AGARD-AR-302

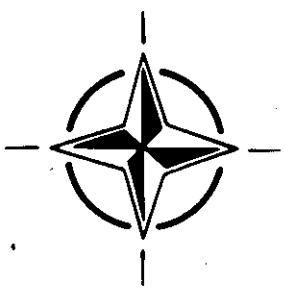
AGARD
ADVISORY GROUP FOR AEROSPACE RESEARCH & DEVELOPMENT
7 RUE ANCELLE 92200 NEUILLY SUR SEINE FRANCE

AGARD ADVISORY REPORT 302

Technical Evaluation Report on the Guidance and Control Panel 52nd Symposium on Software for Guidance and Control

(Les Logiciels de Guidage et de Pilotage)

The Guidance and Control Panel's 52nd Symposium was held at the Helexpo, Thessaloniki, Greece from 7th to 10th May 1991. All papers presented at the Symposium were compiled as Conference Proceedings CP 503.



NORTH ATLANTIC TREATY ORGANIZATION

Published December 1991

Distribution and Availability on Back Cover

41 pp

AGARD
ADVISORY GROUP FOR AEROSPACE RESEARCH & DEVELOPMENT
7 RUE ANCELLE 92200 NEUILLY SUR SEINE FRANCE

AGARD ADVISORY REPORT 302

Technical Evaluation Report on the Guidance and Control Panel 52nd Symposium on Software for Guidance and Control

(Les Logiciels de Guidage et de Pilotage) 2/02

by

Mr Donald E. Dewey
Chief, Research Planning and Administration
Boeing Military Airplanes
Box 3707, MS 4C-15
Seattle, WA 98124
United States

The Guidance and Control Panel's 52nd Symposium was held at the Helexpo, Thessaloniki, Greece from 7th to 10th May 1991. All papers presented at the Symposium were compiled as Conference Proceedings CP 503.

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

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.

Published December 1991

ISBN 92-835-0647-2

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

Preface

Software is of increasing importance in guidance and control systems and indeed in many cases is the pacing item in development. Guidance and control software, while embracing a wide range of software, has emphases which include high integrity considerations, hard real-time constraints, the implications of a still evolving hardware and systems architecture, and the need to meet delivery schedules with high productivity under the constraints of onerous customer requirements for documentation and visibility, and in the light of strong defence and air worthiness standards and requirements. Time schedules are frequently short since much guidance and control software is required early in the flight testing, and typically software development is undertaken in the context of still evolving requirements and developing programme phases.

The software climate in which this takes place is one in which there is a general trend towards high level languages, integration of support tools, introduction of mathematical formalisms into the design and verification processes, control of software sizing and better cost estimating, and frequently a rapid turnover of programmers.

There is often a wide gap between concept and practice, and organizations will succeed which can bridge the gap effectively, bringing modern methodologies, well supported by software tools, to bear on the problem and understanding how to apply these methodologies and use the tools.

To assist this understanding the symposium covered general requirements on the software, software requirements capture, design methods and support environments for real-time software, coding techniques, and verification validation and certification.

Préface

Les logiciels revêtent de plus en plus d'importance dans les systèmes de guidage et de pilotage. En effet, le logiciel est souvent l'élément critique pour le développement des systèmes.

Bien qu'il existe une large gamme de logiciels de guidage et de pilotage, l'accent est mis principalement sur les considérations suivantes: la haute intégrité, les contraintes temps réel du matériel, les conséquences de l'évolution permanente des architectures systèmes et matériel, le respect des délais de livraison pour des volumes de production élevés, la demande onéreuse de documentation de la part du client, les contraintes d'intelligibilité du logiciel, et la rigueur des spécifications et normes militaires et aéronautiques. Les délais sont souvent courts, puisque bon nombre des logiciels de guidage et de pilotage sont demandés dès la première phase des essais en vol; typiquement, le logiciel est crée pendant que les besoins continuent à évoluer dans le contexte des différentes phases évolutives du programme.

L'environnement logiciel de ce processus est caractérisé par les langages évolués, l'intégration des outils de développement, l'emploi de formalismes mathématiques dans les méthodes de conception et de vérification, le contrôle du dimensionnement des logiciels, la recherche d'une meilleure estimation des coûts et le renouvellement fréquent des programmeurs.

Il existe souvent un grand pas à franchir pour passer du concept à la pratique. Les organisations qui réussiront à l'avenir seront celles qui sauront franchir ce pas de façon efficace, en se servant de méthodologies modernes, bien appuyées par des outils de développement, et qui auront compris l'application de ces méthodologies et la mise en oeuvre de ces outils.

