge. Neural technology would appear to offer some advantages here.

5.4 PARAMETER MEASUREMENT

During training, neural networks automatically build up complex, non-linear transfer functions which weight and process the input parameters as necessary to provide the desired output. Analysis of the final network state is a form of knowledge engineering which can yield valuable insights on the relative importance of the various input parameters to the derivation of the final classification. Such insights could be used by hardware engineers to optimize future designs.

5.4.1 From ref. (5), bearing estimation (determining the positions of possibly many sources in the scene from the outputs of an array of sensors) is a problem which has been solved conventionally by matrix methods based on singular value decomposition of the data matrix or an eigen description of the covariance matrix of sensor outputs. An alternative approach (refs. 3 and 4 for example) has been employed in which the bearing estimation problem is mapped on to the Hopfield network with each "neuron" representing the hypothesis that the source is at a particular direction. A mathematical cost function is specified and the evolution equations of the network, which guarantee a convergence to a (local) minimum of the cost function, are solved.

5.4.2 Current research is underway at Lockheed Canada to evaluate the applicability of neural processing to the discrimination of signal source elevations in the presence of multipath effects. Specifically, a network has been trained using signal propagation data containing both specular and diffuse scattering mechanisms. Feeding the network extensive training data for specific sensor/target parameters, trains the network. The goal of this work is to determine the performance of a network in extracting, and then tracking, the presence of specific target characteristics in the presence of the (noisy) propagation environment. Various network architectures are under study to ascertain their relative merits for this application.

As agility in frequency, PRI and other pulse train parameters becomes a more common feature of emitters, the importance of precise and accurate azimuth and elevation measurements becomes greater. With this in mind, Lockheed Canada has undertaken an investigation of the application

of neural processing to high accuracy direction finding (HADDF). Before attempting to design and simulate neural processors for this purpose, it was decided that the preliminary focus would be on elevation measurements for a seaborne scenario. Such measurements are complicated by multipath interference, dependent on profile parameters such as range, sea state as well as emitter characteristics such as RF (see Fig. 2).

A simulation program in Fortran was written to generate data points for emitters approaching an interferometric receiver. Files of such points were generated for emitters at altitudes of from 10 to 100 meters by 10 meter increments using an available signal propagation model based on extensive real data. This range was deemed to be of interest, since obtaining accurate elevation measurements within it presents great difficulty for existing technologies. The network simulations were created using Neural Works Professional II. Several net architectures have been tried, and the best results to date were obtained with a Probabilistic Neural Net (PNN). Success rates as high as 96% on training data and 80% on test data have been achieved. However, these results are regarded as preliminary, since the complete set of experiments originally envisioned has not yet been completed.

A key problem is proper representation of the data. Because the patterns of frequency and amplitude as measured at the receiving array vary repetitively with time (subject to some second-order functions affecting speed of variation, average power, etc.), the network must be sensitive to trends over these variations, or to syntactical representations of events such as multi-path nulls. The planned follow-on investigation involves changing the data pre-processing and network architecture to improve performance. Pre-processing is aimed mainly at eliminating more of the ambiguities in the input data -- or, equivalently, enhancing the uniqueness of subsets of the data.

The choice of the appropriate network architecture is to some extent driven by the nature of the pre-processed input. However, some general observations apply. The probabilistic net is not favoured for hardware implementation, since its size grows with the size of the training set; in our case the training set is potentially huge. Back propagation nets, however, can slow the research because of the typically long training time. The remainder of the experiment plan calls for investigation of various avalanche networks and self-organizing feature nets. Some consideration is also being given to integration of knowledge-based and neural techniques. Lockheed Canada is well placed to investigate this, since several of the in-house neural simulations have been written in Prolog. This renders the integration of the neural net with Prolog-based expert systems extremely easy.

6.0 CONCLUSIONS

Whilst a neural network is a computational structure modelled on biological processes, we believe that neural networks technology has defence applications, specifically as an important new technology for ESM systems. Their inherent parallel processing architecture offers the promise of extremely fast operation using wide bandwidth input data streams. Classification and Associative memory possibilities would look to be most promising with ES assistance as a longer term goal. Their ability to provide an adaptive, nonlinear capability could be most useful in ESM signal processing applications.

Lockheed Canada's experience with neural processing applications in ESM clearly establishes their usefulness as a supplement to conventional ESM processing techniques. It is clear that hardware is rapidly being developed to make realistic practical (rather than theoretical) assessments possible.

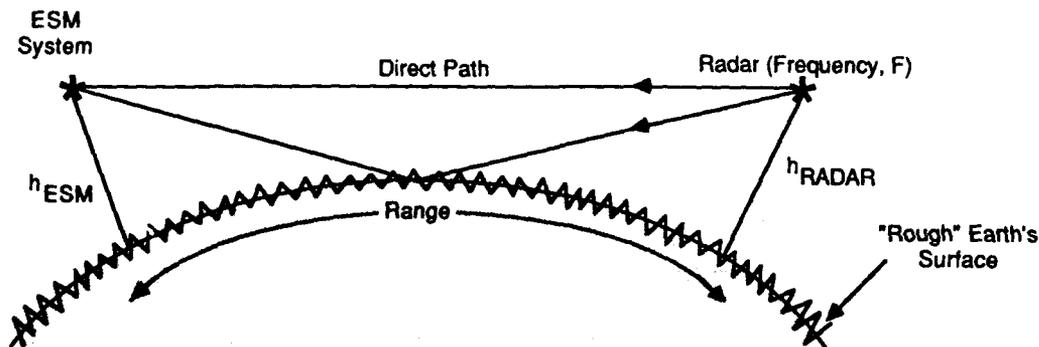


Figure 2.

REFERENCES

- 1) DARPA Neural Network Study, Fairfax, VA. AFCEA publication, Nov. 1988.
- 2) Using Objects to Design and Build Radar ESM Systems. Barry et. al., OOPSLA '87 Proceedings, Oct. 4-8, 1987, pp. 192-201.
- 3) Bearing Estimation using Neural Networks, Jha et. al., IEEE Int. Conf. on Acoustics, Speech and Signal Processing, vol. 4, pp. 2156-2157, New York, 1988.
- 4) Bearing Estimation using Neural Optimization Methods. Jha, et. al., First IEE Int. Conf. on Artificial Neural Networks, pp. 129-133, London, 1989.
- 5) Applications of Neural Networks in Military Systems. by A.R. Webb, MM '90, 11-13 July 1990.
- 6) Prototyping a Real-Time Embedded System in Smalltalk. B. Barry, OOPSLA '89 Proceedings, Oct. 1-6, 1989, pp. 255-265.

GLOSSARY

AFCEA	Armed Forces Communications and Electronics Circuit
AI	Artificial Intelligence
ASIC	Application Specific Integrated Circuit
CW	Continuous Wave
DARPA	Defense Advanced Research Projects Agency
EMCON	Emission Control
EME	Electromagnetic Environment
ES	Expert System
ESM	Electronic Support Measures
HADF	High Accuracy Direction Finding
IKBS	Intelligent Knowledge Based System
LPI	Low Probability of Intercept
MMI	Man Machine Interface
MOP	Modulation on Pulse
PDW	Pulse Descriptor Word
PRF	Pulse Repetition Interval
PRI	Pulse Repetition Frequency
RF	Radio Frequency
RISC	Reduced Instruction Set Computer
SBC	Single Board Computer
TOA	Time of Arrival
TSA	Tactical Situation Analyst
WAM	Window Addressable Memory

Discussion**1. Prof. F. Vagnarelli, Italy**

About the high accuracy direction finding, which kind of equipment has been utilized?

Author:

We have modeled a two-element vertical phase interferometer, against sea-skimming targets. Amplitude and phase information at each antenna is derived and stored for submission to the neural network.

2. T. Schang, France

Concerning the stagger extraction application, what happens when mixed signals are used as an input?

Author:

The net under discussion has a very specialized architecture. Mixed signals and missing data cause the performance to degrade and can cause absence of a clear "winner" in the response vector. This can be used as a signal that the data are not well clustered, triggering the use of other algorithms in the system. Because the network operates so quickly, we intend to apply it to all data coming from our channelizing hardware, using the other algorithms only when it fails. We are also considering training a standard back-propagation net to recognize the typical perturbations in the output vector caused by missing and interleaved data.



COMPONENT PART NOTICE

THIS PAPER IS A COMPONENT PART OF THE FOLLOWING COMPILATION REPORT:

(TITLE): Conference Proceedings of machine Intelligence
for Aerospace Electronic Systems Held in Lisbon, Portugal on
13-16 May 1991 (L'Intelligence Artificielle dans les Systemes
Electroniques Aero spatiaux)

(SOURCE): Advisory Group for
Aeronautical Research and Development, Neuilly-sur-Seine
(France)

TO ORDER THE COMPLETE COMPILATION REPORT USE AD-M242025

THE COMPONENT PART IS PROVIDED HERE TO ALLOW USERS ACCESS TO INDIVIDUALLY AUTHORED SECTIONS OF PROCEEDINGS, ANNALS, SYMPOSIA, ETC. HOWEVER, THE COMPONENT SHOULD BE CONSIDERED WITHIN THE CONTEXT OF THE OVERALL COMPILATION REPORT AND NOT AS A STAND-ALONE TECHNICAL REPORT.

THE FOLLOWING COMPONENT PART NUMBERS COMPRISE THE COMPILATION REPORT:

AD#: _____ TITLE: _____

AD-P 006 335

DTIC
SELECTED
NOV 10 1991

Accession For	
NTIS CRA&I	<input checked="" type="checkbox"/>
DTIC TAB	<input type="checkbox"/>
Unannounced	<input type="checkbox"/>
Justification	
By _____	
Distribution/	
Availability Codes	
Dist	Avail and/or Special
A-1	21

This document has been approved for public release and sale; its distribution is unlimited.

AD-P006 335



NEURAL NETWORK SOLUTIONS TO MATHEMATICAL MODELS OF PARALLEL SEARCH FOR OPTIMAL TRAJECTORY GENERATION

Lyle A. Reibling
 Smiths Industries Aerospace & Defense Systems, Inc.
 4141 Eastern Avenue, S.E.
 Grand Rapids, MI USA 49518-8727

1. SUMMARY

The Grand Rapids Division of Smiths Industries Aerospace & Defense Systems, Inc., has performed Independent Research and Development in neural network technology for vehicle management system applications of optimal trajectory generation in embedded systems. This paper describes the problem and solution method in envisioning massively parallel architectures which incorporate mathematical physics models for trajectory generation applications.)

A difficult problem in search applications is computing the optimal aircraft trajectory in real-time onboard a high performance aircraft, where the objective is to increase the aircraft survivability and mission effectiveness by penetrating enemy threats and minimizing threat radar exposure. Optimal trajectory generation is achieved by searching all possible paths in a multidimensional search space for the path with the smallest accumulated performance measure, which represents total threat exposure. This search problem has many state variables in a large space which places severe demands on computational resources, requiring real-time solutions unavailable from current technology.)

The mathematical model which is the basis for this architecture is based on an application of electrostatic field theory. It describes the problem of finding the best path through a region which contains a variable cost function as a problem in mathematical physics. One such description is that of finding the distribution of current flow through a nonuniform conducting media, such as a plate of inhomogeneous resistive material. Another is finding the propagation of wavefronts through a medium with a nonuniform refractive index.

This research has investigated computer architectures and algorithms characterized by

massive parallelism which can solve trajectory generation problems. An artificial neural network is defined which computes solutions to field theory. Experimental investigation of this technique has shown promising results. The solutions generated by the architectures have been checked against known admissible algorithm results and shown to be correct.

2. INTRODUCTION

Advances in integrated circuit and optical computing technology have made implementation of massively parallel computer architectures feasible. These architectures are more complex than multiprocessor networks, often mimicking the structure of the brain for their inspiration. Computation is not specified, or programmed, in the usual manner of a stored program computer. Rather, the collection of interconnection strengths between parallel processing elements determines the computation of the processing system.

The Von Neumann architecture is a serial processing architecture. These architectures require algorithms which are defined as serial instruction streams. Even multiprocessors, while processing in parallel as a group, are processing serially as individual processors. Problems growing in size and computational complexity make for more severe computing demands upon serial processing architectures. To meet the requirements of these larger, more complex problems, the processing architecture and algorithms need to be more closely matched with the nature of the application problem.

A general purpose sequential processor, or even a group of multiprocessors, may not be effective in solving traditional algorithms for these demanding problems. Many situations that require real-time solutions are satisfied with the rapid computation of near-optimal solutions. This may even be more desirable than

waiting for a slower computation of the optimal solution. Examples include optimal control problems such as aircraft flight control or trajectory generation in which a rapidly changing environment requires a fast solution from the processing architecture.

The motivation of this research is the investigation of computer architectures and algorithms characterized by massive parallelism which have computational properties that yield the solution of combinatorial search problems with application to path planning. The problem is discovering a model which maps onto massively parallel architectures.

3. NATURAL PARALLELISM

Increasingly complex problems create a need for the computing power offered by massively parallel architectures. Advancements in computer architectures and hardware technology have led to practical implementations of massively parallel computing. Because of its characteristic of collective computation, a new way[1] of thinking about the role of application, algorithm, and architecture in the design of massive parallelism is needed. The design challenge is to determine an appropriate model of the interdependence between the *application* problem to be solved, the *algorithm* which solves the problem, and the *massively parallel architecture* which solves the problem using collective computations.

Recent developments in massively parallel architecture technology has increased their capacity while shrinking their size. To capitalize on the advantages offered by these new architectures, a new approach to envisioning design is needed. This new approach is based on the capabilities of massively parallel architecture technology. Processing units of the architecture are dedicated to explicit, unique state assignments of the problem, given a suitable state variable resolution.

Consider how discovery of Nature is done. Nature is observed and natural law is hypothesized. This natural law is often expressed as equations in mathematical physics. Therefore, these equations describe Nature's behavior. When computing solutions to these equations, a numerical simulation of

Nature's behavior is performed. Now consider this viewpoint reversed. Nature is the perfect analog computer for these same equations. Nature is the analogy for solving problems described by the equations of mathematical physics.

Nature is also described by states and operations transforming these states. This provides another analogy to massively parallel computation. In traditional sequential computer systems, a single processor fetches data from memory, modifies the data, and puts the data back into memory. Consider however, memory where the data elements themselves have a computation capability. This computation is accomplished by exchanging and modifying values between the data elements of the memory. If the data elements of the memory correspond to the state assignments of a problem, then the values represent possible state variable solutions. This structure provides an explicit mapping between processing nodes of a massively parallel architecture and state assignments of a mathematical physics problem. The mapping between state assignments of the natural analogy and the processing nodes occur by a sampling of the state space into a computing grid with an appropriate resolution.

The process of applying the paradigm of natural parallelism is to begin by looking for analogies in Nature for the problem. When an appropriate analogy is found, the *natural parallelism concept* is formed. This conceptual embodiment within the paradigm meets the needs of the massively parallel design. The relationship to the application is understood and is explainable. The algorithm is identified through the natural law of mathematical physics which applies to the concept. The architecture is mapped through the numerical techniques yielding the computational structure.

3.1 Applying Natural Parallelism

The trajectory generation algorithm for the architecture is described as a problem in mathematical physics. This application uses an analogy of field theory for the algorithm of the mathematical model. The artificial neural network is used for the architecture. Two examples are provided. The first uses the electrostatic model to compute obstacle avoidance paths. This provides for multiple, alternative paths to avoid discrete or continuous obstacles. The second example uses

the wave propagation model for computing the optimal path.

These mathematical physics problems are expressed as partial differential equations. Thus, the partial differential equation is a natural parallelism in which the problem is viewed as one in computational physics. The variable cost function is similar to the nonuniform resistivity of the plate media, or the nonuniform refractive index. The solution to the appropriate partial differential equation is a scalar potential field for the media. Paths are computed from the vector field orthogonal to the equipotential contours of this field.

4. MATHEMATICAL MODELS

4.1 The Electrostatic Model for Avoidance Trajectory Generation

The mathematical model for the electrostatic based trajectory generation approach is an intuitive description of path planning. The mathematical model is based on a conjecture that the problem of finding the best path through a region which contains a variable cost function is analogous to the problem of finding the maximum current flow through a nonuniform conducting media, such as a plate of inhomogeneous resistive material. The variable cost function is analogous to the nonuniform resistivity of the plate media. Boundary conditions for the location of the source potential and sink potential are analogous to the start and goal nodes, respectively.

The mathematical model is based on electric field theory[2]. From the definition for current density and its divergence in steady state conditions, the n-dimensional electric field potential $\phi = \phi(x_1, x_2, \dots, x_N)$ of a nonuniform conductive media can be derived[3] as the second order partial differential equation

$$\nabla^2 \phi + \rho \nabla \sigma \cdot \nabla \phi = 0 \quad (1)$$

where $\rho = \rho(x_1, x_2, \dots, x_N)$ is the nonuniform resistivity of the media and $\sigma = \rho^{-1}$. The cost function is used to model the resistivity of the conducting media. Expanding the gradient and divergence operators in the two dimensions x and y results in the following second order partial differential equation

$$\phi_{xx} + \phi_{yy} + \rho \sigma_x \phi_x + \rho \sigma_y \phi_y = 0 \quad (2)$$

Since the electric field and current density results from the gradient of a scalar field, they are conservative fields, and the field lines emanate from source charges and terminate on sink charges[4]. These field lines represent the solution to the search problem as multiple paths. Several properties of the paths are noteworthy. First, every heading from the start node has a defined path due to the continuous vector field emanating from the source. Second, every location which has finite resistivity has a path defined through it for the same reason. Third, the model supports zero and infinite cost regions.

The flux (field) lines for the electrostatic model are used as the path definitions. The flux lines are computed once the electrostatic potential field is derived for the problem from the solution to the partial differential equation. There are an infinite number of flux lines which leave the source point and enter the sink point. This is because the source and sink points are singularities in the potential field solution. At every point in the potential field, there is a gradient vector (the electric field vector) with the exception of these two singularities. From the source point, an angle is selected from which to trace out the flux line. This is the initial heading at the start node (source point). Using a small incremental path step, the path progresses along this heading by the magnitude of the path increment. From this point on, the gradient vector can be computed which will define the direction of the path descent over the potential field surface, thus obtaining the path as the flux line.

4.2 The Wave Propagation Model for Optimal Trajectory Generation

The next development is to change the equations of the model in order to achieve the optimal path, rather than the avoidance paths of the electrostatic model. Theory pertinent to a model based on wave propagation phenomena relates various quantities which incorporate the cost function, such as wavefront velocity or refractive index. Fermat's Principle of Least Time explains the propagation of light through material with varying refractive index.

The mathematical model is based on wave propagation theory[5]. The wave equation is described as the condition of a physical quantity, Ψ , which satisfies

91 113 012
210 ST11 16

91-15517



$$\nabla^2 \Psi - \frac{\partial^2 \Psi}{\partial t^2} = 0 \quad (3)$$

where v is the phase velocity of the wave and may in general be a function of the space coordinates (a wave traveling in a non-homogeneous medium). The cost function is used to model the refractive index of the optical media. Equipotential wavefronts emanate from a light source. These wavefronts are orthogonal to the light rays.

The light rays form the paths of the solution. The light rays are computed from the equiphase field of wavefronts. This field is a solution to the wave equation. There are an infinite number of light rays leaving the source point. The source point is a singularity in the solution. The equiphase field is the phase distance of travel by the light rays. From any point, a gradient vector points to the direction of travel of the light ray from the source point. By tracing this vector back to the source point, the optimal path from the starting point to a goal node (the source point) is found.

An expression for the wave equation is desired so the solution is only a function of the spatial coordinates of the medium[6]. Periodic solutions of the wave equation can be found from Equation 3 where the solution is assumed to be separable into two functions. The function Ψ will depend only on the spatial coordinates, and the other function on time. From this is obtained

$$\nabla^2 \psi + K^2 \psi = 0 \quad (4)$$

where

$$K = \frac{2\pi v}{v} = \frac{\omega}{v} \quad (5)$$

where v is the frequency and v is the wavefront velocity. Consider v as the inverse of the cost function, C . The higher the cost, the slower the propagation velocity

$$v = \frac{1}{C} \quad (6)$$

The objective of the optimal path problem is to minimize the accumulated cost along the path

$$\min \int C ds \quad (7)$$

The cost function is modelled inversely as the wavefront velocity for the wave propagation model

$$C = \frac{1}{v} \quad (8)$$

As the cost is higher, the wavefront moves with lower velocity, incurring more periodic cycles of the wave and a higher accumulated phase distance along the wavefront. Since $v=ds/dt$, Equation 7, after substituting Equation 8, becomes

$$\min \int \frac{1}{ds} ds \quad (9)$$

which reduces to

$$\min \int dt \quad (10)$$

which is the principle of least time for the wavefront. Since the solution is only based on the spatial component, this is like taking a "snapshot" of the wave at some unique point in time. The accumulated phase along the light ray corresponds to the accumulated cost, and is a minimum value according to Fermat's Principle of Least Time.

5. NEURAL NETWORK IMPLEMENTATION

5.1 Implementing the Electrostatic Model

The partial differential equation for the electrostatic model was derived for the nonhomogeneous media, used to represent the cost function in generating paths which avoid regions of high cost. This section describes the architecture design of a neural network that computes the numerical solution to the equation. The architecture implements a finite difference approximation to the partial differential equation.

It is desired that the partial differential equation

$$L(u) = A \frac{\partial^2 u}{\partial x^2} + B \frac{\partial^2 u}{\partial x \partial y} + C \frac{\partial^2 u}{\partial y^2} + D \frac{\partial u}{\partial x} + E \frac{\partial u}{\partial y} + F u = 0 \quad (11)$$

be approximated on a unit grid by

$$L_1(u) = \alpha_0 u_P - \sum_{i=1}^n \alpha_i u_{Q_i} \quad (12)$$

with the n neighbors of P being Q_1, \dots, Q_N where $Q_i = (x+\xi_i, y+\eta_i)$. Using a Taylor series approximation for the expansion of u about P results in the system of equations

$$\begin{aligned} \sum_{i=1}^n \alpha_i - \alpha_0 &= F \\ \sum_{i=1}^n \alpha_i \eta_i &= E \\ \sum_{i=1}^n \alpha_i \xi_i &= D \\ \sum_{i=1}^n \alpha_i \eta_i^2 &= 2C \\ \sum_{i=1}^n \alpha_i \xi_i \eta_i &= B \\ \sum_{i=1}^n \alpha_i \xi_i^2 &= 2A \end{aligned} \quad (13)$$

The solution of the coefficients α_i of the above system of equations allows approximating the value of a grid point by solving for u_P

$$u_P = \frac{1}{\alpha_0} \sum_{i=1}^n \alpha_i u_{Q_i} \quad (14)$$

The nonhomogeneous Laplacian equation of the electrostatic model using the quantities $C_x = \sigma_x$ and $C_y = \sigma_y$ is

$$L_1(\phi) = \frac{\partial^2 \phi}{\partial x^2} + \frac{\partial^2 \phi}{\partial y^2} + C_x \frac{\partial \phi}{\partial x} + C_y \frac{\partial \phi}{\partial y} = 0 \quad (15)$$

and the coefficients of Equation 11 are $A=1$, $B=0$, $C=1$, $D=C_x$, $E=C_y$, and $F=0$. To derive a five-point finite difference formula the coefficients shown in Table 1 are substituted into the system defined by Equation 13 and yields the following system of equations

$$\alpha_1 + \alpha_2 + \alpha_3 + \alpha_4 = \alpha_0$$

$$\begin{aligned} \alpha_2 - \alpha_4 &= C_y \\ \alpha_1 - \alpha_3 &= C_x \\ \alpha_2 + \alpha_4 &= 2 \\ \alpha_1 + \alpha_3 &= 2 \end{aligned} \quad (16)$$

i	ξ	η
1	1	0
2	0	1
3	-1	0
4	0	-1

Table 1: Five-Point Coordinates

The solution to this set of equations (16) is

$$\begin{aligned} \alpha_1 &= 1 + \frac{C_x}{2} \\ \alpha_2 &= 1 + \frac{C_y}{2} \\ \alpha_3 &= 1 - \frac{C_x}{2} \\ \alpha_4 &= 1 - \frac{C_y}{2} \\ \alpha_0 &= 4 \end{aligned} \quad (17)$$

Substituting Equation 17 into Equation 14 and solving for u_P gives the finite difference formula

$$\begin{aligned} u_P &= \left(\frac{1}{4} + \frac{C_x}{8} \right) u_{Q_1} + \left(\frac{1}{4} + \frac{C_y}{8} \right) u_{Q_2} \\ &+ \left(\frac{1}{4} - \frac{C_x}{8} \right) u_{Q_3} + \left(\frac{1}{4} - \frac{C_y}{8} \right) u_{Q_4} \end{aligned} \quad (18)$$

The block diagram of the neuron model that implements the five point approximation of Equation 18 is shown in Figure 1.

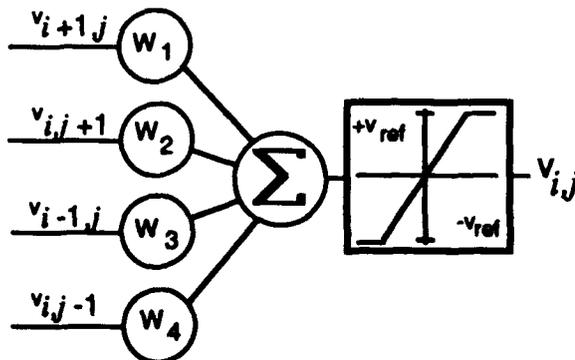


Figure 1: Five-Point Approximation Neuron

An artificial neural network, as defined by the model, uses an electronic circuit shown in Figure 2 as its basic neuron model, with the interconnecting resistors representing the synapses between the neurons. The numerical approximation is computing the solution to the system of equations (18).

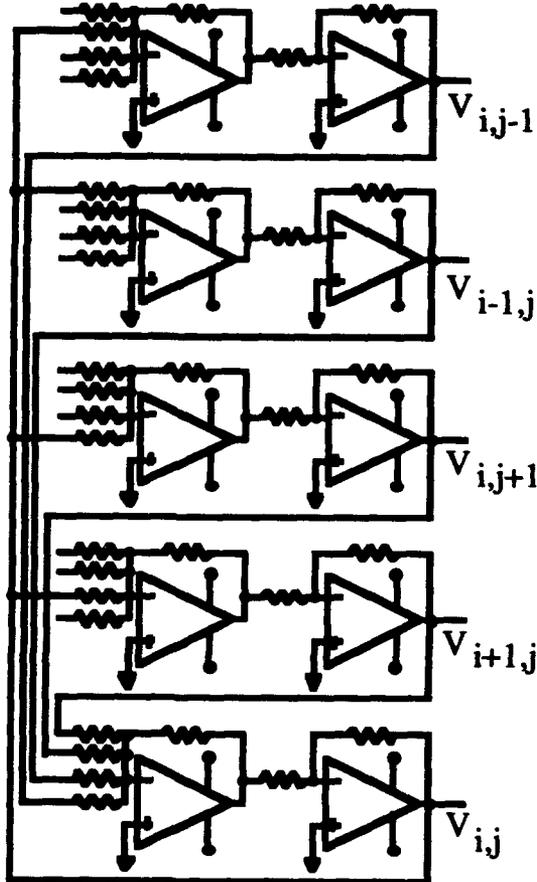


Figure 2: Numerical Approximation Circuit

The neural network is used to perform a parallel computation of the scalar potential field ϕ at every spatial point. For the neural network implementation, the neuron input function $u_{i,j}$ is defined as

$$u_{i,j} = \left(\frac{1}{4} + \frac{\sigma_x}{8\sigma}\right)v_{i+1,j} + \left(\frac{1}{4} + \frac{\sigma_y}{8\sigma}\right)v_{i,j-1} + \left(\frac{1}{4} - \frac{\sigma_x}{8\sigma}\right)v_{i-1,j} + \left(\frac{1}{4} - \frac{\sigma_y}{8\sigma}\right)v_{i,j+1} \quad (19)$$

The synaptic weights implement the non-uniformity of the cost functions (the coefficients of Equation 19 which contain σ_x

and σ_y . The non-linear neuron output function $v_{i,j}$ is defined as

$$v_{i,j} = \begin{cases} -V_{ref} & u_{i,j} < -V_{ref} \\ u_{i,j} & -V_{ref} \leq u_{i,j} \leq +V_{ref} \\ +V_{ref} & u_{i,j} > +V_{ref} \end{cases} \quad (20)$$

The entire search space is constructed by replicating this portion of the neural network over the entire grid. The output of the source and sink neurons which correspond to the start and goal nodes are clamped to $+V_{ref}$ and $-V_{ref}$ respectively. Since the clamping of the source and sink to the maximum and minimum (respectively) potential value occurs on the boundary of the problem, it is appropriate to use these clamped values as the limits of the neuron output function (Equation 20), since all potential values inside the boundary of the problem are guaranteed to lie between these limits.

The avoidance paths are extracted from the neural network. The network computes the scalar potential field, as described previously. This is a surface with a global maximum at the start node and a global minimum at the goal node. A direction is selected for a path at the start node. Gradient descent over the potential surface is used to extract the avoidance path for that direction. Bivariate interpolation of the potential values is used between the grid points to form smooth paths. An illustration of the model is shown in Figure 3. The contour lines represent the cost function for the problem. The field lines represent the avoidance paths generated by the model.

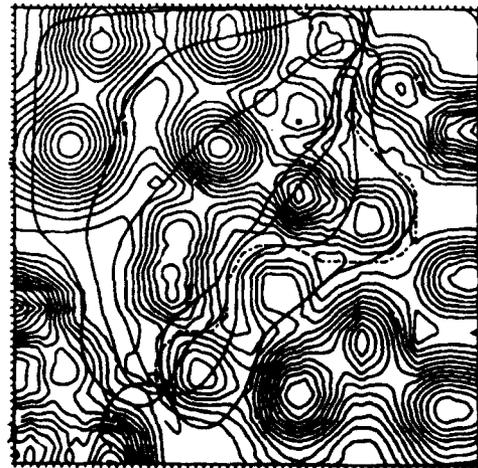


Figure 3: Electrostatic Model Example

5.2 Implementing the Wave Propagation Model

The wave equation was derived for the nonhomogeneous refractive index of the media, used to represent the cost function in generating the optimal path through the region modeled by the media. This neural network architecture computes the numerical solution to the wave equation. The architecture implements a parallel cost propagation for the accumulated phase distance of the wave equation.

The forward propagation of a wavefront in a single dimension, x , is defined with the wave equation

$$\frac{d^2\Psi}{dx^2} + K^2\Psi = 0 \quad (21)$$

The calculation of solutions to this equation also consists of a forward propagation of solution values (a computation wavefront) by using a difference formula. The central difference formula is

$$\Psi_{i+1} = (2-h^2K^2)\Psi_i - \Psi_{i-1} \quad (22)$$

K is related to the grid size, N , and maximum wavelength, so

$$(2-h^2K^2) = 2 \left(1 - \frac{2\pi^2C^2}{N^2C_{\max}^2} \right) \quad (23)$$

Substituting Equation 23 into Equation 22 gives the resulting approximation for the cost function grid

$$u_{i+1} = \left(1 - \frac{2\pi^2C_i^2}{N^2C_{\max}^2} \right) 2u_i - u_{i-1} \quad (24)$$

Since it is assumed that the accumulated phase distance is the accumulated cost function, the solution to the wave equation can be represented as

$$u_k = \cos(C_k + C_{k-1} + \Psi_{k-2}) \quad (25)$$

where without loss of generality $\Psi = \Psi_{k-2}$ and is defined according to the assumption above as

$$\Psi = \sum_{j=0}^{k-2} C_j \quad (26)$$

This means that

$$u_{k-1} = \cos(C_{k-1} + \Psi) \quad (27)$$

and also that

$$u_{k-2} = \cos\Psi \quad (28)$$

A stable solution is

$$u_k = \frac{\sin(C_k + C_{k-1})}{\sin C_{k-1}} u_{k-1} - \frac{\sin C_k}{\sin C_{k-1}} u_{k-2} \quad (29)$$

In two dimensions the wavefront is defined by

$$u_{i,j} = \cos(C_{i,j} + \min_{k,l} S_{k,l}) \quad (30)$$

where k,l are the neighbors of i,j , and also that

$$S_{k,l} = C_{k,l} + \min_{\text{path to } k,l} (\sum C's) \quad (31)$$

where C 's are the accumulated phase distance along the path to k,l .

The neural network architecture for computing solutions to the wave equation is similar to the wavefront propagation formula, but actually propagates costs corresponding to the accumulated phase distance of the wavefront. Hassoun[7] has described a technique for solving path optimization problems using a neural-like architecture with appropriate computational nodes. His approach is based on Dijkstra's algorithm with adaptations made for parallel computation. The architecture is a fine grained locally interconnected network to find the optimal path between two points. An alternative parallel architecture and algorithm is suggested by this research which improves Hassoun's algorithm. Figure 4 is the implementation of the improved neural cell for the network.

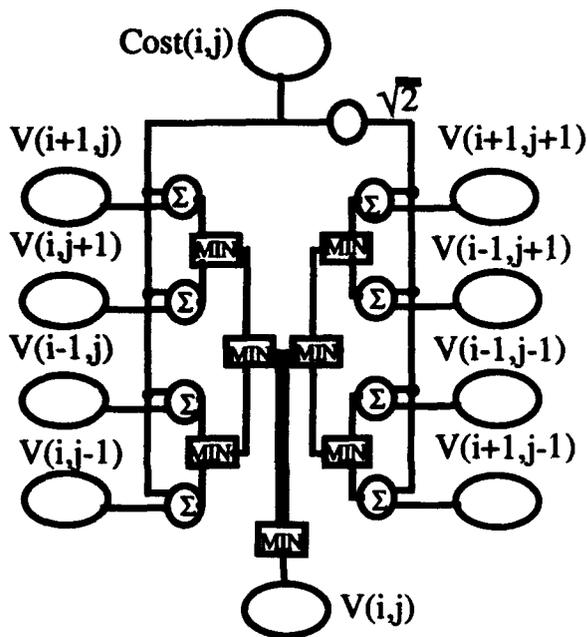


Figure 4: Neural Network Cell Implementation

An illustration of the model is shown in Figure 5. The contour lines represent the accumulated cost values for the problem. The optimal path generated by the model is also shown.

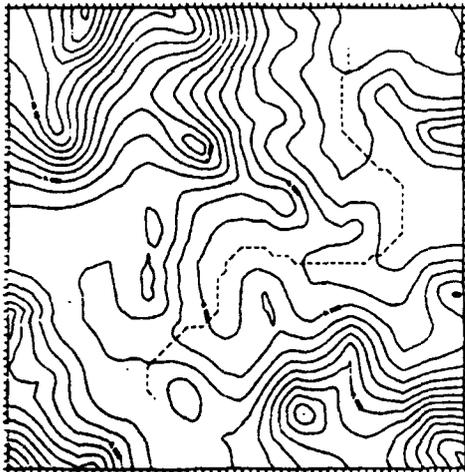


Figure 5: Wave Propagation Model Example

6. CONCLUSIONS

A significant result of this research was the use of a new paradigm to design massively parallel architectures. This new paradigm was called natural parallelism. The natural law which was employed to investigate the paradigm was a class of partial differential equations in mathematical physics. These partial

differential equations describe physical phenomena in electromagnetic field theory. Using natural parallelism, these physical phenomena were shown to describe certain optimization problems. Nature was viewed as an optimization process, exhibiting equivalent behavior for which mathematical expressions are available (that is, the partial differential equations of electromagnetic field theory).

Using the paradigm, an artificial neural network was designed which can solve a class of partial differential equations. Natural parallelism provided two mathematical models in electromagnetic field theory as analogies for the problem of trajectory generation. They were the electrostatic model and the wave propagation model. The architecture for the electrostatic model was shown to provide multiple, alternative, near-optimal solution paths. The wave propagation model was shown to solve the optimal path problem. Its neural network architecture was shown to compute the propagation of wavefronts by least cost propagation.

The partial differential equations were mapped onto the massively parallel architecture using finite difference approximations. The numerical solutions of the approximation formulas effectively simulated the analog computation of Nature. An architecture was designed to compute a five-point finite difference approximation solution to these equations. The five-point finite difference approximation formula provided for the definition of interconnection weights in the architecture. The massively parallel computing architecture was defined using explicit states of the problem as the computing nodes. This architecture computed solutions by exchanging information along interconnections defined in a mesh topology. As this exchange continued, each node's output converged to a value which represented the corresponding state's value in the problem solution. These state outputs were the scalar potential field solution to the partial differential equation. The approximation templates for the finite difference formula defined the interconnection weights for the particular architecture configuration. The significance of this design was a neuron transfer function and weight formulas for computing the finite difference approximation.

This research has shown the use of the natural parallelism paradigm to solve trajectory generation. Artificial neural networks were designed using the natural parallelism concept

and shown to provide good path solutions which correspond to natural analogies of electromagnetic field theory.

7. REFERENCES

- [1] L. A. Reibling. *Natural Parallelism as a Design Paradigm for Massively Parallel Architectures*. PhD Thesis, Michigan State University, June 1991. In preparation.
- [2] Allen Nussbaum. *ELECTROMAGNETIC THEORY for Engineers and Scientists*. Prentice-Hall, Inc., Englewood Cliffs, NJ, 1965.
- [3] L. A. Reibling. *Research and Development in Neural Network Technology and Applications*. Annual Report GR-O9P-0989, Smiths Industries Aerospace & Defense Systems, Inc., October 1989.
- [4] Ernst Weber. *Electromagnetic Fields*. Volume I— Mapping of Fields, John Wiley & Sons, Inc., New York, NY, 1950.
- [5] Henry Margenau and George Moseley Murphy. *The Mathematics of Physics and Chemistry*. D.Van Nostrand Company, Inc., Princeton, NJ, second edition, 1956.
- [6] D. H. Menzel. *Mathematical Physics*. Dover Publications, Inc., New York, NY, 1961.
- [7] Mohamad H. Hassoun and Ashvin J. Sanghvi. "Fast Computation of optimal paths in two- and higher-dimension maps", *Neural Networks*, 3:355-363, 1990.

Discussion

1. R. Klemm, Germany

Can you give some indication on how operational scenarios can be fed into your system? Do you consider optical techniques for parallel computing?

Author:

Scenarios begin with three elements. The current aircraft location in state space, and desired goal location in state space, and the cost function for the state space which includes the threat laydown and effects of terrain making. As the aircraft flies the trajectory, updates to these three elements cause new trajectories to be generated by the system. Dynamics of the environment can be treated through a fast-time prediction using the system interactively with cost function updates according to the threat dynamics model. We plan to investigate optical techniques this year as part of our efforts to develop a physical prototype device.

2. D. Bosman, Netherlands

With electric field equation model, trajectories are *orthogonal* giving iso-potential lines. In the physics analogy no work is involved in travelling along iso-potential lines. Does travelling in that direction involve accumulated danger? Does piece-wise selection of iso-potential trajectory parts provide acceptable alternative trajectories *not* calculated by the analogy?

Author:

Yes. Danger is accumulated along any path, including Iso-potential curves. This is unlike physics, where no work is performed. However, in the physics analogy, resistivity is accumulated along the paths (infinitesimally) as is the danger. No. Iso-potential parts of trajectories would not provide relief from danger. Unlike work, where all the current paths are equivalent in that they have the same end point potentials, these danger paths have unique accumulated danger. This is analogous to Ohm's law, since the current density field varies according to the resistivity.

3. G.L. Cohill, United States

Is the model described in the paper a *constant velocity* model? How would you accommodate a variable velocity parameter?

Author:

Yes. The cost function is developed as a rate of danger divided by velocity. This results in a cost measure along the line integral of the path. Thus, if velocity is constant, it can be factored out of the path integral. To answer the second question, I don't know. Otherwise, the velocity must be specified at every point in the problem space. This will incorporate velocity locally in the space. However, it won't guarantee a smooth velocity profile over the path itself. This requires the development of other ways to incorporate constraints into the system.



Application des Méthodes "RESEAUX DE NEURONES"
à la classification Automatique de cibles
par Jean-Luc REGEF & Jean QUIGNON
THOMSON CSF-DSE DT/SEP/SOP
9, rue des Mathurins, BP 150
92223 Bagneux cedex
France

Introduction

Ce document présente une application des méthodes "Réseaux de Neurones" à la classification automatique de cibles en imagerie infra-rouge.

Cette fonction de classification est un complément à la chaîne de traitements des systèmes optroniques de détection et de pistage. Elle permet d'améliorer les performances globales du système d'arme

- en limitant le taux de fausses alarmes et de fausses pistes,
- en apportant une aide à l'opérateur dans ses prises de décision.

Bien que le système de détection soit testé actuellement sur des cibles aériennes en vue de l'intégration à un système de défense sol-air, il est suffisamment général pour pouvoir s'appliquer également aux systèmes air-sol, air-air, ou à tout autre type de système optronique, de conduite de tir ou de veille, en vidéo infra-rouge ou visible.

Ce papier se propose d'étudier tout d'abord comment peut s'intégrer un module de classification automatique de cibles dans un système d'arme, et ce qu'il peut lui apporter. Puis, après une description technique de ce module, il présentera les premiers résultats obtenus en simulation sur des données réelles.

I Module de classification dans un système d'arme.

1.1. Description fonctionnelle d'un système d'arme, rôle de l'optronique.

Certains systèmes d'arme, embarqués ou non, sont équipés de sous-systèmes optroniques. C'est le cas notamment des systèmes de défense sol-air courte portée, mais aussi des systèmes air-sol ou des missiles fibro-guidés.

Ces équipements assurent les fonctions de détection, d'acquisition et de poursuite (ou d'accrochage) de cible(s). C'est la localisation de la cible qui permet le guidage du missile jusqu'à l'interception.

En ce qui concerne le sous-système optronique, ces fonctions sont généralement menées à bien par des techniques de traitement d'image. Ces techniques sont nombreuses et efficaces. Citons des algorithmes fondés sur la détection de contraste, l'analyse du mouvement ou encore la corrélation de zones d'images.

L'opérateur reçoit en temps réel la visualisation des scènes observées par les senseurs (infra-rouge ou visible), sur laquelle sont incrustés :

- la mire de visée,
- la (les) fenêtre(s) de poursuite.

Il peut intervenir pour :

- aider l'acquisition, par pointage manuel,

- valider ou inhiber une poursuite,
- prendre la décision de tirer.

1.2. Rôle du système automatique de classification de cibles.

Dans des environnements difficiles : fond très bruités, attaque saturante, contre-mesures (leures IR), incendies, ..., les systèmes automatiques risquent d'une part d'être saturés (trop de cibles potentielles) et d'autre part de surcharger l'opérateur (trop d'informations et de demandes de validation). Le rôle d'un système de classification automatique et d'aide à la décision est de pallier à ces inconvénients.

La classification s'effectue à deux niveaux. Au premier niveau, les cibles potentielles sont classées en terme de "cible" ou "fausse alarme". Les fausses alarmes correspondent aux leures IR, constructions, éléments de fond, etc. Cette première classification permet de réduire le taux de fausses pistes et allège ainsi la tâche de l'opérateur et la charge de calcul. Au deuxième niveau, la classification est effectuée sur les éléments de la classe "cible" et permet de caractériser leur catégorie (avion, hélicoptère, missile). Cette deuxième classification permet :

- d'adapter les algorithmes de pistage en conséquence,
- de donner une priorité à chaque cible poursuivie,
- de contribuer à l'évaluation de la menace,
- d'aider à l'évaluation de la qualité d'interception.

II Description technique.

2.1. Méthode neuronale.

Une méthode neuronale est une méthode basée sur l'apprentissage.

Les données d'apprentissage prennent ici la forme d'images qui segmentent la cible présumée. Elles doivent être extraites automatiquement du système de poursuite de façon à être opérationnelles. Ces images correspondent en fait aux alarmes détectées par le sous-système optronique de poursuite.

Un nombre important de données d'apprentissage est nécessaire : typiquement plusieurs dizaines de milliers. Il faut, dans la mesure du possible, que toutes les configurations et présentations possibles de cibles et de fausses alarmes susceptibles d'être rencontrées en phase opérationnelle soient représentées dans l'espace d'apprentissage. Si, par exemple, l'espace d'apprentissage ne contient pas d'oiseaux, il est probable qu'en phase d'exécution, le système les classe comme des avions !

Le réseau de neurones présenté ci-dessous est incapable d'inventer ce qu'il n'a jamais vu ainsi que d'élaborer des raisonnements compliqués, il apprend au contraire par l'exemple, de façon tout à fait empirique.

En phase d'exécution, le réseau de neurones reçoit en entrée une image provenant du sous-système optronique de pistage et rend en sortie le code de la classe reconnue.

2.2. Réseau de classification de cibles.

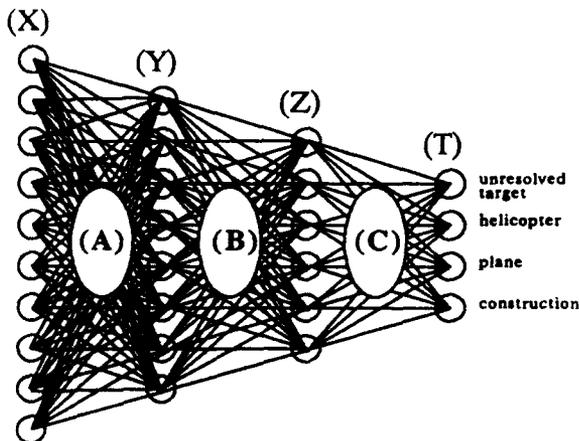
2.2.1. Structure du réseau.

Les images recues en entrée par le réseau sont tout d'abord pré-traitées de façon à leur donner une dimension fixe : 32*32 pixels (fig. 3).

Le réseau comporte quatre couches entièrement interconnectées (fig. 1) :

- la couche d'entrée correspond aux pixels de l'image à reconnaître. Elle forme un vecteur X de dimension 1024;
- le vecteur Y correspondant à la première couche cachée est de dimension 100;
- le vecteur Z (seconde couche cachée) est de dimension 30;
- le vecteur T (couche de sortie) est de dimension 4. Les coordonnées du vecteur de sortie correspondent aux différentes classes possibles (dans notre cas : cible en limite de portée, hélicoptère, avion, construction).

Les coordonnées de ces vecteurs sont appelées les "neurones" et sont des petits processeurs élémentaires. Les "neurones" d'une couche sont connectés à ceux de la couche suivante par des "synapses", qui sont en fait des coefficients de transmission. Ces coefficients (il y en a ici plus de 100 000), forment les matrices A, B et C (fig. 1). Toute l'"intelligence" du système se trouve dans la valeur de ces coefficients.



(fig. 1)

Le classement revient à effectuer les opérations suivantes :

$$Y = f(A \cdot X)$$

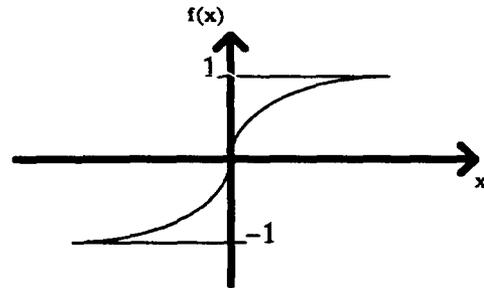
$$Z = f(B \cdot Y)$$

$$T = f(C \cdot Z)$$

où f est une fonction de seuillage (fig. 2).

Chacune des coordonnées du vecteur T représente la

réponse du système à la classe correspondante. La classe sélectionnée est celle qui donne la réponse la plus forte.



(fig. 2)

l'implémentation hard temps réel de ces multiplications vecteur-matrice est d'autant plus réalisable que les dimensions des différentes matrices sont faibles.

2.2.2. Phase d'apprentissage automatique.

La phase d'apprentissage a pour rôle le calcul de ses coefficients synaptiques :

Il est notamment important de minimiser le nombre de coefficients nécessaires pour faciliter la réalisation matérielle, tout en en gardant une quantité suffisante pour que les performances du système de classification soient optimales.

Les premiers essais d'apprentissage utilisant l'algorithme de la retro-propagation du gradient ont donné des résultats encourageants mais pas excellent (80% de bonnes reconnaissances). Une amélioration des performances a été obtenue en procédant en deux étapes de la manière suivante:

a- Première étape :

Elle est destinée à aider la seconde et la rendre plus performante. Elle consiste à regrouper les images de l'espace d'apprentissage topologiquement proches (c'est à dire : qui se ressemblent) et de même classe, en "nuages" linéairement séparables deux à deux. Ce regroupement en nuages s'obtient automatiquement par un réseau de type Kohonen (algorithme proche de celui des nuées dynamiques).

b- Seconde étape :

Elle est chargée du calcul proprement dit des coefficients synaptiques. La première étape permet de donner une signification précise à chacun des neurones des couches cachées, et de guider l'apprentissage. En effet chaque neurone de la première couche cachée est spécialisé dans la séparation des images de deux nuages donnés, les neurones de la seconde couche cachée correspondent aux nuages, et les neurones de la couche de sortie, aux classes. L'apprentissage se fait par retro-propagation du gradient couche par couche. Le système cherche donc à reconnaître le "nuage" auquel appartient une image avant d'en déduire sa classe.

Un simple perceptron à une couche et une sortie permet en effet de séparer avec de très bonnes performances tant en apprentissage qu'en généralisation deux imagerie appartenant à deux nuages donnés. Ce problème est nettement moins complexe que le problème global de classification à résoudre car les données à séparer sont topologiquement regroupées de façon cohérente, et sont linéairement séparables alors que les données du problème total sont fortement mélangées et dispersées. Les performances du perceptron par rapport au problème simple ne sont limitées que par le caractère plus ou moins flou des frontières séparant les nuages.

Après avoir décomposé l'espace d'apprentissage en sous-ensembles cohérents, le problème global de classification peut se décomposer en un grand nombre de problèmes moins complexes que l'on sait résoudre avec de bonnes performances.

Les réponses de tous ces perceptrons (il y en a plus d'une centaine) forment la première couche cachée du réseau (le vecteur Y). Il serait possible de déduire formellement la classe d'un objet de façon logique et statistique à partir de ces réponses. Nous préférons toutefois résoudre ce problème par apprentissage neuronal de façon à mieux tenir compte des valeurs réelles prises par chacun des neurones du vecteur Y. Nous apprenons ainsi au système, par l'algorithme de rétro-propagation du gradient, de déduire du vecteur Y le nuage d'appartenance de l'imagerie (le vecteur Z), puis enfin sa classe (le vecteur T).

Nous remarquons que seule la première couche de synapses travaille sur des données brutes. Ces synapses convergent vers des valeurs nuancées et non saturantes. Par contre les autres couches synaptiques ne travaillent que sur des données symboliques. Ces synapses convergent, eux, vers des valeurs beaucoup plus saturantes (-1 ou +1) ou neutres (0).

Ces algorithmes d'apprentissage sont actuellement implémentés sur une station de travail SUN 4, et tournent en temps différé sur des bases de données réelles extraites de notre bibliothèque de vidéos infra-rouges.

2.2.2. Phase d'exécution.

L'algorithme de la phase d'exécution est beaucoup plus simple que ceux de la phase d'apprentissage. Le but était en effet d'avoir en phase d'exécution un procédé facile à implémenter en temps réel alors que l'apprentissage pouvait s'effectuer en temps différé en laboratoire une fois pour

toutes.

Il s'agit dans un premier temps d'interfacer le système neuronal à un système de poursuite. Cette interface doit être capable de fournir en temps réel au réseau les imagerie sélectionnées, au format 32*32. Il est pour cela nécessaire d'implémenter un algorithme de zoom / dézoom adaptatif. Il n'y a pas nécessité d'implémenter un algorithme très sophistiqué : du moment que les imagerie de l'espace d'apprentissage ont été mise au format par le même algorithme, le système neuronal peut supporter des distorsions importantes.

Puis, la phase d'exécution proprement dite du classifieur consiste à effectuer l'opération : $T = f(C.f(B.f(A.X)))$. Il s'agit de trois multiplications vecteur-matrice et l'application de trois fonctions de seuillage (sigmoïde). Les multiplications vecteur-matrice peuvent s'implémenter efficacement sur un processeur de traitement du signal de type DSP, les seuillages peuvent s'implémenter sous la forme d'une table de conversion.

III Premiers résultats obtenus en simulation sur des données réelles.

Les performances obtenues quant à la discrimination cible / fausse alarme sont de l'ordre de 95% sur tout type de cibles ou de bâtiments, quelles que soient leur présentation et leur éloignement, sur fond de terre, de ciel ou de mer. Les bases d'apprentissage utilisées comportent plusieurs milliers d'imagerie.

Il reste toutefois à valider le procédé opérationnellement, sur une diversité encore plus grande de configurations.

Conclusion

Les résultats obtenus sur des données réelles montrent l'intérêt des méthodes neuronales pour la classification de cibles en optronique. Une maquette (temps réel) de cette fonction est en cours de réalisation, elle sera prochainement intégrée dans un système de poursuite automatique.

Cette approche "Réseaux de Neurones" peut être étendue aux équipements nécessitant une classification ou une reconnaissance automatique des cibles à atteindre dans un environnement complexe.

Ces méthodes fondées sur l'apprentissage constituent une avancée pour les systèmes futurs, "intelligents", dotés de vision artificielle et de capacité de décision.

Discussion

1. Jeff Grinshaw, United States

Was testing done on the training set?

Author:

No, of course testing was done on a testing set that was different than the training set.

2. Jeff Grinshaw, United States

Similar work is being conducted at the Air Force Institute of Technology (AFIT), United States. They use pre-processing of the pixel data. Have you considered using pre-processing, or is it not necessary?

Author:

Pre-processing of the pixel is certainly not a necessity because it works without pre-processing, and it does it very well, and humans do not use pre-processing which is not neural pre-processing. But it is not forbidden to use pre-processing, it can just be more complicated. Most of pre-processing can be implemented into a neural network (with local connections for example).

3. D. Bosman, Netherlands

To a Region of Interest (ROI) of 1024 pixels, usually very little pixels contribute to the recognition process. In a 1000-dimensional space each aspect of every object occupies a cloud which must not interfere with neighboring clouds; if

so, uncertainty results. You stated that the system was trained on 5000 objects. How many aspect/object clouds can be safely (at P% probability of confidence) stored in the 1k dimensional space? Is the amount of 5000 mentioned, given the large volume of the clouds because little numbers of pixels participate in the recognition, already overstraining available capacity?

Author:

In fact the 1024 pixels contribute to the recognition process and not only a very few of them. But as they are linked together, in fact, they can be compressed. The low number of "Clouds" needed is due to the fact there is a high possible compression rate. The way to compress the information to very few symbolic elements which are the classes to recognize, is the aim of the neural learning process.

4. Glenn Johnson, United States

How does the system handle scaling of the image size? Does the system require retraining for images of different sizes? Does not this make for a very large training set?

Author:

There is no pre-processing except that subimages are extracted and formatted into the 52 x 32 pixel format. The identifier must be trained to recognize these scalings of the image, only in the sense that scaling of, for example, 6 pixel image has less detail than scaling of 12 pixel image.

COMPONENT PART NOTICE

THIS PAPER IS A COMPONENT PART OF THE FOLLOWING COMPILATION REPORT:

(TITLE): Conference Proceedings of machine Intelligence
for Aerospace Electronic Systems Held in Lisbon, Portugal on
13-16 May 1991 (L'Intelligence Artificielle dans les Systemes
Electroniques Aero spatiaux)

(SOURCE): Advisory Group for
Aeronautical Research and Development, Neuilly-sur-Seine,
(France)

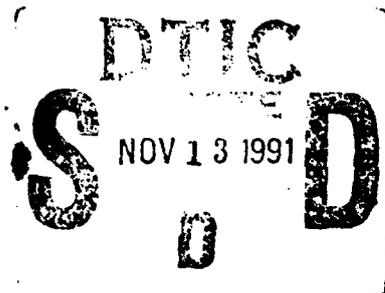
TO ORDER THE COMPLETE COMPILATION REPORT USE AD-M242025

THE COMPONENT PART IS PROVIDED HERE TO ALLOW USERS ACCESS TO INDIVIDUALLY AUTHORED SECTIONS OF PROCEEDINGS, ANNALS, SYMPOSIA, ETC. HOWEVER, THE COMPONENT SHOULD BE CONSIDERED WITHIN THE CONTEXT OF THE OVERALL COMPILATION REPORT AND NOT AS A STAND-ALONE TECHNICAL REPORT.

THE FOLLOWING COMPONENT PART NUMBERS COMPRISE THE COMPILATION REPORT:

AD#: _____ TITLE: _____

- AD-P006 331
- AD-P006 334
- AD-P006 336
- AD-P006 344
- AD-P006 345
- AD-P006 346
- AD-P006 352



NOT REPRODUCIBLE
FOR PUBLICATION OR
DISTRIBUTION

Accession	
NTIS Card	
DTIC TAB	
Unannounced	
Justification	
By	
Distribution	
Availability Codes	
Dist	Avail and/or Special
A-1	

AD-P006 336



THREAT LOCALIZATION
A major breakthrough for Intelligent E.W. Systems

C. MAILLARD
DASSAULT ELECTRONIQUE (D.E.)
55, quai Marcel Dassault
92214 Saint-Cloud
France

PLANCHE 1 : THREAT LOCALIZATION

La survie (survivability) d'un avion de combat moderne est très liée à sa capacité à déterminer la situation tactique précisément, en temps réel.

Nous allons montrer l'impact de la localisation des menaces sur l' "intelligence" des systèmes de contremesures.

Auparavant, nous souhaitons vous présenter DASSAULT ELECTRONIQUE (D.E.).

PLANCHE 2 : SOFTWARE

En ce qui concerne les capacités en logiciel de D.E. :

- 850 spécialistes logiciel,
- création d'une filiale "DE3I" avec IBM,
- 9 ordinateurs centraux, et plus de 4 000 terminaux.

Enfin, D.E. prend place parmi les leaders en Intelligence Artificielle. Grâce à une solide expérience de plus de 10 ans, D.E. a, en utilisant les techniques de l'Intelligence Artificielle, développé et mis au point nombre d'outils qui ont permis de réaliser différentes applications et d'aboutir à des produits.

Ces techniques et outils généraux sont :

- une méthodologie de développement de Systèmes Experts : **MEDIUS**, en complément de la méthodologie de développement de logiciel **MINERVE**,
- le langage **PROLOG**, et surtout le développement d'**EMICAT**, extension de **PROLOG** vers un langage orienté objet permettant l'acquisition de connaissance complexes grâce à des mécanismes de

"frame", d'héritages, de réflexes et de règles de production.

EMICAT est distribué mondialement par **IBM** sous le nom d'**IPW** :

- la participation à **NESTOR EUROPE** dans le domaine des réseaux neuromimétiques ;
- l'utilisation de **VISILOG** pour les études et les développements de projets en traitement d'images.

PLANCHE 3 : A.I. APPLICATIONS

Les applications réalisées chez **D.E.** couvrent de nombreux domaines :

- l'aide logicielle intégrée pour l'informatique documentaire,
- l'aide au diagnostic technique (**CATS** pour les circuits analogiques à base de modèles, **GIBUS** utilisé opérationnellement pour la gestion des batteries de satellites),
- l'aide à la décision et l'aide au commandement (génération de scénarios de déplacement d'armée : **PEGASE** et simulation de la bataille aéroterrestre : **CARNEADE**),
- la synthèse logique de circuits intégrés spécifiques (**ASICs**) : **FRENCHIP**. **FRENCHIP** est commercialisé sous **UNIX**.
- l'aide à l'exploitation de satellites (système de contrôle et de commande de satellite **SATEXPERT**)
- **DASSAULT ELECTRONIQUE** effectue de nombreuses recherches dans le domaine de l'Intelligence Artificielle en

temps réel : un système expert temps réel vous sera présenté en détail par Monsieur SERVEL de D.E. (conférence n° 18).

Enfin, le système d'identification de menaces SEISME, avec de fortes contraintes sur le temps de réponse, est le coeur de ce papier.

Avant de cerner l'impact de l'I.A. sur les systèmes électroniques de défense, il faut commencer par définir les besoins opérationnels en autoprotection.

PLANCHE 4 : FUTURE RWR REQUIREMENTS

L'évolution des menaces est caractérisée par :

- leur agilité,
 - leur mobilité,
 - leur diversité,
 - leur densité.
- * Contre l'agilité des paramètres (fréquence, PRF, PW, ...), une mesure très précise de la direction d'arrivée (DOA) est essentielle : la DOA est le meilleur paramètre de tri, car le seul stable dans le temps.
- * Les autres caractéristiques des menaces rendront leur tri et leur identification de plus en plus difficile. Il est donc nécessaire d'améliorer les capteurs, et bien sûr le traitement.

PLANCHE 5 : RWR BLOCK DIAGRAM

Pour faire face à l'évolution des menaces, une architecture du type présenté sur la planche est nécessaire. Outre les capteurs "classiques", des capteurs de nouvelle génération (Interféromètre large bande instantanée, IPM, Analyse spectrale temps-réel) seront intégrés au niveau de l'impulsion pour obtenir un temps de réponse suffisamment court pour envisager les actions adéquates (brouillage, leurrage, évasive, ARM, ...).

PLANCHE 6 : Threat localization principe

De plus, la mesure précise de l'angle d'arrivée (DOA), (typiquement 10 fois meilleure que celle des équipements standard) essentielle pour le tri et le dé-entrelacement, permet aussi de localiser géographiquement les émetteurs de manière passive, grâce au principe de triangulation déjà démontré sur les équipements ESM.

PLANCHE 7 : Operational advantages

Les avantages opérationnels de la localisation des menaces sont très nombreux, et augmentent considérablement les capacités opérationnelles de l'avion.

De nombreuses notions font appel à un traitement intelligent : corrélation tactique de menaces, ordres d'évitement de la menace la plus dangereuse, calcul de priorité, et bien sûr amélioration de l'identification des menaces (ESM). Mais comment faire ?

PLANCHE 8 : PILOT INTERFACE

Le pilote a besoin d'une interface claire, précise et donc synthétique. La localisation des menaces et un traitement intelligent, sont essentiels pour satisfaire ce besoin.

PLANCHE 9 : PROCESSING

Le traitement nécessaire est décrit sur cette planche pour élaborer la situation tactique en temps réel :

- capteurs mesurant entre autres la direction d'arrivée (DOA) des menaces avec une grande précision,
- une corrélation impulsion par impulsion entre capteurs, basée sur le temps d'arrivée des impulsions,
- un pré-filtrage, utilisant une matrice de comparateurs à fenêtre (Window Addressable Memory) qui permet de réduire considérablement le flot de données par :

- la reconnaissance des plots déjà pris en compte (désentrelacement, ...),
- la mise en évidence, très rapidement, de tout changement de l'environnement (nouvelle menace, changement ou évolution des paramètres d'une menace),
- un processeur pour la localisation des menaces,
- et un calculateur de haut niveau (PMF) pour l'identification, utilisant des techniques d'IA que nous allons développer.

Les deux idées importantes sont les "feedbacks" qui permettent :

- de positionner les valeurs minimum et maximum du pré-filtrage à partir des résultats de la localisation,
- de modifier les capteurs en fonction des résultats d'identification (en cas d'ambiguïté par exemple).

PLANCHE 10 : PRE-FILTERING

Ce tableau résume très schématiquement trois approches possibles pour le pré-filtrage :

- approche "classique",
- approche par "Window Addressable Memory" (WAM), retenue par D.E.,
- approche par systèmes neuronaux.

Des conférences sont prévues lors de ce symposium sur ce point particulier.

Nous pensons, suite aux recherches effectuées, que le pré-filtrage est une bonne application possible à terme pour les systèmes neuronaux, mais il faudra résoudre les difficultés liées à l'intégration du temps (PRF, séquences, ...).

PLANCHE 11 : PHOTO

PLANCHE 12 : A.I. Threat Identification

Cette longue introduction était nécessaire pour bien situer le problème lié à l'identification des menaces.

En résumé, l'évolution des menaces conduit à la nécessité d'améliorer leur analyse :

- analyse spatiale,
- analyse temporelle,
- et bien sûr, mesures des paramètres électromagnétiques (RF).

En conclusion, une description "intelligente" de la menace est nécessaire, et doit être utilisée.

PLANCHE 13 : SEISME

Le système Expert "SEISME" élabore la situation tactique en temps réel.

Les axes de recherche pour ce Système Expert sont :

- modélisation,
- raisonnement dans un monde imprécis et incomplet,
- résolution d'ambiguïté,
- détermination des heuristiques nécessaires,
- évaluation de la validité de la situation tactique élaborée,
- reprogrammation pour reconfigurer le système.

Le but de SEISME a été d'appliquer ces techniques au problème de l'identification, avec des contraintes en temps réel.

PLANCHE 14 : ARCHITECTURE

L'architecture retenue est décrite sur la planche.

On notera :

- les pistes (Tracks) provenant du pré-filtrage,
- les fiches (Forms) techniques,
- les phases (Phases) temporelles,
- les paramètres du système expert.

91-15518



91 1112 018

Le résultat est l'identification des menaces quasi en temps réel.

PLANCHE 15 : DESCRIPTION

La définition de chaque terme est précisée sur cette planche.

A noter qu'une piste est constituée des positions angulaires, distance ainsi que des paramètres ElectroMagnétiques E.M. (RF).

PLANCHE 16 : IDENTIFICATION

La clé d'une bonne identification est la séparation entre les caractéristiques géographiques et électromagnétiques.

Cette discrimination spatiale est essentielle.

Un "mapping" est effectué en cas de non-discrimination spatiale entre les pistes RF et les fiches.

Ensuite, une association instantanée est effectuée, basée :

- sur la correspondance (agreement) du mapping,
- sur le nombre de menaces identifiées,
- sur le nombre de fiches manquantes pour les menaces identifiées.

Des heuristiques sont nécessaires pour maîtriser l'explosion combinatoire résultant de l'association possible de plusieurs menaces pour chaque fiche.

PLANCHE 17 : HEURISTICS

L'idée est d'effectuer un partitionnement du problème, car :

- l'interprétation de quelques pistes est indépendante d'autres pistes,
- peu de pistes évoluent en fonction du temps (l'angle d'arrivée (DOA) évolue très lentement en fonction du temps).

Ces évolutions sont détectées grâce au pré-filtrage.

→ On ne traite donc que les partitions évolutives.

→ L'influence de la précision sur la DOA est capitale sur ce partitionnement, comme le suggère le schéma : il vaut mieux 5 problèmes indépendants de niveau simple que 2 problèmes de niveau très complexes.

PLANCHE 18 : MESA

Le but essentiel de SEISME est de valider l'apport des techniques d'intelligence artificielle pour évaluation de la situation tactique (identification, calcul de priorité, ...)

Ce système expert a donc été couplé au simulateur MESA ("Modélisation et Evaluation des Systèmes d'Autoprotection") qui permet de simuler les conditions opérationnelles, en particulier en tenant compte des relations, liens entre menaces et de leur caractère "réactif", ainsi bien sûr que la modélisation du système d'autoprotection (précision sur la Direction d'Arrivée (DOA accuracy), précision de localisation, ...).

PLANCHE 19 : SEISME Evaluation

Nous avons donc évalué les performances de SEISME dans un environnement opérationnel complexe simulé par MESA.

Nous avons aussi développé de nouveaux principes d'Intelligence Artificielle.

Des extensions sont prévues :

- évitement de menace,
- calcul de priorité.

Enfin, les possibilités d'embarquer ce système sont étudiées (en particulier sur une machine parallèle : Transputer ou machine RISC).

En conclusion, nous avons évalué l'importance relative des différents critères : la discrimination angulaire est essentielle pour obtenir un temps de réponse suffisamment court, car elle

permet de contrôler l'explosion combinatoire.

PLANCHE 20 : CONCLUSION

Les améliorations de performances pour l'identification des menaces avec les techniques d'Intelligence Artificielle ont été démontrées.

Une mesure très précise de l'angle d'arrivée des menaces (et leur localisation) est essentielle pour respecter les contraintes de temps réel.

INTELLIGENCE EW SYSTEMS **DEFENSIVE SYSTEMS ELECTRONICS**

THREAT-LOCALIZATION

A MAJOR BREAKTHROUGH FOR INTELLIGENT E.W. SYSTEMS

**C. MAILLARD
DASSAULT ELECTRONIQUE
55 QUAI MARCEL DASSAULT
92214 SAINT-CLOUD
FRANCE**

The AGARD Aerospace Panel Meeting
LEIGN 13-16 MAY 1991

INTELLIGENCE E.W. SYSTEMS

A.I. APPLICATIONS

- Software Engineering: Information-processing⁽¹⁾, software quality⁽²⁾
- Diagnostic: CATS⁽³⁾, GIBUS⁽⁴⁾....
- CBI PEGASE⁽⁵⁾, CARNEADE Knowledge-Based simulation
- Logical synthesis of ASICs PRENCHP⁽⁶⁾
- Satellite control and exploitation SATEXPERT
- Real-time Avionic
 - Failure handling
 - Expert System for Threat-identification SEZAME⁽⁷⁾
- Neural network design

INTELLIGENCE E.W. SYSTEMS **FUTURE RWR REQUIREMENTS**

THREAT EVOLUTION **RWR REQUIREMENTS**

- **AGILITY** → Direction of Arrival (DOA) = best sorting parameter (the only stable in time)
- **MOBILITY** → Tactical Situation Awareness
Precise real-time Localization
- **DIVERSITY** → Wide Band Receiver : 100 % POI
Superheterodyne Mode :
Excellent sensitivity (LPI)
- **DENSITY** → Improved Sorting Capability :
 - Interferometer for DOA
 - IFM for pulse frequency
 - Fast and powerful processing (VAMP/PMF)

INTELLIGENCE E.W. SYSTEMS

THREAT LOCALIZATION PRINCIPLE

- Phase interferometer for very accurate DOA (Ten times better than a standard RWR)
- PRECISE PASSIVE RANGING BY TRIANGULATION
- PRINCIPLE ALREADY PROVEN IN ESM EQUIPEMENTS

INTELLIGENCE E.W. SYSTEMS **LOCALIZATION OF THREATS OPERATIONAL ADVANTAGES**

TACTICAL SITUATION AWARENESS :

- ▶ Prewarning
- ▶ Tactical correlation

THREAT AVOIDANCE - MINIMUM RISK ROUTING

PRIORITY ASSESSMENT :

- ▶ Tactical correlation
- ▶ Determination of firing envelopes

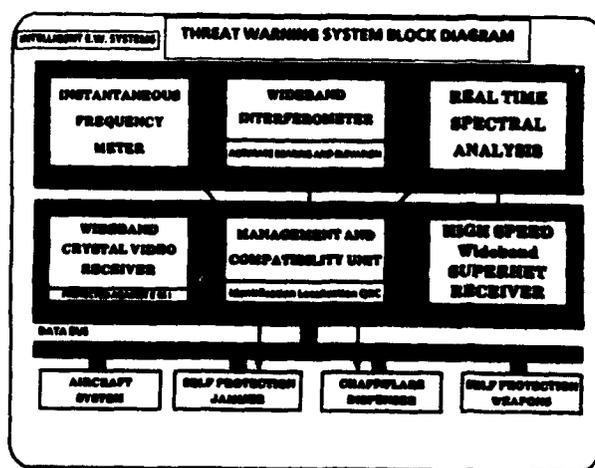
ADEQUATE TIMING FOR EXPENDABLES

PASSIVE TARGET DESIGNATION :

- ▶ SEAD mission
- ▶ Designation for ARM, SPARM

ESM CAPABILITIES

→ LOCALIZATION STRONGLY IMPROVES THE OPERATIONAL PERFORMANCES OF THE AIRCRAFT



INTELLIGENCE E.I.R. SYSTEMS

NEW-GENERATION DEFENSIVE SYSTEMS

- DETECTION and PASSIVE tracking.
- ANALYSIS on a pulse-by-pulse basis
- LOCALIZATION.
- IDENTIFICATION and PRIORITY ASSESSMENT
- PILOT INTERFACE
- FLIGHT REPORT for maintenance and ESM
- PROTECTION against all types of threat
- INTEGRATION and COMPATIBILITY

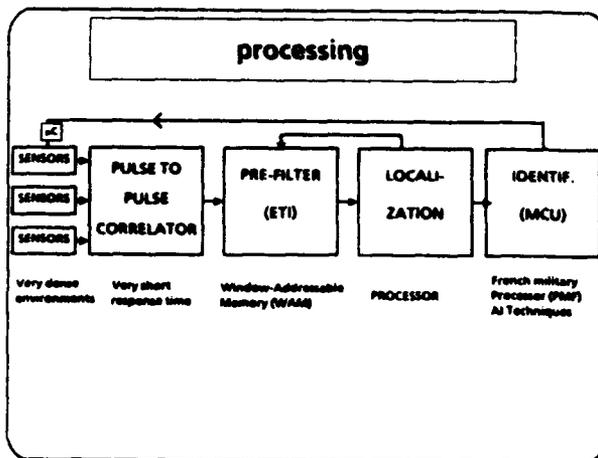
INTELLIGENCE E.I.R. SYSTEMS

CLEAR AND ACCURATE PILOT INTERFACE

AURAL WARNING : PROGRAMMABLE SYNTHETIC VOICE

TACTICAL SITUATION DISPLAY (TSD) : PRECISE RANGING INFORMATION

3" HIGH RESOLUTION OR MULTIFUNCTION DISPLAY (MFD)



INTELLIGENCE E.I.R. SYSTEMS

PRE-FILTERING

CHARACTERISTICS	NEURAL NETWORKS	WAM	CLASSICAL #
Response Time	Fixed, very short	Very short	Short
Parallel Processing	Very high	High	No
Distributed Memory	Yes	No	No
Logic	Analog	Digital	Digital
Noise Robustness	Integrated	"Addressable"	FIXED
Programming by	Experiment	Instructions	No reprogramming
Reliability	Tolerant	Good	Average
Hard-soft integration	Very deep	No	No

PRE-FILTERING : A GOOD POSSIBLE APPLICATION FOR NEURAL NETWORKS

BUT DIFFICULTY FOR TIME INTEGRATION (PRE...)

INTELLIGENCE E.I.R. SYSTEMS

A.I. FOR THREAT IDENTIFICATION

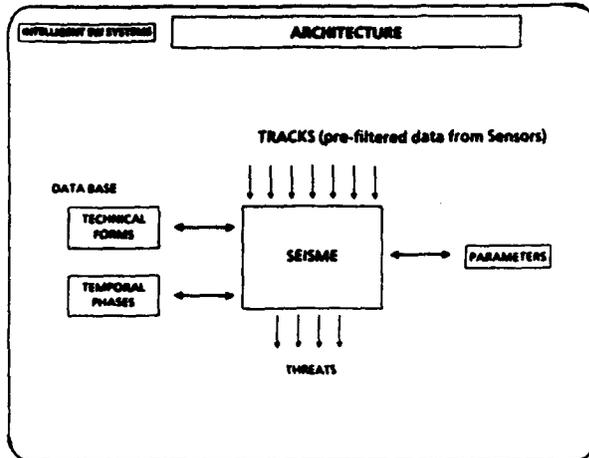
- **PRESENT EQUIPMENT :**
 - Very small part of the available information used for identification
 - Low density
 - Poor angular discrimination
- **FUTURE EQUIPMENT :**
 - Situational awareness based on all the available data
 - Very dense environments
 - Accurate DOA
- **CONSEQUENCES ON THREAT and MISSION ANALYSIS**
 - Spatial Analysis (organization rules, netted defense, geographical repartition, intercorrelation ships)
 - Time Analysis (changes of Emittor modes, sequences, ...)
 - R.F. parameters (PW, PRF, DOA, FREQ)

CONCLUSION : Intelligent description required for the Threat Library

INTELLIGENCE E.I.R. SYSTEMS

SEISME : Elaboration of real-time Tactical Situations

- **MODELING** (to represent the threats and tracks)
- **REASONING** in an imprecise and incomplete world
- **AMBIGUITY RESOLUTION** (Tests and Evaluation Functions)
- **HEURISTICS** to handle combinational explosion
- **EVALUATION** of reliability of the elaborated tactical data
- **PERSISTENT PROGRAMMING** to reconfigure the system :
 - definition of threat libraries
 - rules for the removal of ambiguities

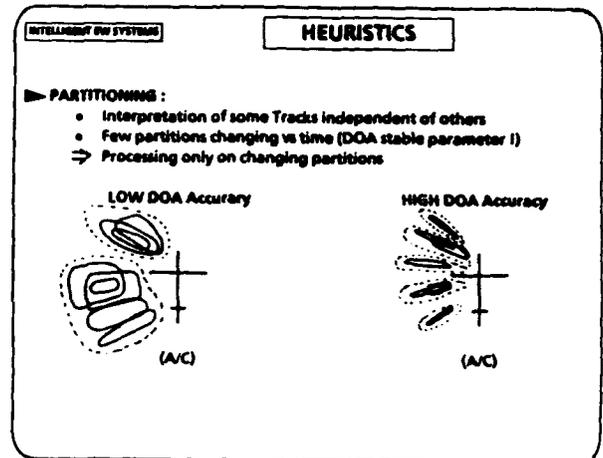
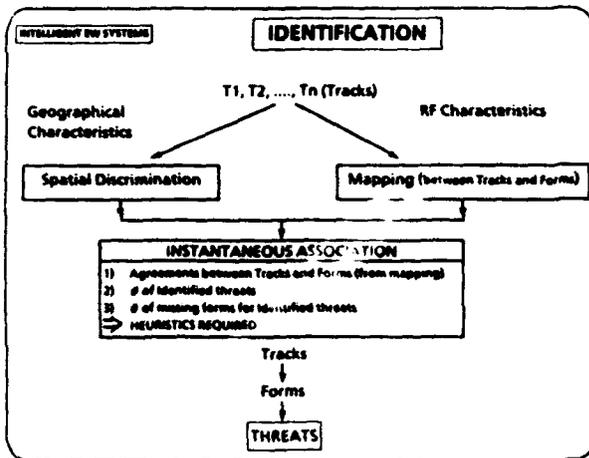


INTELLIGENT EW SYSTEMS **DESCRIPTION**

► **Technical Forms :**
 RF parameters of threats (F, PRF, PW, TOI, ...)
 Types of emission (Agility, compression, polarization, ...)

► **Phases :**
 Logical expression of technical forms (to describe a complete Weapon System)
 $\phi 1 = \text{FORM 1}$
 $\phi 2 = \text{FORM 1 <AND> FORM 2}$
 $\phi 3 = \text{FORM 3 <OR> FORM 4}$

► **Tracks :**
 - Geographical positions (EL, AZ, RANGE) + RF data (same as technical forms)



INTELLIGENT EW SYSTEMS **SEISME EVALUATION**

- OPERATIONAL ENVIRONMENT SIMULATED BY M.E.S.A.
- DEVELOPMENT OF NEW A.I. PRINCIPLES
- EXTENSION TO THE SYSTEM (Threat Avoidance, Priority Assessment, ...)
- POSSIBILITY OF EMBEDDING (Porting on a parallel machine, ...)

• **RELATIVE EVALUATION BETWEEN CRITERIAS :**

⇒ ANGULAR DISCRIMINATION ESSENTIAL FOR A SHORT RESPONSE TIME (TO HANDLE COMBINATIONAL EXPLOSION)

INTELLIGENT EW SYSTEMS **CONCLUSION**

- PERFORMANCE IMPROVEMENTS WITH ARTIFICIAL INTELLIGENCE PROVEN FOR EMITTER IDENTIFICATION
- ACCURATE DOA MEASUREMENT (AND EMITTER LOCALIZATION) ESSENTIAL FOR REAL-TIME REQUIREMENTS



COMPONENT PART NOTICE

THIS PAPER IS A COMPONENT PART OF THE FOLLOWING COMPILATION REPORT:

(TITLE): Conference Proceedings of machine Intelligence
for Aerospace Electronic Systems Held in Lisbon, Portugal on
13-16 May 1991 (L'Intelligence Artificielle dans les Systemes
Electroniques Aero spatiaux)

(SOURCE): Advisory Group for
Aeronautical Research and Development, Neuilly-sur-Seine,
(France)

TO ORDER THE COMPLETE COMPILATION REPORT USE AD-M242025

THE COMPONENT PART IS PROVIDED HERE TO ALLOW USERS ACCESS TO INDIVIDUALLY AUTHORED SECTIONS OF PROCEEDINGS, ANNALS, SYMPOSIA, ETC. HOWEVER, THE COMPONENT SHOULD BE CONSIDERED WITHIN THE CONTEXT OF THE OVERALL COMPILATION REPORT AND NOT AS A STAND-ALONE TECHNICAL REPORT.

THE FOLLOWING COMPONENT PART NUMBERS COMPRISE THE COMPILATION REPORT:

ADM: _____ TITLE: _____

AD-P006 337

DTIC
ELECTE
NOV 13 1991
S D D

Accession For	
NTIS GRA&I	<input checked="" type="checkbox"/>
DTIC TAB	<input type="checkbox"/>
Unannounced	<input type="checkbox"/>
Justification	
By _____	
Distribution/	
Availability Codes	
Dist	Avail. or Special
A-1	21

Approved for Release by NSA on 05-08-2014 pursuant to E.O. 13526

AD-P006 337



A NASA/RAE Cooperation in the Development of a Real-Time Knowledge Based Autopilot

Colin Daysh'
 Malcolm Corbin'
 Geoff Butler'
 Eugene L. Duke°
 Steven D. Belle*
 Randal W. Brumbaugh*

1. Summary

As part of a US/UK cooperative aeronautical research programme, a joint activity between NASA Ames-Dryden and the Royal Aerospace Establishment on Knowledge Based Systems (KBS) has been established. This joint activity is concerned with tools and techniques for the implementation and validation of real-time KBS. This paper describes the proposed next stage of this research, in which some of the problems of implementing and validating a Knowledge-Based Autopilot (KBAP) for a generic high-performance aircraft will be investigated.

More recently, a collaborative project on Knowledge Based Systems (KBS) was undertaken³ to investigate some of the real-time aspects of applying such techniques. Under this programme, a prototype Flight Status Monitor for the X-29 research aircraft, which had been designed in a non-real-time form by NASA, was re-implemented at RAE in the Muse real-time KBS language^{4,5}. This gave a significant speed improvement over the original and, though it still fell some way short of full real-time operation, considerable insights were gained into the requirements which any real-time system would have to meet.

2. Introduction

The research programme described in this paper will be the latest in a longstanding series of collaborations between the Dryden Flight Research Facility of the NASA Ames Research Center in the USA and the Royal Aerospace Establishment (RAE) in the UK. Previously, the Cooperative Advanced Digital Research Experiment (CADRE) programme¹ was established in 1981 to investigate the applicability of non-linear control techniques to a fly-by-wire aircraft. Non-linear control laws developed at RAE were successfully flight tested on the NASA F8 digital fly-by-wire aircraft using the Remote Augmented Vehicle (RAV) facility², in which ground-based computers are used to control a piloted aircraft remotely via telemetry links.

The proposed latest programme will go further in the investigation of real-time KBS system design. Known informally as the Cooperative Real-time Intelligent Systems Program(me) (CRISP), it will investigate the implementation and validation of a Knowledge Based Autopilot (KBAP). While it is recognised that this problem is probably more appropriately tackled by conventional algorithmic techniques, the KBAP provides a simple, well-defined yet real problem within which to explore, develop, and demonstrate real-time KBS concepts and validation and verification techniques for mission-critical systems.

A prototype autopilot has already been implemented using the NASA-developed KBS tool CLIPS⁶. However, this implementation was shown to be fairly slow, and hence inappropriate for a real-time flight control application. The RAE has

¹ Royal Aerospace Establishment, Farnborough, Hants, GU14 6TD, U.K.
² NASA Ames Research Center, Dryden FRF, PO Box 273, Edwards, CA 93523-5000
³ PRC Systems Services, Edwards, CA.

This work has been carried out with the support of Procurement Executive, Ministry of Defence © Controller HMSO, London, 1991.

developed the Fortran Library for EXpert System Development, FLEX⁷, which has been demonstrated to be an order of magnitude faster than other KBS tools on benchmark problems⁸. This paper discusses the likely best method of approaching the reimplementing of the CLIPS autopilot rules in FLEX to produce a Knowledge Based Autopilot which runs in real-time. Some preliminary performance estimates are presented below, based on work done using a generic ruleset typical of the form likely to be taken by the final KBAP system.

One of the main areas of interest in this project is to establish a route by which a system based on KBS techniques could be validated as fit for flight. NASA Dryden have considerable experience in the validation of more conventional algorithmic avionic systems^{9,10,11} and are keen to investigate ways of extending these to cover the more diffuse behaviour exhibited by a Knowledge Based System. If the work is successful, it is hoped that the KBAP would be flight tested in future phases of the project.

3. An Overview of the KBS Toolkits

3.1 CLIPS Expert System Shell

CLIPS is an expert system shell with a LISP-like syntax developed at the NASA Johnson Space Flight Centre. The basic elements of CLIPS are; a fact-list, comprising global memory for data, a knowledge-base containing all of the rules and an inference engine which controls overall execution.

A programme written in CLIPS consists of rules and facts. The inference engine decides which rules should be executed. Basically, rules fire (are executed) in much the same way that an IF THEN statement is executed in a procedural language such as Ada. That is, the rule fires IF certain conditions are true, and it THEN executes a set of actions. The conditions that fire the rule are facts. The rule only fires if certain facts have been asserted (i.e the fact exists and is true).

CLIPS has variables available in which to store values. This is a very powerful tool which enhances

the way in which rules and facts can be manipulated. Variables in CLIPS are written in the syntax of a question mark followed by a variable name. For example:

```
?x
?value
?colour
?sound
```

One of the most useful, and most powerful applications of variables is in pattern matching. This is where the facts on the LHS of a rule are partially replaced by variables. For example, the rule:

```
(defrule variable-example
  (blue exterior)
  (?colour interior)
=>
  (assert (interior-colour ?colour)))
```

Will be fired by the "blue exterior", and by any fact which has two fields, with the word "interior" as its second field. It will then assert a fact that records the first field of the second fact. For example, the rule will be fired when the following facts are asserted:

```
(blue exterior)
(pink interior)
```

The rule will assert the fact (interior-colour pink).

Another useful feature of CLIPS is its ability to do calculations within rules. Expressions to be calculated must be written in prefix form, that is, the expression $2 + 3$ would be written $(+ 2 3)$. Again this is where CLIPS resembles LISP. This facility can be used to assert facts, for example:

```
(assert (answer =(+ 2 3)))
```

would assert the fact "answer 5". It is possible to use variables within expressions, for example:

```
(assert (answer =(- ?x ?y)))
```

This would calculate $?x - ?y$, using the actual values assigned to the variables $?x$ and $?y$, and then assert the fact to record the answer.

Control within a CLIPS program can be achieved by using facts as controls but CLIPS provides a more direct method of control through salience.

This allows assignment of priority to rules, to ensure that the highest priority rule in a set will fire first even if others are available to fire also.

CLIPS provides several commands to help in debugging. One command allows you to continuously watch facts being asserted and retracted. Whenever a fact is asserted or retracted, it will be displayed as such. Assertion of the fact (new_fact) would cause the display

```
=> f-1 (new_fact)
```

If this fact was then retracted, the display would be

```
<=> f-2 (new_fact).
```

It is also possible to watch rules and activations on the agenda as the program is executing.

3.2 Fortran Library for EXpert Systems, FLEX

Flex is a library of Fortran 77 subroutines for developing Expert System modules to interface directly with Fortran programs. It was developed at RAE⁷ and has already found application in the field of aircraft structural design¹².

As well as the subroutines, a separate readable knowledge base is supported, together with forward, backward and hypothesis / constraint inferencing. Basic explanation facilities are also provided. The FLEX library can be called from a higher-level Fortran program to implement the inferencing procedures required by the problem. A knowledge base in FLEX consists of a number of separate rule bases contained within the knowledge base.

3.2.1 Rule Format

Rules are represented in the following form:

```
Rule-identifier
IF property-a AND property-b.....
THEN property-x AND property-t.....
EXP=Explanation;
```

Disjunctive rules (i.e. rules whose antecedents contain facts linked by 'OR' functions) can also be used, in which case and 'AND' operator takes precedence over 'OR'. In other words,

'if A and B or C and D'

means if A and B are both true, or if C and D are both true.

The negative of a property is denoted by a 'not_' (optionally '-' directly before the name (e.g. -mammal means not a mammal).

3.2.2 Representation of Facts.

The facts which correspond to particular properties are stored in an array provided by the user. Each fact can be in one of three states: true, false or unknown. There is no fact list as such, rather, when the rule bases are read in by the driving Fortran program, the properties and their associated values are stored in the aforementioned array. This is unlike a system such as CLIPS, where facts are only stored in the fact list if they have been asserted. In FLEX, all the properties used in a rule base, as both antecedents or consequents, will be placed in the array when the rule base is read in. The fact value associated with each property can then be set explicitly. The setting of a fact value can be considered to be the same as asserting a fact in CLIPS, and similarly, setting a fact value to false can be considered the same as retracting a fact.

4. The Knowledge-Based Autopilot

To illustrate the proposed approach to the verification and validation of KBSs, a rule-based longitudinal altitude-command autopilot example for a high performance fighter aircraft has been designed. The example presented represents a single axis of a three-axis (longitudinal, lateral-directional, and velocity) controller. This controller is being developed and will be qualified as a mission-critical system as part of the research into validation methodologies for operation-critical KBSs.

A simplified representation of the aircraft and control system is shown in figure 1. The objective is to develop and to demonstrate a knowledge-based controller that produces command inputs to the aircraft control system based on a dynamic world model obtained from instruments on the aircraft and

91-15519


91 1113 014

on a simple set of rules. While this task may not represent a suitable end-application of a KBS (because it is easily performed by conventional algorithmic control laws), it provides a simple mission-critical application that is both easy to understand and easy to validate.

The control task requires the autopilot system (whether based on conventional algorithms or a knowledge-based approach) to produce commands that cause the measured aircraft altitude h to be within some specified tolerance Δh of the commanded altitude h_{com} . Additionally, constraints are placed on the altitude rate and the normal acceleration a_n . The constraint on a_n is the same as a constraint on altitude acceleration, but a_n represents a more easily understood and easily measured physical quantity.

The initial requirement for this controller was that it control the aircraft in a consistent, repeatable manner at least as well as a pilot during both the transition mode (going from one altitude to another) and the altitude-hold mode (controlling the aircraft about a specified altitude). The desire was to have it control the aircraft as well as a conventional algorithmic autopilot. An additional goal was to allow off-condition engagement so that the controller would also be effective even without benign initial (engagement) conditions.

These goals and requirements are similar to those initially imposed on the altitude-hold capabilities of the flight test maneuver autopilot for the HiMAT vehicle¹³. The constraints and tolerances were established as baseline figures. From this initial specification, a rule-based system was implemented that combined numeric and symbolic methods. This initial system was tested using a detailed nonlinear simulation model of the aircraft and its control system; the controller achieved excellent results for some initial conditions but performed poorly for many others. This initial result was typical of that experienced when evaluating the initial implementation of a conventional controller on a nonlinear simulation. After several iterations of this process, a fairly detailed statement of performance capabilities and limitations was established. This information, in essence, represents clarification of the statement

of goals and requirements and serves as the basis of a functional specification for the system. An example of a prototype set of rules for the longitudinal altitude-hold section of the rulebase is given in the following table.

TABLE Preliminary Rules for Longitudinal Altitude-Hold Autopilot

Performance boundary rules

- If the altitude acceleration exceeds the positive acceleration limit, move stick forward.
- If the altitude acceleration exceeds the negative acceleration limit, move stick aft.
- If the predicted altitude rate exceeds the positive altitude rate limit, trim stick forward.
- If the predicted altitude rate exceeds the negative altitude rate limit, trim stick aft.

Normal command rules

- If the altitude error is positive and the predicted altitude rate is negative, trim stick aft.
- If the altitude error is negative and the predicted altitude is positive, trim stick forward.
- If the predicted altitude error is positive and the altitude error is small, click stick forward.
- If the predicted altitude error is negative and the altitude error is small, click stick aft.
- If the predicted altitude error is positive and the altitude error is large, trim stick forward.
- If the predicted altitude error is negative and the altitude error is large, trim stick aft.

Definitions:

move	large movement of stick
trim	intermediate movement of stick
click	small movement of stick

4.1 Possible Implementation Using FLEX

The aim of this project is to re-implement an existing KBAP, supplied by NASA and written in CLIPS, in FLEX, the Fortran Library for Expert System Development. The reasoning behind this is that FLEX has proven to be much faster than CLIPS and other currently available expert systems

on benchmarking problems. It is hoped, therefore that a FLEX implementation of the KBAP can be made to run in real time.

The intention is to provide exactly the same functionality as the CLIPS version. Therefore the knowledge used in the CLIPS version has to be extracted and then rewritten in a form that could be used by FLEX. Thus, the FLEX implementation would not require any knowledge engineering in the form of consulting a human expert on autopilots, since this work has already been done by NASA.

The usual method used for translating rules from one expert system language to another is to first rewrite the rules in English, using a structured format of IF THEN constructs. Once this knowledge has been extracted and rewritten into a FLEX knowledge base, Fortran driving code will have to be written to utilize it. FLEX is not a self-contained expert system like CLIPS; rather it is a set of Fortran subroutines that utilize the decision making capabilities of expert systems. The difference between its capability and that of CLIPS will necessitate some changes to the division of labour between the algorithmic and knowledge based parts of the system. In particular, FLEX does not allow arithmetic expressions within rules, so all calculations will have to be done by the algorithmic part of the autopilot which will assert facts for consideration by the rule-base.

A preliminary investigation has been carried out on the conversion of a typical set of autopilot rules into FLEX. This has proved to give very encouraging results. The combination of the restructuring as above, and the greater efficiency of the FLEX inferencing procedure gave a speed improvement well in excess of a factor of ten over the CLIPS version, using a Sun 3 processor. This should provide ample scope for real-time operations, even if the final KBAP rules adopted prove to be more complex than this preliminary set.

Work on implementing Knowledge-Based Systems in conventional languages is continuing at RAE, and an Ada-based KBS toolkit is likely to be available within the timescales of this project¹⁴. It

would be a valuable extension of the CRISP work to investigate a re-implementation using Ada as a comparison.

5. Approaches to the Validation and Verification of the KBAP

The V&V methodology used at Ames-Dryden is the same methodology that has actually been used for all flight-critical control systems in non-commercial aeronautical flight vehicles, including the F-18, Space Shuttle, and B-1 aircraft. This methodology uses a subset of the V&V techniques in use or advocated within the aeronautics community. The larger issues of certification and the validation of highly reliable, fault-tolerant systems have been of lesser concern than those of qualifying and conducting flight validation of flight-critical systems.

The basic methodology for the V&V of conventional operation-critical systems is directly applicable to the V&V of KBSs. In fact, if KBSs are to be used in operation-critical applications, the qualification of these KBSs will have to be performed within the context of established procedures and will have to address the requirements placed upon the qualification of conventional operation-critical systems.

The basis of the Ames-Dryden flight qualification and V&V methodology for embedded flight-critical systems is the incremental verification of systems components, integration testing, configuration management, and flight validation. The design specifications are transformed into hardware and software realizations. This transformation is not a straightforward, one-step process. The transformation of a design specification to an implemented prototype system requires the development and testing of numerous software procedures and hardware circuits, each of which is a prototype of some element in the larger system.

The implementation of system elements or components is supported by a variety of analysis tools and testing techniques^{9,10}. The analysis tools used include failure modes and effects analysis, independent review, static verification, independent calcula-

tions, conjectures, and suspicions. This analysis is conducted on abstract models of the system or of the system components. Linear systems models, aggregate system models, block diagrams, flow diagrams, schematics, source programs, specifications, and simulations are some of the main abstract models used. This analysis of abstract models is used to translate requirement and design specifications into a physical realization.

Simulation testing provides a closed-loop facility wherein the system is exposed to an environment that closely resembles the electronic and data environment in which the system must actually operate. Simulation also provides a facility for testing that the hardware and software of the system are integrated and operating together. Simulation is where the pilot (the system user) is first exposed to the system and allowed to evaluate it; the realism of the simulation is determined by the operating requirements for the flight application of the system.

The V&V methodology used for conventional, embedded operation-critical flight systems provides an established and accepted set of procedures upon which a methodology for KBSs can be based. While this position may be controversial in the AI community, the political and sociological realities of flight research and testing will ultimately dictate that any methodology for the validation of KBSs at least address the currently used methodology for conventional systems.

The proposed approach to the V&V of KBSs relies on the life cycle model shown in figure 2. The life cycle model for a KBS has been a topic of considerable concern to some who have addressed the validation of a KBS, and several models have been proposed^{15,16,17}. These models stress the development and prototyping process in a KBS. The motivation for developing these models is apparently to address the lack of a clear or well-defined statement of system goals and requirements and to highlight the prototyping process common in the development of KBSs. While the proponents of these models would probably contend that there is a fundamental difference between the life cycle of a KBS and a conventional system, another view is

that this apparent difference is more reflective of the maturity of KBSs rather than of anything fundamental.

Because KBSs are just emerging in operation-critical applications, there is little certainty of capabilities and limitations of these systems. The prototyping that is a common feature in the development of a KBS often represents an attempt to establish requirements for a given application. This definition of requirements, capabilities, and limitations through prototyping is not unlike that used in conventional systems when new techniques or application are attempted. The difference is in the body of knowledge and experience behind the use of conventional systems as opposed to that of KBSs. Also reflected in this prototyping is the lack of maturity of artificial intelligence (AI) techniques in general that provides little basis for the selection of control and knowledge representation methods.

There are several issues that are almost certain to create problems for anyone attempting to validate operation-critical KBSs. Perhaps the most serious of these is an unwillingness to treat the current generation of KBSs out of the context of the promises of AI. The current generation of KBSs are not, in general, capable of learning or even modestly adaptive. These systems exhibit few nondeterministic properties. These KBSs may be complex but they are not unpredictable. But so long as there is this persistence in dwelling on the ultimate potential of AI systems instead of on the realities of the system being qualified, it is unlikely that an airworthiness and flight safety review (AFSR) panel would allow flight testing.

A further difficulty arises from the contention the KBSs do not always produce the correct answer. If this is true then a KBS can only be used for tasks in which their performance can be monitored and overridden by a human. Most operation-critical systems are required to perform without human intervention or with only high-level supervision or control. However, a KBS that does not always produce the optimum answer is acceptable as long as it never produces a wrong answer. This latter point is in fact one of the main V&V issues: operation-critical systems must be shown to produce accept-

able solutions in all situations.

There are two key aspects of the proposed approach to the V&V of KBSs:

1. development of a KBS to perform some task that is well-known, well-understood, and for which conventional V&V techniques are adequate; and
2. incrementally and simultaneously expand both the KBS and the V&V techniques to more demanding and complex tasks.

The procedures used for verifying, qualifying, and validating conventional operation-critical flight systems at Ames-Dryden will be applied and modified as required. Because we ultimately plan to carry these experiments to flight using the rapid-prototyping facility², this process will be performed under the aegis of an AFSR panel and will be under periodic review. The subject of this research will be the knowledge-based autopilot (KBAP) described above, that is being developed ultimately to perform aircraft maneuvers normally performed by highly trained pilots.

The research plan is to identify maneuvers of increasing difficulty and to build gradually more complex and adaptive KBS to accomplish those maneuvers. This will include prototyping, evaluation, and a series of initial operating capabilities that will evolve into a sequence of documented requirements for testing against each version of the system. This approach fits well within the model of and practice used with conventional digital systems.

6. Conclusions

This paper has described a proposed collaborative programme of research intended to approach the problem of the validation and verification of mission-critical knowledge-based systems in an incremental manner, using established procedures. The view presented in this paper is consistent with that proposed in Gault⁹ and others:

A validation methodology for ultrahigh reliability, fault-tolerant systems must be based on *a judicious combination of logical proofs, analytical modeling, and experimental testing.*

This methodology must be supported by reliable validated development and test tools that lower the cost and reduce the schedule, if the goal of validation is to be achieved for either highly reliable, fault-tolerant systems or highly complex systems such as are envisioned for KBSs.

The work will focus on the real-time implementation of a knowledge-based autopilot designed by NASA using a real-time KBS tool, FLEX, written at RAE. Preliminary results on a representative rule-set indicate that FLEX will have more than sufficient speed for this application. The possibility also exists of an Ada implementation at a later stage.

The specific structures used within the real-time version may well influence some aspects of the approach taken towards validation, in particular the position of the boundary between the algorithmic and rule-based sections of the autopilot. It is hoped that, if the validation and verification work is successful, then flight tests, using the NASA Ames-Dryden Rapid Prototyping Facility, could be undertaken.

Acknowledgement

The help of Mrs. M. A. Kirby in preparing this document is gratefully acknowledged.

References

1. Butler, G. F., Corbin, M. J., Mephram, S., Stewart, J. F. & Larson, R. R. "NASA/RAE Collaboration on Nonlinear Control Using the F-8C Digital Fly-by-Wire Aircraft", 35th AGARD Guidance and Control Panel Symposium, Lisbon, October 1982, Paper 21.
2. Duke, E. L., Brumbaugh, R. W., & Disbrow, J. D. "A Rapid Prototyping Facility for Flight Research in Advance Systems Concepts", Computer, May 1989, pp 61-66.
3. Butler, G. F. & Duke, E. L. "NASA/RAE Cooperation on a Knowledge Based Flight Status Monitor", AGARD 48th Guidance & Control Panel Symposium, Lisbon, May 1989.

- Paper 34.
4. Reynolds, D. "MUSE: a toolkit for embedded real-time AI", in *"Blackboard Systems"*, ed. R. Englemore & T. Morgan, Addison Wesley, 1988.
 5. Miles, J. A. H., Daniel, J. W. & Mulvaney, D. J. "Real-Time Performance Comparison of a Knowledge-Based Data Fusion System using Muse, Art and Ada", *Artificial Intelligence and Defence, AI'89, Avignon, May 1989*, pp 247-259.
 6. Giarratano, J. C. "CLIPS User's Guide", *Artificial Intelligence Section, Lyndon B. Johnson Space Center, June 1988*.
 7. Butler, G. F. & Corbin, M. J. "FLEX: Fortran Library for Expert Systems", *RAE Working Paper MM 273/88, December 1988*.
 8. Butler, G. F. & Corbin, M. J. "Benchmarks for Knowledge Based Systems", Paper given at U.K.I.T. 88, Alvey, Swansea, July 1988, pp 61-64.
 9. Szalai, K. J., Jarvis, C. R., Krier, G. E., Megna, V. A. Brock, L. D., and O'Donnell, R. N. "Digital Fly-by-Wire Flight Control Validation Experience", *NASA TM-72860, 1978*.
 10. Gault, J. W., Trivedi, K. S. & Clary, J. B., Eds. "Validation Methods Research for Fault-Tolerant Avionics and Control Systems - Working Group Meeting II", *NASA CP-2130, 1980*.
 11. Duke, E. L. "V&V of Flight and Mission-Critical Software", *IEEE Software, May 1989*, pp 39-45.
 12. Harris, J. ap C., Bartholomew, P., Butler, G. F. & Corbin, M. J. "An Application of Expert System Techniques to Structural Optimisation in a Fortran Environment", *OPTI '89, Southampton, June 1989*.
 13. Duke, E. L., Jones, F. P. & Roncoli, R. B. "Development and Flight Test of an Experimental Maneuver Autopilot for a Highly Maneuverable Aircraft", *NASA TP-2618, 1986*.
 14. Corbin, M. J. & Butler G. F. "A Framework for Object-Oriented Programming in Ada", *Ada User, Vol 11, No.3, pp 133-144, July 1990*.
 15. Culbert, C., R. & G. and Savely, R. T. "Approaches to the Verification of Rule-Based Expert Systems", *SOAR '87, First Annual Workshop on Space Operations Automation and Robotics, Aug 1987*.
 16. Richardson, K. & W. and Wong, C. "Knowledge Based System Verification and Validation as Related to Automation of Space Station Subsystems: Rationale for a Knowledge Based System Lifecycle" in *"Second Annual AI Research Forum", NASA Ames, Nov 1987, pp 153-158*.
 17. Stachowitz, R. A., Combs, J. B. and Chang, C. L. "Validation of Knowledge-Based Systems", *AIAA-87-1685, March 1987*.

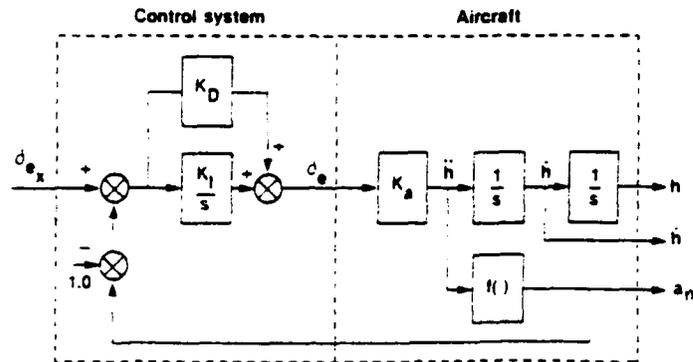


Figure 1. Simplified Longitudinal Model of the Aircraft and Its Control System

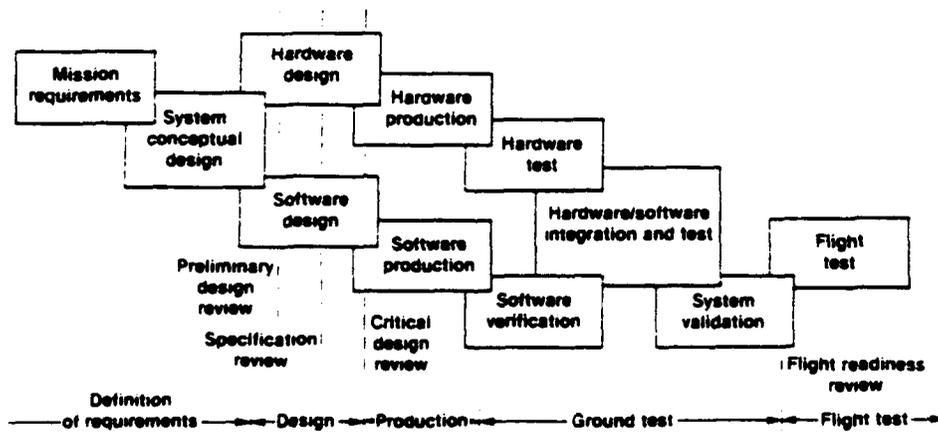


Figure 2. Ames-Dryden Life Cycle for Research Systems

Discussion

1. W.G. Semple, United Kingdom

I understand that you approach the clearance problem by constraining the rule base to be such that all possible states can be identified and assessed. Does that reduce the verification problem to that of conventional code, thus ducking the stated objective?

Author:

If I understood the question, the answer is no, that's just step 1. The project will proceed to bigger rule bases. Yes, we hope in future versions to have a more complex rule set with different autopilot modes and more likelihood of interaction between rules.

2. Carl Lizza, United States

If you have a small, well-defined, simple ruleset to facilitate verification then why use a rule-based paradigm at all? Why not code the rules as procedural logic?

Author:

Coding the system in procedural logic would negate our

effort to explore the verification and validation issues for a knowledge based system.

3. H.A.T. Timmers, Netherlands

Your objective was to develop a knowledge based autopilot which could be certified by the applicable agencies. How did this objective reflect in the actual design?

Author:

In the first version of the knowledge base we have been constrained to a design with relatively few rules which can be shown not to be in conflict with each other.

4. R. Guiot, France

Why not use fuzzy logic control?

Author:

I would be very interested to look into the possibility of using fuzzy logic. I should reiterate that the main purpose of the work is not to produce the best possible autopilot, but to investigate verification and validation issues relating to mission control KBS systems. It will be interesting to see whether fuzzy logic has advantages for verification and validation.



COMPONENT PART NOTICE

THIS PAPER IS A COMPONENT PART OF THE FOLLOWING COMPILATION REPORT:

(TITLE): Conference Proceedings of Machine Intelligence
for Aerospace Electronic Systems Held in Lisbon, Portugal on
13-16 May 1991 (L'Intelligence Artificielle dans les Systemes
Electroniques Aeroespaciaux)

(SOURCE): Advisory Group for
Aeronautical Research and Development, Neuilly-sur-Seine
(France)

TO ORDER THE COMPLETE COMPILATION REPORT USE AD-N242025

THE COMPONENT PART IS PROVIDED HERE TO ALLOW USERS ACCESS TO INDIVIDUALLY AUTHORED SECTIONS OF PROCEEDINGS, ANNALS, SYMPOSIA, ETC. HOWEVER, THE COMPONENT SHOULD BE CONSIDERED WITHIN THE CONTEXT OF THE OVERALL COMPILATION REPORT AND NOT AS A STAND-ALONE TECHNICAL REPORT.

THE FOLLOWING COMPONENT PART NUMBERS COMPRISE THE COMPILATION REPORT:

AD#: _____

TITLE: _____

AD-P006 338

DTIC
ELECTE
NOV 18 1991
S D

AD-N242025	
DTIC	J
AD	
AD	
AD	
AD	
By _____	
Date: _____	
Availability Codes	
Dist	Avail and for Special
A1	21

This document has been approved
for public release and sale. Its
distribution is unlimited.

AD-P006 338



Sandra L. Berning
Wright Laboratory
WL/AAAN-2
WPAFB, OH 45433
USA

Douglas P. Glasson
TASC
Reading, MA 01867
USA

ADAPTIVE TACTICAL NAVIGATION PROGRAM

ABSTRACT

The Avionics Directorate of the Wright Laboratory at Wright-Patterson Air Force Base sponsored development of the Adaptive Tactical Navigation (ATN) system from 1986-1990. The ATN system, developed by TASC under contract to the United States Air Force, is a laboratory prototype which incorporates knowledge-based software designed to perform navigation system management and decision-aiding for the next generation of combat aircraft. The purpose of the ATN system is to manage a future multisensor navigation suite, dynamically selecting the most appropriate navigation equipment to use in accordance with mission goals, mission phase, threat environment, equipment health, equipment availability, and battle damage. The ATN system encompasses functions as diverse as sensor data interpretation, diagnosis, and navigation resource planning.

This paper describes the motivation for pursuing this avenue of research, the ATN design and development processes, results obtained, and lessons learned.

BACKGROUND

To succeed in the complex mission environment of the mid-1990s, combat aircraft will require advanced capabilities. Among these capabilities are: robustness to electronic countermeasures, automatic terrain following/terrain avoidance, effective in-weather operations, and the potential for covert operations through strict emissions management. Also, air defense system deployments to defend high-value fixed targets will require survivability tactics such as standoff/blind or maneuvering weapon deliveries. Realizing all of these advanced mission capabilities requires highly accurate, robust, and reliable navigation system performance.

To meet the full range of mission requirements, complementary high accuracy navigation sensors are being integrated onboard advanced combat aircraft. Among these advanced

systems are the Global Positioning System (GPS), Joint Tactical Information Distribution System (JTIDS), and terrain reference navigation (e.g., Sandia Inertial Terrain Aided Navigation). Advanced techniques are required to take full advantage of all navigation resources while minimizing demands on the pilot/aircrew for system interpretation and management.

INTRODUCTION

The objectives of the Adaptive Tactical Navigation program were to determine the utility of an intelligent system manager controlling the navigation system of a mid-1990's combat aircraft and the suitability of current technology for implementing such a design. The technical effort entailed the design, development, and demonstration of a system which performs system management and decision aiding functions for the projected multisensor navigation suite of mid-1990's attack aircraft. The goal for the ATN system was to provide a high quality navigation solution while minimizing interpretation and interaction ("switchology") demands on the pilot/aircrew.

The ATN system was developed and evaluated in a laboratory environment followed by rehosting and evaluation in an F-16C cockpit simulation dome. Evaluation results indicate operational deployment of a system that utilizes the technology developed under the ATN program could result in enhanced use of navigation resources, reduced pilot workload, improved situational awareness, and intelligent navigation system failure detection and isolation. Collectively, these benefits result in an overall improvement in mission effectiveness.

ATN SYSTEM OVERVIEW

The ATN system consists of two major components: the Basic Navigation System which integrates the sensor outputs into navigation solutions and the Expert Navigation System, which provides system management and decision-aiding functionality.

A set of sensors representative of the technology anticipated to be operational by the mid-1990s was selected as the ATN sensor suite; viz., a Strapdown Inertial Navigation System (INS), GPS, Sandia Inertial Terrain Aided Navigation (SITAN), Synthetic Aperture Radar (SAR), Doppler Radar, and an Electro-Optical (EO) System. The ATN system was designed to capitalize on the strengths of this sensor suite, minimize the workload demands of managing such a complex sensor suite, and compensate for all inherent constraints imposed when using a particular sensor.

ATN SYSTEM DESIGN

Significant issues addressed in the ATN design effort included: Domain of Functionality, Architectures, and Knowledge Acquisition. Key results of ATN System design efforts are summarized below.

ATN Domain of Functionality — System management and advisory functions that the ATN system could and should perform were determined. To select these functions, assessments were performed of both the feasibility of implementing functions and the projected improvement in mission effectiveness and reduction in crew workload. Three broad classes of functions were selected for the ATN effort; viz., Sensor Integration and Navigation Computations, System Management, and Decision Aiding.

The first class of functions (Sensor Integration and Navigation Computations) is performed by a portion of the ATN system termed the Basic Navigation System (BNS). The BNS integrates data from a variety of available navigation sensors to form a usable vehicle navigation state output. This is accomplished by using Kalman filters to estimate time-correlated navigation state parameters and sensor model parameters by optimally combining sensor measurement data. A related, secondary function of the BNS is to provide sensor-related navigation data to the intelligent portions of the ATN system to aid in the recognition of special or problem situations and to support the reconfiguration or adjustment of the BNS to best meet user requirements.

The second and third classes of functions (System Management and Decision Aiding) are performed by the remaining portions of the ATN system which collectively are referred to as the Expert Navigation System (ENS). Table 1 lists the primary functions performed by the ENS. Func-

tions 1 through 10 are system management functions aimed at monitoring and diagnosing sensor/basic navigation behavior and supporting equipment reconfiguration (moding transitions). Functions 11 through 16 in Table 1 support user decision aiding in selecting navigation sensors/configurations appropriate to the current and predicted mission requirements and resolving user-observed navigation anomalies.

Table 1 Expert Navigator Functional Requirements Cross Reference

ATN FUNCTION	ACTIVATED EXPERTS					
	ENVIRONMENTAL MANAGER	MISSION MANAGER	EVENT DIAGNOSIS	SYSTEM STATUS	BOOKING	NAVIGATION SOURCE MANAGER
1 Identify sensor failure/reconfiguration						
2 Detect degraded/improper Kalman filter performance						
3 Select display data appropriate to situation, status of other mission subsystems, NAV system status, crew request						
4 Use redundant good information						
5 Learn detection of NAV system failure, recover and request alternative NAV system configuration appropriate to mission requirements and environment						
6 Manage total processing resources to best advantage						
7 Evaluate NAV system confidence as a function of monitored performance, mission environment, past inputs (not sensed and diagnosed)						
8 Generate reconfiguration options to correct and recover						
9 Generate reconfiguration/reallocation options appropriate to chosen mode						
10 Compute and store (pending) reliability model results and related data/ information (e.g., component failure rate data)						
11 Monitor mission script and environment models including updates from other expert systems (FA) and intelligence gathered insight						
12 Confirm or reevaluate NAV events based on current mission script/environment						
13 Predict mission effectiveness						
14 Evaluate/repair scheduled NAV events						
15 Diagnose anomalous NAV performance events in response to pilot concerns						
16 Resolve inconclusive diagnoses from source management and system levels						

LEGEND
 ✓ Primary Contributor
 • Support Function
 * Preparing/Configuring for ongoing or imminent mission events

ATN Architecture — The BNS architecture (Figure 1) consists of a baro-inertial mechanization and four federated Kalman filters. A federated Kalman filter approach was selected for this application because of its superior ability to recover an uncorrupted navigation solution once a failure is identified. Switching navigation outputs from one filter to another, as is done in the BNS, has disadvantages for aircraft systems that depend on continuous navigation data. From this standpoint, a centralized Kalman filter approach would have an advantage. However, the federated approach offers the ability to recover quickly from soft failures and achieves better data protection with little sacrifice in performance (Ref 1).

G-66388A
6-22-98

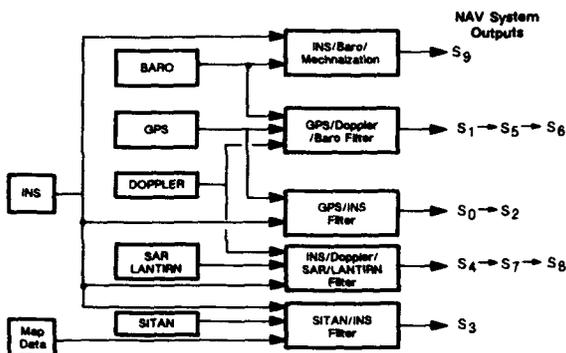


Figure 1 Basic Navigation

Table 2 Navigation System Combinations

SYMBOL	COMBINATION
S0	INS* and GPS4
S1	GPS4
S2	INS and GPS3
S3	INS and TAN
S4	INS, DPR, SAR, and EO
S5	GPS3, DPR, and BARO
S6	GPS3 and BARO
S7	INS and DPR
S8	INS, SAR, and EO
S9	INS

*INS implies INS and BARO in all cases in this table

To establish workable BNS architectures, it was necessary to identify sensible combinations of sensors. Ten alternative navigation system configurations of subsets of the ATN sensor suite were identified. Systems were ranked (Table 2) according to order of relative accuracy. The ENS chooses the most desirable system alternative based on expected accuracy, confidence, and mission segment requirements.

The ENS is a distributed expert system that embodies a broad range of functionality. Partitioning of functions among the experts was designed to localize specific sub-domains of navigation expertise such as mission management, equipment diagnosis, and sensor cross-checking. Global- and module-level architectures were designed for the ENS to realize the desired "intelligent" functionality. The functional organization of the ENS is depicted in Figure 2. The functionality of each expert systems is described below:

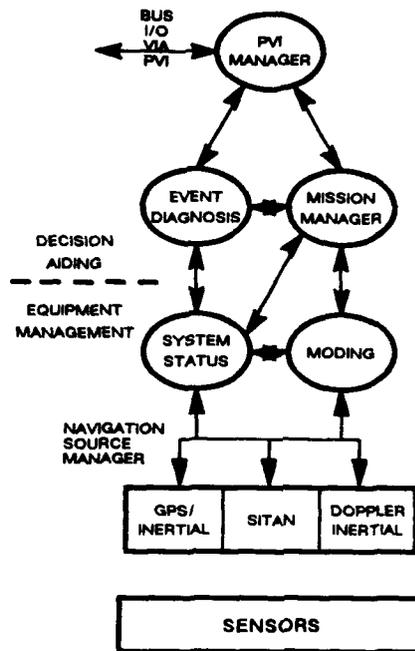


Figure 2 Expert Navigator Functional Organization

- *Navigation Source Manager* — This expert is shown for GPS/INS, SITAN, and Doppler-Inertial. It uses Kalman Filter outputs and design engineering models to monitor equipment performance and to detect and isolate sensor failures or degradations.
- *System Status* — This expert diagnoses system "health" based on reliability data, mission environment, and lower-level diagnoses.
- *Moding Expert* — This expert determines viable component combinations based upon current equipment status and determines appropriate handoff strategies for mode changes.
- *Event Diagnosis* — This expert evaluates planned navigation events (e.g., a waypoint encounter and designation) and diagnoses anomalous or out-of-spec events to support pilot moding decisions.
- *Mission Manager* — This expert stores mission plan and environment (threats, ECM) data and determines which available equipment configurations are appropriate to the current and forecasted mission situation.
- *Pilot-Vehicle Interface (PVI) Manager* — This expert manages communication between the ATN system and the pilot.

Knowledge Acquisition — Knowledge Engineering for the ATN program included both

91-15520
■■■■■■■■■■

91 1113 015

Knowledge Acquisition and Knowledge Base Implementation. Knowledge Acquisition was performed during the design phase of the program. Several different forms of knowledge were acquired including: existing information from engineering reports, results of numerical simulation (e.g., of GPS performance degradation in jamming scenarios), and human knowledge related to navigation.

Human knowledge covered a broad range of expertise in navigation system design, maintenance, and operations. Sources of human knowledge included: operational aircrews, test pilots and engineers, avionics maintenance technicians, navigation system and design engineers, and avionics system engineers. Obtaining knowledge from human sources posed several challenges not usually encountered in "classical" knowledge acquisition. For example, in 1987, when the ATN Knowledge Acquisition was performed, fielded systems had simple navigation suites (e.g., INS with visual or ground-map radar updating) in comparison with the suite envisioned for mid-1990s combat aircraft. Thus, no operational or test aircrews had experience with the multisensor navigation suite selected for ATN. In addition, the interview setting did not replicate the actual onboard environment in which decisions are made. These considerations strongly influenced the interview approach, the information that was obtained, and the selection of knowledge engineering techniques.

Approximately fifty individuals, each an expert in a subset of the domain, were interviewed as part of the ATN Knowledge Acquisition effort. Specific sources of human knowledge and the types of information obtained from those sources were:

- *Tactical and Strategic Aircrews* — Discussions were held with a range of tactical and strategic aircrews, representing varying experience levels and aircraft avionics vintage. Information obtained provided operator perspective of mission priorities, current navigation system management methodologies, sensor cross-checking, display content/format (both actual and desired), workload profiles as a function of mission phase and sensor utilization in a single seat combat aircraft, and workload split between a two-man tactical aircrew. Information was also obtained regarding the operational use of radio navigation aids (LO-RAN), an EO targeting/update system (PAVE TACK), a recently modernized navigation suite that utilizes a Kalman Filter to incorporate Doppler, true air speed, and/or discrete

ground map radar updates, and GPS (on a B-52).

- *Flight Test Groups* — Engineers and flight test pilots associated with GPS, LANTIRN, SITAN, and a digital moving map evaluation provided information related to engineering development and limited operational experience with these advanced systems.
- *Maintenance Personnel* — TASC reliability engineers and Air Force maintenance technicians provided insights into classes of observed behavior of equipment in the field and experience-based diagnosis techniques.
- *Design and System Engineers* — TASC, General Dynamics, and Air Force Navigation analysts and designers actively designed model-based components of ATN and served as consultants on such issues as GPS receiver performance, operational limitations of equipment, current design practice and related limitations, and reliability engineering.

An organized pre-brief and mission-centered questionnaire supported efficient aircrew interviews. This questionnaire was provided in advance to the participating Air Force operational organizations as a preliminary indication of our purpose and for classification guidance. The crew interviews and discussions were centered on a sample tactical interdiction mission. The phases and profiles of this mission are shown in Figure 3. Also, an overview of the navigation suite assumed for the mid-1990s under the ATN program was presented in the pre-brief.

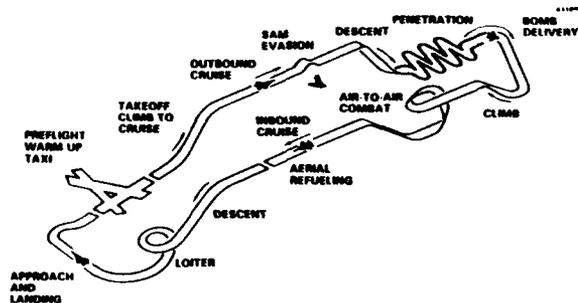


Figure 3 Tactical Mission Profile

ATN SYSTEM DEVELOPMENT AND INTEGRATION

The ATN System was developed using a traditional waterfall approach to software development in which system requirements and design were established before system development/

coding began. This phase of the ATN effort included development of the BNS and ENS architectures and the knowledge based components defined during ATN system design while meeting the requirement for an ATN system which could demonstrate real-time performance and be suitable for subject- interactive testing and evaluation. Development and integration of the ATN System was accomplished in stages:

Development of the Environment Simulation.

Development of the BNS and integration with the environment simulation.

Development of required displays.

Development of knowledge based components.

Development and component testing of the Expert Systems.

Integration and stand alone testing of the Expert Systems.

Integration of the BNS and the ENS and functional testing in the laboratory.

Rehosting and system evaluation in a Cockpit environment.

An overview of each development stage is provided below.

Environment Simulation Development — A complete and suitable Environment and Sensor Simulation (E/SS) was a necessary precursor to development of the BNS. Existing E/SS software (Integrated Navigation System Simulation or INSS) was restructured and augmented to meet this need (Ref 3). A strapdown inertial navigation system model was developed and integrated into the INSS software. In addition, SAR and EO sensor models were integrated into the existing INSS structure and common blocks for communication among the models were added.

BNS Development — Upon completion of the E/SS software, the BNS was developed and integrated with the E/SS. The BNS (Figure 1) consists of a set of navigation filters managed by an executive, a performance monitor to track results, and all necessary interfaces. Except for the performance monitor portion, the BNS operates strictly from sensor outputs provided by the E/SS. None of the "truth data" available from the environment simulation is available to the BNS except for performance assessments.

Displays Development — The head-up, head-down display complement for the engineer-

ing system is shown in Figure 4. The displays were implemented on two Zenith Z-248 PCs; both display computers were driven by data from the BNS and ENS via host processor (9600 Baud) terminal ports. User interaction was implemented via a mouse and keyboard.

The head-up display provides a moving horizon (slaved to the trajectory data from the E/SS) and a HUD layout based on that of the Block 25 F-16C. The HUD display includes two windows for ATN displays; a master caution that displays yellow on warning and red on caution and an ATN icon window for posting graphic symbols for ATN advisories.

The head-down display consists of a data entry display, two multifunction displays and a color moving map. The right multifunction display (MFD) provides a simulated radar display with cursors that align on fixpoints as updates are selected. The most recent aimpoint offset is displayed in the data entry display above this MFD. The left MFD provides a help facility for the graphics icons displayed on the HUD, including the meaning of the icon and the acceptable responses.

The 512 x 512 pixel moving map is implemented in a track-up, decentered orientation. Each pixel scales to 100 meters of actual distance; the map therefore presents 51 km of flight track, or approximately 3 minutes of flight time (this scale approximates a standard 1:250,000 sectional map scale). The data for the map was obtained by digitizing a sectional map of the region in 13 track-up segments. The mission route was overlaid on the original paper map before scanning.

Knowledge Base Implementation — Knowledge Base Implementation for the ATN effort required both selection of appropriate knowledge representation schemes and development of all knowledge based components.

The knowledge representation schemes were selected for each expert system based upon the type of reasoning to be performed within that expert. Representations selected for ATN include:

- Rules
- Procedures
- Relational Data Structures (e.g., frames)
- Symbolic Data
- Heuristically assigned probabilities or weightings.

Because of the dichotomy between the complexity of the ATN sensor suite and the simplicity of fielded systems at the time of ATN knowledge acquisition, the ATN system designers used non-traditional techniques to develop the knowledge

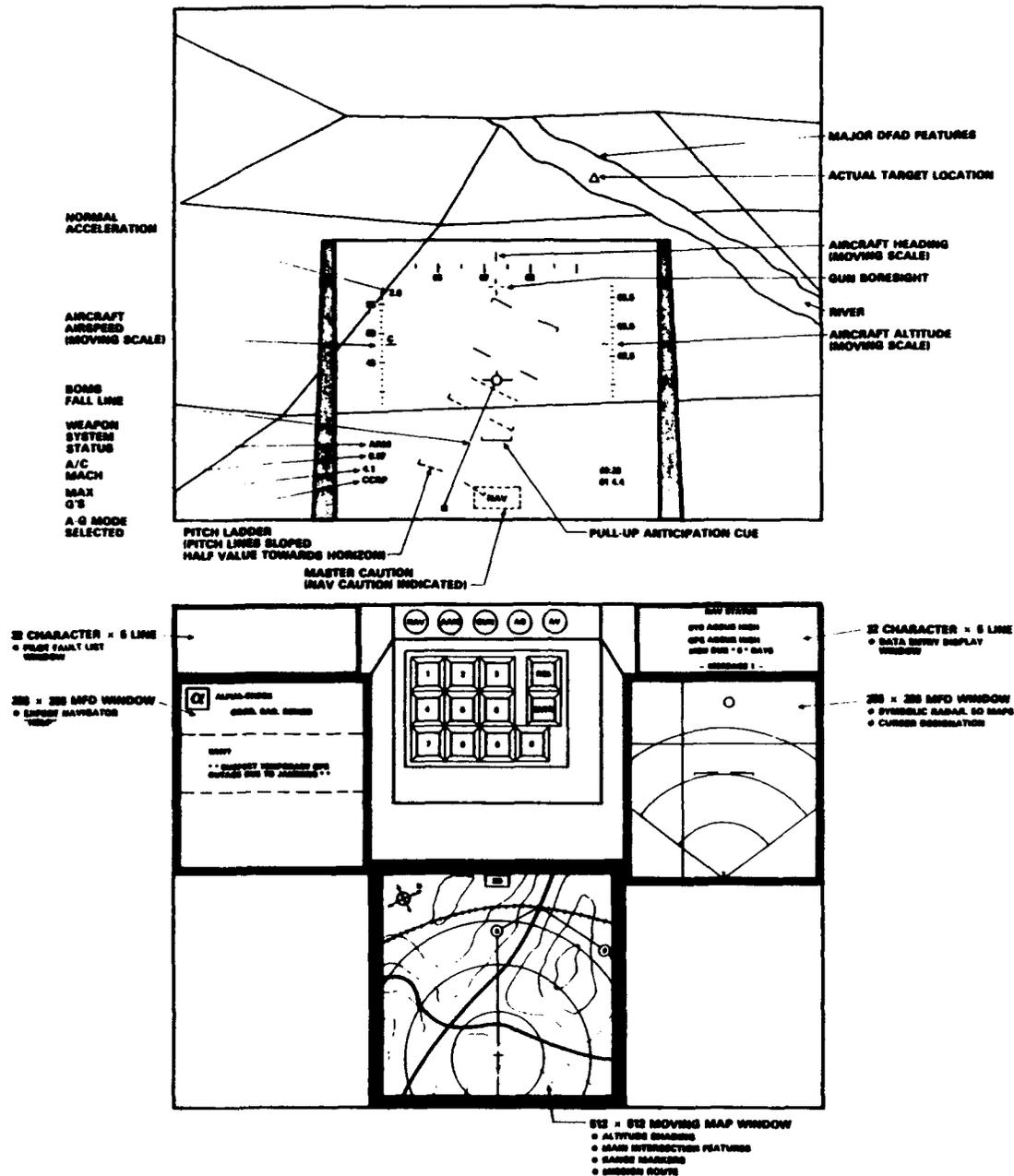


Figure 4 ATN Laboratory System Display

required for the ATN knowledge based components. Techniques included analogies to currently operational systems (e.g., GPS versus LORAN; LANTIRN versus PAVE TACK), and extrapolation to the expected ATN system complexity and mission environment. Thus, a considerable amount of the knowledge for the ENS expert systems was engineered by combining results of operational crew discussions with designer know-

ledge of projected system characteristics and future mission requirements.

Details regarding the knowledge based components for each expert system are summarized below:

Navigation Source Manager — The Navigation Source Manager embodies two forms of knowledge:

- **Knowledge Source Procedures** — These procedures provide a means of consistency checking among GPS, SITAN, and Doppler sensors. This was based upon cross-check procedures described during discussions with Ohio Air National Guard A-7D pilots.
- **Resource Allocation Logic** — The resource allocation logic for this expert system was developed by design engineering methods to lead the search through possible multiple failure combinations (i.e., choices of alternative parity functions).
- **System Status Module** — This is a rule-based system in which rules are organized in "chunks" that correspond to specific subsystems (e.g., INS, GPS receiver, CADC, Radar). Generic forms for the System Status rules were developed as a result of discussions with FB-111 maintenance technicians at Pease AFB, NH. These generic forms fall into three categories (Table 3):

Table 3 System Status Rules

- **A-Priori Evaluation**

Rules for Reliability Factor from Maintenance Information

FAILURE PROGNOSIS	CND FAILURES		
	FREQUENT	OCCASIONAL	RARE
Likely	0.80	0.85	0.95
Nominal	0.85	0.95	1.00
Unlikeiy	0.95	1.00	1.00
- **Quick Diagnosis**

Rules for Environment Quality From ECM and Weather Indices

WEATHER	ECM		
	HIGH	MEDIUM	LOW
Adverse	Adverse	Adverse	Moderate
Fair	Adverse	Moderate	Favorable

Rules for Environment Quality From ECM and Weather Indices

SENSOR STATUS LOG	ENVIRONMENT QUALITY		
	ADVERSE	MODERATE	FAVORABLE
Failed	0.00	0.00	0.00
Suspect	0.60	0.70	0.80
Normal	0.85	0.95	1.00
- **BIT Failure Evaluation**

Rules for Validity of BIT Failure Patterns

```

"IF (BIT_FAILURE_PATTERN = ALWAYS_HAPPENS_PATTERNk)
  THEN VALIDITY = 0.5
ELSE IF (BIT_FAILURE_PATTERN = KNOWN_FAILURE_PATTERNk)
  THEN VALIDITY = 1.0
ELSE VALIDITY = 0.9"
    
```

A Priori Evaluation (modified mission reliability).

Quick Diagnosis (environmental quality effects)

BIT Screening.

Moding Expert — The Moding Expert utilizes symbolic descriptions of sensors and modes and categorical symbolic data that specifies the mission-related attributes of these modes. This symbolic data is manipulated by procedures to determine viable navigation modes based on currently available equipment. The symbolic representations of the modes were derived from the BNS design. Sensor symbolic descriptions and mode attributes were compiled by navigation system design engineers. Examples of the mode representation and related property list format are provided in Table 4 and Figure 5, respectively.

Table 4 Mode Representations

STANDARD MODE HIERARCHY (STATIC)		BASELINE NON-STANDARD CANDIDATES	
S ₀	INU + GPS4	N ₂	INU + GPS3
S ₁	GPS4	N ₂₁	+ RADAR ALTIMETER
S ₂	INU + GPS3	N ₂₂	+ SAR/LANTIRN
S ₃	INU + STN		OR
S ₄	INU + DPR + S/L	N ₃	INU + STN
S ₅	GPS3 + DPR + BAR		
S ₆	GPS3 + BAR	N ₃₁	+ GPS2
S ₇	INU + DPR	N ₃₂	+ DOPPLER + SAR/LANTIRN
S ₈	INU + S/L	N ₃₃	+ DOPPLER
S ₉	INU		OR

LIST ENTRY REPRESENTATION

((<MODE CODE>, <POINTER TO PLIST>, < POINTER TO DESCRIPTIONS>))

EXAMPLE

((S0, S0_PLIST, S0_DESCRIPTION))

Mission Manager — Three knowledge-based elements were developed for the Mission Manager; these are:

Weightings for a mission effectiveness payoff function.

A decision model for pre-attack check of the navigation/bombing system.

Threat models.

Weightings for the mission effectiveness function were based on designer judgment derived from discussions with F-16C, F-4E, and FB-111 aircrews. A set of tentative premises was generated for the relative importance of the navigation mode attributes (see Table 5) as a function of mission phase; these are listed in Table 6.

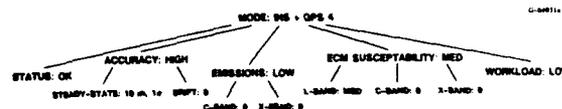


Figure 5 Example Property List

Table 5 ATN Navigation Mode Attributes

MODE	DESCRIPTION	ACCURACY	ECM SUSCEPTIBILITY	EMISSIONS	PILOT WORKLOAD
S ₀	INU + GPS 4	H	M (L-BAND)	L	L
S ₁	GPS 4	H	M(L-BAND)	L	L
S ₂	INU + GPS 3	H	M(L-BAND)	L	L/M*
S ₃	INU + SITAN	M	M(C-BAND)	M (C-BAND)	L
S ₄	INU + DOR + S/L	M	M(X-BAND)	H/M (X-BAND)	M
S ₅	GPS 3 + DPR + BAR	M	H(L-BAND) (X-BAND)	M (X-BAND)	M*
S ₆	GPS 3 + BAR	M	M(L-BAND)	L	M*
S ₇	INU + DPR	M	L(X-BAND)	M (X-BAND)	L
S ₈	INU + S/L	M	M(X-BAND)	H/M (X-BAND)	M
S ₉	INU	L	L	L	M*

Table 6 Tentative Premises for Mission Profile

- **Ground Alignment/Climb/Cruise:**
Judge INU quality from alignment status/observables
Accuracy, robustness, detectability, workload requirements not demanding
- **Low-Level INGRESS:**
Detectability and robustness are primarily navigation concerns
Pilot attention to navigation diagnosis limited
Accuracy requirements not stringent
- **Pre-IP/IP:**
Navigation awareness peaks
Accuracy — as it affects bombing system performance — is primary concern
Critical decision point for updating/moding navigation system or choosing alternative delivery profile
- **Post-IP/Attack:**
Assume NAV system working as confirmed at IP unless told otherwise
No time for diagnosis
- **EGRESS:**
Generally more relaxed navigation requirements
A few points where reliable navigation is important

These premises were refined through the crew interviews and were then codified as the weighting parameters provided in Table 7. These weightings were assigned by designer judgment to reflect, for example, that detectability and ECM susceptibility are greater concerns than accuracy or workload reduction during ingress. Similarly, the crews confirmed that navigation-related workload would be unacceptable during the post-IP/attack phase.

Table 7 Mission Manager Weightings

MISSION PHASE	MODE ATTRIBUTES	ACCURACY	ECM SUSCEPTIBILITY	DETECTABILITY	WORKLOAD
Ground, Climb, Cruise		.4	.4	.1	.1
Low-Level Ingress		.2	.3	.3	.2
Pre-IP/IP		.5	.1	.1	.1
Post-IP/Attack		0	0	0	0
Egress		.2	.3	.3	.2

Field data (e.g., jam tolerance, reacquisition characteristics) related to GPS performance in a jamming environment was not available to the ATN program due to classification restrictions. The ATN approach was, therefore, to build unclassified jamming scenarios and to use high-fidelity simulation to derive performance characteristics data to support ATN design and evaluation.

Event Diagnosis — A Causal Network knowledge representation scheme was adopted for the Event Diagnosis Expert. An example network is shown in Figure 6. The network is an extrapolation, assuming full Expert Navigation System communication between ships and availability of two navigation solutions (aided and unaided INS) within a single aircraft. Interpretation of the network probabilities is shown in Figure 7 along with an example set of probabilities. These values represent a baseline, engineered from designer judgment about the evidence values of the variables in the network.

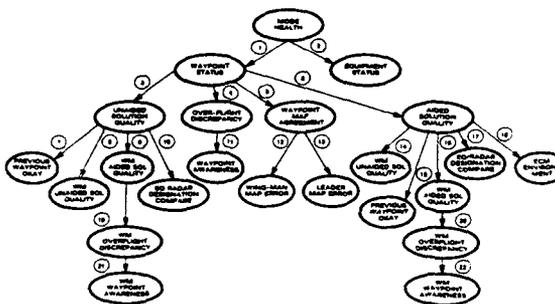


Figure 6 Causal Network for GPS-Aided INS

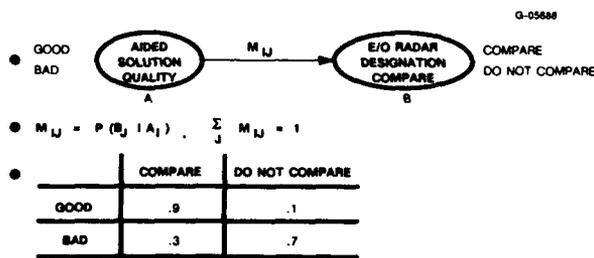


Figure 7 Event Diagnosis — Baseline Probability

Most of the knowledge of the Event Diagnosis module had to be engineered since the navigation suite of ATN does not yet exist, and, therefore, no one is conversant with related troubleshooting procedures. Nevertheless, the module design adapted and generalized the techniques that present day flight crews use to diagnose navigation problems. These techniques (in particular

from F-16C and F-4 crew discussions) utilize crosschecking from ship to ship.

PVI Manager — Knowledge within the PVI Manager consists of heuristically-assigned values for the display periods (timeouts) of ENS dialogue icons and automatic display modes. The time-outs were determined by the module designer based on discussions with F-16C, F-4E, and FB-111 crews; the values so chosen are provided in Table 8. As indicated in the table, more time is assumed available for crew-ENS interaction during the ground/cruise and egress mission segments than during ingress and pre-IP phases.

Table 8 Baseline Timeouts — Seconds

MISSION PHASE ICON TYPE	GROUND-CRUISE	INGRESS	PRE-IP	ATTACK	EGRESS
Event Diagnosis Request	20	10	10	-	20
Moding Advisory	20	10	10	-	20
Waypoint Prompt	60	30	20	-	60

ENS Development and System Integration

— The ENS was developed using the Activation Framework paradigm (Ref 1) to provide a communication and control mechanism for all event-driven functions with links to clock-driven objects. Some important features of this paradigm are: message passing between Activation Framework modules (AFs), scheduling of clock-driven functions that are not AFs, and a bridge mechanism that allows communication between clock-driven functions and AFs. The six experts of the ENS are shown in Figure 8 with a distinction made between AFs and the one non-AF expert. This paradigm facilitated modular design and development, real-time scheduling efficiency, and effective focus of attention.

The ENS modules were developed, verified, and validated individually; the modules were then integrated into the ATN integration environment one at a time. Upon completion of ENS integration, system testing of the ENS was performed, in-

cluding simulated network traffic. In this environment, traffic to and from the network was emulated by scripts and output files. The test scripts included messages to perform a full initialization, a replan following a new threat report, and routine functions such as waypoint checks and diagnoses. Output files were reviewed to establish proper ENS functioning and message passing to the bridge. Upon successful completion of the ENS integration and testing, the ENS was integrated with the BNS for verification and functional evaluation.

The resulting ATN Real-Time Testbed is shown in Figure 9. The system was hosted on commercially-available platforms — the environment file (output of the E/SS) and BNS on a VAX 6310, the ENS on a VAXStation 3100. A single VAX Common LISP process implemented the ENS software; ENS modules were implemented within individual Common LISP packages. Communication among the processors was via ethernet (VAX to VAX) and serial port (VAX to PC). Key functionality provided by this testbed includes a six degree-of-freedom aircraft simulation, high fidelity sensor and threat models, full BNS and ENS implementations, simulated cockpit displays, and an engineering interface. The testbed was used for subject-interactive evaluation of ATN and as a staging platform for integrating ATN into a real-time cockpit simulator.

Testing of the integrated ATN system revealed processing bottlenecks that impaired real-time performance. In the interest of developing general software technology for real-time intelligent system implementation, refinement of the ATN system was performed. This included decreasing the frequency of navigation resource planning in the Mission Manager to one mode change per leg, reducing the numerical precision of the jammer power calculations, and replacing backward chaining inferencing in the System Status expert with tables (the original form of the knowledge). Execution speed of the ENS was increased and real time operation was achieved.

Partitioning of processing during the engineering development provided a single ENS hardware/software/communications implementation that was compatible with both the engineering testbed and cockpit simulator installation. This "plug compatible" implementation enabled upgrading, refining, and, when necessary, troubleshooting in the engineering installation. Thus, cockpit simulator time was reserved for cockpit display development and evaluation testing.

Rehosting to Cockpit Environment — The cockpit integration/evaluation was pursued as a

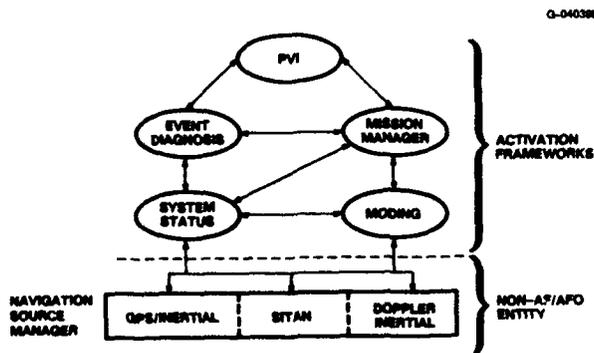


Figure 8 ENS Architecture

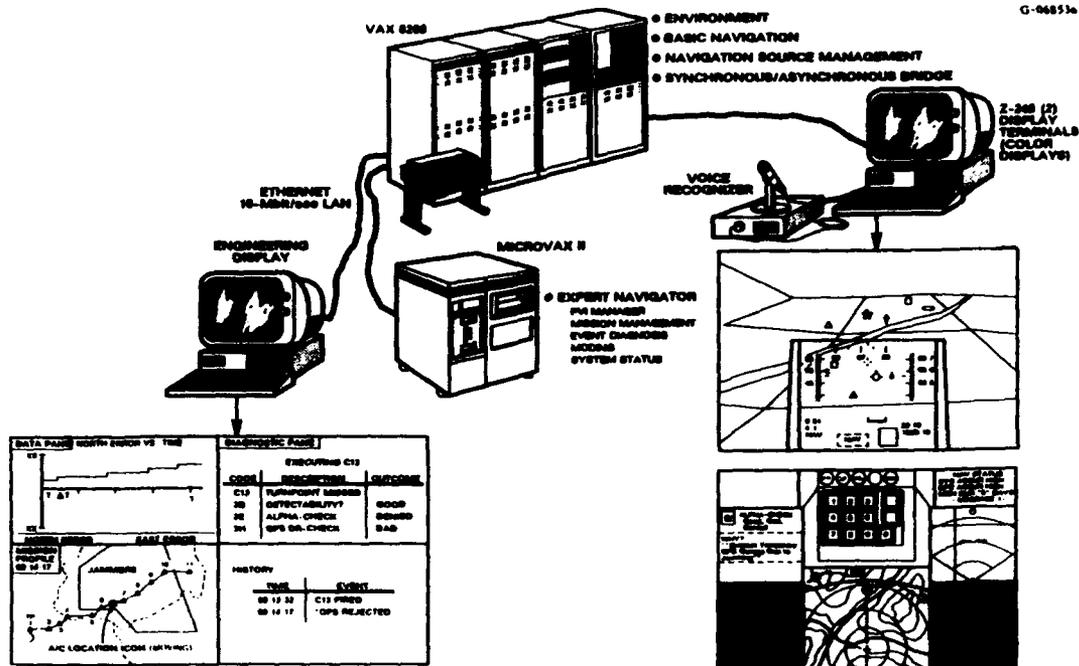


Figure 9 ATN Real-Time Testbed

joint effort with subcontractor General Dynamics/Fort Worth Division (GDFW). The ATN installation was sited in a GDFW 24 ft Dome; an overview of the integration is shown in Figure 10. Much of the integration was derived from existing F-16C simulator functionality. The main simulation processors host the airframe dynamics/flight control, sensor models and simulated fire control computer processing. The main processor(s) also provide stroke-text to the displays and I/O interfaces to other specialized processors. Out-of-cockpit and IR imagery were processed by an Evans and Sutherland CT6 scene generator. Radar imagery was derived from Digital Radar Land Mass System (DRLMS) data.

A number of modifications to the ATN system were implemented to make the system suitable for integration into the F-16 cockpit installation. Among these modifications were:

Reduced-order navigation models to serve as a BNS (INS, GPS, SITAN, RADAR/EO)

Exclusion of the Navigation Source Manager from the cockpit integration

Integration of fully automatic displays Elimination of Waypoint and Event Diagnosis requests

Addition of timeouts for advisories (while flashing) to alert the pilot of mode changes.

ATN SYSTEM EVALUATION

Evaluation of the ATN system was performed in two phases: the Engineering Evaluation Phase conducted in the real time engineering laboratory at TASC, and the Cockpit Evaluation Phase conducted in an F-16 dome at GDFW. The two evaluation phases are summarized below:

Engineering System Evaluation

In the Engineering Evaluation Phase, navigation system performance, resource management, pilot workload, and the pilot/mission interface were evaluated with and without ATN assistance. Subjects for these tests were TASC engineers, two of whom were former rated military personnel with experience in high performance aircraft. This phase of testing used high fidelity models of the navigation sensors and the types of degradations to which they are vulnerable. The computations required to execute these models dictated that they, and the trajectory of the aircraft, be pre-computed before the real time testing. The accuracy of these models allowed the evaluation of different techniques for the detection and evaluation of sensor failures and external interference.

Since the aircraft flight path was pre-computed, a target designation secondary task was

G-17212
10-16-89

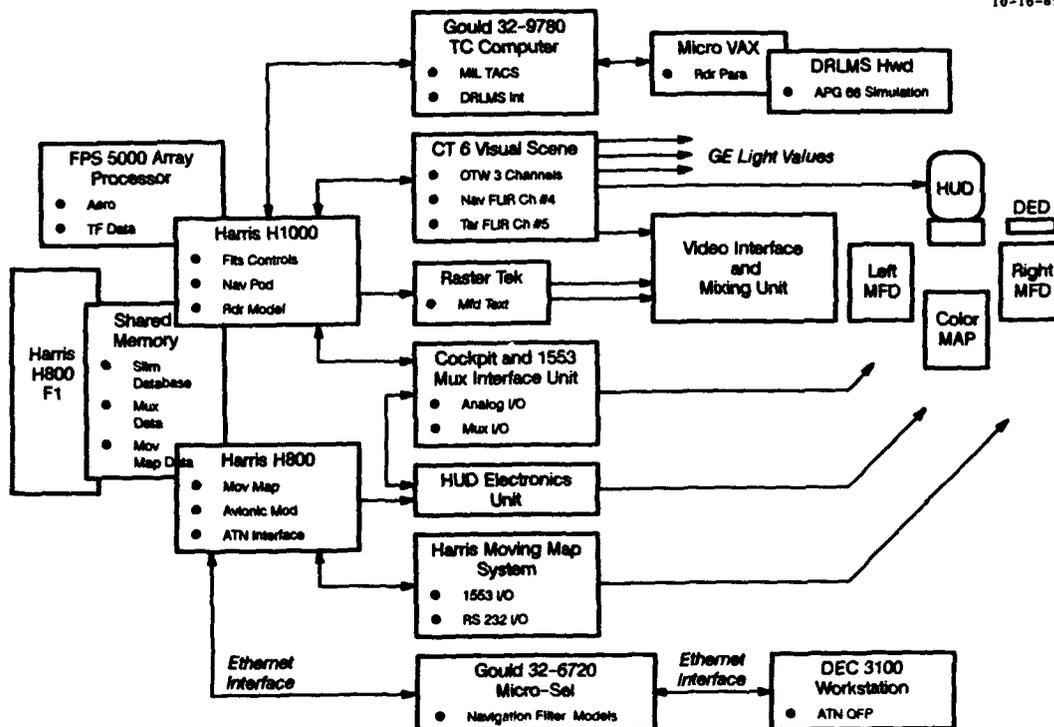


Figure 10 ATN Cockpit Integration

incorporated as part of the real time testing to substitute for actual crew workload. The secondary task display and scoring system are presented in Figure 11 and Table 9. A full description of the experimental method is provided in Reference 1. The primary objectives of this phase were to investigate the behavior of ATN in a subject-interactive mode and to determine ATN effectiveness

in detecting and responding to sensor failures. Key evaluation results include:

The ATN system detected and properly responded to a variety of mission and system events including failures, threat warnings, and map errors

The system interface with the crew was not totally satisfactory; further refinement was required before the cockpit evaluation phase

The ATN system provided optimal moding, response to environment changes (i.e., unbriefed jammers) and resolution of pilot

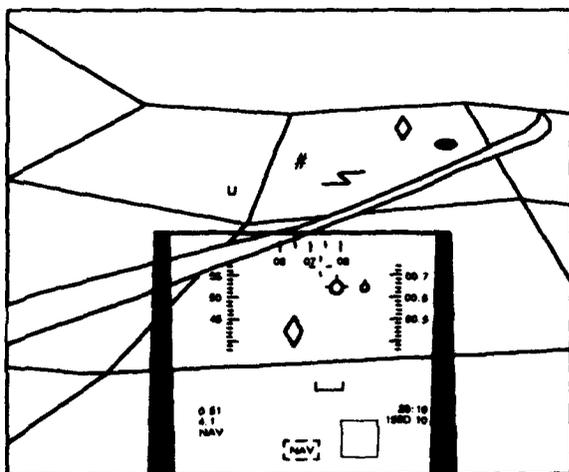


Figure 11 Out of Cockpit Scene and Secondary Task

Table 9 Secondary Task Scoring

ICON	NEUTRALIZATION SCORE
#	
◇	1
◊	2
U	0
○	$U(b,2); 1 \leq b \leq 3$
	0
	-5

observed anomalies (e.g., misplaced waypoints).

Moding decision errors were common in manual mode, particularly with respect to emissions management and over-conservatism (e.g., downmoding in response to a missed waypoint).

Cockpit Simulator Evaluation

The second phase of evaluation, Cockpit Evaluation, was performed in an F-16 Dome in the Flight Simulation Laboratory at GDFW. In this phase, the functionality, crew interface, and mission utility of ATN were evaluated in a simulated operational setting.

The ATN engineering effort and evaluation provided essential groundwork for rehosting and evaluating ATN in an F-16C cockpit simulator. The objectives of the cockpit integration and evaluation were to:

Exercise the ATN system in a realistic task environment

Obtain subjective data on ATN utility and mission effectiveness

Integrate ATN displays into a military standard layout

Assess Refinements and System Personalization

A conventional navigation mode manager was implemented for comparison purposes. This manager was an extrapolation of current practices, using table-driven moding based upon filter covariance values and certain cockpit switch settings (i.e., "RF silent").

A challenging mission (Figure 12) was designed that involved a 40 minute ingress to attack a defended, known location target. Sector jammers were deployed as shown to induce failures of GPS during ingress and to preclude use of GPS during the target attack. The target was a rail bridge crossing a river, having no distinctive collateral features to aid location. A combination of simulated air defenses forced most of the mission to be flown at low levels. Varying visibilities forced reliance on simulated EO systems. EW threats combined with variable terrain roughness prevented total reliance on either GPS or terrain aided navigation. The mission and threats were varied to assess the relative strengths of the ATN system and the conventional system over a range of conditions.

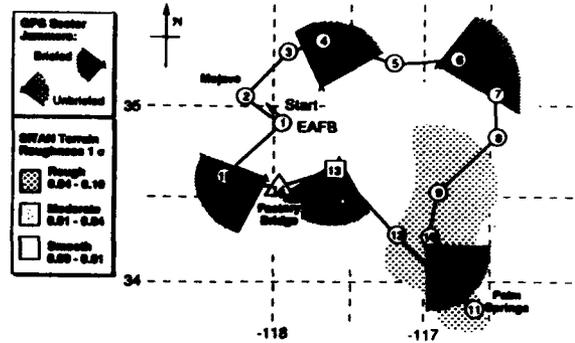


Figure 12 ATN Edwards Gaming Area

Tests were conducted at the GDFW facility during the last three months of 1989. "Pilots" for these missions included contractor engineers and pilots and rated US Air Force participants. During these tests, the system interface with the crew was simplified, and a lag filter fault detection concept was implemented and evaluated.

The availability of precision navigation during the mission, either from the conventional system, or from the ATN system, made a noticeable improvement in the crew's ability to avoid threats and engage the target. A significant benefit demonstrated by ATN was in sensor cuing. By maintaining high accuracy navigation into the attack phase of the mission, direct acquisition of the target was possible using only the EO targeting pod. This capability obviated the need for first acquiring the target with radar, thereby reducing both emissions and required workload in the attack phase. The conventional moding system, by contrast, was vulnerable to countermeasures in the terminal area.

The cockpit simulator evaluation was accomplished in a three cycles of testing and refinement. The cycles were accomplished over a six week period in three installments of one week of tests followed by a week of data analysis and system refinement. 2-3 pilots per cycle participated. Following the final tests, a demonstration was provided for the Government.

Significant revision of the pilot interface was required as a result of the Cycle 1 tests. Initial feedback relative to displays was sufficiently strong that it dominated the subjective results from these tests. The team response was to partially revise the displays prior to the final Cycle 1 tests, complete the revisions for Cycle 2, and perform two additional in-house tests for formal data collection.

Given the relatively small sample of subjects (four GD and two Air Force pilots) trends reported

are interpretations of the evaluation team. Representative error traces and weapon accuracies were obtained to illustrate the trends.

PROGRAM RESULTS/LESSONS LEARNED

Conclusions reached as a result of the ATN program span a broad range of topics, reflecting the equally large range of issues addressed in designing, developing, and evaluating the ATN system. Key results and lessons learned are presented in this section. Additional details are provided in Reference 2.

Functionality — A reasonable set of functions was selected for the ATN system to perform. The functions chosen all fulfilled their required support roles as members of the ATN "community" and included: fault detection, handling unexpected occurrences, using redundant information, and real-time planning/replanning.

The real-time planning/replanning function distinguishes ATN from current table-driven moding algorithms. ATN's Mission Manager uses an on-board model of the mission and navigation requirements to anticipate and configure the navigation system for changes in environment, navigation requirements (i.e., required accuracy, emissions, constraints, and available crew workload), and equipment status. By contrast, the table driven approaches rely on sensor diagnostics (e.g., GPS receiver status), alone, to determine changes in mission environment and predetermined logic to determine moding. The real-time, model-based anticipation of the ATN provides a significant defense against environment-induced sensor data corruption, inappropriate emissions management and increased pilot workload.

A single self-consistency Kalman filter for GPS measurements was effective in detecting GPS degradation. Similarly, a lag filter for a "sanity check" provided noticeable benefits in avoiding use of corrupted data.

A significant observation was that there is an absolute requirement for a fault tolerant inertial data source if one is to realistically meet mission requirements for an IMC interdiction mission. As more real world experience becomes available with advanced sensors, it will be worthwhile to re-investigate fault detection using a combination of system-level and low-level deep knowledge techniques.

Real-time Inferencing, Planning Techniques, and Knowledge Structures — All of the paradigms chosen and developed for the individual experts performed well, given the current state of

the knowledge base. As ATN functionality expands, some modifications or extension of these paradigms may be appropriate.

Key elements of the successful, real-time ATN integration from an architectural standpoint include:

Partitioning of synchronous and asynchronous computations into separate processors.

Distributing control of the ATN system using the Activation Framework Paradigm.

Reasoning about time — in particular, performing localized replanning in the Mission Manager and reasoning about time costs of acquiring information in Event Diagnosis.

Using table lookup procedures rather than rules when warranted for efficiency.

Using the Activation Framework approach obviated efficiency problems associated with architectures that use a blackboard and centralized controller. In addition, use of messages and a structure of message priorities significantly facilitated modularization and eventual integration of the system while providing flexibility of module development.

Distribution of Intelligence — Intelligence is best managed if it is specific; to the extent possible, intelligence and knowledge should be partitioned into specialized domains that cooperate effectively. Within the ATN system, intelligence was distributed according to the following philosophy: Levels of abstraction generally are decided by the type of information used; lower level functions may use raw sensor data while higher levels use derived information.

Crew Interface — ATN evaluation results indicated that pilots want the navigation system to be reliable, accurate, and to work in the background. Crew acceptance of what the ATN system was doing--along with appropriateness of the "touch and feel" aspects--proved to be crucial for system success. In response to strong recommendations from pilot evaluators of the ATN system, the recommended ATN crew interface consists of:

A HUD display showing the current navigation mode and warning or caution status, if active (these display elements flash for 5 seconds following a change of mode or status).

A moving map which serves as a navigation aid and a threat situation display.

A status page presenting the current mode status, results of the last cross-check, and accuracy.

Knowledge Acquisition — Knowledge acquisition was one of the most challenging aspects of the ATN system design and development. The wide range of required knowledge areas and lack of user experience with advanced avionics subsystems necessitated the use of a combination of analysis, simulation, extrapolation, and intuition, in addition to traditional knowledge acquisition techniques. In specific cases, subsystem model simulations with expert interpretation of results provided an important bridge between current experience and projected system behavior. Key in this aspect was the ability to simulate GPS in jamming scenarios. Knowledge acquisition experiences also emphasized the value of having simulation tools available early and being able to affordably run them frequently for the purposes of expanding and validating the knowledge base.

An organized pre-brief and mission-centered questionnaire were essential to efficient discussions with the aircrews. Gaining familiarity with the aircrew's systems, tactics, and vocabulary and being able to relate these to the ATN sensor suite and mission paid large dividends during the interviews and knowledge analysis phases.

System Development — The traditional system development approach used in the ATN program had significant drawbacks. It required all functions to be specified at a uniform level of detail regardless of the actual maturity of the algorithms. Many of the selected AI paradigms had never been implemented in an avionics application. The significant documentation burden diverted resources away from early prototyping and design risk reduction. Much of the early documentation became obsolete due to design changes that were required after attempting to implement the functions. The spiral model of software development that has recently appeared in the literature appears to offer a sounder basis for development of a system of this nature than the waterfall model that the ATN project used. Documentation that is an asset rather than an impediment to a project of this nature requires a high (perhaps Utopian) level of honesty, technical competence, trust, and daring on both the Government and contractor sides.

The phasing of knowledge acquisition interviews could also be modified to support a spiral development approach. Discussions early in the post-requirements definition phase would be useful to help focus functional development and knowledge acquisition planning. A second round of knowledge acquisition would be supported by a first prototype; some level of interaction with this prototype would supplement the mission-centered interviews.

Computer Languages — Common LISP proved to be a good design/prototyping language for the ATN effort. A language offering greater control of memory management would be desirable for future development/implementation of embedded systems applications.

Weapon Delivery Capabilities — Continuous high accuracy navigation (<80 meters) substantially improved the crew's ability to engage a prebriefed target in IMC conditions when using an EO pod and eliminated the need to use the attack radar as part of the target acquisition.

Navigation System Management — A navigation management system that supplemented the standard algorithmic techniques with symbolic knowledge about the system components, threats, and mission as factors in its execution produced measurably better navigation performance. Development of this system manager required the use of a broad range of AI-based structures.

SUMMARY

The ATN project was intended to investigate the utility of intelligent system management for integrated navigation on a next generation combat aircraft. Significant insight was gained into the suitability of current AI paradigms and software development techniques for a project of this nature. The results and experiences during this project indicated that such an intelligent system manager could produce a measurable benefit in the real world. This benefit should become even more significant as operational users' experiences yield a better understanding of the actual behavior of the next generation of navigation sensors.

REFERENCES

1. Acharya, N.A., Dowding, J.P., Glasson, D.P., Matchett, G.A., and Pomarede, J.L., ATN Software Part 1 Specification — Technical Appendices, TASC, AFWAL-TR-88-1159 (AD-B128-656), October 1987.
2. Glasson, D.P., et al., Development and Real-Time Laboratory Demonstration of an Adaptive Tactical Navigation System, TASC, WRDC-TR-90-1034 (AD-B151-955), May 1990.
3. Matchett, G.A., Environment/Sensor System Simulation — Technical Appendices, TASC, AFWAL-TR-88-1160 (AD-B129-671), October 1987.

Discussion

1. D. Bosman, Netherlands

Kalman filters of different design and manufacture may have differing signals delays. For a federated system, it must be assumed that all data originate from the same time-slice. Are the delay differences sufficiently small to be disregarded?

Author:

Since the ATN program was a proof of concept for intelligent management techniques, our navigation simulation was not so detailed as to include simulation of time offsets between sensors or between the Kalman filters associated with the sensor. In an actual system implementation, these differences should not be ignored. In our organization, these issues are being addressed by

advanced Kalman filter efforts rather than system management efforts.

2. H. Timmers, Netherlands

How much from the developed system eventually was rule based and how much "switch-logic" or table driven?

Author:

I would estimate that the knowledge based components of the ATN expert modules are approximately 50% rule based and 50% table look-up and/or algorithmic (i.e., "conventional" techniques). The Mission Manager and the System Status Expert are both primarily rule based. The Pilot Vehicle Interfare Manager and the Moding Expert use more conventional techniques, while Event Diagnosis and the Navigation Source manager are a combination.



COMPONENT PART NOTICE

THIS PAPER IS A COMPONENT PART OF THE FOLLOWING COMPILATION REPORT:

(TITLE): Conference Proceedings of Machine Intelligence for Aerospace Electronic Systems Held in Lisbon, Portugal on 13-16 May 1991 (L'Intelligence Artificielle dans les Systemes Electroniques Aero spatiaux)

(SOURCE): Advisory Group for Aeronautical Research and Development, Neuilly-sur-Seine (France)

TO ORDER THE COMPLETE COMPILATION REPORT USE AD-N242025

THE COMPONENT PART IS PROVIDED HERE TO ALLOW USERS ACCESS TO INDIVIDUALLY AUTHORED SECTIONS OF PROCEEDINGS, ANNALS, SYMPOSIA, ETC. HOWEVER, THE COMPONENT SHOULD BE CONSIDERED WITHIN THE CONTEXT OF THE OVERALL COMPILATION REPORT AND NOT AS A STAND-ALONE TECHNICAL REPORT.

THE FOLLOWING COMPONENT PART NUMBERS COMPRISE THE COMPILATION REPORT:

AD#: _____ TITLE: _____

AD-P006 339

DTIC
ELECTE
NOV 13 1991
S D D

Accession For	
NTIS CRA&I	<input checked="" type="checkbox"/>
DTIC TAB	<input type="checkbox"/>
Unannounced	<input type="checkbox"/>
Justification	
By _____	
Distribution/	
Availability Codes	
Dist	Avail and/or Special
A-1	21

This document has been approved for public release and sale; its distribution is unlimited.

AD-P006 339



LOCALLY LINEAR NEURAL NETWORKS FOR AEROSPACE NAVIGATION SYSTEMS

Steven C. Gustafson and Gordon R. Little
Research Institute, University of Dayton,
513-229-3724, 300 College Park,
Dayton, Ohio, 45469-0140, USA

1. SUMMARY

Neural network software simulations for the representation and prediction of aircraft inertial navigation system (INS) data were developed. These simulations were evaluated using flight test data that sampled INS outputs at a standard rate for neural network testing and at half this rate for neural network training. The simulations used both locally linear neural networks and backpropagation-trained neural networks. Locally linear neural networks have several desirable properties for this application, including interpolation of the training data and representation of linear relationships. For the flight test data two milliradian testing accuracy was generally achieved with five successive and prior INS heading, pitch, and roll increments as inputs.

2. LOCALLY LINEAR NEURAL NETWORK (LLNN) CHARACTERISTICS

Multilayer feedforward neural networks that use backpropagation [Rumelhart, Hinton, and Williams, 1986] or similar training algorithms are by far the most common in engineering applications. However, these neural networks have several limitations. First, they generally lack the coordinate invariance property, according to which the testing output is unchanged if the testing and training (or data) inputs are translated, rotated, or scaled. Second, without excessive training they generally lack the data interpolation property, according to which the testing output is the training output if the testing input is the training input [Poggio and Girosi, 1990]. Third, they generally lack the linear representation property, according to which any testing point is on a linear surface if all training points are on this surface. Finally, they generally lack

the data bootstrapping property, according to which the testing outputs have least squared error for any training points declared to be testing points. In contrast, the locally linear neural networks discussed below have all of the above desirable characteristics.

3. LLNN STRUCTURE AND TRAINING ALGORITHM

Figure 1, adopted from Nielson [1987], gives an example of the need for coordinate-invariant neural networks. Here an expression with five adjustable coefficients is matched to five training points. When the scales of the two independent variables are multiplied by different constants, interpolation using the expression gives different results for the same testing point.

Figure 2 describes the setup, training, and testing of locally linear neural networks [Gustafson, Little, and Olczak, 1990]. There are two essential steps in training: transforming the inputs to invariant coordinates and finding a plane through each training point that satisfies a selected bootstrapping property. In testing the plane through the training point nearest to the testing point (in the transformed inputs) is used to find the testing point output.

Figure 3 shows a simple example of one possible bootstrapping procedure. Here for one input a line is found through a training point such that the line predicts, with minimum squared error, either one training point transformed into a testing point or all but one training points transformed into testing points.

4. INS FLIGHT TEST DATA

Figure 4 shows typical F-16 aircraft INS flight test data that includes heading, pitch, and roll versus time.

Differences in successive samples of heading, pitch, and roll are to be represented and predicted using neural networks so as to maintain the pointing and tracking accuracy of a passive optical device detection system [Gustafson and Little, 1988]. Figure 5 shows a typical plot of such differences between successive samples of pitch versus time.

5. NEURAL NETWORK REPRESENTATION OF INS DATA

In all cases data that sampled the INS outputs at a standard rate was used for neural network testing, and data at half this rate was used for training. Figure 6 shows a standard backpropagation-trained neural network representation of the data in Figure 5. The neural network had 7 hidden neurons and 15 inputs consisting of five successive prior heading, pitch, and roll increments. Training time on a 386-class PC was approximately 1.5 hours. Figure 7 represents the same data using the same neural network but with less training. Note that the "hash" detail of Figure 5 is better represented after a longer training time. Also note that the LLNN would represent all training data exactly because of the data interpolation property.

6. NEURAL NETWORK PREDICTION OF INS DATA

Figure 8 shows another plot of differences between successive samples of flight test pitch versus time. Figure 9 shows neural network prediction of the Figure 8 data using a locally linear neural network that was bootstrapped so that of 5 nearest neighbor training points, all except the closest were transformed into testing points. Figure 10 shows prediction of the Figure 8 data using a backpropagation-trained neural network that had 7 hidden neurons. Both neural networks had as inputs 5 successive heading, pitch, and roll increments prior to a given pitch increment, and both had as output a prediction of this increment. Figure 11 shows histograms of differences between predicted (Figures 9 and 10) and actual (Figure 8) pitch increments for the locally linear and backpropagation-trained neural networks.

Figures 8, 9, and 10 show that the locally linear neural network predicted more of the "hash" structure in the data than the backpropagation-trained neural network. Figure 11 shows that predictions using both neural networks were typically within 0.1 degrees (1.7 milliradians) of actual values and that the error distribution for the locally linear neural network was more symmetric.

7. REFERENCES

S. C. Gustafson and G. R. Little, "Passive Optical Detection Schemes," report for the Center for Artificial Intelligence Applications, A Division of the Miami Valley Research Institute, on contract F33615-87-C-1550, Dec. 1988.

S. C. Gustafson, G. R. Little, and D. M. Simon, "Neural Network for Interpolation and Extrapolation," Proc. SPIE, V. 1294, No. 40, Apr. 1990.

S. C. Gustafson, G. R. Little, and E. G. Olczak, "Locally Linear Neural Networks," Aerospace Applications of Artificial Intelligence Conf. (AAIC), Dayton, OH, Oct. 1990.

S. C. Gustafson and G. R. Little, "Neural Networks for Processing Inertial Navigation Unit Data," Aerospace Applications of Artificial Intelligence Conference (AAIC), Dayton, OH, Oct. 1990.

G. M. Nielson, "Coordinate-free Scattered Data Interpolation," pp. 175-184 in Topics in Multivariable Approximation, C.K. Chui and L. L. Schumaker, eds., Academic, 1987.

T. Poggio and F. Girosi, "Networks for Approximation and Learning," Proc. IEEE, V. 78, pp. 1481-1497, Sep. 1990.

D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning Internal Representations by Error Propagation," pp. 318-362 in Parallel Distributed Processing, D. E. Rumelhart and J. E. McClelland, eds., MIT Press, 1986.

- **Interpolant (after G. Nielson)**

$$z(x,y) = a + bx + cy + dxy + e(x^2 + y^2)$$

- **Feet-Minute Coordinates**

X	Y	Z	
1	1	0	
1	0	0	
.5	.5	1	$z(.75, .5) = .875$
0	1	0	
0	0	0	

- **Inches-Seconds Coordinates**

X	Y	Z	
12	60	0	
12	0	0	
6	30	1	$z(9, 30) = .990$
0	60	0	
0	0	0	

Figure 1. Example of need for coordinate-invariant neural networks.

- **Setup**

1. Training points, m in number, each with n input variables and one output variable.
2. One testing point with n input variables.

- **Training**

1. Find the centroid of the training point input variables. Find the eigenvalues and eigenvectors of the input variable covariance matrix (e.g., using singular value decomposition). Transform the input variables of all training points to the coordinate system having its origin at the centroid and its axes parallel to the eigenvectors with the axis scales in units of the eigenvalues.
2. Find, using the transformed input variables, a plane through each training point that is a least-squares fit to the remaining training points. Execute the least squares fit so that more distant points in the input variables have less weight in accordance with a selected bootstrapping procedure.

- **Testing**

1. Transform the testing point input variables to the coordinate system obtained in training step 1. Using the transformed input variables, find the training point that is nearest to the testing point.
2. Find the plane from training step 2 through this nearest training point. Find the testing point output variable as the intersection of this plane with the transformed testing point input variables.

Figure 2. Locally linear neural network setup, training, and testing.

91-15521


91 1113 010

o **Problem**

Given three data points (x_0, y_0) , (x_1, y_1) , (x_2, y_2) with $x_0 < x_1 < x_2$, draw a line through (x_0, y_0) that recovers with minimum expected value of the mean squared error, the y value or values indicated below when one of two equally likely events occurs:

- (1) Local event - y_1 is lost (i.e., the y value of the nearest neighbor of x_0 is lost).
- (2) Global event - y_1 and y_2 are lost (i.e., the y values of all neighbors of x_0 are lost).

o **Solution**

The slope b of the required line is determined so as to minimize the sum of squared errors S:

$$S = |y_0 + b(x_1 - x_0) - y_1|^2 + |y_0 + b(x_1 - x_0) - y_1|^2 + |y_0 + b(x_2 - x_0) - y_2|^2 / 2$$

The solution with $y_1 - y_0 = v_1$, $y_2 - y_0 = v_2$, $x_1 - x_0 = u_1$, $x_2 - x_0 = u_2$ is:

$$b = (3v_1u_1 + v_2u_2) / (3u_1^2 + u_2^2)$$

This result may also be obtained as the least squares solution of:

$$y_1 - y_0 = b(x_1 - x_0)$$

$$y_1 - y_0 = b(x_1 - x_0)$$

$$y_1 - y_0 = b(x_1 - x_0)$$

$$y_2 - y_0 = b(x_2 - x_0)$$

Figure 3. Simple example of one possible bootstrapping procedure.

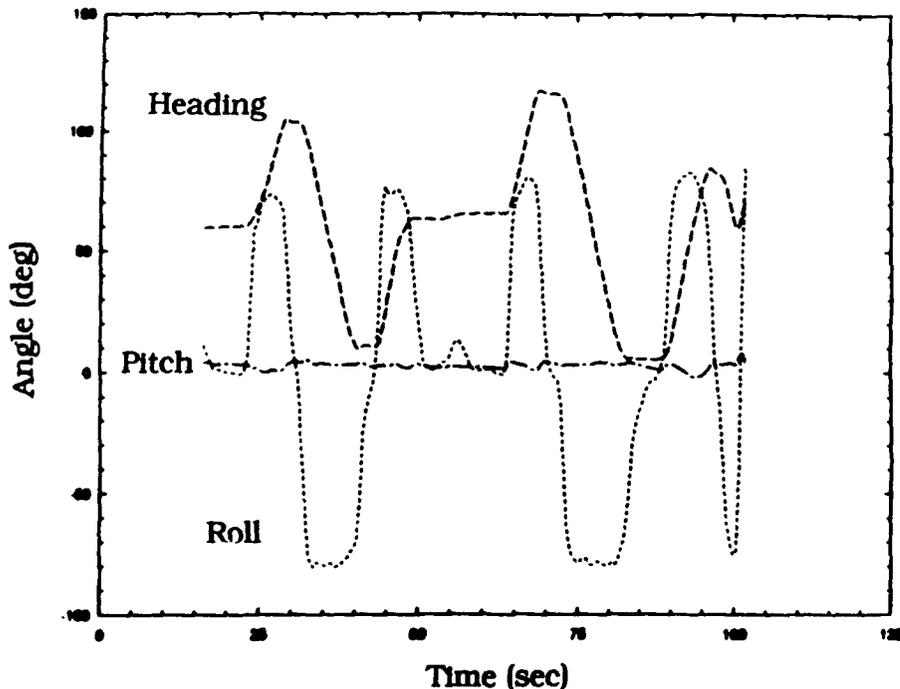


Figure 4. Typical F16 aircraft INS flight test data.

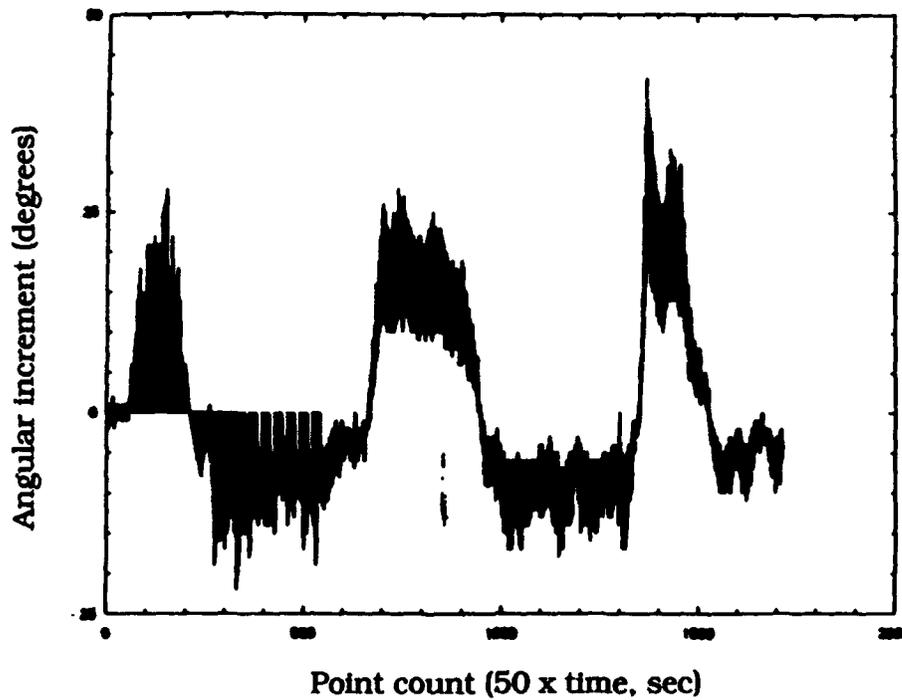


Figure 5. Differences between successive samples of flight-test aircraft pitch angle versus time.

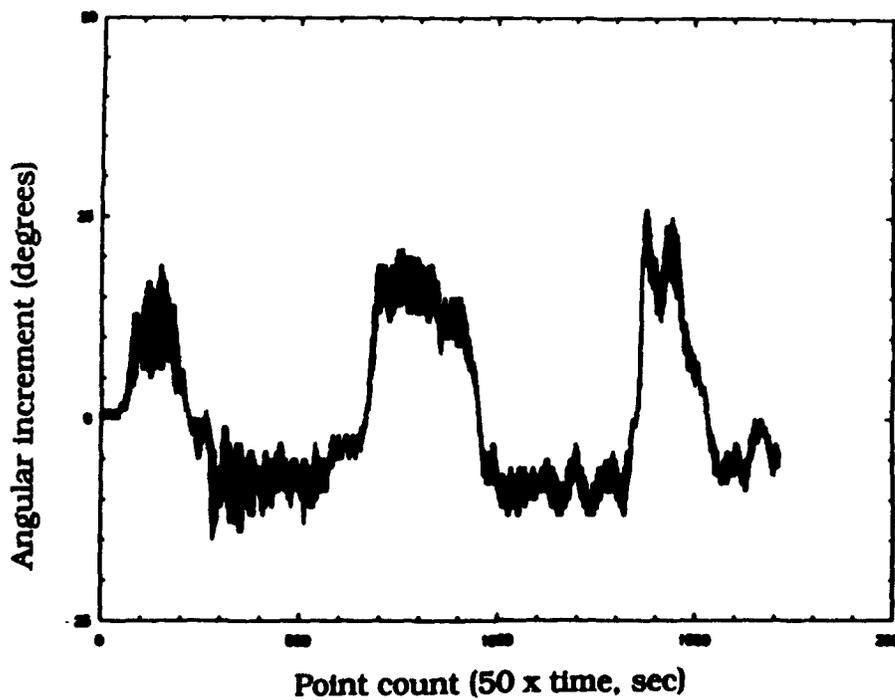


Figure 6. Neural network representation of Figure 5 data (backpropagation training with 7 hidden neurons and 5 successive prior heading, pitch, and roll increments as inputs).

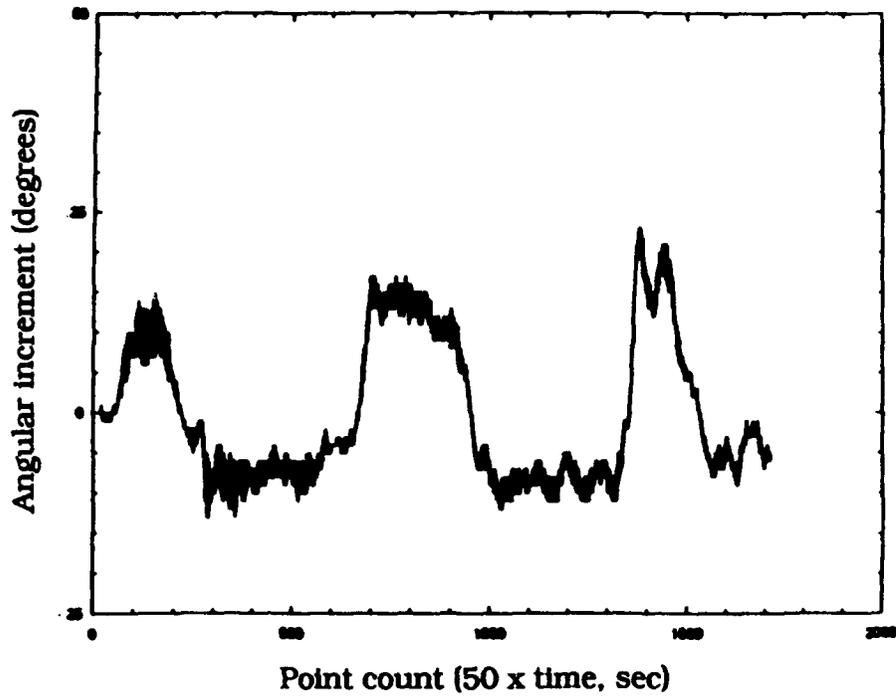


Figure 7. Reduced-training-time neural network representation of Figure 5 data.

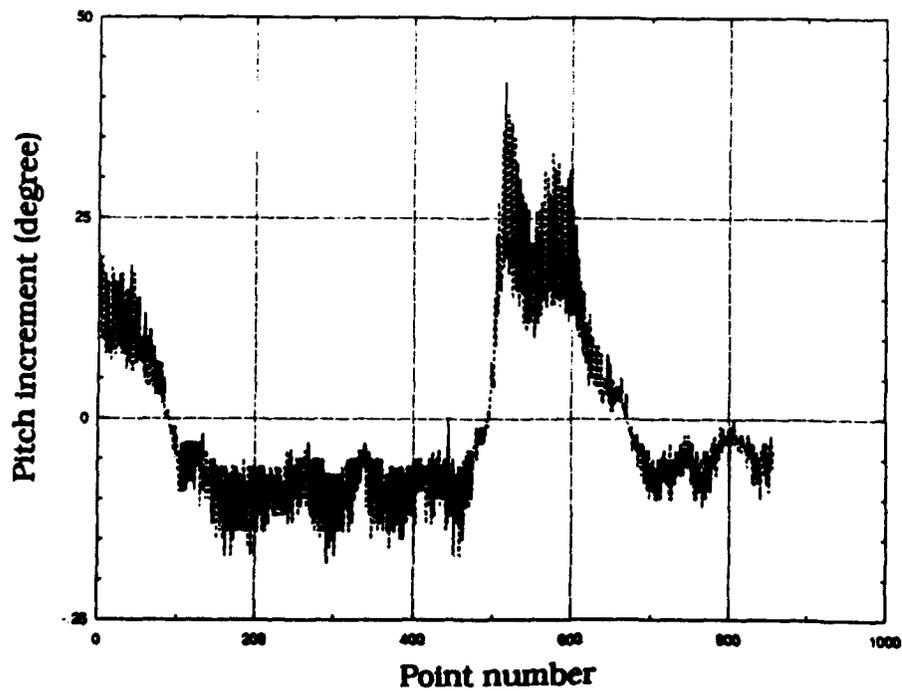


Figure 8. Differences between successive samples of flight-test aircraft pitch angle (0-1000 points) versus time.

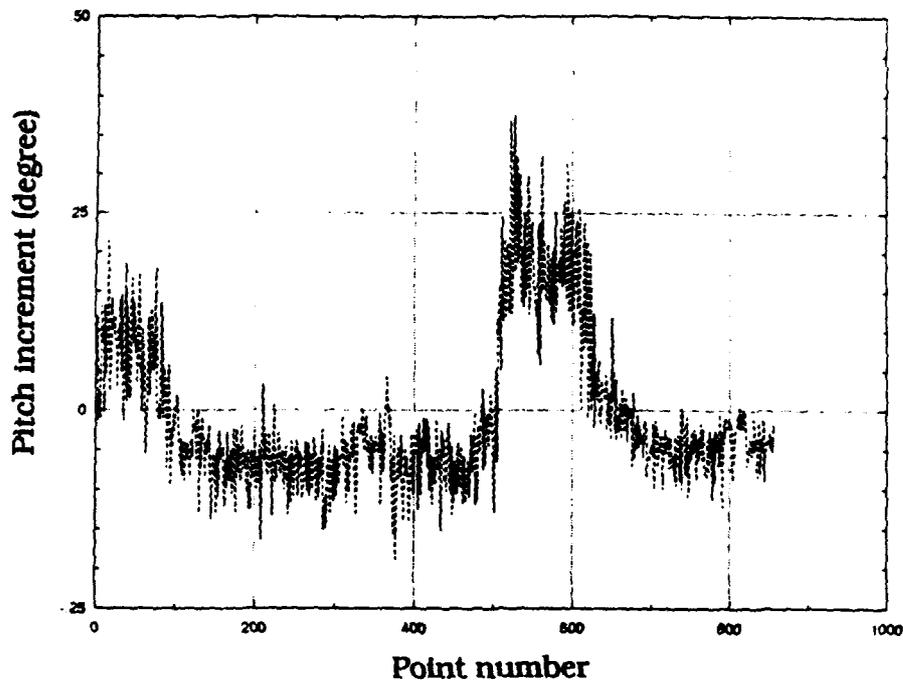


Figure 9. Locally linear neural network prediction of Figure 8 data (5 successive prior heading, pitch, and roll increments as inputs; 5 nearest neighbor training points bootstrapped so that all except closest were transformed into testing points).

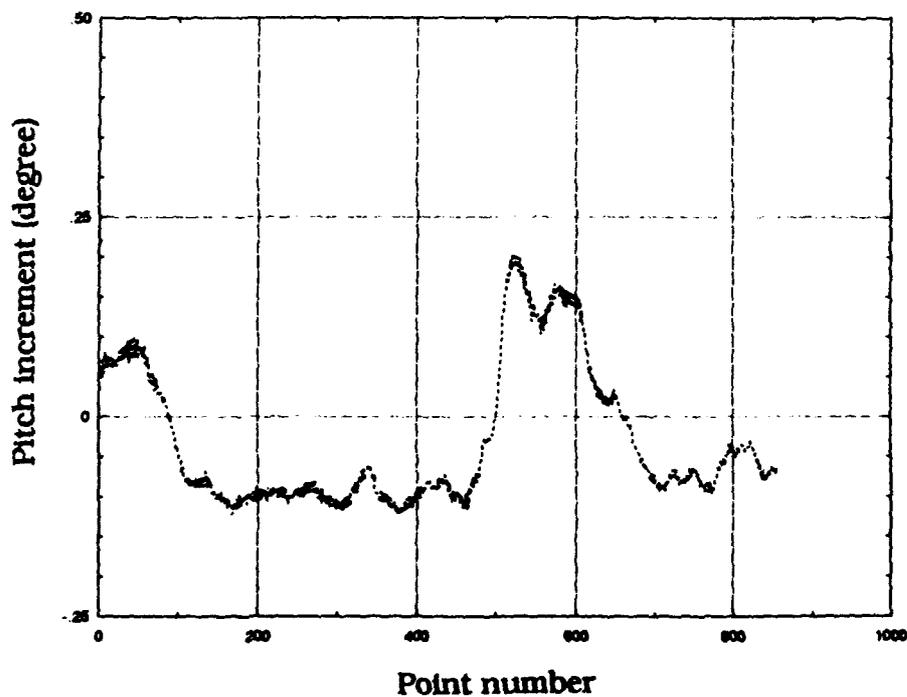


Figure 10. Backpropagation-trained neural network prediction of Figure 8 data (5 successive prior heading, pitch, and roll increments as inputs; training with 7 hidden neurons).