Afin de faciliter cette compréhension, le symposium a examiné les sujets suivants: conditions générales requises pour les logiciels, élaboration des spécifications, méthodes de conception et environnements de soutien pour les logiciels temps réel, techniques de codage, et vérification, validation et certification.

Guidance and Control Panel

Chairman: Dr E.B. Stear
Corporate Vice President
Technology Assessment
The Boeing Company
PO Box 3707
Mail Stop 13—43
Seattle, WA 98124-2207
United States

Deputy Chairman: Mr S. Leek
Scientific Adviser
British Aerospace (Dynamics) Ltd, PB 256
PO Box 19
Six Hills Way
Stevenage
Herts SG1 2DA
United Kingdom

TECHNICAL PROGRAMME COMMITTEE

Chairman:	Professor J.T. Shepherd	UK
Members:	Dr André Benoît	BE
	Mr B. Jaeger	FR
	Mr U.K. Krogmann	GE
	Lt F. Hatzivasiliou	GR
	Mr K.A. Helps	UK
	Mr S. Haaland	US
	Dr J. Niemela	US
	Dr E. Zimet	US

HOST PANEL COORDINATOR

Lt Fivos Hatzivasiliou
Hellenic Air Force
Technology Center (KETA)
Terpsithea Post Office
16501 Glyfada, Athens
Greece

PANEL EXECUTIVE

Commandant M. Mouhamad, FAF

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

Mail from US and Canada:
AGARD—NATO
Attn: GCP Executive
Unit 21551
APO AE 09777

Tel: 33(1) 47 38 57 80
Telex: 610176 (F)
Telefax: 33(1) 47 38 57 99

ACKNOWLEDGEMENTS/REMERCIEMENTS

The Panel wishes to express its thanks to the Greek National Delegates to AGARD for the invitation to hold this meeting in their country and for the facilities and personnel which made the meeting possible.

Le Panel tient à remercier les Délégués Nationaux de la Grèce près l'AGARD de leur invitation à tenir cette réunion dans leur pays et de la mise à disposition de personnel et des installations nécessaires.

Contents

THE 52ND GUIDANCE AND CONTROL PANEL TECHNICAL EVALUATION REPORT

by
Mr. Donald E. Dewey
Chief, Research Planning and Admistration
Boeing Military Airplanes
Seattle, Washington U.S.A.

1.0 INTRODUCTION

The 52nd Guidance and Control Panel (GCP) Symposium was held in Thessaloniki, Greece from May 7 to May 10, 1991. The symposium was on the important subject of "Software for Guidance and Control."

As stated in the announcement made for this symposium, "software is of increasing importance in guidance and control (G&C) systems and indeed in many cases is the pacing item in development." This importance is characterized by the need for G&C software to meet the highest standards of flight safety, meet hard real-time operational constraints, be responsive to the evolving hardware and system architectures, and still meet stringent delivery schedules in a cost-effective way. The 52nd GCP Symposium was structured to address these challenges and the various approaches being taken to meet them.

To meet these challenges, there is a strong movement towards using high-level languages (especially Ada); domain specific support tools; mathematical formalisms in the design and verification processes; and methods to manage the development cost, schedules, and integrity of the system software.

The most notable conclusion drawn by most attendees was that support tools and methods for improving software integrity and software development productivity are becoming available and are effective.

2.0 SYMPOSIUM PROGRAM

This symposium was organized in seven technical sessions, each preceded by introductory remarks by the session chairman.

Session I, Tools and Methods From a User's Viewpoint, contained two papers which surveyed the tools and methods available for software design in general, and the tools and methods considered for the EFA project in particular.

Session II, General Requirements on Software, contained three papers which included a comparison of the United States DOD-STD-2167A military standard and the RTCA DO-178A/CUROCAE ED-12A civil aircraft standard, a description of a requirements and traceability management tool, and a discussion of a coprocessor approach to relieving the overhead burden in real-time, software systems.

Session III, Integrated Programmes Support Environments, contained four papers which discussed the features of various software development support environments. Included in the discussions were the VISA, GALA/GALI, BVA, VISA, PLAS, and AGLAE workstations, and a modular, multipurpose command and control workstation.

Session IV, Software Requirements, contained three papers which concentrated on methods to improve software productivity and verify correctness. Two of the papers described the formal methods approach and one paper described a unified software specification development and simulation process method.

Session V, Design Methods for Real-Time G&C Software, contained six papers that described different methods for developing real-time G&C software. Included in the methods were a system called network programming (a decentralized approach), a data-oriented requirements implementation scheme called DORIS, and a commercially available tool set called DSP-CITpro. Also, in this session, were papers on how the predictive functional control technique has been used to design control laws, and a paper on the U.S. Naval Weapons Center Analyst Workbench, which is used to analyze large quantities of flight test and simulation test data.