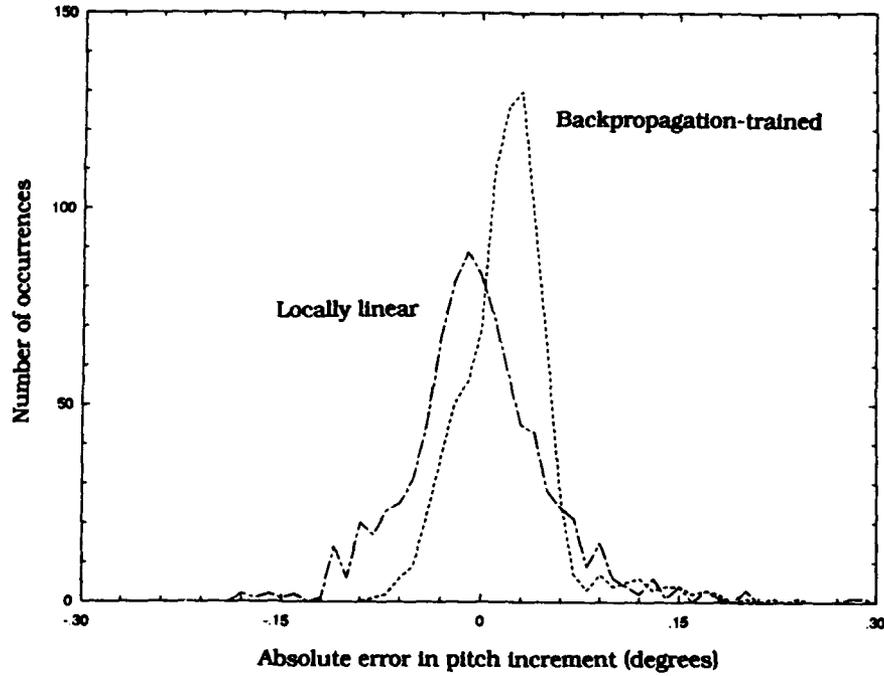


Figure 11. Histograms of differences between predicted (Figure 9 and 10) and actual (Figure 8) pitch increments for locally linear and backpropagation-trained neural networks.



COMPONENT PART NOTICE

THIS PAPER IS A COMPONENT PART OF THE FOLLOWING COMPILATION REPORT:

(TITLE): Conference Proceedings of machine Intelligence
for Aerospace Electronic Systems Held in Lisbon, Portugal on
13-16 May 1991 (L'Intelligence Artificielle dans les Systemes
Electroniques Aeroespaciaux)

(SOURCE): Advisory Group for
Aeronautical Research and Development, Neuilly-sur-Seine
(France)

TO ORDER THE COMPLETE COMPILATION REPORT USE AD-A242025

THE COMPONENT PART IS PROVIDED HERE TO ALLOW USERS ACCESS TO INDIVIDUALLY AUTHORED SECTIONS OF PROCEEDINGS, ANNALS, SYMPOSIA, ETC. HOWEVER, THE COMPONENT SHOULD BE CONSIDERED WITHIN THE CONTEXT OF THE OVERALL COMPILATION REPORT AND NOT AS A STAND-ALONE TECHNICAL REPORT.

THE FOLLOWING COMPONENT PART NUMBERS COMPRISE THE COMPILATION REPORT:

ADM: _____ TITLE: _____

AD-P006 340

S DTIC
ELECTE
NOV 13 1991
D

Acquisition For	
ADIS - GRAAI	J
ADIS - IAS	01
ADIS - IAS	02
ADIS - IAS	03
ADIS - IAS	04
ADIS - IAS	05
ADIS - IAS	06
ADIS - IAS	07
ADIS - IAS	08
ADIS - IAS	09
ADIS - IAS	10
ADIS - IAS	11
ADIS - IAS	12
ADIS - IAS	13
ADIS - IAS	14
ADIS - IAS	15
ADIS - IAS	16
ADIS - IAS	17
ADIS - IAS	18
ADIS - IAS	19
ADIS - IAS	20
ADIS - IAS	21
ADIS - IAS	22
ADIS - IAS	23
ADIS - IAS	24
ADIS - IAS	25
ADIS - IAS	26
ADIS - IAS	27
ADIS - IAS	28
ADIS - IAS	29
ADIS - IAS	30
ADIS - IAS	31
ADIS - IAS	32
ADIS - IAS	33
ADIS - IAS	34
ADIS - IAS	35
ADIS - IAS	36
ADIS - IAS	37
ADIS - IAS	38
ADIS - IAS	39
ADIS - IAS	40
ADIS - IAS	41
ADIS - IAS	42
ADIS - IAS	43
ADIS - IAS	44
ADIS - IAS	45
ADIS - IAS	46
ADIS - IAS	47
ADIS - IAS	48
ADIS - IAS	49
ADIS - IAS	50
ADIS - IAS	51
ADIS - IAS	52
ADIS - IAS	53
ADIS - IAS	54
ADIS - IAS	55
ADIS - IAS	56
ADIS - IAS	57
ADIS - IAS	58
ADIS - IAS	59
ADIS - IAS	60
ADIS - IAS	61
ADIS - IAS	62
ADIS - IAS	63
ADIS - IAS	64
ADIS - IAS	65
ADIS - IAS	66
ADIS - IAS	67
ADIS - IAS	68
ADIS - IAS	69
ADIS - IAS	70
ADIS - IAS	71
ADIS - IAS	72
ADIS - IAS	73
ADIS - IAS	74
ADIS - IAS	75
ADIS - IAS	76
ADIS - IAS	77
ADIS - IAS	78
ADIS - IAS	79
ADIS - IAS	80
ADIS - IAS	81
ADIS - IAS	82
ADIS - IAS	83
ADIS - IAS	84
ADIS - IAS	85
ADIS - IAS	86
ADIS - IAS	87
ADIS - IAS	88
ADIS - IAS	89
ADIS - IAS	90
ADIS - IAS	91
ADIS - IAS	92
ADIS - IAS	93
ADIS - IAS	94
ADIS - IAS	95
ADIS - IAS	96
ADIS - IAS	97
ADIS - IAS	98
ADIS - IAS	99
ADIS - IAS	100
ADIS - IAS	101
ADIS - IAS	102
ADIS - IAS	103
ADIS - IAS	104
ADIS - IAS	105
ADIS - IAS	106
ADIS - IAS	107
ADIS - IAS	108
ADIS - IAS	109
ADIS - IAS	110
ADIS - IAS	111
ADIS - IAS	112
ADIS - IAS	113
ADIS - IAS	114
ADIS - IAS	115
ADIS - IAS	116
ADIS - IAS	117
ADIS - IAS	118
ADIS - IAS	119
ADIS - IAS	120
ADIS - IAS	121
ADIS - IAS	122
ADIS - IAS	123
ADIS - IAS	124
ADIS - IAS	125
ADIS - IAS	126
ADIS - IAS	127
ADIS - IAS	128
ADIS - IAS	129
ADIS - IAS	130
ADIS - IAS	131
ADIS - IAS	132
ADIS - IAS	133
ADIS - IAS	134
ADIS - IAS	135
ADIS - IAS	136
ADIS - IAS	137
ADIS - IAS	138
ADIS - IAS	139
ADIS - IAS	140
ADIS - IAS	141
ADIS - IAS	142
ADIS - IAS	143
ADIS - IAS	144
ADIS - IAS	145
ADIS - IAS	146
ADIS - IAS	147
ADIS - IAS	148
ADIS - IAS	149
ADIS - IAS	150
ADIS - IAS	151
ADIS - IAS	152
ADIS - IAS	153
ADIS - IAS	154
ADIS - IAS	155
ADIS - IAS	156
ADIS - IAS	157
ADIS - IAS	158
ADIS - IAS	159
ADIS - IAS	160
ADIS - IAS	161
ADIS - IAS	162
ADIS - IAS	163
ADIS - IAS	164
ADIS - IAS	165
ADIS - IAS	166
ADIS - IAS	167
ADIS - IAS	168
ADIS - IAS	169
ADIS - IAS	170
ADIS - IAS	171
ADIS - IAS	172
ADIS - IAS	173
ADIS - IAS	174
ADIS - IAS	175
ADIS - IAS	176
ADIS - IAS	177
ADIS - IAS	178
ADIS - IAS	179
ADIS - IAS	180
ADIS - IAS	181
ADIS - IAS	182
ADIS - IAS	183
ADIS - IAS	184
ADIS - IAS	185
ADIS - IAS	186
ADIS - IAS	187
ADIS - IAS	188
ADIS - IAS	189
ADIS - IAS	190
ADIS - IAS	191
ADIS - IAS	192
ADIS - IAS	193
ADIS - IAS	194
ADIS - IAS	195
ADIS - IAS	196
ADIS - IAS	197
ADIS - IAS	198
ADIS - IAS	199
ADIS - IAS	200
ADIS - IAS	201
ADIS - IAS	202
ADIS - IAS	203
ADIS - IAS	204
ADIS - IAS	205
ADIS - IAS	206
ADIS - IAS	207
ADIS - IAS	208
ADIS - IAS	209
ADIS - IAS	210
ADIS - IAS	211
ADIS - IAS	212
ADIS - IAS	213
ADIS - IAS	214
ADIS - IAS	215
ADIS - IAS	216
ADIS - IAS	217
ADIS - IAS	218
ADIS - IAS	219
ADIS - IAS	220
ADIS - IAS	221
ADIS - IAS	222
ADIS - IAS	223
ADIS - IAS	224
ADIS - IAS	225
ADIS - IAS	226
ADIS - IAS	227
ADIS - IAS	228
ADIS - IAS	229
ADIS - IAS	230
ADIS - IAS	231
ADIS - IAS	232
ADIS - IAS	233
ADIS - IAS	234
ADIS - IAS	235
ADIS - IAS	236
ADIS - IAS	237
ADIS - IAS	238
ADIS - IAS	239
ADIS - IAS	240
ADIS - IAS	241
ADIS - IAS	242
ADIS - IAS	243
ADIS - IAS	244
ADIS - IAS	245
ADIS - IAS	246
ADIS - IAS	247
ADIS - IAS	248
ADIS - IAS	249
ADIS - IAS	250
ADIS - IAS	251
ADIS - IAS	252
ADIS - IAS	253
ADIS - IAS	254
ADIS - IAS	255
ADIS - IAS	256
ADIS - IAS	257
ADIS - IAS	258
ADIS - IAS	259
ADIS - IAS	260
ADIS - IAS	261
ADIS - IAS	262
ADIS - IAS	263
ADIS - IAS	264
ADIS - IAS	265
ADIS - IAS	266
ADIS - IAS	267
ADIS - IAS	268
ADIS - IAS	269
ADIS - IAS	270
ADIS - IAS	271
ADIS - IAS	272
ADIS - IAS	273
ADIS - IAS	274
ADIS - IAS	275
ADIS - IAS	276
ADIS - IAS	277
ADIS - IAS	278
ADIS - IAS	279
ADIS - IAS	280
ADIS - IAS	281
ADIS - IAS	282
ADIS - IAS	283
ADIS - IAS	284
ADIS - IAS	285
ADIS - IAS	286
ADIS - IAS	287
ADIS - IAS	288
ADIS - IAS	289
ADIS - IAS	290
ADIS - IAS	291
ADIS - IAS	292
ADIS - IAS	293
ADIS - IAS	294
ADIS - IAS	295
ADIS - IAS	296
ADIS - IAS	297
ADIS - IAS	298
ADIS - IAS	299
ADIS - IAS	300
ADIS - IAS	301
ADIS - IAS	302
ADIS - IAS	303
ADIS - IAS	304
ADIS - IAS	305
ADIS - IAS	306
ADIS - IAS	307
ADIS - IAS	308
ADIS - IAS	309
ADIS - IAS	310
ADIS - IAS	311
ADIS - IAS	312
ADIS - IAS	313
ADIS - IAS	314
ADIS - IAS	315
ADIS - IAS	316
ADIS - IAS	317
ADIS - IAS	318
ADIS - IAS	319
ADIS - IAS	320
ADIS - IAS	321
ADIS - IAS	322
ADIS - IAS	323
ADIS - IAS	324
ADIS - IAS	325
ADIS - IAS	326
ADIS - IAS	327
ADIS - IAS	328
ADIS - IAS	329
ADIS - IAS	330
ADIS - IAS	331
ADIS - IAS	332
ADIS - IAS	333
ADIS - IAS	334
ADIS - IAS	335
ADIS - IAS	336
ADIS - IAS	337
ADIS - IAS	338
ADIS - IAS	339
ADIS - IAS	340
ADIS - IAS	341
ADIS - IAS	342
ADIS - IAS	343
ADIS - IAS	344
ADIS - IAS	345
ADIS - IAS	346
ADIS - IAS	347
ADIS - IAS	348
ADIS - IAS	349
ADIS - IAS	350
ADIS - IAS	351
ADIS - IAS	352
ADIS - IAS	353
ADIS - IAS	354
ADIS - IAS	355
ADIS - IAS	356
ADIS - IAS	357
ADIS - IAS	358
ADIS - IAS	359
ADIS - IAS	360
ADIS - IAS	361
ADIS - IAS	362
ADIS - IAS	363
ADIS - IAS	364
ADIS - IAS	365
ADIS - IAS	366
ADIS - IAS	367
ADIS - IAS	368
ADIS - IAS	369
ADIS - IAS	370
ADIS - IAS	371
ADIS - IAS	372
ADIS - IAS	373
ADIS - IAS	374
ADIS - IAS	375
ADIS - IAS	376
ADIS - IAS	377
ADIS - IAS	378
ADIS - IAS	379
ADIS - IAS	380
ADIS - IAS	381
ADIS - IAS	382
ADIS - IAS	383
ADIS - IAS	384
ADIS - IAS	385
ADIS - IAS	386
ADIS - IAS	387
ADIS - IAS	388
ADIS - IAS	389
ADIS - IAS	390
ADIS - IAS	391
ADIS - IAS	392
ADIS - IAS	393
ADIS - IAS	394
ADIS - IAS	395
ADIS - IAS	396
ADIS - IAS	397
ADIS - IAS	398
ADIS - IAS	399
ADIS - IAS	400
ADIS - IAS	401
ADIS - IAS	402
ADIS - IAS	403
ADIS - IAS	404
ADIS - IAS	405
ADIS - IAS	406
ADIS - IAS	407
ADIS - IAS	408
ADIS - IAS	409
ADIS - IAS	410
ADIS - IAS	411
ADIS - IAS	412
ADIS - IAS	413
ADIS - IAS	414
ADIS - IAS	415
ADIS - IAS	416
ADIS - IAS	417
ADIS - IAS	418
ADIS - IAS	419
ADIS - IAS	420
ADIS - IAS	421
ADIS - IAS	422
ADIS - IAS	423
ADIS - IAS	424
ADIS - IAS	425
ADIS - IAS	426
ADIS - IAS	427
ADIS - IAS	428
ADIS - IAS	429
ADIS - IAS	430
ADIS - IAS	431
ADIS - IAS	432
ADIS - IAS	433
ADIS - IAS	434
ADIS - IAS	435
ADIS - IAS	436
ADIS - IAS	437
ADIS - IAS	438
ADIS - IAS	439
ADIS - IAS	440
ADIS - IAS	441
ADIS - IAS	442
ADIS - IAS	443
ADIS - IAS	444
ADIS - IAS	445
ADIS - IAS	446
ADIS - IAS	447
ADIS - IAS	448
ADIS - IAS	449
ADIS - IAS	450
ADIS - IAS	451
ADIS - IAS	452
ADIS - IAS	453
ADIS - IAS	454
ADIS - IAS	455
ADIS - IAS	456
ADIS - IAS	457
ADIS - IAS	458
ADIS - IAS	459
ADIS - IAS	460
ADIS - IAS	461
ADIS - IAS	462
ADIS - IAS	463
ADIS - IAS	464
ADIS - IAS	465
ADIS - IAS	466
ADIS - IAS	467
ADIS - IAS	468
ADIS - IAS	469
ADIS - IAS	470
ADIS - IAS	471
ADIS - IAS	472
ADIS - IAS	473
ADIS - IAS	474
ADIS - IAS	475
ADIS - IAS	476
ADIS - IAS	477
ADIS - IAS	478
ADIS - IAS	479
ADIS - IAS	480
ADIS - IAS	481
ADIS - IAS	482
ADIS - IAS	483
ADIS - IAS	484
ADIS - IAS	485
ADIS - IAS	486
ADIS - IAS	487
ADIS - IAS	488
ADIS - IAS	489
ADIS - IAS	490
ADIS - IAS	491
ADIS - IAS	492
ADIS - IAS	493
ADIS - IAS	494
ADIS - IAS	495
ADIS - IAS	496
ADIS - IAS	497
ADIS - IAS	498
ADIS - IAS	499
ADIS - IAS	500
ADIS - IAS	501
ADIS - IAS	502
ADIS - IAS	503
ADIS - IAS	504
ADIS - IAS	505
ADIS - IAS	506
ADIS - IAS	507
ADIS - IAS	508
ADIS - IAS	509
ADIS - IAS	510
ADIS - IAS	511
ADIS - IAS	512
ADIS - IAS	513
ADIS - IAS	514
ADIS - IAS	515
ADIS - IAS	516
ADIS - IAS	517
ADIS - IAS	518
ADIS - IAS	519
ADIS - IAS	520
ADIS - IAS	521
ADIS - IAS	522
ADIS - IAS	523
ADIS - IAS	524
ADIS - IAS	525
ADIS - IAS	526
ADIS - IAS	527
ADIS - IAS	528
ADIS - IAS	529
ADIS - IAS	530
ADIS - IAS	531
ADIS - IAS	532
ADIS - IAS	533
ADIS - IAS	534
ADIS - IAS	535
ADIS - IAS	536
ADIS - IAS	

**PILOT'S ASSOCIATE:
EVOLUTION OF A FUNCTIONAL PROTOTYPE**

Major Carl S. Lizza
Captain Sheila B. Banks
Lieutenant Michael A. Whelan

Wright Laboratory (WL/KTA)
Wright-Patterson Air Force Base, Ohio, USA
45433-6553

AD-P006 340

1. SUMMARY

The Pilot's Associate Program has completed its first phase of functional development which included two significant program milestones: Demonstration 2 and Demonstration 3. The early successful development of complex, knowledge-based decision-aiding systems allowed the program to shift focus to a more near-term, embedded-avionics application of the technology. This change in philosophy forced an evolution away from a program designed to explore artificial intelligence without processing bounds, towards a more realistic prototyping of functionality consistent with, and constrained by, the realities of an avionics processing architecture of an actual aircraft. This paper focuses one of the unique developmental approaches, the technical progress made through this phase of the program, and the accomplishments and disappointments with the design approach. Lastly, a brief look into the lessons learned as artificial intelligence technology was applied to a dynamic and complicated domain will be presented.

2. INTRODUCTION

The Pilot's Associate program is a joint Defense Advanced Research Projects Agency (DARPA)/U.S. Air Force program which began in February 1986 as an application demonstration for the DARPA Strategic Computing program. The program was organized into two phases with two demonstrations in Phase 1 (referred to as Demonstration 2 and Demonstration 3), and a final man-in-the-loop demonstration and evaluation, Demonstration 4, in Phase 2. The original program philosophy was to explore the potential of artificial intelligence to improve the decision-making ability of the pilot of an advanced, single-seat fighter. As an application of Strategic Computing, there was no limitation

on processing architecture. In fact, creating a technology pull for Strategic Computing by establishing new processor requirements was a fundamental desire. However, after the resounding success of Demonstration 2, both technical and operational, DARPA directed a shift in philosophy towards near-term avionics processors specific to an existing or planned aircraft. This shift away from the unbounded processing world of Strategic Computing forced associated shifts in functionality scoped by the bounds of limited avionics processors, both in capability and availability in the selected aircraft. The new philosophy was to use the remainder of Phase 1 to develop Pilot's Associate as a fully functional, nonreal-time prototype, and then to transition that system in Phase 2 into an avionics-consistent processing architecture for the real-time demonstration and evaluation. There would be no absolute requirement to use flight-worthy, military qualified processors in Phase 2 due to availability of some of the newer hardware and expense; the goal is consistency in architecture and performance.

The Pilot's Associate concept, Figure 1, somewhat conceptualizes another evolution of program philosophy during the first phase of the program. The original vision of artificial intelligence, and expert or knowledge-base systems in particular, has proven too narrow with respect to the desired behaviour of the Pilot's Associate. That is, rather than exploring the application of a technology, artificial intelligence in this case, the program has evolved into the application of whatever techniques apply to effect the desired intelligent behaviour ascribed to Pilot's Associate. This new philosophy is perhaps best described as an "intelligent systems" approach which makes no assumptions about underlying technologies. This may appear to be a subtle, semantic argument, but it truly reflects a major evolution

in program philosophy to be requirements driven rather than technology driven.

A more complete description of the original goals and program philosophy, and a description of the five major functional components can be found in [1, 2, 3]. Much of the following description of the early Phase 1 work through Demonstration 2X is drawn from [4].

There have been two unique approaches to system development explored independently by two

contractor teams in Phase 1. An article on the Lockheed Aeronautical Systems Corporation program will appear in the June 1991 issue of IEEE Expert. The remainder of this paper will provide some insight into the other effort as performed by the McDonnell Aircraft Company team. The reader is advised that the ensuing technical discussion of the program is written from the perspective of the McDonnell team effort and is intended to apply solely to that effort.

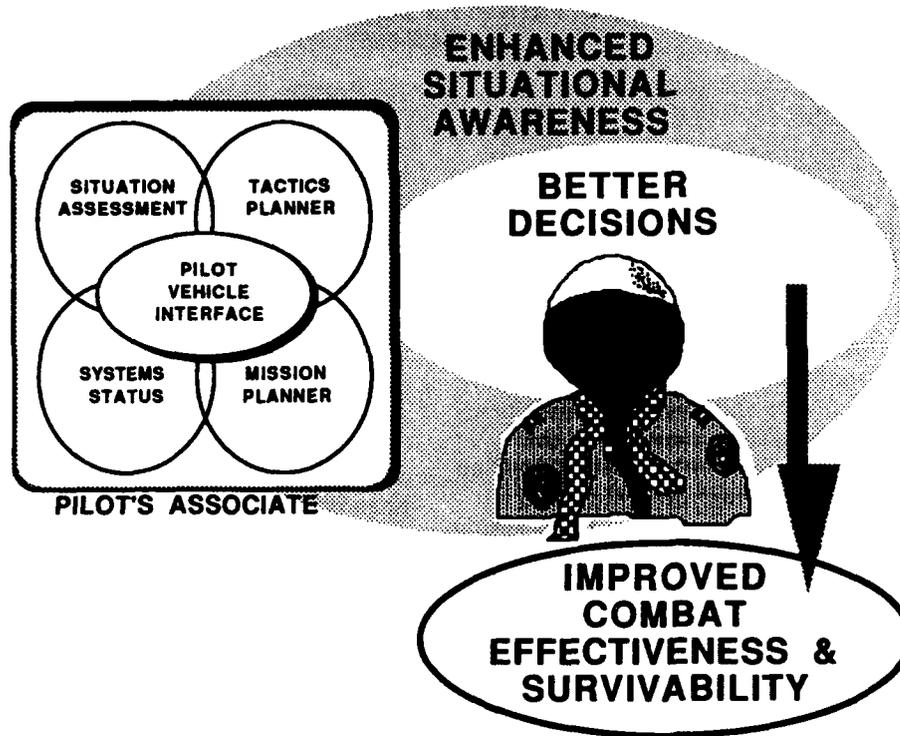


Figure 1.
Pilot's Associate Concept

3. DEMONSTRATION 2

The McDonnell team approach to a Pilot's Associate system focused on air-to-ground functionality. The original system design was consistent with the "technology pull" objective of Strategic Computing. The design was oriented towards a hypothetical, massively-parallel, shared-memory processor architecture as the most appropriate configuration for a real-time Pilot's Associate at the conclusion of Phase 2. This design decision was founded on studies which concluded that 70-90% of rule-based system operations were pattern-matching operations. This design philosophy resulted in a homogeneous, fine-grained rule structure used to

implement every aspect of the system, including some procedural code.

A major part of the design included the concept of a single, global blackboard facilitated by shared-memory. Since a requirement for the software implementation called for distributed, symbolic processors (six Explorers and a microVax), a technique was devised to allow the host language, ART¹, to run synchronously on each machine. To simulate the global blackboard, working memory changes had to be propagated throughout all of the subsystems. The flight simulation would execute one "tick", then the subsystems would execute their agenda and propagate

1 Automated Reasoning Tool, Inference Corporation

changes to the other subsystems, then the cycle would repeat. In addition to the overhead of this process, advanced features of ART, such as schemas and viewpoints, were exploited at further expense of processing time.

Work in this early development effort was centered on detailed design and specifications, with small prototypes dedicated to communications and control architecture. The first, major functional build and integration occurred as the prototype for Demonstration 2, and, for the first time, severe performance problems surfaced. In June 1988, Demonstration 2 occurred; however, only one of six original mission segments was supported by fully integrated software. The inability to complete a full demonstration was primarily caused by a run time of approximately 120:1 — 1 minute of simulation time resulted in two hours of processing time. The result of this extremely slow run time was an almost impossible development and testing environment.

3.1 SYSTEM PERFORMANCE ANALYSIS.

Following Demonstration 2, a serious, in-depth analysis of system performance was undertaken. The approach used in the system analysis was to perform both timing and impact analyses. The timing analysis provided quantitative data on the Demonstration 2 system performance. The impact analysis identified key features of the Demonstration 2 system design and implementation that influenced performance. From this analysis, design change recommendations were generated. A key factor in generating the design changes was the recognition that the goals of the program were changing from those of Demonstration 2, as described above. Although McDonnell was not to immediately address the redirection to a near-term aircraft, the knowledge of the changes did provide this early influence. Also influencing the design recommendations was the work of the real-time investigation effort, which had been directed to address near term, real-time implementation approaches for Pilot's Associate, discussed in a later section of this paper.

The result of the timing and impact analysis confirmed the almost painfully obvious conclusion: an approach towards large-scale parallelism forced expensive compromises in knowledge structures and near-term implementation. The following points further detail the poor run-time performance of the Demonstration 2 system:

1. The strong commitment to large scale parallelism was the primary factor in the decision to have a synchronous Pilot's Associate/Simulation and module operation, and in the adoption of the fine-grained rule structure. The choice of a synchronous architecture led to additional buffering and process waiting. This problem was compounded because the implementation of the synchronous architecture was not streamlined for run-time efficiency. Module synchronization was implemented by splitting the right-hand side of the ART function which lead to additional communications and buffering overhead.
2. The modeling of a massively-parallel environment required the use of a fine-grained rule structure leading to an increased rule base size, increased pattern and join net size, and increased redundancy in rules because of the restriction to have only one right-hand side action. In addition, sequential dependencies were represented explicitly in the left-hand sides of rules by control tokens.
3. The natural functional partitioning imposed by subsystem boundaries was maintained with some effort. The functional partitioning could have been streamlined for the particular hardware configuration. The partitioning led to processor overloads and an inefficient, multi-processing environment.
4. Hypothetical planning was implemented using the ART viewpoint construct which added overhead to schema processing extending the step time and increasing the memory requirements. The general approach to using ART data structures also contributed to increased memory and step time. These data structures included the exclusive use of schemas, viewpoints, and hierarchical data structures.

A revised system architecture was recommended as a result of this analysis. The new architecture abandoned fine-grained rules in favor of small-scale parallelism, coarse-grained rule structure, and control implemented by a priority structure and agenda management. For system implementation the recommendations included: eliminate the use of viewpoints; reduce the use of schema-based, hierarchical data structures; use facts and LISP structures in addition to schema; and make more use of procedural code. Modifying functional partitioning to reduce communications and using asynchronous

91-15522



91 15522 017

communications between Pilot's Associate and the flight simulation were also recommended.

3.2 DEMONSTRATION 2X

Demonstration 2X, was scheduled for February 1989 to address the rebuilding and recovery of the Phase 1 design and implementation.

Demonstration requirements included successful completion of the first three of the six Demonstration 2 air-to-ground mission segments with a performance goal of 10:1 real-time.

The process involved in Demonstration 2X development required re-building all of the communications and control structure, and recoding a major portion of the individual subsystems during only five months of development time. The revised architecture, with concentration in the functional areas of Mission Planner, Pilot-Vehicle Interface, and Situation Assessment, would be the emphasis of development for this demonstration.

The Demonstration 2X Pilot's Associate was hosted in both a laboratory simulation on a graphics workstation, and in a full cockpit within a 40-foot dome equipped with a high-fidelity, out-the-window visual system. The advanced simulation capability provided for qualitative evaluation of Pilot's Associate capabilities in a realistic combat environment as early as possible in the development process. Initial prototype software development, testing, and debugging took place in the laboratory. Then the software was ported to equivalent hardware in the dome for more complete testing and evaluation prior to the next prototype cycle.

Mission Scenario. The Demonstration 2X mission was a single-ship, air-to-ground battlefield air interdiction mission with a mobile column of tanks as the primary target. The mission events included forward edge of the battle area (FEBA) penetration, ingress to the target area at low level and high speed, target attack, and low level egress through threat defenses. The Pilot's Associate aircraft was modeled using F-15 capabilities with advanced air-to-ground systems.

The mission begins just prior to crossing the FEBA when the Pilot's Associate performs the "fence check" to configure weapons and sensors for the mission. Nominal altitude for the mission is 200 feet above ground level with an airspeed of 450 knots. Soon after FEBA penetration, an unknown surface-to-air missile

(SAM) site search radar "pops-up" which impacts the planned route but at sufficient distance to allow Pilot's Associate to create a new route to avoid the new threat. After successfully avoiding the SAM threat, and guided by the Pilot's Associate mission plan to minimize additional SAM exposure, a target redirect message is received over data link to attack an alternate target location of tanks. Pilot's Associate creates a low-altitude, terrain-following route plan to the new target. The plan minimizes exposure to the known missile sites by exploiting terrain to mask the Pilot's Associate aircraft. As the pilot closes in on the target, the Pilot's Associate configures sensors and displays consistent with the requirements for target acquisition by selecting: the High Resolution Map mode of the air-to-ground radar at long range; the Forward Looking Infrared Receiver at short range; and for target attack using the sensor on-board the Maverick missile. Immediately following weapons launch, another unknown SAM appears within a threatening radius of the aircraft. Pilot's Associate develops a short-term route plan to quickly get the aircraft out of the area of SAM exposure. While still within the lethal radius of the SAM site, the pilot exploits the large, heads-down display of the terrain superimposed with the SAM range and line-of-sight information to help maintain an awareness of possible exposure and particularly dangerous areas. Pilot's Associate highlights the most critical SAM site and includes a dynamic, "within parameters" shot indication to advise the pilot of critical exposure. After the threatening SAM is successfully evaded, Pilot's Associate safely guides mission egress through areas of high threat defenses.

Technical Overview. Using the results and recommendations from the Demonstration 2 system performance analysis, the entire approach to air-to-ground functionality took on a new perspective. The rule structure was completely changed into a knowledge source format consistent with a design for real-time processing. The knowledge sources were written in a generic form and precompiled into ART rules for execution. The system used a distributed blackboard which consisted of LISP variables and ART working memory, as shown in Figure 2. To achieve improved run-time performance, the use of ART was tailored, and inefficient data structures (schema and viewpoints) were avoided. The subsystems updated the global blackboard asynchronously and communications between local blackboards were expressed as system goals. Global system goals were established by problem events, such as

receiving a mission target redirection, or by inference of pilot goals from cockpit actions. The system goals were managed by a System Executive (SE), an additional subsystem in this Pilot's Associate implementation. SE contained knowledge sources to recognize system problem events, established goals to deal with the events, and monitored for their success or failure.

The full-cockpit, high-fidelity demonstration of the Demonstration 2X system executed in real-time, significantly exceeding the performance goal. This system performance was attributed to factors which lessen the validity of the claim to real-time execution. The Demonstration 2X Pilot's Associate system was a functional skeleton; the breadth and depth necessary for

system robustness was severely lacking in many areas. A major influence on performance was the method of determining the search space for route planning purposes. The danger maps used by the route planner were computed off-line prior to the simulation; a process which required several minutes to complete. The danger map algorithm was analogous to the cost matrix computation in the dynamic programming route planner. If on-line, these computations would have been a significant performance drain. A more difficult threat environment could also cause the A* route planner responsiveness to degrade significantly. Therefore, the only credible claim is that the performance improvement from the Demonstration 2 system was impressive.

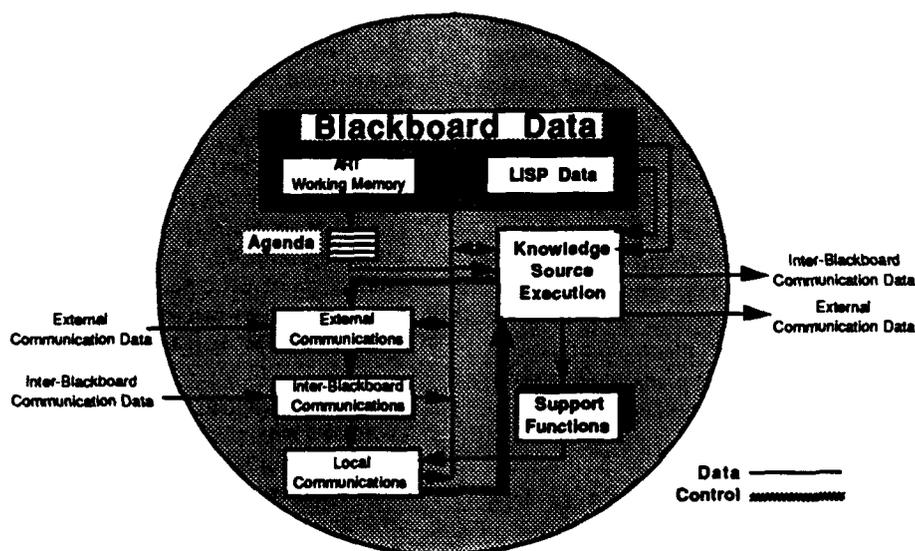


Figure 2.
Demonstration 2X Subsystem Internal Architecture

Pilot-Vehicle Interface (PVI) experienced a major redesign during the Demonstration 2X development period. Elements included in the PVI design were display management, a pilot awareness model, pilot intent, and a task scheduling and workload model. Most effort concentrated on the display management function which used scripts to establish display priorities. Although a full complement of design elements were included in the Demonstration 2X PVI design, most functions were implemented only in skeletal form.

Situation Assessment (SA) consisted of two support functions as resources to perform assessments for other Pilot's Associate subsystems. The first support resource, a danger map algorithm, defined the search space for the route planning algorithm of the Mission

Planner. The danger map, a 2-dimensional map fixed at a 200 foot altitude, was a composite of known threat lethality and intervisibility information which provided a measure of "danger" associated with each point on the map. Because the danger map was computer off-line for Demonstration 2X, SA simply maintained an interface to the danger map. The other support resource, a spatial data base, quickly computed the intersection of objects, such as route paths and threat regions, to flag threatening segments of the route plan when replanning was required. SA also added value to sensor data on threats to the Pilot's Associate aircraft by inferring threat status, such as search or track; threat potential, such as time to intercept envelope and earliest shot indication; and threat priority. SA considered only ground threats in the assessments.

Mission Planner and Tactics Planner (MTP) functions were combined, although, a response decision to the pop-up SAM event was the only tactics planning function implemented. This decision amounted to a choice between a long-term avoid response and a short-term evade response, and was driven primarily by distance from the threatening SAM site. The choice to avoid or evade the new threat then drove mission planning activities for countering the threat.

To develop route plans, the mission planner subfunction used a two-level, A* heuristic search over the danger map. First, a waypoint planner performed a coarse search of the the danger map search space and selected appropriate waypoints for the route plan. This planner also selected points for the high resolution map acquisition of the target and the initial point for target attack. A fine-grained planner then developed routes between waypoints. As a post-planning process, a resource planner allocated fuel and aircraft speed to the route plan to maintain a fixed time-on-target. As the aircraft is flying the route plan, a plan conformance monitor evaluated fuel levels, speed of flight, and the path corridor definitions to detect violations with potentially adverse mission impact. The framework for resource- or mission-level replanning existed to remediate violations which affected mission effectiveness or survivability, however, this functionality was not implemented by the demonstration.

In response to the SAM avoid or evade recommendations developed by the tactics planner subfunction, the mission planner either planned a mission level response or a tactical level response, respectively. The mission level response replanned ingress, attack and egress portions of the mission as required using the waypoint planner, route planner, and resource planner. The tactical level response replanned only the affected segments of the mission plan, as identified by Situation Assessment, using the route planner and the resource planner.

The A* route planner was implemented in LISP, with all other code in MTP in the previously mentioned knowledge source format.

System Status was practically nonexistent at Demonstration 2X. The only responsibility attributed to this function was fuel monitoring and reporting of fuel status to PVI and MTP.

3.3 DEMONSTRATION 3

A period of development time lapsed between the Pilot's Associate Demonstration 2X milestone and the beginning of effort on the Phase 1 redirection to apply Pilot's Associate technology to a current or planned service aircraft. The aircraft platform chosen for the air-to-ground effort was an advanced Navy F/A-18. Because of the time lapse of development effort and the extensive analysis accomplished on the Demonstration 2 system, Demonstration 3 was not scheduled until Oct 1990.

The Demonstration 3 Pilot's Associate used both the laboratory simulation on a graphics workstation, and the full cockpit with in a 40-foot dome equipped with a high-fidelity, out-the-window visual system as in Demonstration 2X. Both the laboratory simulation and the actual dome used have changed, but the dual-site approach to cyclic software development, testing, and evaluation proceeded as before. The Pilot's Associate system development continued the use of LISP, the knowledge source format, and symbolic processors due to the flexibilities of the language and the efficient development environment provided. Some of the support functions were coded in other programming languages for run-time efficiency.

Mission Scenario. The Demonstration 3 scenario was considerably more flexible than that of Demonstration 2X. Advances in both the laboratory and flight simulation allowed rapid changes in the environments to test Pilot's Associate capabilities under varying conditions. Two basic air-to-ground missions were defined for Demonstration 3: a two-ship Day/Night Strike mission, and a two-ship Battlefield Air Interdiction mission. The visible difference between mission types was either a tactical (e.g. a marshalling tank column) or a strategic target (e.g. fuels storage area). Other selectable mission variables included threats, aircraft stores, and weather. Demonstration 3 concentrated on the target acquisition and attack phase of the chosen mission. Mission events included air threats, SAM threats and launches, sensor use and failures, target identification and attack, and two-ship rejoin.

Technical Overview. Progress toward Demonstration 3 occurred rapidly. Again, with the luxury of the time lapse and the extensive analysis of system parameters required to achieve real-time operation, the development

effort refined the goals and approach to incorporate new ideas.

The basic structure of the blackboard architecture remained from the Demonstration 2X system with several enhancements implemented to make the blackboard more useful for the Demonstration 3 system. The blackboard enhancements included modification of the knowledge source syntax to allow specification for urgency and importance, and modification of the ART agenda to allow heuristic scheduling. These important features allowed for real-time control and were essential aspects in Pilot's Associate realization of the Demonstration 3 run-time goals.

Along with the real-time distributed-blackboard architecture, the primary component of the system software framework was the *task network* concept. The task network is used to model Pilot's Associate and pilot activity, and the activity of external agents. This structure allowed the PVI to reason about pilot actions and the pilot/Pilot's Associate interactions. The network also provided a mechanism for communicating and responding to exceptional circumstances that may arise in the dynamic, tactical, combat environment. Therefore, the network provided a platform for integrating the monitoring activities of the assessors with the planning activities, and the cockpit management provided by the pilot-vehicle

interface. Also, this framework provided a consistent software foundation for integrating the varied domain knowledge.

As models of system activity, task network nodes could be further decomposed into more specific tasks or represented as primitive tasks. Primitive tasks referred to either Pilot's Associate or pilot actions. Domain knowledge, associated with a task, included methods for deciding when the task is appropriate, for implementing the task, for monitoring of task execution and dependencies, for handling exceptions, and generating scheduling parameters such as importance and urgency for a specific task. Pilot's Associate plans were presented as partially ordered graphs of tasks. Partial ordering in this context referred to the ability to represent concurrent tasks with no single ordering constraint. The ability to represent a task as a set of less abstract tasks led to a hierarchical representation. This task hierarchy allowed exception handling (control knowledge) to be applied locally or by higher task abstraction level. Exceptions were events — determined from the monitoring of aircraft resources, pilot actions, the external environment, and the operation of other Pilot's Associate tasks — that could significantly effect current or planned tasks. Exceptions and exception handling provided the method to adapt the task network to the dynamic environment.

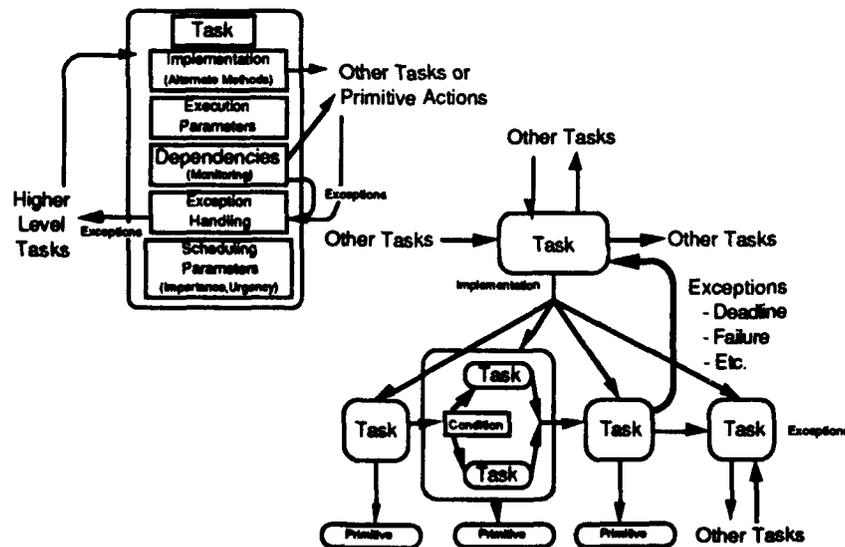


Figure 3.
Demonstration 3 Generic Task and Plan Representation

Pilot's Associate task scheduling and execution were performed by an underlying software layer

that integrated the Pilot's Associate blackboard architecture with a real-time operating system.

The task network and the blackboard architecture worked in concert to assure timely execution of primitive tasks. The task network communicated with the blackboard by asserting tokens when tasks were activated. These tokens triggered knowledge sources which performed the following: monitoring for task completion or exceptions, calculating pilot information requirements, and performing actions which have been authorized for automation. This task network/blackboard interface was a fundamental component of the Demonstration 3 software architecture.

The Demonstration 3 Pilot's Associate still lacked the functionality depth required for a fully robust system, and any claims of real-time execution must continue to be evaluated on this basis. However, the real-time architecture developed for this prototype established a strong framework for further functional integration in Phase 2 development.

Pilot-Vehicle Interface (PVI) functional objectives — to support pilot decision-making, to present information intelligently, and to assure alignment of Pilot's Associate and pilot goals — continued to be the driving force for the design. This design, however, changed from the Demonstration 2X implementation and relied on the task network software architecture. The PVI design linked active tasks from the task network to pilot information requirements (PIRs) by the pilot information requirements generator. This function was derived from a mission decomposition in which pilots detailed their mission activity and the information required to perform this activity. The pilot information requirements were used as input into the cockpit display manager which chose, scheduled, and then executed the display agenda. The cockpit display manager selected a list of candidate display elements based on the PIRs, mission context, and the pilot workload assessment function. The displays were scheduled by priority and pilot explicit selection. If the PIRs were satisfied by a current display, they were marked as such. If the display of choice was occupied by a higher priority requirement, an alternate choice was made until all PIRs were satisfied by display scheduling. The PVI then made use of a pilot task and goal monitor which assessed pilot actions and active tasks to determine if the active tasks were appropriate, or if exceptions to the current planning assumptions had occurred. The design also included pilot awareness assessor and error monitor functions, but these functions were not implemented in the demonstration system.

Situation Assessment (SA) retained the functionality of Demonstration 2X, but changed implementation to reflect real-time analysis efforts and interactions with the task network. SA monitored for tactical objects, such as threats, targets, support forces, and environmental factors which had possible impact on the current, active plan in the task network. Because of the uncertain, current state and the unpredictable, future state of a dynamic environment, certain assumptions on the current state of the world were made for planning purposes. If those assumptions were violated by tactical object changes, SA generated notice, via exceptions, that the planning assumptions were no longer valid and alternate planning must be considered. SA also coordinated information requests and usage constraints and formulated a sensor request plan for a sensor manager external to Pilot's Associate. SA continued to support functional interfaces to the spatial data base and danger map. However, unlike the Demonstration 2X system, the Demonstration 3 system updated the danger map on-line. This made the subsystem a true assessor, able to: assess changes in the outside environment and how they relate to other Pilot's Associate activity; update the Pilot's Associate model of the environment to reflect changes; and produce an accurate, timely danger map for the pilot and mission planner.

Mission Planner/Tactical Planner (MTP) had one Demonstration 3 functional goal: to represent a robust mission and target attack plan. To accomplish this goal, MTP continually monitored the mission and target attack plan for effectiveness with respect to changes in the aircraft capabilities and environment. Since the Demonstration 3 mission concentrated specifically on the target attack segment of an air-to-ground mission, replan only considered the target attack segment and a global mission replan was not required.

MTP design included three major functions: a plan monitor, plan execution function, and a plan generator. The plan monitor architecture derived its structure from the task network. The task network allowed detection of assumption violations associated with the current planning, and prediction of the impact of the assumption violations on the current plan steps. Although the detection method for planning violations was altered, the plan parameter monitor software remained mostly unchanged. The plan executor architecture, enhanced by task network support, continued to use the existing route plan execution software. The plan generator function was

supported by the task network architecture and consisted of four controlling functions: a strategy selection function, route planner, target attack planner, and countermeasure planner. The strategy selection function was performed using planning scripts and the task network planning structure. The planning scripts used declarative representations of the subgoals that must be achieved to satisfy the system goal which initiated the planning process. The task network provided an alternate representation of planning strategies which allowed a more general, but limited number of strategies to be considered for planning. The new target attack planner consisted of attack templates which defined stereotypical delivery representations and cooperative tactics. Each template was adapted to the current mission situation as planning occurred. This approach defined a limited target attack modification search space and allowed rapid comparisons for selecting alternative plans due to situation contingencies. The countermeasure planner identified threat suppression plans which mainly consisted of executing the appropriate maneuvers to establish optimal geometry with respect to the threat. Each threat suppression plan was represented as a flight plan consisting of a set of maneuver segments with subtasks inserted at appropriate places in the plan indicating where to apply countermeasures, such as jammers and deployment of expendables. Plan selection dependence factors included threat information, timing constraints, location to target area, environmental constraints provided by Situation Assessment, and aircraft constraints provided by Systems Status.

System Status (SS) increased tremendously in scope from Demonstration 2X, but remained a skeleton of what was envisioned in the original plans. The role of SS was to provide continual assessment of the aircraft's capability, and to assist in remediation planning to remedy system failures and maintain aircraft capability. The implementation of this role was somewhat less grand, consisting of rather simple capability models of aircraft resources: navigation system, fuel, flight dynamics, sensors, weapons, signature, and countermeasures. Since mission and tactical plans required use of specific aircraft resources, SS maintained the availability status of each resource. SS also had parameters defined for each model to determine if the resource was viable or failed. The failed status was

communicated to the system through plan assumption violations, via exceptions, similar to Situation Assessment.

System Executive (SE) played an important role in the system design for this effort. The functional requirements for SE were defined as follows: coordinate Pilot's Associate problem solving; manage system-wide timeliness; maintain consistency between pilot and Pilot's Associate problem solving goals; and support run-time resource management. These requirements were blended in an elegant design in which SE was viewed as a functional requirement, not an added component. The design involved both a centralized component (central executive) and a module component (embodied by the task network). The central executive was the effort covered solely under SE efforts.

The System Executive maintained a model of system activity. Whenever exceptions were generated by the subsystems, the task network hierarchy allowed local exception handling, or, depending on the exception, propagation of the exception to a higher abstraction level. If the exception was propagated to the highest level in the task network, this constituted violation of a system level goal and SE central control assumed responsibility for system response to account for the exception. These response plans included subsystem tasks to realign the Pilot's Associate goals with the current state. The tasks contained subsystem parameters and were SE's primary means for affecting work in progress in the subsystems. These parameters included deadlines, solution granularity (completeness), importance, status reporting rate, and assessment focus. SE modified a parameter, such as deadline or importance, in order to affect the solution associated with a task. Problem solving goals and strategies, response plans, and subsystem parameters and status were all posted and coordinated on the blackboard. All plan coordination knowledge and execution were implemented as knowledge sources. The SE implementation considered all requirements (except for the support of system-level resource management which should use models of subsystem resources), and their relationships and capabilities. and compared the models to run-time resource states to determine control decisions. Support of system-level resource management was not planned until after the Demonstration 3 milestone.

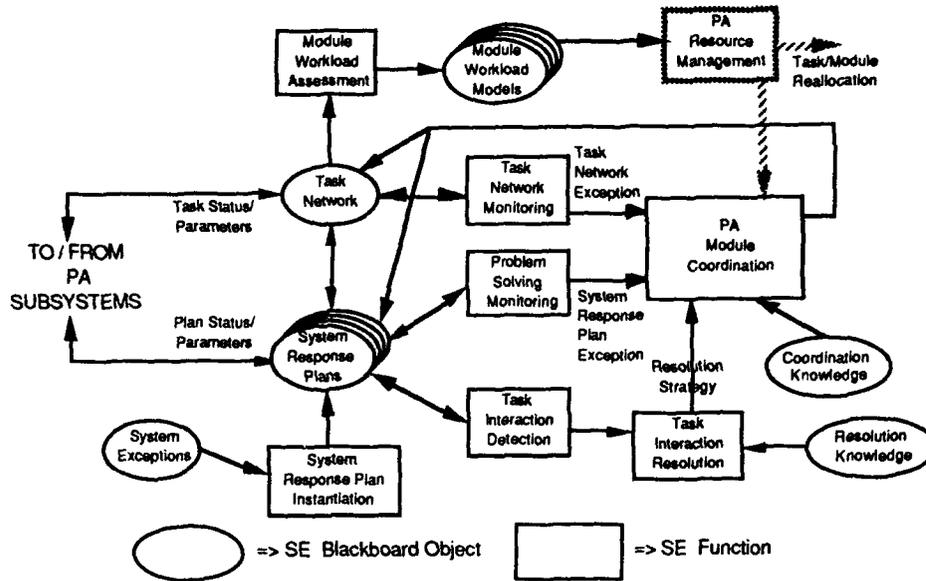


Figure 4.
Demonstration 3 System Executive Functional Design

Real-Time Investigation Efforts. The real-time performance analysis of this Pilot's Associate approach consisted of the following tasks: an assessment of the Pilot's Associate real-time risks; an assessment of technologies to address these risks; and the selection of prototypes to demonstrate the selected technologies. Throughout the performance of these tasks, the analysis assumed that real-time performance in a time-stressed system required three system characteristics: execution speed to guarantee delivery of results within the allowed time; system responsiveness to dynamic task demands by undertaking new tasks or shifting resources from the current tasks to higher priority ones; and graceful adaptation when workload exceeds processing capacity, ensuring critical tasks are completed. Speed of execution could be obtained from increasing the number and power of processors as well as from efficient algorithms. However, it was reasonable to assume hardware and software advancements would not meet the raw speed required for full real-time operation. Accordingly, the approach and conclusions emphasized obtaining system responsiveness and graceful adaptation to achieve intelligent and reactive system behavior.

The assessment of the Pilot's Associate real-time risks coincided with the Demonstration 2 system development phase and identified bottlenecks in that design approach. These bottlenecks included: lack of responsiveness to pilot or external events; inability to keep up with data

glut and task overloading; inability to respond to dynamic changes in workload due to mission contingencies; and inability of Pilot's Associate modules to send and receive messages asynchronously. Once identified, these real-time risks were then used to guide the development of prototype concepts for later implementation in the project demonstrations.

The technology assessment activity was approached by the following tasks:

1. Investigate technology innovations.
2. Investigate alternative hardware and software architectures.
3. Map relevant solution candidates to the proposed architectures.
4. Evaluate solution candidates against the technical criteria.
5. Select the most appropriate hardware and software model, and the most significant solution candidates for this model.

Three models of a system architecture were considered as candidates for Pilot's Associate. Evaluations of these architectures resulted in selection of medium-scale parallelism with asynchronous execution as the target architecture — this was consistent with the next generation advanced aircraft architectures or projected upgrades. Fourteen software innovations were identified to reduce the risk of real-time operation by providing more system responsiveness. Each innovation was assessed

for relevance against each of the three hardware architecture models and analyzed. Although every innovation was potentially a viable solution candidate for development, the study resulted in the selection of control reasoning and focus of attention. Control reasoning was defined as using knowledge about task demands, time constraints, goals, and system resources to select methods and formulate schedules. Focus of attention meant using designs that permit quick and efficient shift of system attention.

The prototype selection activity involved comprehensive analysis and matching of the technical innovations suggested by the hardware and software investigation task, to the real-time risk areas indicated by the bottlenecks identification task. The prototypes that were selected emphasized the utility of control reasoning to enhance timeliness and graceful adaptation, and focus of attention to provide real-time responsiveness and graceful adaptation under severe time stress. The choice of control reasoning techniques as a focus for the demonstrations was influenced by the desire to move the Pilot's Associate effort toward implementation in the generation of advanced aircraft now being designed.

The first real-time prototype featured an event-driven architecture, asynchronous event handling, and the use of control reasoning, focus of attention, and an execution margin. The focus of attention aspect was implemented by multiple event channels for knowledge source execution and a top-level control cycle designed to process event channels in order of priority. Control reasoning dynamically assigned importance and urgency ratings for each knowledge source; a weighted combination of ratings based on a dynamic scheduler goal, and formulation of an agenda of knowledge sources based on priorities. This demonstration provided features for tuning responsiveness and timeliness. What the demonstration did not provide was a well-defined framework for structuring problem solving in the Pilot's Associate domain.

The second real-time prototype was to demonstrate timely and responsive performance against a representative scenario and to bring the developed real-time concepts and designs closer to the Pilot's Associate system design. This demonstration featured three scenario situations, each requiring different responses in terms of system-level timing and activity. With the focus of attention structure retained from the

first prototype, control reasoning was implemented to select knowledge sources during the top-level cycle and allocate time to the knowledge sources to meet the deadlines. The theme of this successful demonstration was to show intelligent reactivity to the demands of varying events. Intelligent reactivity meant that the system had an awareness of deadlines and the ability to produce appropriate results — producing the best response given sufficient time, and producing an acceptable response given inadequate time.

The results and concepts of the real-time investigations were readily applicable to the evolving definition of the Pilot's Associate system. These ideas played a vital role in the definition of the real-time distributed blackboard and also in the definition of a real-time Pilot's Associate.

4. CONCLUSION

The Pilot's Associate program has successfully completed the first phase of development and demonstration of an intelligent system in an extremely complex and demanding domain. The promise of operational utility of a Pilot's Associate by improving the combat decision-making capability of the pilot of an advanced fighter is more evident as the program moves into the second phase of development. As development continues, the program objectives shift from system functionality development to the real-time implementation, demonstration, and experimental evaluation of the Pilot's Associate. The technical advances, and lessons, from the first phase of development have provided a firm baseline for the real-time software engineering tasks ahead.

5. REFERENCES

1. Holmes, J., Retelle, J., "The Pilot's Associate: Exploiting the Intelligent Advantage", AGARD 51st Guidance and Control Symposium, Madrid, Spain, September 1990
2. Morishige, R.I. and Retelle, J.P., "Air Combat and Artificial Intelligence," *Air Force Magazine*, October 1985
3. Retelle, J.P., "The Pilot's Associate - An Aerospace Application of Artificial Intelligence," *Signal*, June 1986
4. Lizza, C., "Pilot's Associate: A Perspective on Demonstration 2", Proceedings of the Computers in Aerospace Conference, October 1989.

Discussion

1. E. Gliatti, United States

How do you maintain good documentation in rapid prototyping activity?

Author:

Document each prototype cycle with specifications, scenarios, etc. beforehand then a progress report with performance measures and lessons learned is prepared after the prototype. As built design and specification document can be prepared at the end of the effort. The prototype reports become an important asset having captured the process history.

2. G.H. Hunt, United Kingdom

To achieve the best synergy between the intelligence of the pilot and the intelligence imbedded in the Pilot's Associate presumably requires a very effective man-machine interface. Are the displays currently used in the program adequate for this purpose?

Author:

The current displays effort is adequate only to support the test and demonstration in the program. A concerted effort will likely be needed towards intuitive displays as well as the psychology of interaction between the pilot and an intelligent machine.

3. D. Bosman, Netherlands

Now having two intelligent systems (human and technical), both systems could assess the performance and health of each other. The system could warn the pilot for danger potential in his decisions, off-track reasoning, etc. Do you plan to incorporate such capability in the Pilot Associate?

Author:

The foundation for PA monitoring the performance of the pilot exists in the current model. Only rudimentary pilot workload or resource modelling is in the current system, but the ability to use such information is better defined. The availability of pilot state sensors for GLOC or spatial disorientation or workload only enriches the current ability to effectively manage displays and workload. However, this entire area of monitoring and control is a very "pilot-sensitive" issue.

4. R. Guiot, France

What about pilot creativity?

Author:

The overriding program philosophy, the pilot is in charge, demands that the PA does not obstruct or stifle creativity on the part of the pilot. Especially in the area of offensive tactics, the role of PA is to provide support without forcing him to lead the computer. When lost, the computer must "understand" its role to continue providing support as possible and to remain quiet and non-intrusive until it has caught up with the situation.

5. G. Bourassa, Canada

- (a) Why are you moving the software from LISP to C++ before going to Ada?
- (b) When will the comparative evaluation of C and Ada be complete?
- (c) Will the results be published?
- (d) What is the target hardware?

Author:

- (a) Because of a lack of maturity in the distributed Ada run time operating system.
- (b) At the Phase 2 milestone in Mar 92.
- (c) Yes in the final report to be published in DTIC.
- (d) MIPS R3000 RISC processors.



COMPONENT PART NOTICE

THIS PAPER IS A COMPONENT PART OF THE FOLLOWING COMPILATION REPORT:

(TITLE): Conference Proceedings of machine Intelligence
for Aerospace Electronic Systems Held in Lisbon, Portugal on
13-16 May 1991 (L'Intelligence Artificielle dans les Systemes
Electroniques Aeroespaciaux)

(SOURCE): Advisory Group for
Aeronautical Research and Development, Neuilly-sur-Seine
(France)

TO ORDER THE COMPLETE COMPILATION REPORT USE AD-A242025

THE COMPONENT PART IS PROVIDED HERE TO ALLOW USERS ACCESS TO INDIVIDUALLY AUTHORED SECTIONS OF PROCEEDINGS, ANNALS, SYMPOSIA, ETC. HOWEVER, THE COMPONENT SHOULD BE CONSIDERED WITHIN THE CONTEXT OF THE OVERALL COMPILATION REPORT AND NOT AS A STAND-ALONE TECHNICAL REPORT.

THE FOLLOWING COMPONENT PART NUMBERS COMPRISE THE COMPILATION REPORT:

AD#:

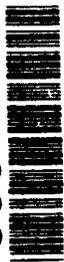
TITLE:

AD-P006 341

DTIC
ELECTE
NOV 13 1991
S D

Accession For	
NTIS CRA&I	<input checked="" type="checkbox"/>
DTIC TAB	<input type="checkbox"/>
Unannounced	<input type="checkbox"/>
Justification	
By	
Distribution	
Availability Codes	
Dist	Availability For Special
A-1	21

This document has been approved for public release and sale; its distribution is unlimited.



DEVELOPMENT OF TACTICAL DECISION AIDS

W. G. Semple
British Aerospace (Military Aircraft) Ltd
Warton Aerodrome
Preston PR4 1AX
UK

1 SUMMARY

Increasing complexity of sensors, weapons and counter-measures in combat aircraft is introducing a requirement for computer aids to the crew both for data manipulation and in more 'intelligent' tasks which have hitherto been considered the province of man rather than machine.

British Aerospace is pursuing a programme of Research and Development of such computer aids - Mission Management Aids - to assist the pilot in the management of combat missions.

As well as algorithms of conventional type, taking advantage of the enormous and apparently continuing advances in processing hardware, many such aids will need to exploit new concepts from software engineering and artificial intelligence.

We have these new tools by which we expect that we can meet the new requirements. However, many of the 'conventional' solutions are new to our experience, at least within the stringent requirements of real-time airborne computing, and many of the new tools are only partly developed or partly understood. This is especially true for some Tactical MMAs.

To achieve continuous useful output from R & D effort towards diverse functional requirements, using diverse and in some cases immature techniques, requires careful attention not only to the functional decomposition of the MMA family but to the skill and technology base from which it is constructed.

2 INTRODUCTION

Combat aircraft are being equipped with increasingly complex sensors, counter-measures and weapons, and are being faced with hostile aircraft, missiles and ground installations which are likewise more complex. At the same time, there is strong economic pressure to reduce the crew in number: consolidation of single-seat as the norm for air-to-air operation and an increasing requirement for single-seat operation in air-to-ground operations, both battlefield support and even deep strike.

A need for computer aids to the crew arises from two directions.

- 1) To obtain maximum performance from these increasingly complex systems involves supporting tasks more suited to machines than to the crew, such as calculation of missile engagement parameters, electro-magnetic signature recognition and numerical data fusion.
- 2) The increasingly complex control and integration of these systems together with the constraints on the numbers of the crew is leading (and in some cases, has already led) to excessive crew workload, with consequent loss of effectiveness. There is therefore a need for the computer to assist, or even to deputise for, the human in some of the tasks normally considered the province of man rather than machine, such as aircraft systems monitoring, sensor scan and mode control, and tactical decision making.

Such computer aids, which assist the pilot to manage the mission, have come to be known as Mission Management Aids (MMAs). They are required in order to achieve and to maximise mission effectiveness and vehicle survival.

The advances in computing technology, both in hardware and in the software techniques to which the hardware can be applied, already allow a degree of airborne computing which

would have been unthinkable when today's service aircraft were being designed. These advances are available for the combat aircraft being designed now, and it is clear that the advance has not yet stopped: the capability which the next generation of airborne design may achieve is indicated by that of ground equipment available now.

We have therefore a set of problems to resolve, and we have new and powerful hardware tools and rapidly evolving software tools and techniques, with which to try to solve them, wholly or partly. We do not, however, have the solutions. There are many difficulties and constraints:

- i) some of the problems have never been solved before, so that we have neither the solution, nor a practiced technique for finding the solution;
- ii) some of the problems have been solved, or are easily solvable, on large ground machines or in non-real time, but have not yet been solved for real-time operation in a (small) aircraft;
- iii) some of the problems are believed, or suspected, to be insolvable within acceptable timescales/equipment/budget/technology, but may usefully be part-solved;
- iv) the nature of the solution is not always clear - this can be true of the practical solution even if not of the ideal - so that while we know the problem which we wish to solve, and we believe that technology will allow a solution, we may not know exactly where the solution lies. We will recognise it when we find it, but we can not expect that we will always be looking in precisely the right place.

Problems like these may be the stuff of life to research and development engineers, but are not so welcome to businesses faced with an R & D requirement which is both long-term and uncertain. These considerations colour the whole approach to the planning and direction of MMA research, and do so in a way which conditions not only the direction of the research but also the way in which the engineer must carry it out.

This paper reviews the work done to date by British Aerospace (Military Aircraft) Limited from the standpoint of the above considerations. Drawing on experience to date, it considers the MMA family by function, by technology and by timescale to maturity, and highlights some of the questions which must be addressed by project customers and engineers.

In particular, the paper considers MMAs which assist the pilot with tactical planning, both for attack and defence. Such MMAs are currently envisaged as being informative or advisory rather than autonomous in action, and are known as Tactical Decision Aids (TDAs).

3 CLASSIFICATION OF MMAs

MMAs can be classified by mission function, by the type of service rendered, or by the class of enabling technology (that is, by the individual or group skills or disciplines required to realise the MMA).

3.1 Classification by Function

Some of the functions which MMAs could provide are primary functions directed at the mission objectives, such as attack planning; others are supporting functions which are necessary to achieve the objectives of the mission but which are not sought in their own right. An example of the latter class might

be defensive route planning; self-defence is often necessary to achieve attack or survival, but combat aircraft are not purchased and operated for the purpose of defending themselves.

For the purposes of this paper, if both primary and supporting functions are required in order to achieve the mission objectives in the expected combat environment, then both classes are considered as mission functions. Some supporting crew-aiding functions which are not specific to the combat mission role, such as general utilities management, can also be admitted, if they are modified by combat mission considerations or have common enabling techniques.

The degree of need for an MMA will vary from function to function, depending on the function itself and the suitability of the human to carry it out, on the scenario (its density, typical timescale and technology level), and on the numbers of crew available (perhaps dictated by other factors such as refit and re-use of existing aircraft, or a role in which there is intense but short-duration activity). Accepting for the present that some functions need not be provided by MMAs, we can identify a set of functions which can be provided by MMAs, that is, functions which could be given to a competent machine.

- Navigation;
 - (re)routing for waypoint/target timing;
- Systems Management:
 - Sensors (with tactical and fusion awareness),
 - Utilities (communications, engines, fuel);
- Situation Awareness,
 - sensor measurements,
 - sensor fusion (state and attribute measurements),
 - attribute inference,
 - behavioural inference,
 - tactical evaluation and prioritisation.
- Attack Planning and Execution:
 - air-to-air,
 - air-to-ground, fixed target - (re)routing only,
 - air-to-ground, mobile target - attack steering,
 - air-to-ship - (re)routing and attack steering,
 - anti-submarine;
- Defence Planning and Execution:
 - air-to-air - aspect and energy, decoys,
 - surface-to-air - (re)routing, screening, decoys;

It is perhaps at this point that we should consider exactly what is and what is not an MMA, as opposed to a conventionally engineered automation or software function. Territory can be identified which clearly belongs or which clearly does not: a computer programme capable of anticipating enemy tactics, devising effective counter plans and operating a control system to cause the vehicle to carry them out, clearly qualifies; a conventional autopilot almost as clearly does not. Where exactly between the two the boundary lies is a matter for debate and of arbitrary definition.

An incomplete definition, but which is sufficient for the present discussion, would be to define an MMA as a computer-based assistant to the pilot which makes use of 'artificial intelligence' (we will beg the definition of that!) or unusually ingenious processing or processing requiring task-specific hardware, to carry out one or more of the following tasks in a mission-specific sense: manage the aircraft, monitor its status, manage the sensors and/or assess sensor data, advise or take a course of action. Thus data fusion is admitted if it includes tactical awareness, but is excluded if it comprises only a Kalman filter, however advanced. Aircraft management is admitted if it demands tactical flexibility within the mission framework, or a qualitatively similar type of flexibility.

This paper will concentrate on MMAs with a strong tactical content, in terms of either awareness or planning ability, since it is these which are most demanding in terms of enabling technology and most restrictive, as yet, in terms of the service level

which can be provided. The service level and technology concepts are considered in the following sub-sections.

3.2 Classification by Service

MMAs can be divided according to the type of service which they provide. An MMA can

- A) inform;
- B) diagnose/assess;
- C) advise/decide/control.

Essentially, this progression hangs on the degree of authority which the crew invest in the MMA. This in turn is a function of the trust which the crew has in the MMA, which reflects its robustness and/or its completeness.

A shortfall in robustness may result from an inherent weakness or uncertainty in a machine intelligence procedure or from an inability to respond adequately to circumstances not (fully) allowed for in its design or in its knowledge base. The latter may be regarded as a form of incompleteness, which would also include absence of algorithmic or logical cover for sufficient circumstances (typically an incomplete search of the parameter space or of sufficient combinations of conditions).

Tactical Decision Aids can include Tactical MMAs of types (A) and (B) and, if operated in an advisory rather than decisive capacity, type (C).

At the informative level, an MMA can already be made fully reliable, simply because its scope can be limited as much as may be necessary to achieve reliability. The MMA is simply saying: "the following parameters have the following values". The pilot can then make of this what he will.

The diagnostic or assessing MMA requires more confidence in its abilities, but the pilot must nevertheless be aware of its limitations. The pilot will know what parameters the MMA has taken into account and may have confidence in the ability of the MMA to make certain defined judgements based on these parameters. The pilot will, however, be able to assess the importance of those parameters or effects which the MMA does not take into account.

The distinction between advisor and decider is one of degree rather than of kind, and the distinction between decider and controller is one only of implementation: of whether the necessary wiring and hydraulics are present.

For an MMA to decide on a course of action, there must be grounds for believing that the advice is sound. If a comprehensive questioning of the MMA is possible, then the MMA in effect reduces to type (B) — the pilot can make his own final judgement. If time does not permit such a questioning, then the pilot must either accept or reject the MMA's advice *in toto*. In order to accept it, he has to assume that all relevant factors have been correctly taken into account.

Simple systems, especially mechanical ones, have for long had such authority: autopilots, landing systems, fire extinguishers, IFFN. We have a long way to go, however, before the machine will reliably outperform the best humans in air combat.

3.3 Classification by Technology

We can consider a sequence of capabilities which we can seek of MMAs.

- A) conventional - number-crunching which, although possibly complex and inventive in its conception, is uninventive in its execution within the computer;
- B) smart - logic and number-crunching giving superficially intelligent processing but always responding to foreseen circumstances in a foreseen way;
- C) intelligent - the MMA will handle in a reasonable way, circumstances which have not been 'wired in' (although they may be only variations on a pre-considered pattern).

Analogies to (A) to (C) can be found in the pilot himself. If (A) corresponds to unconscious operations such as local muscle control, including reflexes, or to the initial sum-and-difference etc. processing of retinal signals, then (B) would correspond to sub-conscious processes such as muscle co-ordination and sequencing or to optical pattern decomposition and recognition, and (C) would correspond to conscious thought and reasoning.

Examples within TDAs and related functions would be (A) the statistical fusion of sensor measurements, (B) the fusion of state data with partially conflicting attribute and operational intelligence data, (C) the anticipation of possible tactical moves by the targets and the consequences thereof.

It is reasonable to suppose that the tactical MMA will require capabilities of type (C), whereas, moving upstream in the data flow towards the sensors, the computer will have capabilities moving towards type (A).

Taking for granted the advent of powerful airborne computers, and assuming that the power of these will increase in the next few years at least until comparable with the light-weight machines now available on the ground, we can consider the tools and techniques available to the research and development engineer to provide these capabilities in an MMA:

- conventional geometric, statistical, etc., computing;
- recent techniques using conventional computing, many of which are still being developed, such as Kalman filtering or the more ingenious applications of Dempster-Schaeffer analysis within quasi-AI logical frameworks;
- artificial intelligence techniques which reduce to conventional computing, eg. Fuzzy logic or (arguably) A* search;
- out-and-out new-wave techniques, such as knowledge-based systems in which the knowledge base is processed by an 'inference engine', or neural networks;
- high-level programmer-friendly software environments and objective languages;

As we progress down the list, we move into areas of greater power to solve 'intelligence' demanding problems, but at the same time we move into realms of less experience, less understanding and less proven integrity. Fuzzy logic is 'rigorous' in the sense that it is mathematically sound and uses conventional computing logic with verifiable software, but is sometimes subject to customer suspicion. While Inference engines are themselves formally predictable, their interactions with their knowledge bases are not always predictable and the system as a whole may therefore not be verifiable. As yet, they are often too inefficient for demanding real-time problems. Similarly, the highest level languages and environments can have problems of both efficiency and reliability.

3.4 Tactical MMA Technology vs Function

Figure 1 illustrates an outline architecture containing sensor fusion, situation assessment, sensor manager and tactical planner. The α , β notation corresponds to that of reference 2.

(The term *Situation Assessment* is used with slightly different meanings in different countries and organisations, covering one or more of the functions grouped together in the figure from initial Sensor Fusion through to Target Prioritisation. Boundaries are difficult to define, and the terms *Situation Awareness* and *Target Prioritisation* are used in this paper where any degree of precision is required.)

Not all the elements in figures (1) and (2) are MMAs, but clearly all are closely related, and their technologies merge.

- i) Fusion of sensor measurements, including track estimation, may well be achieved by fixed mathematical formulae (including handling of uncertainties or contradictions): it is likely to be implemented as a robust and verifiable function of the equipment rather than the overall system-cum-role and is not essentially an MMA function;
- ii) Fusion with mission briefing data is likely to require inclusion of tactical judgements along with handling of

uncertainties or contradictions, and thus moves towards intelligent judgement rather than mechanical function: for the present, it is considered to be an MMA function, although whether part of a tactical decision aid will depend on the degree of behavioural data which is processed and on how that data is subsequently processed or displayed.

- iii) Target Prioritisation can be implemented at almost any desired level, from the simplest range and range-rate formulae to deepest tactical reasoning weighted against the mission objectives. Functionally, it can be considered to be a Tactical Decision Aid at any level.
- iv) The Sensor Manager is an MMA, although not a TDA.
- v) Attack Planning is clearly a TDA function.

The relationship between the Sensor Processing and MMA functions and the type of technology they require is illustrated by the matrix of figure 2. It is not claimed that the characterisations given by the figure are absolute, but they indicate a general drift towards a need for AI technology which occurs as we move functionality to the right in figure 1.

The Sensor Fusion and Manager systems are included for completeness. Although the former is not an MMA and the latter is not strictly a Decision Aid, the whole family are intimately connected and the Sensor Manager is a Mission Management Aid which, in a combat environment, requires tactical knowledge. Depending on the amount of autonomy which the Sensor Manager is required to carry, there is scope for greater or lesser degrees of 'intelligent' tactical knowledge. In terms of the technology required to implement it, the Sensor Manager therefore forms a bridge between the largely numerical processing functions upstream and the largely logical processing functions downstream.

4 THE APPROACH TAKEN

MMA developments within BAe(MAL) are progressed by goal-driven prototyping projects backed up and guided by a continuing programme of requirements analysis.

4.1 The Requirements Analysis

The general requirement for MMAs is clear from experience in the design and operational support of combat aircraft to date and from discussion with aircrew. However, to clarify and refine the detail and priorities, British Aerospace's overall MMA programme includes a Requirements Definition activity. This involves aircrew with tactical and Human Factors specialists in mission simulations, designed to identify the priority tasks to be assigned to MMAs and to identify which aspects of these tasks, including the extent and the balance of autonomy, which crew would wish to delegate to the machine. The results from these investigations guide the selection of topics and objectives for all MMA programmes, including those on Tactical Decision Aids.

For overall tactical planning MMAs, particularly air-to-air, where some problems are believed not to be solvable in the short-term with present knowledge and (airborne) hardware, the objective has been to develop an understanding of the nature of the logic and mathematics of possible solution techniques, the quality of data required, the amount of computation and data transfer required, and the processing architectures, in preparation for longer-term programmes.

4.2 Prototyping Projects

The objectives derived and prioritised by the Requirements Research programme guide the evolution of an overall strategic MMA programme, within which individual goals are pursued by demonstrator sub-programmes. Each such sub-programme comprises an iterative cycle of brainstorming, prototyping and evaluation, thus:

- i) recommendations from previous cycle plus new ideas;
- ii) design of next prototype;
- iii) implementation on a workstation;

91-15523



91 1118 018

- iv) initial appraisal - possible return to step (i);
- v) implementation on combat simulator or other simulation facility;
- vi) mission simulation and aircrew appraisal and comment.

Consideration has been given to conventional, procedural processing, and to declarative and AI/KBS techniques. It is believed that conventional implementation will be required for airborne application in the short-to-medium term, although this will not preclude 'conventional' implementation of some AI techniques. It is also believed that AI/KBS will be necessary for many of the more complex and 'intelligent' later MMAs. Therefore, a balance is maintained between the two at the research stage, including the use of procedural functions within an otherwise KBS-shell-based program. Most work, however, is carried out in conventional languages, for possible short-term implementation, for efficiency of execution (necessary for simulation assessment), and for compatibility with collaborating departments' existing equipment and expertise.

The C language has been found to be the most suitable for experimental and prototype coding. It gives the greatest flexibility of interface to the workstations, and requires little or no code modification on transfer to the disparate processors of the avionic and flight simulation rigs. The MUSE language (a derivative of Prolog) is used for KBS elements. The developed prototypes will enable design specifications to be drawn up from which 'production' MMAs would be implemented in languages such as ADA.

British Aerospace is pursuing MMA development across the range of functions, including Tactical MMAs and the related Sensor Management functions. BAe is a participant in a Joint Venture programme with GEC, Smith's Industries and RAE, discussed in a later paper in this symposium (reference 1). The Joint Venture has concentrated on defence against ground threats, while BAe's in-house effort has concentrated on air-to-air combat.

Exploratory work has been conducted on air-to-air TDAs of a type, although not a degree, which would allow an MMA to control the aircraft in BVR air combat. Development to a degree which would allow such an MMA to enter service is seen as a comparatively long-term goal. Work is being paced to maintain a balance in the rate of development of AI and KBS techniques, of tactical algorithms, of sensor and counter-measure technology, and of available airborne hardware.

Reference 2 describes a prototype TDA, COMTAC, which uses conventional, or Procedural, computing techniques. Since the publication of reference 2, COMTAC has been further developed and has been implemented in the Air Combat Simulator at Warton. In a parallel activity, a Knowledge-based TDA prototype, KATS, explored the application of KBS, and the TACAID programme (reference 3) addressed the integration of the two approaches.

Recognising the priorities of the next release of service aircraft, effort is concentrating on MMAs with shorter lead times, in particular on Sensor Management, Situation Awareness and Attack Planning Aids of the informative rather than decisive type, type (A) of section 3.2. TDAs are being developed which process parameters of the measured scenario and present derived tactical information to the crew in a concise and quickly assimilable form. These TDAs are essentially real-time visual aids in which the pilot can see tactical portrayals of the battle scenario from which he can make the tactical judgements.

5 PROGRESS AND LESSONS

Recent years have seen steady advances in knowledge and capability. In some areas, success and progress exceeded that which was anticipated, while in others, inevitably, some retreat and consolidation was forced. In the latter cases almost as much as in the former, knowledge was gained which was of value in planning the next step. Usually, when something was found not to work, it led quickly to finding out what does work.

5.1 MMA Requirements Analysis

Although MMA Requirements had been pursued for a number of years by discussion, the experimental activity specifically directed towards this began in the summer of 1990, when the necessary simulation facility, the Part Task Simulator (PTS) at Dunstfold, was sufficiently developed.

The PTS is designed for rapid-prototyping of man-machine interface and MMA requirements definition studies within simulated missions. It gives a cockpit-like environment, with computer-generated outside-world graphics, in which pilots can 'fly' simulated combat missions. It can be linked to an advanced avionics development cockpit when more realistic and/or two-aircraft simulations are needed.

Initially, an MMA is represented by human operators. This serves two purposes: it avoids biasing the trial by the preconceptions or limitations inherent in any particular man-machine interface, and it avoids the problem that, in general, the MMAs whose requirements are to be investigated will not yet exist! The pilot is able to delegate such functions as he chooses to the MMA: progressively, the real (silicon) MMAs are introduced to reproduce the functions which the pilot has delegated to the human. When the silicon MMA is in place, an integrated investigation of requirements and solution can be progressed.

Trials so far have established patterns of crew preferences for how, when and to what extent, the crew would wish to delegate control of functions such as target selection and sensor modeing to the MMA.

There are types of MMA operation in which the reduction in the pilot's workload obtained by the MMA can be lost to the increase in the pilot's effort in controlling and monitoring not only the MMA itself, but the status of the pilot/MMA partnership. It was found that certain patterns of delegation could lead to confusion about who, pilot or MMA, controlled what function, while with other patterns of delegation the pilot retained a clear understanding of how the man-machine partnership was operating. Narrow and deep, but movable, delegation, can lead to unproductive workload to monitor the delegation state itself. This is usually relieved by a system of progressively deepening broad delegation.

The crew requirements and workload studies also underline the need to maintain a balance between the two prime objectives of MMA research: on the one hand to reduce the workload of the crew faced with increasingly complex systems, on the other hand to optimise the performance which the crew can obtain from these systems. It would be all too easy to fill the cockpit with computer-based aids to weapon system management which could achieve excellent performance ... if the pilot had sufficient hands and eyes and ears and speed and concentration to be able to operate them!

The requirements findings have, on the whole, supported the pragmatic decision to proceed at a measured pace with the more intelligent, type (C), TDAs, at least for air-to-air combat. They serve also to ensure that we do not lose sight of the importance of the man-machine interaction in our quest for ever more capable machines.

5.2 Tactical Planning Aids

The COMTAC demonstrator discussed in section 4 has been installed in the Air Combat Simulator at Warton, and has been flown by pilots in scenarios containing computer-controlled escorted bomber raids. Flying the prototype has given pilots a datum from which to comment on its tactical behaviour and from which it is possible to identify features which pilots consider to be of value in their own right. Already, spin-off benefits have been obtained: for example, displays design work for the trial has given displays for multiple-target missile fire control which are considered to be an advance on previous designs.

The TACAID programme to combine Procedural and Declarative approaches to MMA design is described in an earlier paper

of this symposium (reference 3).

A major spin-off from the COMTAC programme has been an MMA which was initially similar to COMTAC in concept but with modified displays, superimposing on the attack map display, tactical parameters related to a simpler but denser set of profiles. This MMA became known as TACMAP.

TACMAP was well received by pilots, but with recommendations for improvement. These recommendations evolved steadily into a substantial change in concept. In collaboration with pilots and MMI specialists, the MMA team has evolved TACMAP away from its COMTAC roots into a quite distinct TDA family, with potential to mature from concept R & D to product development within two or three years.

The conceptual changes which allowed TACMAP to grow rapidly towards maturity involved in essence moving away from the advisory, type (C) TDA, to concentrate on informative and diagnostic types (A) and (B), with the consequent ability to progress effectively using only technology types (A) and (B), advanced-conventional and Smart. The explicit steering advice has been dropped, and emphasis has been placed on the cartogram, showing tactical parameters in a time-projected sense across the battle area. It has been found that considerable abstraction, even ambiguity, can be included in the tactical meaning of the parameters yet, with practice, the pilot can respond to the display to obtain more successful combat steering than without the MMA. This means that

- i) the definitions of many parameters modelling physical or tactical quantities can be greatly corrupted towards great algebraic/trigonometric simplicity (with care!), allowing rapid updates;
- ii) the uncertain behaviour of the enemy, and multiple-choice by own pilot further into the engagement, need not be represented rigorously or the possibilities displayed individually; they can be amalgamated into composite parameters having suitable cueing behaviours as decision points approach or uncertainties arise.

The abstraction principles have been demonstrated in simulations to be sound, enough has been done on the workstations to prove in concept that the caveats under item (ii) above can

be satisfied, and TACMAP's tolerance to sensor error has been shown to be satisfactory.

A further spin-off from the COMTAC programme has been a body of work from which to distill rapid algorithms, embodying a degree of tactical understanding, for preliminary prioritisation of threats and targets. This can, in appropriate modes, serve a number of purposes. It can support the prioritisation capability in the sensor manager, it can provide a displays filter for the crew and for TACMAP, and it will give a data reduction filter for the comprehensive tactical planning MMAs which will, in time, enter service.

6 CONCLUSIONS

Overall, it has been found that, operating within the constraints of current hardware and software technology,

- 1) rapid progress can be made towards TDAs which can be ambitious in their intended performance and operational effectiveness, provided they do not attempt to push too quickly into the domain of real-time intelligence.
- 2) Steady progress can be made with a reasonable level of investment into TDAs which require a high degree of real-time intelligence, provided that the MMA programme does not attempt to push too far ahead of the world level of technology within the AI/KBS community. To advance faster requires that the MMA community advance a range of techniques which have application in other areas, thus losing the general synergy between Applications research and wider Enabling Technology research.
- 3) Combining the above approaches gives a consistent framework for the activity as a whole. The longer-term goals prompt useful and achievable shorter-term sub-systems, while the self-determining sub-systems generate enabling technology and an overall 'feel' for the problems to feed the longer-term projects.
- 4) The requirements (i) to enable the crew to maximise system performance and (ii) to keep the crew workload within acceptable bounds can not be addressed independently. Addressed independently, each can adversely affect the other.

References

1. Lovesay, E. J., "Integrating Machine Intelligence into the Cockpit to aid the Pilot", AGARD AvP Sym 61, May 1991, paper 19
2. Mitchell, N., "Computer Aided Tactics in the Cockpit", AGARD CP 440, October 1988, paper 25
3. Roberts, K., "TACAID — A Knowledge Based System for Tactical Decision Making", AGARD AvP Sym 61, May 1991, paper 7

Discussion

1. G. Bourassa, Canada

Please clarify the designation of TACAID as a "selection" mechanism.

Author:

I believe that this designation was made in Paper 7, which I presented on behalf of Dr Roberts, not in Paper 17. I believe that the point Dr Roberts wished to make was that, rather

than searching the parameter space for an optimum solution, with the option to generate alternative sub-optimal solutions (e.g. A*), the TACID heuristics "select" plans and plan-steps according to trigger states, without cross-comparison of competing possibilities. (Although there is scope for local optimization when the physical parameters of the plan, such as speed, height, heading, etc., are algorithmically quantified for implementation.) Thus any one state of the scenario and of the rule base will generate (select) one and only one plan description.

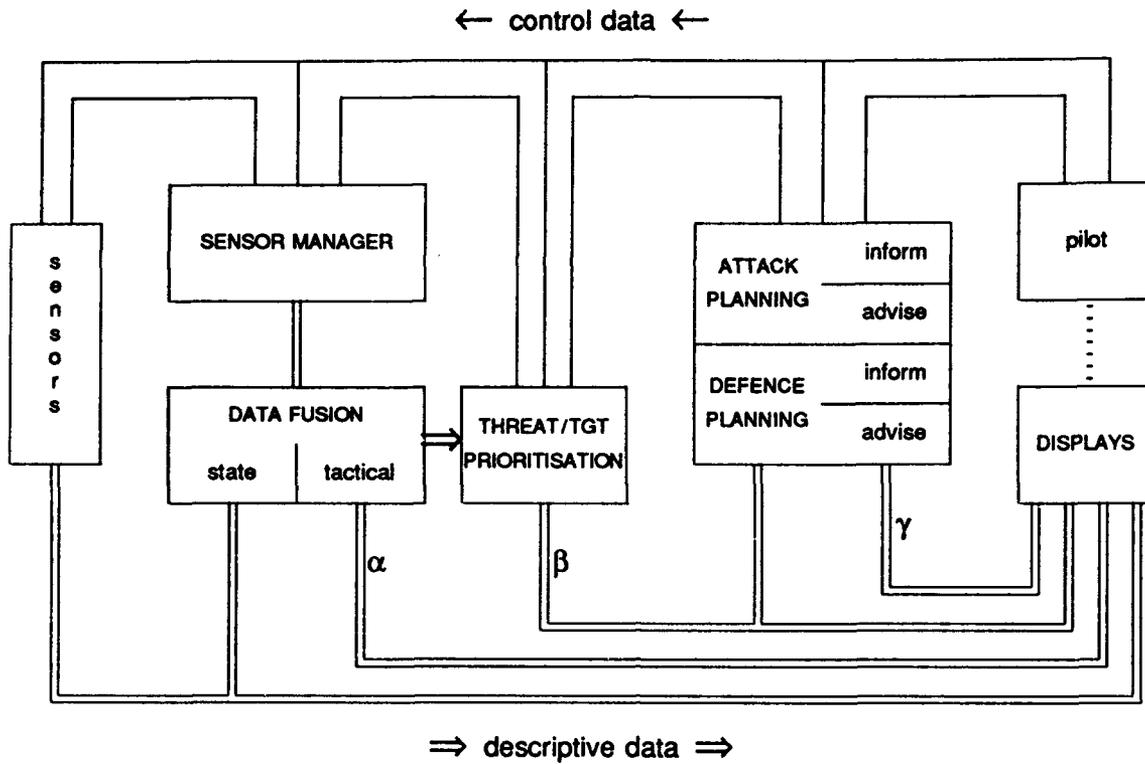


Figure 1: Illustrative Sensor + TDA Architecture

	ADVANCED CONVENTIONAL	SMART	INTELLIGENT
DATA FUSION (Measurement)	3	3	0
DATA FUSION (Behavioural)	3	3	1
SENSOR MANAGER	3	3	2
TARGET PRIORITISATION	2	3	2
ATTACK/DEFENCE PLANNING — Informative	2	3	2
ATTACK/DEFENCE PLANNING — Advisory	2	3	3

0 = not required 1 = useful 2 = valuable 3 = essential

Figure 2: Sensor + TDA Technology Required



SYSTEME EXPERT EMBARQUABLE SUR

 AVION D'ARMES POUR EVALUATION

 DES PERFORMANCES TEMPS REEL

D. SERVEL - A. HAVRE
 DASSAULT ELECTRONIQUE
 55 Quai Marcel Dassault
 92214 Saint-Cloud
 France

1. RESUME

Cet article traite des questions d'embarquabilité de fonctions d'intelligence artificielle. Il détaille l'étude effectuée par DASSAULT ELECTRONIQUE et DASSAULT AVIATION visant à évaluer les performances temps réel que requiert l'embarquabilité d'un système expert avionique.

Une maquette de système expert d'aide à la compréhension des pannes et de gestion des procédures d'urgence à bord du démonstrateur RAPALE A a fourni le support de cette étude.

Les conclusions positives de l'étude et les perspectives futures qu'elle a dégagées constituent l'objet de cette publication.

2. APPORT DES TECHNIQUES D'INTELLIGENCE

 ARTIFICIELLE POUR L'ASSISTANCE AU PILOTE

2.1 Contexte de l'étude

L'étude, objet de cette présentation, a été réalisée conjointement par DASSAULT ELECTRONIQUE et DASSAULT AVIATION dans le cadre d'un marché financé par le Service Technique des Télécommunications et des Equipements aéronautiques (STTE) de la Délégation Générale pour l'Armement (DGA).

Dans les problèmes d'évaluation de l'état des différents systèmes de l'avion (moteur, génération électrique et hydraulique, freinage, etc.), un point particulière-

ment important pour assister le pilote au cours de sa mission est le filtrage des alarmes et la gestion des procédures d'urgence.

Ceci se traduit par l'analyse de l'incidence de la panne pour informer le pilote, lui conseiller les actions limitant l'impact de la panne et le prévenir des réductions du domaine d'utilisation de l'avion.

Dans cette optique, on s'intéresse essentiellement aux dépendances inter-systèmes (la génération électrique est par exemple résultante de l'entraînement des alternateurs par les moteurs). Le raisonnement lié à ce problème comprend chronologiquement trois étapes :

- . Effectuer, en fonction des phases de la mission, un filtrage initial intelligent des pannes pour ne conserver que l'information essentielle en cabine (la panne la plus critique en général).
- . Evaluer les conséquences de la panne origine et des anomalies résultantes de façon à faire face le mieux possible à la nouvelle configuration.
- . Proposer les actions curatives selon les procédures d'urgence de telle sorte que l'on puisse préserver au maximum l'intégrité de l'avion, ce qui suppose souvent des limitations de domaines.

Un tel raisonnement fait appel à la connaissance des systèmes et des équipements de l'avion et de son Système de Navigation et d'Attaque (SNA).

Apporter une solution à ce problème impli-

que de plus la prise en compte de toutes les contraintes liées au temps. Il est en effet indispensable de proposer des actions avec des temps de réponse acceptables pour le pilote. Ceci ne veut pas dire que ces temps de réponse doivent obligatoirement être très courts, mais qu'ils doivent être compatibles avec le déroulement réel d'une mission et avec les impératifs de sécurité.

De plus, les connaissances utilisées sont suffisamment larges pour inclure des représentations fonctionnelles des équipements, des connaissances de surface sur les dysfonctionnements, des connaissances sur le principe de déroulement des missions, et enfin des connaissances d'ergonomie et de comportement pilote.

2.2 L'approche intelligence artificielle

L'intelligence artificielle peut apporter de nombreuses réponses à ces problèmes.

Tout d'abord, la qualité des mécanismes d'inférence, aujourd'hui bien rodés, répond bien à la complexité des raisonnements que l'on veut mener à bien. Les progrès des techniques de modélisation des connaissances liés aux avancées en matière de logique permettent de bien saisir, sans la restreindre par un moule trop pesant, la diversité des connaissances. Enfin, l'approche nouvelle du point de vue de la programmation, introduite par les systèmes à règles de production et les langages orientés objets doit permettre d'assurer la modularité et la flexibilité indispensables à de telles applications.

Ainsi, les techniques et les langages de l'intelligence artificielle permettront de modéliser certains raisonnements que doit faire le pilote de manière à alléger sa tâche dans un environnement agressif ou de l'aider dans des opérations complexes, laborieuses ou répétitives, ceci grâce à une transcription dans un calculateur de l'expertise et de l'expérience acquise par les opérationnels, l'avionneur et les équipementiers.

3. PROBLEMES TEMPS REEL INHERENTS A

L'INTELLIGENCE ARTIFICIELLE EMBARQUEE

Tout système fonctionnant en temps réel doit faire face à deux types de contraintes :

- . Les contraintes propres aux données à traiter : les informations sont en général hétérogènes, parfois incomplètes, incertaines ou contradictoires, et dans tous les cas changeantes parce qu'à valeur temporelle. A cette nature intrinsèque des données s'ajoutent en plus les contraintes d'échanges par messages, synchrones ou asynchrones, aléatoires et surtout indépendants des traitements.
- . Les temps de réponse impartis et les délais accordés pour l'exécution des traitements : un système temps réel doit pouvoir faire face à des pointes en charge de calcul, gérer des tâches multiples, des conflits, des interruptions. Cette maîtrise nécessaire de l'exécution implique donc des capacités de traitement élevées, des traitements gérés par tâches de priorités différentes, des niveaux d'interruptions dans ces traitements, voire un moniteur de gestion des tâches.

A l'inverse, un système expert est peu préparé à affronter de telles contraintes :

- . Les informations sont plus souvent symboliques que numériques, les données ont en général une valeur durable, si ce n'est définitive, ce qui fait que dans la plupart des cas les problèmes de non-monotonie sont écartés.
- . Les performances ont un caractère difficilement prévisible, en particulier en ce qui concerne les temps d'exécution des raisonnements, puisqu'ils dépendent des stratégies utilisées, de l'ordre dans lequel sont traitées les informations ...

Le problème de la mémoire vient également s'ajouter à ces contraintes. Les applications IA sont en général "gourmandes" en mémoire, et utilisent toutes un "garbage collector" pour leur gestion. Ceci a pour conséquence :

- . Un coût en temps de récupération de la

mémoire.

- . La nécessité d'avoir une gestion dynamique de la mémoire.
- . Des difficultés d'interfaçage avec les langages algorithmiques, les informations n'ayant pas une place allouée fixe.
- . Des problèmes de complétude si l'on ne dispose que d'une mémoire limitée.
- . Un temps d'exécution finalement non prévisible.

Dans le domaine embarqué, les contraintes évoquées sont cruciales et l'IA doit s'y adapter si elle veut avoir sa place. Dans cette optique, différents axes d'étude et de recherche se sont développés :

- . Au niveau matériel avec l'emploi de processeurs parallèles ou de calculateurs dédiés IA.
- . Au niveau logiciel avec la modélisation des données, le raisonnement temporel et le développement de techniques de gestion du "garbage collector".

En se limitant à l'aspect logiciel, il s'agit de maintenir la cohérence malgré l'évolution continue des faits. Pour la modélisation des données, ceci passe par une représentation temporelle des connaissances : les faits sont datés ou ont une validité paramétrable en fonction du temps. Pour le raisonnement, on cherche à exprimer des relations temporelles qui se traduisent par des stratégies d'ordonnement selon des échéances fixées, une prise en compte des délais dans la priorité d'exécution des règles et dans le degré de validité du raisonnement final. Ceci dans le but de prévoir les performances au moyen d'approximations et d'incertitudes dans le raisonnement. Le gain en vitesse d'exécution est souvent lié à la diminution de la qualité du raisonnement.

Ces considérations sur le raisonnement temporel amènent naturellement à des concepts de tâches d'inférence. Ceci se traduit par des blocs de règles ininterrompibles (les règles étant elles-mêmes activables en fonction de critères de priorité, de temps alloué ...) et par un gestionnaire de ces tâches qui prend en compte les interruptions logiques, les attentes d'événements, les priorités ...

La motivation centrale de l'étude était de confronter les 2 approches possibles pour

une IA embarquée, à savoir l'interprétation (portage direct de Prolog sur calculateur embarqué) et la traduction (compilation de l'application Prolog en un langage classique), en regard des problèmes décrits précédemment.

4. METHODE DE DEVELOPPEMENT POUR L'ETUDE

La méthode de développement retenue a permis d'obtenir un prototype aussi représentatif que possible selon différents axes :

- . Implémenter un système expert reflétant fidèlement l'aide apportée au pilote par une détermination du domaine d'application, un recueil d'expertise aussi complet que possible et une validation par les experts.
- . Mettre en évidence les contraintes temps réel sur des scénarios de mission réalistes, par une définition appropriée du formalisme de représentation des connaissances et une réalisation intégrant au maximum les structures particulières à mettre en oeuvre pour l'achèvement d'un système expert temps réel.
- . Apporter des éléments de réponse relatifs au problème de l'intégration d'un système expert dans un système embarqué global.

Dans le souci d'atteindre ces trois objectifs, l'étude a été décomposée en différentes phases :

1. Spécification du cadre global de l'étude
2. Acquisition de l'expertise,
3. Définition du formalisme de représentation de la connaissance,
4. Génération de scénarios de mission,
5. Réalisation du système,
6. Validation du système,
7. Conclusions sur les contraintes liées à l'implantation d'un système expert dans un calculateur embarqué,
8. Conclusions sur l'intégration d'un

système expert dans un système embarqué.

5. ASPECTS TEMPS REEL CRITIQUES ABORDES

DANS L'ETUDE

L'objet principal de l'étude a été d'apporter des éléments de réponse aux différents problèmes liés aux contraintes d'embarquabilité d'un système expert.

Cette étude a également permis de dégager un certain nombre d'axes de réflexion pour l'avenir.

Les points sensibles que l'on désiret appréhender et sur lesquels a porté l'effort sont les suivants :

1. Volume mémoire nécessaire pour implémenter l'application IA sur un calculateur embarqué.
2. Capacité de traitement requise pour un calculateur embarqué afin de respecter les temps de réponse imposés par l'utilisation opérationnelle.
3. Examen et recensement des opérations sensibles ou répétitives qu'il serait intéressant de micro-programmer afin d'accélérer les traitements de la même manière que ce qui est fait actuellement pour les langages classiques.
4. Nécessité d'utiliser, sur un calculateur embarqué, un Prolog compilé ou alors simplement interprété dans le même but d'accélérer les traitements et de tenir les performances.
5. Compatibilité au sein d'un système d'intelligence artificielle entre les parties purement "inférentielles" utilisant Prolog et celles faisant appel à de l'algorithmique et utilisant un langage classique.
6. Caractéristiques fonctionnelles d'un moteur d'inférence "adéquat" pour l'implémentation d'un système expert embarquable.
7. Structures particulières à mettre en oeuvre au point de vue des faits et des connaissances pour gérer efficacement

des informations temporelles et parfois issues de plusieurs sources.

8. Détermination des caractéristiques de l'interface entre le système expert et d'autres systèmes avion déjà existants pour les échanges d'informations.
9. Nécessité de concevoir une forme de "moniteur" pour gérer les priorités entre les raisonnements. Ceci est une conséquence directe de la prise en compte d'informations temporelles par un mécanisme d'interruption sur arrivée d'événements.
10. Adaptation à la non-monotonie des raisonnements du fait de la manipulation d'informations dont la validité est fonction du temps.

6. CONCLUSIONS SUR L'EMBARQUABILITE

ET PERSPECTIVES

6.1 Conclusions temps réel

6.1.1 Volume mémoire et capacité de traitement

L'étude a montré que l'utilisation des techniques IA était tout à fait envisageable dans le cadre d'une application temps réel.

La réponse aux deux principales interrogations, volume mémoire et capacité de traitement, n'est néanmoins pas immédiate. L'antagonisme existant entre la rapidité d'exécution et le volume mémoire utilisé d'une part, et les temps de réponse et la justesse du raisonnement d'autre part, imposent de faire un certain nombre de choix.

Dans le domaine de la gestion des alarmes, il est bien évident que le pilote ne peut se contenter d'une évaluation partielle de l'état des systèmes de l'avion ; la priorité dans l'application a donc été donnée à la justesse du raisonnement.

Mais il faut veiller, malgré tout, à faire parvenir les alarmes au pilote dans des temps raisonnables, c'est-à-dire compati-

bles avec le déroulement réel d'une mission et les impératifs de sécurité.

Privilégier les aspects complétude du raisonnement et temps de réponse a orientée l'étude vers une approche compilation de l'application. Cette compilation (des règles, des faits et du moteur) se fait en passant par un langage algorithmique classique (Pascal à titre d'essai). L'avantage procuré par cette méthode est une accélération des temps de traitements (au détriment du volume mémoire généré).

Dans ces conditions, le volume mémoire nécessaire à l'implémentation de l'application dans un calculateur embarqué classique DASSAULT ELECTRONIQUE, a été estimé à 200 Koctets.

Le gain en temps de réponse du programme Pascal par rapport à la maquette Prolog a quant à lui été estimé à 10 environ, ce qui permet de satisfaire l'objectif fixé au départ (Ce temps de réponse, c'est-à-dire la durée, dans une version opérationnelle, entre le moment où arrivent les événements en entrée du système et celui où on obtient les conclusions, avait été fixé comme inférieur à la seconde).

6.1.2 Traitements microprogrammés

La validation a montré une forte influence des opérateurs liés au temps dans les règles sur le temps d'exécution de celles-ci. Ceci est dû au calcul de l'heure au moment de l'application de la règle. Si dans un calculateur cible on ne dispose pas d'une horloge absolue (notion de temps vrai), il sera souhaitable de microprogrammer la séquence de calculs élémentaires destinés à l'obtention d'une référence de temps absolue.

6.1.3 Prolog compilé ou interprété

Le système prototype était initialement implémenté en un Prolog interprété (avant portage de l'application en Pascal). L'avantage de cette technique est la grande dynamique, mais la lenteur qui en découle va à contre courant d'une utilisation temps réel.

Dans ces conditions, le choix d'un Prolog compilé paraît d'autant plus nécessaire que le calculateur embarqué est d'archi-

tecture classique. L'embarquabilité exigerait néanmoins, au minimum, la définition et la réalisation d'un moteur dédié à la machine cible et "taillé sur mesure" en éliminant les fonctionnalités privilégiant l'aspect convivial du produit dans la phase de développement.

Le problème de récupération de la mémoire vient également conforter ce choix ; cette opération dépend de la nature du langage suivant qu'il est compilé ou interprété. Il est préférable de s'orienter vers des outils "garbages incrémentaux" à temps de déclenchement et d'exécution fixés ou des outils "garbages free" qui permettent une bonne maîtrise du temps.

6.1.4 Moteur d'inférence

L'outil utilisé pour modéliser la maquette (EMICAT de DASSAULT ELECTRONIQUE) se présente comme une extension de Prolog vers un langage orienté objet.

Le développement de la maquette a montré que de nombreuses fonctionnalités offertes par EMICAT, à la fois très générales et très puissantes, ne trouvaient qu'une utilisation très réduite pour notre application. L'hypothèse selon laquelle on embarque le système développée dans l'environnement EMICAT suppose donc que l'on élargue ou modifie certains de ces mécanismes. En revanche, certaines fonctionnalités pourraient être enrichies dans le cadre d'une utilisation temps réel, en particulier les logiques et opérateurs liés au temps développés pour le filtrage d'alarmes.

6.1.5 Interruption du raisonnement et non-monotonie

Dans l'application, la priorité a été donnée à l'acquisition d'informations externes qui se fait indépendamment des traitements. Plusieurs scénarios critiques en arrivée d'événements ont été joués et ont montré le bon fonctionnement des mécanismes de prise en compte des nouvelles données.

La conséquence de ce point précis est l'invalidation de certaines conclusions obtenues par le système expert avant l'arrivée d'événements perturbateurs. La validation a souligné que le mécanisme de mémorisation d'états de la base de faits

affichait une certaine faiblesse, due à sa "brutalité" d'application, face à des arrivées d'événements trop fréquentes remettant en cause les raisonnements en cours.

6.2 Conclusions sur l'intégration à un SNA

En complément des caractéristiques de performances du calculateur dédié à recevoir l'application IA, l'étude avait pour objectif d'apporter des conclusions quant à l'intégration d'un système expert dans un Système de Navigation et d'Attaque :

- . dégager des principes, fonctions et interfaces qui pourraient être retenus pour une IA embarquée future,
- . étudier la faisabilité de l'intégration d'un tel système expert de filtrage d'alarmes dans un système d'armes embarqué déjà existant.

La taille mémoire estimée du système prototype de filtrage d'alarmes qui a servi de support à notre étude est de 200 Koctets. Les calculateurs embarqués d'avions de type Rafale possèdent une capacité mémoire de quelques Méga-octets. L'intégration d'un tel système expert sur ce type d'avion est donc possible, mais nécessiterait néanmoins l'implantation d'un calculateur spécifique, au niveau logiciel, pour les applications IA.

En général, sur les avions d'armes, la tâche cyclique la plus lente est activée à la fréquence de 6,25 Hz. La prise en compte des événements à cette fréquence de 6,25 Hz suppose que le système expert soit capable d'exécuter l'ensemble des raisonnements et d'aboutir à un diagnostic en moins de 160 ms. Dans le cas contraire, le problème de la non-monotonie et donc de la remise en cause des conclusions intermédiaires reste entier si l'on ne redéfinit pas les principes d'activation des tâches.

Le système expert qui a été développé pour cette étude ne prend en compte que des informations élaborées d'états ou de pannes et ne traite pas les informations primaires issues des capteurs. Pour qu'il soit intégrable à un avion existant, il est nécessaire de disposer de ces informations élaborées.

L'intégration est de plus facilitée si ces informations élaborées existent sous forme numérique et si elles transitent par un bus.

6.3 Perspectives futures

L'une des perspectives suite à l'étude est d'approfondir les conclusions relatives à l'approche traduction du système expert en un langage classique.

Le point central de cette approche est le développement du traducteur. La première phase consiste à définir un formalisme de représentation de l'expertise permettant au mieux la traduction. Le traducteur est de préférence dédié à un type d'application (la synthèse de pannes) ce qui permet de cibler le formalisme et ainsi d'optimiser la traduction vers un langage classique, une trop grande généralité étant, pour le sujet qui nous intéresse, un obstacle à l'efficacité.

Les avantages procurés par cette approche sont de plusieurs ordres :

- . la traduction en un langage classique (Ada principalement) permet une meilleure maîtrise du volume mémoire et des temps d'exécution,
- . elle permet d'intégrer le développement du système expert à la chaîne de production des logiciels de systèmes d'armes classiques, ce qui permet entre autres toutes les étapes de tests et de validation que la criticité d'un logiciel embarqué impose,
- . enfin, le portage sur un calculateur embarqué permet de faire évoluer le système expert dans un véritable environnement temps réel, sans adaptation spécifique.

Les autres voies de recherche dégagées par cette étude méritent également d'être considérées. Elles concernent principalement le développement d'un calculateur embarquable dédié à l'IA, le portage de Prolog sur calculateur embarqué, la conception d'outils de base, de moteur d'inférence temps réel, ... qui font l'objet d'autres études à DASSAULT ELECTRONIQUE.

COMPONENT PART NOTICE

THIS PAPER IS A COMPONENT PART OF THE FOLLOWING COMPILATION REPORT:

(TITLE): Conference Proceedings of Machine Intelligence for Aerospace Electronic Systems Held in Lisbon, Portugal on 13-16 May 1991 (L'Intelligence Artificielle dans les Systemes Electroniques Aeroespaciaux)

(SOURCE): Advisory Group for Aeronautical Research and Development, Neuilly-sur-Seine (France)

TO ORDER THE COMPLETE COMPILATION REPORT USE AD-M242025

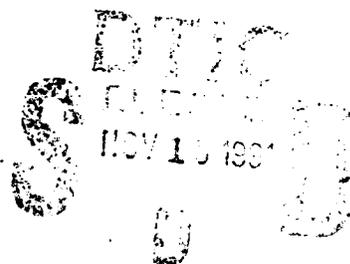
THE COMPONENT PART IS PROVIDED HERE TO ALLOW USERS ACCESS TO INDIVIDUALLY AUTHORED SECTIONS OF PROCEEDINGS, ANNALS, SYMPOSIA, ETC. HOWEVER, THE COMPONENT SHOULD BE CONSIDERED WITHIN THE CONTEXT OF THE OVERALL COMPILATION REPORT AND NOT AS A STAND-ALONE TECHNICAL REPORT.

THE FOLLOWING COMPONENT PART NUMBERS COMPRISE THE COMPILATION REPORT:

AD#: _____ TITLE: _____

AD-

AD-P006 342



Accession For	
NTIS CRA&I	<input checked="" type="checkbox"/>
DTIC TAB	<input type="checkbox"/>
Unannounced	<input type="checkbox"/>
Justification	
By	
Distribution/	
Availability Codes	
Dist	Avail and/or Special
A-1	21

This document has been approved for public release and sale; its distribution is unlimited.

AD-P006 342



INTEGRATING MACHINE INTELLIGENCE INTO THE COCKPIT TO AID THE PILOT

by

Dr E.J. Lovesey and R.I. Davis
Royal Aircraft Establishment
Mission Management Aid Project
Farnborough, Hants GU14 6TD
United Kingdom

SUMMARY

Combat aircraft of the 21st Century will have to be increasingly well equipped to counter the future threat, which itself will have become far more potent. This spiralling system complexity will result in unacceptably high cockpit workloads unless some form of automation is provided to aid the pilot. UK Industry and Government have combined to form a Joint Venture team to investigate the problem and to develop a Mission Management Aid which will assist aircrew throughout the mission, whatever and wherever it is. The paper outlines some of the problems to be overcome and suggests the major functional areas which will constitute the Mission Management Aid.

Key Words:-

Artificial Intelligence, Automation, Mission Management Aid, Man-Machine Interface, Threat, Workload.

2. INTRODUCTION

At the end of this Century, Machine Intelligence will form an integral part of the Man/Machine combine in future combat aircraft.

Machine Intelligence will be used to fuse data from dissimilar sources, to apply threat values to the fused data and to optimise a flight path for the mission. This will relieve the pilot of many difficult and lengthy cognitive processes and enable him to devote considerably more time to situational awareness and tactics, particularly when under stress.

In the past, it has been possible to absorb periods of high workload by delaying or sharing tasks between crew members. Now the situation has been reached where, despite automation, the complexity of the avionics data output is at such a level that the crew are increasingly less able to cope with the workload peaks. This situation is exacerbated with the tendency of combat aircraft to become single seat designs. Furthermore, the density and increasing sophistication of enemy surface-to-air weapons greatly adds to the complexity of the task which confronts the pilot on an air-to-ground strike mission.

Machine Intelligence in the form of the Mission Management Aid (MMA) will provide a solution to this difficult problem, by providing the pilot with data, processed to fit his needs. This Aid will correlate data from many sources such as aircraft and avionics systems, data bases, data links etc and present it to the pilot in an easily assimilable way. The MMA will fuse the data from separate sources and

sensors, apply confidence factors and threat weightings before finally suggesting optimum flight plans which will minimise sortie costs. In addition, the MMA will monitor and evaluate the aircraft systems, cue the pilot to the most appropriate action and generally take care of the normal "house-keeping" duties, performed by the crew. The MMA will act as an advisor to the pilot, much in the same way that a navigator does now. (Further details of the MMA can be found in AGARD Papers by Catford and Gray¹ and Gibson and Garrett²).

If required, the pilot will be able to interrogate the MMA through the successive processing stages, back to the raw data. However, this facility is expected to be required infrequently as the pilot gains confidence with the MMA.

For the full potential of the MMA to be realised, a number of areas will have to be very carefully studied, problems identified and possible solutions discussed. For example, great care will have to be taken in understanding the pilots information requirements and in the corresponding presentation of MMA data in a way that is easily assimilable and reduces his workload. On a broader level, the scope of the MMA requires examination. Until now, it has concentrated on an European conflict. With the recent dramatic Political changes in Europe, the MMA will need to be flexible to function in less well defined operational "out of area" scenarios of which the recent Gulf War is an example.

Finally, the method of validating the MMA must be addressed. The advantages of the MMA must clearly be demonstrated before it will be accepted.

Research into this area is well underway and is being undertaken at RAE Farnborough, by the MMA Joint Venture Team, consisting of BAE (Military Aircraft Ltd), Smiths Industries Aerospace & Defence Systems, the Royal Aerospace Establishment, Farnborough and GEC-Marconi, represented by GEC Ferranti Defence Systems Ltd, GEC Avionics Ltd and GEC Sensors Ltd.

2. WORKLOAD PREDICTIONS AND NEED FOR AN MMA

It could be argued that an MMA is unnecessary as the aircrew will always make up for the shortcomings of the aircraft systems. Although this may have been true in the past, it is now an increasingly unreliable method of overcoming the problems associated with complex systems.

It is difficult to measure and quantify workload, particularly in a real environment. It is even more difficult to assess the effects of battle stress upon crew performance and then to estimate if the crew has spare capacity to cope with peaks in workload.

As aircraft and their systems have evolved over the years, two trends have become evident. Firstly, speeds have increased. Fig. 1 shows the almost exponential rise in combat aircraft top speeds. Secondly, as Fig. 2 shows, the number of instruments to be monitored in the cockpit has also increased dramatically. Even though this trend appears to have been arrested in the 1970's with the introduction of CRT's in the cockpit, the quantity of information to be monitored actually increased with the advent of multi-function electronic displays.

Maximum Speed in M.P.H

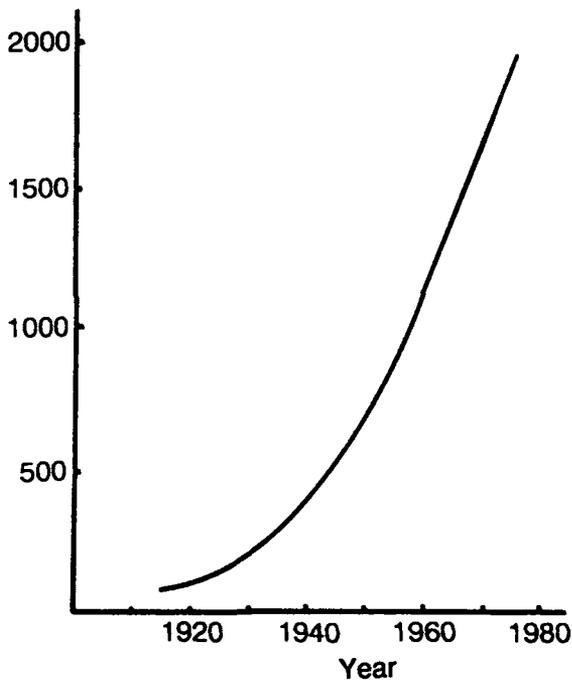


Fig 1. MAXIMUM SPEED OF COMBAT AIRCRAFT WITH IN-SERVICE DATE

Figure 3 shows the steady increase in the number of systems present in fighter aircraft from Hurricane to F18. Not only has the number of systems risen but the complexity of the systems has also increased considerably.

Thus, the combat crew of today fly faster and lower, but have less time in which to monitor and control a greater number of systems which are themselves more complex than before. These factors all contribute to an unacceptably high workload.

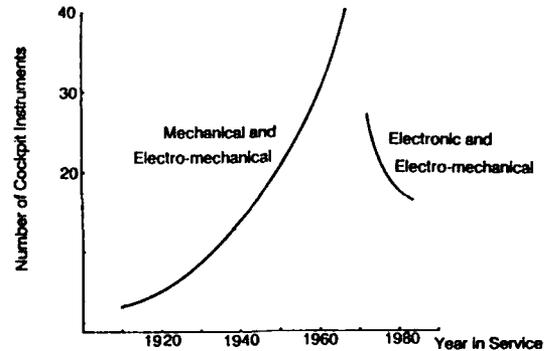


Fig 2. COMBAT AIRCRAFT INSTRUMENT NUMBERS WITH IN-SERVICE DATE

In the past it has been assumed that when overloaded with tasks, the crew will shed the less important tasks to enable the essential tasks to be concentrated upon.

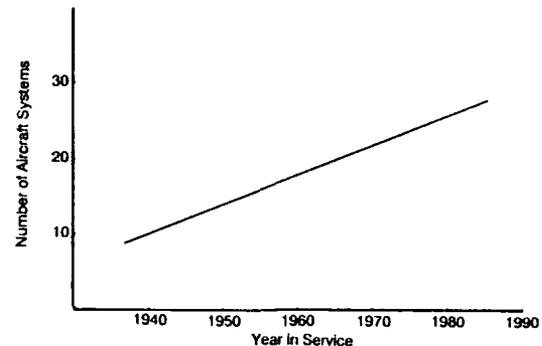


Fig 3. GROWTH IN COMBAT AIRCRAFT SYSTEMS WITH YEAR IN SERVICE

This does in fact usually occur, but it is not generally realised that the shed tasks often require a great deal of effort to be brought back on line after the workload peak has been passed. There is, in fact, a workload hysteresis loop around which the crew have to go. Figure 4 depicts in general terms, the succession of events. As workload increases, the crew performance climbs to a maximum. After the maximum is reached, tasks are shed rapidly, leaving only the primary task and the overall performance drops to a much lower level. The shed tasks cannot be immediately absorbed again but have to be steadily built up from a low overall level of performance.

Some form of automation or assistance for the crew is clearly required to prevent the maximum workload being exceeded.

In the past, attempts have been made to do this by automating some of the crew's tasks. Often, the tasks that the man can do well have been automated, (possibly because they are understood, can be defined and therefore duplicated by machines). The man then has been left to do the tasks which he is not necessarily good at, such as monitoring but which a machine might do far better than the man.

This has resulted in a less than satisfactory inefficient man/machine system which has sometimes produced a higher crew workload than when not automated. In a recent NASA study, over half of 200 Boeing 757 pilots said that they felt automation actually increases workload. Nearly half of the pilots were concerned about the possible loss of flying skills when there is too much

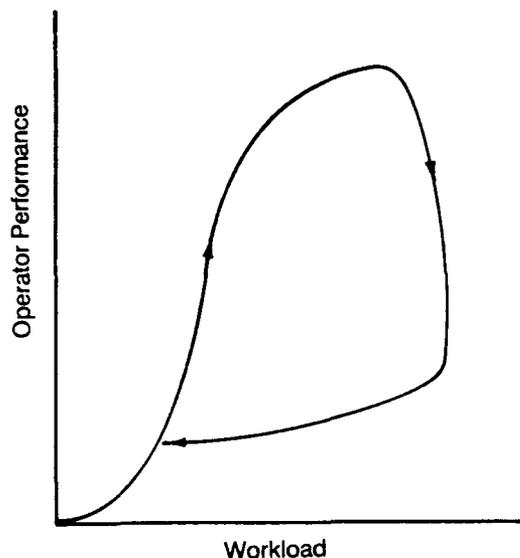


Fig 4. THE WORKLOAD HYSTERESIS CURVE

automation. Some pilots thought that their hand/feet/eye co-ordination is faster than their programming and monitoring capabilities.

Thus, it is essential that any assistance provided to the pilot, in terms of automation or aids, concentrate on the activities which man is not good at and leave him the tasks at which he is superior to the machine. This is the philosophy which has been adopted for the Mission Management Aid.

3. THE MISSION MANAGEMENT AID

The primary aims of a Mission Management Aid are to advise the aircrew such that survivability is increased, particularly in complex high threat or unfamiliar environments, and that workload is reduced to allow the aircrew to concentrate on the tasks at which the man is superior to the machine, such as inductive reasoning based on experience.

To achieve these aims, the MMA has been broken down into a small number of component parts, or MMA Core Functions, which are shown in Figure 5.

The first core function in the MMA is that of:-

Sensor Fusion

Sensor Fusion takes in information from the aircraft's Sensor Suite,

Communications Systems, Terrain and Tactical Databases and Pilot Input. The data is processed in two stages into an alpha scene (a model of the outside world with associated confidence levels).

The first stage is the correlation of tracks into positions and velocities. This involves the alignment of data with different accuracies, temporal and spatial reference frames and the subsequent association of tracks into multiple resolved tracks with confidence levels.

The second stage is attribute fusion, the identification of objects using Sensor data in the form of RADAR or IR signatures. The output from Fusion is a set of objects with their positions, velocities and identities, where each quantity has an associated confidence level.

The alpha scene produced by Sensor Fusion is fed into the Situation Assessment function. Situation Assessment is a multistage filtering process. Firstly, objects identified as friendly are filtered out of the alpha scene for separate processing, because, although they do not constitute a threat, their presence may influence the overall assessment of the threat environment. The remaining unknown and hostile objects are assessed using numerical, behavioural, operational and associated intelligence to form a model of the integrated threat environment.

The objects/systems are then prioritised, with those adjudged to be the most important (threatening, vulnerable or supportive) filtered off to make up the beta scene.

The output beta scene comprises a set of objects containing, position, velocity, identity, status and threat value information. The beta scene is the primary input to the third MMA Core Function, the Planner.

The Planner constructs tactical plans (gammas) including a gamma* option, that is perceived to be the most favourable. Its plans are based on the mission objectives and constraints, the current situation (beta scene) and available resources, such as fuel, weapons, countermeasures and supporting aircraft.

Major functional blocks within the Planner are:-

Route Options:-

Construction of coarse tactical plans in terms of 3 dimensional waypoints.

Velocity Options:-

Velocity selection and modification of the planned route to ensure that the mission fuel and time constraints can be met.

Attack and Countermeasure Options:-

Based on the mission objectives, values of potential targets and threats and the current status of the

91-15524



91 1113 010

aircraft's weapons and countermeasures, this function provides options for an attack/defence strategy to be incorporated into the gamma.

Situation Prediction:-

Evaluation of the utility of a gamma, taking account of the expected threat response along the planned route, given the effects of terrain screening, planned countermeasure usage and the value of any targets encountered.

Tactical Routing:-

Forms a detailed plan for the 'immediate future' requirements based on constraints derived from the gamma.

The final gamma output contains several levels of data covering the employment of weapons and countermeasures velocity requirements, waypoints, detailed tactical route and anticipated hostile weapon release points.

In addition to this primary thread of operation, the three Core Functions interact with a number of management functions. These managers are basically resource schedulers and housekeepers for the aircraft systems most closely connected with the Core MMA.

The Sensor Manager:-

Process requests from the Core MMA for extra information, checks the possible consequences of using the appropriate sensor and finally issues commands to the affected sensors.

The Navigation Manager:-

Performs terrain referenced navigation, terrain-following avoidance utilising information generated by the Core MMA Functions.

The Communications Manager:-

Controls the Communication Systems ensuring that information is transmitted in a timely manner when there is a high probability of effective reception and a low probability of intercept.

The Health & Status Manager:-

Monitors the Health and Status information provided by the aircraft systems. In the case of health reports, this will involve prioritisation of data dependent on urgency/criticality of the report, based on the current phase of the mission and in the light of other reports. This information along with the general status data is made available to all relevant Core MMA Functions and for possible presentation to the Pilot.

It should be stressed that the MMA Core Functions and other functions are dynamic and are constantly updating the alpha, beta and gamma scenes as the aircraft progresses through the mission. Clearly, the interaction between the MMA and pilot is potentially very complex.

For this interaction to be effective, the Pilot Interface Manager and Man/Machine Interface (MMI) needs to be very carefully designed. The Aircrew must be provided with the information that is required at the relevant time. They must not be overwhelmed with information, yet they must be given the opportunity to interrogate the core functions as far back as the raw sensor data. As the MMA evolves and is improved, the aircrew will increase their confidence in it and will feel less need to check back on the earlier processing stages. Much of the MMI design will evolve through simulation and feedback from aircrew participating in simulator experiments.

There is always the danger that because we have the technology and software to provide a complex sophisticated solution to a problem it must be adopted despite a cheaper and simpler solution being available. Frequently this has been the case and the user has been saddled with an expensive piece of equipment which is difficult to use, resulting in a very low overall system performance. The MMA must not fall into this trap.

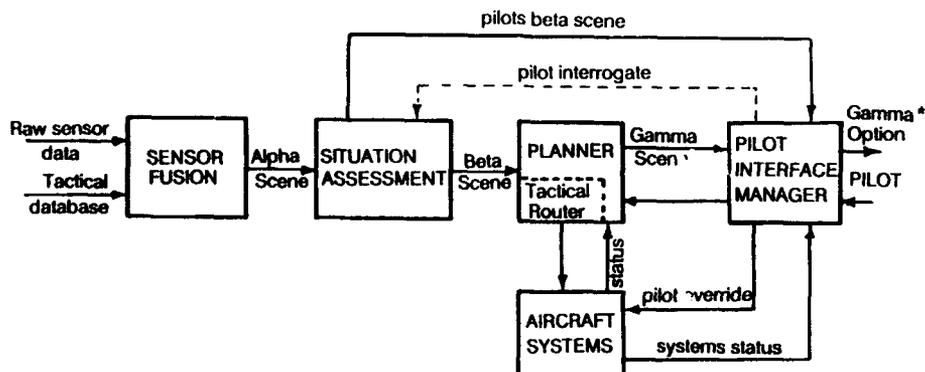


Fig 5. MMA CORE FUNCTIONS and INTERFACES

5. THE PSYCHOLOGY OF CREW MMA INTERACTION

There always will be a fundamental difference between the way the crew interact with each other and the way crew will interact with an MMA.

Currently, a great deal of information is exchanged between crew members over the intercom. system. The crew rely on a large vocabulary which includes a sub-set of technical terms and jargon specific to the mission. (See Annex of Air-to-Air combat transcript from intercom). It also relies upon a personal or social knowledge of each other which is used to build up team confidence. It is difficult to imagine joining an MMA in the mess bar after a difficult sortie to discuss how the next mission should be planned! Despite the generally poor quality of intercom. systems, some intonation signifying urgency, etc, accompanies speech which provides further information, to the recipient. In the case of side-by-side seating, as in Support Helicopters, much information is obtained from body language, head and arm movements (This has been observed frequently by the author but never quantified).

When relying on an MMA many of these cues will be absent and therefore the MMA performance will have to provide additional assistance to compensate for these shortcomings. (It could be argued that eventually "intelligent" Direct Voice Input and Synthesised Speech will have large vocabularies with 100% recognition rates and provide accurate intonation related to the situation but this, the author believes, will be beyond the MMA's timeframe).

6. MMA INTEGRATION WITH ADVANCED AVIONICS ARCHITECTURES

Until recently, most aircraft systems tended to be installed as separate items with little attention paid to their

REFERENCES

1. J R Catford and I D Gray Research into a Mission Management Aid. AGARD 49th Symposium of the Guidance and Control Panel. 10-13 October 1989, Toulouse.
2. C P Gibson and A J Garrett Towards a Future Cockpit - the Prototyping and Pilot Integration of the MMA. AGARD Symposium on Situational Awareness in Aerospace Operations 2-6 October 1989, Copenhagen.
3. E J Lovesey An attempt to quantify some Factors that affect the Operational Effectiveness of a System. Applied Ergonomics 1987, 18.4, 305-310.
4. D Hughes Glass Cockpit Study Reveals Human Factors Problems. Aviation Week 7 August 89, 32-36.
5. E J Lovesey Pilot Sensors for Air Combat, Proceedings of The Fighter Helicopter Conference, 18th - 19th January 1990, London.

integration as a complete package. There is now a move towards true integration of avionics in the form of Advanced Avionics Architectures and Packaging (A3P).

The primary aims of A3P are to overcome the problems of inadequate system integration, as mentioned above, and to reduce the long term costs of ownership of an aircraft and its systems.

The timescales and the objectives of the MMA and A3P programmes have much in common. Some consideration should be given therefore to ensure that both programmes are in phase with each other and that there is a minimum of duplication of effort. Currently, the MMA is seen to fit comfortably within the A3P framework.

7. CONCLUDING REMARKS

Machine intelligence in the form of a Mission Management Aid will be essential to enable the crew of combat aircraft to perform effectively in the 21st Century.

The MMA, as envisaged at present, will fuse the data from many sources, apply an "intelligent" assessment of the threats to provide an awareness of the situation and advise the aircrew of the best course of action to minimise threat, within the boundaries set by the mission constraints, e.g. fuel, expendables, flight envelope etc.

Thus, it will reduce workload and allow the crew to concentrate on the high level decision making, confident that the MMA is performing the "housekeeping" duties of system monitoring, fuel state calculation and lower level decision making. It will advise the crew of changes of route or manoeuvre which are caused by unexpected events which can occur at any time during the mission.

Without an MMA to assist them, the crew of the 21st Century combat aircraft will, with increasing frequency, be overwhelmed by workload problems and at a severe disadvantage with those combat aircraft and crews who have artificial intelligence to assist them.

ANNEX ATRANSCRIPT OF COCKPIT CONVERSATION BETWEEN
ONE F4 CREW DURING A 2 F4 vs 2 F16 vs 2 F5 ENGAGEMENT
AT AN AIR COMBAT MANOEUVERING RANGE

ARMAMENT: F4 - Radarguided/IR Missiles + Guns;
F5 - IR Missiles + Guns ; F16 - IR Missiles +
Guns

INITIALISATION: Each pair at the point of an equidistant
triangle - approx 30 nm separation - live
when pass within 1.5 miles of centre.

CALL-SIGN: Zippy

<u>Time(s)</u>	<u>Pilot</u>	<u>Navigator</u>	<u>RTO/Other</u> (Heading, Range etc)
-17	Looking for bogeys ...descent now, unloading		
0	OK	Fight's on Mike Check we've got Sparrow, Master Arm Ready	
	Check	Ready we have	
	Check live	CWs on today	
10			0,3,0 - 12 1,2,0 - 17 2,1,0 - 14
20	Affirmative no discrete contact ...	0,3,0 - 8 1,2,0 - 16 2,1,0 - 11
40		on the nose, 18 9000 ft, 600 kts No discrete contact yet	0,3,0 - 5 1,2,0 - 14
	...burners coming out	High on the nose, 12. Pair of bogeys heading towards, 10 miles at 30,000	
	Roger, 8 miles closing, ..slightly left, fighting wing 7 miles bogeys are 50 degrees high	
75		Just about on top, now	
85	Nothing seen7 miles, we're live, we're livewe're livekeep on trucking Mike we've got 1 Mach, Mach 1 not enough Scopes clear of bogeys	
100	Roger	10,000 ft we've got Mach 1.05, not enough	

	7,9	All	
120	(Mx firing tone)	He's breaking port	
	(Mx firing tone)	OK, fire missile, fire missile, snap rt 190	
	OK	Take a Fox 1, take a Fox 1	
	190 now	GO Sidewinder, after it. ...Ollie's coming stbd I think, I'm not sure	
	OK he is confirmed	OK, come stbd then, 20 right, range 5. 20 rt, range 5, pulling up, look high, look hi Look hi	
135	Looking up no, not a thing	Look 20 rt, range 3, 20 right, range 3 high	
	Nothing seen	20 right range 3	
150	Nothing Seen	Well, he'll be going over the top shortly, we're going to bug-out base ht	
	OK, we're bugging out now.....		
160	Roger, Hawkeye's bugging out South	Hawkeye....	
	(Mx firing tone)	Missile on Hawkeye	
170	OK (Mx firing tone)	Keep going, keep going, don't break	
		Fight's out let, long range	
		Derek,	
180	I'll keep going holding base ht	Keep going	Roger
	Reheat cut		L-B, Fox 2 ..hdg 2,2,0 good kill..
190 that	Vector for home	OK that's good	Transmit
	That's a negative kill from L-B, we're 2 miles in behind	
200	port	OK, come port	
	Roger	Make your hdg 0,6,0	Derek (Maintain hdg for bugging
220		A10	Knock-it- -it-off.

Discussion

1. D. Bosman, Netherlands

Is it possible to quote a "typical" (statistical average?) time duration or constant associated with the workload/performance hysteresis loop? How many minutes for skills acquired in over-learned training for example?

Author:

The hysteresis loop shown is only a concept at this state. I am trying to accumulate data to help to quantify the shape of the curve more accurately. Any data supporting the hysteresis hypothesis will be welcome.

2. P. Bouchard, United States

Please describe the example of pilot overload in the Gulf War which you cited when briefing your hysteresis loop of pilot effectiveness versus workload.

Author:

According to Flight International, a Tornado and crew were downed when the crew became engrossed in a guided weapon delivery — and ignored electronic warnings of two approaching missiles.

3. G.H. Hunt, United Kingdom

The constraints are put into route-planning in order to minimize the difficulty of the flying task and hence reduce the pilot workload?

Author:

No constraints are placed on the pilot's flying task. Should the pilot deviate from the MMA recommended path, the MMA will reassess the threat at the new point and suggest a new optimal route. However, the aircraft flight path is calculated taking account of normal maneuver limits which depend upon fuel and weapon load etc. and the MMA's route calculations to include a first order ride comfort function.

4. George Chapman, United States

What techniques does your system use to carry out your optimization of resources? Does your method allow changes in the objective-function (cost function)?

Author:

A pragmatic approach which models a human planning paradigm is used. The basic principle is to start with one (or more) simple plans; to analyse what is wrong with a plan, and thus decide how best to improve it. A Suitable Modification method is applied to the plan which is then evaluated. Selection of a plan for analysis is dependent on its cost and number of descendant plans. This approach attempts to deal with both combinational explosion of options and a real-time requirement to always have a complete plan available. The resolution of the cost function varies with depth of search and in the mission planner. In both mission planner and tactical routing functions, the first function to run analyses the search problem to be solved as well as the aircraft status (e.g. mission phase, fuel level, etc.) and sets up the various weighting factors accordingly.



COMPONENT PART NOTICE

THIS PAPER IS A COMPONENT PART OF THE FOLLOWING COMPILATION REPORT:

(TITLE): Conference Proceedings of machine Intelligence
for Aerospace Electronic Systems Held in Lisbon, Portugal on
13-16 May 1991 (L'Intelligence Artificielle dans les Systemes
Electroniques Aero spatiaux)

(SOURCE): Advisory Group for
Aeronautical Research and Development, Neuilly-sur-seine
(France)

TO ORDER THE COMPLETE COMPILATION REPORT USE AD-M242025

THE COMPONENT PART IS PROVIDED HERE TO ALLOW USERS ACCESS TO INDIVIDUALLY AUTHORED SECTIONS OF PROCEEDINGS, ANNALS, SYMPOSIA, ETC. HOWEVER, THE COMPONENT SHOULD BE CONSIDERED WITHIN THE CONTEXT OF THE OVERALL COMPILATION REPORT AND NOT AS A STAND-ALONE TECHNICAL REPORT.

THE FOLLOWING COMPONENT PART NUMBERS COMPRISE THE COMPILATION REPORT:

AD#: _____ TITLE: _____

AD-

AD-P006 343

DTIC
 ELECTE
 NOV 13 1991
 S D D

Accession For	
NTIS	CRARI J
DTIC	AB
Classification	U
Justification	
By	
Distribution	
Availability Codes	
Dist	Avail and/or Special
A-1	U

AD-P006 343



AI FOR RPVs, SENSOR DRIVEN AIRBORNE REPLANNER (SDAR), FOR A ROBOTIC AIRCRAFT SENSOR PLATFORM (RASP)

DR. R. M. WILLIAMS AND J. J. DAVIDSON
 NAVAL AIR DEVELOPMENT CENTER
 CODE 505
 WARMINSTER, PA 18974-5000 USA

ABSTRACT

This report describes the Robotic Aircraft Sensor Platform (RASP) simulation developed at the Naval Air Development Center in Warminster, PA. It is extracted from the final report of a three year research effort into the architectures needed to develop real-time Artificial Intelligence (AI) techniques for autonomous aircraft. A hardware and associated software architecture has been developed to use on-board sensor information for high level AI decision making. The decisions then direct the flight path of the aircraft and camera gimbal through complex environments. This project has produced a system architecture that breaks the bottleneck of flyable real-time AI control systems. The work has transitioned into three new efforts; a flight test effort for the Unmanned Air Vehicle Joint Program Office, an investigation into use of SDAR for novel systems of sensors and platforms such as the Tactical Imaging System, and a study of applying this technology to manned platforms to assist human operators. ←

MISSION

The selected mission begins with a search of an area of ocean, scanning for ships. When a ship is sighted the aircraft must change its course and aim the camera. Avoiding danger in this environment requires the aircraft to maintain a safe stand-off distance from all ships. In addition, since most eyes on a ship point forward, the safer path of approach is around the stern. The payoff of the replanned maneuvers is one optimum picture of the ship, off her beam (side) and nearly filling the field of view. Losing interest after the best beam shot, the aircraft will carefully maneuver away and continue the search pattern for other ships.

It was realized early in the project that a hyper-brilliant "R2-D2" was beyond the scope of current technology. Other projects such as DARPA's ALV (Advanced Land Vehicle) and Pilots Associate, and the Air Force/DARPA RAV (Robotic Air Vehicle) had taken very ambitious approaches and were not performing as well as expected. These AI systems run on LISP language processors, requiring rooms full of computer hardware, and not performing in real-time. Instead we chose to investigate a system capable of providing sensor driven artificial intelligence

autonomous control using more efficient programming languages and processors.

The first year of the effort was spent researching the prior art, studying all known published autonomous vehicle work. There is a continuous range of vehicle autonomy ranging from tele-operated with a remote linked human making control decisions, to completely autonomous or robotic with no human involved after launch. We were interested in the completely robotic vehicles because use of the system by the Navy often requires radio silence. The robotic aircraft available or under development at this time can be grouped into broad categories by their mission. The most ambitious systems perform an overland attack mission, a very complex scenario requiring computing resources at the extreme limit of current technology. The other major branch of systems is intended for surveillance missions. The "smart" systems currently available are only waypoint flyers with no autonomous replanning capability. The developing systems using AI techniques are for the long duration, high altitude platforms that require AI logic for recognition of tactics and long term decision implications. Maneuvering of the aircraft in these systems is used to achieve long term goals.

The decision was made to develop a near-term system that provides a very definite improvement in the operation of short to medium range reconnaissance aircraft at medium to low altitudes. The charge-coupled diode (CCD) imaging camera was chosen as the primary sensor because it is small, passive and inexpensive, yet can provide unparalleled evidence of ship identity and disposition. The low and slow aircraft flight allows simple cameras and lenses to achieve good images of ships. The slow flight also gives the computer time to make decisions and allows the aircraft to turn with a small radius. RASP was restricted to over water missions because the images and scenarios are simpler than overland missions, yet the collected data is very useful to the Navy. The mission was then defined by a detailed analysis of flight phases. Aircraft piloting navigation modes and camera control modes were defined for each flight phase.

The Sensor Driven Airborne Replanner was developed to link sensor signal processing to traditional control systems. This sort of link is performed by air-to-air missiles when closing on a target. The difference here is that the RASP is required to perform target-related maneuvers that are much more cognitively complex than a missile end-game. This justified the artificial intelligence link between the sensor information database and the control systems. (see Fig.1)

SDAR

SDAR accepts the current camera azimuth, elevation, and focal length from the Camera System, the latitude, longitude, and length of the ship that is currently in the field of view (if any) from the Vision System, with the current RASP latitude, longitude, and altitude, roll, pitch, and yaw from the Navigation Sensors. Based upon that information, and upon data that has been saved from previous cycles, it chooses a state (see Section 2) and its associated arguments. Then, based upon that state, it generates commands to (1) the Autopilot, specifying the desired turn rate, (2) the Camera System specifying camera gimbal rates and focal length commands, and (3) the transmitter, specifying whether or not it should transmit. When in PHOTO state, SDAR also provides the transmitter with the length, speed, heading, and position of the ship that is being imaged.

SDAR must work intelligently and in real time. Yet it must be small, light, and inexpensive to be of practical use in unmanned autonomous vehicles (UAVs). Thus, exhaustive research on hardware/software architectures for real-time AI was conducted. The selection of the software architecture of SDAR, the programming language in which it is written, and the processor on which it will be executed was made for this research. This section describes those design decisions.

The Software Architecture

SDAR uses the AI "rule-based" software architecture. This architecture was selected because it accommodates change yet provides a very clear mental model of the logic process. That flexibility and clarity proved to be extremely valuable during the evolution of SDAR.

SDAR's inference engine uses "forward chaining". Forward chaining means the system works with facts to make conclusions. (Backward chaining systems try to prove tentative conclusions based on the facts.) The match phase inefficiency found in many forward chaining rule-based systems is avoided in SDAR by rule priority processing and by the high efficiency of the inference engine code. Also the rule base was minimized to avoid match phase time problems.

The forward chaining was selected primarily because backward chaining is inappropriate for SDAR. SDAR's data base is very volatile; the locations of ships and the location of RASP itself change constantly in real-time. Thus the contents of the data base constantly

change in real-time. The database is so volatile that it would be impossible for SDAR to complete non-trivial backward chains before the contents of the database changed. When the contents of the database change during the creation of a backward chain of subgoals, subgoals that have already been proven must be "reproven" based upon the new contents of the database. Backward chaining would thus be very complex and inefficient.

SDAR uses "procedural attachments" to its rules' conditions and actions. Procedures attached to rule conditions examine the Data Base, but also compute information from the raw data stored in the database and issue navigation sensor commands. Procedures attached to rule actions manipulate the database in addition to sending control signals directly to the Airframe, Transmitter, and the Camera System.

Procedural attachments were used for this research because they yield the advantages of both the rule-based and procedural software architectures. The AI rule-based architecture provides the flexibility needed to evolve intelligent behavior, but generally does not execute in real-time. The standard procedural architecture can often execute in real-time, but is relatively inflexible and high level goals are unclear. Procedural attachments in SDAR provide both the flexibility and clarity of AI systems and the speed of real-time systems.

Each time the inference engine is executed, the action of exactly one rule is fired, thus generating the appropriate control signals to RASP's effectors. The procedures that are attached to rule actions collectively form an asynchronous control system.

SDAR is a hybrid AI/control system in which both the AI component and the control component have been designed to work with each other. Control components normally have the "luxury" of executing synchronously; SDAR must, and does, execute asynchronously. AI components normally have the "luxury" of executing relatively slowly; SDAR must, and does, execute in real time.

The Programming Language and Processor

Many systems developed for real-time AI flounder because of the large size and high cost of the fielded hardware. The software techniques used by the theoretical AI researchers are, by necessity, very "fuzzy" structures. The very complex data structures result in very large memory requirements. Use of the data structures requires searching or chaining through long sequences of linked lists. Current AI specialized processors (i.e., TI/DARPA CLM and Symbolics IVORY) are optimized for these operations by tailored microcode, memory speed aids such as caches and pipelines, large bulk memory such as hard disks, and 20 to 30 MHz processor clocks. This hardware approach is power hungry, expensive and heavy for autonomous vehicles. The very features of Lisp that help with the design of AI solutions also cause the problems with

utilizing Lisp for real-time airborne systems.

SDAR is written in the Forth programming language. Forth is similar to LISP in that it uses lists as its primary algorithmic structure; it differs from LISP in that it uses stacks instead of lists as its primary data structure. The stacks are best handled by a postfix notation instead of LISP's prefix. With Forth, much of LISP's generality is possible.

Moreover, Forth has an enormous advantage with respect to hardware. A new processor, the RTX, has been developed that executes the Forth language primitives in hardware gate logic. Both LISP and Forth normally run under a virtual machine emulated on the actual host. The RTX is a realization of the virtual machine in use for Forth since 1969. This development allows high level Forth code to be executed at unprecedented speeds. The RTX is small and inexpensive thus fitting RASP's packaging requirements. SDAR was developed by this project for application on the RTX processor.

SDAR consists of eight components: database, rule base, inference engine, preplanner, rule compiler, attached condition procedures, attached action procedures, and database maintainer.

Inference Engine

The inference engine is the heart of SDAR. It implements the match, conflict resolution, fire, and repeat phases of the forward chaining algorithm. The inference engine implements the match and conflict resolution phases by iterating through the rules in the SDAR rule base to find the "best" rule. The "best" rule is defined as the rule with highest priority among all rules whose condition is true.

For each rule that is examined, the inference engine first determines if its priority is greater than the priority of the best rule found so far. If it is not, then the rule's condition is not evaluated. If it is, then the rule's condition clauses are evaluated one at a time.

For each condition clause that is evaluated, a call is made to a short machine code routine that tests the result. A false clause causes immediate clearing of the stack elements pushed by the other condition clauses of the rule. (This is to allow condition clauses to pass parameters to the action clauses without concern for the order of the logic.) The false clause also clears the return stack to abort out of the rule execution and return to the inference engine for consideration of the next rule. A true clause simply allows the condition part of the rule to continue executing. If the last condition clause evaluates true, then the current rule is labeled the best rule, and the executable address of the rule's action is saved. The process continues until all the rules have been examined.

After the match and conflict resolution phases are finished, the inference engine fires the best rule. It does so via a subroutine call to the executable address of the best rule's action part. When the action has been performed the inference engine returns control to the high level word which initiates another execution of the inference engine.

In traditional rule-based systems, the match and conflict resolution phases are distinct. In the match phase, the inference engine evaluates all of the rules' conditions (thus forming the conflict set) with no use of the rules' priorities. Then, if the conflict set contains more than one rule, the conflict resolution phase examines the priorities of those rules to determine which to fire. (The rule ordinal position is not useful for resolving the conflict because the rules are order independent.) Thus in traditional rule-based system, the match phase is inefficient because every rule condition is evaluated every time the inference engine is executed. This is very time consuming for a system with many rules (>2000).

The "Rete Algorithm" is a highly respected approach for efficiently implementing the match phase. Rete uses the fact that most rule conditions use a very small portion of the data base. If the pertinent data has not changed since the last execution of the inference engine, the rule condition must have the same logical value. The idea is to only evaluate the rule conditions that have a chance of changing their state. Therefore, all changes to the data base are monitored, which is simple if only rule actions can change the database. The Rete algorithm must have compiled tables defining which clauses or rule conditions can be affected by given database changes.

The Rete algorithm only helps when the database changes slowly. In SDAR this condition is not met. The conflict resolution phase is unchanged by Rete. The rule priorities are not considered until after the conflict set is formed.

As described above, SDAR's inference engine interleaves the match and conflict resolution phases for efficiency. The idea of examining a rule's priority before its condition was invented (independently at NAVAIRDEVCON, although perhaps not exclusively) in the course of developing the SDAR inference engine. This method evaluates a rules condition clause only if the associated priority is high enough. It is believed that this algorithm has more speed-up potential than Rete for sensor driven database applications.

Preplanner

The Preplanner is executed before RASP is launched and is the interface between the mission planner and RASP. Through the preplanner, the mission planner communicates the ship importance specification and the mission plan to SDAR.

More precisely, the preplanner assigns values to the arrays which relate a ship's length, speed, heading, distance, and

91-15525



91 1113 020

direction to its length-value, speed-value, heading-value distance-value and direction-value (respectively). It also assigns values to the variables used to combine those individual values into a total value: length-factor, speed-factor, heading-factor, distance-factor, direction-factor, #photo-sessions-factor, interest-factor and handiness-factor.

The preplanner assigns values to the arrays used to compute length-value, speed-value, and heading-value to length-factor, speed-factor, and heading-factor, via an interactive session with the mission planner. The preplanner simply assigns constants to all other arrays and variables.

The preplanner assigns a value to the mission plan through a similar interactive session.

Rule Compiler

The SDAR rule compiler takes condition/action rules and produces executable object code. The rules are compiled into two data structures in memory. The actual object code that fetches or calculates the truth status of a clause is compiled the same as all the other software in the processor. In addition the executable addresses of the condition and action rule parts are stored in a data array called the rule list. The rule priorities are also stored in the rule list. The object code for the rule clauses is compiled as call address tables for the procedural attachments. The last call in each clause is to a machine code routine to test the clause truth and directly manipulate the return stack accordingly.

The compiler takes full advantage of the programmer's access to compiler primitives in Forth. The SDAR rule base compiler takes pages of ASCII text source code and produces executable machine code optimized for execution by the inference engine. The programmer's access to the fundamental operating system variables and the control the programmer has over the structure of the dictionary facilitate the detailed manipulation of the machine code laid down by the compiler.

The last crucial capability used in the SDAR rule compiler is the control of the execution time of a defined word. In the SDAR rule compiler a three distinct "run-times" are used, corresponding roughly to "passes" in a standard compiler. The time phases are 1) during the compilation of the compiler 2) during execution of the compiler, that is, compile time for the rules 3) during the execution of the compiled rules. A lower level word defined for use in a compiling word may be easily constructed to affect the dictionary and the stack at any of the three time phases. It is these words executing at the two compile times and manipulating the dictionary that makes the SDAR rule compiler simple.

Simulation of RASP

A computer simulation, incorporating mathematical models of the non-SDAR components of RASP and its environment, was produced to provide a test bed for

SDAR development. The major components of the simulation and their relation to the SDAR is shown in Figure 2.

The simulation was developed separately from SDAR. In order to provide the most realistic test for SDAR the simulation variables such as aircraft position are not directly available to the SDAR system. Simulated navigation sensors have access to the perfect knowledge in the simulation models and make that information available to SDAR after including error modeling. Initially the real-time control of the Airframe and Camera System was done manually. Next, the autopilot was implemented and tested under various waypoint configurations and wind conditions. Finally, SDAR was included and rules were loaded incrementally to debug the RASP behavior one state at a time.

The simulation was run repeatedly with different scenarios to judge the appropriateness of SDAR's decisions. As a result, SDAR's rules were edited several times. The flexibility of the rule-based approach made it a simple matter to do such editing. The maneuver efficiencies and stabilities were also tuned during the system integration phase.

Before the simulation begins, the user executes the preplanner to enter a mission plan and a ship importance specification. The preplanner displays a map-view of an area of ocean on a video screen. The user then specifies a mission plan by repeatedly using cursor keys to choose points on the map. Each chosen point thus becomes a waypoint of the mission plan. As the cursor moves, the screen displays its position in miles. The generated list of waypoint positions can be stored on disk with a keystroke. These can also be easily edited.

The user enters a ship importance specification through a four-phased dialogue using a keyboard and video screen. In Phase 1 the user specifies which ships (in terms of their lengths: are large ships important or small ones?) should be considered important and he does so by interactively filling in a table that relates each feasible ship length to a number in the range 0 to 100 (the "length value" of a ship with that length). Phase 2 does the same job, but for ship speed. (Are fast ships important today? Slow ones?) Phase 3 does the same job, but for ship heading. (Are ships heading north important today? East?) In Phase 4, the user specifies the relative weights that should be used by RASP to compute a ship's importance from its length value, speed value, and heading value, as defined in Phases 1-3. The user also specifies RASP's environment. The positions, headings, and speeds of ships are specified through a process that is similar to the mission plan specification. The user can specify the direction and speed of the wind.

During the simulation, the behavior of RASP and the ships is displayed on three video screens: the map view screen, the airborne camera screen, and the system status screen. The map view screen

graphically displays a map of an area of ocean. It shows (1) the waypoints of the mission plan, (2) the track of RASP's platform through time, (3) the "footprint" of RASP's camera on the ocean's surface through time, (4) the actual track of each ship through time, and (5) the estimated track of each ship that has been sighted but is currently not in the field of view through time. The map view screen also indicates when RASP is transmitting to the home ship by "beeping".

The Airborne Camera screen graphically displays what is seen by RASP at each instant. More precisely, when a ship is in RASP's field of view, a synthesized image of the ship appears on the airborne camera screen. The orientation of the icon is correct, and its length is accurately scaled. The system status screen is an alphanumeric display that shows data concerning the current state of RASP.

CONCLUSIONS

This project has resulted in a system that breaks the bottleneck that has previously prevented use of artificial intelligence for real-time control applications in unmanned air vehicles (UAVs). A P I solution has been developed at NAVAIRDEVGEN to make dumb sensor platforms (i.e. Remotely Piloted Vehicles (RPVs)) into smart UAVs in the early 90s. This work has led to our role in a Tri-Service Joint Program Office effort to flight test autonomous aircraft maneuvering to optimally position sensors. This work has also led to novel systems of sensors and platforms potentiating systems such as the Tactical Imaging System that can transmit imagery over satcom links. In addition a variant of SDAR is being developed to assist C operators on manned aircraft.

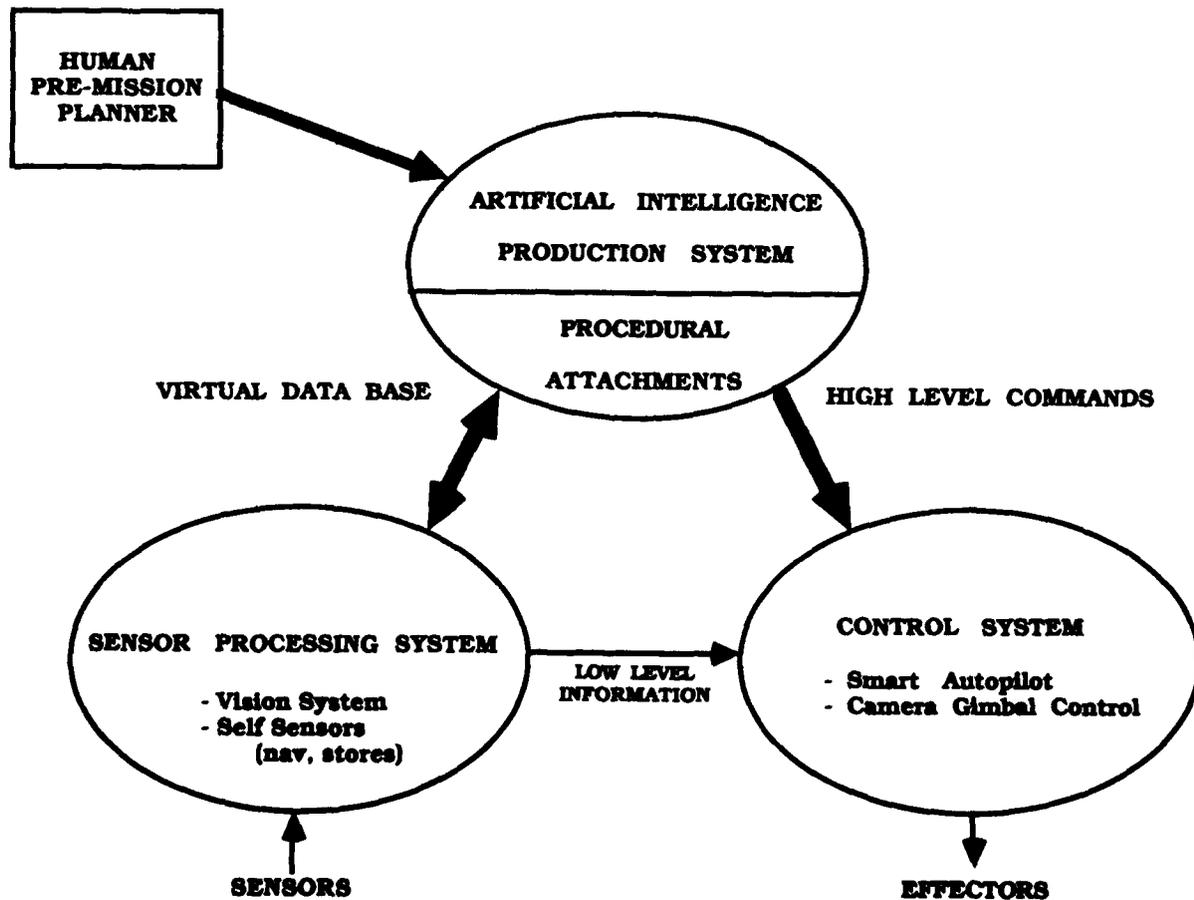


Figure 1. Sensor Driven Airborne Replanner System Architecture

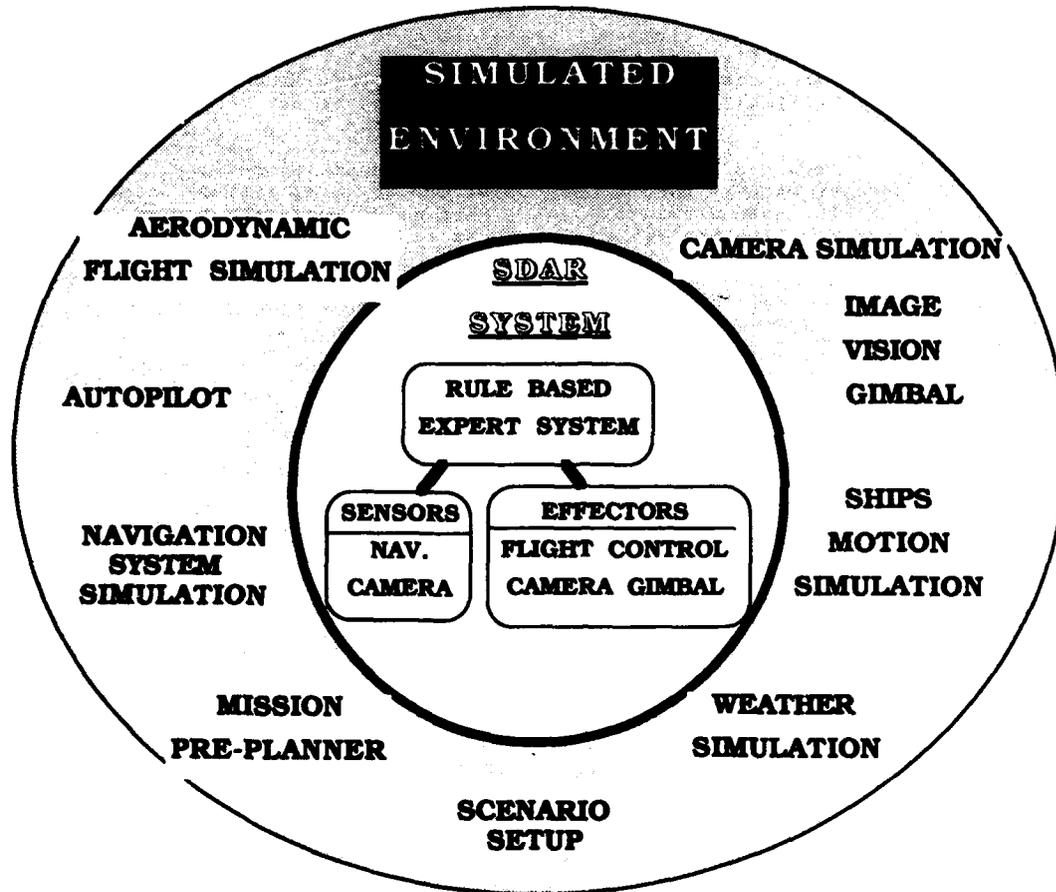


Figure 2. SDAR Laboratory Simulation

Discussion

I. F. Vagnarelli, Italy

- (a) What is the flight altitude?
- (b) Is the system usable in all weather?
- (c) Is the system equipped for ECM?

Author:

- (a) The altitude used is 6000 ft.
- (b) Only good weather is included for the environment.
- (c) No, the system is currently used only for civil applications.



COMPONENT PART NOTICE

THIS PAPER IS A COMPONENT PART OF THE FOLLOWING COMPILATION REPORT:

(TITLE): Conference Proceedings of machine Intelligence
for Aerospace Electronic Systems Held in Lisbon, Portugal on
13-16 May 1991 (L'Intelligence Artificielle dans les Systemes
Electroniques Aeroespaciaux)

(SOURCE): Advisory Group for
Aeronautical Research and Development, Neuilly-sur-Seine,
(France)

TO ORDER THE COMPLETE COMPILATION REPORT USE AD-N242025

THE COMPONENT PART IS PROVIDED HERE TO ALLOW USERS ACCESS TO INDIVIDUALLY AUTHORED SECTIONS OF PROCEEDINGS, ANNALS, SYMPOSIA, ETC. HOWEVER, THE COMPONENT SHOULD BE CONSIDERED WITHIN THE CONTEXT OF THE OVERALL COMPILATION REPORT AND NOT AS A STAND-ALONE TECHNICAL REPORT.

THE FOLLOWING COMPONENT PART NUMBERS COMPRISE THE COMPILATION REPORT:

AD#: _____ TITLE: _____

- AD-P006 331
- AD-P006 334
- AD-P006 336
- AD-P006 344
- AD-P006 345
- AD-P006 346
- AD-P006 352

DTIC
S NOV 13 1991 D

For information on the availability of this report, contact the Distribution Statement Center, 1215 Jefferson Davis Highway, Alexandria, VA 22304-6145.

Accession	
NTIS	DTIC
DTIC	DTIC
Unannounced	Justification
By	
Distribution	
Availability Codes	
Dist	Avail and/or Special
A-1	

AD-P006 344



A THREAT MANAGEMENT SYSTEM

by
K. HOLLA, B. BENNINGHOFEN
MBB-Military Aircraft Division, FE37
P.O. Box 80 11 60, 8000 München 80
Germany

1. SUMMARY

TMS - the Threat Management System - is a company prototype development of a support system for the crew of fighter bombers at low level flight. In particular TMS is supposed to provide support in high dense groundbased airdefence scenarios during the

- 1) preparation phase,
- 2) penetration phase and
- 3) attack phase

of a mission applying methods of terrain and threat avoidance combined with sensor information fusion.

2. INTRODUCTION

In reaction to growing capabilities of modern airdefence systems both groundbased and airborne and thus to increasing requirements for mission enforcement modern military aircraft are characterized by increasing system complexity. The composition of a growing number of demanding subsystems imposes additional workload and thus in consequence operational problems to the crew of such integrated systems. In order keep these complex systems manageable by the crew also in future operating in complex scenarios several national programs like Pilot's Associate, Mission Management Aid, or Copilote Electronique have been launched to investigate solutions by applying new software technologies like Artificial Intelligence in particular knowledge based systems. TMS is a project within this scope.

2.1 Scenario

High dense groundbased airdefence scenarios are best described by multiple layers of overlapping threat areas given by the active radii of the missile systems. Such airdefence belts equipped with very effective weapons have to be penetrated in order to get to preplanned targets.

Today's general procedure for penetrating such scenarios is the combination of high velocity, low set

clearance height and the application of any (onboard) ECM as appropriate.

2.2 Cockpit Situation

Increasing system and scenario complexity means that the crew has to cope with larger amounts of data in shorter time by correlating this information flow mentally under stress conditions. In case of a single event they might be able to manage the situation but for several simultaneously appearing problems it is almost impossible for human beings to solve the problems taking into account all effecting constraints. Therefore it becomes more and more important to provide suitable onboard support systems.

2.3 Development Goals

Onboard support systems for the crew of future airborne weapon systems need to provide

- comprehensive situation awareness
- tactical recommendations for decision making

in order to avoid human beings becoming the limiting factor for system performance, but put man into action where he is unique as system manager and final decision point.

Modern information technology, particularly methods of Artificial Intelligence, allow supervised autonomous functionality on lower levels of intellectual capability like perception, deduction, rulebased knowledge processing, and execution of well defined tasks. Therefore functions like search, control, transformation, interpretation, evaluation, and also planning can be taken over by systems based on such modern software technologies in combination with conventional algorithmic solutions.

By support of such systems the workload in the cockpit implied by many low level operational tasks will be reduced and the crew can concentrate on high level decisions based on compact, correlated information, what leads to an improvement of

- survivability and
- mission success.

The main goal of the system is the onboard support for the crew in critical situations. Only if advised by the crew or in well defined situations TMS should take over the aircraft control.

2.4 Application Areas for Onboard Knowledge Based Systems

Based on the above described situation several distinct application areas of complementing knowledge based systems in combination with algorithmic procedures can be derived.

Providing the best available scenario information is one essential task. The information from several sources at different levels of detail gained at different times and affect by some degree of uncertainty has to be combined to a consistent description of the threatening scenario the aircraft has to operate in.

Next this detected scenario has to be assessed in terms of threat levels for the aircraft having in mind the capability of the aircraft and its onboard resources and weighting this against the features of the groundbased airdefence systems. A complete situation assessment requires also detailed knowledge about the status of the own platform combined with information about limitations in flight envelope or sensor support in due to subsystem failures. The emergency procedure management has to be taken over by the system.

Based on this evaluation an onboard situation dependent plan for an flight route through the scenario can be generated in order to minimize the exposition to threats. Terrain masking based on digital terrain data and the threat scenario in combination with silent navigation like TRN (Terrain reference navigation) is a proper mean for covert operation and thus for threat avoidance. The planning system has to keep in mind the very strict limitations of time and fuel management for a mission.

In order to suppress the effectiveness of the airdefence systems of the given scenario a planning system for tactical measurements provides advice for the application of onboard ECM equipment, flight maneuver, and weapons in combination with a resources management.

Of particular importance for manned aircraft is the man-machine-interface. In situation dependent formats providing just the required information the decision base for the crew has to be displayed. User modeling allows for an adaption to the level of experience of the crew if appropriate.

3. THE THREAT MANAGEMENT SYSTEM

TMS covers a particular area out of the set of possible applications of software technology for military aircraft based on the above outline. Scenarios in which TMS provides its support are best described by high dense layered groundbased airdefence belts like the so called European scenario (figure 1).

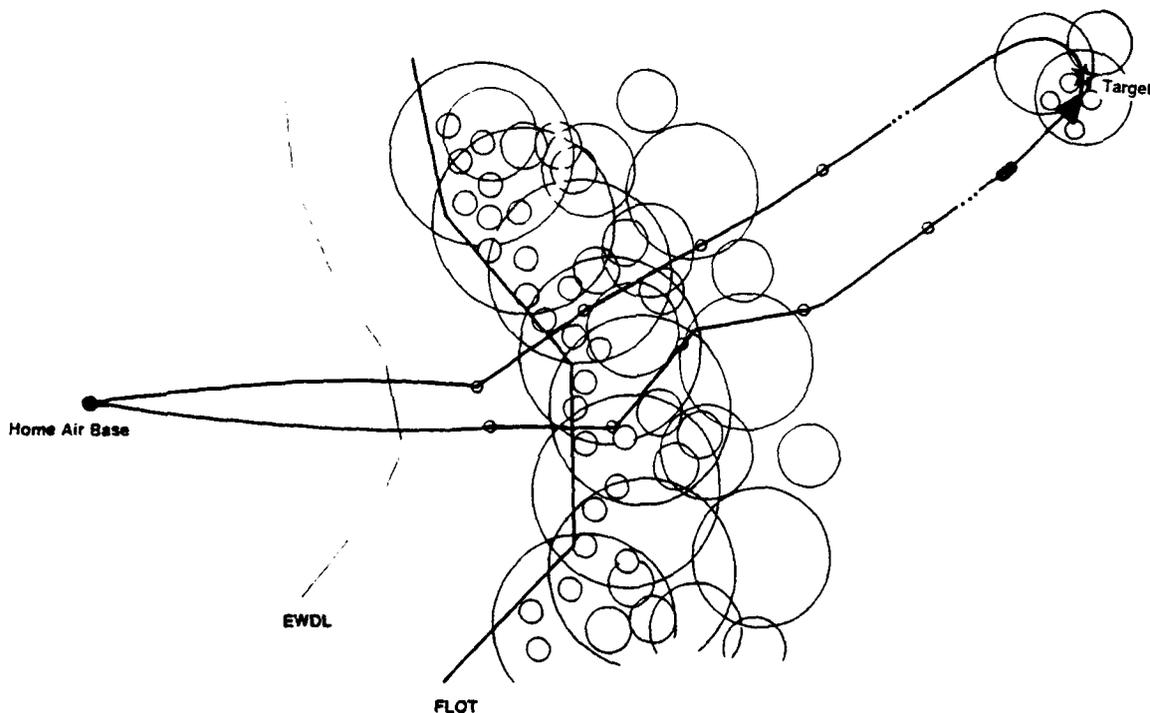


Figure 1: Example of an European scenario

The realized basic functionality of TMS is

- recognition of the actual scenario
- threat assessment
- planning/replanning of an optimized flight-path
- tactical display.

3.1 TMS Concept

Three main-components form the Threat Management System (figure 2)

- Data/Information Fusion
- Situation Assessment and Plan Generation
- Man/Machine Interface

These components are embedded in a simulation environment for development, test and demonstration of TMS outside a real target system.

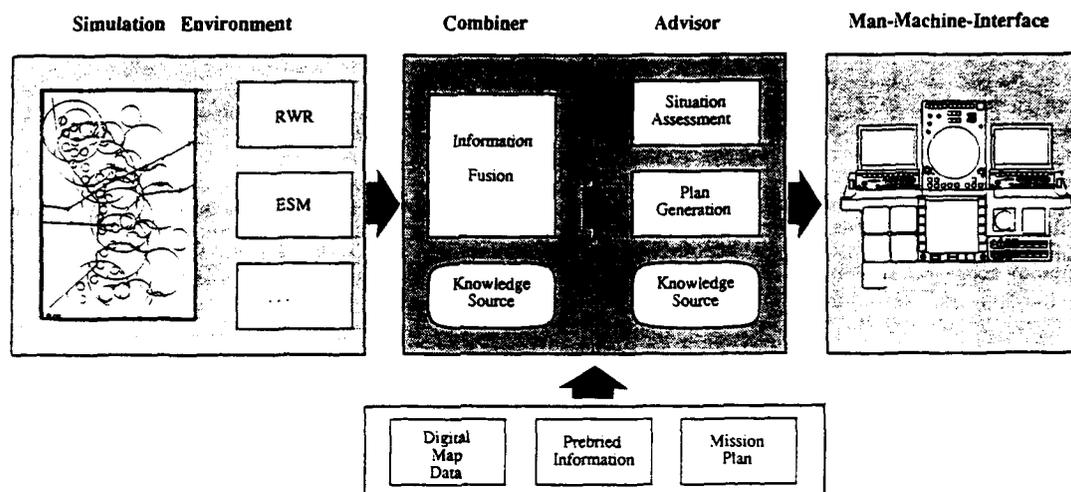


Figure 2: System Concept

3.2 Simulation Environment

For TMS a quite elaborated dynamic simulation environment has been developed in order to feed TMS with realistic data as far as its functionality is concerned. Such an environment allows for system verification up to a high degree.

The simulation environment is composed of

- digital terrain and cultural data
- groundbased airdefence sites
- dynamic emitter models
- aircraft model
- ECM sensor models

and provides features for setting up any ground-based scenario out of the list of available airdefence sites. Each emitter might be assigned its specific parameters including its scanpattern within a selected sector. The aircraft model can be equipped with one or more ECM sensor model providing position and attitude at each instance to the sensor models for evaluation of the receiving conditions.

The simulation is realized by means of object oriented programming completely. Classes for sites,

emitters, aircraft, sensors etc. have been created. They include the behavioral description of an object and inherit it to son objects.. Instances describe a particular realization of an object. Actions are initiated by messages.

For a future development a knowledge based module will control a tactical behavior of the threats, e.g. control of moding depending on the approach of the aircraft and doctrines..

3.3 Information Fusion

The growing number of information sources onboard military aircraft requires means for merging the information about the same object from different sources to one comprehensive display as information source for the crew. Beside active and passive onboard sensors operating in different frequency slots and directions with different sensitivity and resolution radio links to remote sources are in use. Also intelligence information about the different orders of battle (MOB, EOB, AOB, GOB) as briefing data as well as digital terrain and cultural data are important information sources for deriving best knowledge about the actual threat scenario.

T20 SIT 16

91-15526



3.3.1 Fusion Concept

The TMS fusion component is based on information from passive sensors alternatively one or both

- RWR - Radar Warning Receiver
- ESM - Location Receiver

and on the two data sources

- prebriefed scenario
- digital terrain data.

Other sources can be integrated but are not in use right now.

At the first instance of operation of the fusion unit its output is identical with the contents of the prebriefed scenario. If time proceeds it will be updated continuously by actual information about the real scenario.

A Basic Combiner provides fundamental functionality like the administration of the incoming information from different sources, the keeping of the combined tracklist, the resolution of identification conflicts, and the management of higher levels of the fusion process. TMS knows three levels of specific fusion operations

- the geometric combiner
- the geographic combiner
- the tactical combiner.

with each of them requiring an increasing amount of processing time. The geometric combiner processes primarily identification-, direction- and distance information provided by the different sources. The location of an emitter measured by a sensor is attached with an uncertainty about its real position. The geographic combiner reduces the location error by analyzing terrain-, surface-, and cultural data of the surrounding area. Finally the tactical combiner processes knowledge about the Ground Order of Battle and the equipment of the related units as well as doctrines for deployment and tactical behavior.

The information fusion can be enhanced by an additional sensor management unit, which coordinates situation dependent the usage of the different onboard sources by techniques of mode selection and sensor cueing.

3.3.2 Fusion System Architecture

The selected architecture for the TMS information fusion unit is based on blackboard system concepts

The blackboard, a global database, keeps the actual problemsolving-state. It consists of inputdata, partial solutions, alternatives and final solutions, all of them objects of the solution-space. They are hierarchical organized into levels of analysis; therefore sensordata arrive at the lowest level. The datastructure on the blackboard is a dynamic entity that evolves over time representing the cur-

rent best hypothesis. The knowledge-sources (geometric, geographic, tactical) respond opportunistically to changes on the blackboard driven by the focus of attention.

3.4 Advisor

3.4.1 Situation Assessment

Based on the output information of the fusion unit, the combined scenario, TMS assesses the degree of danger of separable areas of the scenario dependent on threat-type and areas of overlapping fighting zones. The result of this assessment is a threat-map displaying weighted threat-areas on a digital map. We have selected circles representing the fighting zones of threats because the more dedicated pattern depend on the direction an aircraft is approaching a site and this information isn't available for the first iteration.

Once an aircraft is tracked by an emitter its primary task is to unlock the track before a missile can be launched. One mean for this is to use areas masked by terrain for the flight route, i.e. areas which can't be controlled by a radar because of the structure of the terrain. To provide this terrain and threat avoidance functionality TMS calculates a more detailed threat-map where both threat deployment and terrain masking is considered (figure 3).

Part of situation assessment is the actual knowledge about the status of the own platform, because any limitation due to subsystem failure decreases the capacity of the aircraft and thus effects the plan generation massively. System status evaluation isn't in the scope of TMS right now.

3.4.2 Tactical Advice

Based on the previous described situation assessment suitable measures can be planned to reduce the risk for crew and platform. Generally the procedures

- avoidance
- reduction of detectability
- deception and jamming
- fighting

are possible. The tactical advisor selects and combines these measures situation dependently.

Tactical experience is heuristic knowledge gained in a multitude of simulation, training flights, and sorties continuously being updated and adapted to new developments. Thus despite some general procedures the knowledge base of an tactical advisor must be tailored to the respective configuration of the aircraft and to the knowledge about the capability of the systems in the scenario.

TMS is supposed to provide both capacities route planning/replanning and plan generation for use of onboard resources in combination with appropriate

maneuvers.

3.4.2.1 Rerouting

Detected differences to the prebriefed scenario or sudden changes require a replanning of the preplanned mission route based on the mission goals and the actual knowledge about the scenario. TMS provides a route planner that can be applied for global mission route planning as well as for local rerouting. It takes into account constraints for time- and fuel-consumption and makes sure that the replanned route is in the neighborhood of the preplanned mission route. Similar important is that the route planner inherently avoids routes which lead into a dead end. Critical sections of the flight path can be seen on the display (figure 3).

Route planning for terrain and threat avoidance requires not only lateral but also vertical route components. The consequence is that a modified 3-D

flight control algorithm has to be developed which allow to follow such a preplanned route.

3.4.2.2 ECM Advisor

Route planning and replanning reduces the exposition of the aircraft to threats and minimizes the time-interval of crossing a fighting zone, but it doesn't eliminate the threat. Therefore it is important to plan the employment of onboard resources in combination with flight maneuvers. Onboard resources are limited therefore it is important to employ them purposeful adapted to the current situation and to manage their consumption having the needs for the overall mission in mind. Unplanned flight maneuvers within a highden. scenario are very critical because they might directly lead into an exposition to several threats.

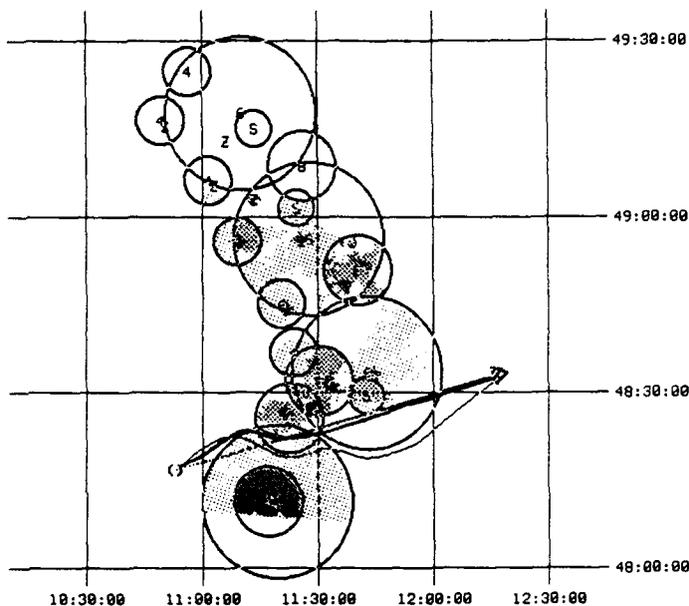


Figure 3: Tactical Display

3.5 Man-Machine Interface

At the man-machine interface the required information has to be provided to the crew of airborne platforms situation dependently in compact but comprehensive form. A combination of a 2-D top-down view tactical color display providing the overview about the scenario on top of a map and the planned route over a long distance (figure 3) and a perspective 3-D display showing the situation for the next seconds of flight are in use with TMS respectively under development.

4. DEVELOPMENT ENVIRONMENT

The TMS prototype completely is developed in

CLOS (Common Lisp Object System) for means of flexibility, maintainability and control of complex data structures. Therefore a development environment as shown in figure 4 is applied. Parts of TMS meanwhile could be proved providing valid functionality and therefore are in the process of being ported to target processors like 680xx or R3000.

5. REAL-TIME ASPECTS

At least three levels of real-time have to be considered

- Reaction times less than one second: A new upcoming threat has to be displayed immediately to the crew. Fast algorithms for in-

- formation fusion can be applied.
- Response times at about one second: The route planner for example plans a new route over 100 miles within this time-frame. Because route replanning is initiated by changes in the scenario or by request of the pilot this processing time seems to be reasonable.

- Several seconds of reaction time: Functions like trend monitoring allow several seconds of update.

Thus in each case the appropriate time scale has to be analyzed and its design has to be adapted to the allowance in functionality.

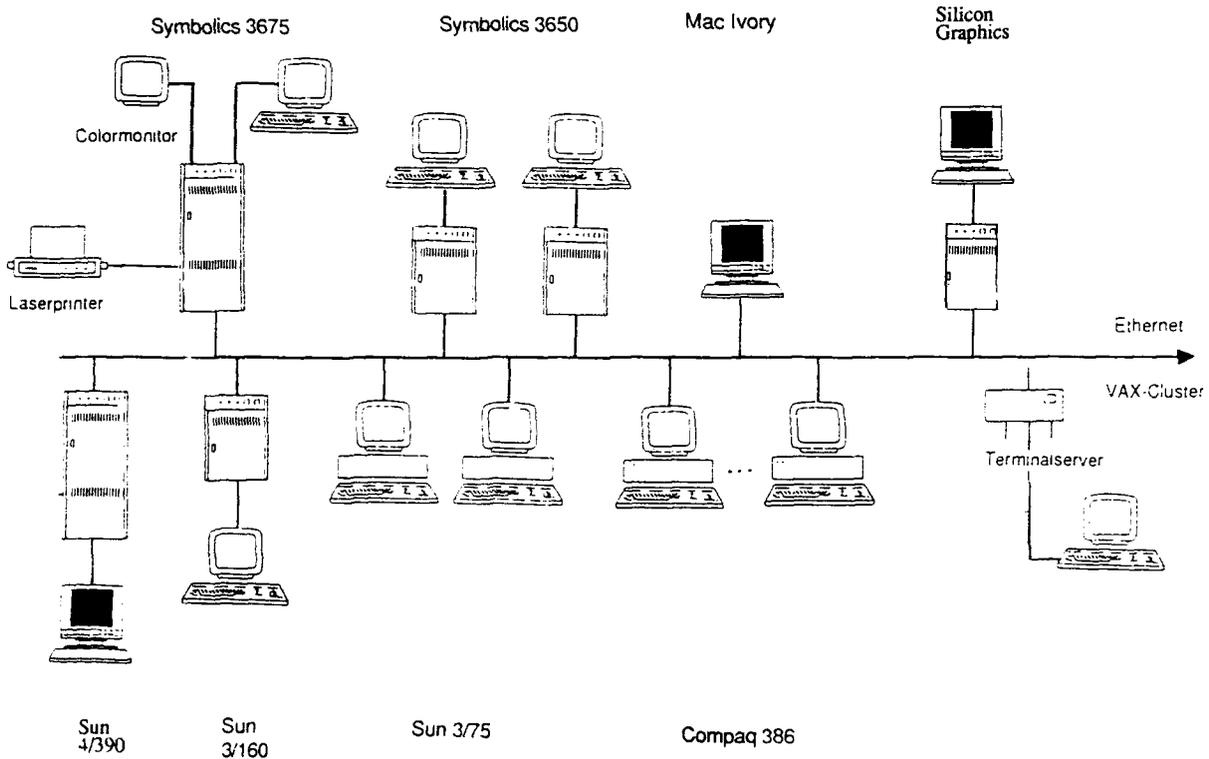


Figure 4: Development Environment

6. CONCLUSIONS

Future developments in avionics systems will adopt integrating support systems like TMS in an evolutionary way. For onboard application of knowledge based systems a wide area of functionality has been identified and at least as prototype solution realized. The major challenge for the next future will be the development of operational systems in target environments.

Discussion

1. Dr K. Holla, Germany

The air defenders will react to the intelligent threat reaction systems, and a viable strategy becomes using "spoo" or false threat emitters to try and channel the aircraft to heavily defended corridors. Do you consider a cost association with reacting to emitters?

Author:

From my view to beat this problem you need at least three information sources (knowledge sources): Intelligence information (prebriefed data to know where sites have been seen, also if they are currently not active; Remote sensor information about communication activities to know the communication links and the superior control section; and knowledge about doctrine and tactical behavior to be prepared for such "spoofing".



INTELLIGENCE ARTIFICIELLE APPLIQUEE

AU DIAGNOSTIC DE PANNES

Michel COURTOIS
DASSAULT AVIATION
DIVISION SYSTEMES D'ARMES
78, Quai Marcel DASSAULT
BP 300 92552 SAINT CLOUD CEDEX
FRANCE

Gilles CHAMPIGNEUX
DASSAULT AVIATION
DIVISION SYSTEMES D'ARMES
78, Quai Marcel DASSAULT
BP 300 92552 St-CLOUD CEDEX
FRANCE

Bernard DUGAS
DASSAULT AVIATION
DIVISION ETUDES AVANCEES
78, Quai Marcel DASSAULT
BP 300 92552 St-Cloud CEDEX
FRANCE

1. RESUME

Il est généralement admis que la moitié du coût d'utilisation des avions de combat correspond aux dépenses concernant la maintenance : recharges, nombre d'heures de maintenance par heure de vol, etc... Pour réduire ou maîtriser ces coûts, il est essentiel de développer un concept de maintenance permettant une automatisation et une fiabilité maximales des opérations de diagnostic. De plus, ces diagnostics doivent être les plus fins possibles, le but final étant de tendre vers une maintenance deux niveaux (maintenance en ligne et maintenance industrielle). Pour tenir ces objectifs, l'utilisation de l'Intelligence Artificielle appliquée à la détection et à la localisation de pannes, fait l'objet de nombreuses études et semble être une voie intéressante pour la résolution du problème posé. Le système de diagnostic développé sur les avions d'armes de DASSAULT AVIATION repose sur le concept de la maintenance intégrée et l'utilisation de l'intelligence artificielle est bien adaptée à ce concept. La principale caractéristique de cette maintenance intégrée est de réaliser la détection et la localisation de pannes par vérification de l'intégrité des éléments matériels et logiciels constituant les différentes chaînes fonctionnelles. Ce principe est utilisable pour la maintenance sur avion (en ligne) et en atelier (hors ligne et maintenance industrielle). Les performances intéressantes relevées sur les avions actuellement en service peuvent être améliorées de façon sensible par l'utilisation de l'Intelligence Artificielle à différents niveaux.

2. INTRODUCTION

Un système performant de détection et localisation des pannes doit avoir pour objectifs :

- l'Accroissement de la sécurité des vols
- l'utilisation optimale des systèmes de l'aéronef en cas de panne
- la détection et la localisation des pannes dans un temps le plus réduit possible (proche du temps réel)
- une fiabilité du diagnostic la plus élevée possible (taux de fausse dépose le plus réduit possible)
- une finesse de diagnostic importante (diagnostic au niveau de l'unité remplaçable en atelier à l'intérieur de l'unité remplaçable en ligne)

Ces objectifs doivent être atteints si l'on veut

- diminuer les temps de remise en état
- diminuer le coût en recharges
- diminuer le nombre et la qualification des mécaniciens.

Pour obtenir le résultat recherché, il est primordial qu'au début du développement d'un système d'armes, il soit établi la politique de maintenabilité maintenance la mieux adaptée pour l'ensemble des constituants, tous niveaux de maintenance confondus.

Le principe de maintenance Intégrée de DASSAULT AVIATION a commencé à être étudié en 1978 et a depuis été appliqué à tous les systèmes d'Armes développés depuis cette date. Ce principe est entièrement algorithmique et a des performances très intéressantes. En parallèle, nous avons réalisé un certain nombre d'études sur des systèmes experts destinés au diagnostic de pannes. L'expérience acquise grâce à ces études n'a pas remis en cause le concept de maintenance Intégrée, mais a au contraire montré qu'une utilisation conjointe de ce type de maintenance et de l'I.A. permettait d'envisager un accroissement important des performances. Nous allons donc décrire le principe de la maintenance intégrée et indiquer à quels points de son organisation il est intéressant d'utiliser des systèmes experts.