Session VI, Ada Applications contained five papers which described the experiences and "lessons learned" after using Ada on various programs. The programs included the RAFALE avionics system, an experimental Lynx helicopter flight control system, and the EFA pilot ejection control and seat sequencer systems. Also, included in this session, were papers on the CAMP program and on a United Kingdom study to evaluate the suitability and problems of using Ada language in flight safety critical systems.

Session VII, Automated Software Generation Approaches, contained four papers on the research and findings of the NATO/AGARD G&C Panel Working Group 10 (WG 10). WG 10 began investigating automated software generators in 1988 as a means to reduce the impact of software development time, technical risk, and complexity on G&C system development programs. WG 10 defined and investigated four approaches to software generation: reusable software modules, expert systems, program transformation techniques, and fourth generation languages. A report on the findings of WG 10 will be published later this year (1991).

3.0 REVIEW OF SYMPOSIUM SESSIONS

3.1 SESSION I: TOOLS AND METHODS FROM A USER'S VIEWPOINT

Chairman: Professor J. T. SHEPHERD, (UK)

Paper 1: A Survey of available tools and methods for software requirements capture and design

D. THEWLIS, GEC-Marconi Ltd., Stanmore, Middlesex, UK

This paper discusses the basic problem of capturing the real requirements of a system design. The current requirements capturing methods used at GEC are the Mascot, Jackson System Design (JSD), and Yourdon.

GEC uses four requirements capturing methods: Teamwork and Software Through Pictures (Yourdon-type methods) and Speedbuilder and PDF (Jackson-type methods). According to GEC, the problem with these methods is that they assume all the requirements are known and a top-down design can be made. However, they have found that if the tools are used in a strictly top-down way (defining the top-level requirements, then decomposing down to lower requirements) the resulting lower level designs are poor and do not meet the desired quality standards. The lower level design process failed because the requirements were incomplete and changed throughout the program. The solution they used was to use a bottom-up design approach and use the tools as a method for recording and maintaining the system design.

Shortcomings in their approach may be overcome by the newer object-oriented design methods, the author believes. For this reason, the European Space Agency has developed the HOOD, a hierarchical object-oriented design method, that has been adopted by the European Fighter Aircraft consortium. Tool sets are now being developed for the HOOD method.

Paper 2: Tool supported software development—experiences from the EFA project.

W. M. FRAEDRICH, BMVg, Bonn, GE

This paper describes what software development standards and methods were agreed upon for the EFA program by the multinational participants. The selection of standards had to be worked out jointly, since one nation could not unilaterally impose standards on other participating nations. The process followed by the participants was to, first, find the "lowest common denominator" of standards being used by the participants and, then, establish agreements as necessary on the remaining standards.

Early in the EFA program four documents, called the Requirements of the Officials, established the guiding requirements for the program. They were:

a. European Staff Target (EST, 11.10.84) which specified the use of a common high-order language.
b. European Staff Requirement (ESR, 09.12.85) which identified Ada as the preferred high-order language and stated that the Ada programming support environment (APSE) should be used, if available.
c. European Staff Requiremnt for Development (ESR-D, 19.09.87) which specified when exceptions to Ada or the tools were permitted and who could approve waivers.
d. Weapon System Design and Performance Specification (WSDPS, 01.10.88) which specified that the software should be developed using a software development environment (SDE) and the SDE should include the CORE/EPOS and other tools agreed upon by the customer.

In early 1988, Eurofighter Company presented the results of a study showing Ada could be used in all safety critical applications. The study showed that the reliability of Ada programs is comparable to that of assembler programs (if not greater), "if any restriction on the use of the Ada language is strictly adhered to (SAFE Ada) and the static code analyses at source code level is made." This finding led the EFA program to accept Safe Ada for use in the flight critical portions of the EFA software program.

The system design environment selected for the EFA (EFA SDE) includes:

a. CORE/EPOS for system design.
b. HOOD toolset for software design.
c. SPARK examiner for static code analyses.
d. TESTBED for dynamic tests/analyses.
e. XD-Ada for Ada compiler.
f. IPSE based on perspective

The authors believe that the system design tools are the most critically needed. However, some of the EFA project particpants worked for a long time without the tools, due to time-consuming delays in obtaining license agreements and a clear definition of the participants' needs. The CORE/EPOS tool is still not developed sufficiently for all members of the multinational group. The HOOD tool has been available since December 1990 and will be used. The Ada compiler for the standard 68020 microprocessor is available and functioning. The SPARK examiner and TESTBED tools have been available since Novemeber 1990. Updates are required, but this does not seem to pose a problem.