3. PRINCIPE DE DIAGNOSTIC PAR TEST D'INTEGRITE

Le principe essentiel sur lequel repose la maintenance intégrée est le test d'intégrité qui peut être défini comme suit :

- quand, pour un équipement, on teste séparément l'intégrité du logiciel et l'intégrité du matériel, l'ensemble de ces deux tests est équivalent à un test d'intégrité global
- quand, pour un équipement, on teste séparément son fonctionnement interne et ses interfaces avec l'extérieur, la somme de ces deux tests est équivalente à un test d'intégrité global
- quand, pour un équipement, un test garantit l'intégrité de l'équipement, ce test garantit le bon fonctionnement de l'équipement et il n'est donc pas nécessaire de simuler le fonctionnement pour valider la fonction de transfert de l'équipement.
- Dans la mesure où le fonctionnement interne de l'équipement peut être modifié à des fins de recherche de panne, ce fonctionnement peut être utilisé pour tester valablement les interfaces de l'équipement avec l'extérieur. Exemple : en faisant exécuter séparément dans l'équipement une fonction destinée uniquement à en valider les entrées/sorties et un test d'intégrité de la fonction de transfert, on a validé l'équipement aussi bien que par un test complet par simulation de fonctionnement.

Ceci suppose que la mise au point de l'équipement a été menée correctement, ce qui est l'hypothèse de base du raisonnement.

Ces principes ont pour conséquence :

- de permettre des tests permanents dans les équipements. En effet, un test d'intégrité sur le fonctionnement interne de l'équipement peut n'avoir aucun effet sur les interfaces de celui-ci et ne perturbe donc pas le reste du système
- de simplifier la mise en oeuvre : un test par simulation de fonctionnement suppose que si un équipement appartient à une chaîne, toute la chaîne participe au test, alors qu'un test d'intégrité interne est localisé dans un équipement.
- de permettre un fonctionnement spécifique des équipements à des fins de contrôle de l'intégrité des interfaces. Si ce fonctionnement est étendu à l'ensemble du système, il est alors possible de tester les interfaces des équipements sans se préoccuper des chaînes fonctionnelles.
- de simplifier de façon extrêmement importante la base de connaissance d'un système expert dédié au diagnostic de pannes. En effet, les modèles comportementaux des équipements se réduisent à des modèles de fonctionnement électronique de chaque équipement sans se préoccuper des fonctions de transfert assurées par logiciel.

Il est, par ailleurs, important de souligner les points suivants :

- Les tests de sécurité des équipements sont de toute façon obligatoires pour assurer la sécurité du vol. De la même manière, des tests sont nécessairement effectués pour prévenir le pilote des dégradations du fonctionnement opérationnel des équipements. Si ces tests sont convenablement conçus, leurs résultats peuvent également servir au diagnostic de pannes, sous réserve qu'ils satisfassent aux performances demandées.
- Les tests destinés à un échelon de maintenance donné peuvent souvent être utilisés avec profit à un autre échelon.
- Si un équipement est inclus dans une chaîne fonctionnelle, et s'il existe un moyen de dialogue entre les équipements de la chaîne (Bus numérique multiplexé par exemple), cet équipement devra contribuer à la maintenance globale de la chaîne (par exemple, le test des sorties de cet équipement pourra fournir des informations pour le test des entrées de l'équipement suivant dans la chaîne).

3.1. PERFORMANCES DEMANDEES AUX EQUIPEMENTS

Dans la construction de la maintenabilité

globale d'un Système, il est demandé que chaque équipement soit conçu avec les objectifs suivants :

- Utilisation maximale des capacités internes des équipements dans le but de limiter et de standardiser les outillages de maintenance
- Orientation des tests vers la localisation de l'élément en panne
- Contribution des équipements à la maintenance de l'ensemble du système

Par "élément" on entendra :

- l'Unité Remplaçable en Ligne (URL) pour la maintenance en Ligne
- l'Unité Remplaçable en Atelier (URA) pour la maintenance hors ligne
- le composant élémentaire pour la maintenance industrielle.

3.2. LIMITATIONS

Il est évident que l'ensemble des dispositions de testabilité applicables à un équipement conduit nécessairement à des augmentations de matériel et/ou logiciel embarqués. Ces augmentations doivent être contrôlées et réfléchies car elles peuvent entraîner :

- Une diminution de fiabilité
- Une diminution de sécurité
- Une augmentation de poids ou de volume
- Une augmentation du prix

Il est donc nécessaire dans tous les cas d'examiner tous ces aspects de façon à réaliser un compromis acceptable entre toutes ces exigences qui sont parfois contradictoires.

En particulier, il est nécessaire sans trop pénaliser les temps d'exécution des procédures de maintenance, d'exploiter au maximum les possibilités offertes par des moyens mis en oeuvre au sol, soit en interne au système avionique embarqué, soit en externe.

Par exemple, pour les équipements comportant un opérateur programmé et une mémoire vive, il peut apparaître préférable de charger les programmes de test et de les faire exécuter en mémoire vive lors des opérations de maintenance plutôt que de les faire résider en permanence en mémoire programme : on évite ainsi de pénaliser la taille de la mémoire, on diminue les problèmes de sécurité liés à la maintenance, et on ne demande à l'équipement que d'être capable de charger et d'exécuter ces programmes dans sa zone de mémoire vive.

4. PRINCIPE GENERAL D'ORGANISATION DU DEPANNAGE

Le dépannage des équipements est organisé suivant trois niveaux d'intervention :

- Maintenance en Ligne avec localisation des Unités Remplaçables en Ligne (URL)
- Maintenance Hors Ligne des équipements avioniques avec localisation des Unités Remplaçables en Atelier (URA)
- Maintenance Industrielle des équipements avioniques avec localisation des composants.

Le découpage matériel de nos systèmes étant organisé en Boîtes noires, nous n'avons pas pour objectif principal de localiser directement sur avion les Unités Remplaçables en Atelier en panne. Cependant, la localisation d'URA en panne est possible dans un certain nombre de cas comme nous le verrons plus loin.

D'autre part, tous les résultats des diagnostics effectués au niveau de l'avion sont mémorisés sur un média informatique et transmis directement à l'atelier hors ligne pour faciliter les opérations de diagnostic d'URA en panne lorsque cellés-ci ne sont pas localisées au niveau de l'avion. De la même manière, les testeurs de maintenance hors ligne constituent une base de données pour informer la maintenance industrielle des résultats de test qui ont conduit au diagnostic de panne des URA déposées.

5. MAINTENANCE EN LIGNE

La maintenance en ligne est l'ensemble des opérations de maintenance exécutées par les unités opérationnelles sur leurs aéronefs en utilisation. On distingue deux phases dans la maintenance en ligne des systèmes :

- Une maintenance en temps réel, pendant le fonctionnement opérationnel du système pour laquelle l'accent sera mis sur les autotests des équipements avec l'impératif de ne pas perturber le fonctionnement normal.
- Une maintenance complémentaire au sol, pendant laquelle l'ensemble des systèmes adoptera un fonctionnement adapté, qui permettra le test des liaisons inter-équipements, la réalisation des tests trop perturbants pour être effectués en fonctionnement normal, etc...

L'utilisation de la maintenance intégrée offre les avantages suivants :

- Mise en oeuvre quasi-instantanée

- Grande indépendance vis-à-vis des évolutions des logiciels des équipements
- Exploitation sur avion des résultats avec possibilité d'avoir une bonne connaissance des pannes qui se sont produites en vol et qui ne sont pas toujours décelables au sol.
- Recherches de pannes ou validations sans simulations des conditions de vol, d'où une plus grande rapidité d'exécution des procédures
- Réduction très importante du nombre de moyens de tests extérieurs à l'avion
- Maintenance quel que soit le mode de fonctionnement opérationnel
- Grande souplesse dans les procédures de dépannage
- Câblage avion spécifique à la maintenance très faible
- Faible emprise des dispositifs de maintenance intégrée sur le matériel et sur le logiciel des équipements.

5.1. MAINTENANCE TEMPS REEL

Elle s'appuie essentiellement sur les surveillances et autotests permanents non perturbants, réalisés par les équipements sur eux-mêmes et sur leurs périphériques de la façon la plus complète possible.

Ces tests ont pour but de vérifier le bon fonctionnement de l'équipement tant au plan du matériel qu'au plan du logiciel. Ils permettent soit de détecter une panne, dont l'origine peut être sûre ou ambiguë, soit un retour au bon fonctionnement. Les résultats de ces contrôles, après les traitements préliminaires réalisés au niveau de l'équipement, sont émis, via un bus numérique, à destination d'un organe de calcul central qui effectue les tests complémentaires sur les parties des équipements ne pouvant pas être couvertes par les autotests internes (dialogue bus par exemple). Puis pour chaque équipement, après filtrage et synthèse éventuels, cet organe assure la mémorisation et la datation des événements relatifs aux équipements. C'est cet organe qui est chargé d'assurer la restitution au mécanicien ou au pilote au retour du vol. Dans ce mode, quand la localisation de panne peut être réalisée, elle est automatique.

5.1.1. FONCTIONNEMENT DES COMPTES RENDUS DE MAINTENANCE (CRM)

C'est sous ce vocable qu'est désignée la maintenance temps réel et l'exploitation qui

en est faite dans l'avion. Les CRM ont une double procédure d'utilisation :

- Les Comptes Rendus Enregistrés (CRE)
- Les Comptes Rendus Instantanés (CRI)

5.1.1.1 LES COMPTES RENDUS ENREGISTRÉS

Ils correspondent aux enregistrements, réalisés dans l'organe de calcul central, des événements détectés sur les équipements, et effectués pendant le fonctionnement opérationnel. Il y a enregistrement pour chacun des cas suivants :

- Passage d'un état de bon fonctionnement à un état de panne
- Passage d'un état de panne à un autre état de panne
- Passage d'un état de panne à un état de bon fonctionnement.

Ces enregistrements permettent, au retour, de restituer les événements. Cette restitution est réalisée sous deux formes :

- Les pannes en fin d'utilisation
- L'historique des événements.

Pannes en fin d'utilisation

Cette procédure permet de fournir à l'opérateur la liste des équipements encore en panne au moment de l'arrêt de l'enregistrement. Ces équipements sont classés en deux listes :

- Ceux qui sont en panne avec localisation sûre ; ces équipements peuvent être remplacés immédiatement, sans recherche complémentaire.
- Ceux qui sont en panne avec localisation ambiguë ; ces équipements demandent une recherche complémentaire afin de déterminer qu'elle est l'URL en défaut à remplacer. C'est pour ce type de panne, entre autres, que l'on fait appel à la maintenance complémentaire au sol.

Les enregistrements sont effectués dans une zone mémoire non volatile à bord de l'avion afin que ces informations puissent être relues et exploitées même après mise hors tension de l'avion. A chaque nouveau vol, les enregistrements du vol précédent sont effacés.

Historique des événements :

Il permet de restituer la chronologie des événements survenus et enregistrés pendant l'utilisation opérationnelle. Son exploitation peut :

- permettre de comprendre certains phénomènes par la connaissance de l'enchaînement de ceux-ci,

- permettre de connaître les événements fugitifs : pannes qui sont apparues puis disparues et ne sont donc plus présentes au retour.

5.1.1.2 LES COMPTES-RENDUS INSTANTANES (CRI)

Cette procédure utilise exactement les mêmes contrôles que ceux effectués pour les comptes rendus enregistrés. La différence réside dans le fait qu'il n'y a pas de mémorisation, mais présentation en temps réel à l'opérateur des résultats des tests. Celui-ci peut donc voir évoluer le fonctionnement des équipements du point de vue des autotests permanents.

C'est cette procédure qui est utilisée pour valider une réparation quand un équipement a été remplacé suite à l'exploitation des pannes en fin d'utilisation.

5.1.2. ORGANISATION DES ECHANGES D'INFORMATIONS

Chaque équipement envoie sur le bus numérique multiplexé un message dit de servitude qui comprend :

- Un mot de panne constitué d'un nombre entier d'octets, le format étant unique pour un sous-système donné. Ce mot de panne fournit :
 - Une indication de bon fonctionnement global de l'équipement ou une indication de bon fonctionnement par URL dans le cas d'équipements multi-URL.
 - Une indication de bon fonctionnement pour chacune des entités fonctionnelles de chaque URL constituant l'équipement.
- Un mot de mode, qui fournit l'indication du mode dans lequel l'équipement fonctionne dans le cas des équipements multimodes.
- Un mot d'Etat Interne, qui fournit pour chaque URL constituant l'équipement :
 - Une information de panne par URA constituant l'URL
 - Une information de résultat bon ou mauvais de chaque auto-test élémentaire exécuté dans l'équipement.

Il y a donc autant de Mots d'Etat Internes qu'il y a d'URL dans un équipement. Ce dispositif permet de faire, en ligne, un diagnostic d'URA en panne, par la maintenance temps réel.

- Un mot d'Alarme, qui permet de faire apparaître au pilote une alarme et éven-

tuellement une consigne, en cas de panne détectée.

Cet ensemble d'informations est envoyé séquentiellement (toutes les 160 ms environ) vers l'organe de calcul centralisé pour être traité.

Chaque information transmise par un équipement par l'intermédiaire du message de servitude doit être consolidée et filtrée au niveau de chaque équipement avant d'être transmise.

Au niveau global, ces informations sont traitées afin de constituer les Comptes-Rendus de Maintenance (CRM).

Chaque CRM comprend :

- Un mot de panne globalisé à partir des informations du mot de panne de l'équipement.
- Un mot de mode
- Un mot d'Etat Interne
- Un mot d'Identification
- Un mot de datation. (résolution inférieure à 200 ms).

5.1.3. ORGANISATION DU TRAITEMENT GLOBAL (VOIR FIG.1)

Toutes les informations contenues dans les messages de servitude sont tout d'abord traitées dans un module de traitement appelé "Filtrage inter-système". Cette fonction a pour but de rechercher les événements à l'origine d'un mauvais fonctionnement afin de détecter la panne et la séparer de ses conséquences.

La sortie de cette fonction est reliée à trois autres qui sont :

- Les modules liés à l'élaboration des alarmes pilotes et des consignes associées
- Les modules liés à la gestion des modes dégradés
- les modules liés à la génération des diagnostics de pannes pour remise en état par le mécanicien.

A l'heure actuelle, tous les traitements sont algorithmiques, mais nous avons entrepris, dans le cadre d'une étude "co-pilote électronique", la résolution d'un certain nombre de fonctions par Intelligence Artificielle. Les fonctions étudiées sous cet aspect sont :

- Le filtrage inter-systèmes
- Les modules liés à la génération des alarmes et des consignes pilote
- La gestion des modes dégradés.

Les principaux problèmes rencontrés concernent principalement, le temps de réponse qui doit être très proche du temps réel car lié à la sécurité du vol. A l'heure actuelle, le temps de réponse en algorithmique est de l'ordre de 400 à 500 ms. Le second problème correspond à la taille des logiciels à embarquer ainsi qu'à la puissance de calcul requise.

Lorsque ce problème sera résolu, il sera alors possible de résoudre de la même manière les logiciels de traitement des diagnostics de pannes qui ont aujourd'hui des temps de réponse homogènes avec les alarmes pilote, c'est-à-dire de l'ordre de 400 à 500 ms.

L'approche globale développée montre que nous avons le maximum de garanties d'homogénéité entre toutes ces fonctions vitales, car élaborées à partir de données identiques et formalisées de façon homogène entre tous les équipements et systèmes de l'avion.

Du point de vue des performances, la maintenance intégrée aujourd'hui permet d'avoir un diagnostic non ambigu dans 70% des cas de panne au niveau des URL avec un taux de fausse dépose inférieur à 10%. L'apport de l'intelligence artificielle devrait nous faire atteindre un taux de bon diagnostic de l'ordre de 80 à 85% avec un taux de fausse dépose inférieur à 5%.

Le diagnostic d'URA en panne par la maintenance intégrée temps réel se situe aux environs de 30 à 40% des cas ; ce pourcentage devrait être sensiblement amélioré par l'utilisation de l'intelligence artificielle.

Compte tenu des contraintes temps réel extrêmement sévères pour ces fonctions exécutées pendant le fonctionnement opérationnel de l'avion, nous ne pensons pas faire voler en expérimentation un système basé sur l'intelligence artificielle avant 1998.

5.2. MAINTENANCE COMPLEMENTAIRE AU SOL.

Elle a pour objet de permettre la localisation de pannes lorsque la maintenance temps réel n'a pas été en mesure de le faire.

En effet, pendant le fonctionnement opérationnel d'un équipement ou système, il n'est pas envisageable de réaliser les tests complémentaires perturbants qui permettent de localiser les pannes de façon exhaustive.

Ces contrôles complémentaires, effectués au sol, peuvent être des tests déclenchés, des positionnements d'informations à des valeurs fixées, ou des lectures de valeurs d'informations reçues ou émises par les équipements.

Afin de faciliter la tâche de l'opérateur et de diminuer les durées de contrôles, les équipements ou les systèmes dont la technologie le permet, abandonnent le fonctionnement opérationnel au profit d'un fonctionnement adapté à la localisation de la panne et la vérification des éléments constituant l'équipement ou le système. De plus, la maintenance complémentaire au sol, comme son nom l'indique, est un complément à la maintenance temps réel et ne remet pas en cause les résultats obtenus par celle-ci.

Pour une maintenance rapide et efficace, les contrôles relatifs à la vérification du bon état d'un équipement se font par un contrôle de l'intégrité matérielle des composants, par vérification de l'intégrité du logiciel et non par simulation de fonctionnement. Des tests correctement réalisés sur le logiciel d'un équipement valident, a priori, complètement ce logiciel.

Dans ce mode, la maintenance fournit à l'opérateur un ensemble d'outils que celui-ci utilise à sa convenance. Dans ce cas, la localisation de panne est le résultat de la réflexion de l'opérateur utilisant ces outils, éventuellement assisté par des logiciels automatiques ou semi-automatiques.

A ce stade, il est évident qu'un système expert peut être d'une grande utilité.

5.2.1. FONCTIONNEMENT DE LA MAINTENANCE COMPLEMENTAIRE

La maintenance complémentaire au sol a pour but de fournir à l'opérateur les moyens d'investigation nécessaires à l'identification de l'élément en défaut dans le cas des pannes de localisation ambiguë. Les pannes de localisation ambiguë sont celles qui mettent en cause plusieurs URL et pour lesquelles il n'a été possible, en temps réel, que de détecter une anomalie au niveau d'une fonction, mais sans pouvoir effectuer la localisation.

De prime abord, les dispositifs mis en oeuvre par la maintenance complémentaire au sol vont fournir des points d'accès intermédiaires à l'intérieur des fonctions de manière à permettre la localisation. Ces points intermédiaires et les dispositifs correspondants vont permettre :

- de lire des informations en ces points
- de positionner ces informations à différentes valeurs
- de déclencher des tests perturbants
- etc

La mise en oeuvre de ces dispositifs peut être effectuée selon les cas par les moyens suivants :

- Manuellement par l'opérateur (ce qui requiert de sa part une bonne connaissance du fonctionnement de la maintenance intégrée ainsi que des systèmes à contrôler)
- Manuellement par l'opérateur, mais assisté par des "outils logiciels"
- Automatiquement par des programmes de test et de contrôle

Pour certaines utilisations ou fonctions, il est possible d'automatiser l'utilisation de la maintenance complémentaire au sol, au moyen de logiciels spécifiques chargés temporairement dans l'organe de calcul central gérant la maintenance.

Ces logiciels sont nécessaires dans les cas suivants :

- lorsque la sécurité est en jeu et que des contrôles stricts sont nécessaires
- lorsque les paramètres à manipuler sont soit trop rapides, soit trop complexes et donc incompatibles avec une exploitation manuelle.

On est aussi amené à utiliser la maintenance complémentaire au sol lorsque le pilote signale une anomalie et qu'aucune trace n'est présente dans les Comptes Rendus de Maintenance.

5.2.2. CONTROLE DES INTERFACES ET DE LA CONTINUITÉ DES LIAISONS

Les équipements et les systèmes sont de plus en plus numérisés et informatisés. Ce type de technologie est particulièrement bien adapté à la maintenance intégrée. En effet, avec ce type d'équipement, il est possible de séparer le contrôle en deux parties indépendantes.

- d'une part le contrôle de l'intégrité du logiciel d'application
- d'autre part le contrôle de l'intégrité de chacun des composants de l'équipement.

Si ces opérations sont correctement menées, elles suffisent à valider le bon fonctionnement de l'équipement, et pour une grande part en temps réel, pendant le fonctionnement opérationnel.

Le point faible sur le plan de la localisation de panne étant les interfaces, le câblage et les capteurs, la maintenance complémentaire va donc être orientée essentiellement vers la vérification des éléments constituant ces circuits et le contrôle des liaisons.

- Cas des informations reçues par un équipement :

Un équipement numérique qui reçoit des paramètres analogiques doit d'abord les convertir sous forme numérique avant de pouvoir les utiliser dans ses algorithmes : il en résulte donc que son circuit de réception se comporte comme un appareil de mesure.

- Cas des informations émises par un équipement :

Un équipement numérique qui fournit des paramètres analogiques doit d'abord calculer la valeur en numérique au cours de son traitement, puis convertir ce paramètre sous forme analogique ; il se comporte donc comme un appareil de génération de signal.

Pour vérifier une liaison analogique, il suffit de faire en sorte que les valeurs émises par un équipement source puissent être fixées par l'opérateur et que celles reçues par l'équipement destinataire soient présentées à l'opérateur, afin que celui-ci puisse juger de la qualité de la liaison. Avec un rebouclage de la sortie de l'équipement émetteur sur un de ses codeurs d'entrée, il est possible de vérifier que la valeur émise est bien conforme à la valeur de positionnement, et donc de réaliser la localisation de la panne.

5.2.3. ORGANISATION GÉNÉRALE DE LA MAINTENANCE COMPLÉMENTAIRE

L'ensemble des traitements liés à la maintenance complémentaire au sol est assuré par le calculateur qui centralise la gestion de la maintenance. Ce calculateur, pour cette fonction, interface l'opérateur à l'ensemble des systèmes de l'avion. Le dialogue avec l'opérateur s'effectue soit par l'intermédiaire des commandes et visualisation du cockpit qui sont alors spécialement configurées pour ce fonctionnement, soit par l'intermédiaire d'un poste spécifique accessible depuis le sol, soit éventuellement par un ensemble de dialogue déporté et relié au bus avion par une prise coque. (voir FIG.2).

Le fonctionnement maintenance complémentaire au sol mettant en jeu dans la plupart des cas un nombre relativement important d'équipements, n'est pas utilisé pour la localisation d'URA en panne afin de ne pas pénaliser la fiabilité globale de l'avion.

Ce mode de fonctionnement est aujourd'hui entièrement algorithmique. Cependant, il est particulièrement bien adapté à l'intelligence artificielle du fait qu'il n'y a pratiquement pas de contrainte temps réel. Les

logiciels du système expert ainsi que les bases de données peuvent être chargés dans le calculateur hôte à la place du logiciel opérationnel puisque celui-ci n'est pas utilisé en maintenance complémentaire au sol. Par ailleurs, le dialogue avec l'opérateur est extrêmement réduit du fait que les actions de test et leurs résultats sont entièrement gérés informatiquement par l'intermédiaire de la fonction gestion de la maintenance complémentaire. Les dichotomies peuvent être effectuées automatiquement et les actions de test suivantes automatiquement choisies et déclenchées par le système expert en fonction des règles définies. Le rôle de l'opérateur est par le fait même extrêmement réduit et pratiquement limité à la description des symptômes vus par le pilote.
(voir FIG. 3).

A l'heure actuelle, on considère que la maintenance complémentaire au sol permet de compléter les performances de la maintenance temps réel pour obtenir un taux global de localisation d'URL en panne de 95%. La rapidité du test et son efficacité sont fonction de la qualité de l'opérateur en utilisation manuelle. On considère que dans ce mode, environ 95% des défauts sont localisés en moins de deux heures par un mécanicien de bon niveau. Il faut en outre remarquer que les validations de chaînes fonctionnelles d'armement sont toutes effectuées automatiquement à l'aide de logiciels gérant les procédures. Les performances actuelles montrent que la validation d'une configuration d'emport quelle qu'elle soit dure largement moins d'une heure.

L'apport de l'intelligence artificielle dans ce cas peut être intéressant pour homogénéiser les résultats. Il serait envisageable de considérer qu'une action de diagnostic quelle qu'elle soit dans ce mode dure moins d'une heure avec une efficacité meilleure que 95% des cas de panne (taux de fausse dépose inférieur à 5%). Par ailleurs, le niveau de formation des mécaniciens pourrait être bien inférieur à ce qui est demandé aujourd'hui.

Le développement de ce type de système expert n'est encore pas entrepris à l'heure actuelle car le bien fondé économique de ce développement n'est pas encore démontré compte tenu des performances attendues de la maintenance temps réel d'une part, et de l'efficacité des tests réalisés de façon algorithmique observée jusqu'à maintenant. Il faudra attendre encore 1 an ou 2 d'essais en vol avant de prendre une décision.

6. MAINTENANCE HORS LIGNE

La maintenance hors ligne est l'ensemble des opérations de maintenance exécutées

par les échelons de soutien technique des utilisateurs.

Le but de la maintenance hors ligne est d'assurer le dépannage et le contrôle des URL. Les URL sont décomposés en URA qui sont des éléments interchangeable au niveau de l'atelier hors ligne. De même que la maintenance en ligne a pour objet le maintien en condition de l'avion, la maintenance hors ligne a pour objet le maintien en condition des URL.

La maintenance hors ligne s'effectue en atelier et nécessite la mise en oeuvre de moyens matériels et logiciels. (Ces moyens sont généralement fournis par un moyen de test à caractère universel). Elle s'applique à l'URL, et porte sur la détection et la localisation de l'URA en panne, sans intervention de l'opérateur au sein de l'URL pour le contrôle de bon fonctionnement ; éventuellement avec accès à un connecteur interne à l'URL pour les contrôles complémentaires de localisation d'avarie.

Les réparations doivent s'effectuer par dépose/pose de l'URA en panne à l'aide d'outillages standard excluant le fer à souder.

Il faut noter que comme pour la maintenance en ligne, la maintenance hors ligne est basée sur les principes de maintenance intégrée, c'est-à-dire vérification de l'équipement par contrôle de l'intégrité des composants plutôt que par simulation de fonctionnement, et ceci à chaque fois que cela est possible, en tenant compte de la technologie des URL et des URA. Le but est bien évidemment de réduire les temps de réparation d'une URL et par voie de conséquence, limiter l'occupation du banc de test pour les opérations effectuées sur une URL donnée. Dans ce même but, on profitera des résultats fournis par la maintenance en ligne pour accélérer les opérations en atelier, notamment en évitant de refaire des autotests déjà réalisés sur avion.

6.1. PRINCIPE

Les procédures de test sont habituellement articulées en :

- Test de bon fonctionnement
- Test de localisation d'avarie
- Le test de bon fonctionnement a pour but de vérifier l'intégrité du matériel et du logiciel de l'équipement. Il est réalisé sur une URL pour vérifier son bon fonctionnement soit après destockage, soit pour détecter la fonction en panne afin d'aiguiller le programme de test vers la localisation d'avarie adéquate. Il peut également être réalisé en atelier après rechargement du logiciel dû à un changement de standard.

- Le test de localisation d'avarie a pour but de localiser précisément l'avarie et donc en déduire qu'elle est l'URA en panne qu'il faut remplacer.

En fait, cette séparation n'est plus celle utilisée aujourd'hui. Chaque équipement est décomposé en Ensemble de Composants Testables par la méthode d'intégrité. Chaque test d'Ensemble de Composants Testables fait l'objet d'un chapitre de test qui vérifie soit le bon fonctionnement (bonne intégrité), soit détecte la panne et la localise au niveau d'une URA.

6.2. MODE OPERATOIRE

Les équipements sont testés au niveau de l'atelier hors ligne dans les cas suivants :

- Lorsque l'URL a été détectée en panne sur avion par la maintenance temps réel et que l'URA fautive est parfaitement localisée (par lecture du mot d'Etat Interne par exemple). Dans ce cas, les opérations sont :
 - Remplacement de l'URA fautive dans l'URL (il n'est pas nécessaire de confirmer la panne)
 - Contrôle de l'URL réparée par une vérification de bon fonctionnement
- Lorsque l'URL a été détectée en panne sur avion par la maintenance temps réel mais sans localisation sûre de l'avarie. Dans ce cas, les opérations sont :
 - Lecture du Mot d'Etat Interne et utilisation des résultats de la maintenance en ligne pour orienter le programme de test vers le chapitre correspondant à la fonction en défaut. (Il n'est pas nécessaire de confirmer la panne par le passage en vérification de bon fonctionnement, car celle-ci a été détectée par les autotests ; il ne subsiste qu'un problème de localisation et on utilise les résultats des tests temps réel pour aller rapidement vers le chapitre de test qui localisera la panne).
 - Après localisation de l'avarie : remplacement de l'URA fautive
 - Contrôle de l'URL réparée par une vérification de bon fonctionnement global.
- Lorsque l'URL a été détectée en panne sur avion au moyen de la maintenance complémentaire au sol, généralement, dans ce cas, la localisation n'est pas assurée et on se retrouve donc ramené au cas précédent, à la différence que le Mot d'Etat Interne ne fournit pas de ren-

seignement précis. Si les informations de la maintenance complémentaire le permettent, on oriente le programme vers le chapitre convenable du programme, sinon on utilise la vérification du bon fonctionnement qui assurera l'orientation vers le chapitre où la panne pourra être localisée en fonction des résultats des contrôles effectués.

6.3. FONCTIONNEMENT DES PROCEDURES DE TEST (FIG.4)

Les procédures de test de maintenance hors ligne sont entièrement gérées par un mécanisme appelé "Mécanisme de Décision", dont les fonctions sont les suivantes :

- Gestion de la "mise en place" de l'URL sous test sur le testeur
- Analyse des données fournies par les données de maintenance intégrée temps réel au niveau avion et en particulier le Mot d'Etat Interne
- Eventuellement analyse du Mot d'Etat Interne instantané de l'URL sous test
- Eventuellement analyse des données fournies par la maintenance complémentaire au sol (soit par l'intermédiaire de l'opérateur, soit à partir d'une base de données en provenance de la maintenance en ligne).
- Décision sur l'exécution du chapitre de test le plus pertinent en fonction des résultats.
- Exécution éventuelle d'autres chapitres en fonction des résultats fournis par les chapitres précédents, en tenant compte des données de fiabilité, de la facilité relative des déposes d'URA, etc.
- Etablissement du diagnostic
- Après échange de l'URA fautive, gestion de l'exécution du test de bon fonctionnement
- Elaboration d'une base de données pertinentes pour la maintenance industrielle

Ce mécanisme d'exécution sera réalisé pour le test des équipements RAFALE par un système expert implanté dans le testeur.

L'utilisation de ce principe, liée à l'application la plus extensive possible du test d'intégrité permet d'envisager un temps moyen d'immobilisation du testeur inférieur à 1 heure pour la remise en état d'une URL.

7. MAINTENANCE INDUSTRIELLE

La maintenance industrielle est l'ensem-

ble des opérations permettant la localisation d'un composant défectueux d'une URA et son échange.

Les moyens nécessaires à la localisation du composant défectueux d'une URA sont en général des bancs de test automatisés.

Tout ce qui a été développé au paragraphe 6 concernant la maintenance hors ligne, peut s'appliquer de la même manière à la maintenance industrielle. Les données amont de ce niveau de maintenance sont fournies par le testeur de maintenance hors ligne et éventuellement les données extraites de l'avion et confectionnées par la maintenance en ligne temps réel.

Aujourd'hui, la décision d'appliquer à la maintenance industrielle un "Mécanisme de Décision" du même type que celui de la maintenance hors ligne, n'a pas encore été prise.

8. DEVELOPPEMENT DES TECHNIQUES D'INTELLIGENCE ARTIFICIELLES APPLIQUEES A LA MAINTENANCE TEMPS REEL

Les actions menées par l'équipe d'Application de l'Intelligence Artificielle de la Division Systèmes d'Armes s'inscrivent dans le cadre d'un projet ambitieux d'assistance au pilote appelé "Copilote Electronique".

Ce projet lancé en 1986 par la DRET et conforté par le STTE, vise à utiliser l'IA pour réaliser un système d'aide à la décision, embarqué dans un avion monoplacé de combat. L'objectif est de fournir au pilote des informations pertinentes pour aider à la décision.

Le contexte temps réel embarqué introduit de nombreuses spécificités tant théoriques que pratiques, qui nécessitent une démarche originale vis-à-vis des solutions classiques de diagnostic hors ligne.

Ceci nous a amené à étudier avec la société DASSAULT ELECTRONIQUE les filières d'embarquabilité de système d'Intelligence Artificielle dans le domaine du filtrage d'alarmes. (étude STTE "Maquette de Système Expert Embarquable sur Avions d'Armes".

Il s'agissait :

- de prouver la faisabilité de l'embarquabilité d'un système expert avionique sur une application représentative
- de proposer une filière (matérielle et logicielle) pour le développement d'un tel système.

Les contraintes de prise en compte du temps d'interruptibilité des traitements, d'asynchronisme des arrivées d'informations et de limitation des volumes mémoire, ont été couvertes de manière à pouvoir dimensionner les puissances de calcul.

Ces premiers éléments de chiffrage obtenus pour un filtrage d'alarmes multisystèmes (moteur, électricité, hydraulique et freinage) au regard des matériels compatibles du RAFALE A, nous ont amenés à envisager avec DASSAULT ELECTRONIQUE, une approche de développement sur un générateur de système expert riche, complété par une traduction vers un langage de production de logiciel embarqué (ADA). L'optimisation temps réel de certaines fonctions peut être obtenue en dédiant le traducteur à un type d'application précis (i.e. le filtrage d'alarmes inter-systèmes).

Nous nous proposons d'étudier cette approche pour l'avion RAFALE et un calculateur type CMF-air.

En parallèle de cette démarche sur la filière de traduction vers ADA. Nous travaillons au développement des fonctions d'assistance en matière de surveillance des systèmes de l'avion, de reconfiguration après pannes, de filtrage d'alarmes et d'exécution de procédures d'urgences.

Avec la société SAGEM, nous préparons des travaux qui permettront de diagnostiquer les pannes simples puis les pannes multiples, de maintenir un vecteur d'état généralisé, et de proposer au pilote des reconfigurations liées au contexte de la mission.

Les traitements d'Intelligence Artificielle permettent une réflexion sur les données brutes circulant sur les bus, grâce à la maintenance intégrée ainsi que des prédictions sur les évolutions des systèmes (dysfonctionnements probables, dérives...). Ainsi une estimation de la capacité de l'avion à remplir sa mission sera rendue possible.

Ces traitements mettront en œuvre des techniques symboliques de diagnostic par règles expertes combinées à des algorithmes de traitement du signal, des techniques de programmation déclarative en langage orienté objet, de gestion d'hypothèses et de programmation sous contrainte pour la planification des réactions.

9. DEVELOPPEMENT DES TECHNIQUES D'INTELLIGENCE ARTIFICIELLE APPLIQUEES AU DIAGNOSTIC DE PANNES HORS TEMPS REEL

L'expérience du Département Intelligence Artificielle de DASSAULT AVIATION dans le domaine du diagnostic de pannes a été obtenue à travers deux projets :

- Une étude effectuée en collaboration avec ALCATEL ALSTHOM sous contrat DRET et concernant une fonction du système d'armes du MIRAGE F1.
- Une étude effectuée en collaboration avec le Centre Technique de Communication et d'Intégration d'Ateliers et concernant le diagnostic de pannes d'une machine-outil à commande numérique.

Ces deux projets, ainsi qu'une activité de recherche et développement complémentaire ont apporté au Département Intelligence Artificielle une expérience correspondant à un effort poursuivi depuis sept ans. Cet effort se traduit aujourd'hui par la disponibilité d'un "Environnement Logiciel pour la Maintenance Corrective" (ELMC) à l'état de maquette.

L'objectif de l'outil ELMC est le suivant :

A un équipement industriel donné, (un avion, un système, un équipement,...) on veut pouvoir associer un logiciel d'aide à la maintenance corrective spécifique. Les capacités de ce logiciel doivent être adaptables selon le contexte, et sa réalisation doit être effectuée à un coût minimum.

Une originalité de l'outil ELMC consiste à générer automatiquement le logiciel d'aide à la maintenance à partir d'un environnement de développement. (Fig. 5). Chaque type d'utilisateur peut ainsi utiliser les formalismes et connaissances qui leur sont nécessaires.

Une seconde originalité tient à l'adaptivité de l'application aux ressources disponibles : la souplesse de l'application peut être modulée en fonction de la puissance de calcul disponible et des temps de réponse nécessaires.

Ces deux avantages sont obtenus en faisant coopérer les techniques classiques de diagnostic de pannes (maintenance intégrée par exemple) et les techniques de l'intelligence artificielle. Différents niveaux de connaissance sont utilisés : en particulier, l'emploi de l'environnement de développement consiste à réaliser une description hiérarchisée de l'équipement selon des points de vue fonctionnel, structurel et comportemental.

Le développement d'un prototype plus ergonomique de l'outil ELMC est en cours dans l'environnement SMECI de la Société

ILOG. L'expérience accumulée permet maintenant de répondre rapidement à tout besoin dans le domaine du diagnostic de pannes ; en particulier, l'application du savoir-faire acquis à la maintenance hors ligne des équipements de l'avion RAFALE est en cours, par le biais de l'étude d'un outil d'aide à la Spécification des Procédures de test.

10. CONCLUSION :

La maintenance intégrée telle qu'elle a été développée sur nos avions d'armes et leurs équipements, permet d'obtenir des performances très intéressantes sur le plan du diagnostic des pannes. La politique globale retenue, consistant à faire profiter les échelons de maintenance aval des résultats obtenus par les échelons amont sous forme informatique (par l'intermédiaire de médias appropriés) permet d'envisager l'utilisation de l'Intelligence Artificielle de manière rationnelle.

La mise en oeuvre conjointe de la maintenance intégrée basée sur le test d'intégrité et de l'intelligence artificielle permet le développement de systèmes expert de diagnostic de panne à coût faible :

- En temps réel sur avion (à terme relativement lointain)
- En maintenance complémentaire au sol (à moyen terme)
- Aux ateliers de maintenance hors ligne et industrielle (en cours de réalisation)

Les avantages attendus de cette configuration sont :

- Augmentation de la finesse de diagnostic automatique en temps réel (localisation d'URA en panne)
- Réduction des temps de diagnostic et de validation de chaînes fonctionnelles en maintenance complémentaire au sol, tout en réduisant le nombre et la qualification des mécaniciens nécessaires;
- réduction du temps moyen de dépannage d'une URL en atelier (de l'ordre de 45 minutes à 1 heure).

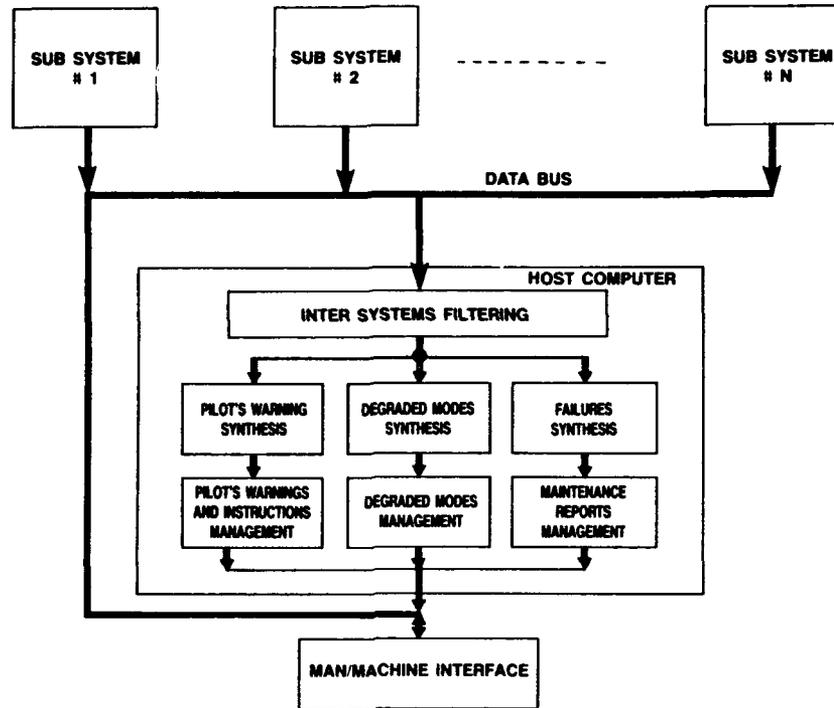


FIG1 MAINTENANCE TEMPS REEL
ORGANISATION DU TRAITEMENT GLOBAL

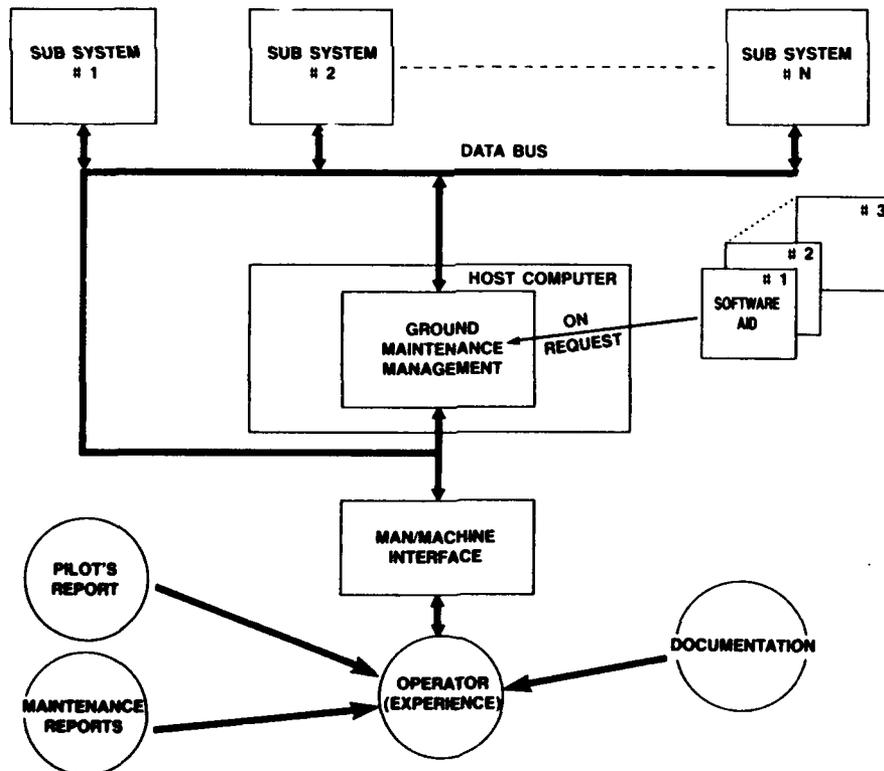


FIG2 MAINTENANCE COMPLEMENTAIRE AU SOL
ORGANISATION ALGORITHMIQUE ACTUELLE

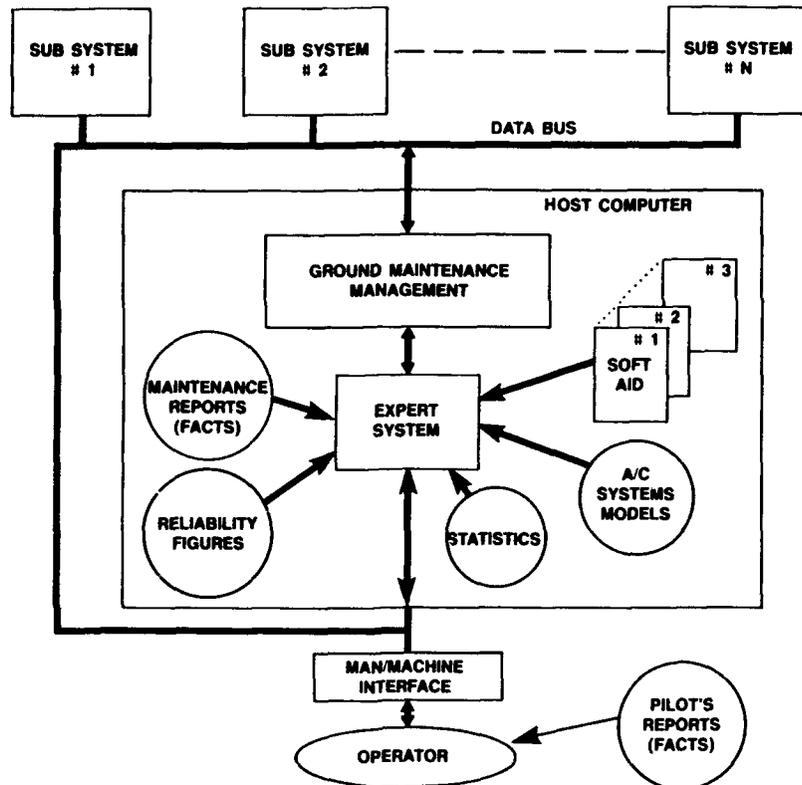


FIG3 MAINTENANCE COMPLEMENTAIRE AU SOL
UTILISATION D'UN SYSTEME EXPERT

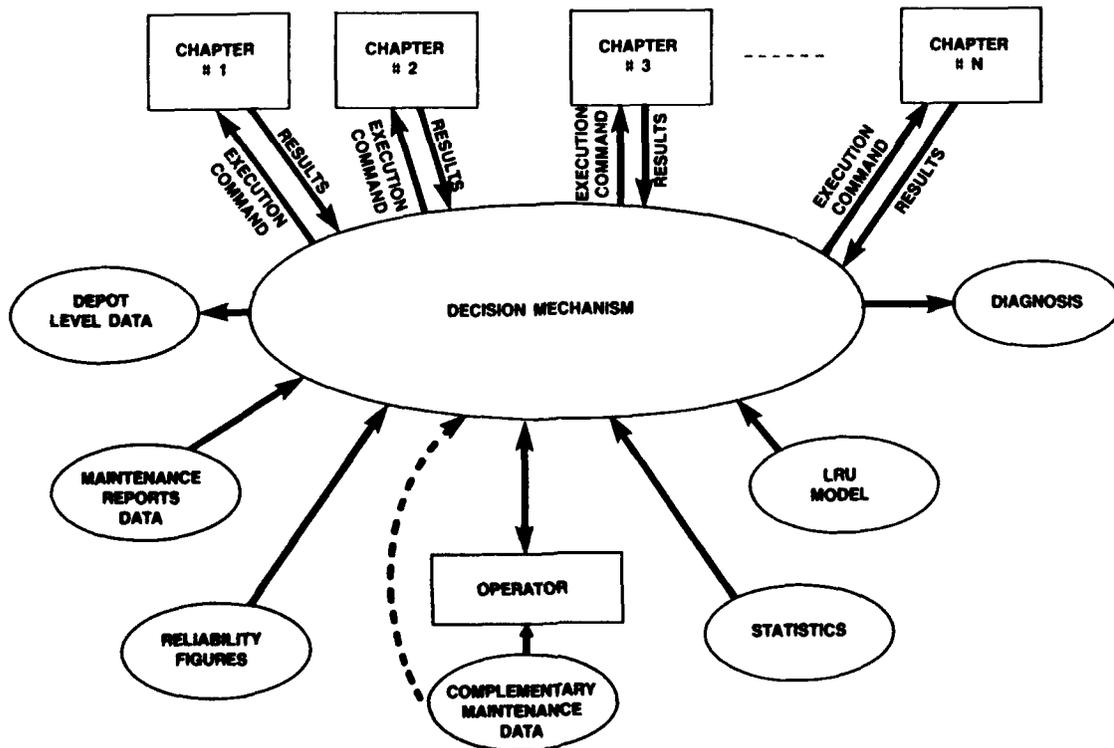


FIG4 MAINTENANCE HORS LIGNE / GESTION DES PROCEDURES DE TEST

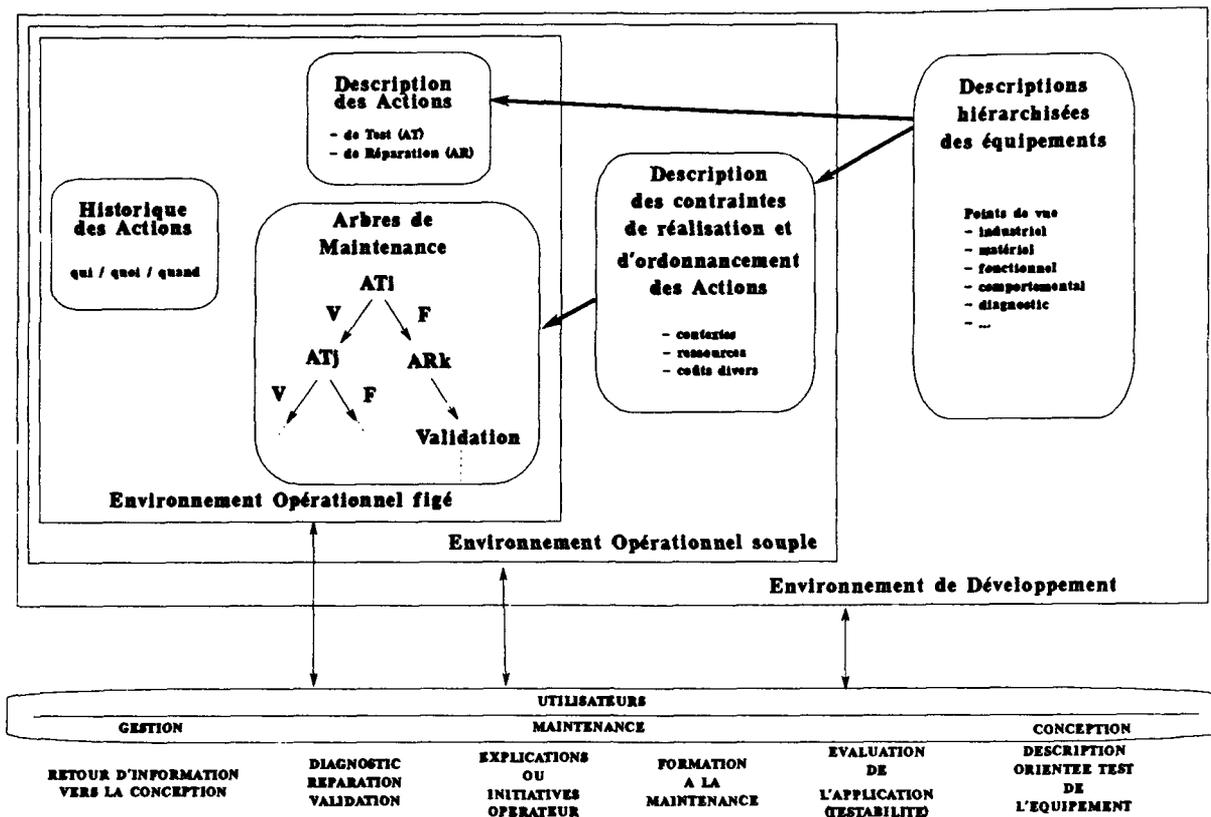


Figure 5 : Architecture de l'Environnement Logiciel pour la Maintenance Corrective (ELMC)

Discussion

1. J. Bart, United States

Did the 1983 study on the MIRAGE develop data on the reduction of retest events using AI techniques?

Author:

Due to the good performance of our system having less than 15% of false removal rate, the focus of this study was to evaluate the software volume and processing power required to embed in avionics equipment, a knowledge based real time diagnostic system and to determine the proper technical approach.

COMPONENT PART NOTICE

THIS PAPER IS A COMPONENT PART OF THE FOLLOWING COMPILATION REPORT:

(TITLE): Conference Proceedings of Machine Intelligence
for Aerospace Electronic Systems Held in Lisbon, Portugal on
13-16 May 1991 (L'Intelligence Artificielle dans les Systemes
Electroniques Aerspatiaux)

(SOURCE): Advisory Group for
Aeronautical Research and Development, Neuilly-sur-Seine,
(France)

TO ORDER THE COMPLETE COMPILATION REPORT USE AD-N242025

THE COMPONENT PART IS PROVIDED HERE TO ALLOW USERS ACCESS TO INDIVIDUALLY AUTHORED SECTIONS OF PROCEEDINGS, ANNALS, SYMPOSIA, ETC. HOWEVER, THE COMPONENT SHOULD BE CONSIDERED WITHIN THE CONTEXT OF THE OVERALL COMPILATION REPORT AND NOT AS A STAND-ALONE TECHNICAL REPORT.

THE FOLLOWING COMPONENT PART NUMBERS COMPRISE THE COMPILATION REPORT:

AD#: _____ TITLE: _____

AD-P006 331
AD-P006 334
AD-P006 336
AD-P006 344
AD-P006 345
AD-P006 346
AD-P006 352



NOT FOR PUBLICATION
EXCEPT BY AUTHORITY OF
DISTRIBUTION STATEMENTS

Accession no.	
NFIS	CRUI
DTIC	TAB
Unannounced Justification	
By _____ Distribution /	
Availability Codes	
Dist	Avail and/or Special
A-1	

AD-P006 345



EXPERT SYSTEM FOR THE TORNADO GROUND-BASED CHECK-OUT SYSTEM

by

J. FEY, J. MARANGOS, M. MERX, W. MANSEL
 MBB-Military Aircraft Division, FE 363
 P.O. Box 801160, 8000 München 80
 Germany

1.0 SUMMARY

The maintenance effort in terms of time and cost of modern avionic systems is driven by the increasing complexity. To achieve the required operational availability of aircraft, the maintenance turn-around-times and logistic support needs to be optimized. Expert Systems seem to be a promising approach to increase the effectiveness of on-board and ground-based test and diagnosis systems. A knowledge based approach, using inference techniques, that can also handle uncertain and incomplete information, was used to support detection and isolation of problems in avionic equipments. A technology demonstrator, supporting the TORNADO check-out system, has been developed and tested. The expert system, called TORRES (TORNADO Radar Readiness Expert System), supports debriefing staff with various levels of experience. The scope of the error detection encompasses the TORNADO Terrain Following and Ground Mapping Radarsystem, down to module level. The main task of TORRES is the identification and isolation of errors that occurred during the previous flight. The expert system is also able to exclude errors, that were generated by other systems capable of changing the state of the radar system and isolate the cause. Other influences like EMC, exceeding avionic systems acceleration limits and weather effects are taken under consideration and reported. If an error is left without an associated case, special test runs are suggested. An explanation facility generates detailed debriefing reports.

2.0 INTRODUCTION

The maintenance and support of technically complex systems develops into an increasingly difficult task. This problem is compounded by the shortage of qualified and well trained specialists. As a result, a lot of support and maintenance work has to be performed by rather quickly trained staff.

The proliferation of more comfortable to use expert systems appears as a partial remedy to the problem. An expert system has the accumulated knowledge and experience readily available and with an integrated explanation facility will support the understanding and the comprehension of the produced results. This type of system has been sufficiently successful and appears to gain support by

its users.

The design of a KBS generally depends on the assumption of cooperative information sources, be it electronic or human. However, field tests have shown, that there are instances of antagonistic knowledge sources. As an example may serve TORRES, the Tornado Radar Readiness Expert System. TORRES is a diagnose and verification system to support fault detection for the TORNADO nose radar system. It is an interesting example, because the complexity of the system is very high and it is highly interconnected with various other subsystems.

3.0 ONBOARD CHECK-OUT SYSTEM

The TORNADO fighter aircraft is equipped with an onboard check-out system. During startup of the avionics suite of the aircraft and during flight, the TORNADO maincomputer software conducts a series of tests with the Built In Test Equipment (BITE) of the different subsystems. The BITE of the individual subsystems isolates possible faults and after some processing with combinatorial logic displays its result as a flashing diode on the Central Maintenance Panel (CMP) to indicate a faulty Line Replacable Unit (LRU). A warning is furthermore displayed on a Central Warning System (CWS) in the cockpit to alert the crew of a possible malfunction. Figure 1 'Signal Representation' shows part of the signal flow of the onboard check-out system. The maincomputer software has limited capabilities to clearly identify the detected and recorded problems by the BITE. It furthermore has no means to track down problems not detected by the BITE, but merely examines the physical connection between the maincomputer and the avionic system.

Unfortunately, mishandling of the avionics equipment can cause the generation of a transient fault, which appears as a latched fault in the CMP. To compound the problem, operators of the aircraft are sometimes reluctant to admit or unaware of those mishandling errors. The conflict arising is, does an LRU need replacement because of the CMP fault indication, even if subsequent checks indicate no fault, or not.

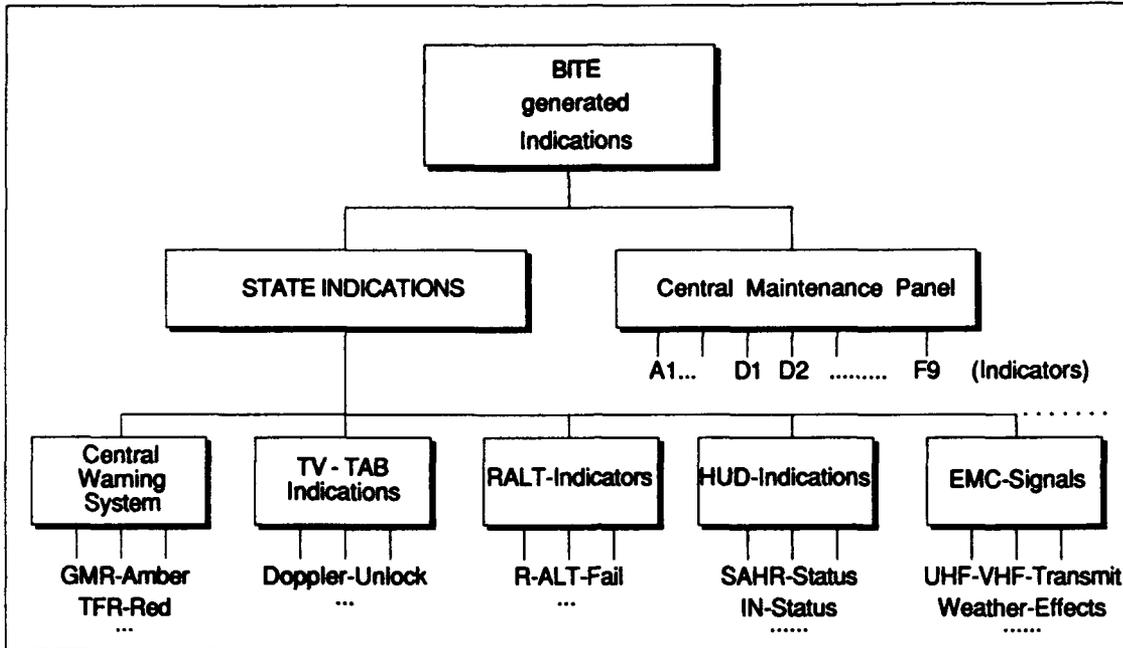


Figure 1: Signal Representation

Depending on the technical experience of the maintenance personnel, a frequently applied method of fault elimination was the complete exchange of the suspected Line Replaceable Unit (LRU). This tedious fault diagnosis negatively influences the following issues.

- turn-around times
- availability of the weapons system
- logistics and material flow

Because of the demanding functional requirements, modern avionic systems are composed of several complex and highly integrated systems and subsystems which have to be easily maintainable. These requirements with respect to maintenance demand the utilization of modern information processing methods. The software has to utilize all the available knowledge sources (analytical, empirical knowledge) and has to exhibit a flexibility that extends past the boundaries of purely sequential program execution. These requirements favour the utilisation of rule-based systems for ground-based diagnose systems for maintenance.

4.0 EXPECTED ADVANTAGES

Due to the nature of the task of fault detection and verification, the solution finding process heavily depends on the recorded data and has therefore to be flexible and opportunistic. The ability to focus on certain problem areas during the development of the KBS was utilized to focus on difficult to isolate faults. The explanation facility of the

KBS provides furthermore clues for the human user to better understand results produced by the KBS, thereby increasing efficiency. Some of the advantages are:

- improved fault diagnosis
The accumulation of several areas of knowledge and experience provides a knowledge base much more powerful than the sum of its individual parts. This enables the maintenance personnel to better detect and isolate faults and furthermore increase the individuals knowledge through the explanation facility.
- accelerated maintenance
Maintenance times are reduced since fault detection is more accurate, thereby reducing time- and material intensive disassembly and assembly of the subsystems.
- shorter turn-around times
Due to the accelerated maintenance, the turn-around time of the individual aircraft is also reduced, increasing the availability of the aircraft.
- reduction of cost of ownership
Better fault isolation and faster maintenance decrease the cost of ownership.
- know-how transfer
The explanation facility of TORRES will supplement and enhance the know-how and experience of the maintenance personnel.

• conservation of knowledge

The knowledgebase of TORRES offers continuous access to knowledge of several areas. Newly acquired experience or knowledge can be added to the domain knowledge to keep the KBS always on the highest level of experience.

5.0 SYSTEM IMPLEMENTATION

The implementation of TORRES had been divided into two phases. During the first phase three demonstrators were developed running on Texas Instruments and Symbolics LISP machines. The developing environment used was the expert systems shell ART of Inference Corporation [1]. These demonstrators were used as experimental platform to study appropriate techniques and methods for the acquisition and representation of knowledge and to develop a useful inference engine to be used in the KBS. The domain knowledge of the system was drawn from several areas. It includes information from system design, from the development and integration and from experience gathered by system use and maintenance.

TORRES is designed to support debriefing staff at various levels of experience. The scope of error detection encompasses the TORNADO Terrain Following Radar (TFR) and the Ground Mapping Radar (GMR) from the system level down to the module level. It was realized however, that the diagnosis process could not be restricted to the two radar systems only. Faults and problems occurring in related systems could easily propagate erroneous signals into the two radar systems and create

malfunctions there. This required the inclusion of related system faults into the knowledge base. Additional influences like EMC, weather effects and excessive g-forces had to be taken into account also. Figure 2 'TORRES Rule Structure' shows the relation of the different rule classes to each other.

During the second phase, TORRES was implemented using the expert system shell KEE (IntelliCorp [2]), that offered the advantage of porting the resulting software to conventional hardware without restricting its overall functionality. For this step a Compaq 386/20 running under UNIX was used.

6.0 SYSTEM PERFORMANCE

TORRES is invoked during the post-flight inspection of the radar system. During the debriefing session, malfunctions of the radar system are analyzed with the objective to decide upon the appropriate maintenance action to be taken. If the fault cannot be traced to a subsystem within the radar system, TORRES will identify the most probable system to be the generator for the propagated fault. This would be a well suited interface to an overall avionic debriefing system consisting of several individual KBS for the various subsystems.

6.1 Solution Process

At debriefing the recorded results of the BITE along with other gathered information provided by the crew is loaded into TORRES either as a textfile, interactively or manually

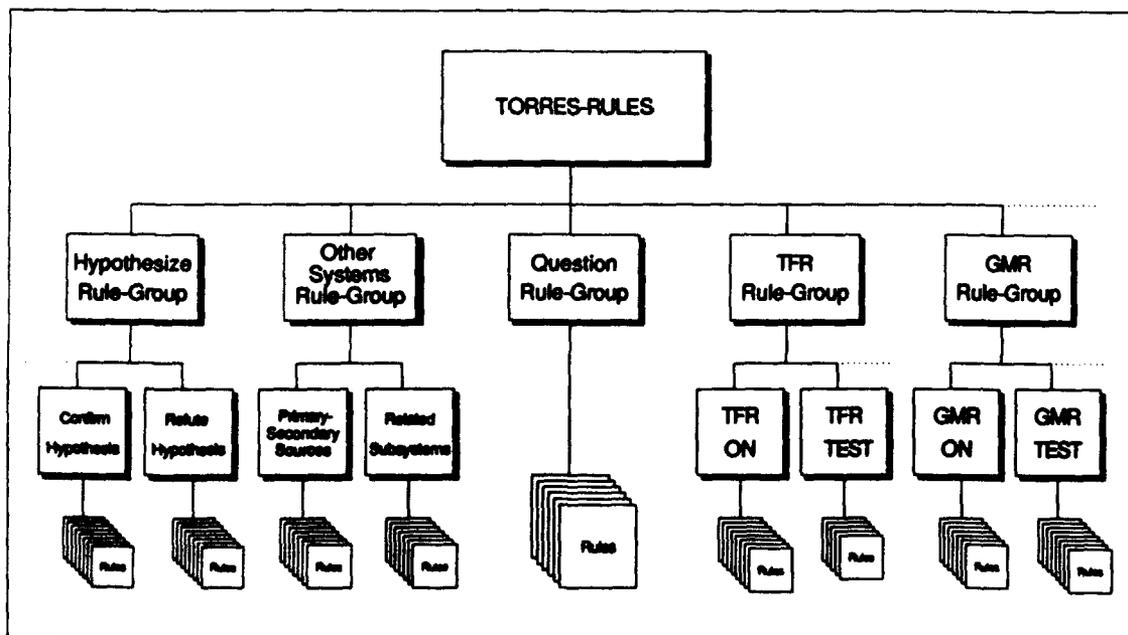


Figure 2: TORRES Rule-Structure

91 1113 022

91-15527
 ■■■■■■■■■■

by an editor. A preprocessor in the KBS scans the file for keywords to help focus on the relevant topics. Additional information, be it related knowledge or experience, is associated with the keywords to establish a base for further processing. One or more hypothesis are selected with respect to the accumulated facts and information. The hypothesis are assigned one of four certainty classes, that range from 'suspected fault' to 'definite fault'. The hypothesis with the highest certainty class is selected and a process is started to evaluate the probability of the fault. TORRES provides a graphical user interface and begins a questioning session to obtain additional information from the pilot or the crew. The questions can be answered with either 'YES', 'NO' or 'UNKNOWN'. Yes and no answers are directly used in the solution process while an unknown is once counted as a yes and as a no. One answer will contribute in the end to a larger extent to the result than the other, influencing the certainty class of the hypothesis. The result is a hypothesis of a fault with an associated certainty class. According to this procedure the rest of the hypotheses are processed. The output containing information about the diagnosis process and explanations is displayed on the screen and printed as a text file. It can consist of series of actions to be performed to eliminate the fault, e.g. replace LRU4 or control the wiring between LRU1 and LRU4.

6.2 Software Environment

The Expert System Shell KEE provides the environment in which TORRES will run. KEE however, needs as supporting environment a platform that hosts Common LISP (Symbolics LISP Machines, TI Explorer, workstations or PCs). In our example a PC (Compaq 386/20) was used running under UNIX (386/ix [3]) with Lucid Common LISP. The Symbolics LISP machine and the TI Explorer provide a graphical interface right on the KEE level, whereas on the PC, additional interfacing to MS-Windows is necessary, which requires the emulation of MS-DOS on an intermediate level.

This additional functionality is achieved using VP/ix of Interactive Systems, an MS-DOS emulation program. To guarantee the above mentioned functionality, two processes are active under UNIX. The processes shown in Figure 3 'Software Environment' are:

- Lucid LISP with KEE and the application TORRES
- VP/ix MS-DOS emulator with MS-Windows as user interface

Interprocess communication between the KEE application TORRES and MS-Windows is provided by UNIX.

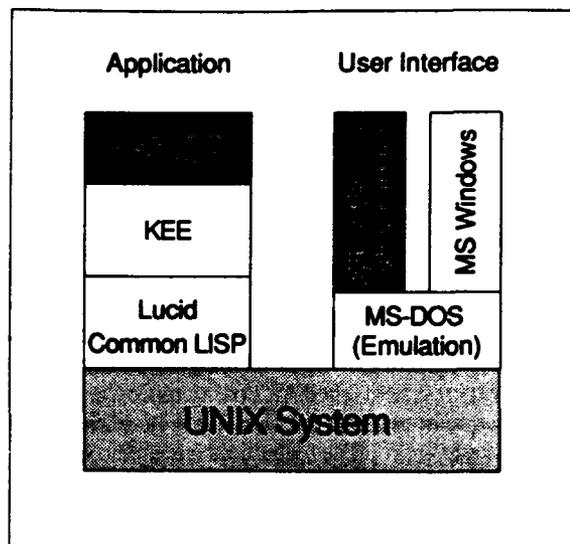


Figure 3: Software Environment

6.3 Execution Speed

Tests with respect to the execution speed have been conducted on two machines. One was a Symbolics 3650 running under GENERA [4] and the other was a PC Compaq 386/20 running under UNIX. The test data sets for both machines were identical and consisted of aircraft information provided by a flight crew after a hypothetical mission.

SYSTEM	Compaq 386/20	Symbolics 3650
Operating System	UNIX-VP/ix	Genera
Software Environment	Lucid-Lisp	Common Lisp
Set-Up [sec]	120	320
Diagnosis [sec]	880	250

Table 1: Performance Results

TORRES was started on both machines with these files. Since both operating systems GENERA and UNIX support multi-tasking, the results obtained could strongly depend on the actual workload of the system at the time. To increase the transparency of the results, two types of tests were conducted, one for the time required to load the application and another for the time required to obtain the

diagnosis results. The outcome of the tests are shown in Table 1 'Performance Results' and Figure 4 'Performance Comparison'.

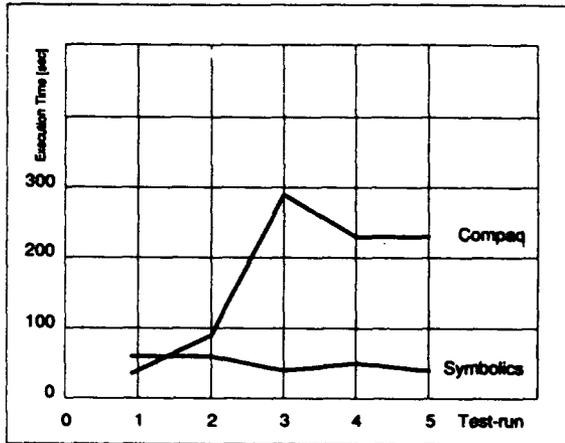


Figure 4: Performance Comparison

The set-up time of TORRES on the PC was shorter by the factor of 2.7 : 1 than on the Symbolics. Responsible for this difference is the multitasking feature of GENERA and the overhead to invoke TORRES within KEE. The speed performance is clearly different for the diagnosis task. Here the Symbolics is leading by the factor of 3.5 : 1 over the PC. Responsible for the fast and very steady performance of the LISP machine is the support provided to LISP by the special hardware and the sophisticated concept of garbage collection (dynamic and ephemeral garbage collector) provided by GENERA. The weaker performance

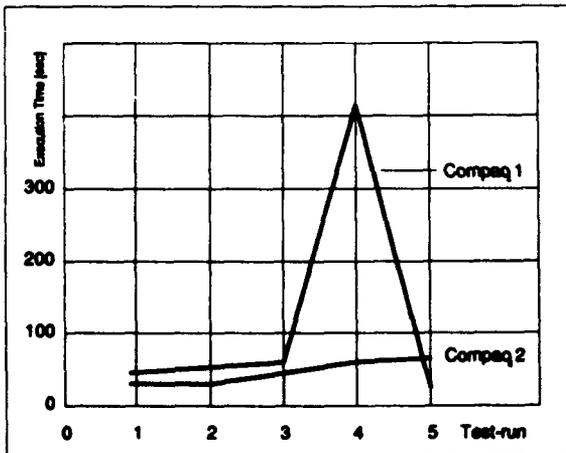


Figure 5: UNIX Tuning

on the PC under UNIX can be attributed to the following reasons. The task of garbage collection was not optimized and therefore less efficient than that of GENERA. Several UNIX system parameters, like kernel streams, message

passing, paging and shared memory access had to be tuned to improve the operating system efficiency. The graphs in Figure 5 'UNIX Tuning' show the improvement achieved by the tuning measures. They are significant during the first test runs, deteriorate however during subsequent runs. They are assumed to asymptotically approach a steady state, well above the Symbolics execution time.

7.0 CONCLUSIONS

With the application TORRES a rule-based diagnosis system has been realized to provide on-line support for the TORNADO maintenance crew. The system runs on LISP machines (Symbolics, TI Explorer) and on conventional PC based hardware. The graphical user interface supports comfortable usage. Through the improved fault diagnosis, accelerated maintenance is achieved, reducing turn-around times thereby increasing system availability.

REFERENCES

- [1] ART Automated Reasoning Tool, V. 3.2 1987, Inference Corporation.
- [2] KEE, Knowledge Engineering Environment, V. 3.1 Dec. 1987, IntelliCorp.
- [3] UNIX 386/ix, V. 1.06 July 1988 and VP/ix, V. 1.1.0 Aug. 1988, Interactive Systems Corporation.
- [4] Operating System Genera, V. 7.2 Feb. 1988, Symbolics Corporation.

Discussion

1. E. Labatt, United States

Could X-windows and other new software be used instead of MS-windows, and a DOS emulator to speed up the processing time on the Compaq computer?

Author:

I agree with you, that the system performance is influenced by the overhead of using the MS DOS emulator as interface to MS-Windows. Probably the proposed tools would provide a better performance. However, at the time of development they were not available for use on a PC.



COMPONENT PART NOTICE

THIS PAPER IS A COMPONENT PART OF THE FOLLOWING COMPILATION REPORT:

(TITLE): Conference Proceedings of machine Intelligence
for Aerospace Electronic Systems Held in Lisbon, Portugal on
13-16 May 1991 (L'Intelligence Artificielle dans les Systemes
Electroniques Aeroespaciaux)

(SOURCE): Advisory Group for
Aeronautical Research and Development, Neuilly-sur-Seine,
(France)

TO ORDER THE COMPLETE COMPILATION REPORT USE AD-N242025

THE COMPONENT PART IS PROVIDED HERE TO ALLOW USERS ACCESS TO INDIVIDUALLY
AUTHORED SECTIONS OF PROCEEDINGS, ANNALS, SYMPOSIA, ETC. HOWEVER, THE
COMPONENT SHOULD BE CONSIDERED WITHIN THE CONTEXT OF THE OVERALL COMPILATION
REPORT AND NOT AS A STAND-ALONE TECHNICAL REPORT.

THE FOLLOWING COMPONENT PART NUMBERS COMPRISE THE COMPILATION REPORT:

AD#: _____ TITLE: _____

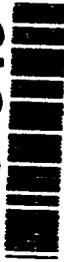
- AD-P006 331
- AD-P006 334
- AD-P006 336
- AD-P006 344
- AD-P006 345
- AD-P006 346
- AD-P006 352

DTIC
S NOV 13 1991 D

For publications and information
distribution to all users

Accession	
NRS C-21	
DND F-8	
Unannounced Justification	
By _____	
Distribution/	
Availability Codes	
Dist	Availability Special
A-1	

AD-P006 346



Using AITEST to Troubleshoot a Radar Modulator (RM) Unit:

A CASE STUDY IN THE APPLICATION OF EXPERT SYSTEMS TO INTERMEDIATE-LEVEL TESTING

Moshe Ben-Bassat*, Daphna Ben Arie, Israel Beniaminy, Jonathan Cheifetz, Michal Eshel,
Noam Fogel, Yael Karov, Irene Menkin, Mordechai Scla, Michal Shalev, Hagai Shemtov

IET - Intelligent Electronics

14 Raoul Wallenberg St., Ramat Hachayal

Tel Aviv, 69719, Israel

Tel: 972-3-497814 Fax: 972-3-497818

Abstract

AITEST is a real life expert system designed to serve as a decision aid and productivity tool for test engineers and technicians. Oriented for the functional level, AITEST is designed to troubleshoot large scale UUT's (Unit Under Test) that contain analog, digital and mechanical modules in electronic, electro-optic, hydraulic or mechanical systems and devices.

This paper describes a typical application of AITEST in an intermediate - level maintenance facility. The UUT (Unit Under Test) discussed in this application note is a Radar Modulator (RM) embedded in the Radar System of a military aircraft.

Following a brief description of the Radar Modulator structure and functions and its current test tools, we proceed to describing our experience in using AITEST to troubleshoot this device.

Keywords

- Diagnosis
- Expert Systems
- Equipment Maintenance

Introduction

AITEST is a real life expert system designed to serve as a decision aid and productivity tool for test engineers and technicians. Oriented for the functional level, AITEST is designed to troubleshoot large scale UUT's (Unit Under Test) that contain analog, digital and mechanical modules in electronic, electro-optic, hydraulic or mechanical systems and devices.

For a given set of test results, AITEST's diagnostic algorithm identifies candidate faulty modules and ranks them in a descending order of their likelihood of failure. It may also zoom-in for further assessing the fault likelihood of each individual sub-module. When the number of possible faults is relatively large so that no final conclusion can be immediately reached, AITEST identifies those areas in the UUT that are likely to contain faulty modules, and suggests diagnostic goals for the next stage. The user may accept these suggestions, or set his own diagnostic goals.

Once the goals have been set, AITEST identifies and evaluates the tests that may be used for achieving them, and proposes the most cost-effective test. The sequence of tests is by no means predetermined; rather it is adaptively reordered based on test responses, thereby eliminating redundant tests.

AITEST is capable of diagnosing multiple faults, and allows an unlimited number of hierarchical levels in the UUT structure. It is therefore applicable to all levels of testing; field service, intermediate labs and depot labs.

* Also with: Tel Aviv University Faculty of Management
Ramat Aviv, Tel Aviv 69978, Israel

AITEST also includes many non-AI (Artificial Intelligence) features that, together with the diagnostic engine, form a comprehensive solution for the practicing test engineers and technicians. These include:

OnDoc (on-line documentation) is AITEST's tool for producing and using textual and pictorial documentation of the device being tested, test instructions, repair procedures and logistics information. Based on object-oriented programming techniques, OnDoc offers the flexibility of a hypermedia/hypertext-like service manual.

AITEST's **DB-LRN** is a database tool used for storing and retrieving test results, diagnoses and repair actions. Beyond the internal database tools, AITEST can export its database files to commercial DBMS's. AITEST, of course, utilizes this database to learn from experience, continuously refining and improving its knowledge.

AITEST has been designed with an open architecture to enable flexible interfacing with test instrumentation and ATE. It can be used either manually, embedded in an ATE system or as a controller of test instrumentation.

AITEST has already been applied to numerous real life UUT's coming from a wide variety of fields; military and aerospace devices, peripheral equipment for computers (monitors, printers, disk drives), communication boards (PABX, radio telephone), automotive devices, and test

equipment. This paper describes a typical application of AITEST in an intermediate - level maintenance facility.

The UUT (Unit Under Test) discussed in this application note is a Radar Modulator (RM) embedded in the Radar System of a military aircraft. The RM unit is currently tested using a Radar Test Bench System (RTBS). RTBS is a traditional tester equipped with stimulus generators and measurement devices. A set of tests has been defined for the unit, for which the RTBS produces PASS/FAIL results. The primary diagnostic tool is a service manual that includes 50 pages of fairly lengthy diagnostic charts. Following a brief description of the Radar Modulator structure and functions and its current test tools, we proceed to describing our experience in using AITEST to troubleshoot this device.

Radar Modulator: Overview of Structure and Functions

The primary function of the RM is to produce a modulated pulse output, which, after amplification, activates the transmitter tube. The unit is functionally divided into the following modules (see Figure 1):

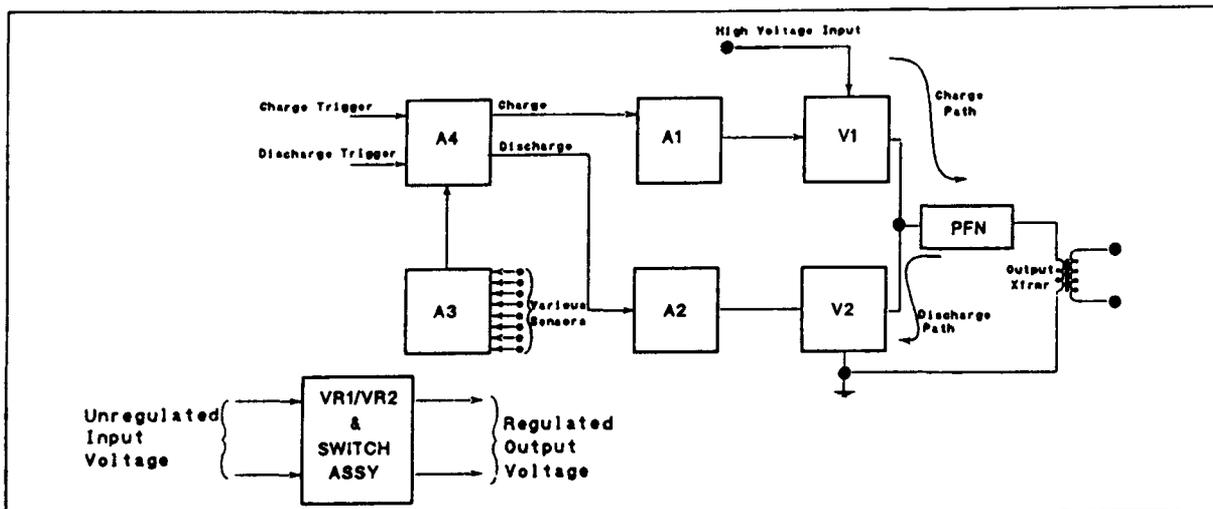


Figure 1: Radar Modulator Functional Description

1. V1 & V2 Charge and Discharge tubes, which act as switching devices.
2. A1 Charge trigger pulse generator module. Its main functionality is similar to that of an amplifier.
3. A2 Discharge trigger pulse generator module.
4. A3 Fault monitoring module.
5. A4 SCR gate driver whose main function is to enable/inhibit charge/discharge triggers depending on fault conditions detection (e.g. charge pulse generator overcurrent condition).
6. VR1 & VR2 Voltage regulators and switching assembly (K1 through K4).

Current practice in testing the Radar Modulator (without AITEST)

Using the service manual, a technician troubleshoots the RM in the following way:

The technician runs a series of about 20 pre-ordered Acceptance Tests (e.g. Time Totalizing Meter Test, Pulse Output Test etc.). If he completes running the acceptance tests without a single FAIL result, the testing procedure is terminated. In case of a FAIL result in one of the acceptance tests, the technician is referred to the diagnostic flowchart for fault isolation. The flowchart involves a total of about 23 diagnostic tests (e.g. checking test point number 164 - charge pulse gate test, or D14 + 25V regulated voltage test, etc.).

In some cases, the diagnostic chart ends with a definite isolation of the fault. In many other cases, however, the diagnostic chart ends with a group of suspected modules and leaves final isolation to the technician. For instance, one branch of the flow chart ends with: "Test and Repair V1, V2, U1-2...". In any case, the flow chart only refers to top-level modules, and further isolation is always left to the next stage of testing (sub-assembly testing).

Limitations of Current Practice

There are several major disadvantages in using the conventional diagnostic flowchart approach:

- a) The total number of test result combinations that need to be analyzed by any diagnostic chart is in the order of 2^n ; where n is the average number of diagnostic tests per case (assuming that every diagnostic test may have only a PASS or FAIL result).

For the RM unit, a typical number of tests per faulty unit is 16, which means that we have to specify the diagnostic interpretation of about 65000 (2^{16}) combinations of test results. Since it is inconceivable to cover all these combinations, the diagnostic chart does not go all the way down and often ends up with a list of suspected faulty modules. In these cases, final isolation depends on the experience, skill and knowledge of the individual technician who is testing the unit. This situation is, of course, undesirable in case of a junior inexperienced technician.

- b) The test execution sequence dictated by the chart is predetermined, and may not be the most cost-effective. Short-cuts through the flow chart are not allowed. If the technician deviates from the chart's guidance, the chart becomes useless for subsequent steps.
- c) The process of searching forward and backward through the service manual is cumbersome and quite time consuming. Quite often the operator forgets where he is coming from, or what are the test results so far. In fact, too often he ignores the service manual altogether.

- d) Service manuals do not preserve and make use of the experience (expertise) gained by the electronic engineers/technicians. When an experienced technician leaves, his knowledge is lost, and the less experienced operator starts from scratch.
- e) The time and cost of building a comprehensive service manual based on fault-isolation flow charts is very substantial. If the UUT goes through revisions (implying revisions in the test plan), the revision of its diagnostic chart is very difficult, and no inherent mechanisms are available to maintain consistency.

The following sections present the AITEST approach to fault isolation. The AITEST system consists of two main parts. The first part, the KB (Knowledge Base) program, is used to construct and maintain the knowledge base of the UUT (Unit Under Test). The second part, the RT (Run Time) program, drives the system during the testing process. Its core is the diagnostic executive, which operates on the knowledge base to guide the technician in troubleshooting the unit.

Preparation Stage: Creating the RM Knowledge Base (KB)

Using the KB program of Aitest, the construction of the UUT - specific knowledge base for the Radar Modulator took about 100 hours. This included:

- a) Creating the Radar Modulator's block diagram. Following an analysis of the UUT structure from the test point of view, we identified 25 top level modules, 18 of which are further subdivided into sub-modules. The total number of lower level SRRU's (Smallest Repair-Replaceable Units) is 95. Figure 2 shows the top level block diagram, while figure 3 shows the internal structure of one top level module.
- b) Describing 43 different tests, of which about 20 are grouped together in a section called ACCEPTANCE tests, and the rest are DIAGNOSTIC tests. These tests are the same tests used by the flow chart. Test description includes the test path and several other parameters such as stimulus and measurement types and the time it takes to perform the test.

The path of a test includes all modules whose malfunction may cause the test to FAIL. This information is crucial for obtaining correct diagnostic assessments. Therefore, the task of drawing the test paths should be put in the hands of a person who understands well the UUT and its tests. This person does not necessarily have to be the UUT designer (even though this is recommended). In fact, in the case of the Radar Modulator, the UUT-specific knowledge base was created by the engineers/technicians who maintain the system. (Over 10 years passed since it

91-15533


91 1113 023

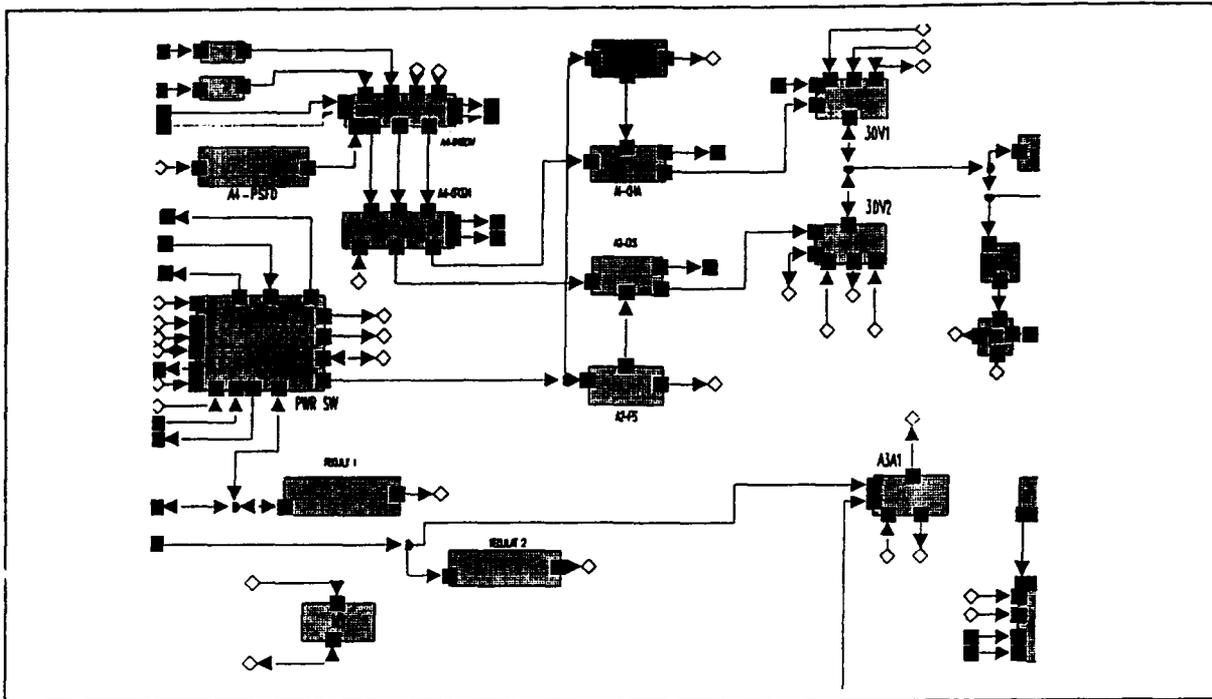


Figure 2: Part of the Radar Modulator's top-level block diagram

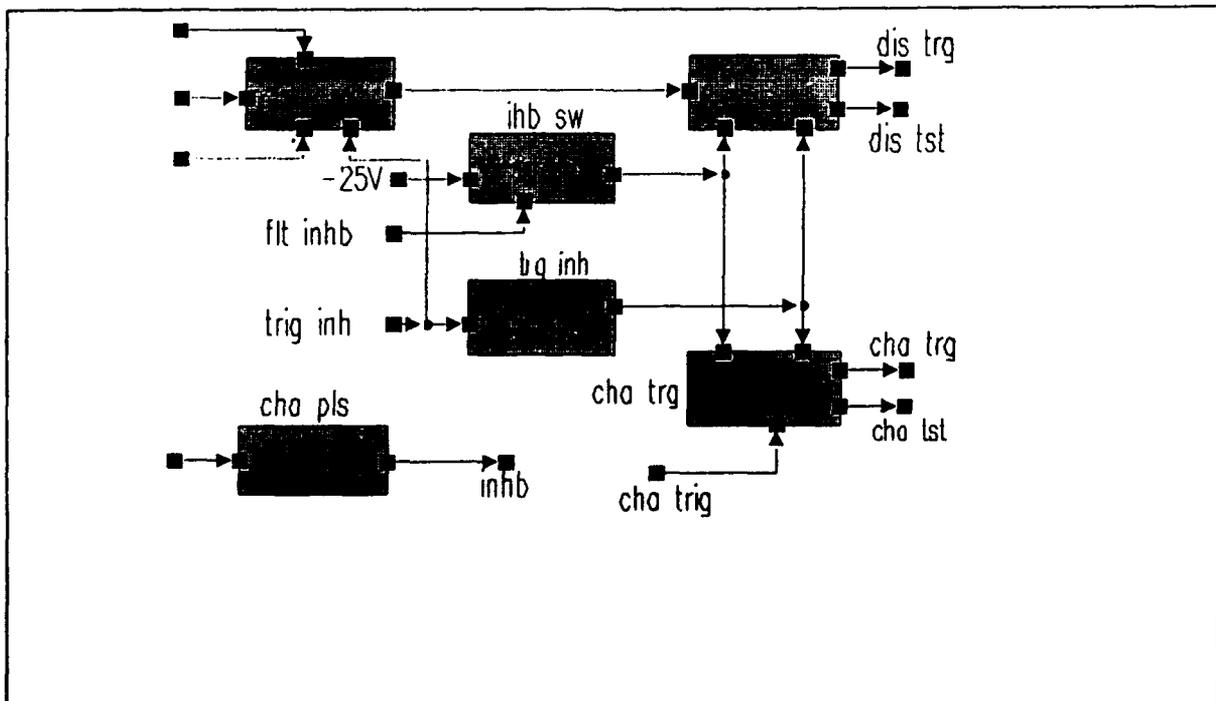


Figure 3: Radar Modulator's internal description of one top-level module (A4-INHIBIT SWITCH)

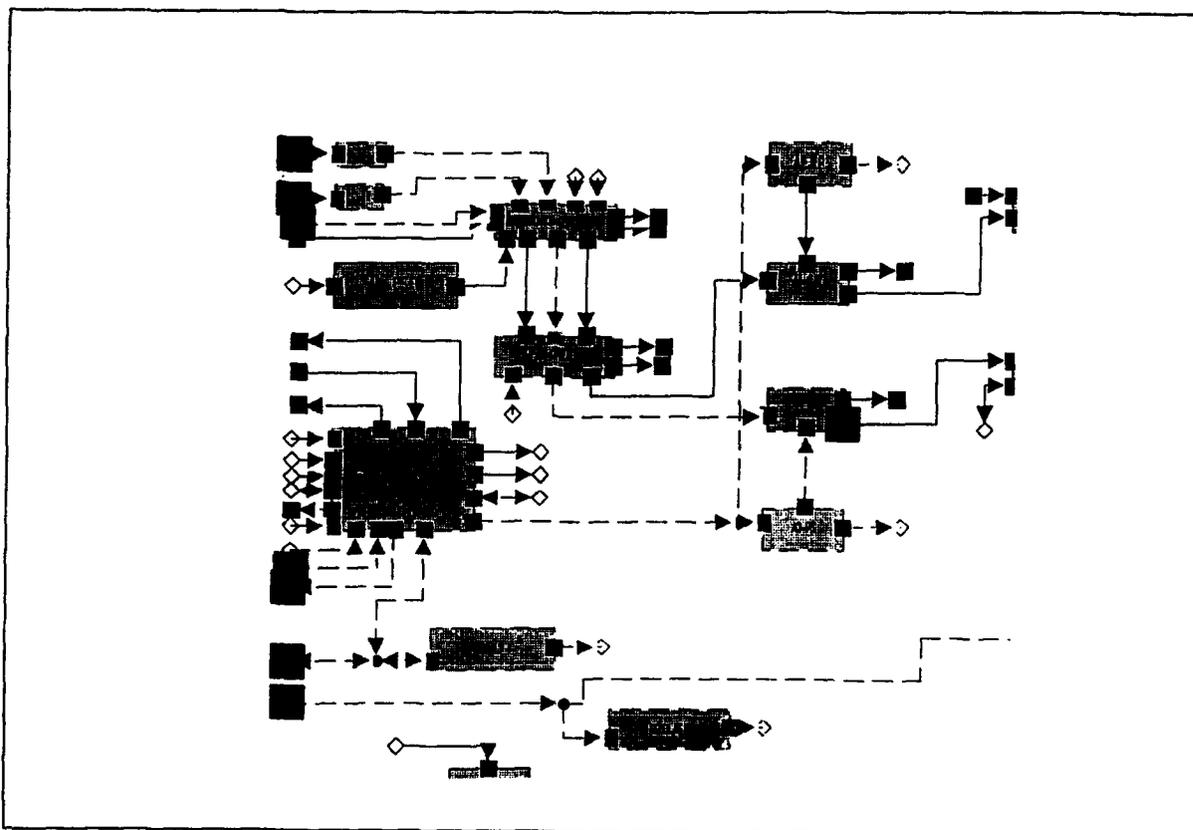


Figure 4: A typical example of a test path (test TP165PR) (dashed lines)

was delivered). The service manual and other documentation provided with the unit were very instrumental.

A typical example of a test path is shown in figure 4. Figure 5 shows the parameters required to fully specify the test. In this figure, the field "Test Program" contains the name of the test program that an ATE should run in order to execute this test. In the case of manual testing, this field may be left empty or it may include the name of a short program or text file that provides instructions to the technician.

The field "Description" provides the option to include documentation related to the test - e.g. a pictorial description of the location of stimulus and response points.

It should be emphasized that the diagnostic flow-chart of the RM is NOT programmed into AITEST. The basic concept behind AITEST's design is:

Tell AITEST (once only) about:

- the UUT structure, in terms of a functional block diagram, and
- the TESTS that may potentially be used (i/o points, test path, stimulus and measurement)

and AITEST will:

- automatically generate the diagnostic interpretation of any combination of test results, and
- effectively manage the order of test execution.

Once the description of the block diagram and the tests is completed, AITEST will comment on any inconsistencies or missing parts. After their correction, the KC (Knowledge Compilation) program is run. As a result, a set of files - RM-KB files - is generated, and this completes the preparation step. These files may be copied to diskettes and distributed to all of the service laboratories where the Radar Modulator is maintained. In the service laboratories, only the RT program of AITEST - described in the subsequent sections - is required.

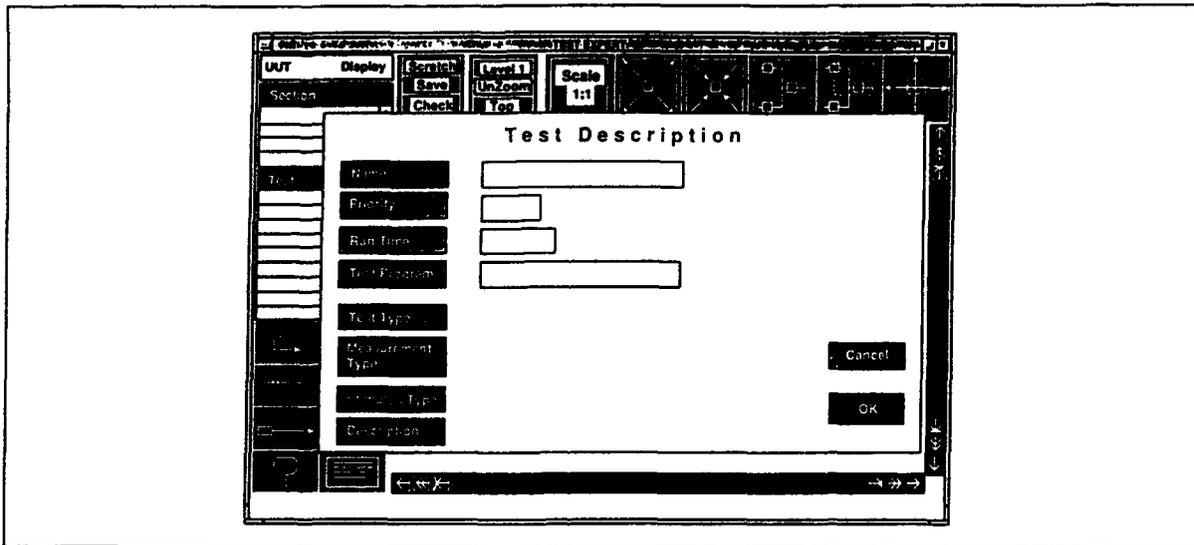


Figure 5: Parameters required to fully specify a test

Run Time (RT) Operation

The following sections describe the use of AITEST to troubleshoot the RM unit, using the RT program.

Test case - example 1

Let us assume that a fault exists in charge trigger generator-A1, which causes the absence of the charge pulse generator output. AITEST first recommends to perform acceptance tests until reaching a FAIL result. AITEST then deviates from the preordered sequencing and automatically identifies the next best test to perform at each stage.

The first 3 test results (consistent with A1 being faulty), are:

1. Time TOT (Time Totalizing Meter) = PASS
2. Time DLY (10 minutes Time Delay) = PASS
3. TP70-PR(is MOD Pulse Out Present?) = FAIL

At this stage, AITEST identifies a set of modules that are most worth pursuing at the next stage and sets them as the goals. AITEST continues by *automatically* selecting the next sequence of tests to be performed. Each test is selected depending on previous test results, and AITEST's estimation of the test's information contribution to further isolate the fault among the suspected modules. Here is what happens in our session:

1. TP163-PR (is TP163 output pulse present?) = PASS (This is the right response consistent with A1 being faulty).
2. TP163-CO (is TP163 Wave Form Correct?) = PASS

3. TP165-PR (is TP165 Output Pulse Present?) = PASS
4. TP165-CO (is TP165 Wave Form Correct?) = PASS
5. TP164-PR (is TP164 Output Pulse Present?) = FAIL
6. TP64 (is TP64 - Trig Inhibit - Present? if no = PASS) = PASS

At this stage AITEST displays the following message at the top left part of the display: "THERE ARE NO ADDITIONAL TESTS THAT CAN IMPROVE THE DIAGNOSIS". The central part of the screen shows the RM's block diagram in various colors where AITEST's diagnostic assessment is expressed by a color-coded representation (c.g. red for FAULTY, yellow for SUSPECTED etc.). This color actually encodes a numeric value indicating a fault probability interval. This probability interval is displayed by AITEST upon request. In our case the top ranked module is A1 whose color is reddish, indicating VERY SUSPECTED level (see Figure 6).

Notes

1. It took only 9 tests (out of about 50) to isolate the faulty module. AITEST automatically skips performing tests which are considered irrelevant. A test is considered irrelevant if it does not contribute additional information with regard to suspected modules which have to be further isolated
2. The operator, even an unskilled one, may easily follow the test sequence suggested by AITEST to isolate the fault and replace module A1 at this stage.

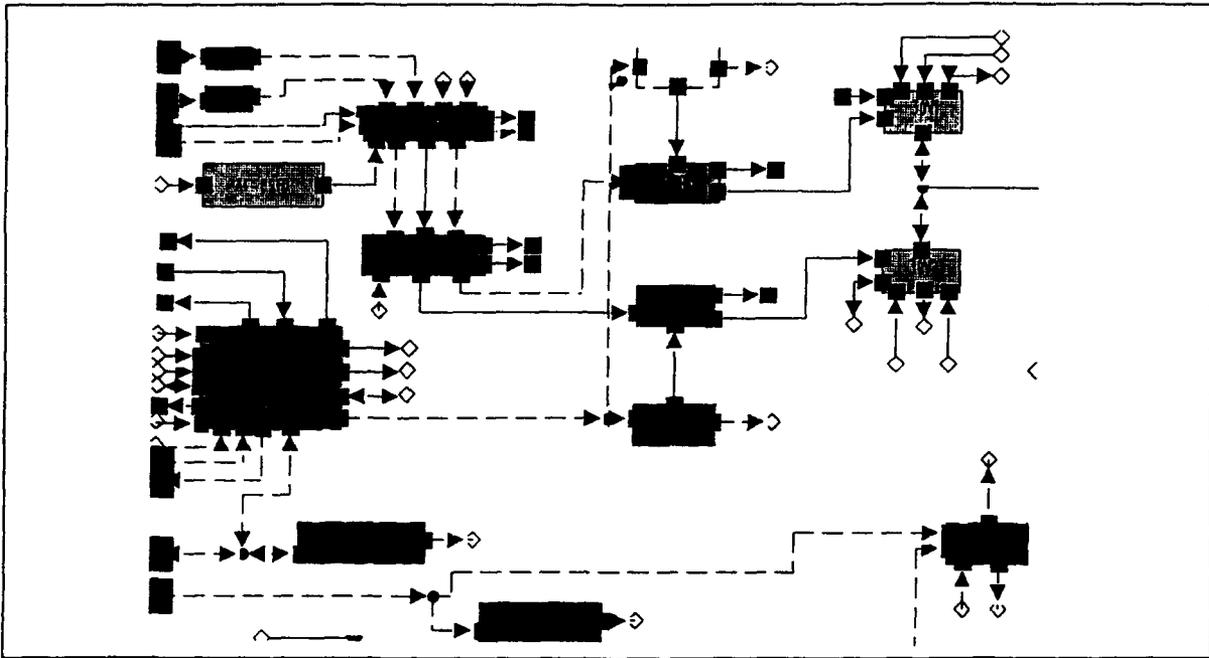


Figure 6: AITEST's diagnosis, a fault in VR1

Test case - Example number 2

Assume that a fault exists in VR1 (+25V regulator).

Again, AITEST recommends to perform a sequence of acceptance tests until reaching a FAIL result.

The first 3 test results (consistent with VR1 being faulty), are:

TIME TOT = PASS
 TIME DLY = PASS
 TP70 PR = FAIL (MOD PLS OUT is absent)

Following this FAIL result, AITEST dynamically selects the next tests, based on their estimated information contribution:

TP163 PR = FAIL
 BITE 5 = PASS
 BITE 6 = PASS
 BITE 7 = PASS
 BITE 8 = PASS

BITE 5-8 are a series of tests designed to check the fault monitor's (A3) - operation. Since A3 is not directly influenced by the +25V regulator, a "PASS" test result is indeed what was expected.

INV1 = FAIL
 TP165 = FAIL
 TP161 = FAIL
 TP158 = FAIL
 D14 = FAIL

At this stage AITEST stops and ranks VR1 (+25V regulator) at the top with a red color indicating a FAULTY state.

Notes:

1. Since VR1 is connected to the input voltage line, which affects many other modules, it took 13 (out of 50) tests to isolate the faulty module (VR1).
2. At the request to explain its line of reasoning for each module (or sub-module), AITEST displays the list of tests that contributed to the diagnosis directly or indirectly. This explanation may be further clarified by requesting the path of a test from the list to be displayed.

AITEST's Ability to Learn from Past Experience (Data Base and Learning Program)

As mentioned earlier in this article, service manuals do not preserve or make use of the expertise gained by the electronic engineers and technicians. AITEST uses statistical learning algorithms to improve its initial diagnostic ability. The improvement in diagnostic performance is proportional both to the number and variety of different test scenes saved in the statistical data base.

In this case study, (the RM UUT) about 50 test cases have been saved in order to refine AITEST's initial diagnostic estimates. This refinements brought AITEST's diagnostic performance to a stable high performance start point on the learning curve before putting it to actual use.

Obviously, the data base is useful in laboratory management as well as in learning. The lab manager can use AITEST's data base to generate statistics and reports, and can also port it to the general logistic data base.

Conclusions

The basic disadvantages of the conventional testing approach (using diagnostic flow-charts) are:

- a) Exponential (2^n) number of test result combinations to cover.
- b) Predetermined sequence of test execution
- c) Time consuming process of searching forward and backward through lengthy manuals.
- d) Service manuals do not preserve experiential knowledge.
- e) Time and cost of building comprehensive flow-charts.

AITEST solves these basic problems:

- a) It automatically generates the diagnostic interpretation of *any* combination of test results.
- b) It automatically and effectively manages the order of test execution.
- c) It provides on-line documentaion of the UUT including schematics, test instructions, repair/replace procedures, etc. Together with OnDoc, AITEST constitutes an electronic service manual of the UUT.
- d) It preserves and makes use of experiential knowledge (Data Base and Learning program).
- e) It eliminates the cost of building fault isolation flow-charts. .

References

- [1] *Ben-Bassat M., Ben-Arie D., Beniaminy I., Cheifetz J. and Klinger M.*, AITEST - A real life expert system for electronic troubleshooting (A description and a case study), Proc. IEEE Conference on AI Applications, San Diego (1988).
- [2] *Ben-Bassat M., Ben-Arie D., Ben Zvi I., Beniaminy I., Cheifetz J, Sela M., Shalev M.* - Evaluation of real life expert systems. Proc. Autoestecon'89, Philadelphia (1989).



COMPONENT PART NOTICE

THIS PAPER IS A COMPONENT PART OF THE FOLLOWING COMPILATION REPORT:

(TITLE): Conference Proceedings of machine Intelligence
for Aerospace Electronic Systems Held in Lisbon, Portugal on
13-16 May 1991 (L'Intelligence Artificielle dans les Systemes
Electroniques Aero spatiaux)

(SOURCE): Advisory Group for
Aeronautical Research and Development, Neuilly-sur-Seine
(France)

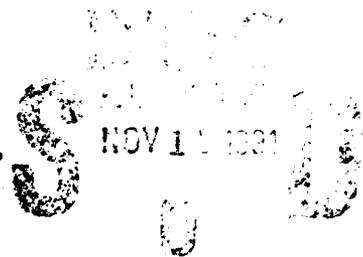
TO ORDER THE COMPLETE COMPILATION REPORT USE AD-M242025

THE COMPONENT PART IS PROVIDED HERE TO ALLOW USERS ACCESS TO INDIVIDUALLY AUTHORED SECTIONS OF PROCEEDINGS, ANNALS, SYMPOSIA, ETC. HOWEVER, THE COMPONENT SHOULD BE CONSIDERED WITHIN THE CONTEXT OF THE OVERALL COMPILATION REPORT AND NOT AS A STAND-ALONE TECHNICAL REPORT.

THE FOLLOWING COMPONENT PART NUMBERS COMPRISE THE COMPILATION REPORT:

ADM: _____ TITLE: _____

AD-P006 347



Accession For	
NTIS - ORNL	<input checked="" type="checkbox"/>
NTIS - TAB	<input type="checkbox"/>
Microfilm and	<input type="checkbox"/>
J. Information	<input type="checkbox"/>
By _____	
Distribution _____	
Availability Codes	
Dist	Avail. or for Special
A-1	21

This document has been approved for public release and sale; its distribution is unlimited.

AD-P006 347

A KNOWLEDGE-BASED ASSISTANT FOR DIAGNOSIS IN AIRCRAFT MAINTENANCE

by

M.A. Piers and J.C. Donker
 National Aerospace Laboratory NLR
 P.O. Box 90502
 1006 BM Amsterdam
 The Netherlands

1. SUMMARY

As a result of the demands upon maintenance organisations to increase availability of aircraft and the increasing complexity of aircraft systems, a need for tools and facilities to enhance the effectivity and efficiency of maintenance emerges. The National Aerospace Laboratory NLR performs applied research with concern to the use of knowledge based systems for condition monitoring and diagnosis in complex technical systems. This paper describes a feasibility study (KADAM) on the application of knowledge based systems for diagnosis of complaints in aircraft systems. The specific application selected for the KADAM project (Knowledge-based Assistant for Diagnosis in Aircraft Maintenance) is a knowledge based system to be used by ground engineers for troubleshooting of an aircraft airconditioning system.

This paper will address the approach taken in the project and review the results, including the design of the proof-of-concept system. Particular attention will be paid to the identification and formalisation of methods for diagnosis.

assistant could provide easy and rapid access to the information which is normally acquired from bulky manuals. Also, knowledge and experience from experts could be made available to the less experienced ground engineer. Maintenance data bases could be coupled to the KBS in order to provide component reliability information and the maintenance history of a complaint.

These KBS facilities should decrease the time spent on aircraft systems maintenance and the number of incorrect component removals (RTOK). There might also be effects towards decreased knowledge and experience requirements for ground engineers.

Although a KBS solution is not a condition to the objectives mentioned, a KBS approach does have the advantage of enabling a relatively clear separation between knowledge on (generally applicable) diagnosis methods and knowledge on system structure and -functionality. Also the explanation feature of a KBS facilitates learning and lessons learned from operational use of the system can be relatively easy implemented.

2. INTRODUCTION

Maximum deployability of aircraft is of prime importance to military operators. This imposes high demands upon the maintenance organisation to ascertain maximum availability of the fleet. At the same time, maintenance costs are to be kept at a minimum. Hence, a need for maintenance efficiency with respect to time as well as for effectiveness continuously exists. This problem is compounded by the ever increasing complexity of aircraft systems requiring more knowledge and experience from the maintenance crew.

A solution to the problems outlined above could be the availability of a knowledge-based system (KBS) to assist the maintenance crew in the process of diagnosing failures in complex systems. A knowledge-based diagnosis

As a result of the above, world-wide development activities have started on knowledge-based systems for diagnosis in aircraft systems as well as entire aircraft. Fault diagnosis systems are being developed as on-board aircraft systems and as ground-based systems.

Under contract with the Netherlands Agency for Aerospace Programs (NIVR), the National Aerospace Laboratory (NLR) has investigated the feasibility of knowledge-based systems to assist ground crews in corrective maintenance of complex aircraft systems. For this feasibility study, the Fokker 100 air-conditioning system was chosen as a representative example of a complex aircraft system. One of the results of the KADAM-project (Knowledge-based Assistant for Diagnosis in Aircraft Maintenance) is a proof-of-concept knowledge-based system.

3. PROBLEM DEFINITION

Maintenance may in general terms be defined as the activities required to bring an item into a functional condition or to keep an item in a functional condition. Hence, maintenance can be divided into corrective activities and preventive activities respectively. Corrective maintenance due to problems which occurred during operational use of an aircraft is commonly referred to as "troubleshooting".

If a malfunction occurs during

aircraft operation, the crew fills out a complaint form ("slip") which describes the malfunction, flight conditions related to the occurrence, possible action taken by the crew (in-flight trouble shooting or fault containment) and any other information of relevance to the complaint.

The slip (Fig. 1) is the main source of complaint information used by the ground engineer for diagnosis of the problem during corrective maintenance on the ground.

FLIGHT NO.	DEP STA	A/C REG	DOMNY	SEQ No	ATA SYSTEM	POS	KSSU COMP IDENT NO	SERIAL No IN	SERIAL No OUT	DEL												
416	BFS	CNB	100186	NO46																		
SUBJECT : Pressurization COMPLAINT : Upon lift off and touchdown there is a pressure surge for a few seconds of approx. 2000'/min				ACTION : Replaced cabin press controller Out: 102644-6 # 115-359 In : 102644-6 # 96-883 <table border="1" data-bbox="768 745 1115 913"> <thead> <tr> <th>ACTION STATION</th> <th>DD MM</th> <th>GMT</th> </tr> </thead> <tbody> <tr> <td>AMS</td> <td>1101</td> <td>0300</td> </tr> <tr> <th>RELEASE STATION</th> <th>DD MM</th> <th>GMT</th> </tr> <tr> <td></td> <td></td> <td></td> </tr> </tbody> </table>							ACTION STATION	DD MM	GMT	AMS	1101	0300	RELEASE STATION	DD MM	GMT			
ACTION STATION	DD MM	GMT																				
AMS	1101	0300																				
RELEASE STATION	DD MM	GMT																				

Figure 1 Example complaint form (slip).

Diagnosis takes approximately one third of the total time used for corrective maintenance.

It is a task which in general requires a considerable amount of knowledge and experience of the ground engineer. Errors in diagnosis result in high costs, not only because the problem will not be solved, additional maintenance is required and aircraft availability is reduced, but also because fully servicable system components may be replaced, resulting in unnecessary component maintenance activities (RTOK: Re-Test O.K.) and large and expensive stocks of spare parts.

The proof-of-concept knowledge based system developed under the KADAM project provides support to diagnosis activities. It helps the engineer to establish the exact complaint, to identify and evaluate the symptoms and to determine which maintenance action is required to resolve the problem.

In order to limit the scope of the project, one aircraft system was selected as the application area, viz. the airconditioning system of the Fokker 100 aircraft. This system was selected because it has well defined interfaces with other aircraft systems

and it contains mechanical-, pneumatic- and electronic components. The Fokker 100 airconditioning system (airco) is considered representative for modern, non-avionics aircraft systems.

The main differences between the KADAM application and other AI avionics testers are that there is no single system testunit available, system components are distributed over the entire aircraft, components are often not directly accessible and the fact that the maintenance engineer usually has to work from functional discrepancies (complaint descriptions) instead of BIT/BITE results (Built-In-Test data).

4. KNOWLEDGE ACQUISITION

The two main areas of knowledge applied during troubleshooting of aircraft systems are knowledge of the structure and functionality of the system which in general are documented during system design, for example in a Fault Isolation Manual, and troubleshooting experience of ground engineers. Knowledge from experience is usually transferred to less experienced personnel through personal communication and does in general not

result in effective feedback to the manufacturer of the aircraft. Because these two knowledge areas are partially overlapping and may show discrepancies, both areas were surveyed extensively. Maintenance documentation and design data has been provided by Fokker Aircraft B.V., the manufacturer of the Fokker 100. Information on maintenance operations has been made available to the KADAM project by Martinair Holland and KLM Royal Dutch Airlines, the operators of the aircraft.

The primary airconditioning maintenance document utilized in the KADAM project is the Fault Isolation / Maintenance Manual. This manual provides Fault Code Diagrams (basically logic trees of system functionality) which are used by the ground engineer to establish a Fault Code. The Fault Code determines which corrective action must be taken.

It should be noted that a model of system structure and -functionality of a sufficient level of detail to allow causal reasoning was not envisioned for KADAM because the effort required would be of prohibitive scope. In addition, the functional- and structural relations that could be determined using the model are already generated for maintainability and reliability analysis purposes during system design. The results of these analyses are implicit representations of system structure and -functionality. One of the secondary objectives was to determine whether these implicit representations of system structure and functionality could be utilized to build a working KBS.

Operational maintenance information has been acquired through involvement of a member of the project group in the maintenance organisation. The main objective of this period was to identify diagnosis methods as utilized by ground engineers. At the central maintenance support center of the airline all complaint slips are kept on file for later reference. A survey of the slips concerning airconditioning complaints provided insight in complaint descriptions, complaint histories, failure distributions, etc. Participation in the maintenance activities of ground engineers enabled an assessment of operational aspects of maintenance and the diagnostic procedures applied by experienced maintenance personnel. One of the methods employed in order to identify methods was by means of scenario's. Scenario's are virtual complaints, prepared by the project team, which were presented to an

experienced ground engineer in a format identical to a normal complaint slip. The engineer was asked to diagnose the complaint and to contemplate on the thought process behind the diagnosis.

In addition, a large number of slips covering five years of operation with a fleet of five aircraft were reviewed by a member of the project group together with a senior ground engineer in order to infer the diagnostic reasoning. Since related slips (of a re-occurring complaint) were grouped together, it was also possible to assess the differences in the diagnoses as a result of the availability of the complaint history in the form of subsequent slips.

5. ANALYSIS

5.1 Diagnosis Methods

One of the objectives of the KADAM project was to identify and formalize diagnosis methods. These methods may be generally applicable, hence could be used in other applications. This activity not only created a frame work for the knowledge based system, it also facilitated the recording of maintenance- and diagnosis experience. Analysis of the interviews on the scenario's and slips, showed that particular portions of the available information and particular operations with this information are used as standard elements of the diagnostic process.

These elements and the relations between these elements are combined and utilized by the ground engineers in several different ways, depending on the operational situation and the available information. Three different diagnosis methods can be identified which do not only differ with regard to the elements utilized, but also in the relations between the elements. Not all elements and relations are used for each of the three methods. A summarized description of the elements, relations and methods is provided below. For a full description refer to (Piers and Donker, 1989.).

Observables: all information related to system condition that is available or can be made available on the aircraft for diagnosis .

Symptoms: all Observables which are related to an event that results in a complaint. A failure may induce Symptoms.

Hypothesis: a mental representation (of the ground engineer) concerning a

91 1113 024

91-15532



possible Cause of the occurring Observables.

Causes: events, states or conditions which result in Observables.

Knowledge: the aggregate of system-structure and -functionality knowledge (declarative), knowledge from experience (heuristic) and knowledge of the history of a complaint.

Compare: an operation in which information resulting from different elements but concerning the same subject is related to each other.

Solution: an action which eliminates a Cause.

Test: an action which determines whether the Symptoms still persist after the Solution has been implemented.

Arrows indicate non-specific relations between elements. This may be a simple sequential relation but also the fact that an element influences the relation between two other elements.

Method 1

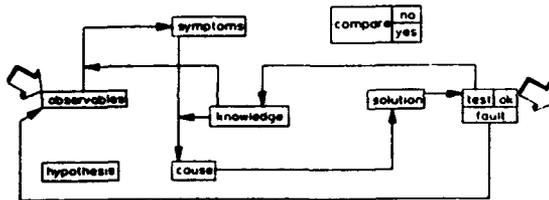


Figure 2 Method 1.

In the entire spectrum of observables, some observables do occur which are determined to be specific to a failure (symptoms). From these symptoms, a cause is inferred which is eliminated (solution). Then a test will be performed in order to conform that the symptoms do no longer occur.

Method 2

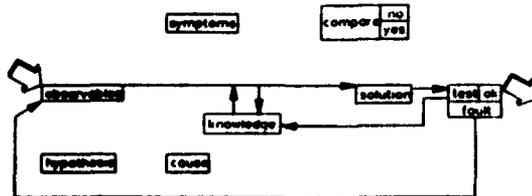


Figure 3 Method 2.

The second method is a sequence of elements which may not even be called

a diagnostic procedure. It is however a way of solving complaints which is applied in aircraft maintenance. This method represents the situation where there is no time to perform an elaborate diagnosis and the particular complaint does not pertain to a critical system function. Findings of this project and other studies indicate that in these cases, the maintenance crew will select an action based on previously encountered Observable-Solution relations.

Method 3

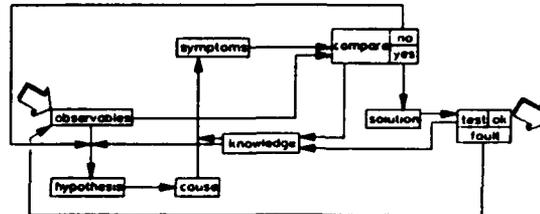


Figure 4 Method 3.

In short, the maintenance crew forms a hypothesis on the cause of a complaint based on the available information. The cause hypothesis results from the observables and is based on knowledge. Then the symptoms which can be expected as a result of this hypothetical cause are determined and subsequently compared to the observables which occurred in reality. If the symptoms are a subset of the observables, the hypothesis is not negated. If enough time is available, tests are usually performed to check the correctness of the hypothesis.

5.2 Diagnosis Bottlenecks and the Role of Experience

Based on the information accumulated and analysed in the KADAM project, a number of problems with concern to the diagnosis aspects of maintenance have been identified. These problems exhibit a negative influence on the efficiency of maintenance.

The main cause of diagnosis difficulties is the information which is (not) written on the complaint forms by the flight crew. Frequently, the minimum information required for diagnosis is lacking. For example, the flight conditions at the moment of occurrence of the complaint are not always included in the complaint description. Because the possibilities to simulate flight conditions on the ground are very limited, a considerable portion of the complaints (28%) can not be duplicated on the ground (CND) and can therefore not effectively be solved. The survey of

all slips also indicated that slightly different descriptions of complaints with the same cause result in different diagnosis results and hence different corrective actions.

The accessibility of the documentation for system- and maintenance information hinders the use of these sources of information. Manuals are bulky, refer to other manuals which may not be readily available and manuals often do not address complaints that are frequently encountered. In addition, modifications to aircraft systems are often implemented long before the applicable manuals are updated.

A considerable amount of the knowledge of the experienced ground engineer concerns efficient use of the available manuals. Lack of relevant experience may hamper diagnosis, in particular if complaint information is incomplete or uncertain and when the most likely cause must be selected from a number of possible causes. On the other hand, the KADAM project showed that a number of skills attributed to experience (heuristics) in reality concern common knowledge of the structure and functionality of the system (declarative). A common example of this declarative knowledge often mistaken for heuristic knowledge concerns complaints which do not require corrective action because the observables mentioned in the complaint description are normal flight phase-dependent variations in system performance.

Another major bottleneck in diagnosis is the lack of information on the history of a complaint. Frequently, corrective actions which do not solve the problem are repeated when the complaint occurs again because the maintenance crew is not aware of the previous maintenance actions related to this particular complaint. This precludes a correct perception of failure rates of system components which may be of importance to selecting a most likely cause from a number of possible causes. This problem is compounded by a lack of feedback to the maintenance crew of the results of shop maintenance on the replaced components. The fact that the ground engineer is not informed about whether or not the replaced component was really a failed item and hence the cause of the problem, does not facilitate an effective build-up of experience.

6. DESIGN AND IMPLEMENTATION.

Because the KADAM project is a

feasibility study the main objective of the project is to demonstrate that relevant knowledge on aircraft maintenance can be acquired and utilized to implement a proof-of-concept knowledge based system. Therefore, and as a result of budget constraints, the project plan called for a decision on which knowledge would be implemented after the knowledge acquisition phase. It was determined to implement method 1 only since the information required is readily available.

As shown in Figure 2, this method uses the symptom to arrive at a cause which implies a so-called top down approach. Relations of this type are usually derived using Fault Tree Analysis (FTA), the results of which can be found in the Fault Isolation Manual (FIM). Hence the FIM may be utilized as a basis for implementing Method 1. The FIM contains implicit knowledge on the structure and functionality of the system.

The design of the proof-of-concept system is such that the two other methods, can be added later.

Design and implementation of the proof-of-concept system has been based on NEXT 2 (NLR Engineering X-pert system Toolkit). NEXT 2 proved to possess the capabilities needed to meet the requirements. A diagnosis method was implemented. Descriptive knowledge on structure and functionality are represented in frames, the problem solving knowledge was implemented in production rules.

At the start of a consultation the user chooses one of the three methods for the diagnosis. (As mentioned before, only method 1 is currently available). The user provides some initial information a.o. aircraft tailnumber, departure station etc., (which is used for logistic purposes and to facilitate access a complaint history database) and an indication in which subsystem the complaint occurred. Thereafter, KADAM queries the user in order to further identify the specific complaint. This process results in the identification of a Fault Code which references to the Maintenance Manual. If the user continues the consultation, KADAM asks a number of additional questions in order to assess which symptoms do occur, this may involve requests from KADAM to perform tests. Based on the symptoms established, KADAM identifies the failed component and also provides an overview of the maintenance task required to solve the problem. In case it is not possible to narrow the possible causes down to one component,

the possibly failed components are listed in order of probability. So as to demonstrate the ability of the knowledge based system to interface with external software and databases, the interface capabilities of NEXT 2 have been used to perform calculations with external FORTRAN algorithms and to survey files from an external SQL database.

7. TEST AND VALIDATION

Validation testing has recently been carried out by operator- and manufacturer representatives. Although the results are still being evaluated, the main outcome is that the majority of the requirements have been met. However, the constraints to the scope of the implementation (only method 1), rendered a system which exhibits some of the shortcomings also encountered when using manuals for diagnosis which is no surprise since the implementation is based on the manual. Of course the rapid and easy access to the information provided by the proof-of-concept system facilitates a timely diagnosis. In addition, the lack of an adequate complaint description remains a prohibitive factor to effective diagnosis for the manual procedure as well as the knowledge based system supported procedure.

8. FURTHER WORK

One of the main issues for further work after the feasibility phase of the KADAM project will be the implementation of methods 2 and 3. Experience on symptom-cause relationships can be collected and implemented in method 2 but the relationships are based on subjectively perceived failure rates and symptom-solution relations which are not always supported by feedback on correctness of a maintenance action. Therefore the validity is limited and method 2 can probably be implemented in a more effective way through reference to a valid component- or subsystem reliability database (which incorporates feedback from shop maintenance) and a complaint history database.

As shown in figure 4, method 3 requires knowledge to infer a hypothesis from the observables and knowledge to predict symptoms from the hypothetical cause. The first relation is not unlike method 1, potentially based on incomplete and/or uncertain information. What needs to be added to implement method 3 is the second relation: reasoning from a cause to

symptoms. The knowledge/information required to implement this relation is in many cases generated during system design. The Failure Modes and Effects Analysis (FMEA) is a bottom up analysis which is used to determine the effect of component failures on system functions. (Note that this is basically the reverse of the FTA). Based on knowledge of the functional- and structural relations of the system every possible component malfunction is evaluated for its effect on system functioning, which means that the symptoms of the failure are predicted. Consequently, this information can be used to implement method 3.

The FMEA of the Fokker 100 airconditioning system was however not available in a format which could be used in the KADAM KBS without a major conversion effort. This fact bears another general observation: a considerable amount of the information required to build a KBS for system diagnosis is readily available or can be made available with relatively little effort during system design. If however, the utilisation of design analysis (intermediate) results for a KBS is not foreseen during design, the required information may not be preserved at all or may be generated in a format which does not facilitate its use for a KBS.

9. CONCLUSIONS

The main conclusion of the KADAM project is that knowledge of the structure and functionality of a complex system can indeed be implemented in a knowledge based system which provides useful support for diagnosis. The knowledge used to build the KBS is generated during aircraft system design and is implicitly available in the results of maintainability and reliability analyses. The KADAM project also demonstrated that experience based methods for diagnosis as applied by maintenance personnel can be identified, formalized and implemented. The role of experience in these methods can be identified and formalized. It was shown that a considerable portion of maintenance skills which are attributed to experience are in fact based on declarative knowledge which is implicitly available in the proof-of-concept knowledge based system such as flight phase dependent valid ranges of performance parameters and effective use of manuals. Experience in the form of "tricks of the trade" such as alternative ways of finding symptoms can be identified formalized, but

direct integration with methods 1 and 3 may not be advisable and result in inconsistencies, the experience should be validated first.

A further conclusion does not directly concern the knowledge based system itself or its development but the complaint information available for diagnosis. If insufficient complaint information is available, in particular with regard to flight conditions, system configuration and symptoms at the moment of occurrence of a complaint, an effective diagnosis is often precluded. Providing complaint information is primarily a task of the flight crew. Filling out a complaint form may interfere with cockpit duties and essential information is often omitted. A knowledge based system is not likely to reduce this problem. Obviously, on-board monitoring systems could provide valuable information in this realm.

A final finding of the KADAM project which has also been shown in other studies is that a limited availability of information on complaint history may cause duplications of faulty maintenance actions. A knowledge based system can provide access to a complaint history database and take complaint history into account when selecting a cause from a number of possible causes.

REFERENCES

Ali, M. and Scharnhorst, D.A. (1985). Sensor-based fault diagnosis in a flight expert system, Proc. 2nd Conf. on AI Applications (CAIA'85), pp. 49-54.

Chandrasekaran, B. and Mittal, S. (1983) Deep versus compiled knowledge approaches to diagnostic problem solving. Int. J. Man-Machine, Vol. 19, pp. 425-436.

Czeh, R. (1957). Studies of complex behaviour and their relation to troubleshooting in electronic equipment. Human Resources Research Office, George Washington University.

Davis, R. (1984). Diagnostic reasoning based on structure and behaviour. Artificial Intelligence, Vol. 24, pp. 347-410.

Dekleer, J. and Williams, B.C. (1987). Diagnosing multiple faults. Artificial Intelligence, 1987, pp. 97-130.

Milne, R. (1987). Strategies for diagnosis. IEEE Transactions on Systems, Man and Cybernetics, Vol.

SMC-17, No 3, pp. 333-339.

Pau, L.F. (1986). Survey of expert systems for fault detection, test generation and maintenance. Expert Systems, Vol. 3, nr. 2, pp. 100-108

Piers, M.A. and Donker, J.C. (1989). Research into the feasibility of knowledge-based systems for aircraft maintenance - Theory and practice of diagnosis in aircraft maintenance. National Aerospace Laboratory NLR, NLR TP 89185.

Rouse, W. (ed.), (1980). Symposium on human detection and diagnosis of system failures, Roskilde, Denmark, 1980.

Young, M. (1987). A framework for describing troubleshooting behaviour using default reasoning and functional abstraction. IEEE Proc. 3rd Conf. on AI Applications (CAIA'87), Feb. 1987, pp. 156-162.

Discussion

I. B. Tarhan, Turkey

What is your method for fault isolation which is not observed by ground crew people?

Author:

KADAM KBS is used for troubleshooting of complaints occurring during operational use and reported by the flight crew or observed by maintenance crews during turn-around/pre-flight. Faults not reported are not addressed. If a fault reported by the flight crew can not be duplicated on the ground, for example because flight conditions can not be simulated, KADAM will still support diagnosis based on complaint description.



COMPONENT PART NOTICE

THIS PAPER IS A COMPONENT PART OF THE FOLLOWING COMPILATION REPORT:

(TITLE): Conference Proceedings of machine Intelligence
for Aerospace Electronic Systems Held in Lisbon, Portugal on
13-16 May 1991 (L'Intelligence Artificielle dans les Systemes
Electroniques Aero spatiaux)

(SOURCE): Advisory Group for
Aeronautical Research and Development, Neuilly-sur-Seine
(France)

TO ORDER THE COMPLETE COMPILATION REPORT USE AD-M242025

THE COMPONENT PART IS PROVIDED HERE TO ALLOW USERS ACCESS TO INDIVIDUALLY AUTHORED SECTIONS OF PROCEEDINGS, ANNALS, SYMPOSIA, ETC. HOWEVER, THE COMPONENT SHOULD BE CONSIDERED WITHIN THE CONTEXT OF THE OVERALL COMPILATION REPORT AND NOT AS A STAND-ALONE TECHNICAL REPORT.

THE FOLLOWING COMPONENT PART NUMBERS COMPRISE THE COMPILATION REPORT:

ADM: _____ TITLE: _____

AD-P006348

DTIC
ELECTE
NOV 13 1991
S D D

Accession For	
NTIS CRA&I	<input checked="" type="checkbox"/>
DTIC TAB	<input type="checkbox"/>
Unannounced	<input type="checkbox"/>
Justification	
By _____	
Distribution/	
Availability Codes	
Dist	Avail and/or Special
A-1	21

This document has been approved for public release and sale; its distribution is unlimited.

AD-P006 348

INTEGRATED COMMUNICATIONS, NAVIGATION, IDENTIFICATION, AVIONICS (ICNIA) EXPERT SYSTEM FOR FAULT TOLERANT AVIONICS

by

Mark E. Minges
WL/AAAI-3
Wright-Patterson AFB, Ohio
45433-6543 USA

1. SUMMARY

Embedded Avionics Systems for aircraft are becoming increasingly more complex. Very High Speed Integrated Circuits (VHSIC) and semi-custom devices are used to gain many-fold increases in processing power and capability. Mission and operational requirements dictate a high availability and fault detection capability defined quantitatively as ninety-eight percent detection of all faults and isolation of ninety percent of those faults to a line replaceable module (LRM), or ninety-five percent of the faults to two LRMs. The Integrated Communications, Navigation, Identification Avionics (ICNIA) program utilizes a module Maintenance Node (MN) which aids high speed testing of the LRM, and gives the ability to isolate the fault (s) to one or more modules. The MN uses the concepts of set scan design, pseudo-random test vector generation, output response compression, and separate set scan loops to test the Small Scale Integration (SSI) - Medium Scale Integration (MSI) logic on the LRM. The resultant data is made available to the expert system within ICNIA so that real-time fault detection and isolation can be achieved.

The objective of the expert system within ICNIA is to detect and isolate faults in near real-time and minimize the false alarm rate. The technical approach is to utilize Built-In-Test (BIT) as the data collection mechanism for the expert system. BIT commands the resources to perform online and offline test via MNs and reports the results to the Integrated Test and Maintenance (ITM) function. ITM contains the expert system and performs the fault isolation and detection using a forward and backward rule-based design in real-time in the aircraft. The payoffs include ninety-eight percent detection of all faults, in flight reconfiguration, increased operational availability and improve supportability.

2. Introduction

Avionics systems continue to incorporate increasing amounts of technology to meet processing demands and in turn the complexity of these systems also increase. With higher complexity, the cost also increases and thus greater emphasis is placed on maintainability and reliability. Complicated equipment means faults are more difficult to locate and require more time for repair and replacement, which means an overall decline in reliability.

Artificial Intelligence (AI) is a rapidly developing technology which produces a number of tools with the potential of solving many problems that arise in complex electronic systems. The Department of Defense (DOD) has realized the importance of AI in solving many of today's high-tech problems and thus is investing in AI solutions.

A subset of AI is the expert system. An expert system is a computer program that performs specialized knowledge-based tasks. The Integrated Communications, Navigation, Identification Avionics (ICNIA) program utilizes an expert system to employ fault-tolerance, reconfiguration and graceful degradation. The expert system improves system readiness and raises the probability of mission success.

3. What is ICNIA?

ICNIA is a tri-Service program with the Air Force, Army and Navy as participating services. ICNIA employs advanced state-of-the-art technologies to develop several Advanced Development Model (ADM) units that will be tested in a laboratory environment and undergo flight demonstration. Its key objectives are to demonstrate that multiple Communications, Navigation, Identification (CNI) functions can be integrated into a modular airborne avionics system.

The ICNIA terminal concept relies on the use of rapidly emerging advanced Radio Frequency (RF) and digital technologies to provide aircraft pilots more CNI availability and flexibility than ever before. By incorporating advanced technologies such as Built-In-Test (BIT) and reconfigurability, the ICNIA terminal is capable of maintaining operation through multiple failures. With the use of BIT technology, in-flight reconfiguration is possible to support the pilot's needs for different CNI function priorities throughout his mission.

Prior to ICNIA, individual avionics systems were developed for each new requirement resulting in a proliferation of discrete systems. The ICNIA system consolidates into a single integrated terminal as many as sixteen separate avionics systems, operating in the frequency band from 2 MHz to 2 GHz. ICNIA is composed of a minimum number of dynamically reconfigurable modular building blocks which can be used to tailor the system capability for particular mission or aircraft and will greatly increase mission availability, reduce single point failures; and greatly reduce maintenance and consequently Life Cycle Costs.

4. ICNIA DESCRIPTION

Before discussing the fault tolerant and machine intelligence aspects of the system an overview of ICNIA will be given. ICNIA is a complex multi-function, multi-processor system with real-time processing requirements. Key elements to be described include the functional requirements, the architecture, the hardware and the software.

4.1 Functional Requirements

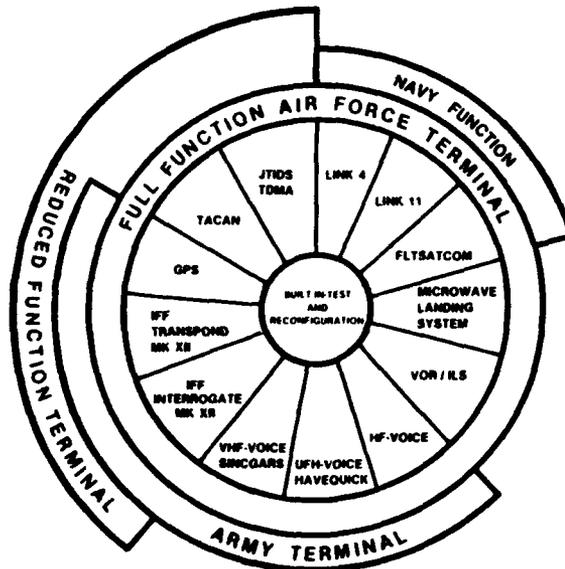
The ICNIA terminal implements a number of CNI functions existing as single, discrete boxes in the current inventory and is based upon a distributed processing architecture. The hardware and software that make up the ICNIA terminal provide the same CNI performance as the discrete function, plus maintain an accurate system health status. Figure 1 shows the CNI functional requirement.

4.2 Functional Architecture

The ICNIA functional block diagram is shown in Figure 2. The partitioning is broken into three groups: the Radio Frequency (RF) group, the Signal Processing (SP) group, and the Data Processing (DP) group. This partitioning maximizes the flexibility for growth and multiple mission configurations. Furthermore, logistics and acquisition costs are minimized through the use of common modules that makeup the three groups.

4.3 Hardware Description

The integration aspect of ICNIA is based upon a set of common hardware modules in which the CNI functions are distributed. The hardware modules can be programmed to represent any combination of the CNI functions. If a hardware module fails, another is dynamically reprogrammed to implement the CNI function. This dynamic reconfiguration guarantees that the most mission critical function with the highest priority is always available.



SYSTEM	FUNCTION
JTIDS	AJ VOICE & DATA
HAVEQUICK	AJ VOICE
GPS	AJ NAVIGATION
SINCGARS	AJ VOICE
LINK 4	DATA
LINK 11	DATA
FLTSATCOM	DATA
TACAN	NAVIGATION
MLS	NAVIGATION
IFF	TRANSPOND INTERROGATE
NARROWBAND	UHF AM VOICE VHF AM / FM VOICE HF SSB VOICE

Figure 1 CNI Functional Requirements

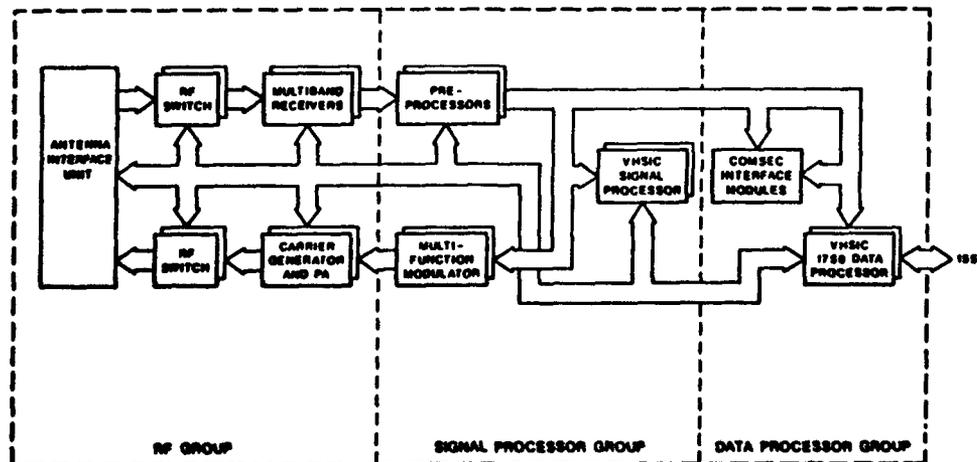


Figure 2 ICNIA Block Diagram

The ICNIA hardware is partitioned into three equipment groups and is based upon the functional partitioning. These groups are the RF group, consisting of receivers and transmitters; the SP group, made up of the high speed real-time preprocessors and signal processors; and the DP group, utilizing 1750A data processors. The equipment groups are interconnected via a network of high speed data buses.

4.3.1 RF Group

The RF group performs the transmission and reception of the CNI signals. This group is divided into L-Band and UHF/VHF frequency ranges. The RF group contains the Antenna Interface Unit (AIU), the receiver switch, the receivers, and the Power Amplifiers (PA). The RF group provides selectable connections between the receivers and the Universal Matched Filters (UMF) in the signal processor group. This capability provides flexible resource assignment and fault tolerance.

4.3.2 The Signal Processing Group

The signal processing group converts the CNI baseband signals into digital data. Each Line Replaceable Unit (LRU) consists of four UMF's, a Multi-Function Modulator (MFM), a Signal Processor (SP), a Time Frequency Reference Unit (TFR), and a Red to Black Transfer Unit (RBX). The UMF performs baseband phase conversion, demodulation, detection, and signal acquisition. It sends the digitized data to the SP for processing. The MFM performs carrier modulation for all transmitted CNI signals. The TFR provides a stable time base to maintain time synchronization. The RBX interfaces the RF group with the SP group.

4.3.3 The Data Processing Group

The data processing group performs all CNI application processing, ICNIA system management, integrated test and maintenance, and aircraft interfacing over the 1553B avionics bus. The Data Processors (DP) use a first generation VHSIC, MIL-STD-1750A architecture, and the software is written in JOVIAL and Ada.

4.4 Software Architecture

ICNIA is unique among avionics systems in several respects. Its reconfiguration capabilities are highly dependent on software. In addition to substantial data processing, the system employs programmable signal processors which perform functions traditionally assigned to dedicated hardware. Processing is distributed among four data processors and two signal processors. Each processing element is capable of performing any task assigned to its respective type. In the event of a processor failure, executable code is reconfigured such that the highest priority functions are assigned to surviving processors, thereby accomplishing a key ICNIA requirement for fault tolerance and graceful degradation.

Like the hardware, the ICNIA operational software is highly modular to support maintainability, testability, and software reuseability. The software is divided into three Computer Program Configuration Items (CPCIs): the data processing CPI, the signal processing CPI and the Data Processing Operating System (DPOS) executive. Like the hardware modules, the software modules, both data processing and signal processing can be reconfigured in response to processor failures and/or changes in mission priorities. Figure 3 shows this software structure.

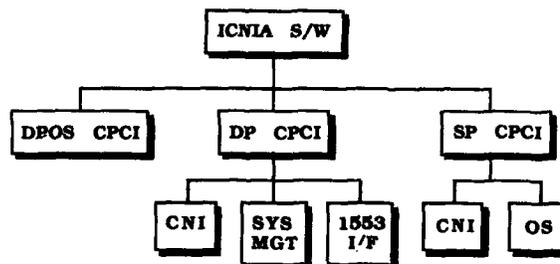


Figure 3 Software Tree

4.4.1 Data Processing Software.

Jovial J73 and Ada programming languages are used and the functional elements of the data processing software include: System Management Software (SMS), Fault Detection and Isolation (FDI) software, software reconfiguration and multiprocessing. In lines of code, the data processing software accounts for more than two-thirds of the operational software of ICNIA.

The systems management software contains the Resource Management (RM) software which controls allocation of hardware resources for mission planning, real-time modification of the mission by the pilot and availability of resources as determined by the integrated test and maintenance function. The SMS software also contains the Terminal Control (TC) software which starts and stops the hardware and software resources and oversees terminal operation.

Software for the Integrated Test and Maintenance (ITM) function contains the machine intelligence which provides the fault detection and isolation to support in-flight resource management and reconfiguration (online test) and non-real-time testing to improve the economics of avionics maintenance (offline test). Real-time fault detection and isolation is required because hardware and software elements are reallocated in real-time in response to mission demand and priorities.

4.4.2 The Signal Processing Software.

The signal processing software performs a variety of high-speed operations on incoming and outgoing signals. A high-speed VHSIC signal processor is used to meet the requirements of the demanding real-time environment. The signal processor CPI is comprised of the executive processing and applications software. The executive and applications software use a macro and micro instruction set developed specifically for the signal processor.

91-15531

91 118 025
 16

This brief introduction of the ICNIA system illustrates its capabilities and its complexities. The multi-function, multi-processor system operating under real-time constraints requires a sophisticated system to detect and isolate faults. An expert system was designed for ICNIA to achieve the goals of a fault detection rate of 98 percent and a fault isolation rate of 90 percent to a single line replaceable module.

5. ICNIAS FAULT TOLERANT DESIGN

The technology for in-flight reconfiguration of the ICNIA system is software-dependent, and is made possible by the Built-In-Test capability and the large number of common modules. BIT allows the system to constantly measure its health status and to reassign or reconfigure modules when a failure is detected. The automatic reconfigurability feature makes the system fault tolerant and increases the mean-time-between-critical-failure (MTBCF).

BIT technology plays a major role in a fault tolerant design. Furthermore, an expert system is required to handle and correctly interpret the data. The expert system within ICNIA is called the Integrated Test and Maintenance (ITM) function and is the "brain" for fault detection and isolation. The ICNIA BIT capability consists of hardware and software sensors which provide system health status information to the ITM function. One of the key sensor elements in the ICNIA design is the Maintenance Node (MN) which is the focal point for performing the board level tests supporting the integrated test and maintenance philosophy of ICNIA.

The ICNIA BIT concept of operation, types of BIT, the BIT architecture, the Maintenance Network, and the Integrated Test and Maintenance function will be developed in the following sections.

5.1 Bit Concept of Operation

Figure 4 illustrates the BIT concept of operation. During normal terminal operation, the hardware and software status indicators are polled by the ITM function to monitor the system integrity (nominally once per second). Upon detection of a system error or processing failure, the BIT fault indicators are accumulated and compared against a threshold. When a processed BIT indicator exceeds a predefined threshold, a reaction is triggered by ITM. This reaction usually consists of performing more tests.

Reactions are dependent on the criticality of the BIT indicator. The two primary actions taken by ITM include replace the asset from the availability pool and/or perform offline tests on the removed asset.

The use of extensive online and offline BIT allows the ICNIA terminal to be supported with two levels of maintenance (organizational and depot) instead of the normal three levels. An indicator is provided on LRMs which identifies a failure so that a maintenance technician knows which LRM to remove and replace.

To alleviate the traditional problem of being unable to reproduce a flight failure after the unit has been returned to the depot, the ICNIA BIT design incorporates a health status memory function which is utilized by the maintenance node on the LRM. The critical data parameters stored in the memory can be analyzed to aid the depot technicians in troubleshooting the failed unit. The health status messages are correlated with time to enable this memory to act as a fault history file.

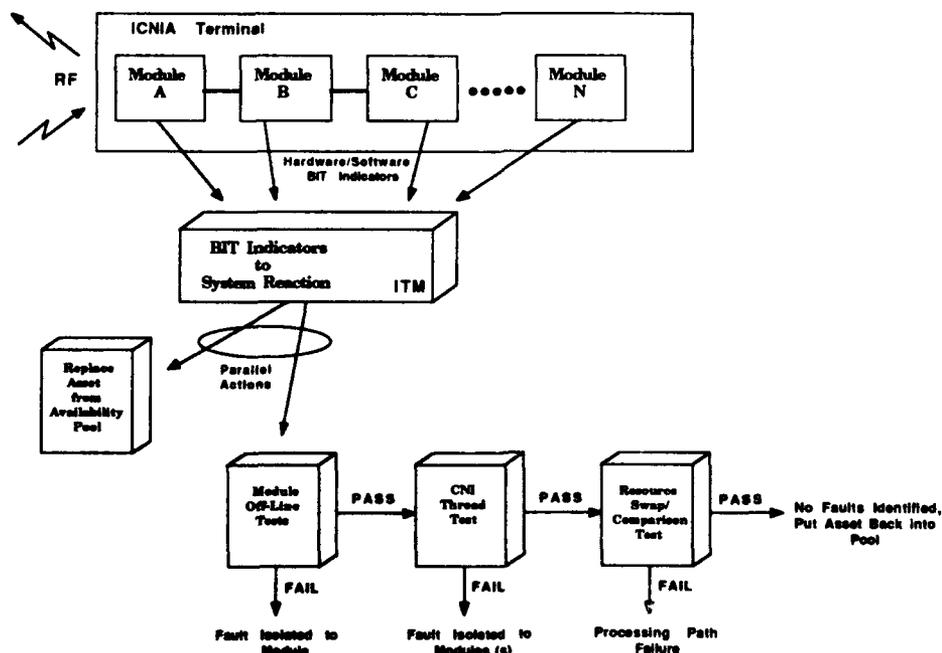


Figure 4 BIT Concept of Operation

5.2 Types of BIT

The four major types of BIT implemented in ICNIA are Online, Offline, Thread, and Standalone BIT.

5.2.1 Online BIT

Online BIT is the process in which system health information is collected while a hardware resource is operational. This process is performed passively leaving the resource available for performing CNI functions and does not affect the state of any of the hardware registers, queues, or memories. Online BIT may explicitly refer to a specific resource, or it may be more general, as in the case of the collection of CNI status data on a given function. Online BIT is provided by both hardware and software sensors.

5.2.2 Offline BIT

Offline BIT is the process where data is collected on a hardware resource while it is unavailable for a mission function. The testing may reset the state of the hardware registers, queues, or memories. Offline BIT is carried out during system startup, and during a mission after a system problem has been identified through the online BIT process.

5.2.3 Thread BIT

Thread BIT is the process of testing two or more hardware modules in cascade to determine system integrity/mission readiness or to perform fault isolation. Thread BIT performs three functions: (1) Support initialization testing of the system to determine mission readiness, (2) Monitor the health of operational CNI threads, (3) Support fault isolation. Thread testing is not a primary means of fault isolation or fault detection. Thread testing is simplistic and helps to support the overall fault tolerant design.

5.2.4 Standalone BIT

Standalone BIT is the process of collecting data while the entire terminal itself is not being used. It occurs only when the terminal is not processing CNI signals. Standalone BIT is used for more rigorous fault isolation testing than can be performed in the other three modes.

5.3 ICNIA Built-In-Test Architecture

Figure 5 provides a top-level view of the ICNIA BIT architecture. The best way to describe this architecture is to divide it into three sections: the central intelligence, the communications, and the sensors.

The central intelligence is the expert system which makes decisions concerning resource health based on BIT indicators and system rules. The central intelligence lies within the integrated test and maintenance (ITM) software function which is part of the Data Processor CPCI.

BIT communications are carried out via the high speed bus, the Bus Interface Units (BIUs), the serial maintenance bus, the 1553 avionics bus, and the software BIT functions. The Maintenance bus is primarily used for system initialization, set scan and interconnect testing and results reporting.

The sensors in the ICNIA BIT architecture are the local hardware and software elements which detect faults or collect local status and performance data. The key sensor element in the ICNIA design is a logical device termed the Maintenance Node (MN). Digital modules in ICNIA contain a Maintenance Node which initialize the module and execute set scan and interconnect test of logic devices. These module tests are invoked under command of a Maintenance Protocol Processor (MPP).

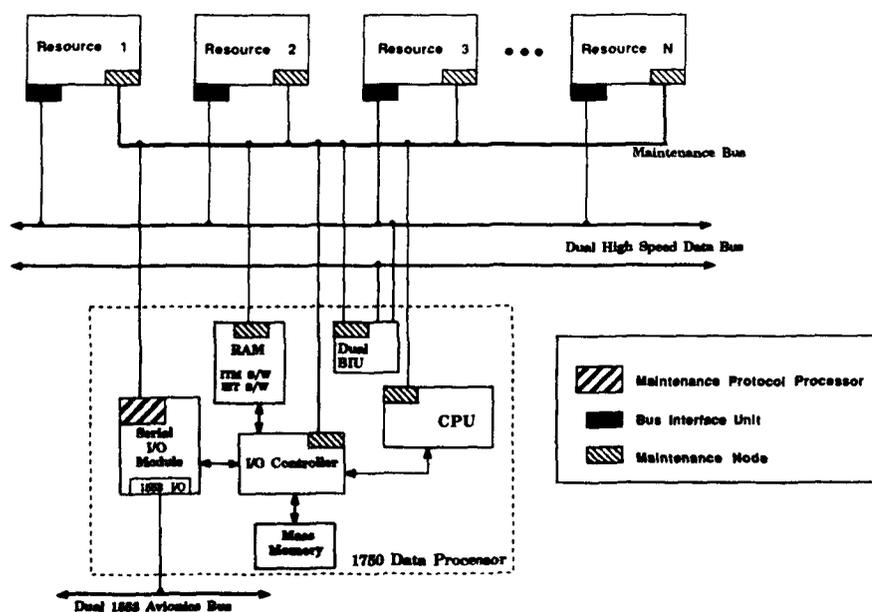


Figure 5 ICNIA BIT Architecture

5.4 Maintenance Network

The Maintenance Network designed into the ICNIA terminal provides a special purpose BIT data collection and communication system. The Maintenance Network architecture is made up of: Maintenance Nodes (MNs), residing on digital modules; a Maintenance Protocol Processor (MPP), which resides on the DP serial I/O module (and acts as the Maintenance Bus controller); and the dual redundant Maintenance Bus which links the Maintenance Nodes with the MPP.

5.4.1 Maintenance Node

The Maintenance Node is the focal point for performing the board level test supporting the integrated test and maintenance philosophy of the ICNIA system. The MN performs the following activities:

It conducts externally controlled offline board/chip level tests and stores the results in EEPROM.

It sends both board and chip level test results to the Maintenance Protocol Processor.

It stores status and parameter values used by devices on a board for normal operations.

It stores online board fault status information and the time of fault occurrences in EEPROM.

The MN (in the offline mode) exercises individual VHSIC, standard cell, and gate array (GA) chips, as well as Small Scale and Medium (SSI/MSI) logic (74xx/54xx logic) configurations confined to one board or module. The method utilized is the "set scan loop" technique. This method is carried out by generating multiple test vectors using an internal Pseudo-Random Sequence Generator (PRSG) and inputting the vectors in order to stimulate a device. It then reads the response, compresses the results (Signature) with a Signature Analyzer (SA), and then compares the generated signature to a previously stored "good" signature for determining operational status of the tested device.

5.4.2 Maintenance Protocol Processor

The Maintenance Protocol Processor serves as the interface between the ICNIA 1750A Data Processor and the Maintenance Nodes distributed throughout the system. The MPP receives test commands from the DP, via the DP's input-output processor (IOP), then formats and transmits the commands to the appropriate Maintenance Nodes. In addition, the MPP serves as the Maintenance Bus controller. In this role, the MPP distributes all test commands (originating from the DP) and is responsible for commanding MNs to transmit over the bus.

6. INTEGRATED TEST AND MAINTENANCE

As mentioned before, the ICNIA "brain" for fault detection and isolation is the ITM function, which

is part of the data processing CPCI. ITM evaluates status and test results in order to detect and isolate faults. One of the key requirements of the ITM function is to filter and smooth BIT sensor information so that fault false alarms can be minimized. The implementation of the ITM BIT fault isolation algorithms takes the form of a rules based system.

ITM is designed to make decisions using a knowledge base which is augmented by correlation of BIT results and other status data generated by the hardware and software. ITM decisions are based on a foundation of basic rules derived from the physical properties of the hardware and the mathematical properties of the CNI algorithms.

6.1 ITM Overview

The ITM design implements an artificial intelligence expert technique known as fact based inferencing. Fact based inferencing makes decisions concerning fault detection and isolation in a comprehensive and accurate manner by deferring decision until all the required facts have been compiled. For example, if a hardware module is suspected of failing, a typical response by ITM is to request further testing so that more comprehensive reports (facts) can be gathered. The driving reason that fact based inferencing was chosen for the ITM design is that this technique accomplishes fault detection and isolation with near 100 percent accuracy. This level of accuracy mandates a design that provides complete fault coverage as opposed to making uncertain decisions quickly.

Since fact based inferencing requires additional testing, ITM's response cannot be provided in real-time. However, detecting faults quickly is important if the system is to remain operational during critical use periods. Because of this, a double inference architecture was developed for ITM which uses a forward chaining inference to identify hardware suspected of having a fault, and a backward chaining inference which establishes credibility of a particular hardware component by gathering additional facts. Forward chaining is fast, but uses extensive memory, while backward chaining is slow and uses moderate amounts of memory. Typically, the actions that backward chaining requests consist of tests which are disruptive to the normal hardware operation, so they must be done offline.

6.2 Software Architectural Overview

The ITM and BIT fault isolation software is organized into two major divisions. The first is the generic Built-In-Test/Fault Isolation (BIT/FIT) Kernel which contains the inference algorithms, and the second is the ITM and BIT Shell. The ITM and BIT Shell Software are designed to have a clear cut division between the generic portions and the system specific portions. The Shell is functionally divided into two parts. The first part is called the Generic Shell software, and the second is the ICNIA Specific Shell. The term generic is used to mean "not specific to a particular hardware set". The design produces a complete fault detection and isolation package capable of being used in any hardware system while minimizing the amount of programming required to develop the system specific portions of code. Figure 6 illustrates the relationship of the interfaces.

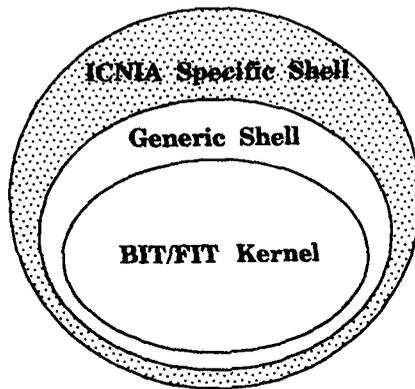


Figure 6 Interface Hierarchy

The Generic Shell Software is divided into three major functional units. The units include the New Facts Generator, the Action Arbitrator, and the Results Processor. The BIT/FIT Kernel performs most of the expert processing in the fault isolation process. It contains a forward and backward inference engine and several data base interfaces to the Generic Shell software. The interfaces include the New Facts List, the Thread Table, the Suspect List, the Action List, and the Results List. Figure 7 illustrates the ITM and BIT architecture.

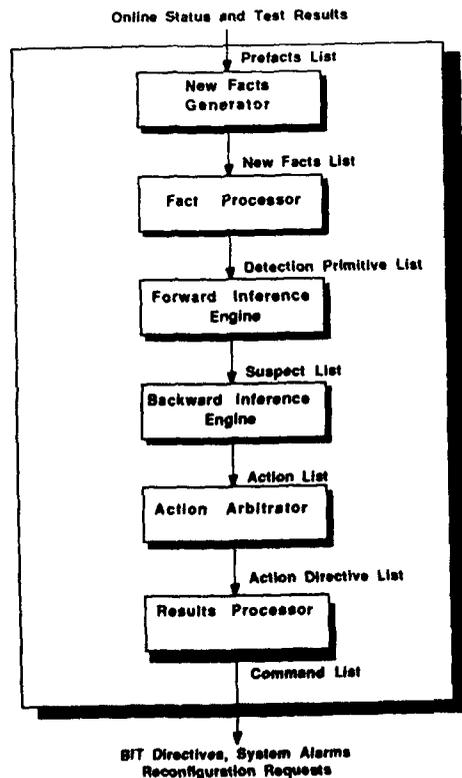


Figure 7 ITM and BIT Architecture

Before the design of the Integrated Test and Maintenance software is presented, an explanation of the rules and the terms associated with the rules is presented.

6.3 Rules Language

An "atomic formula" is the basic element of a rule. It is a predicate function which has a value of true or false. For the fault detection and isolation application, an atomic formula is true when the associated fault has been observed. For example, an atomic formula may be defined for a receiver fault. The predicate function, Receiver_Fault is true (or observed) if a fault is detected for the receiver specified. Receiver_Fault (Receiver_X) is true for Receiver_1 if Receiver_Fault was observed on Receiver_1.

A "rule" is a statement which is used to make inferences based on the state of the system. A rule has the following structure:

If (antecedent) Then (conclusion)

An "antecedent" is composed of disjunctions and/or conjunction of atomic formulas. However, the "conclusion" consists of only conjunctions of atomic formulas (af). An example of a rule is shown below.

Rule: IF [af1 AND NOT af2] OR af3 THEN af4 AND af5

The antecedent of Rule 1 is "[af1 AND NOT af2] OR af3" and the conclusion is "af4 AND af5".

Atomic formulas are divided into three groups: primitives, intermediate rules, and goals.

"Primitives" represent information supplied by the external world. They only appear in the antecedent of a rule. In the fault detection and isolation case, a primitive is a resource status indication whether a fault was detected. An "intermediate rule" is an intermediate conclusions of rules. A "goal" is only part of the conclusions of rules. It is the end result. A "goal" is only part of the conclusions of rules.

The Rules Compiler is responsible for the generation of the rules tables used by the inference engines. The compiler breaks the rules into clauses. A "clause" is a simple rule which has only conjunctions of primitives in its antecedent and conjunctions of goals in its conclusion. The Compiler eliminates all intermediate states or intermediate rules, leaving a clause logic tree with a depth of one (rules transformed in this way are said to be in "conjunctive normal form"). Thus, the compiler does the bulk of the rule tree processing and the engines are able to derive goals directly from "anded" combinations of primitives, eliminating the need for a tree search.

6.4 BIT/FIT Kernel Software Overview

The BIT/FIT Kernel Software is composed of the Forward and Backward Inference Engines, and a Fact Processor that handles the input to the inference engines. The external interfaces with the inference engines are implemented via data structures. These

data structures consist of the Rule File (input), the Active Thread Tables (input), the New Facts List (input), the Action List (output) and the Results List (output). Figure 8 gives a high-level view of the inference algorithm architecture with its external interfaces.

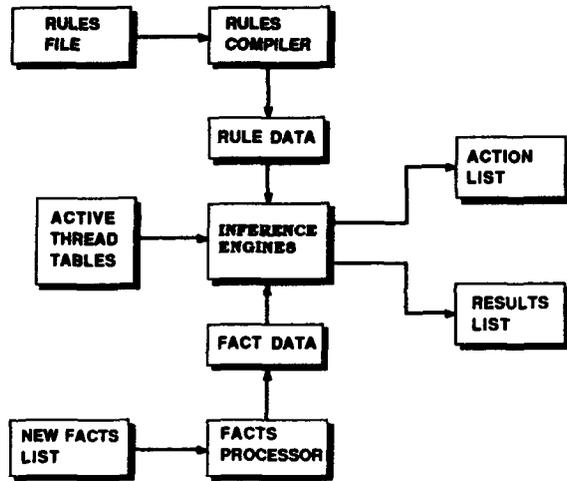


Figure 8 Inference Algorithm Architecture

Figure 8 shows that the internal interfaces can be separated into two types of data structures, rule data and fact data. The rule data is the interface between the Rules Compiler and the inference engines. This data consists of the Detection and Action Clause Tables, and the Detection and Action Clause Mapping Matrices (these tables are a representation of the rules in what is called "normal conjunctive form", that is, a rule transformed into strings of "anded" primitives "ored" together). The fact data provides the interface between the Fact Processor and the inference engines. This interface contains the Action Detection Primitive List, the Detection and Action Primitive Status Buffers and the Backward Engine Start Flag. The interface to a specific application is accomplished with the rules in the Rule File and the Action Thread Tables. These data structures will be discussed later.

The Fact Processor receives the new system facts, determines their types (detection or isolation) and forwards them to the appropriate inference engine. A new system fact may be a changed detection or isolation primitive.

The Forward Chaining Inference Engine processes new detection primitives (online system status). Based on this information and the rule data, it generates a list of system resources suspected to have caused the error.

The Backward Chaining Inference Engine processes the list of suspects generated by the Forward Chaining Engine. It proves each suspect to be functional or faulty based on action results and rule data. Action results are received in isolation primitives that are put by Fault Detection/Fault Isolation (FD/FI) onto the New Facts List. An

isolation primitive represents that state of a diagnostic test or interrogation request by the Backward Engine during a previous cycle. The requested action is intended to determine the health status of a resource.

Some assumptions and limitations are associated with the inference engine architecture. The first assumption is that though statusing may occur frequently, only primitives whose values have changed are reported to the New Fact List. Secondly, the inference algorithms operate on Boolean values only. Thus status obtained in other forms must be transformed to Boolean representations to be used. Another limitation deals with thresholding and aging. In fault detection and isolation applications, the resource status information may be provided in the form of fault indicators or counters. Usually, a fault is processed immediately upon being reported to the diagnostic and test system. However, some faults may require a threshold (i.e. a number of faults detected in a specified length of time) before the fault is processed as an actual error. The assumption was made that thresholding is performed externally to the inference engines. The engines expect status information consisting of fault indicators such that a fault is only indicated if its thresholding requirement has been met. Finally, the assumption is made that active signal threads are reported to the Active Thread Tables as signal paths are defined or reconfigured. Maintaining these tables is essential for the unification process that is necessary when predicate atomic formulas are used in the rules. Unification is the determination of the specific resource to which the predicate atomic formula is referring.

6.5 Fact Processor

The Fact Processor is a preprocessor for the inference engines. It's main function is to update fact data in the system, and notify the appropriate inference engine when inferencing is necessary.

6.5.1 Data Structures

The New Facts List contains all new fact reports. New facts are new primitives that have been collected along with their current state (fault observed/not observed). Note that a primitive consists of a type and a device number.

Primitive 1	Type	Device	State
Primitive 2	Type	Device	State
		:	
		:	

Figure 9 New Facts List

The Detection Primitive List contains a list of the detection primitives to be processed by the Forward Chaining Inference Engine.

Primitive 1	Type	Device
Primitive 2	Type	Device
		:
		:

Figure 10 Detection Primitive List

The Backward Engine Start Flag indicates that there are new isolation primitives to be processed by the Backward Engine. It is set by the Fact Processor when new isolation primitives are received.

The Detection Primitive Status Buffer contains the status (observed, not observed) of all detection primitives.

	Device 1	Device 2	
Primitive Type 1	State	State	...
Primitive Type 2	State	State	...
		⋮	

Figure 11 Detection Primitive Status Buffer

The Action Primitive Status Buffer contains the action indicator for the isolation primitives. The indicators are initially set to "not requested" and "not proven".

Figure 13 shows the data flow of the Fact Processor.

	Device 1	Device 2	
Primitive Type 1	Requested	Requested	...
	Results	Results	
Primitive Type 2	Requested	Requested	...
	Results	Results	
		⋮	

Figure 12 Action Primitive Status Buffer

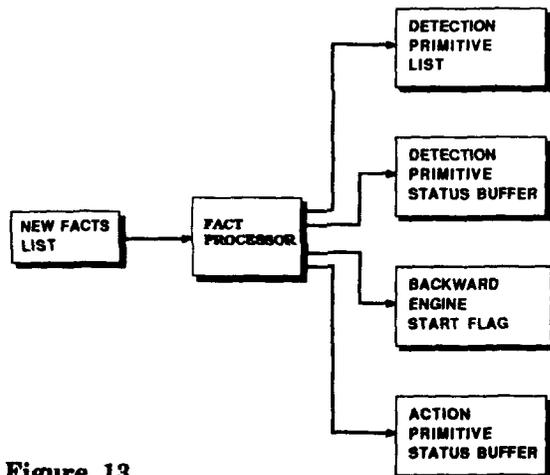


Figure 13

Fact Processor Data Flow Diagram

6.5.2 Processing

When the New Facts List is updated, the Fact Processor begins processing and continues until the New Facts List is empty. The Fact Processor takes each new fact from the New Facts List and determines if it is a detection or isolation primitive. If it is a detection primitive, the Detection Primitive Status Buffer is updated and the primitive is moved to the Detection Primitive List.

If the new fact is an isolation primitive, the Action Primitive Status Buffer is updated and the Backward Engine Start Flag is set. An update to the Action Primitive Status Buffer consists of updating the state field of the isolation primitive.

6.6 Forward Chaining Inference Engine

The Forward Chaining Inference Engine uses rules to generate a list of resources suspect to be bad. Therefore, the conclusion of each rule is a list of one or more resources which are suspected to be faulty. This conclusion is based on the current state of the antecedent's detection primitives in the Detection Primitive Status Buffer. If the specified combination of a rule's detection primitives are true (i.e. the associated fault indicators have been observed), the rule infers that the resource is suspected to be faulty.

6.6.1 Data Structures

The Forward Chaining Engine uses several data structures to control its processing. These data structures are listed below along with brief descriptions.

The Detection Primitive List contains all the new detection primitives. A primitive consists of a primitive type and a device number. The table is illustrated in Figure 9.

The Detection Clause Mapping Matrix contains a mapping from each primitive to the clauses which include it. This mapping is constant for any particular rule set, and is created by the Rules Compiler.

Primitive Type 1	Clause #	Clause #	...
Primitive Type 2	Clause #	Clause #	...
		⋮	

Figure 14 Detection Clause Mapping Matrix

The Detection Clause Table contains the clauses or simple rules which infer conclusions (eg., a resource is suspected) from facts (eg., bad status detected on a receiver). This is a constant table for any particular rule set and is created by the Rules Compiler.

Clause 1	Original Rule #	Susp Type	Prim Type	Prim Type	...
		Negation	Negation	Negation	
Clause 2	Original Rule #	Susp Type	Prim Type	Prim Type	...
		Negation	Negation	Negation	
	⋮	⋮		⋮	

Figure 15 Detection Clause Table

The Suspect List contains a list of all the goals that have been reached, i.e. it contains a List of resources suspected to be faulty, along with information concerning how these resources came to be suspected.

The Detection Primitive Status Buffer contains the resource status information. It contains information for each detection primitive in the system. This table is illustrated in Figure 11.

Suspect 1	Type	Device #	Whiteboard	Susp Thread	Susp Clause
Suspect 2	Type	Device #	Whiteboard	Susp Thread	Susp Clause
			⋮		

Figure 16 Suspect List

The Active Thread Tables contain current signal threads in the system. These tables are shown below.

	Resource Type 1	Resource Type 2	
Active Thread 1	Device #	Device #	...
Active Thread 2	Device #	Device #	...
		⋮	
	Device 1	Device 2	
Resource Type 1	List of Active Threads Device is in	List of Active Threads Device is in	...
Resource Type 2	List of Active Threads Device is in	List of Active Threads Device is in	...
		⋮	

Figure 17 Active Thread Tables

The Backward Engine Start Flag indicates that there are new suspects to be processed by the Backward Engine. It is set by the Forward Engine when new suspects are identified.

The following diagram shows the relationship of these data structures to the Forward Chaining Inference Engine.

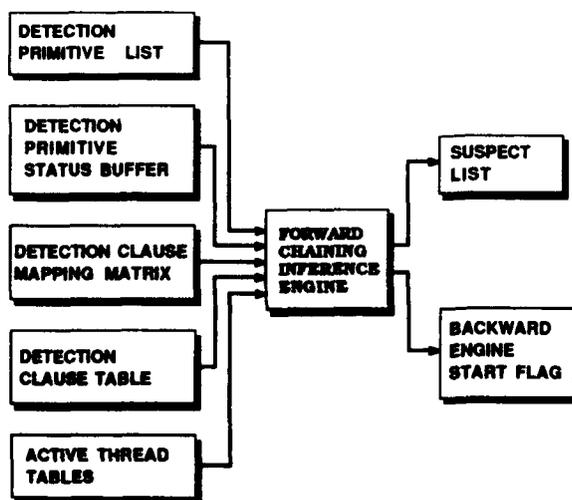


Figure 18

Forward Chaining Inference Engine Data Flow Diagram

6.6.2 Processing

The Forward Chaining Inference Engine is designed to generate a list of resources suspected to be in error based on the detection primitive status information provided. The engine's processing is driven by the Detection Primitive List. The engine selects the first primitive in the list and uses it as an index into the Detection Clause Mapping Matrix. The Detection Clause Mapping Matrix maps each primitive to a set of clauses is "fired" by the Forward Chaining Inference Engine. If all the primitives in the antecedent of a clause are true (taking their possible negation into account), then the conclusion is true when its fault indicator in the Detection Primitive Status Buffer is set to true (observed). A detection primitive may be negated by adding a "not" in the rule directly preceding the primitive. This means that the primitive state is equal to false (not observed) and will cause the primitive to be processed in the rule as if it was true. The engine repeats this process while the Detection Primitive List contains new detection new detection primitives.

6.7 Backward Chaining Inference Engine.

Rules for the Backward Chaining Inference Engine are written to prove a suspected resource is functional (good) or faulty (bad). The resource is assumed to be good. Therefore, the rules state the conditions needed to prove the resource is bad. The antecedent of each rule contains logical combinations of isolation primitives and the conclusion is one or more resources shown to be in error.

Isolation primitives are different from detection primitives in that an isolation primitive's final value is not always known. The isolation primitives value is initially set to "error not proven". To determine an isolation primitive's final value, an action must be requested. An action is a diagnostic test or interrogation designed to determine the status of a resource. When the action is performed and the results are reported, the isolation primitive's value is "PROVEN false" if its final value is false. This approach was developed to distinguish between a default value of "not proven" (and thus false until the test results are received) and a final value of false.

To facilitate processing, each isolation primitive has two action indicators: Action Request and Action Results. The relationship of the values of these action indicators to the value of the primitive is illustrated in the following table.

Action Requested	No	No	No	Yes	Yes	Yes
Action Result	Not Proven	Proven True	Proven False	Not Proven	Proven True	Proven False
Value of Primitive	False	False	False	False	True	False

Figure 19 Relationship of Action Indicators to Isolation Primitive Values

1. Action Requested - indicates whether the action has been requested

2. Action Result - indicates if the action has been processed and the value of the isolation primitive. The value is as follows:

- Proven True - error observed
- Proven False - error not observed
- Not Proven - assume error not observed for the moment

6.7.1 Data Structures

The Backward Chaining Inference Engine uses several data structures to control processing. These data structures are listed below with brief descriptions.

The Suspect List contains a list of the resources suspected to be faulty, along with information about how the resources came to be suspected. This table is shown in Figure 16.

The Backward Engine Start Flag indicates that there are new isolation primitives or new suspects to be processed by the Backward Engine. It is set by the Fact Processor when new isolation primitives are received, and by the Forward Engine when new suspects are identified.

The Action Primitive Status Buffer contains the action indicators for the isolation primitives. The indicators are initially set to "not requested" and "not proven". The Action Primitive Status Buffer is shown in Figure 12.

The Action List contains a list of requested actions. Each entry has a field for the action type and the specific device.

Action 1	Action Type	Device
Action 2	Action Type	Device
	⋮	⋮

Figure 20 Action List

The Results list contains the results of the fault detection and isolation operation, that is, the status of the resources suspected to be faulty.

Result 1	Suspect Type	Device	Result	Convict Clause	Suspect Clause	Suspect Thread
Result 2	Suspect Type	Device	Result	Convict Clause	Suspect Clause	Suspect Thread
			⋮			

Figure 21 Results List

The Action Clause Mapping Matrix contains a mapping from each suspect to the clauses which could prove it to be faulty. This mapping is constant for any particular rule set.

Suspect Type 1	Clause #	Clause #	...
Suspect Type 2	Clause #	Clause #	...
			⋮

Figure 22 Action Clause Mapping Matrix

The Action Clause Table contains the clauses or simple rules which infer conclusions (like resource is faulty) from facts (like set scan test on BIU 1 failed). This is a constant table for any particular rule set.

Clause 1	Original Rule #	Susp Type	Prim Type	Prim Type	...
		Negation	Negation	Negation	
Clause 2	Original Rule #	Susp Type	Prim Type	Prim Type	...
		Negation	Negation	Negation	
	⋮	⋮		⋮	

Figure 23 Action Clause Table

Figure 24 shows the relationship of these data structures to the Backward Chaining Inference Engine.

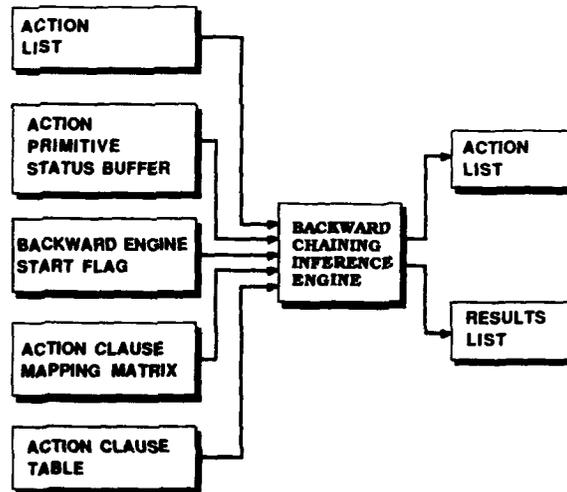


Figure 24 Backward Chaining Inference Engine Data Flow Diagram

6.7.2 Processing

The Backward Chaining Inference Engine begins processing by building a whiteboard for each new suspect on the Suspect List. A whiteboard is used to record the progress in the verification of a suspect's functionality. This is necessary since a conclusion may not always be reached in one inference cycle. A whiteboard is built by indexing with suspect type into the Action Clause Mapping Matrix, generating a list of clauses which could prove the suspect to be faulty. A clause status indicator for each clause is allocated in this area of memory (the whiteboard).

Next, the engine begins processing each clause by attempting to prove the clause is true. To do this, the engine must prove all the clause's primitives are true. To show an isolation primitive is true requires that an action be requested, processed and a true result received. The Action Primitive Status Buffer provides the current status of each primitive. If the required action has not already been requested, then the engine will place that action on the Action List and set the action request indicator. If the action has been requested, but the results have not been received, the engine cannot make any further decisions about that primitive. In either case, processing will move on to the next primitive in the clause. If the action has been requested and processed, the final value of the primitive is known.

If one primitive is proven false (taking it's possible negation into account), the clause is false and processing continues to the next clause for the suspect. When a clause is proven false, the clause indicator on the whiteboard is set to verification complete to prevent repetition of the clause evaluation on later inference cycles. If all of a clause's primitives are true the clause is true and the suspect is proven to be faulty .

When either all clauses are proven false or one clause is proven true, the associated suspect is added to the Results List with the appropriate result (resource good/resource bad). This processing is performed for all suspects on the Suspect List. After all suspects are processed, the whiteboards are deleted for which final results were found. Figure 25 illustrates the requirements for proving a suspect's functionality.

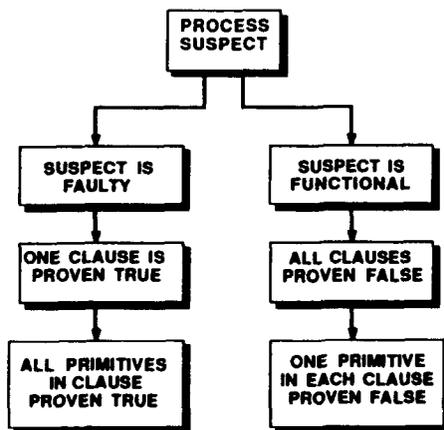


Figure 25 Proving a Suspect Faulty or Functional

6.8 Generic ITM Shell

6.8.1 New Facts Generator

The New Facts Generator is a fact processor responsible for collecting new facts and routing them to the inference engines. A fault filtering scheme using thresholding and aging is performed on the facts used in the fault detection process.

6.8.1.1 Data Structures

The fault filtering algorithm is implemented using the following data structure.

The Threshold Table contains an entry for each detection primitive. The event counter is used to record the number of bad status events, and it is compared to the event threshold. The age counter is used in conjunction with the age period to invoke aging on the fact periodically. The aging flag is used to indicate whether aging is desired during the current processing cycle.

The Prefacts List and the New Facts List are identical in format and are illustrated in Figure 9.

Primitive 1	Event Counter	Event Threshold	Age Counter	Age Period	Age Flag
Primitive 2	Event Counter	Event Threshold	Age Counter	Age Period	Age Flag
			:		

Figure 26 Threshold Table

The following diagram illustrates the data flow through the New Facts Generator.

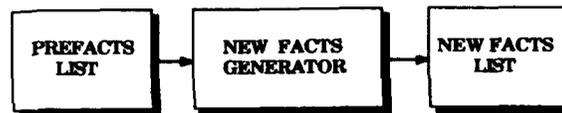


Figure 27 New Facts Generator Data Flow

6.8.1.2 Processing

Execution starts with the initialization of the Threshold Table. The event and aging threshold values for each primitive are supplied by an external initialization task. Once the initialization process is completed, the task enters the main processing loop. The sequence begins by waiting for a timer task cycle start indication. Next, each prefact on the Prefacts List is read in and processed one at a time. The main processing loop finishes with an attempt to age each of the detection facts.

Prefact Processing of isolation facts consists of simply writing the fact to the New Facts List. Prefact processing of detection facts involves accessing the Threshold Table to determine whether a new fact should be generated and sent to the BIT/FIT Kernel. The reception of the prefact is used to trigger a request for aging to be performed at the end of the current processing cycle. If the prefact indicates a good status, no further processing is performed. If the prefact indicates a bad status the event counter is incremented and compared to the event threshold. If the threshold has been exceeded then a new fact is generated and written to the New Facts List. When a new fact is generated, aging of that fact is disabled for that cycle. This creates some hysteresis in the new fact generation process for the purposes of preventing

the event counter from crossing the threshold in both directions during the same cycle.

The aging process is performed for each detection fact which has its aging flag set in the Threshold Table. The flag was set earlier in the cycle to indicate that online status information was received. This scheme prevents aging from being performed on resources which have been proven faulty or are currently under offline test. The aging algorithm initially sets the age counter to the age period, and then decrements it each cycle. When it reaches zero, the event counter is decremented and compared to the event threshold. If the threshold has been crossed, a new fact indicating a good status is generated.

6.8.2 Action Arbitrator

The Action Arbitrator is responsible for scheduling action and managing resource health.

6.8.2.1 Data Structures

The Action Arbitrator uses the following data structures:

The Action Priority Matrix contains an entry for each detection and isolation primitive.

Primitive 1	Priority
Primitive 2	Priority
	⋮

Figure 28 Action Priority Matrix

The Action State Matrix contains an entry for each selection and isolation primitive. Each entry consists of a list of devices and their required states for execution of the action.

Primitive 1	Resource Type	Device #	Device State
	⋮	⋮	⋮
Primitive 2	Resource Type	Device #	Device State
	⋮	⋮	⋮

Figure 29 Action State Matrix

The Resource State Table is used to maintain information needed by the Action Arbitrator to make

Resource 1	State	Maintenance Count	Priority
Resource 2	State	Maintenance Count	Priority
		⋮	

Figure 30 Resource State Table

action scheduling decisions. The state field is used to indicate healthy, faulty, requesting maintenance, or performing maintenance. The maintenance count is used to track how many actions currently require the resource to be in maintenance. The priority field indicates the current priority of an action using the resource.

The Command List is used as an input stream for a variety of messages from other tasks. Possible command identifications include resource failed, resource healthy, maintenance response, online status on, or online status off.

Command ID
Operation Type
Device #

Figure 31 Command List Entry

The Action Queue is identical in format to the Action List given in Figure 20.

The Action Response List is identical in format to the New Facts List given in Figure 9.

The following diagram illustrates the data flow through the Action Arbitrator.

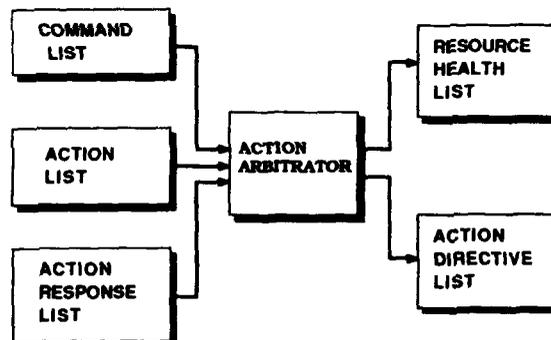


Figure 32 Action Arbitrator Data Flow Diagram

6.8.2.2 Processing

Execution starts with the initialization of the Action Priority Matrix and the Action State Matrix. The priority and a list of devices and their required states during test execution are supplied for each action or primitive by an external initialization task. Once the initialization process has completed, the task enters the main processing loop. The sequence begins by waiting for a timer task start of cycle indication. Next, each action is read off the Action List and inserted into the Action Queue. Once the Action List has been emptied, the Command List is processed one message at a time until it is empty. The Command List may contain such things as maintenance request responses, resource health reports, or online status enable/disable commands.

After the Command List has been processed, each fact or action response is read off the Action Response List and processed one at a time. This list is used in the implementation of a closed loop action/response protocol. The main processing loop finishes with the processing of the Action Queue and the initiation of online status.

Actions pulled off the Action List are inserted into the Action Queue unless there is already an identical queued action. The priority for a particular action is obtained from the Action Priority Matrix, and it is used to insert an action into the sorted queue after all of the actions of equal or higher priority. As an action is inserted, maintenance requests are written to the Resources Health List for all devices listed in the Action State Matrix as requiring maintenance. The Resource State Table state and the maintenance count are also updated to reflect the maintenance request.

Action responses are used to signal the completion of the executing action. When a response is received the corresponding list of devices is fetched from the Action State Matrix. If a device is in maintenance mode, the maintenance count is decremented and a maintenance complete message is written to the Resource Health List. The priority field in the Resource State Table is also cleared to indicate that the resource is no longer busy from the execution of the action.

The Command List is used to collect all of the inputs to the Action Arbitrator that do not fall into the category of inferencing actions and action responses. These inputs include maintenance request responses from the Resource Health Table Manager, online status enable or disable command from the Resource Health Table Manager, and resource health reports from the Results Processor.

A maintenance response from the Resource Health Table Manager is received on the Command List for each maintenance request issued. Upon reception on their response the Resource State Table state is changed from requesting maintenance to performing maintenance.

Resource health reports read off the Command List are used to update the Resource State Table and are sent on to the Resource Health Table Manager. If the maintenance count is zero for the resource specified in a report of healthy, then the state is set to healthy. If it is not zero, then the health report is written back to the Command List to be processed next cycle. If the report indicated failure, the state in the Resource State Table is set accordingly.

The Action Queue is processed from beginning to end by attempting to schedule each action. A particular action is scheduled only if the list of devices and states given in the Action Priority Matrix exactly matches the current state of the devices found in the Resource State Table Priority fields for those devices indicate that they are not under test. Actions are scheduled by writing them to the Action Directive List.

Online status is initiated simply by attempting to schedule all of the actions corresponding to detection primitives. This method prevents online status requests from being generated for offline or failed devices.

6.8.3 Results Processor

The Results Processor is responsible for the post-processing necessary after the inference engines have either convicted or acquitted a suspected resource.

6.8.3.1 Data Structures

The Results Processor uses one primary data base to store the special processing instructions required for each resource received from the Results List.

The Results Processing Table contains information on what special processing must be performed for each convicted or acquitted resource. The opcoded field indicates either health reporting, error logging, action initiation, thread passing, suspect generation, good fact generation, or bad fact generation. Each instruction includes a pass and fail tag on which it's invocation is based, such that all instructions marked for pass are executed when the result indicated suspect acquittal, and all instructions marked for fail are executed when the result indicates suspect conviction.

Resource 1	Op Code	Obj Type	Device #	Pass	Fail
			⋮		
Resource 2	Op Code	Obj Type	Device #	Pass	Fail
			⋮		

Figure 33 Results Processing Table

The following diagram illustrates the data flow through the Results Processor.

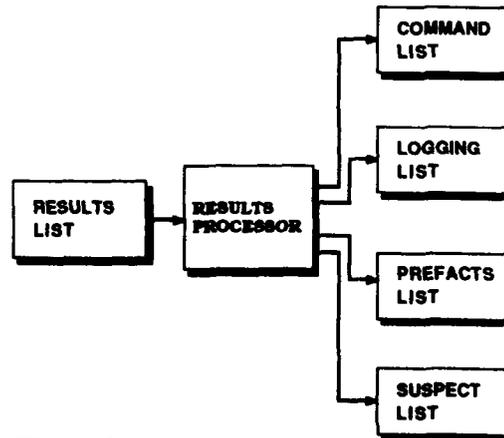


Figure 34 Results Processor Data Flow Diagram

6.8.3.2 Processing

Execution starts with the initialization of the Results Processing Table. The special processing instructions for each resource are supplied by an external initialization task. Once the initialization process has completed, the task enters the main processing loop. The sequence begins by waiting for input from the results list. The results list is an interface to the BIT/FIT Kernel used by the Backward Inference Engine to report the convictions and acquittals of suspects. Each result is read in and processed one at a time.

Based on the resource designated in the result, the proper group of special processing instructions are extracted from the Results Processing Table. The six types of special processing instructions include reporting resource health to the Action Arbitrator Command List, writing a result to the logging list, writing a special action to the Action List, generating a backward inference engine thread to be used with the next suspect, writing a suspect to the suspect list, and writing a fact to the Prefacts List. If the result indicates pass, then an instruction is executed and it is tagged as pass. If the result indicates failure then an instruction is executed and it is tagged for failure. The type and device number fields define a resource when used with reporting, logging, thread passing, or suspecting, and they define a primitive when used with action initiation or fact generation.

7. CONCLUSION

The Expert System within ICNIA utilizes a double inference architecture for the Integrated Test and Maintenance software function. Forward chaining inference is used to identify hardware suspects of having a fault, and backward chaining inference is used to establish credibility of a particular hardware component by gathering additional facts. This allows for a comprehensive fault detection and isolation system to isolate 95 percent of all known faults to a single line replaceable unit and 90 percent of all known faults to two or more line replaceable units.

Discussion

1. J. Bart, United States

How many rules are being implemented in the ICNIA KBS for fault tolerance?

Author:

The total number of rules in the Expert System is approximately 300. Two-thirds of the rules (or 200) are for the Backward Chaining Inference Engine. More rules (in general) are required for Backward chaining because Backward chaining (suspect substantiation) is more difficult to convict suspects. That is more rules are needed to convict suspects than to identify them. One-third of the rules (or 100), are for the forward chaining inference engine.

2. Prof. A.N. Ince, Turkey

As a result of integrating communication navigation and identification functions what has been achieved in terms of savings in weight, power and volume?

Author:

Weight savings from 360 lbs to 250 lbs, power reduction of 50%, and size has reduced from 5.0 cu. ft to 4 cu. ft based on the integration of these functions within the system.



COMPONENT PART NOTICE

THIS PAPER IS A COMPONENT PART OF THE FOLLOWING COMPILATION REPORT:

(TITLE): Conference Proceedings of machine Intelligence
for Aerospace Electronic Systems Held in Lisbon, Portugal on
13-16 May 1991 (L'Intelligence Artificielle dans les Systemes
Electroniques Aero spatiaux)

(SOURCE): Advisory Group for
Aeronautical Research and Development, Neuilly-sur-Seine
(France)

TO ORDER THE COMPLETE COMPILATION REPORT USE AD-A242025

THE COMPONENT PART IS PROVIDED HERE TO ALLOW USERS ACCESS TO INDIVIDUALLY AUTHORED SECTIONS OF PROCEEDINGS, ANNALS, SYMPOSIA, ETC. HOWEVER, THE COMPONENT SHOULD BE CONSIDERED WITHIN THE CONTEXT OF THE OVERALL COMPILATION REPORT AND NOT AS A STAND-ALONE TECHNICAL REPORT.

THE FOLLOWING COMPONENT PART NUMBERS COMPRISE THE COMPILATION REPORT:

AD#: _____ TITLE: _____

AD-P006349

NOV 13 1991

A-1. 21

THIS DOCUMENT IS UNCLASSIFIED
DATE 10/13/98 BY 1043/SP/STP/STP

AD-P006 349

A DEVELOPMENT-MEMORY APPROACH
FOR ENHANCING AVIONICS SOFTWARE LOGISTICS

Marc J. Pitarys
United States Air Force
Wright Laboratory (WL/AAAF-3)
WPAFB, OH 45433
United States of America

91-15528

Advanced avionics systems will execute Ada software on multiple parallel computers. The size and complexity of the avionics software will be massive. Of particular importance is how the software will be supported once it is delivered to the Air Force. Software support is that part of the avionics software life cycle that deals with the correction of deficiencies and the addition of enhancements to the avionics software. Much of the time spent by personnel is information searching and inferencing. These tasks are made more complicated when information that was useful to the developer is not included in the software documentation that is delivered with the weapon system.

This paper will first cover the challenges facing avionics software support personnel. It will address an advanced software support concept called Development-Memory (DM). This research is being accomplished by the Air Force Wright Laboratory's Avionics Directorate with contracted support from Hughes Aircraft Company's Radar Systems Group. DM relies on Machine-Intelligence technology in its goal to enhance avionics software support.

DM technology refers to the processes, tools, and techniques required for achieving the goal of development knowledge transfer. The DM will contain the motivation and rationale that influenced the development of the software. The DM and its support software would serve a large-scale software development/support effort the same way human memory serves a single individual. Machine-Intelligence (MI) technology is needed for creating and utilizing the DM; for example, the use of expert systems, semantic networks, knowledge architectures, intelligent information organization and retrieval tools. These Machine-Intelligence (MI) aspects will be discussed in this paper within the context of the development and use of a DM based software support system.

BACKGROUND ON AVIONICS SOFTWARE LOGISTICS

Software support (logistics) activities occur after the weapon system is fielded. The majority of the Air Force's software support activities occur at Air Logistic Centers (ALCs). Typically each ALC is responsible for supporting several weapon systems of the same type. Weapon system software is supported by using an Integration Support Environment (ISE). The ISE includes the needed hardware and software to analyze, change, integrate, test, verify, and validate the weapon system's software. For example, the ISE contains the embedded avionics computers, simulation models and computers, high-speed networks, test software and

instrumentation, workstations, software tools and documentation, an out-the-window display, and the avionics controls and displays.

The software that is supported executes in real-time on embedded computers. The software source code is large and complex since it implements avionics applications such as fire control, radar, stores management, navigation, and electronic warfare. In addition, the software contains a vast amount of documentation that was created during its development. The Air Force is now developing its avionics software using the Ada Programming Language and DOD-STD-2167A (Defense System Software Development). DOD-STD-2167A describes over twenty documentation items that can be delivered from a software development effort. For a typical avionics software project these documents consist of hundreds of pages, contain multiple volumes, and reference other documents. It is this documentation and source listings that the support personnel use to maintain and upgrade the avionics software.

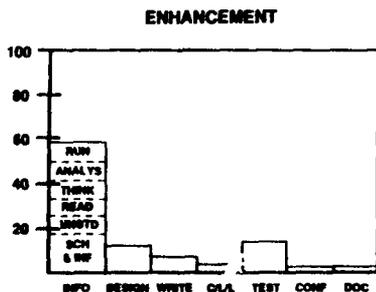
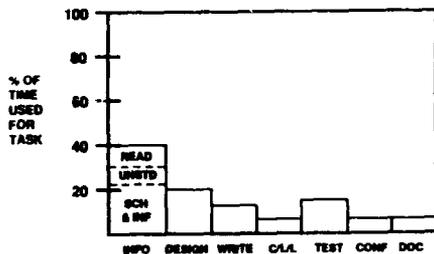
The majority of the software life-cycle costs occur during the post deployment support phase of the weapon system. With this in mind the Air Force established a comprehensive program to enhance the PDSS process, called the Embedded Computer Resources Support Improvement Program (ESIP). WL/AAAF is responsible for the research and development (R&D) aspects of ESIP. One contract R&D effort, called Modular Embedded Computer Software (MECS), is with Hughes Aircraft Company. The objective of the MECS effort is to develop new technology and techniques to support the maintenance of distributed fault-tolerant, multiprocessor avionics software.

The software that is supported executes in real-time on embedded computers. The software source code is large and complex since it implements avionics applications such as fire control, radar, stores management, navigation, and electronic warfare. In addition, the software contains a vast amount of documentation that was created during its development. The Air Force is now developing its avionics software using the Ada Programming Language and DOD-STD-2167A (Defense System Software Development). DOD-STD-2167A describes over twenty documentation items that can be delivered from a software development effort. For a typical avionics software project these documents consist of hundreds of pages, contain multiple volumes, and reference other documents. It is this documentation and source listings that the support personnel use to maintain and upgrade the avionics software.

91 11 20 020

The majority of the software life-cycle costs occur during the post deployment support phase of the weapon system. With this in mind the Air Force established a comprehensive program to enhance the PDSS process, called the Embedded Computer Resources Support Improvement Program (ESIP). WL/AAAF is responsible for the research and development (R&D) aspects of ESIP. One contract R&D effort, called Modular Embedded Computer Software (MECS), is with Hughes Aircraft Company. The objective of the MECS effort is to develop new technology and techniques to support the maintenance of distributed fault-tolerant, multiprocessor avionics software.

Before developing any new technology or techniques, it was decided that the support process needed to be studied. It was important to determine what aspects of software logistics were the most difficult and consumed the greatest amount of time. With this knowledge, the contract resources could be concentrated on areas that would have the most effect. First, a discrete model of the software maintenance process was constructed. "Once the process model was defined, a survey of 20 Hughes Radar Systems Group's tactical software engineering staff who had substantial radar software maintenance experience was performed (Figure 1). The results of the survey indicated that the "information gathering" consumed the majority of the time. Furthermore, "information searching and inferencing" was the largest part of information gathering (approximately 1/5th, see Figure 2). In addition, the survey respondents indicated a need for higher quality information, more comprehensive documentation, and improvement in the means of locating desired information within the documentation." (1)



DEFICIENCY

FIGURE 1

COMBINED SURVEY AVERAGES ASSUMING 70% ENHANCEMENT AND 30% DEFICIENCY UPGRADING ASSIGNMENTS.

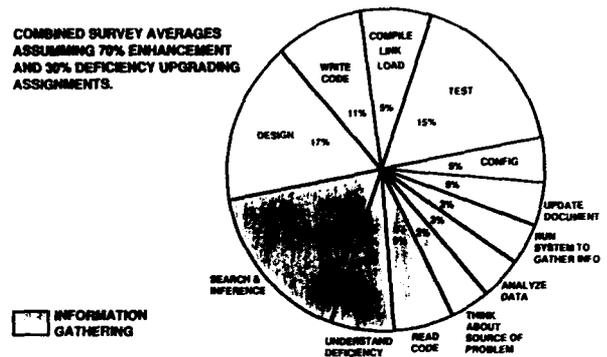


Figure 2

What the MECS survey revealed was the existence of a development-maintenance knowledge gap. Even though developers transfer the system software and documentation to the maintainers, their understanding of the system is not transferred. Knowledge that the developers find useful is not being recorded. With the results of the survey, research began on ways to reduce the development-maintenance knowledge gap. One outcome of the MECS research was a concept of an Integrated Software Development Product (ISDP).

INTEGRATED SOFTWARE DEVELOPMENT PRODUCT

The developer's understanding of the system is based on many documents and other pieces of information. For example, technical papers, briefings, and internal documents help the developer understand the system and its software. Often this information is never included in the delivered documentation. When the system's software requires support, maintenance personnel must retrace the steps the developer took to understanding the system, in addition to recreating information to gain the required knowledge. Basically, an Integrated Software Development Product (ISDP) will allow the maintainer to view the system through the eyes of the original developers.

This ISDP will contain information normally lost to software support personnel. It provides a way of integrating the information contained within the deliverable documents and information used by developers in creating the system into a single entity. Building an ISDP is more than just collecting documents and linking them together. It requires processes, tools, and techniques to collect knowledge from software developers and transfer it to software maintainers. An approach to realizing an ISDP is through the development of a component called the Development Memory (DM).

DEVELOPMENT MEMORY

An ISDP can be realized with the support of a DM technology. The DM involves an information architecture, tools for collecting and transferring knowledge, some of which rely on machine intelligence techniques, and the human operator. The end result of using the DM is that the human appears more intelligent when working. Work is ongoing on the DM. "The work is focusing on making the DM capable of the following:

- 1) Recording and storing, for later reuse, information pertinent to the development process.
 - 2) Allowing for quick and effortless access and comprehension of information needed to support the system."
- (2)

Recording software developer understanding is not as simple as having the developer write one more conventional document. Human memory readily allows for the relating of many different pieces of information in different ways. Conventional linear text documents do not. Therefore one would think that hypermedia documents should be used to record the developers understanding. However, this is not enough. The reason is human memory automatically places and maintains information it receives while hypermedia documents do not provide this service.

The DM approach is motivated by software support concerns. However, developers stand to benefit from its use as well, since they will have the capability to access needed information. The following sections will discuss the various pieces of the DM and highlight the MI aspects.

DEVELOPMENT MEMORY KNOWLEDGE ARCHITECTURE

The DM is intended to be constructed as a result of the software development process. "The DM knowledge architecture reflects the hierarchical structure of development roles. This structure roughly parallels the organization of the development. A development role represents a part played by one or more developers in meeting a specified development objective.

(Figure 3) These objectives can be high-level like "radar tactical software development", but more often would be low-level, such as "Range While Search Mode Scan Function Development"." (3)

If the information in a DM is to be useful, it must have continuity and a coherent arrangement so that both developers and maintainers can understand it. Therefore it is advantageous to have the DM parallel the actual development in its organization. Large scale avionics software efforts are organized in a hierarchical fashion. In the development hierarchy, each development-role has a superior and most likely one or more subordinate development roles. Each has specific objectives and resources assigned to it through its superior. (Figure 4)

The organization needs to address development-role relationships that are established across hierarchical boundaries. For example, the development of avionics software is divided into overlapping efforts of functional requirements preparation and software design, code, and unit test. Each major effort is a distinct branch of the development hierarchy. Development roles are defined as preparing requirements or software design, code, and unit test. "This is ideal from a management perspective. However, for an effort to succeed the branches need to directly interact with one another. Lateral interactions can exist within the structure, however through procedure rather than structure (Figure 5)." (4) These lateral assignments result in information being transferred across hierarchical boundaries.

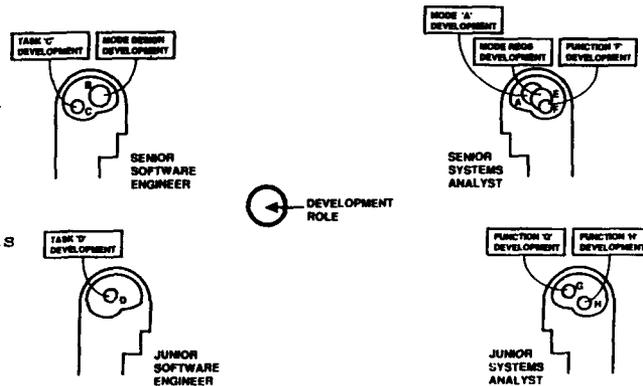


FIGURE 3

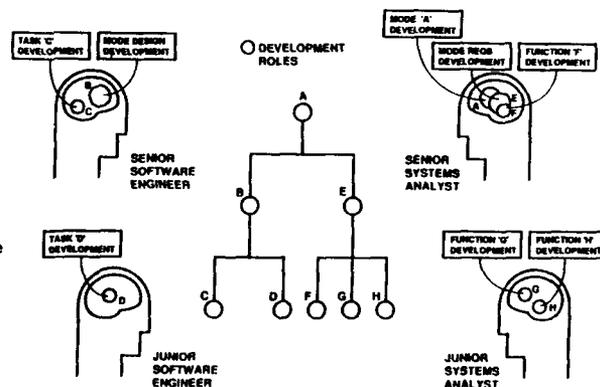


Figure 4

Hierarchical Organization of Development Roles

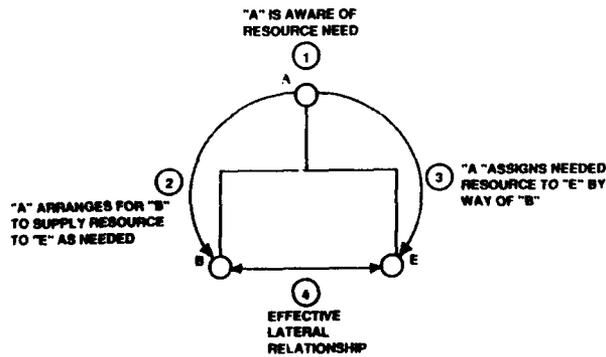


Figure 5

Hierarchical Protocol of Lateral Interactions

020 5113 026 16

With the DM knowledge structure mirroring the organization of the development effort the next area that will be addressed is the acquisition of the knowledge. This can be thought of as memorization. Again, this knowledge will be organized in the fashion described above. The knowledge acquisition will help generate the development memory.

The question now arises as to how will the knowledge be acquired from the developers. The DM needs the knowledge from the developers and from the development roles they perform.

"Developers often experiment with ideas using paper or white boards. This can be referred to as the development space. When the ideas are finalized or a solution obtained it is not recorded in a permanent or organized fashion. To alleviate this problem the DM based system allows for experimentation within the development space. When ideas are concluded, the developer's work with his solution is already on the system. The work can easily be edited for appearance purposes and transferred to the DM's knowledge base." (5)

"One can also gain critical information for understanding the design of the software by examining the relationship between development roles in the organizational hierarchy. This information is contained within the assignation space. Assignation spaces describe the agreed upon objectives of the development role, lists the resources and other items entitled by the development role, and describes what the obligations of the development role are with respect to other development roles. Every development role within the organizational hierarchy shares an assignation space with each of its subordinates. Lateral assignation spaces also exist across hierarchical boundaries. This provides information concerning any shared responsibilities for fulfilling design specifications.

The designer's interpretations of the specifications are held in translation spaces. Translation spaces are not shared. They are meant to be an area where individual responsibilities and objectives of a development role may be addressed. The DM system has access to all of the translations systems.

The developer ultimately creates deliverable end products such as source code and documentation. Each development role has an end-product space where the contributions to the final product are stored." (6)

Knowledge of the system is contained within the development, assignation, translation, and end-product spaces. However, knowledge is available as a result of the interrelationship of the spaces. The spaces in which the developer operates provides the motivational information for the particular action he performs such as programming a certain type of algorithm.

Human memory retrieves information in response to the stimuli of associated information, conditions, and situations. Our memories, with the help of cues from the outside world, remind us to do things. Similarly, developers will need to be reminded frequently to store significant information if their full understanding is to be captured. The organization of many interrelated items of information is a formidable task. Naturally, the developer would find this undesirable since it would interfere with his or her work. Automating this would help considerably.

What is needed, in addition to a hypermedia styled knowledge base, is an active component that can perform the following services:

1) Assistance in the timely placement and organization of information pertinent to the development process.

2) Assistance in the timely retrieval and presentation of development memory information.

The MECS effort devised the concept of having a Development Secretary (DS) provide these services. A description of the DS follows in the next section.

DEVELOPMENT SECRETARY (DS)

Recall that a DS was conceived to provide the active services of the DM described in the previous section. The DS acts like a human secretary does within an organization. The DS assists in the collection and organization of information without having to know all the technical details. The DS uses the knowledge architecture described above much like a human secretary uses a file plan. The DS makes use of elements from two MI paradigms to acquire and transfer knowledge. The two paradigms are expert systems and semantic networks.

To provide the services required by the DM the DS needs a certain degree of intelligence; in this case MI. A semantic network is needed so that the DS can understand what the user is doing. A semantic network represents language. The network is made up of nodes that are interconnected by arrows. The nodes can represent nouns while the arrows are verbs that describe some sort of action. The development of a semantic network for the DS allows it to understand what the user is doing for the development effort. In addition, it allows the DS to ask what the user is doing if the action is incompatible with the existing semantic network.

The following is a description of the DS's semantic network. The nodes of this network can be thought of as subjects, objects or both. The interconnections (arrows) between the nodes are called relationships. The arrow's source node is referred to as a subject. Subjects can contain one or more states. The arrow's destination node is called an object. Subject and objects are described by nouns or modified nouns while relationships are described by verbs. To understand the semantic network an example is described below.

Consider a problem being worked on by a developer. We will call this Problem X. Problem X initiates several ideas. "Initiates" is the relationship. The ideas which are the objects are Idea A, Idea B, and Idea C. In this example, Problem X can be in the "solved" or "unsolved" state. The developer begins to work on Idea A. The DS is aware of this and associates all work performed by the developer to Idea A. If the user moves on to work on Idea Y, the system is able to ask why Idea A was abandoned, by using a simple rule contained in the rule base. When the user answers the question, the DS can store it in the development memory. If the user moves on and begins work on another Problem, then the DS can ask if it should change the state of the problem to solved from unsolved. In addition it will store the work performed on the idea in the knowledge base for later retrieval by maintenance personnel. In the course of working on Idea A, the user may identify the idea possesses a cost advantage. Here the network can be expanded by the user telling the DS that Idea A possess an advantage in cost. Now Idea A is the subject, "possesses" is the relationship, and "advantage" is the object. (Figure 6)

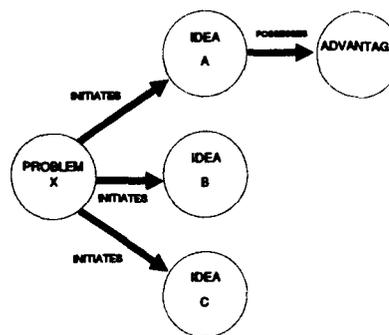


Figure 6 Semantic Network

The Air Force has received a concept demonstration of the DM. It executes on an Apple Computer Company Macintosh Computer. The entire demonstration was built using a hypertext based tool called SuperCard by Silicon Beach Software Inc. The demonstration illustrates many of the concepts discussed in this paper. At this moment a prototype of the DS is being built by Hughes Aircraft Company for the United States Air Force. The prototype is being built with SuperCard.

In the example described, the DS has been programmed to know what relationships apply to the node called Problem. It knows what states the subject Problem contains. Furthermore, it is aware of what relationships pertain to the object called idea. This knowledge is expressed as rules and is stored in the rule base. In conventional expert systems, a mechanism exists for invoking a rule when an event occurs. When a rule is invoked a condition is tested to see what action is to be performed. One action causes another rule to be invoked. A situation can result where a chain of rules is invoked. For the DS, a large chain of rule invocations is not envisioned. However the rule base may contain a large number of rules since rules may be added as more subjects are encountered in the development effort.

Without the rule base the semantic network is not useful. Rules will be added to the DS by the developer or by another individual we will refer to as the "System Manager". The concept calls for the System Manager to create the initial semantic network and adds the applicable rules. More rules can be added as the project evolves by both the System Manager or by the individual developer to handle the needs of his particular development role.

The DS helps in the collection and organization of the knowledge that will be useful in supporting the weapon system. When the DM is transferred to the support organization, support personnel will get a view of the system from the developer's perspective. The ideas used, adopted, and discarded by the developer for solving problems that may arise again during PDSS, will be available for review as a result of the DM. Rational for the development of the implementation of particular algorithms will be accessible. By having this information, the time spent by support personnel for information search and inference will be reduced.

SUMMARY

Avionics software support is a complex activity. A majority of the time spent by personnel in maintaining weapon system software involves information search and reference. The Air Force's Wright Laboratory is performing research and development to reduce the amount of time spent upgrading weapon system software. The MECS effort is focussing on this area by working on a concept called DM. The overall goal of DM and its DS based tool is to transfer knowledge from the developers to the support personnel in the form of an ISDP. The DM involves the collection and application of knowledge and therefore relies on a knowledge architecture and MI paradigm that uses expert system concepts and semantic networks. A concept demonstration has been completed, while a prototype of the DS is now under development.

ACKNOWLEDGEMENTS

The Author wishes to acknowledge the assistance given by the Hughes MECS Principal Investigator, Richard Falcioni, in the preparation of this paper.

REFERENCES

1. Falcioni, R.A, "Modular Embedded Computer (MECS) Fourth Quarter Review, 28 SEP 89, pp 9-10
2. Pitarys, M.J. and Falcioni, R.A., "Applying Development-Memory Technology to Avionics Software Support", IEEE/AIAA/NASA 9th Digital Avionics Systems Conference Proceedings, 15-18 OCT 89, p 94
3. Ibid. 2.
4. Ibid. 1 pp 41-42
5. Ibid. 2
6. Ibid. 2

COMPONENT PART NOTICE

THIS PAPER IS A COMPONENT PART OF THE FOLLOWING COMPILATION REPORT:

(TITLE): Conference Proceedings of machine Intelligence
for Aerospace Electronic Systems Held in Lisbon, Portugal on
13-16 May 1991 (L'Intelligence Artificielle dans les Systemes
Electroniques Aero spatiaux)

(SOURCE): Advisory Group for
Aeronautical Research and Development, Neuilly-sur-Seine
(France)

TO ORDER THE COMPLETE COMPILATION REPORT USE AD-M242025

THE COMPONENT PART IS PROVIDED HERE TO ALLOW USERS ACCESS TO INDIVIDUALLY AUTHORED SECTIONS OF PROCEEDINGS, ANNALS, SYMPOSIA, ETC. HOWEVER, THE COMPONENT SHOULD BE CONSIDERED WITHIN THE CONTEXT OF THE OVERALL COMPILATION REPORT AND NOT AS A STAND-ALONE TECHNICAL REPORT.

THE FOLLOWING COMPONENT PART NUMBERS COMPRISE THE COMPILATION REPORT:

ADM: _____ TITLE: _____

AD-P006350

DTIC
S ELECTE D
NOV 13 1991

Approved For	
NTIS COPY	✓
DATE	
BY	
DATE	
FILE	
A-1	21

This document has been approved for public release and sale; its distribution is unlimited.

AD-P006 350



C³I GRAPHICAL APPLICATIONS

Earl C. LaBatt Jr.
Rome Laboratory
RL/COAA, Griffiss AFB, NY 13441-5700
United States

1. SUMMARY

Past research efforts in the field of Command, Control, Communications, and Intelligence (C³I), have concentrated on the application and refinement of advanced Artificial Intelligence (AI) concepts. However, the man-machine interface to these advanced applications has experienced little improvement and does not adequately reflect some of the AI concepts embedded within these systems.

The development of more advanced computing techniques requires a more sophisticated interface to the system. These enhancements cannot be adequately conceived by a user using traditional display techniques.

This paper reveals the underlying AI concepts that help facilitate the transfer of information between acquired C³I data and an operator. Many of these concepts are discussed in regard to their application in current Tactical Air Force research projects. In addition, the paper discusses the limitations of previous program interfaces and the current advances in graphical interface techniques.

2. PREFACE

As the volume of data that an operator must analyze increases, it has become apparent that better techniques are required for presenting this influx in a more manageable manner. AI concepts are capable of alleviating much of this problem through intelligent manipulations of the data and by providing recommendations for decision-making. The current need among Tactical Air Force operators is for improved visualization of the spatial relationships, object interdependencies, decision reasoning, modification effects, and other alternatives that are incorporated into an AI solution.

The graphical enhancements provided in this report were derived from the Graphic Systems Representation (GSR) study. This study was initiated in June 1989 and concluded in March 1991 by General Electric Company Research and Development Center for Rome Laboratory. The scope of the project was to develop novel graphical techniques for performing Force Level Planning. The results of this study are intended to be applied to future planning systems. Therefore, some of the graphical techniques that are discussed in this paper are not currently implemented, but are presented with the notion that they may be incorporated into future systems.

This paper focuses on two decision aid systems that incorporate several AI concepts, the Tactical Expert Mission

PLAnner (TEMPLAR) and the Identification of Command and Control Operations Nodes (ICON). TEMPLAR is a frame-based system that aids mission planners in developing an Air Tasking Order (ATO). ICON aids intelligence analysts in tracking enemy movement and is based on a blackboard architecture that provides input into the planning process. This paper will not attempt to give a complete description of the AI encompassed in ICON or TEMPLAR; rather, the paper focuses on the specific implementation of a few of their AI concepts to show how representative graphical output can enhance the systems.

The following two sections describe TEMPLAR and ICON, respectively. Within each section, a background of the system is presented, followed by a discussion of a specific domain problem in each of the ensuing main subsections. Each domain problem is discussed using the following format. An introductory section describes one of the domain problems incorporated into the system. Then, an AI description subsection discusses the AI concept used to solve the problem, including its ability to screen and process data for recommended decisions within the application. Finally, a graphical enhancement subsection describes some of the limitations of the implemented display techniques and proposes some graphical enhancements to better represent the AI derivations.

3. TEMPLAR BACKGROUND

TEMPLAR was initiated in September 1985 and completed in October 1988 in conjunction with TRW. It is designed to assist the Ninth Air Force in ATO planning, in areas where essentially no comprehensive system of automation existed. A more thorough discussion of TEMPLAR's history is contained in [5].

TEMPLAR demonstrates how AI concepts can be applied to the Tactical Air Force problem of Air Tasking Order (ATO) generation. An ATO is generated by Tactical Air Force planners to fight tomorrow's war. The plan they produce is basically a schedule that describes the force level use of aircraft and their associated resources, against a list of nominated targets over the next 24 hour period.

TEMPLAR was developed using Knowledge Craft^R, an AI language tool-kit, which was developed at Carnegie-Mellon University. Included within Knowledge Craft^R are problem solving techniques and a frame-based knowledge representation language with procedural attachments and inheritance. Knowledge Craft^R assists knowledge engineers and AI system builders by dramatically reducing the required effort in building

knowledge-based systems. For a complete description of the capabilities of Knowledge Craft^R, reference [3] and [4].

A requirement of TEMPLAR was to support **mixed initiative** planning. Mixed initiative planning is the ability to interleave planning by the user and the system (allowing either the system or the user to plan any part of the ATO). TEMPLAR implements mixed initiative planning by filling out forms. This implementation allows the user to fill in part of the form without affecting the planning process. Every field on a form in TEMPLAR is tagged to denote if it is user-owned or autoplanner-owned. Each field maps to exactly one slot in one frame within Knowledge Craft^R.

TEMPLAR contains four distinct autopanning modules. Each of these modules generates mission lines (a line of data describing the aircraft configuration for a mission). The package planner generates mission lines on the Target Planning Worksheet (one of many implemented forms). Of the four modules, only package planning will be discussed in the following sections.

3.1 Package Planning

Package planning is the process of determining the aircraft, the time-over-target, and the ordnance (a package), that will be assigned to a given target. A package number is assigned to each package for easy identification. The ability to derive an efficient and satisfying plan, that encompasses many targets, is a long and tedious process. Since an infinite amount of resources are not available, cooperation and coordination among available resources become key elements of the planning process.

The planner is constrained in several areas of the planning process. An obvious constraint to the planner is the number, position, and abilities of the aircraft he has available to complete the plan. The planner is also constrained by a prioritized

list of targets, with approximate time-over-target times associated with each target. Given these constraints, along with weather and other unaccountable logistical problems, the planning process becomes increasingly complex.

Figure 1 shows a TEMPLAR worksheet containing a list of Offensive Counter Air (OCA) targets to be planned (OCA is one of many mission types that may be planned with the worksheet). This worksheet functions as a summary of the OCA packages to be planned. Note that the targets are associated with a package number and an estimated time over target on this worksheet. The following paragraphs, summarize a sequence of events that could be used by a planner, in planning these packages.

A planner may organize the development of attack packages into phases. The following phases are applied to each of the Target Planning Worksheets the planner is required to plan: estimate the aircraft usage assuming that each worksheet receives its first weaponeering choice for its first DMPI (Desired Mean Point of Impact), determine the set of weaponeering/tanker combinations, calculate the order for examining viable weaponeering options, and determine threats to the aircraft en route to the nominated target.

Upon completion of these initial steps, each worksheet is addressed individually to determine threat dependencies. The following steps are applied to each worksheet: attack missions are planned; a laser designator is added; if required, force protection is added; if required, Wild Weasel missions are added; electronic combat missions are scheduled, if required and Wild Weasel support is not possible; if fuel is available on a tanker and sorties are available, a reconnaissance mission is scheduled. A more complete description of the package planning process can be found in [5]. TEMPLAR's implementation of the autoplanner function is discussed in [6].

PLANNING: TARGET PLANNING WORKSHEETS - OCA			
00001		00001	
00002		00002	
00003		00003	
00004		00004	
00005		00005	
00006		00006	
00007		00007	
00008		00008	
00009		00009	
00010		00010	
00011		00011	
00012		00012	
00013		00013	
00014		00014	
00015		00015	
00016		00016	
00017		00017	
00018		00018	
00019		00019	
00020		00020	
00021		00021	
00022		00022	
00023		00023	
00024		00024	
00025		00025	
00026		00026	
00027		00027	
00028		00028	
00029		00029	
00030		00030	
00031		00031	
00032		00032	
00033		00033	
00034		00034	
00035		00035	
00036		00036	
00037		00037	
00038		00038	
00039		00039	
00040		00040	
00041		00041	
00042		00042	
00043		00043	
00044		00044	
00045		00045	
00046		00046	
00047		00047	
00048		00048	
00049		00049	
00050		00050	
00051		00051	
00052		00052	
00053		00053	
00054		00054	
00055		00055	
00056		00056	
00057		00057	
00058		00058	
00059		00059	
00060		00060	
00061		00061	
00062		00062	
00063		00063	
00064		00064	
00065		00065	
00066		00066	
00067		00067	
00068		00068	
00069		00069	
00070		00070	
00071		00071	
00072		00072	
00073		00073	
00074		00074	
00075		00075	
00076		00076	
00077		00077	
00078		00078	
00079		00079	
00080		00080	
00081		00081	
00082		00082	
00083		00083	
00084		00084	
00085		00085	
00086		00086	
00087		00087	
00088		00088	
00089		00089	
00090		00090	
00091		00091	
00092		00092	
00093		00093	
00094		00094	
00095		00095	
00096		00096	
00097		00097	
00098		00098	
00099		00099	
00100		00100	

Master Menu

Set Up

G & A

Disp Specs

Prev ATOs

TOEC

ToG - BAI

ToG - NT

Prev Form

Form Menu

FUEL

WINDOWS

AUTOPLAN

INITIANG

Figure 1: Sample OCA Target Planning Worksheet (menu options on the right)


```

{{ deliver-house-materials
  IS-A: construction-activity delivery
  SUB-ACTIVITY-OF: build-house
  EXPECTED-DELIVERY-DATE: "20 May 1991"
  SHIPPED: t
  DELIVERED: nil
  DESCRIPTION: "Dispatch materials to the construction site" }}

```

Figure 3: Deliver-house-materials Schema

```

{{ build-house
  IS-A: construction-activity house-development
  SUB-ACTIVITY-OF: build-housing-development
  CONSTRUCTION-MANAGER: Joe_Wilson
  DURATION: (18 mo)
  DESCRIPTION: "Construct house number 14" }}

```

Figure 4: Build-house Schema

3.1.2 TOT Sliders

It is essential for the planner to keep track of many interrelated goals and constraints while building an ATO. However, these relationships (described by related schema) are blended between several forms in TEMPLAR. To the user, these relationships become vague and not implicit; consequently, while attempting to formulate a large plan, he may become focused on an individual section of the plan and lose the "big" picture. This imbalance could result in a less efficient plan. This problem stems from the user being unable to view, on a single screen, a package's interrelated sorties (individual aircraft) and temporal constraints. A graphical technique, known as TOT (Time-Over-Target) sliders, can provide this visualization capability, allowing the planner to visualize the geographic, temporal, and resource properties of each sortie.

TOT sliders display a sortie's mission window, time-over-target, flight times, and turn-around time (see figure 5). The mission window is shown as a blue horizontal line, defining an axis upon which the rest of the components of a TOT slider can move. Time-over-target is displayed as a green rectangular box, centered over the mission window. The outgoing and returning flight times are represented as dashed blue horizontal lines attached, respectively, to the top and bottom of the time-over-target box. The turn-around time is a dashed green line, that is attached to the end of the return flight time. The schedule slippage sensitivity (e.g. a long flight, penetration of a high threat zone, scheduled near the

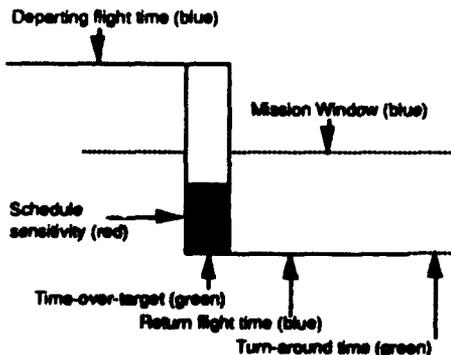


Figure 5: TOT Slider

end of the mission window) of the sortie is shown by the amount of red in the time-over-target box.

The mission window is taken as a fixed entity, upon which the remaining parts of the TOT slider (fixed in relation to each other) may slide. As the planned time-over-target nears the end of the mission window, the red within the box increases. The complete TOT slider (except the mission window line) will become red if the time-over-target box is pushed off the mission window line through user interaction ("clicking and dragging" on the TOT slider).

In multiple sortie packages, constraints may exist between two or more sorties, because a temporal relationship exists between them (see figure 6). TOT sliders may be defined as predecessors or successors of each other. Should a sortie start before its predecessor sortie has finished, a red constraint violation line is drawn between the two time-over-target boxes. This line disappears when the TOT sliders are moved to avoid the constraint.

In figure 6, the top sortie is a predecessor to the bottom sortie. To remove the displayed constraint, the sorties must be moved apart by the distance of the constraint line. Once the sorties are moved sufficiently apart, the constraint line is erased.

The final constraint that is associated with the TOT slider is a limited resource constraint (see figure 7). The bottom of the display shows a simple plot of the resources available to the planner. The plot reflects the amount of resources left for allocation at any one time. As the TOT sliders are moved around, the plot is updated to show the changes in resource levels. The plot is drawn in black when there is a positive amount of resources available, and in red when the resource becomes over committed. Reference [1] contains a more detailed description of mission scheduling with TOT sliders.

TOT sliders are extremely flexible in the visualization of time, resource, and spatially-constrained problems. Many interrelated sorties and packages may be displayed with several resource plots, to give the planner the "big" picture. The ability to visualize a plan can greatly enhance the planner's capability to produce

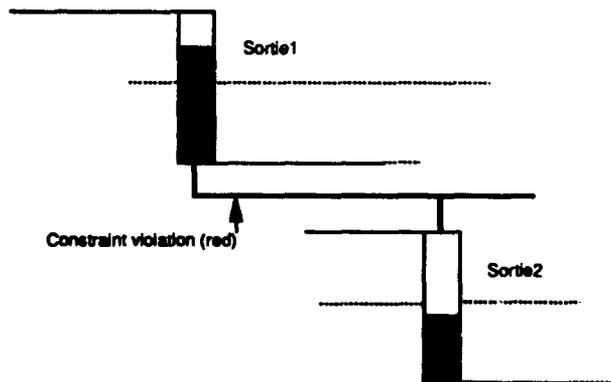


Figure 6: Constrained TOT Sliders

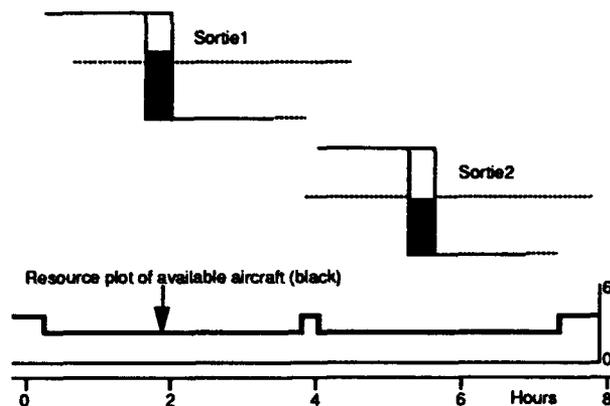


Figure 7: Resource Allocation by Two TOT Sliders

a successful plan.

3.2 Resource Allocation

Resource allocation is an integral part of the package planning process, because it determines what resource will be used to satisfy a need. This process is much like a bookkeeping process - resources are recorded and tracked according to their availability.

Resource allocation occurs at many levels of the planning process. Tanker tracks, weapons, sorties, and time are among some of the resources that must be allocated. As these resources become constrained, the ability to develop a comprehensive package becomes more complex. Some of the difficulty in seeing a complete picture is visible between figure 1, containing the list of all the OCA targets, and figure 2, representing part of the planning for a target.

In the process of developing an ATO, resource allocation can be viewed as a complex network of checks and balances. Affecting one resource in a part of the planning cycle could have a dramatic effect on another part of the plan. Therefore, resource allocation is not a simple process, but rather a series of small checks throughout the planning cycle.

3.2.1 Constrained Schemata and Demons

Demon information on a slot is another method for the slot-control schema. A demon is a function that is associated with a particular slot in a schema. When the slot is updated (beyond a given set of conditions), the demon is executed. Therefore, demons provide a reactive action, executed under specific circumstances in the systems knowledge.

Within TEMPLAR, demons perform quality control functions during the development of an ATO. Demons are primarily used in resource tracking, calculating mission length, and unit scheduling. The allocation of aircraft to missions is also maintained by demons in TEMPLAR. Demons are responsible for controlling the internal records of the allocations and notifying the user if a particular subunit becomes over tasked. Reference [5] for a more complete listing of the automatic checking incorporated into TEMPLAR.

Demons within Knowledge Craft^R provide procedural attachment by being associated with Lisp functions. If a slot contains a demon, access to that slot causes the demon to "fire" (execute) and recalculate the value of each activity associated with it. Multiple demons may be associated with one slot to aid in maintaining internal consistency of the Knowledge Base. Demons are defined by describing the conditions on which it will "fire," the effect it will have on the value, if it will "fire" before or after the last command, and the action it will take upon "firing." Reference [4] for more information on defining demons.

Within TEMPLAR, demons are used to control the updating of related fields. They are assigned to monitor a field, wait until it is updated, then calculate and update other related fields. The demons actions within TEMPLAR are not shown explicitly to the user, however, the actions described in the allocation of aircraft (above) could be displayed as described in the following section.

3.2.2 Resource Allocation Networks

Resource allocation networks focus on the allocation of aircraft to targets (see figure 8). The network of resource and target icons provide a convenient visualization of the allocation of resources to targets, and alternative uses for each resource. Using this visualization technique, the user becomes more aware of his constraints and options.

On the network, time flows from the left to the right. A green octagonal icon and a stack of boxes to its right represents the resource, with a label for the resource type, location, and unused quantity. Targets are represented in red boxes, with two meters on either side and a stack of boxes on the far left. Each box, within the stack of boxes, may be tagged with a green dot, to indicate a type of aircraft that may be used against the target. The orange meter to the left of the target shows the kill probability and the meter to the right of the target shows the value of the target.

Targets and resources are connected by a blue line between a box in each of the icon's stack of boxes. The box that is connected on the resource side contains the number of resources that are allocated to

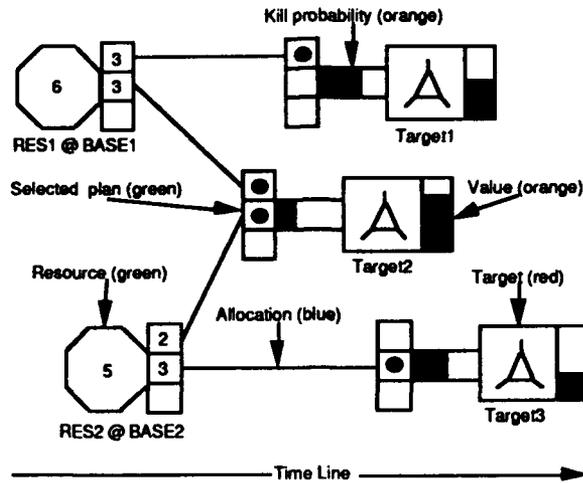


Figure 8: Resource Allocation Network

that target. Since time flows from the left to the right, the horizontal distance of the connecting line indicates the duration of the mission. A resource that may be reused after a mission can be shown to the far right of the target icon by a box with the number of resources in it. This box may then be treated as a normal resource box. An excellent description on the uses of the resource allocation networks may be found in [1].

Figure 8 shows a sample allocation of resources from two bases being used against three targets. By the positioning of the three targets, it is clear that Target2 will be the first target attacked, followed by Target1 and Target3, respectively. The connecting lines show that three resources are designated to Target1 and Target2 from the first base, and two resources to Target2 and three resources to Target3 from the second base. The meters on each side of the target icons show the kill probability and the value of each target.

The natural language capabilities of TEMPLAR tell the planner if a resource is available using numbers and text. However, numbers are difficult to work with, and text from a natural language interface (e.g. large, small, no, ok) may be ambiguous. Risk and reward meters may be associated with every connecting line within the resource allocation network (see figure 9). These meters can display the risk of losing the aircraft, the reward of using an aircraft, and aircraft range limitations. These meters may also reflect the effect of the current/predicted weather for the mission.

Resource allocation networks allow the visualization of time, resource, and spatially constrained problems (e.g. target based). These networks also utilize bar charts for easy comparison and sensitivity checks. These graphical portrayals allow the user to understand the resource

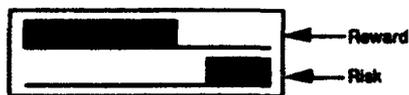


Figure 9: Risk Reward Meter

allocation situation at a glance and visualize the risks and rewards associated with many feasible resource allocation alternatives. This type of synergism is not possible with a purely text-based interface that requires the user to read all of the details.

4. ICON BACKGROUND

The ICON decision aid was initiated in July 1987 and completed in December 1989 in conjunction with PAR Government Systems. ICON is a field-demonstrable advanced-development prototype. ICON aids the intelligence analysts of the Tactical Air Control Center (TACC) Combat Intelligence Division (CID) in the development and maintenance of the Command, Control, and Communications Order of Battle (C³OB). The ultimate goal of ICON is to provide assistance that will enable more timely and accurate inputs to battle planning, and result in a more responsive ATO. Reference [2] presents a more complete history of ICON.

ICON was developed using an object-oriented approach for the design and implementation of the software. The knowledge representation provides a natural and flexible way to model knowledge about enemy units and C² structure. The system relies on a blackboard architecture, an object-oriented knowledge representation, and probabilistic rule-based inference mechanisms. Figure 10 shows the software architecture implemented in ICON.

The blackboard architecture provides the means for bringing diverse problem-solving techniques to bear on the problem. The blackboard contributes to the ability of the system to handle the piecemeal stream of data that must be analyzed. The data required for analysis is taken from Entity Data Records (EDRs), which are intelligence reports from the field. A probabilistic rule-based inference mechanism provides a way of handling the uncertainty associated with data that may be sparse and of varying degrees of confidence.

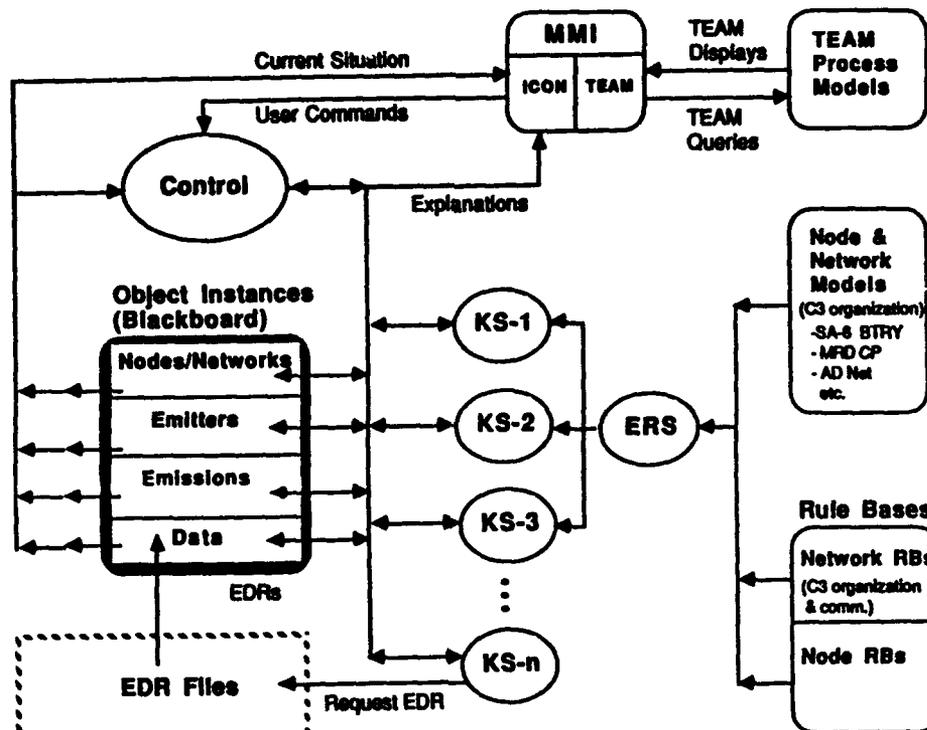


Figure 10: ICON Software Architecture

4.1 Determining Threats from EDR Messages

Analysis of the incoming EDR messages is part of the Command, Control, and Communications Countermeasures analysis function within the TACC. These analyzers receive intelligence information and are tasked with determining the positions of the threats. Much of the information that is received by the analysts is vague and seemingly unrelated. The more deceitful an enemy is and the fewer capabilities you have for collecting intelligence, the more vague and irrelevant the information will appear. Analysts are also overwhelmed at times with bits and pieces of information that cannot be correlated and cross-referenced in a timely manner.

The increased use of mobile threats creates an even more complex situation for the analyst. As the threats move, the analyst must be able to determine if two different reports on the same type of signal are actually different threats, or the same threat that is moving. Many threats are capable of broadcasting in several different frequencies. This ability requires the analysts to retain the knowledge of the frequencies that different types of threats are capable of broadcasting.

EDR messages provide an immense amount of information. However, valuable information is often hidden among several EDR messages, or not readily apparent. The ability to plan an effective ATO may result in a planner's capability to decipher the information contained within the EDR messages. The following section describes the blackboard architecture that is used to capture and correlate all of the data contained in the EDR messages.

4.1.1 Blackboard

The ability to determine a threat from EDR messages requires a system that is capable of handling fragments of data. A blackboard system is implemented to allow for the ability to generate incremental solutions. The system incorporates, in a central data structure, a model of problem solving as defined by the problem domain. A group of independent experts, called "knowledge sources" (reference section 4.2.1), are allowed to manipulate this domain. Through incremental manipulations of the domain, a solution to a problem is generated. The blackboard is shown as part of the software architecture (labeled **Object Instances**) in figure 10.

The design of the blackboard is based on an event-driven class hierarchy, where an event is a recorded change to the blackboard. These classes are used to define the blackboard's system functionality. The Event List classes contain and coordinate the processing of the event class instances. Event processing is tasked with finding knowledge source objects which are "triggered" by an event. For each "triggered" knowledge source that is discovered, the blackboard is asked to add an activity object (created from the activity class that describes "triggered" knowledge sources) to itself.

The ICON blackboard provides a representation of the identified and hypothesized enemy units and their C² relationships. In other words, at any given moment the information encapsulated on the blackboard represents a partial solution to the C² node identification problem. The

blackboard also tracks evidential relationships between pieces of information (derived from the analysis of the EDR data) on various levels. Reference [2] for more information on the ICON blackboard system and the BBTHING framework used to develop the ICON blackboard.

The stages in the process of analyzing EDR data are represented in the ICON blackboard by four levels: Data, Emissions, Emitters, and Nodes/Networks. EDR records are added to the system at the lowest level, the Data level. Emissions are identified on the Emissions level, being derived from multiple or single data records on the Data level. Emissions are signals that have been detected over a period of time at a specific location. Equipment that causes the emissions can be identified on the Emitters level. The Emitters level represents individual pieces of equipment, which were derived from a series of emissions over a period of up to a few hours. The highest level in the blackboard contains C² Nodes, that are identified by groups of emitters. C² network organization is indicated by identifying C² relationships from a pattern of emissions from particular emitters. These networks are represented on the top level as links between objects.

Objects represent data and enemy emissions, equipment, and units. Each object belongs to a class, and each class of objects belongs to only one level of the ICON blackboard. At the lowest level, the data objects represent the EDR records (input into the ICON system). The data is propagated through the remaining levels of the blackboard by the "firing" of knowledge sources. The knowledge sources independently analyze the data, deriving updated information for the blackboard. For more information on the knowledge sources reference section 4.2.1.

4.1.2 Situation Maps

As data is propagated through the blackboard, the user is unaware of its existence until it reaches the top level. Due to the vagueness of some types of EDR messages, it may take long periods of time before enough information is gathered to make a positive identification. Giving the analyst information on this vague knowledge could help them in determining possible future threats.

A situation map portrays data in a three-dimensional surface that encompasses geographic and temporal variations. Situation maps allow incomplete data to be plotted on a map background (see figure 11). All of the data represented in the blackboard has a degree of uncertainty associated with its accuracy; for example,

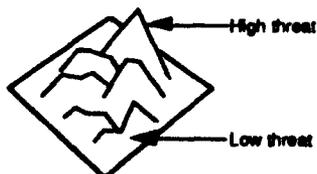


Figure 11: Threat Situation Map

the data contains a latitude/longitude reference point (some degree of unknown). Using the X and Y values of the latitude/longitude area, and a Z value of certainty, the data may be plotted as a three-dimensional surface. Varying color patterns may be added to depict other relevant attributes. This type of map would be continuously evolving as new data is received and updates are made to the blackboard system.

Figure 11 shows a threat situation over part of a grid. Note that no terrain is mapped onto this map. The situation map reflects the threat intensity throughout the grid block by the height of the "hills." The left side shows a small ridge of "hills," indicating low level threat reporting. The right side shows a larger ridge of "hills," indicating a higher level threat. "Valleys" are signs of threats not being detected or not existing at a particular latitude/longitude. Currently, no icons have appeared on the map. This implies that the sum of the reporting are currently not conclusive enough to be labeled.

The situation map represents the current information held within the blackboard. When messages reach the top level of the blackboard they become identified, and the situation map is replaced by the target's icon. Likewise, if data becomes too old and is discarded from the blackboard, that part of a situation map would be removed. These animated situation maps would contain the most current unidentified intelligence data.

Situation maps provide an analyst with insight to probable enemy activity. From these maps, the analyst can watch future activity develop. This ability can aid the analyst in determining trends in enemy activity. Situation maps may also be applied in other areas where the interaction between multiple related values need to be visualized. Reference [1] for a complete explanation on the uses of situation maps and their application to Force Level Planning.

4.2 Information about Identified Targets

Currently the CID Intelligence Analysts have little automation to help them in analyzing EDR files. EDR files arrive at the post where they are read and interpreted by the analysts. ICON provides an automated way to process the EDR files, allowing the analysts to spend more time analyzing the data.

Once ICON identifies a target, the analysts require a natural and explicit method for analyzing the data. ICON provides these methods by plotting the targets on a map and allowing the user access to the rationale (EDR messages) used to identify the target.

The following sections discuss how ICON's use of backward chaining, icons, and spreadsheets enable the user to retrieve information about the identified targets. This part of ICON captures the essence of incorporating graphics and text.

4.2.1 Backward Chaining

ICON uses a blackboard (reference 4.1.1) to aid in the identification of targets through

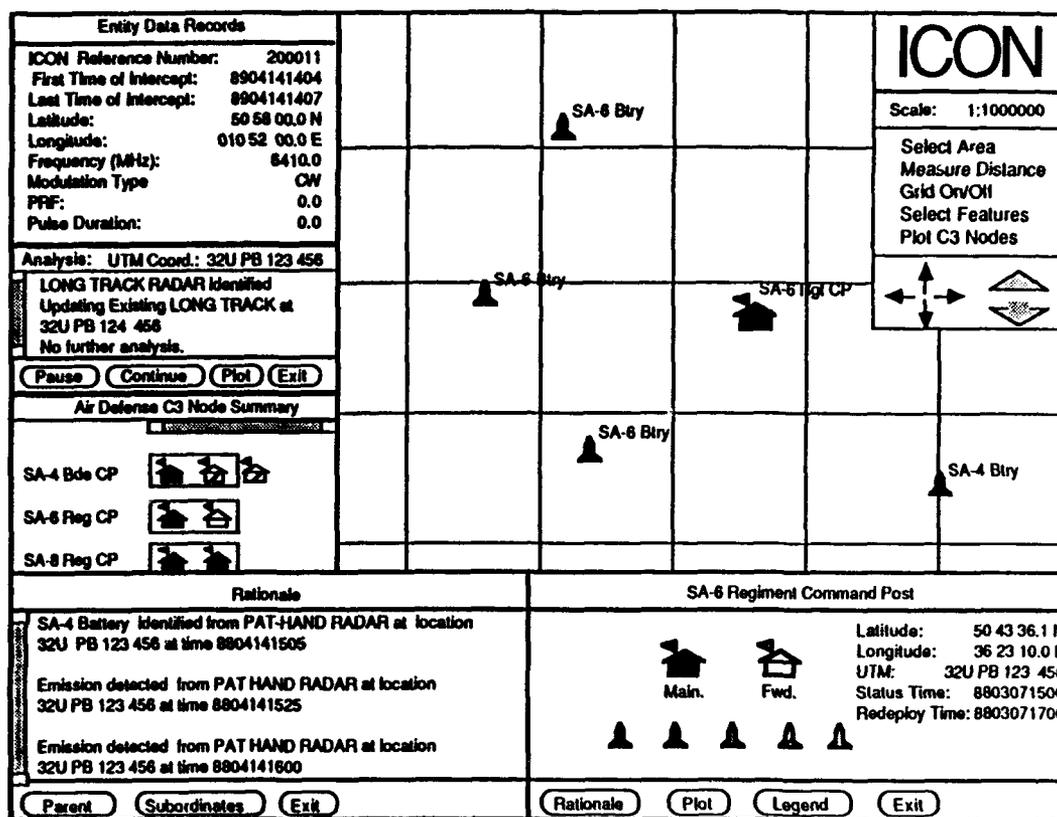


Figure 12: Sample ICON Display

piecemeal data. A trail (log of update messages) is kept on the EDR data as it is propagated through ICON's blackboard, allowing ICON to track the flow of data.

When additional information on the identification of a particular target is requested, backward chaining takes place on the trail. By revealing the trail that obtained the answer, the user is made aware of the process made in identifying the target. Figure 12 shows the rationale window (displays the trail) for a Command Post in the bottom left corner. The trail is formed by the successful "firings" of the knowledge sources on data in the blackboard.

The knowledge sources that use the Embedded Rule-based System (ERS) are derived from a common base class (ERS KS), that is derived from the class provided by the blackboard system (for more information on ERS reference section 4.3.1). These knowledge sources are associated with a rule base and a set of functions for gathering evidence from the blackboard and performing actions to update the blackboard. Figure 10 labels these knowledge sources as **KS-1** .. **KS-n**. These knowledge sources represent a group of independent experts that can generate a solution to a problem by manipulating data within the blackboard. Reference [2] contains a complete description on the knowledge source implementation.

Creating a new object at the highest level of the blackboard results in the plotting of a new icon. This icon is displayed at the approximate latitude/longitude of the threat as indicated in the EDR messages. The icon represents that conclusive evidence has been

derived from the EDR messages and is contained within the blackboard to suggest that a particular enemy asset exists at a particular latitude/longitude.

4.2.2 Icons and Spreadsheets

ICON uses graphic icons to portray units of enemy assets. These assets are plotted at their approximate location, on a two dimensional map of the terrain. ICON also provides labeled cities, a coordinate grid, a Forward Edge of the Battle Area (FEBA), and a Fire Support Coordination Line (FSCL), to help orientate the analyst on the battlefield. The map portion of figure 12 shows only some known units and the coordinate grid.

ICON uses simple icons that intuitively remind the user of the object it represents, with varying icon colors to indicate an icon's current status. Enemy units, portrayed as a single icon on the map, represent many vehicles and equipment that are associated with a particular unit. Due to the size of the display, ICON is not able to plot each member of the unit and provide an uncluttered view of the battlefield. Therefore, ICON provides an alternative method for retrieving more information about a particular unit.

To reveal more information about the displayed assets, the user "clicks" (with a mouse) on an icon. Backward chaining (described in section 4.2.1) is performed on the icon's blackboard representation to reveal the identified pieces of equipment known to compose the unit. This information is displayed in a tree format in a separate

display window (lower right window in figure 12).

The tree format shows the unit icon at the top, with each of the known assets within the unit linked underneath it. Many analysts, aside from knowing the available assets that are associated with a unit, also want to know how the associated assets became known. The past messages used to determine an enemy asset may be accessed through the rationale window.

To determine the rationale (EDR messages used to identify an object), the user "clicks" (with a mouse) on a limb of the tree. The backwards chaining process (described in section 4.2.1) is invoked on the blackboard's representation of the link to reveal the evidence supporting the components identification. This evidence is presented in text format in a separate window. The bottom portion of figure 12 shows the tree window, with the rationale window on its left.

ICON demonstrates a good blend of textual spreadsheets and graphic icons. ICON shows that the use of graphics can be used to enhance a text-based system and still allow the user access to text or exact numbers when needed. It does not limit a user to a graphic icon or require an extensive search to find the exact data.

4.3 Target Analysis over Time

Part of an analyst's job is predicting future enemy activity. By comparing current trends in threat movements against known doctrine or past performances, a correlation may (help) determine the future moves of the threats. The ability to notice these trends early in the discovery of a threat may provide additional insight to an enemy's goals.

Upon noticing trends, the analyst may determine the predicted path of the enemy and possibly some of his targets. By obtaining this knowledge, effective countermeasures can be prescribed. This information may show a vulnerable area in the friendly force defenses or reveal a valued enemy resource or target. Although friendly forces are not shown in ICON, this capability could be added in the future.

The ability to compare past and current trends in enemy force deployment can be extremely helpful to an analyst. However, acquiring the past data and drawing conclusions based upon the data is complex.

4.3.1 Inference Mechanism

When a knowledge source is "triggered," ERS is notified to execute the associated rule base. The knowledge source then provides the information from the blackboard to the executing rule base. The rule base action functions process and return updates to the knowledge source. Upon receiving the updates from ERS, the knowledge source performs analytical updates to the blackboard. Figure 10 shows the communication between ERS and the rule bases, and ERS and the knowledge sources.

ERS represents the collection of rules (links) that form a rule base as an inference network. Within this network, nodes represent antecedent conditions and consequent statements with an associated degree of belief (value reflecting the importance of the antecedent condition and consequent statements). Links represent rules which may have an associated additive weight of evidence.

Two types of nodes are used in the inference network, leaf and goal nodes. A leaf node represents evidence provided from the blackboard or model data structure. A goal node that is not an antecedent condition for any rule represents conclusions whose certainty is to be determined by the rule base. A goal node is usually associated with an action function that performs some change to the blackboard.

The rule base may execute one or more action functions when a single knowledge source is "triggered." The action functions that are executed depend upon the evidence collected, the certainty of evidence (value assigned to goal nodes by the rules as a function of the amount of evidence), and the strategy for action function execution specified for the rule base. An in-depth discussion on the rule base may be found in [2].

Although not currently implemented, the rule-based inferencing available in ERS could be used to determine objectives and intent of threats by filling it with the threats doctrine.

4.3.2 Coordinated Mission Maps

ICON does not animate the icons on its display. Once a threat is detected and put on the screen it will remain there until the analyst determines the data is too old. ICON treats the data as a time-tagged snapshot of the world. However, by adding some intelligence about the decay rate of information and the tactics of the enemy, ICON could animate its data.

Adding intelligence about the decay rate of information would allow ICON to remove old data. As the information concerning a specific threat becomes old (no new information to backup the old information), the confidence level (a value that represents the degree of belief that an object exists) in the threat would subside. As the confidence level begins to decay, the icon would turn back into a situation map (described in section 4.1.2) and may eventually disappear.

Adding enemy tactics (possibly within the inference mechanism described in section 4.3.1) would allow ICON to animate the projected moves of the enemy. The icons displayed on the grid portion of figure 12 represent the icons that could be animated. ICON could track the enemies known positions and play them back to the analyst. From these movements and the knowledge of enemy tactics, ICON could begin to animate projected moves. This capability could greatly enhance the analyst's ability to recognize key coordination points in the enemy's attack, intentions, and objectives.

Applying spatial and temporal coordination constraints on the data being tracked can reveal other threats that may exist, but have not been found. Refueling, supply lines, and weapon depots are a few items that could be found by animating and observing the threat activity. The time-stepped animation of events may also allow the slack times and critical rendezvous of the enemy's plan to be seen. Reference [1] for a complete description of coordinated mission maps and their application to Force Level Planning animation.

5. CONCLUSIONS

Applying the graphical enhancement techniques derived in the GSR study to ICON and TEMPLAR proved to be a worthwhile task. The process of applying these techniques revealed that some of the enhancements already existed (ICON's use of icons and spreadsheets), other enhancements could be used directly from the study (TEMPLAR's use of TOT sliders and resource allocation networks), and a few of the enhancements required tailoring to the problem domain (ICON's use of situation maps and coordinated mission maps). Therefore, the task demonstrated that the graphical enhancements derived in the study could be applied to existing AI applications.

This task also provided insight into the amount of information that is captured within the AI concepts, but not shown to the user (possibly due to time and processing constraints). TEMPLAR's related schemata and ICON's blackboard are two examples of information captured but not displayed. The relationships between TEMPLAR's schemata are spread across several worksheets (making the relationships very difficult to detect) and ICON only displays data (objects) on the top level of the blackboard (not allowing the user to see other information contained within the blackboard). The graphical enhancements discussed on each of these concepts would allow this data to be shown clearly to the user.

Each application has its own requirements for textual and graphical data. Graphics should be used to help the user of the system, and not over-used to the point of causing more confusion. One way to avoid adding confusion to a system is to expose the user to the enhancements as part of an iterative cycle: analysis of the system, implementation of enhancements, obtain user feedback, apply feedback to analysis of the system. In this context, this paper has only addressed the analysis phase in the iterative cycle.

Users must be made aware of the enhancements that can be provided to them by combining graphics and AI concepts into their current methodologies. Implementing a system that strictly mirrors a nonautomated methodology may not be using the computer to its maximum potential. By composing an appropriate blend of AI and graphics, much of the influx of data currently felt by the user may be avoided. However, an absence of this blending could result in unfriendly, inefficient, or misleading systems.

This paper has attempted to demonstrate the need for synergism between AI concepts and

representative graphical output. By focusing on a few examples, it is the intent of the paper to convey the overall usefulness of this synergism.

6. REFERENCES

- [1] "Graphical Techniques for Force Level Planning," Final Technical Report, March 1991, General Electric Company Corporate Research and Development, sections 2-6.
- [2] "Identification of Command and Control Operations Nodes (ICON)," Final Technical Report, RADC-TR-90-236, 10 May 1990, PAR Government Systems Corporation, sections 1-3.
- [3] Knowledge Craft^R 3.2 Training Manual Volume 1, version 3.2, 05/13/88, Carnegie Group Inc., Knowledge Craft^R Introduction section.
- [4] Knowledge Craft^R Volume 1, version 3.2, 05/27/88, Carnegie Group Inc., Overview section and section 4.
- [5] "Tactical Expert Mission Planner (TEMPLAR)," Final Technical Report, RADC-TR-89-328, January 1990, TRW Defense Systems Group, pp. 1-2, 11-12, 15, 29-30.
- [6] "Tactical Expert Mission Planner (TEMPLAR)," User's Manual (Final), 9 November 1988, TRW Defense Systems Group, pp. 58-82, 138, 145.



AD-P006 351

ENGINEERING GRAPHICAL ANALYSIS TOOL (EGAT)
DEVELOPMENT PROGRAM

Victor R Clark
Joseph R Diemunsch
Wright Laboratories
WL/AAAN-2
WPAFB OH 45433
USA

1. SUMMARY

The Air Force Avionics Laboratory has sponsored many efforts to develop real-time Artificial Intelligence (AI) systems. One of these systems, the Adaptive Tactical Navigation (ATN) program, developed a proto-type system, to intelligently control a future tactical fighter's integrated navigation sensors. ATN was developed using a distributed communicating expert object architecture called the Activation Frame (AF). Using the AF architecture, ATN was able to achieve real-time execution. Real-time execution was primarily achieved due to the AF's distributed control scheme which eliminates many of the bottlenecks associated with centralized schedulers. Unfortunately, with these increased benefits, there is increased complexity associated with correctly setting the distributed control parameters. The Engineering Graphical Analysis Tool (EGAT) was developed to overcome these limitations by providing a user friendly, graphical AF development tool. EGAT provides the capability to dynamically monitor and modify the AF control parameters.

This paper describes the Activation Frame (AF) architecture, and the development and implementation of the Engineering Graphical Analysis Tool (EGAT). The EGAT system is used to dynamically monitor and modify the decentralized control parameters of the AF architecture, a communicating expert object paradigm.

2. EGAT DEVELOPMENT PROGRAM

Through current and past development efforts, such as the Adaptive Tactical Navigation (ATN) and the Advanced GPS Receiver (AGR) Programs, the Activation Frame (AF) architecture has demonstrated its suitability to implement real-time intelligent avionic systems [5]. The Engineering Graphical Analysis Tool (EGAT) was developed to alleviate some of the problems that were encountered during the development of these real-time avionic systems. This paper will describe the utility of using the AF architecture to develop real-time intelligent avionics systems. In addition, the EGAT system will be described in detail. Due to the nature of the EGAT system, it is necessary to understand the functionality of the AF architecture. The next section will describe the AF architecture. This information will then be used as a basis to describe the EGAT system.

2.1 THE ACTIVATION FRAME (AF) ARCHITECTURE

The key to integrating intelligent systems into conventional avionic platforms is to develop efficient mechanisms which insure real-time operation. The Activation Frame

(AF) architecture was created to give real-time AI developers the capability to efficiently implement their knowledge based systems while satisfying real-time constraints [4]. Early attempts to develop real-time machine intelligence systems relied on a centralized blackboard scheme (Figure 1). Under this scheme, an intelligent blackboard manager would determine which expert module would execute next. Unfortunately, under the centralized approach, the majority of the processor time was used by the scheduler. Another major problem with early real-time AI development systems is that they were extremely difficult to integrate to conventional sequential software. Most of the time these systems ran in their own environment (usually in Lisp), and did not interface very well to other types of software. To alleviate these problems, several techniques were developed which utilize a decentralized control scheme (Figure 1). One of these schemes, the

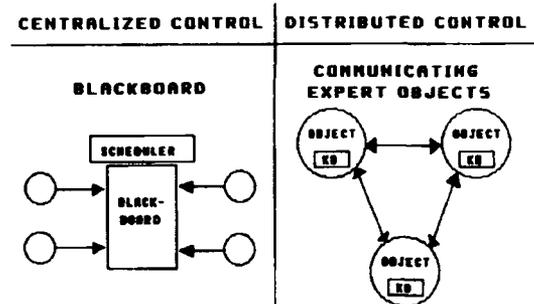


Figure 1. Real-Time Control Methodologies

AF architecture, is based on the Communicating Expert Object paradigm. The AF architecture utilizes a decentralized control mechanism to provide fast efficient scheduling and context switching [3]. This implies that the control knowledge is not located in one central area, but rather distributed evenly among the different modules that make up the system. This is accomplished through a message passing, object oriented paradigm in which the system expertise is located in modules called Activation Frame Objects (AFOs). The AFOs are designed to work in much the same fashion as human experts. Each AFO has a self contained knowledge base that allows it to reason about a specific domain. There is no shared knowledge or common memory between the AFOs. In order to communicate, prioritized messages are sent between the different AFOs in the system. In addition, each AFO has multiple message input ports (Figure 2). These multiple input ports

allow the real-time AI designer to set up AND/OR relationships between the different input ports of the AFOs to insure that all of the information needed to reach a conclusion is available. When this occurs, the AFO is considered to be primed, and the AFO is then ready to receive the processor for execution.

ACTIVATION FRAMEWORK ARCHITECTURE

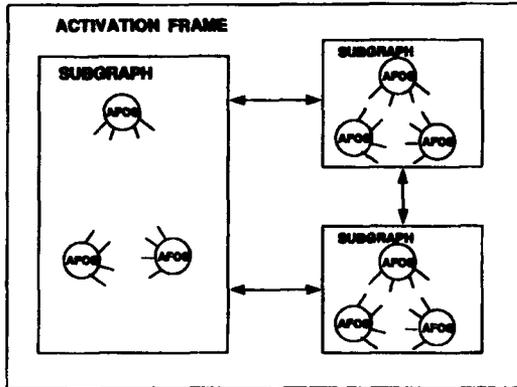
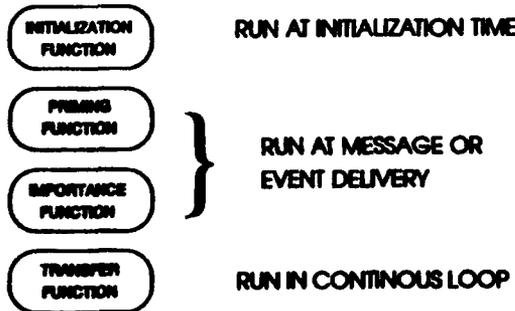


Figure 2, AF Hierarchy

Every AFO is made up of four functions: initialization, priming, importance, and transfer (Figure 3):

The initialization function is responsible for setting up all of the AFO communication channels and initial control values.

The priming function is used by the AF architecture to insure that an AFO has all of the information it needs to execute. This function is also responsible for setting up the AND/OR relationships between the AFO input ports.



* SYSTEM DEFAULTS PROVIDED

Figure 3, AFO Functions

Once an AFO is primed, the importance function determines the overall AFO importance level based on the individual priorities of the messages on the message input queues. The default importance function takes the summation of all the message priorities on the input queues to determine an overall AFO importance level. However, this can be modified to satisfy a particular application. If a system is primed, and it has the highest overall importance level, then it executes its transfer function.

The transfer function is responsible for reading in the AFO's input messages, reasoning over the information, and sending the results to the other AFOs in the system.

By modifying these four functions, the real-time intelligent system designer has maximum flexibility to tailor the AF control structure to suit his purposes.

A collection of AFOs that work in a common problem area is called a subgraph (Figure 2). The subgraphs are primarily used to partition the knowledge base in a logical manner to more easily visualize complex problems. The subgraph structure parallels the way that human experts work in a group to achieve a common goal. The subgraph structure does not place any physical restrictions on the message traffic between the AFOs. An AFO in one subgraph is free to communicate to an AFO in any other subgraph. However, for multiple processor environments, the subgraph level is useful for determining the proper distribution of AFOs in the system.

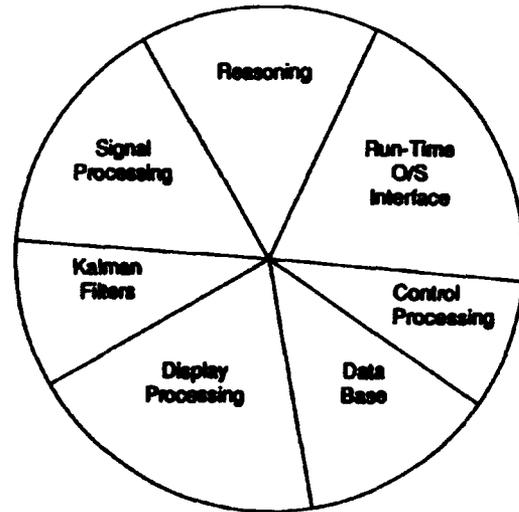


Figure 4, Intelligent System Breakdown

As seen in Figure 2, a collection of subgraphs form an AF. In the AF architecture, there is one AF per processor. The scheduling of the AFOs occur at this level. However, the scheduling mechanism is considerably different than conventional centralized control systems.

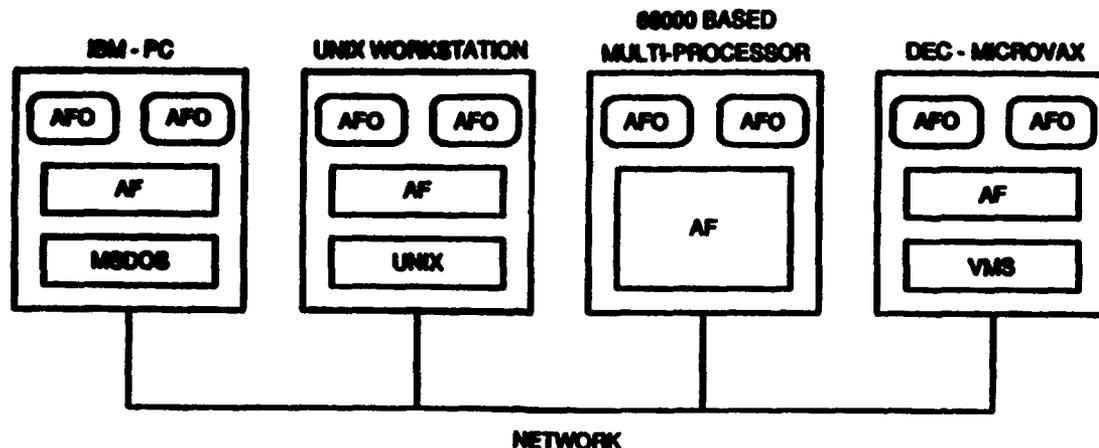


Figure 5, AF Operating Environment

The role of the AF scheduler is to determine which AFO has the highest importance level. The AFO which is both primed and has the highest importance level will execute next. Since the AFOs calculate their own priming and importance functions, the AF scheduler does not have to contain any domain knowledge about the individual AFOs. This results in a very fast, efficient distributed control structure. For multi-processor environments, there would be one AF per processor. Another very attractive feature of the AF architecture is that it is implemented in C and Ada versions. This makes it ideal for imbedded avionics applications because it can be easily integrated into existing avionics software. This is critical for future avionics systems, because only a small portion of the avionics system will be related to machine intelligence (Figure 4) [2]. It turns out that integration is one of the major problems associated with embedded intelligent systems. The knowledge engineer is usually able to get the individual intelligent modules to work. However, when the system is integrated together, it is very difficult to get the different pieces to work together. Compounding this problem is the fact that modern intelligent avionics control systems are usually made up from a mix of heterogeneous processors. This means that a CPU may need to talk to a Signal Processing Chip, or it may need to send information to a graphic display. Because of this, integration may take as much as 60 percent of the development effort [2]. The AF architecture solves this problem by providing a common framework between different classes of processors. The AF architecture has been successfully implemented on PC's all the way up to fault tolerant multi-processor systems (Figure 5). The hardware underneath the AF architecture is hidden from the user. Therefore an application developed for one run time environment can be easily transition to another environment as desired.

2.2 EGAT OVERVIEW

As described above, the AF architecture has many advantages over a conventional centralized architecture. Unfortunately, with this increased performance, there are also some difficulties which are unique to a distributed control system.

One of the greatest difficulties is properly setting the distributed control parameters. For imbedded intelligent avionics systems, it is critical that the proper functions are executed in a timely fashion. In the case of the AF architecture, the message flow between the AFOs determine the AFO execution order. Because of this, the sending order and priority of the AFO messages are critical to insure the correct behavior of the intelligent system. If the priorities of the messages are not correctly set, then an important message may be lost in the system. In addition, if the individual AFO importances are inappropriately assigned, the system will not have the correct context switching to insure proper focusing of the system, and important AFOs may not execute in time. The task of selecting the appropriate message and AFO importance priorities is very arbitrary and heavily based on the specific domain knowledge. Usually, relative importance levels are collected during the knowledge acquisition cycle and fine tuned during rapid prototyping. This process becomes even more complex in a distributed environment because control information is divided throughout the system. Therefore, when one AFO is modified, it effects all of the AFOs in the system. This phenomena makes it extremely difficult to fine tune the system. The EGAT system was developed to greatly simplify the development of real-time AF systems [1].

2.3 EGAT DESCRIPTION

The EGAT system is a graphical analysis tool which allows the user to examine the different levels that make up the AF

91-15536



91 1113 029

SUBGRAPH LEVEL

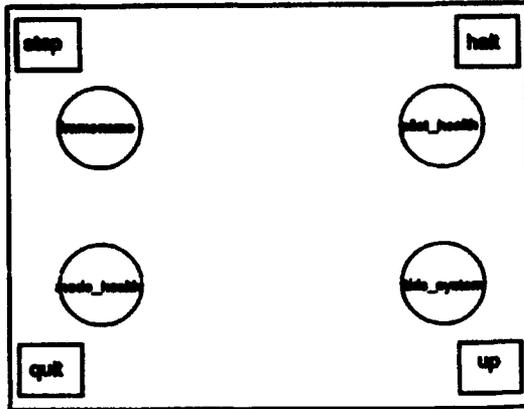


Figure 6. Subgraph Level

architecture. When the program is first started, the EGAT system displays the subgraph level (Figure 6). At this level, each one of the circles represent a subgraph in the system. For this particular example, there are four subgraphs shown: **framesname**, **mode health**, **pilot health**, and **jtids system**. In addition, when the EGAT system is running, there will be colored lines between the subgraphs that represent the level of inter-subgraph communication. These colors follow the color spectrum with purple representing the lowest message traffic all the way up to red, representing high message traffic. Determining the amount of message traffic between the subgraphs is extremely useful for distributing the subgraphs on multiple processors. By minimizing the inter-subgraph communication, the inter-processor communication can also be minimized when the subgraphs are placed on different processors.

At the corners of the EGAT system are the four action boxes: **halt**, **stop**, **up**, and **quit**. The **halt** action box allows the user to freeze all the running AF system at any time to examine its current state. While the system is halted, the user can single step through the system by selecting the **stop** action box. The **up** action box allows users to move from lower levels of the EGAT display to higher levels (i.e. AFO level to subgraph level). And finally, the **quit** action box allows the user to exit the system. These four action boxes are available at every level of the EGAT system.

For more detailed information about the individual AFOs that make up the different subgraphs, the user can use a mouse to select a particular subgraph. At this point, the EGAT system will display the AFO Level (Figure 7). In the AFO Level, each AFO is represented by a circle. For the example display shown in Figure 7, there are six AFOs shown. Each one of the AFOs is made up of an inner and outer circle.

The inner circle represents the current AFO state. As described earlier, every AFO in the system will be in one of three states: inactive, primed, or running, represented

AFO LEVEL

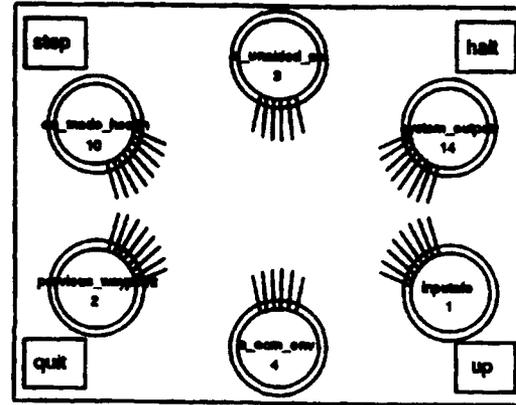


Figure 7. AFO Level

by a green, yellow, or red color respectively. Additionally, every AFO has an outer circle which graphically shows the AFO's current importance level. These levels are represented by one of seven colors in which purple is the lowest value, following the color spectrum, all the way up to red representing the highest overall AFO importance level. Furthermore, every AFO in the system has a set of eight attached lines representing the AFO's input ports. When one AFO sends a message to another AFO in the subgraph, a flashing line will appear from the center of the sending AFO to the appropriate input port of the receiving AFO. This allows the EGAT user to trace the message flow between the different AFOs in system. If the user wishes to determine the contents of the message or more detailed information about an AFO, then by using the mouse to select a particular AFO, the AFO Information Level will be displayed.

The AFO Information Level is responsible for displaying all of the detailed information about an AFO. For the example shown (Figure 8), the AFO, **a previous waypoint**, contained in the subgraph, **mode health**, has a current importance level of 21. In addition, the AFO also has two active input ports with no current messages on either port. The **<-** and **->** boxes are used to view and scroll text longer than the size of the message display box. The **PREV** and the **NEXT** boxes allow the user to switch between the different messages on a particular input queue. This condition will occur if several low priority messages are sent to a particular input queue or if the AFO is not primed. In order to use either the **<-**, **->** or **PREV**, **NEXT** boxes, the user must first select the appropriate message input port by "clicking" on it once with the mouse. After this the message input port will become active and the scrolling functions will be activated.

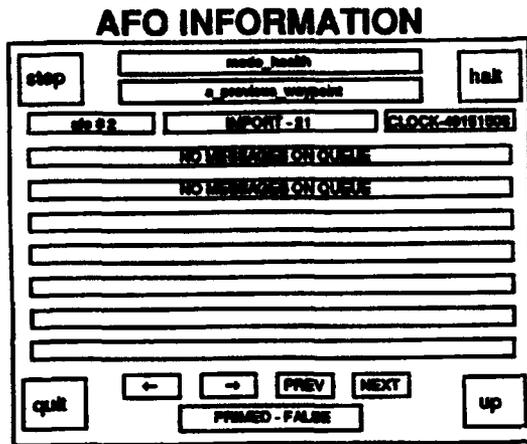


Figure 8. AFO Information Level

The real power of the EGAT system is not in its ability to just examine the inter-workings of a running AF system, but rather, its ability to dynamically alter the AF's control and message passing parameters. As described earlier in the paper, one of the key difficulties associated with using the AF architecture and other distributed control systems is to correctly set the control parameters. In order to properly set up the AF control mechanisms, the real time-AI designer must select the correct overall AFO importance levels and the individual AFO message activation levels. In addition, the real time designer must be concerned that the priming and importance functions have been correctly set to insure the proper focusing of control system. With the EGAT system,

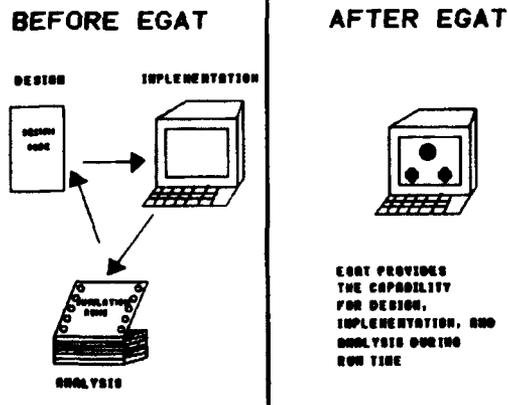


Figure 9. EGAT Payoffs

the user can freeze the system (halt action box) and go into the AFO Information level to modify the AF control parameters. Before the development of the EGAT system, real-time intelligent system developers were forced to run their system and capture the inter-AFO messages to a file. One this was accomplished, they would have to

manually examine the inter AFO-message traffic to determine if the system was functioning as desired. If any AF parameters needed to be modified, they would have to go back and modify the original source code, re-compile the code, rerun the system, and manually analyze the results (Figure 9). If the wrong parameters were selected, they would have to repeat this whole process again. Needless to say this was a very length, arduous process that limited the effectiveness of AF design. With the EGAT system, the user does not have to manually analyze the results of the run-time system. With EGAT's single step and halt capabilities, the developer can insure that the run-time system is functioning as desired. If some of the AF control parameters need to be modified, the developer can change the parameters, without recompiling the system. In this way the developer can see the results of his changes in real-time. Clearly providing the ability to single step and dynamically modify AF control parameters are the most useful features of the EGAT system.

3.0 PROGRAM STRUCTURE

This section of the paper will focus on the structure of both the AF architecture and the EGAT system.

3.1 AF STRUCTURE

In order to understand the overall structure of the EGAT system, it is important to understand the basic structure of the AF architecture. The EGAT system interfaces directly to the AF architecture which allows manipulation of the data structures, and provides halting and single stepping functionality to the running AF system. The major data structures of the AF system are shown in Figure 10. At the heart of the AF architecture is the `frm_struct` structure. There is one `frm_struct` structure per processor. The `frm_struct` points to the `af_struct` structure and the `ptb_struct` structure. The `af_struct` structure contains all of the internal details of a single AFO. There is one unique `af_struct` structure for every AFO in the system. In addition to the `af_struct` structure, the `frm_struct` structure also points to the port table structure, `ptb_struct`. As described earlier, every AFO in the system has at least one message input port. Each message input port has a unique entry in the port table structure, `ptb_struct`. In this manner, any AFO in the system simply needs the name of the destination AFO to send a message. The data in the `ptb_struct` is used to get the destination AFO's port structure, `pt_struct`. The `pt_struct` structure hangs directly off of the `af_struct` structure and is used to store the AFO's incoming message. There is always one `pt_struct` that corresponds directly to an AFO input port. Linked to the port structure is the actual AFO messages. These messages are stored in the `msg_struct` structure. The `msg_struct` structure

AF STRUCTURES

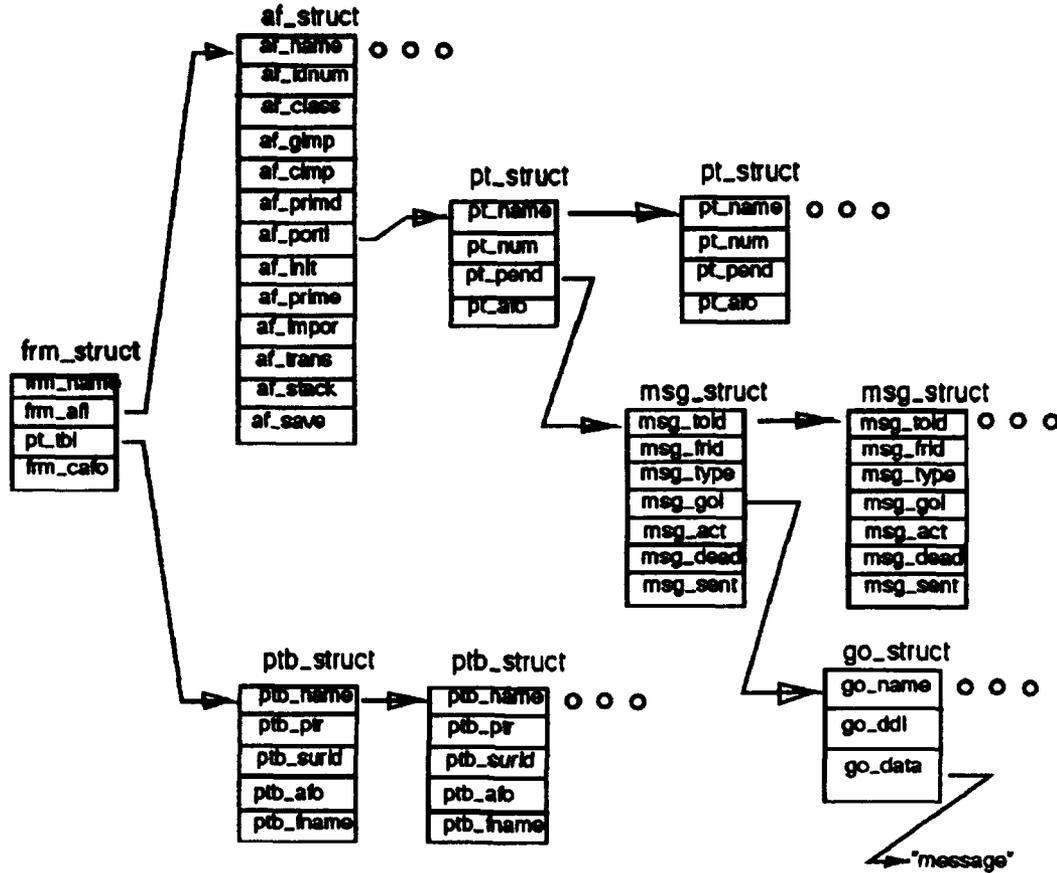


Figure 10. AF Structure

contains several important fields associated with the AFO messages. They are as follows: the message importance level, **msg_act**; the length of time the message is valid, **msg_dead**; and the actual message data, **msg_gol**. The **msg_gol** field contains a pointer to elements called Generalized Objects (GOs). GOs were developed as a means to transfer data elements between heterogeneous computers. The GO is made up of a Data Description Language (DDL) and a pointer to the data. The DDL tells the receiving computer how much and what type of storage to allocate for the message. In this fashion, the message passing between different types of computers become transparent to the user.

3.2 EGAT STRUCTURE

The structure of the EGAT system is shown in Figure 11. The EGAT internal data structures closely resemble the hierarchical nature of the AF architecture. The **HEAD DATA** structure stores the information to be displayed in the subgraph level. Contained in the **HEAD DATA** structure

are several major fields that enable EGAT to access and graphically represent the AF architecture. The two fields that enable EGAT to directly interface to the AFO system are **afname** and **af number**. With this information, EGAT can gain direct access to the **frm_struct** of the AF architecture. Additionally, the **x,y,radius, number afo, key,** and **line_struct** fields are used by the EGAT system to store graphical information. In addition to the subgraph information, the **HEAD DATA** structure also contains a pointer field, **afo_ptr**, to the different AFOs that make up the subgraph. The AFO information is contained in the **AFO DATA** structure. This structure is responsible for displaying the AFO Level of the EGAT system. The primary fields of the **AFO DATA** structure are as follows: **afname, afoname,** and **afo number** fields are used to directly access the AFO architecture; **x,y,radius, key, import key,** and **import text_key** fields are used for EGAT graphic displays; and finally the **box** field points to the **BOX DATA** structure. The **BOX DATA** structure is responsible for storing the information needed to display the EGAT AFO Information

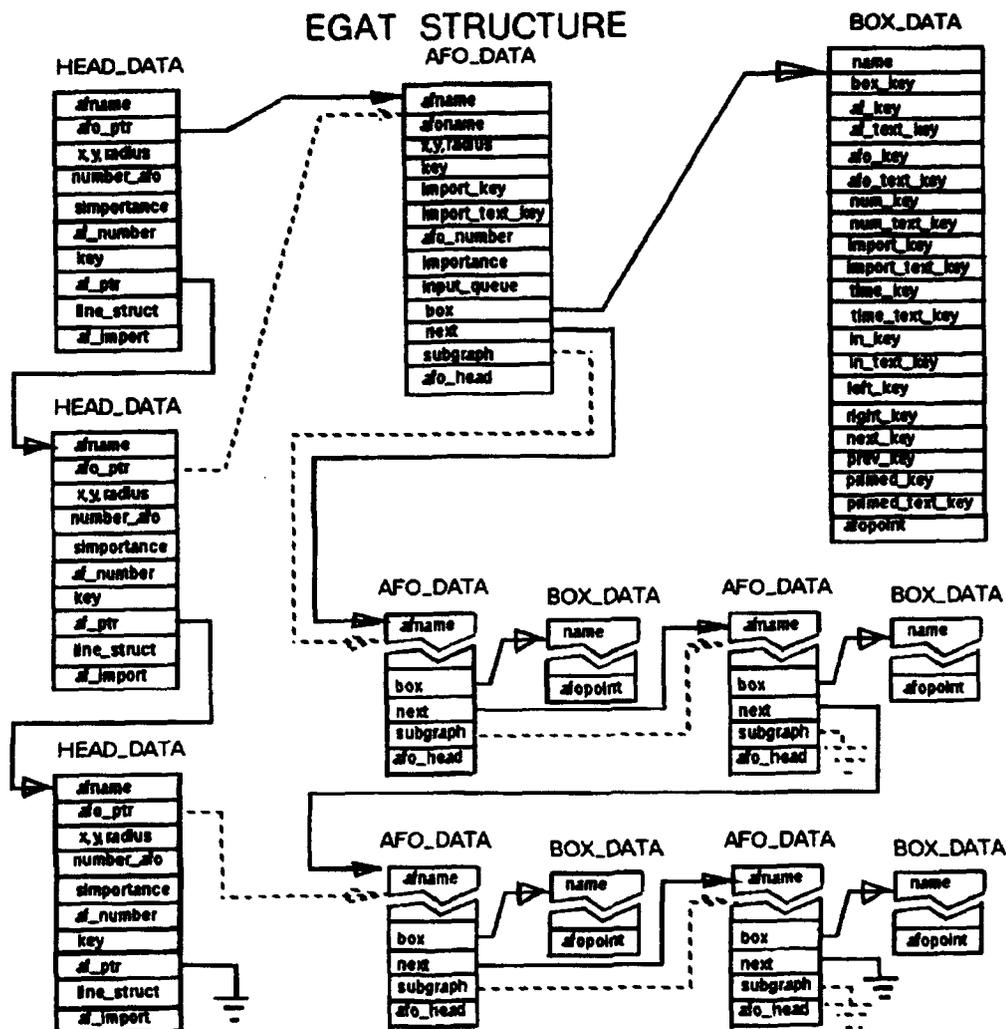


Figure 11. EGAT Structure

Level. The majority of the fields in this structure are responsible for holding graphic information that is necessary to display and scroll text and graphics. In addition, the **afopoint** field points back to the parent AFO. Through this pointer, the EGAT system can interface directly to the AF architecture at the AFO Informational Level.

4.0 EGAT OPERATION

The EGAT system connects directly into the AF architecture, which not only allows the monitoring of the running AF system, but also gives the user the ability to modify the AF system to obtain efficient and correct operation. This is extremely useful when developing embedded intelligent avionics systems because it allows the developer to see the internal workings of the system before it gets inserted into an embedded environment. With EGAT, the system designer can develop a system in a

workstation environment, and then transition his application to other platforms.

EGAT was designed to be easy to operate and user friendly, thus providing useful insight into the operation of the AF system. After startup, EGAT can be used to monitor the running AF operation. The AF system can either be examined in a free running mode or the system can be single stepped in order to analyze performance in detail. The free running mode is useful for determining the flow of the overall system. The AF architecture provides an excellent environment for distributing different tasks on multiple processors. The free running mode can be used to help the designer determine which tasks should be placed on multiple processors. The AF system runs in real-time with the message traffic represented at the Subgraph Level by the changing colored lines between the subgraphs.

A more effective way of determining the message by message operation of the AF system is in the single step mode. In this mode the AF system is halted and allows the user to see the static effect of all message passing. In this mode the correct operation can be determined because it is easy to see what effects the messages have on the system.

Several types of AF control problems can be analyzed quickly with EGAT, including bottlenecks, livelock, and starvation conditions. A bottleneck occurs when an AFO has a large number of important messages in its input queue, but does not have all of the information it needs to be primed. In this case, the AFO will have a high importance level; however, it will not even be considered for execution. EGAT can help in determining the bottleneck situation by observing which AFOs have an internal green circle, meaning that they do not have the information needed to become primed, but at the same time have a red or orange outer ring meaning that their importance is high. If this state persists then it shows that this AFO is not getting the information needed to run and is a bottleneck. This situation can be helped by going into the AFO information level on the bottleneck AFO and injecting messages to prime the AFO. From this the effects on the AF system can be determined.

The next condition, livelock, occurs because an AFO's importance level is not properly set and does not drop enough to release the processor after it has sent out its messages. In addition, this condition could also result from a group of AFOs continuing to loop processor control amongst themselves and never releasing the processor for other AFOs. Livelock can be determined by an AFO releasing its messages and immediately returning to a primed and then running state without any other AFO having a chance to obtain the processor. This situation can be helped by going into the running AFO's Information Level and forcing its importance level down. This will allow another AFO with a higher AFO importance level to run. A similar method can be used to break out of a loop by reducing the importance level of several or all of the AFOs in the loop to allow an AFO outside of the loop to get the processor. The effects on the system can then be observed.

The last condition, starvation, occurs when an AFO is primed, but never gets a chance to execute. Starvation can be determined with EGAT by observing an AFO or several AFOs that are primed for long periods of time but never obtain a high enough importance to run. This situation is represented graphically by the AFOs internal circle being yellow, meaning it's primed, and it's outside ring never reaches higher than a purple, blue or light green, meaning the AFOs importance does not reach a sufficient importance level to run. A fix to this situation is to go to the AFO Information Level and increase the importance level of the AFO to see the effects on the system.

As can be seen by these debugging features EGAT is very useful to provide insight into

the inter-workings of the AF system. From these insights, major bugs in the running system can be found and corrected while the system is still running. The effects are then determined and allows the designer to more permanently fix the external files that contain the AFO and message importances. This ultimately allows the system designer to develop more efficient and effective AF systems.

5.0 Real-Time AI Development Issues

Before real-time embedded AI systems can be integrated into avionics applications, two major issues need to be addressed, namely verification/ validation and maintenance of real-time embedded intelligent systems. Currently, verification and validation

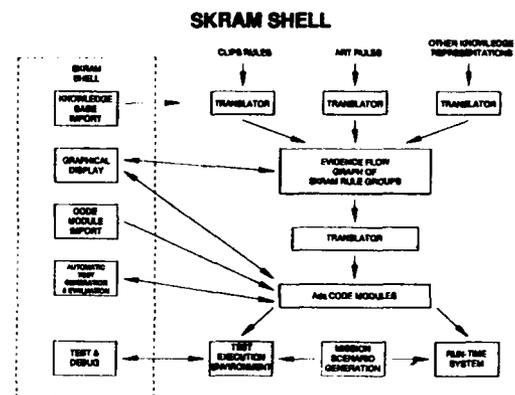


Figure 12, SKRAM architecture

issues are difficult enough for conventional sequential software, let alone data driven distributed intelligent systems. Most intelligent systems that are developed for real-time operation tend to embed the knowledge base into the inference engine in order to increase performance of the system. This makes it extremely difficult to initially verify and validate the run-time code, and nearly impossible to maintain the code as the knowledge in the system is modified. In order to overcome these limitations, the Avionics Laboratory is sponsoring several contractual efforts to examine methodologies to verify, validate and maintain real-time intelligent systems. The first of these programs, Knowledge Representations into Ada Methodologies (KRAM), developed methodologies to verify the knowledge base and automatically transition it into efficient Ada code using the AF architecture as the run-time system. The second major program, Shell for Knowledge Representation into Ada Methodologies (SKRAM), is developing an integrated development maintenance environment based on the research performed under the KRAM effort (Figure 12). The EGAT system will be

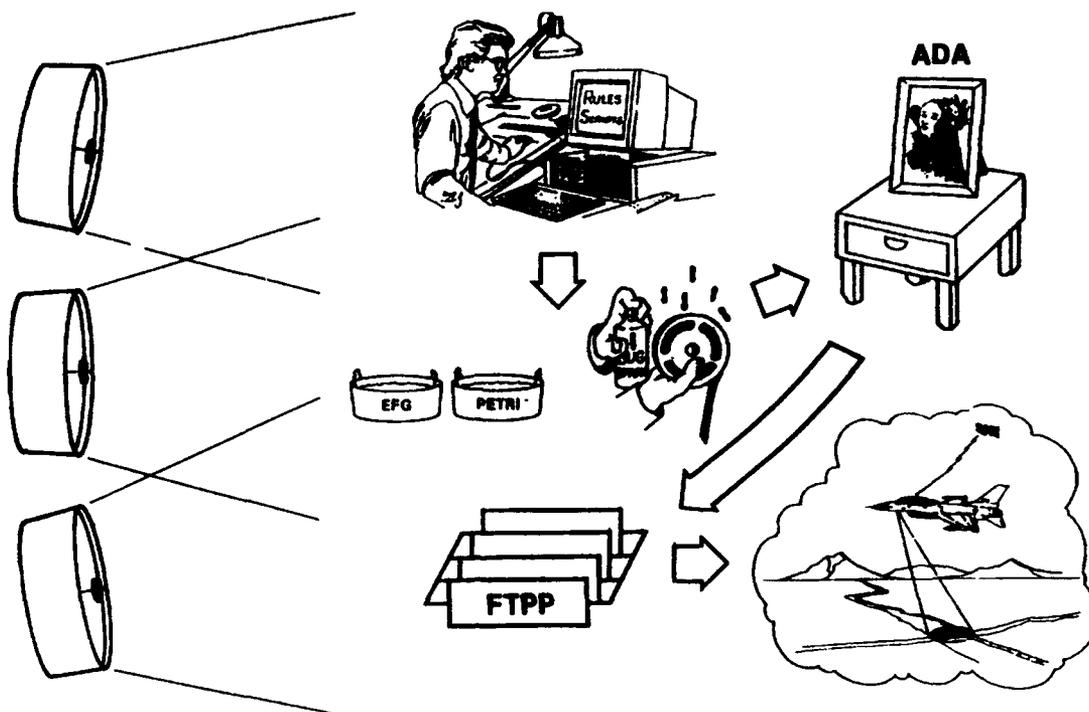


Figure 13, Transition Path

integrated into SKRAM environment to provide a robust tool for modifying the AF control parameters of the run-time system.

6.0 Conclusions

As the complexity and amount of information needed to control complex avionics systems increase, so will the need increase for intelligent real-time software to help in the management of these systems. In order to take maximum advantage of available resources and satisfy real-time constraints, more real-time intelligent systems are moving towards a decentralized control scheme. Because of the increased complexities associated with decentralized distributed control, the EGAT system was developed to aid knowledge engineers and system developers in the development of real-time embedded avionics systems. The EGAT system provides a user friendly environment to interact with and modify a running AF architecture. For the reasons described above, the AF Architecture is ideally suited for real-time imbedded avionics systems. With it's ability to run in either an Ada or C environment, it's modular design, and it's distributed control scheme, the AF architecture will provide an excellent run-time environment for embedded intelligence avionics systems. The only major difficulty associated with the AF architecture and other distributed systems is the increased complexity associated with setting up the initial distributed control scheme. With the development of the EGAT system, the system designer has the capability to develop and debug his system on a workstation environment, then transfer it to an

embedded platform.

The intent of the EGAT tool and our other real-time AI development tools are to shed some light on the problem of transitioning real-time AI systems from the laboratory into the operational environment. With the use of the KRAM and SKRAM environments, the knowledge engineer can verify and validate his knowledge base and automatically convert the knowledge base into Ada code using the AF architecture as the basic run-time system. At this point the knowledge engineer can use the EGAT tool to tune the AF control parameters. After this, the run-time system can be placed on a multiple processor environment to increase run-time performance. In this way, we hope to provide a complete transition path from the laboratory to the operational environment (Figure 13).

7.0 References

1. Clark, V.R. and J.R. Diemunsch, "Engineering Graphical Analysis Tool (EGAT)", Report WRDC-TR-90-1136, Avionics Laboratory, Wright Research and Development Center, Wright Patterson AFB, Ohio, October 1990.
2. Green, P.E., J. Duckworth, L. Becker, S. Cotterill, "Maintenance System For AI Knowledge Bases," Small Business Innovation Research Program Phase I Final Report, The Real-Time Intelligent Systems Corporation, Worcester, MA, February 1991.
3. Green, P.E., "The Activation Framework Software Package AFC C-Language Version 2.1 User's Manual", The Real-Time Intelligent Systems Corp., Worcester, MA, February

1990.

4. Green, P.E., "AF: A Framework for Real-Time Distributed Cooperative Problem Solving", in Distributed Artificial Intelligence, ed. M.N. Huhns, pp. 153-175, Pitman Publishing, London, England, 1987.

5. Final Technical Report for AF/WRDC under Contract No. F33615-86-C-1043, The Analytic Sciences Corp., Reading, MA, February, 1990.

6. Green, P.E. and W.R. Michalson, "Real-Time Evidential Reasoning and Network Based Processing," Proc. IEEE First Annual Conf. on Neural Networks, San Diego, CA, June 1987.

7. "Knowledge Representation into Ada Parallel Processing," Final Report under NASA Contract NASA-18565 Task 10, The Charles Stark Draper Laboratory, Inc., Cambridge, MA 1990.

8. "Validation and Verification of Intelligent Systems Using Evidence Flow Graph Techniques," Final Report under NASA Grant NAG-1-964, Worcester Polytechnic Institute, Worcester, MA 1990.

9. Green, P.E., D.P. Glasson, J.M. Pomerade, and N.A. Acharya, "Real-Time Artificial Intelligence Issues in the Development of the Adaptive Tactical Navigator," Proc. Space Operations-Automation and Robotics Workshop, NASA Johnson Space Center, Houston, TX, August 1987.



COMPONENT PART NOTICE

THIS PAPER IS A COMPONENT PART OF THE FOLLOWING COMPILATION REPORT:

(TITLE): Conference Proceedings of machine Intelligence
for Aerospace Electronic Systems Held in Lisbon, Portugal on
13-16 May 1991 (L'Intelligence Artificielle dans les Systemes
Electroniques Aero spatiaux)

(SOURCE): Advisory Group for
Aeronautical Research and Development, Neuilly-sur-Seine
(France)

TO ORDER THE COMPLETE COMPILATION REPORT USE AD-M242025

THE COMPONENT PART IS PROVIDED HERE TO ALLOW USERS ACCESS TO INDIVIDUALLY AUTHORED SECTIONS OF PROCEEDINGS, ANNALS, SYMPOSIA, ETC. HOWEVER, THE COMPONENT SHOULD BE CONSIDERED WITHIN THE CONTEXT OF THE OVERALL COMPILATION REPORT AND NOT AS A STAND-ALONE TECHNICAL REPORT.

THE FOLLOWING COMPONENT PART NUMBERS COMPRISE THE COMPILATION REPORT:

AD#: _____ TITLE: _____

- AD-P006 331
- AD-P006 334
- AD-P006 336
- AD-P006 344
- AD-P006 345
- AD-P006 346
- AD-P006 352



NOT REPRODUCIBLE
FOR PUBLICATION
WITHOUT PERMISSION
OF THE DTIC

Approved For	
NHS - G-1 ✓	
DTIC - F-3	
University of Justification	
By _____	
Distribution /	
Available By Order	
Dist	Available for Special
A-1	

AD-P006 352



A KNOWLEDGE-BASED INTELLIGENT TUTORING SYSTEM: ACQUIRE™-ITS

by

B.A. Schaefer, R. Side, R. Wagstaff, O. Magusin, & I.R. Morrison

Acquired Intelligence Inc

#205-1095 McKenzie Ave

Victoria, B.C. V8P 2L5

Canada

91-15535



1. SUMMARY

Acquired Intelligence Inc has developed an automated knowledge acquisition system and expert system shell known as ACQUIRE™. A project has been undertaken to extend the ACQUIRE™ system for use as a domain independent, knowledge-based Intelligent Tutoring System. The intent of this project was to provide the tools that would enable corporate or governmental personnel to rapidly develop their own expert system knowledge bases and rapidly convert these knowledge bases for use in task/case oriented training. A major thrust of this project was to provide tools that would enable computer users, rather than computer professionals, to develop the training packages. This extension to ACQUIRE™ has involved the development of two major modules: the Curriculum Author Module; and the Tutoring Module.

Using the Curriculum Author Module experts or instructors are able to develop a curriculum composed of lessons in a case-study approach. The system is oriented toward "learning by doing", i.e., students learn by solving actual cases - supported by simulation if required. These cases are packaged into lessons by the expert to demonstrate important and related concepts. The Curriculum Author, utilizes the structure and content of the expert knowledge base to help the expert select the best cases to package together into lessons.

The Tutoring Module monitors a student's progress in solving cases and constructs a representation of the student's growing knowledge as a "naive" ACQUIRE™ knowledge base. The Tutoring Module gradually merges the student's naive knowledge base with the expert's by identifying incorrect rules, gaps in the knowledge, and, most importantly, poor structuring of the knowledge (i.e., incorrect packaging of rules) which indicates misconceptions about how knowledge should be used. The Tutoring Module accomplishes this merger by selecting and presenting appropriate cases, from the lessons constructed by the expert, to help the student elucidate the correct knowledge in active problem solving. Additionally, the Tutoring Module explicitly demonstrates differences between the student's knowledge base and the expert's.

ACQUIRE™-ITS is written in the C Programming Language for execution on personal computers running MS-DOS or UNIX. The system has an X-Window interface and incorporates hypermedia. ACQUIRE™-ITS is designed such that it is applicable to a broad range of industrial and governmental applications. As with ACQUIRE™, these intelligent training applications will be available on affordable computers and will be developed directly by the experts.

2. INTRODUCTION

A client of Acquired Intelligence Inc expressed significant interest in the development of a cost effective environment for building intelligent tutoring applications. This client required knowledge-based intelligent tutoring in an environment that will enable computer users, rather than computer professionals, to develop the knowledge bases and training courses.

The past decade has realized technological developments that facilitate delivery of these requirements. Knowledge-based systems have come into routine use while intelligent tutoring systems have realized their first commercial successes [1]. During this period both knowledge-based systems and intelligent tutoring systems were employed in training, each with their own strengths and weaknesses. Knowledge-based systems technology had developed to the point that numerous domain independent shells existed to assist in the development of applications; however, the technology was not designed with training in mind. The training capability of knowledge-based systems relied exclusively on human learning capabilities rather than on inherent instructional techniques. Intelligent tutoring systems were designed specifically for training, however, the technology had not advanced as far as the creation of intelligent tutoring system shells. Applications were handcrafted and consequently extremely costly to produce.

These technological developments, along with growing demand from industry for cost-effective computerized training tools, prompted Clancey [2] to propose knowledge-based intelligent tutoring systems as a practical solution applicable to a wide variety of industrial problems. Knowledge-based intelligent tutoring systems employ the knowledge represented in expert system applications as the subject matter to be taught. In these systems, the student learns in apprenticeship fashion by examining cases run through the inference engine. The tutoring component guides and queries students about their understanding throughout the execution of a case, as well as across cases.

Clancey argued that the focus of student learning should be the correct conclusions to draw in a wide variety of circumstances and the factors involved in determining those conclusions. To this end, his method employs case-based instruction to give the student "experience" in situations that make important aspects of expert knowledge explicit. He has found that much of the necessary information required to teach expert knowledge is inherent in the knowledge bases of existing expert systems. Clancey constructed a number of special purpose programs to extract such information, from knowledge-bases implemented in the M.I expert system shell, for use in instruction.

This approach opens up the possibility of creating generic, intelligent tutoring system shells, like those for expert systems. Clancey has demonstrated that this approach represents a real instructional alternative to traditional instruction methods, automated or otherwise. Given the early stage of this research, however, there remain limitations to this approach.

First, the approach is best suited to teaching specific skills rather than general overviews of various fields. Given the number of specific skills for which training is in short supply, this restriction is not a real hindrance to the utility of the approach.

Second, the approach inherits the central limitation of expert system technology - the knowledge acquisition problem. It is very difficult and time consuming to acquire knowledge from domain experts, and it is similarly difficult to construct a well structured knowledge base that provides complete and thorough coverage of the problem domain. Clancey's approach relies on the assumption that a knowledge engineer has performed an adequate job of eliciting domain knowledge from an expert and produced a well structured and complete knowledge base. This is a time-consuming and error-prone process that represents a more serious limitation to the utility of knowledge-based intelligent tutoring system technology. However, with the advent of automated knowledge acquisition systems [3] it has become much more reasonable for experts to develop well structured and accurate knowledge bases on their own.

Third, given that the system described by Clancey only permits case execution, queries on case execution, and recording of student performance, it could be viewed as more of an intelligent testing system rather than an intelligent tutoring system. This type of system would benefit greatly if the knowledge base was augmented with multimedia background information, in the form of text, graphics and video. This could be realized if this background information was in turn available to the authoring system for inclusion in courses and lessons and then executed by the tutoring system. Furthermore, Clancey makes little use of student modelling to help guide instruction, leaving instruction exclusively to human instructors to specify through the authoring facility. This requires considerable effort from the instructor to anticipate all of the possible points of confusion that students may encounter and to provide cases that cover them all.

The pragmatic approach that Clancey has taken provided a good starting point for development of the required tools. However, we needed to overcome some of the limitations of his approach to produce a truly robust system for practical use. A central concern was the knowledge acquisition problem. To alleviate this problem the decision was taken to base the intelligent tutoring system on the ACQUIRE™ expert system environment [3,4] (described below). ACQUIRE™ deals explicitly with the acquisition of domain knowledge from human experts, making the adapted system easier and faster to use by a broad audience.

To extend the ACQUIRE™ expert system environment for intelligent tutoring purposes (ACQUIRE™-ITS) three new modules were developed: the *Tutor*; the *Curriculum Author*; and the *Hypermedia module*. These modules were interfaced to the *Knowledge Acquisition System* and the *Expert*

System Shell that currently make up the commercially available ACQUIRE™ system.

The Curriculum Author module enables the user to construct case-oriented courses and lessons. The Tutor guides the student through these cases until the student can perform like an expert. The system is interfaced to the Hypermedia environment with the potential to enhance communication with the user through text, graphics, video information, and knowledge-based simulation. In its simplest form, the user is able to access linked textual and graphic information located in the knowledge base; enhanced versions of the Hypermedia interface will include animation and video capability. Hypermedia environments are capable of representing object-oriented graphical simulations which in turn could be driven by the Expert System Shell. A similar approach was used effectively by Towne and Munro [5] in the implementation of the Intelligent Maintenance Training System for the bladefold system of helicopters in the US Navy.

The major strength of this approach is a solution to the critical restriction of knowledge-based intelligent tutoring systems, the knowledge acquisition bottleneck. The knowledge acquisition capabilities of ACQUIRE™ are utilized to extract expertise from experts directly, without undergoing the time-consuming process of conventional knowledge acquisition. ACQUIRE™ strengthens the approach by producing highly structured knowledge bases constructed with a strong methodological bias towards thoroughness and completeness of domain knowledge. This reduces the need to add knowledge later which was missed during initial construction. Little additional effort is required to build training applications once the expert knowledge base is constructed; the representation of information inherent in the knowledge base permits automatic creation of case-oriented lessons for a full course curriculum. Finally, the integration with the Hypermedia module created the potential to incorporate knowledge that is more general than the task-oriented knowledge inherent in the knowledge base. This integration of hypermedia makes the training contents more general and broadens the scope of the tutoring that can be offered to the student.

The major components of ACQUIRE™-ITS will be described in the following section.

3. SYSTEM OVERVIEW

3.1 ACQUIRE™

ACQUIRE™ provides an advanced computing environment for domain experts to develop and execute expert system applications. For developing applications, ACQUIRE™ has a *Knowledge Acquisition System*, and an *Expert System Shell*. Developed applications are executed in the *Run-Time Shell*. The Knowledge Acquisition System presents a new paradigm for acquiring knowledge. The main goal of this system is to ease the knowledge acquisition bottleneck by removing the need for a knowledge engineer. This is accomplished by an easy-to-use interface and a structured methodology for developing a knowledge base. The Expert System Shell component provides easy-to-use debugging and what-if facilities that help develop and test knowledge bases. ACQUIRE™ has been successfully used to develop knowledge

bases in domains as diverse as combat intelligence [6], clinical neuropsychology [7] and equipment maintenance [8].

This section will discuss the Knowledge Acquisition System and the Expert System Shell. This will be followed by a discussion of the modules that extend ACQUIRE™ into the intelligent tutor system ACQUIRE™-ITS: the Curriculum Author; the Tutor; and the Hypermedia system.

3.1.1 KNOWLEDGE ACQUISITION SYSTEM

Through the application of research in cognitive psychology a knowledge acquisition methodology has been developed that enables experts to explicate their knowledge in the form of an executable knowledge base [3,4,6]. This methodology is embodied in ACQUIRE™ which forms the cornerstone of the ACQUIRE™-ITS project. It provides the means to build and maintain the knowledge bases required for this project. That is, ACQUIRE™ elicits and structures the knowledge that subsequently forms the content of training courses that are authored and executed through ACQUIRE™-ITS.

In order to acquire knowledge directly from domain experts ACQUIRE™ was based on a knowledge acquisition methodology that divided the knowledge engineering task into steps that are easily and accurately performed. When completed, these steps, result in a highly structured and thorough knowledge base that is ready for execution.

The first step involves eliciting *Objects*, the basic building blocks of the knowledge base. Through a form based editor, a set of objects that define the scope of the knowledge are entered. This step enables the expert to focus on the scope of the problem without being encumbered by issues relating to the structure of the knowledge or the details of the rules. Objects can be any concrete or abstract entity consisting of a name, a *value set* and sometimes a *mapping*. A value set is a set of symbolic values that are associated with the object. For instance, an object called Age Group could have the value set Baby, Child, Pre-teenager, Teenager, Adult, and Senior Citizen. Furthermore, every value set has a value called UNKNOWN which is used when the value for an object is unknown or simply not available. Mappings are used when a symbolic value has a numeric counterpart to convert that numeric quantity to a symbolic value.

With the basic building blocks of the knowledge base defined the expert can concentrate on the high-level structure of the knowledge base. This second step of building an ACQUIRE™ knowledge base involves linking the objects into an *influence network* that serves as a general, conceptual overview of the solution space that will subsequently be further articulated. The influence network simply outlines which object can affect which other objects by linking them when knowledge about the state of one can be informative about the state of another. The expert builds the influence network by using a graphical interface and browsers of objects. To aid the expert in building the influence network a machine learning algorithm is incorporated that takes the network that the expert has produced and restructures it through object substitution and by suggesting the addition of new objects to the network. Since this algorithm recommends restructuring the network, rather than operating on its own, the expert retains full control over whether or not to accept the recommendations. For example,

the algorithm can detect where an intermediate abstract object can be used to reduce the size of the network. If the expert accepts the recommendations of the algorithm, the expert simply names the new object and the network is restructured. After the algorithm has finished processing the network and to the degree that the expert has accepted its recommendations, the influence network will be smaller and more efficient.

The third and final major step of creating an ACQUIRE™ knowledge base involves both the packaging of objects from the influence network into rule definitions, and the assignment of a pattern of object-value pairs to a consequent object-value pair. In these two steps the expert can focus on the details since the object definitions and their influence structure are already provided.

Rules in ACQUIRE™ come in two types: the first being the standard *production rule* if-then construction and the second being *action tables*. A production rule describes an object-value pattern such that if the premise is true then the conclusion is true.

In ACQUIRE™ production rules are best reserved for rules involving arithmetic or mathematical manipulations rather than for experiential knowledge. Production rules have a fundamental weakness in that they require the expert to supply all possible patterns of objects and values to produce a complete knowledge base. This may be problematic since experts may not be aware of all possible patterns resulting in a good chance that they will miss obscure patterns. Also, experts are rarely accurate at producing production rules as there are usually numerous exceptions to any rule [3,4] and there is a tendency to state production rules at a far greater level of generality than one employs in the practice of one's skill.

Action Tables, on the other hand, present the expert with a table of specific object/value patterns. Action Tables guide the expert by supplying specific object/values patterns and all that is required from the expert is to supply the pattern specific conclusions.

Action Tables ease the creation of rules since experts are very good at recognizing patterns [3] enabling the creation of complex rules in a very short time. A major benefit of action tables is their completeness. In production rule systems, completeness is rarely attained, whereas with action tables, every object/value pattern is described and the expert must consider each pattern even if the conclusion is unknown to the expert. If a pattern occurs in which the expert does not know the conclusion, the expert can correspond with other experts in the field and come to some consensus for the pattern. In this way the knowledge base can ultimately reflect what the expert currently knows as well as what the expert is capable of knowing with further experience.

With the ACQUIRE™ knowledge acquisition paradigm, the expert leaves the details of the rules until the knowledge base structure is fully defined. If the expert is using action tables for rules then all that is required from the expert is to match patterns and to "fill in the blanks" producing a complete and accurate knowledge base.

91-15535



030
611
10

3.1.2 EXPERT SYSTEM SHELL

The Expert System Shell component of ACQUIRE™ provides the expert with a means to debug and refine knowledge bases. This component provides facilities such as tracing, stepping and what-if analysis of the knowledge base as a case is being run. Other aids consist of detailed reports, a graphical representation of the fired-rule network, and the ability to inspect the contents of the entire knowledge base.

Before the expert can use the Expert System Shell *case data* has to be entered. Cases can be any real life situation or cases can be made up to test the boundaries of the knowledge bases. Cases consist of object/value pairs. Objects within cases are those objects that can be assigned a value from observation. Case objects are usually only used in the premise of rules since there is generally no need to set these objects to new values. To collect case data, ACQUIRE™ has two features: the first feature, the *Case Form Editor*, is the mechanism used to define the way case data is to be entered; the second feature, the *Case Editor*, is the mechanism used to enter case data. The Case Form Editor allows the expert to tailor input forms to suit the needs of the domain as every domain will have a different input representation for entering the case data. If the data has to be collected at run time then the shell collects the data by prompting the user.

The final aspect of ACQUIRE™ concerns its reporting capabilities. ACQUIRE™ supplies a set of reports that provide information on case execution. These reports range from a very detailed view of the execution to a simple summary of the execution. Additionally, users can build their own customized reports that can explain the results of an execution in plain english.

3.2 ACQUIRE™-ITS

Once a knowledge base has been developed and tested with ACQUIRE™, ACQUIRE™-ITS provides the tools to utilize that knowledge base as the primary course content in an intelligent training application. This system makes intelligent training applications, developed directly by the experts with the domain knowledge, available on affordable computers.

The first major module in ACQUIRE™-ITS, the *Curriculum Author*, lets experts develop a course curriculum composed of lessons using a case-study approach. Students learn by solving actual or hypothetical cases, packaged into lessons by the expert to demonstrate important concepts and their interrelationships. The structure and content of ACQUIRE™ knowledge bases guides the expert in construction of the curriculum.

The second major module, the *Tutor*, monitors student progress during case solution. The Tutor attempts to mold the student's knowledge according to the structure of the ACQUIRE™ expert knowledge base. This is accomplished by identifying incorrect rules, gaps in rule coverage, and poor organization of knowledge, especially an incorrect packaging of objects into rules, which can reveal basic misconceptions about how knowledge should be structured and used. The Tutor selects and presents appropriate cases, from the lessons constructed by the expert, to help the student elucidate the correct knowledge

as it is used in active problem solving. If the student's progress indicates that appropriate knowledge has not been acquired, the Tutor explicitly demonstrates the correct expert knowledge.

ACQUIRE™-ITS incorporates our *Hypermedia System* so that knowledge bases and training sessions can be augmented with background textual information, audio, graphics / simulations and video. Each of these ACQUIRE™-ITS components are described in greater detail in the following sections.

3.2.1 CURRICULUM AUTHOR

ACQUIRE™ produces knowledge bases that organize the knowledge at three different levels from the general to the specific. At the most general level we have the objects influence network which can be graphically displayed as the object graph. Next, we have the rule definitions that specify the objects that are used to determine the values of other objects. This level can be viewed graphically as the rule graph. The most specific level of knowledge organization is the assignment of specific values within each rule definition. This most specific level of knowledge organization can be viewed graphically through fired rule graphs for specific cases. These three levels of knowledge organization provide the architecture, the reasoning paths and ultimately the content to be imparted to the students. This content, and its organization is supplied to the Curriculum Author. As a result the user of the Curriculum Author need not worry about the generation of course content but rather concentrates on the more manageable task of selecting specific aspects of the content/knowledge to package into lessons.

The Curriculum Author provides tools that are designed to exploit the architecture of the knowledge base to facilitate rapid development of the course curriculum. In the Curriculum Author the development of a course curriculum is accomplished through selections from browsers, traversing the various graphs and through access to the entire knowledge base and case base.

Each *curriculum* consists of one or more *lessons*. Each lesson consists of one or more *cases* chosen to demonstrate some area of expertise that the Curriculum Author user wishes to impart to the student(s). Within each case, objects may have one or more associated *questions*.

Questions are created to probe the understanding of the structure and content of the knowledge base. *Object questions* deal with the static elements of the expert domain: nameable entities, their value sets and object interconnections. *Rule questions* deal with inferential aspects of the knowledge-base: situational relationships, and consequential value assignments. The system supplies a range of questions to choose from for each object. Graphical as well as textual depictions of each question are generally available. The user can create specific questions or modify the system supplied questions. For each question the user can create an associated explanation.

Cases highlight a portion of the knowledge embodied in the knowledge base. These cases can be historic cases that have been accumulated through execution of the knowledge base or cases created specifically for training. Each case portrays a situation or task that has occurred or could occur in the domain

of knowledge. A case, when executed through the expert knowledge base can provide a situation or task specific vehicle for apprenticeship learning.

Lessons, in the Curriculum Author, are the tool used to package related tasks, situations or content for presentation to the student(s). Lessons can be as broad or narrow in scope as the user wants. The scope of the lessons is simply a function of the cases selected and the questions assigned to those cases.

Course development proceeds quickly as the Curriculum Author user does not need to attend to the development of the course subject matter. Rather the user simply selects cases, selects questions and assigns those questions to cases within the context of lessons.

3.2.2 TUTOR

Once the curriculum has been completed, the Tutor is ready for use by the student. The Tutor module is used to guide the student through the curriculum following the case-study, apprenticeship learning, approach. This module can be run in one of two modes: *learn* or *explore*. The student will generally run the Tutor in learn mode. Here the Tutor presents a predefined curriculum to the student in the order determined through the Curriculum Author. The student's progress is tracked and compared with the expert knowledge base. In this way, the Tutor can make intelligent decisions about which parts of the curriculum a student has learned and which parts require more work. Reports are available to both the instructor and to the student detailing where progress has or has not been made. Explore mode enables the student to inspect the expert knowledge base and to run cases that have been suggested by the instructor through the inference engine. This mode provides the student with detailed access to the expert reasoning employed in specific situations or tasks.

When the Tutor is invoked in learn mode, the student has the option of selecting a specific lesson to study from or working with the lesson that the Tutor selects. Once a lesson has been selected, the cases for that lesson are executed through the Expert System Shell inference engine. The student is presented with questions and possible answers and is required to select an answer for each question. If the correct answer is selected, the session continues with the next question.

If an incorrect answer is selected, the Tutor decides whether the question should be presented again, whether a different question should be presented to probe the nature of the student's misunderstanding, or whether the answer should be supplied along with an explanation. The decision to present the question again, to present an alternate but related question, or to reveal the answer to the student is based on the student's history of responses for the curriculum and parameters set by the Curriculum Author user. Questions that a student is having serious problems with are reassigned to subsequent cases so that the student's performance can be re-evaluated after the benefit of additional case-study. Once a case has been completed the student has the option of inspecting it and examining it more closely so as to better understand the reasoning process that the expert used.

As each question is presented to the user, it's type is examined to determine what additional information to display. When a

question deals with the influence network or the rule structure, a graphic is displayed that gives the student a visual indication of the information required from them. These graphics are initially incomplete and are completed as the student answers the question.

When the Tutor is invoked in explore mode, the student is able to inspect any part of the expert knowledge base. The instructor provides case input forms that the student can use to create cases that can be executed. These case input forms are made available for any lesson that the student has completed. Explore mode enables the student to study and review the subject matter contained in a lesson through examination of the experts reasoning on specific cases. The student benefits from access to the case input information, the variety of case execution reports, and from access to the expert knowledge base.

The two modes of operation in the Tutor can be seen as handling complimentary aspects of the instruction. Explore mode handles initial study for a particular lesson and any review that may be necessary. Learn mode concentrates on testing the breadth of the student's understanding, analysing student difficulties and directing the student towards aspects of the knowledge base that, once understood, alleviates particular misunderstandings.

3.2.3 THE HYPERMEDIA SYSTEM

The Hypermedia system provides the ACQUIRE™-ITS user with an environment to augment the case-study approach with text, graphics and video. This is accomplished through the Curriculum Author where hyperbases can be created to supply introductory material, student instructions, background information, explanatory information, etc. Each hyperbase is implemented as a series of related groups of information that are linked together so that a student can easily move from one group of information to another.

The Hypermedia system, including it's various *hyperbases*, is made available to the Tutor in an inspect only mode. Through the Curriculum Author hyperbases can be linked to each curriculum, each lesson and each case. In addition, hyperbases can be linked to explore mode and to the Tutor's explanation facility. All hyperbases have a table of contents that enables the student to rapidly move directly to a specific piece of information as well as enabling the student to assess what information has already been reviewed and what information has not.

4. CURRENT STATUS AND FUTURE DIRECTIONS

ACQUIRE™-ITS is currently operating in a prototype stage of development. Development has proceeded in the C Programming Language. UNIX has served as the operating system for development while DOS has served as the operating system for delivery of the prototype. A character based windowing system (Vermont Views) has been used for the user interface throughout the early development work. The user interface has been converted to X windows and will in turn be converted to Microsoft Windows.

At project onset it was determined that concurrent development of an industrial application would provide end-user feedback through the entire development cycle. MacMillan Bloedel Research and the Specialtyboard Division of MacMillan Bloedel's K3 Particle Board operation were subcontracted to develop a training application. The application chosen was the operation and maintenance of the coating line - a process used to apply finishing coats to K3 particle board [8]. The application was chosen for two reasons: the process was sufficiently complex to provide a reasonable test of ACQUIRE™-ITS (involving a series of ovens, sanders and roll coaters each having their own peculiar problems); and training had been identified as a priority for this process.

MacMillan Bloedel staff built three knowledge bases in ACQUIRE™ while the early development work on ACQUIRE™-ITS was underway. The experts involved had no special training in expert system technology. Upon completion of these knowledge bases a prototype of ACQUIRE™-ITS was delivered to MacMillan Bloedel and work commenced on curriculum development for the three knowledge bases. Again, the experts had no previous experience with instructional technology. Curricula have now been developed and executed to the satisfaction of the MacMillan Bloedel experts. Testing with students will take place in the near future.

This concurrent application development has driven a number of refinements that will be carried out over the next six months. To further expedite curriculum development facilities are being designed to provide automatic lesson generation, automatic lesson completion and automatic case generation.

Automatic lesson generation will analyse the knowledge base and suggest lessons. Once the lessons have been accepted the system will assemble cases to cover the subject matter.

Automatic lesson completion will take a partially specified lesson and complete the specification with additional questions, answers and new cases. Automatic case generation will produce cases that meet instructional guidelines specified by the Curriculum Author user. The user will be able to tell the Curriculum Author how the case is to be generated by indicating what part or parts of the knowledge base to use. In circumstances where there is an area of the knowledge base that is complex or obscure, it may be instructive to have the case include this area. These cases will then be used in lessons and in explore mode.

The role of the Hypermedia System will also be expanded. During case execution the hypermedia module will be employed to drive graphical simulations. Through explore mode this enhancement will give the student the opportunity to obtain a graphical perspective on the changes that will occur in a given case when input values are changed.

5. DISCUSSION

ACQUIRE™-ITS is a domain independent intelligent tutoring system designed to enable computer users to develop skill oriented training courses directly. The choice to base ACQUIRE™-ITS on the ACQUIRE™ system realized several crucial advantages. The knowledge acquisition bottleneck was alleviated and well structured knowledge bases were available for exploitation in training. Course authoring was kept

independent of knowledge base development consequently facilitating modification of each. Development of the Hypermedia system along with the conversion from character based windows to graphical based windows has extended the range of instructional tools available to the Curriculum Author and the Tutor.

The concurrent application development work with MacMillan Bloedel has confirmed that the tools provide a practical means of utilizing existing industrial personnel to construct complex, automated training courses. The industrial feedback and experience gained through development of this application should result in enhanced intelligent tutoring tools that will generalize to a broad range of industrial and governmental training needs.

6. REFERENCES

1. Woolf, B., "Representing complex knowledge in an intelligent machine tutor". *Computational Intelligence*, 3, 45-55, 1988.
2. Clancey, W.J., & Joerger, K., "A practical authoring shell for apprenticeship learning". *Proceedings of Intelligent Tutoring Systems*, Montreal, 1988.
3. Smith, B.J., Schaefer, B.A., & Morrison, I.R., "Investigation of automated knowledge acquisition". Technical Report #31947-7-0003/01-sz, National Research Council of Canada, Ottawa, Ontario, 1988.
4. Schaefer, B.A., "Knowledge acquisition research at Acquired Intelligence Inc." *Canadian Artificial Intelligence*, Vol. 15, 27-28, 1988.
5. Towne, D.M., & Munro, A., "The intelligent maintenance training system". In J. Psotka, L.D. Massey & S.A. Mutter (Eds.), *Intelligent Tutoring Systems: Lessons Learned*, Hillsdale, NJ: Lawrence Erlbaum Associates, 479-528, 1988.
6. Morrison, I.R., & Sihota, P., "Knowledge acquisition for combat intelligence". Technical Report for Command and Control Division, Defence Research Establishment Valcartier, February, 1990.
7. Russell, D.L., Smith, B.J., Schaefer, B.A., & Morrison, I.R., "The application of expert system technology to clinical psychology." *Canadian Psychology*, Vol. 28, 231, 1987.
8. Sihota, P., Yap, S., Schaefer, B.A., & Wagstaff, R., "Application of ACQUIRE™, an automated knowledge acquisition system, to industrial maintenance and maintenance training". *Proceedings of IEEE Pacific Rim Conference on Communications, Computers and Signal Processing*, May, 1991.

ACQUIRE™ is a trademark of Acquired Intelligence Inc.; MS-DOS is a trademark of Microsoft Corporation; IBM PC is a trademark of International Business Machines; UNIX is a trademark of AT&T Bell Laboratories; Vermont Views is a trademark of Vermont Creative Software.



COMPONENT PART NOTICE

THIS PAPER IS A COMPONENT PART OF THE FOLLOWING COMPILATION REPORT:

(TITLE): Conference Proceedings of Machine Intelligence
for Aerospace Electronic Systems Held in Lisbon, Portugal on
13-16 May 1991 (L'Intelligence Artificielle dans les Systemes
Electroniques Aero spatiaux)

(SOURCE): Advisory Group for
Aeronautical Research and Development, Neuilly-sur-Seine
(France)

TO ORDER THE COMPLETE COMPILATION REPORT USE AD-A242025

THE COMPONENT PART IS PROVIDED HERE TO ALLOW USERS ACCESS TO INDIVIDUALLY
AUTHORED SECTIONS OF PROCEEDINGS, ANNALS, SYMPOSIA, ETC. HOWEVER, THE
COMPONENT SHOULD BE CONSIDERED WITHIN THE CONTEXT OF THE OVERALL COMPILATION
REPORT AND NOT AS A STAND-ALONE TECHNICAL REPORT.

THE FOLLOWING COMPONENT PART NUMBERS COMPRISE THE COMPILATION REPORT:

AD#: _____

TITLE: _____

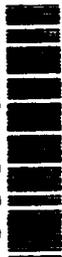
AD- P006353

DTIC
ELECTE
NOV 13 1991
S D D

This document has been approved
for public release and sale; its
distribution is unlimited.

Accession For	
NTIS CRA&I	<input checked="" type="checkbox"/>
DTIC TAB	<input type="checkbox"/>
Unannounced	<input type="checkbox"/>
Justification	
By _____	
Distribution/	
Availability Codes	
Dist	Avail and/or Special
A-1	21

AD-P006 353



Reasoning with Uncertain and Incomplete Information
in Aerospace Applications¹

by

J.C. Donker
National Aerospace Laboratory NLR
Anthony Fokkerweg 2
1059 CM Amsterdam
The Netherlands

SUMMARY

In many real-life application areas such as aerospace, decisions have to be taken based on imperfect knowledge. If decision makers are to be supported by computer systems, it is desirable that this type of knowledge can be represented. In the past years, new methods have been developed to represent various kinds of imperfectness, such as incompleteness, inexactness or uncertainty.

Since 1987, NLR cooperates with the Theoretical Informatics Group of Delft Technical University to study ways in which methods to represent and reason with uncertain or incomplete information can be developed which are both theoretically well-founded and practically applicable. NLR efforts are directed towards problems expected in practical use of these methods. In 1990, we investigated the applicability of the Dempster-Shafer theory. We modeled parts of the initiation and identification problems in multi-radar tracking. The application will be described in this paper. It will be shown that the Dempster-Shafer theory promises improvements over the Bayesian approach, but also that the latter currently is a more advanced theory than the former.

1. INTRODUCTION

At NLR, the feasibility of machine intelligence techniques has been demonstrated for a number of civil and military aerospace applications. Human knowledge in combination with other types of information has been captured and represented. From this experience we have concluded that the methods used to reason with information that is uncertain or incomplete are less adequate for larger applications.

Ideally, we would want a method to fulfil the following requirements:

- universality of representation: all types of imperfectness can be modeled
- adequacy of representation: the information is accurately modeled
- verifiability by having well-defined formal properties such as soundness and completeness
- computational efficiency
- robustness
- compliance with human reasoning
- ease of understanding/manipulation.

Until now, no such methods exist, and it may well be that no such method can ever be found, because of the variety of ways in which our information can be imperfect.

After an investigation of existing methods (Ref. 1) it turned out that serious effort was needed to solve this lack of good methods required for our applications.

For the past four years, NLR has cooperated with the Theoretical Informatics Group of Delft Technical University to study methods for reasoning with uncertain or incomplete knowledge. Main subjects studied were reasoning with fuzzy logic (Ref. 2), and non-monotonic logics and certainty measures that represent our ignorance about the world (Ref. 3).

This paper has been presented at the AGARD symposium on Machine Intelligence for Aerospace Electronic Systems, Lisbon, May 16, 1991.

91 1113 081

91-15534



Also, the Dempster-Shafer theory was investigated. The theory was applied to two subproblems in multi-sensor fusion, more specifically in multi-radar tracking. This application will be described in the paper. It will be shown that the Dempster-Shafer theory promises improvements over the Bayesian approach, but also that the latter currently is a much more advanced theory than the former.

In order to provide the reader with a more general framework, the relation between the work described in this article and the subject of the conference is described in section 2, and the notions used in reasoning with incomplete or uncertain information are presented in section 3, as well as a short overview of existing methods.

2. MULTI-SENSOR FUSION

Observation of the world with multiple sensors is becoming more and more common in a number of application areas NLR is involved in, such as integrated modular avionics, command and control, multi-target tracking, advanced robotics and air traffic control. The issues involved in integrating multiple sensors into the operation of a system vary from low-level sensor fusion issues to high-level interpretation of the meaning of the aggregated information. Bogler (Ref. 4) mentions four requirements for multi-sensor fusion methods applied to target identification:

- Sensor modeling: Each sensor is likely to be contributing information in its own sensor-specific level of abstraction. The method must be capable of accurately modelling the information provided by each sensor source.
- Fusion and display: The method must be capable of fusing the information provided, of computing a statistical measure for the resultant identification quality, and of accurately displaying the information to the user. In case of the target identification, the displayed information must be hierarchically classified in terms of target nature (friend/foe), class (fighter/bomber/missile), and target type.
- Conflict resolution: The method must be capable of resolving any potential sensor conflicts.
- Wide range of applicability: Lastly the design must be flexible to changing scenarios, i.e. it must not be predicated upon assumptions about the operational scenario which, to a large degree, is unknown a priori."

It is not hard to see that (1) these requirements also hold for the other application areas mentioned above and (2) that the requirements form a subset of the requirements we have for reasoning with incomplete and uncertain information in general.

For these reasons, we consider methods for reasoning with imperfect information relevant for many machine intelligence applications in aerospace electronic systems. As an application-oriented test case, we have studied Dempster-Shafer theory for multi-radar tracking.

3. REASONING WITH INCOMPLETE OR UNCERTAIN INFORMATION

Humans are capable of reasoning when the available information is imperfect. This is the case if the information has one or more of the following characteristics:

- the information is incomplete: not everything is known or modeled which pertains to a problem;
- the information is uncertain: the information is not precisely specified. It does not pertain to the information set as a whole, as does incompleteness, but to the contents of the information;
- the information is inexact: the information is vaguely or ambiguously specified. This notion also relates to the contents of the information.

Various approaches have been developed to formalize reasoning with imperfect information. Some of them are purely symbolic, others make use of

numbers; some are close to formal logic, others are much less formalized; some are focussed on reasoning with exceptions, others tackle the problem of belief revision. There is as yet no consensus as to what constitutes the best method for dealing with imperfect information. A number of methods will be mentioned in this section, to give the reader an impression and a starting point for further reference.

3.1 Early systems

The two best-known early numerical systems that deal with imperfect information in knowledge-based systems are the MYCIN system with the certainty factor model (Ref. 5), and the PROSPECTOR system incorporating a subjective Bayesian method (Ref. 6).

The certainty factor model is used by many rule-based knowledge-based systems, because of its intuitive appeal and its computational efficiency. As many researchers have explained (e.g. Ref. 7), the model is not well-founded from a mathematical point of view. Our experience has learned that the model works reasonably well for problems in which the expert can tune the certainty factors. This is only possible if the knowledge base is not too complicated.

The subjective Bayesian method will be described in section 3.3.

Since research on reasoning with imperfect information has not yet yielded alternative models which are both computationally feasible and semantically clear, the certainty factor model in particular still is in wide-spread use.

3.2 Logic-based systems

Logic-based systems based on propositional logic or predicate logic are well suited to model information of which truth or falsity can be established with certainty. These classical logics cannot deal very well with imperfect knowledge, however. As it remains to be attractive to reason based on a formal system with which properties such as soundness and completeness can be proved, a number of extensions have been defined.

A classical logic is a formal system composed of:

- (i) a language
- (ii) a set of axioms
- (iii) inference rules, and a description of the manner in which these axioms and rules are to be used in order to yield theorems.

Moreover, a semantics is associated with the formal system, to define meaning. Generally, it is defined by means of a function which assigns a truth value to every formula.

The classical logic can be extended in several ways. The following excerpt from a special issue of the International Journal of Intelligent Systems (Ref. 8) shows the large diversity which has mainly come into existence in the 1980s:

- The language is extended, by introducing predicates (supposition-based logic) or operators ("possible", "necessary" in modal logic, "I believe" in autoepistemic logic, operators indicating the past or the future in temporal logic,...). It is equally possible to use an implication different from the material one (conditional logic), to introduce quantifiers other than "For all..." and "There exists..." or to take into account the vague character of predicates (fuzzy logic) or of quantifiers. Finally, uncertainty degrees can be assigned to the propositions (possibilistic logic for instance).
- New axioma schemata are added (especially in the circumscription approach).
- New notions of inference are defined, which permit the formulation of answers of the type "P is true in a certain vision of the world..." (in default logic, we speak of extensions). A new notion of semantics may correspond to it (maximal (minimal) model in the non-monotonic logics, universe of possible worlds in modal logic for example). We no longer impose certain properties of classical deductive systems,

such as inference monotonicity in particular (nonmonotonic logics). We do not limit ourselves to the classical rule of modus ponens; we authorize new inference rules (necessitation rule in modal logic for example)."

A large number of these logics are described and compared in the journal; Turner (Ref. 9) also describes logics and their relevance to Artificial Intelligence.

Roos (Ref. 3) has developed a new non-monotonic preference logic, which combines the advantages of existing approaches while avoiding some of their disadvantages. Furthermore, a deduction process in which reason maintenance is integrated is defined for the logic (see also section 3.5).

Most of the non-classical logics are still under study; only a small number of implementations exist.

3.3 Probabilistic systems

With respect to numerical approaches to reasoning with uncertainty, much experience is already available. Especially approaches based on formal probability theory have been in use much longer than any other of the approaches described in this paper.

A hectic debate is still going on, however, in the philosophical, mathematical and artificial intelligence communities with respect to the meaning of the word probability.

Since the turn of the century, three views have dominated (Ref. 10):

- Objectivistic or frequentistic: Probability measures the ratio of occurrences to observations for a proposition, in the long run.
- Personalistic, subjectivistic, or judgmental: Probability measures the confidence that an individual has about a particular proposition's truth.
- Necessary or logical: Probability measures the extent to which one set of propositions, out of logical necessity and apart from human opinion, confirms the truth of another. Probability thus measures inferential soundness or provability. Proponents generally view this view as an extension of logic.

Pearl, one of the proponents of reasoning with probabilities (instead of logics, other non-numerical or heuristic approaches), gives an excellent overview of the development of probability theory (Ref. 11). The following is mainly based on his description.

The Italian mathematician Cardano (1501-1576) is believed to be the first to have formulated the notion of probability in gambling in terms of the number of distinguishable ways that events may occur. His "objective" view of probability developed into a mathematical theory of combinatorics, due to Fermat, Pascal, Huygens, Bernoulli, De Moivre and Laplace; it was Poisson who defined probability as a limit of a long-run relative frequency. Borel and Kolmogorov developed the modern axiomatic foundations of mathematical probability, in which the concept of random experiment plays a crucial role: of such an experiment, the outcome cannot be predicted with certainty, but the experiment is of such a nature that the collection of every possible outcome can be described prior to its performance; furthermore, the experiment can be repeated under the same conditions.

In parallel to these mathematical developments, an alternative view of probability came into being with Bernoulli's suggestion that probability is a degree of confidence that an individual attaches to an uncertain event. This concept, aided by Bayes' rule (Ref. 12) was evident in the writings of Laplace and De Morgan and later in the work of Keynes (Ref. 13) and Jeffreys (Ref. 14). It was not until the 1950s with the development of statistical decision theory, that Bayesian methods gained their current momentum. The defining attributes of the Bayesian school are (1) willingness to accept subjective opinions as an expedient substitute of raw data and (2) adherence to Bayes conditionalization as the primary mechanism for updating belief in light of new information.

The heart of the Bayesian techniques lies in the celebrated inversion formula:

$$P(H|e) = \frac{P(e|H)p(H)}{P(e)},$$

which states that the belief we accord a hypothesis H upon obtaining evidence e can be computed by multiplying our previous belief $P(H)$ by the likelihood $P(e|H)$ that e will materialize if H is true. $P(H|e)$ is sometimes called the posterior probability, and $P(H)$ is called the prior probability. Whereas a formal mathematician might dismiss this equation as a tautology stemming from the definition of conditional probabilities, the Bayesian subjectivist regards the equation as a normative rule for updating beliefs in response to evidence. The conditional probabilities then are primitives of the language and faithful translations of the English expression "... , given that I know A ."

PROSPECTOR is one of the few expert systems that relies on Bayesian decision theory and Bayes' rule. Instead of using the prior and conditional probabilities directly, it uses odds-likelihood functions. The advantage that likelihood methods have over prior and conditional probabilities is that people feel far more comfortable providing likelihood ratios. Since the original prior and conditional probabilities can be recovered from the likelihood ratios, this approach can use Bayes' rule to propagate evidence. To use the PROSPECTOR approach, conditional independence of evidence is required under both a hypothesis and its negation. Since these assumptions are rarely satisfied, useability of the approach is severely restricted.

The Dempster-Shafer theory is another example of a probabilistic approach. Because of its importance to this article, it is separately described in section 4.

3.4 Endorsements

A qualitative approach to reasoning with imperfect information is the theory of endorsements (Ref. 15).

In most real world settings, the "strength of evidence" actually is a summary of several factors. Numerical approaches summarize supporting and opposing evidence. Cohen argues that an intelligent reasoner usually discriminates among these factors. The theory of endorsements is centred around the idea of dealing with reasons for believing or disbelieving a hypothesis. By making explicit the knowledge about uncertainty and evidence, endorsements show how to reason with the knowledge directly.

3.5 Reason maintenance systems

The same ideas which have led to the endorsements approach can be found in Truth Maintenance Systems, or more generally, reason maintenance systems. The idea behind these systems is that justifications or assumptions for reasoning steps shall be kept, separate from the reasoning system. Reason maintenance is especially important in the case of non-monotonic reasoning. If a contradiction occurs between consequences of decisions taken in an earlier stage and consequences derivable from new information as it comes available, the reasoning system can consult the reason maintenance system to provide ways by which the system can return to a consistent state. For this reason, these systems are sometimes called belief revision systems. Well-known reason maintenance systems are the Truth Maintenance System developed by Doyle (Ref. 16) and the Assumption-based Truth Maintenance System of de Kleer (Refs. 17,18,19). In general, implementation of these systems is not easy; memory and performance problems are not easy to avoid.

3.6 Fuzzy systems and possibility theory

Fuzzy reasoning concerns are distinct from those of the formalisms considered so far. It deals with vague notions in information people use, such as "Visibility is rather low today". Fuzzy logic can be considered as a synthesis of fuzzy set theory and mathematical logic. It uses fuzzy sets instead of propositions and predicates. These fuzzy sets are used to denote linguistic notions. They are different because they have implicit meaning. Usually, this meaning is not taken into account explicitly, because not doing so makes it easy to transform fuzzy logic into a kind of multivalued logic. Hellendoorn has proposed an alternative approach using shifted hedges, in which the meaning of the fuzzy sets is taken into account (Ref. 2).

Since the publication of Zadeh's paper "Fuzzy sets as a basis for a Theory of Possibility" (Ref. 20), possibility theory has become an important keyword in fuzzy set theory. An important thesis in that paper is that when one's main concern is with the meaning of the information rather than with its measure, the proper framework is possibilistic rather than probabilistic in nature. In this way a fuzzy variable (concerning a meaning) is associated with a possibility distribution in much the same way as a random variable (concerning a measure) is associated with a probability distribution (Ref. 2).

First implementations in this category exist primarily for fuzzy control applications.

4. THE DEMPSTER-SHAFER THEORY

In his work on statistical inference (Ref. 21,22), Dempster criticised Bayesian inference for its insistence that we assess a subjective prior distribution, even if we do not know much about it. He proposed an alternative approach which avoids the need for these prior distributions. A consequence of this approach is that the notion of point probability is replaced by upper and lower probabilities. Shafer has built a theory of belief functions based on the earlier work of Dempster. The idea behind the Dempster-Shafer theory of belief functions is described in a paper by Shafer (Ref. 23):

" The theory of belief functions provides two basic tools to help us make probability judgments: a metaphor that can help us organize probability judgments based on a single item of evidence, and a formal rule for combining probability judgments based on distinct and independent items of evidence.

Here is the metaphor for organizing our probability judgments. We think of our belief (or our "probability", if you will) as a whole and imagine committing parts of that whole ("probability masses") to various propositions. The amount we commit represents a judgment as to the strength of the evidence that specifically favours that proposition. It is not required that belief not committed to a given proposition should be committed to its negation, nor that belief committed to a given proposition should be committed more specifically."

Shafer has worked out this metaphor into a mathematical theory of evidence (Ref. 24). The theory is most easily introduced in the case where we are concerned with the true value of one quantity. If we denote the quantity by θ , and the set of its possible values by Θ , then the propositions of interest are precisely those of the form "The true value of θ is in T ", where T is a subset of Θ . Thus the propositions of interest are in a one-to-one correspondence with the subsets of Θ , and the set of all propositions of interest corresponds to the set of all subsets of Θ , which is denoted by the symbol 2^Θ . The same formalism can be extended to situations where θ is an arbitrary parameter that takes possibly non-numerical values. Shafer calls the set Θ the **frame of discernment**. In order to work out the intuitive picture wherein a portion of belief can be committed to a proposition but need not be committed either to it or its negation, a portion of belief

committed to one proposition is thereby committed to any other proposition it implies. In terms of a frame of discernment, this means that a portion of belief committed to one subset is also committed to any subset containing it. So of the total belief committed to a given subset A of a frame Θ , some may also be committed to one or more proper subsets of A, while the rest will be committed exactly to A, i.e. to A and to no smaller subset.

In the following sections, the mathematical formulation of the Dempster-Shafer theory is presented. In section 4.1, belief functions are defined; in section 4.2, Dempster's rule for combination of beliefs is presented.

4.1 Belief functions

If Θ is a frame of discernment, then a function $m: 2^\Theta \rightarrow [0,1]$ is called a **basic probability assignment** whenever

$$(1) \quad m(\emptyset) = 0$$

and

$$(2) \quad \sum_{A \subseteq \Theta} m(A) = 1$$

The quantity $m(A)$ is called A's **basic probability number**, and is understood to be the measure of belief that is committed exactly to A. To obtain the measure of total belief in A, one must add to $m(A)$ the quantities $m(B)$ for all proper subsets B of A:

$$\text{Bel}(A) = \sum_{B \subseteq A} m(B)$$

A function $\text{Bel}: 2^\Theta \rightarrow [0,1]$ is called a **belief function** over Θ if it is given by this formula for some basic probability assignment $m: 2^\Theta \rightarrow [0,1]$.

Remark.

The class of belief functions can be characterized without reference to basic probability assignments:

If Θ is a frame of discernment, then a function $\text{Bel}: 2^\Theta \rightarrow [0,1]$ is a belief function if and only if it satisfies the following conditions:

- (1) $\text{Bel}(\emptyset) = 0$
- (2) $\text{Bel}(\Theta) = 1$
- (3) For all $n > 0$ and for each collection of subsets A_1, \dots, A_n of Θ :

$$\text{Bel}(A_1 \vee A_2 \dots \vee A_n) \geq \sum_{I \subseteq \{1, \dots, n\}} (-1)^{|I|+1} \text{Bel}\left(\bigcap_{i \in I} A_i\right)$$

Furthermore, the basic probability assignment that produces a given belief function is unique and can be recovered from the belief function:

Suppose $\text{Bel}: 2^\Theta \rightarrow [0,1]$ is the belief function given by the basic probability assignment $m: 2^\Theta \rightarrow [0,1]$. Then,

$$m(A) = \sum_{B \subseteq A} (-1)^{|A-B|} \text{Bel}(B)$$

for all $A \subseteq \Theta$.

End of remark

Bel(A) does not reflect to what extent one doubts A -i.e. to what extent one believes its negation \bar{A} . A fuller description consists of the degree of belief Bel(A) together with the degree of doubt:

$$\text{Dou}(A) = \text{Bel}(\bar{A}).$$

The quantity

$$P'(A) = 1 - \text{Dou}(A)$$

expresses the extent to which one fails to doubt A -, i.e. the extent to which one finds A credible or plausible. Shafer calls this function the upper probability of A.

The significance of the belief and plausibility measures is based on the fact that it can be shown (Ref. 21) that they provide an upper and a lower bound on the probability of B, that is,

$$\text{Bel}(B) \leq \text{Prob}(B) \leq \text{Pl}(B).$$

Since in a probabilistic environment the best information we can have is knowledge of the probability of each set, the smaller $\text{Pl}(B) - \text{Bel}(B)$ the more we know about the situation.

4.2 Dempster's rule of combination

Combination of distinct bodies of belief is achieved via in Dempster's rule of combination. Shafer writes (Ref. 24): "Given several belief functions over the same frame of discernment but based on distinct bodies of evidence, Dempster's rule of combination enables us to compute their orthogonal sum, a new belief function based on the combined evidence. "

In mathematical form, the rule is stated as follows:

Let Bel and Bel' be belief functions induced by the basic probability assignments m and m', respectively. If $\sum (m(A_i) \cdot m'(B_j) \mid A_i \cap B_j \neq \emptyset) = 0$, then Bel and Bel' are called not combinable. Otherwise, $\text{Bel} \oplus \text{Bel}'$, the combination of Bel and Bel' by Dempster's rule, is the belief function induced by $m \oplus m'$, where

$$m \oplus m'(A) = \frac{\sum_{A_i \cap B_j = A} m(A_i) \cdot m'(B_j)}{\sum_{A_i \cap B_j \neq \emptyset} m(A_i) \cdot m'(B_j)},$$

if $A \neq \emptyset$ and $m \oplus m'(\emptyset) = 0$.

4.3 Evidential reasoning with Dempster-Shafer theory

There are two major ways to think about the Dempster-Shafer theory. The first one described above is in terms of subjective probabilities, the second one is in terms of evidence. To this end, Shafer introduces **simple support functions** to denote bodies of evidence whose effect is to support the subset A to a degree s:

A belief function $\text{Bel}: 2^\Theta \rightarrow [0,1]$ is called a **simple support function** if there exists a non-empty subset A of Θ and a number s, $0 \leq s \leq 1$, such that

$$\text{Bel}(B) = \begin{cases} 0 & \text{if } B \text{ does not contain } A \\ s & \text{if } B \text{ contains } A \text{ but } B \neq \Theta \\ 1 & \text{if } B = \Theta. \end{cases}$$

There are other types of support functions, which will not be discussed here. The notion of support functions is important, because they seem to constitute the subclass of belief functions appropriate for the representation of evidence.

To represent complete ignorance, i.e. we do not have any evidence at all, we commit all our belief to the frame of discernment - that is, we assign $m(\Theta)=1$, and $m(A)=0$ for every proper subset of Θ .

If we do know something more, an adequate summary of the impact of evidence on a particular proposition A must include at least two items of information: a report on how well A is supported and a report on how well its negation \bar{A} is supported. Since a proposition is plausible in light of the evidence to the extent that evidence does not support its negation, these two items of information can be conveyed by reporting the proposition's degree of support, $S(A)$, together with its degree of plausibility:

$$Pl(A) = 1 - S(\bar{A})$$

The relations between the subjective terms and the evidential terms is given in table 1 (Ref. 24)

Table 1.
Subjective versus evidential vocabulary for Dempster-Shafer theory

Subjective vocabulary		Evidential vocabulary	
Degree of belief	$Bel(A)$	Degree of support	$S(A)$
Degree of doubt	$Dou(A)=Bel(\bar{A})$	Degree of dubiety	$Dub(A)=S(\bar{A})$
Upper probability	$P'(A)=1-Bel(\bar{A})$	Degree of plausibility	$Pl(A)=1-S(\bar{A})$

5. NLR'S BAYESIAN MULTI-SENSOR TRACKING SYSTEM

At the National Aerospace Laboratory NLR, a multi-sensor tracking system has been developed for both civil and military applications. The task of the tracking system is to provide state information (tracks) about flying objects in the area under consideration. This area is covered by one or more sensors, possibly of different type, which provide data about the environment to the tracking system. The data are received by the most important module of the tracking system, the track continuation module, which updates its tracks based on these data. Data which can be associated with existing tracks are kept; all other measurements are handed over to the track initiation module. Each sensor system has its own track initiation module responsible for initiation of tracks for flying objects in its area of observation. Once a track has been initiated, it is immediately handed over to the continuation system.

Compared to α - β , Kalman-based and state of the art tracking systems, the NLR tracking system provides better track continuity, more accurate expectations of position and velocity (especially in case of sudden manoeuvres), a lower sensitivity for measurement errors, more complete additional information in the form of probabilities of modes of flight (turns, accelerations and straight modes) and consistent estimates of its own accuracy (Ref. 25) The computational power required is far less than for other, recently developed high-performance algorithms for tracking manoeuvring aircraft (Ref. 26). It has been demonstrated that the tracking system will run in real time on presently available multi-processor systems (Ref. 27).

In many parts of the tracking system, Bayesian methods have proved to be effective. The track continuation module for instance consists of a combination of those approximate Bayesian methods that proved to be the most efficient for the main problems of track continuation: Extended Kalman filtering for non-linear dynamics, Probabilistic Data Association for unassociated measurements and Interacting Multiple-Model filtering for sudden manoeuvres.

Other parts of the tracking system offer opportunities to evaluate methods initially representing ignorance, and later representing new information as it becomes available. The following aspects of the multi-sensor tracking problem were interesting from an uncertainty/incompleteness point of view:

- reflection
- assignment of A codes
- track initiation
- track association
- object identification.

Two of these were chosen to gain experience with the Dempster-Shafer approach, viz. the identification problem and the track initiation problem. These will be described in more detail in the next section.

Other methods for reasoning with imperfect information possibly offer better solutions for the other aspects mentioned above. In particular, non-monotonic reasoning seems to be an important area to be investigated.

6. DEMPSTER-SHAFER THEORY FOR MULTI-RADAR TRACKING

The track initiation problem and the identification problem have been chosen from an initial list of possible additions to the multi-sensor tracking system. These fall into three categories:

- solutions for problems not already captured in some way by the tracking system;
- solutions to problems which have been modeled in the tracking system using Bayesian methods, where Dempster-Shafer theory might reduce complexity;
- solutions to provide increased accuracy.

The identification problem falls into the first category, the initiation problem is an element of the second category.

6.1 Track initiation

The track initiation problem deals with the initiation of tracks in the presence of more than one flying object. After assessment of a number of possible situations (Ref. 28), the following setup has been chosen: A number of n radars, located at physically different locations p_j , provide measurements of an object O .

To model this problem in the Dempster-Shafer way, three steps must be taken:

- reduction of measurements at approximately the same time to exactly the same instant in time;
- assignment of belief to the measurements;
- combination of the belief functions of all measurements.

Our modelling of the initiation problem has concentrated on the definition of a basic probability assignment in such a way that combination of belief could be performed in a natural way.

As the first step does not detract from our model, we will skip it here for simplicity purposes. The problem is then formulated as follows:

Suppose that a group of q radars is located at p_1 and a group of $r-q$ at location p_2 . We assume that all radars provide measurements at times t_i . At time t_i , r measurements have been done (each radar has provided one measurement). The situation is depicted in figure 1. With x , we denote the measurements by radars at location p_1 , and with $+$, the measurements from radars at location p_2 . The measurements of radar i are denoted by m_i .

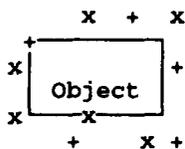


Figure 1. Location of radars

Direct combination of measurements in the standard Dempster-Shafer way would give:

$$m_i \cap m_j = \begin{cases} \emptyset & \text{if } m_i \neq m_j, i \in \{1, \dots, q\} \text{ and } j \in \{q+1, \dots, r\} \\ m_i & \text{if } m_i = m_j, i \in \{1, \dots, q\} \text{ and } j \in \{q+1, \dots, r\}. \end{cases}$$

The situation $m_i \cap m_j$ will practically never occur, as that would mean that the measurement from radar i is identical to the measurement of radar j . Therefore, the straight forward interpretation does not provide a realistic model.

To solve this, we construct for each measurement i an area around the measurement. The idea is that we now will assign belief to the idea that given the measurement i , the object is somewhere within the area. We take the area to be a sphere b_i with centre m_i and radius r_i , and we will use four radars: two identical ones at location p_1 and two identical ones at location p_2 .

Suppose that at a certain instant in time, the radars provide the following measurements (+, x are the measurements from the radars at locations p_1 and p_2 respectively):

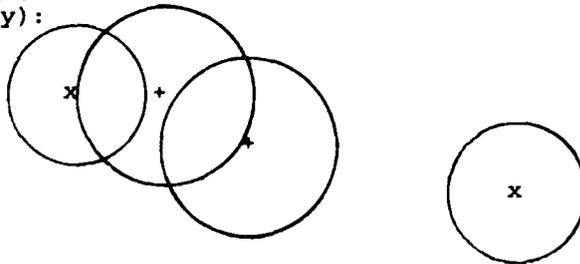


Figure 2. Measurements and spheres for all four radars

To assign belief functions, we first look at the measurements from the radars located at p_1 .

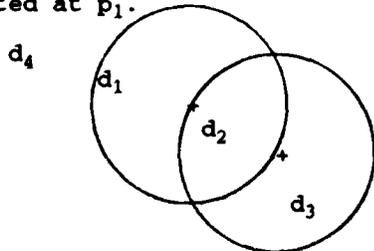


Figure 3. Measurements and spheres for the two radars at location p_1 .

Four subareas can be distinguished, viz.:

- d_1 : the sphere around the first measurement
- d_2 : the intersection between the two spheres
- d_3 : the sphere around the second measurement.
- d_4 : the range of the radars outside the spheres.

To these areas, a basic probability assignment (bpa) will be made. This assignment will in practice be dependent on the accuracy of the radar, the weather conditions, the accuracy of the measurement itself, statistics, etc. We abstract from the precise form of the bpa. Suppose we assign some belief to the idea that the object is located within the sphere b_1 and that we assign the rest of our belief to all points within the range of the radar. We then have:

$$m(b_1) = y_1$$

$$m(\Theta) = 1 - y_1, \text{ with } \Theta \text{ all possible spheres with centre within range } R_{p_1} \text{ of radar } i.$$

As can be seen from the definition of a simple support function (SSF) in section 4.3, the belief function Bel defined by this bpa is a simple support function.

Combination of the two SSFs is depicted in figure 4:

Θ_1	d_1	Θ
b_2	$b_1 \cap b_2$	d_3
	b_1	Θ_1

Figure 4. Combination of two SSFs into the belief function SSFP1.

$$m(d_1) = y_1 \cdot (1 - y_2)$$

$$m(d_2) = y_1 \cdot y_2$$

$$m(d_3) = (1 - y_1) \cdot y_2$$

$$m(\Theta_1) = (1 - y_1) \cdot (1 - y_2)$$

For the radars located at p_2 , we get (see figure 2):

$$m(b_3) = y_3$$

$$m(\Theta_2) = 1 - y_3$$

$$m(b_4) = y_4$$

$$m(\Theta_2) = 1 - y_4$$

Θ_2	d_4	Θ
b_4	$b_3 \cap b_4 = \emptyset$	d_5
	b_3	Θ_2

Figure 5. Combination of two SSFs into the belief function SSFP2.

With normalization constant $K = 1 / (1 - y_3 \cdot y_4)$:

$$m(d_4) = y_3 \cdot (1 - y_4) \cdot K$$

$$m(d_5) = (1 - y_3) \cdot y_4 \cdot K$$

$$m(\Theta_2) = (1 - y_3) \cdot (1 - y_4)$$

Combination of SSFP1 with SSFP2 results in:

SSFP2

Θ_2	Z	Z	Z	$R_{p1} \cap R_{p2}$
d_3	\emptyset	\emptyset	\emptyset	Y
d_4	$b_1 \cap b_2$	$b_1 \cap b_2 \cap b_3$	$b_2 \cap b_3$	Y
	d_1	d_2	d_3	Θ_1

SSFP1

Figure 6. Combination of SSFP1 and SSFP2 into final belief function

$$Y = \Theta_1 \cap d_i = \begin{cases} d_i & \text{if } m_i \in R_{p1} \text{ and } i \in \{3, 4\} \\ \emptyset & \text{elsewhere.} \end{cases}$$

$$Z = \Theta_2 \cap d_i = \begin{cases} d_i & \text{if } m_i \in R_{p2} \text{ and } i \in \{1, 2, 3\} \\ \emptyset & \text{elsewhere.} \end{cases}$$

In this way we can again construct the simple support function from the diagram as we did before.

Remark. The combination of belief functions, which we in the example performed first per location, can be done in arbitrary order, as associativity is guaranteed by Dempster's combination rule.

In ref. 27, more can be found on the modelling of the initiation problem. We will now turn to the second problem.

6.2 The identification problem

The objective of identification is to generate a classification of objects, based on track information generated by the Bayesian multi-sensor tracking system. The objects present in the area under consideration must be classified in accordance with the measurements, into one of a number of classes. As an example, we take:

- the object is a bird (bi)
- the object is a non-flying object (nfo)
- the object is a transport aircraft (ta)
- the object is another, non-transport aircraft (nta)
- the object is a unidentified object due to false measurements (fm).

Together, the classes form the frame of discernment Θ .

At each point in time, the following statements shall be made about an object in order to be able to classify it:

1. The object is (1) moving with constant speed in a constant direction, (2) accelerating in constant direction or (3) turning.
2. The position of the object is inside vs. outside the range of the sensors.
3. The object is at a certain altitude, or is climbing/descending.
4. The object belongs to one of four speed categories:
 - v1: the velocity is 0 m/s.
 - v2: the velocity is in the interval (0 m/s, 5 m/s]
 - v3: the velocity is in the interval (5 m/s, 300 m/s)
 - v4: the velocity is larger than 300 m/s.

These four items from components of a measurement.

If the object is moving with constant speed in a constant direction, than this can be seen as evidence for bi , ta and nta . This is also the case if the object is turning.

If the position of the object is outside the range of the sensors, than this can be seen as evidence for fm . If the position falls inside the range, than evidence for bi , nfo , ta and nta .

If the object remains at altitude, evidence is accumulated for nfo , ta , nta and bi . If the object varies in altitude in a small time interval, evidence is accumulated for ta , nta and bi .

The speed category $v1$ forms evidence for nfo , nta (e.g. helicopters) and bi . Speed category $v2$ forms evidence for nta and bi , $v3$ for nta and ta , and $v4$ for nta .

The easiest way to proceed is to define a belief function for each measurement, and then combine them using Dempster's rule of combination. Disadvantage of this method is that no use is made of dependency between measurements at different time intervals. A passenger aircraft for instance will make few turns, go to an economical altitude and stay there if it can. The components 2 and 3 of the measurement can be left out without deteriorating the model too much.

It would be better to construct belief functions that use as many measurements as possible.

To this end, we define a time interval T in which several measurements take place. Per component, a belief function is constructed. Ultimately, these belief functions are combined into an evaluation function.

Building these belief functions is a matter of constructing functions that comply with our intuition. We know of no standard method to translate this type of knowledge into belief functions. For every problem that is tackled with the Dempster-Shafer theory one will have to use one's imagination. For the identification problem, we have constructed belief functions for each of the components of a measurement. Each knowledge element was translated into a numerical formula. For instance, if we the number of instances in which the object is flying at constant speed in a constant direction, our belief in the object being a transport aircraft increases. To reflect this, we have chosen a quadratic function:

$$m(ta, bi) = \left[\frac{ncm}{x} \right]^2, \text{ where } ncm \text{ is the number of constant movements and } x \text{ the number of measurements.}$$

For the combination of other aircraft and birds we take:

$$m(nta, bi) = 1 - \left[\frac{ncm}{x} \right]^2$$

The other belief functions are given in ref. 28. For this problem, it turned out to be not too hard to define the functions. What is a problem, however, is that Dempster's rule can provide counterintuitive results if the belief functions are not defined correctly. Unfortunately, Shafer's first formulation of the requirements for the use of this rule was rather vague and despite several attempts to clarify the meaning of the requirements, the existing formulations are still not completely clear. Recently, Voorbraak has concluded that the applicability of the Dempster-Shafer theory is rather limited, since bodies of evidence have

to satisfy very strong constraints in order to be called Dempster-Shafer-independent (Ref. 29).

7. CONCLUSIONS AND FURTHER WORK

The idea of not committing evidence to any specific event or set of events until evidence is gained, is a very appealing one from an intuitive point of view. The Dempster-Shafer theory uses the concept of basic probability functions and belief functions to represent this idea. In practice, working with the Dempster-Shafer theory is not easy, and it is not completely clear when Dempster's rule of combination is applicable.

Application of the theory to the multi-radar tracking problem has brought greater understanding of the difficulties of using the theory.

One point that has currently received more attention in the research community is the relation between the Dempster-Shafer theory and other theories. Especially the relation between the Dempster-Shafer theory and the Bayesian subjective probability theory and between the Dempster-Shafer theory and possibilistic reasoning deserves further attention.

Further investigations are to be directed towards comparison of several methods for reasoning with uncertain and incomplete information. First of all, comparison of Bayesian methods and the Dempster-Shafer theory is needed. The multi-sensor fusion environment applied to multi-radar tracking has proven to be a good environment to perform this kind of work. Other methods for reasoning with imperfect information are needed for multi-radar tracking purposed, too. Non-monotonic reasoning methods deserve investigation in this context.

8. ACKNOWLEDGEMENT

The author wishes to thank Ir. S.R. Choenni for his efforts in applying the Dempster-Shafer theory to the multi-radar tracking problem, and Dr. J. Hellendoorn, Dr. H.A.P. Blom and Ir. R.A. Hogendoorn for their cooperation in the modeling effort.

9. REFERENCES

1. R.J.P. Groothuizen, Inexact reasoning in expert systems; an integrating overview. NLR TR 86009.
2. J. Hellendoorn, Reasoning with fuzzy logic. PhD Thesis, Delft Technical University, Dec. 1990. Also published as NLR TP 91024 U.
3. N. Roos, What is on the machine's mind? Models for reasoning with incomplete and uncertain knowledge. PhD Thesis, Delft Technical University, Feb. 1991. To be published as NLR Technical Paper, 1991.
4. P.L. Bogler, Shafer-Dempster Reasoning with Applications to Multisensor Target Identification Problems. In: IEEE Transactions on Systems, Man and Cybernetics, Vol. SMC-17, No.6, Nov./Dec. 1987.
5. E.H. Shortliffe, B.G. Buchanan, A model of inexact reasoning in medicine, in: B.G. Buchanan, E.H. Shortliffe (eds.), Rule-based Expert systems. The MYCIN Experiments of the Stanford Heuristic Programming Project, Addison-Wesley, Reading, 1984.
6. R.O. Duda, P.E. Hart, N.J. Nilsson, Subjective Bayesian Methods for Rule-based Inference Systems. Technical Note 124, Artificial Intelligence Center, SRI International, Menlo Park, 1976.
7. L.C. van der Gaag, Probability-based Models for Plausible Reasoning. PhD Thesis, University of Amsterdam, September 1990.
8. Lea Sombé, Reasoning under incomplete information in Artificial Intelligence, A Comparison of Formalisms Using a Single Example. In: International Journal of Intelligent Systems, Vol. 5, No. 4, September 1990, pp. 323-472.
9. R. Turner, Logics for Artificial Intelligence. Chichester, Ellis Horwood Ltd., 1984.

10. K.-C. Ng, B. Abrahamson, Uncertainty management in expert systems. In: IEEE Expert, April, 1990, pp. 29-41.
11. J. Pearl, Probabilistic reasoning in intelligent systems: Networks of plausible inference. San Mateo, Morgan Kaufman Publishers, Inc., 1988.
12. T. Bayes, An essay towards solving a problem in the doctrine of chances. In: Phil. Trans. 3, pp. 370-418. Reproduced in: Two papers by Bayes, ed. W.E. Deming. New York, Hafner, 1963.
13. J.M. Keynes, A treatise on probability. London, MacMillan, 1921.
14. H. Jeffreys, Theory of probability. Oxford, Clarendon Press, 1939.
15. P.R. Cohen, Heuristic Reasoning about Uncertainty: An Artificial Intelligence Approach. London, Pitman, 1985.
16. J. Doyle, A Truth Maintenance System. In: Artificial Intelligence, Vol. 12, 1979, pp.231-272.
17. J. de Kleer, An Assumption-based Truth Maintenance System. In: Artificial Intelligence, Vol. 28, 1986, pp. 127-162.
18. J. de Kleer, Extending the ATMS. In: Artificial Intelligence, Vol. 28, 1986, pp. 163-196.
19. J. de Kleer, Problem solving with the ATMS. In: Artificial Intelligence, Vol. 28, pp. 197-224.
20. L. Zadeh, Fuzzy sets as a basis for a Theory of Possibility, in: Fuzzy Sets and Systems. Vol. 1, No. 1.
21. A.P. Dempster, Upper and lower probabilities induced by a multi-valued mapping. In: Ann. Mathematical Statistics, Vol. 38, 1967, pp. 325-339.
22. A.P. Dempster, A generalization of Bayesian inference. J. of the Royal Statistical Society, ser. B 30, 1968, pp. 205-247.
23. G. Shafer, Two theories of probability. Technical Report, Department of mathematics, University of Kansas. Lawrence, Kansas, 1978.
24. G. Shafer, A mathematical theory of evidence. Princeton, Princeton University Press, 1976.
25. H.A.P. Blom, R.A. Hogendoorn, F.J. van Schaik, Bayesian multi-sensor tracking for advanced air traffic control systems. AGARDograph 301 on Computation, Prediction and Control of Aircraft Trajectories, 1990. Also published as NLR MP 88056 U.
26. Y. Bar-Shalom, K.C. Chang, H.A.P. Blom, Tracking a manoeuvring target using Ipnut Estimation versus the Interacting Multiple Model algorithm, IEEE Transactions on Aerospace and Electronic Systems, Vol. 25, March 1989, pp. 296-300.
27. J.F. Gerlofs, Technical feasibility of the multi-radar jump diffusion tracker, NLR Memorandum IR-86-030 L, Amsterdam, 1986.
28. S.R. Choenni, The Dempster-Shafer theory and applications of the Dempster-Shafer theory for multi-radar tracking (in Dutch). NLR memorandum IN-90-016, September 1990.
29. F. Voorbraak, On the justification of Dempster's rule of combination. In: Artificial Intelligence, Vol. 48, March 1991, pp. 171-197.



REPORT DOCUMENTATION PAGE

1. Recipient's Reference	2. Originator's Reference AGARD-CP-499	3. Further Reference ISBN 92-835-0628-6	4. Security Classification of Document UNCLASSIFIED
5. Originator	Advisory Group for Aerospace Research and Development North Atlantic Treaty Organization 7 rue Ancelle, 92200 Neuilly sur Seine, France		
6. Title	MACHINE INTELLIGENCE FOR AEROSPACE ELECTRONIC SYSTEMS		
7. Presented at	the Avionics Panel Symposium held in Lisbon, Portugal 13th—16th May 1991		
8. Author(s)/Editor(s) Various	9. Date September 1991		
10. Author's/Editor's Address Various	11. Pages 308		
12. Distribution Statement	This document is distributed in accordance with AGARD policies and regulations, which are outlined on the back covers of all AGARD publications.		
13. Keywords/Descriptors Machine intelligence Artificial intelligence C ³ I Knowledge based systems Expert systems			
14. Abstract A large amount of research has been conducted to develop and apply Machine Intelligence (MI) technology to aerospace applications. Machine Intelligence research covers the technical areas under the headings of Artificial Intelligence, Expert Systems, Knowledge Representation, Neural Networks and Machine Learning. This list is not all inclusive. It has been suggested that this research will dramatically alter the design of aerospace electronics systems because MI technology enables automatic or semi-automatic operation and control. This symposium was organized to present the results of efforts applying MI technology to aerospace electronics applications. The symposium focused on applications research and development to determine the types of MI paradigms which are best suited to the wide variety of aerospace electronics applications. Papers presented at the Avionics Panel Symposium, held in Lisbon, Portugal, 13th—16th May, 1991.			

<p>AGARD Conference Proceedings 499 Advisory Group for Aerospace Research and Development, NATO MACHINE INTELLIGENCE FOR AEROSPACE ELECTRONIC SYSTEMS Published September 1991 308 pages</p> <p>A large amount of research has been conducted to develop and apply Machine Intelligence (MI) technology to aerospace applications. Machine Intelligence research covers the technical areas under the headings of Artificial Intelligence, Expert Systems, Knowledge Representation, Neural Networks and Machine Learning. This list is not all inclusive. It has been suggested that this research will dramatically alter the design of aerospace electronics systems because MI technology enables automatic or semi-automatic operation and control.</p> <p>P.T.O.</p>	<p>AGARD-CP-499</p> <p>Machine intelligence Artificial intelligence C'I Knowledge based systems Expert systems</p>	<p>AGARD Conference Proceedings 499 Advisory Group for Aerospace Research and Development, NATO MACHINE INTELLIGENCE FOR AEROSPACE ELECTRONIC SYSTEMS Published September 1991 308 pages</p> <p>A large amount of research has been conducted to develop and apply Machine Intelligence (MI) technology to aerospace applications. Machine Intelligence research covers the technical areas under the headings of Artificial Intelligence, Expert Systems, Knowledge Representation, Neural Networks and Machine Learning. This list is not all inclusive. It has been suggested that this research will dramatically alter the design of aerospace electronics systems because MI technology enables automatic or semi-automatic operation and control.</p> <p>P.T.O.</p>	<p>AGARD-CP-499</p> <p>Machine intelligence Artificial intelligence C'I Knowledge based systems Expert systems</p>
<p>AGARD Conference Proceedings 499 Advisory Group for Aerospace Research and Development, NATO MACHINE INTELLIGENCE FOR AEROSPACE ELECTRONIC SYSTEMS Published September 1991 308 pages</p> <p>A large amount of research has been conducted to develop and apply Machine Intelligence (MI) technology to aerospace applications. Machine Intelligence research covers the technical areas under the headings of Artificial Intelligence, Expert Systems, Knowledge Representation, Neural Networks and Machine Learning. This list is not all inclusive. It has been suggested that this research will dramatically alter the design of aerospace electronics systems because MI technology enables automatic or semi-automatic operation and control.</p> <p>P.T.O.</p>	<p>AGARD-CP-499</p> <p>Machine intelligence Artificial intelligence C'I Knowledge based systems Expert systems</p>	<p>AGARD Conference Proceedings 499 Advisory Group for Aerospace Research and Development, NATO MACHINE INTELLIGENCE FOR AEROSPACE ELECTRONIC SYSTEMS Published September 1991 308 pages</p> <p>A large amount of research has been conducted to develop and apply Machine Intelligence (MI) technology to aerospace applications. Machine Intelligence research covers the technical areas under the headings of Artificial Intelligence, Expert Systems, Knowledge Representation, Neural Networks and Machine Learning. This list is not all inclusive. It has been suggested that this research will dramatically alter the design of aerospace electronics systems because MI technology enables automatic or semi-automatic operation and control.</p> <p>P.T.O.</p>	<p>AGARD-CP-499</p> <p>Machine intelligence Artificial intelligence C'I Knowledge based systems Expert systems</p>

<p>This symposium was organized to present the results of efforts applying MI technology to aerospace electronics applications. The symposium focused on applications research and development to determine the types of MI paradigms which are best suited to the wide variety of aerospace electronics applications.</p> <p>Papers presented at the Avionics Panel Symposium, held in Lisbon, Portugal, 13th—16th May, 1991.</p> <p>ISBN 92-835-0628-6</p>	<p>This symposium was organized to present the results of efforts applying MI technology to aerospace electronics applications. The symposium focused on applications research and development to determine the types of MI paradigms which are best suited to the wide variety of aerospace electronics applications.</p> <p>Papers presented at the Avionics Panel Symposium, held in Lisbon, Portugal, 13th—16th May, 1991.</p> <p>ISBN 92-835-0628-6</p>
<p>This symposium was organized to present the results of efforts applying MI technology to aerospace electronics applications. The symposium focused on applications research and development to determine the types of MI paradigms which are best suited to the wide variety of aerospace electronics applications.</p> <p>Papers presented at the Avionics Panel Symposium, held in Lisbon, Portugal, 13th—16th May, 1991.</p> <p>ISBN 92-835-0628-6</p>	<p>This symposium was organized to present the results of efforts applying MI technology to aerospace electronics applications. The symposium focused on applications research and development to determine the types of MI paradigms which are best suited to the wide variety of aerospace electronics applications.</p> <p>Papers presented at the Avionics Panel Symposium, held in Lisbon, Portugal, 13th—16th May, 1991.</p> <p>ISBN 92-835-0628-6</p>

AGARD does NOT hold stocks of AGARD publications at the above address for general distribution. Initial distribution of AGARD publications is made to AGARD Member Nations through the following National Distribution Centres. Further copies are sometimes available from these Centres (except in the United States), but if not may be purchased in Microfiche or Photocopy form from the Sales Agencies listed below.

NATIONAL DISTRIBUTION CENTRES

BELGIUM

Coordonnateur AGARD -- VSL
Etat-Major de la Force Aerienne
Quartier Reine Elisabeth
Rue d'Evere, 1140 Biuxelles

CANADA

Director Scientific Information Services
Dept of National Defence
Ottawa, Ontario K1A 0K2

DENMARK

Danish Defence Research Board
Ved Idrætsparken 4
2100 Copenhagen Ø

FRANCE

O.N.E.R.A. (Direction)
29 Avenue de la Division Leclerc
92320 Châtillon

GERMANY

Fachinformationszentrum
Karlsruhe
D-7514 Eggenstein-Leopoldshafen 2

GREECE

Hellenic Air Force
Air War College
Scientific and Technical Library
Dekelia Air Force Base
Dekelia, Athens TGA 1010

ICELAND

Director of Aviation
c/o Flugrad
Reykjavik

ITALY

Aeronautica Militare
Ufficio del Delegato Nazionale all'AGARD
Aeroporto Pratica di Mare
00040 Pomezia (Roma)

THE UNITED STATES NATIONAL DISTRIBUTION CENTRE (NASA) DOES NOT HOLD STOCKS OF AGARD PUBLICATIONS, AND APPLICATIONS FOR COPIES SHOULD BE MADE DIRECT TO THE NATIONAL TECHNICAL INFORMATION SERVICE (NTIS) AT THE ADDRESS BELOW.

SALES AGENCIES

National Technical
Information Service (NTIS)
5285 Port Royal Road
Springfield, Virginia 22161
United States

ESA/Information Retrieval Service
European Space Agency
10, rue Mario Nikis
75015 Paris
France

The British Library
Document Supply Centre
Boston Spa, Wetherby
West Yorkshire LS23 7BQ
United Kingdom

Requests for microfiches or photocopies of AGARD documents (including requests to NTIS) should include the word 'AGARD' and the AGARD serial number (for example AGARD-AG-315). Collateral information such as title and publication date is desirable. Note that AGARD Reports and Advisory Reports should be specified as AGARD-R-nnn and AGARD-AR-nnn, respectively. Full bibliographical references and abstracts of AGARD publications are given in the following journals:

Scientific and Technical Aerospace Reports (STAR)
published by NASA Scientific and Technical
Information Division
NASA Headquarters (NTT)
Washington D.C. 20546
United States

Government Reports Announcements and Index (GRA&I)
published by the National Technical Information Service
Springfield
Virginia 22161
United States

(also available online in the NTIS Bibliographic
Database or on CD-ROM)



Printed by Specialised Printing Services Limited
40 Chigwell Lane, Loughton, Essex IG10 3TZ