REVIEW OF SESSION I:

D. Thewlis highlighted one of the major problems in the process of developing guidance and control software: identifying and capturing the full system requirements at the beginning of the program. As he points out, total system requirements are not known at the beginning of a program (or for that matter, for much of the earlier portions of a program), but, often, many of the lower details of the system are known. So why not design the lower design details while defining the total system and then fit the pieces together at a later time? The suggestion was generally accepted by the symposium attendees.

W. Fraedrich's position on the acceptability of "Safe Ada" (restricted language set and other conditions) for use in flight safety critical applications was challenged by some symposium attendees. However, sound reasons for not accepting the position of W. Fraedrich were not given, but there are differing opinions on the suitability of Ada in flight critical applications, depending on the company affiliation and, at times, the technical discipline.

3.2 SESSION II: GENERAL REQUIREMENTS ON SOFTWARE

Chairman: Mr. S. HAALAND, (US)

Paper 3: Military and civil software standards and guidelines for guidance and control

K. W. WRIGHT, Smiths Industries Aerospace & Defence Systems, Cheltenham, UK

This paper compares the two widely used standards covering the development of software in the military and civil avionics industries. These two standards are the United States DOD-STD-2167A military standard and the RTCA DO-178A/EUROCAE ED-12A civil aircraft standard. This comparison is of considerable interest due to recent attempts to show the degree of transferability of systems certified under one standard to the other.

First, the authors point out the differences in scope of the two standards. The purpose of DOD-STD-2167A is to provide a procurement specification for deliverable software in the form of computer software configuration items. RTCA DO-178B describes the software development and management methods and techniques that are to be used in the development of systems. Because the DOD standard is principally a procurement standard, there are detailed requirements for the software development documentation required as deliverables. The guidelines of the RTCA civil standard are aimed at giving the certification authorities the assurance that the software has been developed per the regulations and, as such, there are no formal, specified formats required for the documentation.

In summary, the author concludes that the documentation and certification of the software provided under the DOD standard can be used to satisfy the RTCA civil standards, if the additional documents required by the RTCA standard, such as the Accomplishment Summary and the Quality Assurance Plan, are provided. On the other hand, it is highly unlikely that the information and process following the RTCA standard will be acceptable for military aircraft, due to the documentation flexibility permitted by the RTCA standard.

Paper 4: Requirements traceability and management.

G. M. CROSS, Marconi Underwater Systems Ltd., Addlestone, UK

This paper describes how a new system development support tool was developed and used at Marconi Underwater Systems Limited. The new tool, called the RTM (requirements and traceability management), is used to capture the system requirements at the requirements capture portion of the program (which may span more time than just the requirements analysis portion of the program). The RTM is then used throughout the system design, coding, unit test, software integration, and acceptance portion of the program.

The RTM does not take the place of other analysis, design, coding, configuration, or documentation tools. RTM does provide a frame of reference during the other phases of the development program, providing a basis for some degree of testing the particular stage of the system development for compliance to the system requirements.

Paper 5: Coprocessor support for real-time ADA.

R: K. PAGE, Naval Weapons Center, China Lake, CA, US

This paper proposes that system designers of real-time systems like those encountered in guidance and control systems should consider using a coprocessor configuration to:

a. Relieve the main program tasking processor of some of the overhead burden of time and task management.
b. Provide adequate granularity for the representation of time.
c. Provide sufficient range for the representation of time.

REVIEW OF SESSION II:

Mr. K. Wright's report on the recent activity to resolve the differences (or at least to understand them)

between the United States and European software standards is timely and important. There is considerable interest, within the NATO community, in making the two standards more compatible. RTCA DO-178 is currently being reviewed and updated by the RTCA Special Committee 167 and EUROCAE Working Group 12. The current target date for the completion of this work and publication of the new standard is December 1991.

At the symposium, there was considerable interest in the coprocessor work being conducted at the Naval Weapons Center in the United States. The coprocessor concept has merit and is being explored as a means for meeting the needs of systems requiring high bandwidths. This approach is similar to the transputer concept. The question was raised about whether or not it would be advisable to include a coprocessor directly on the same chip as the processor. There are advantages to this concept, but the main disadvantage would be the loss of flexibility of being able to select any processor and coprocessor type and configuration.

3.3 SESSION III: INTEGRATED PROGRAMMES SUPPORT ENVIRONMENTS

Chairman: Mr. J.B.SENNEVILLE, (FR)

Paper 6: Atelier de développement de logiciels de pilotage—guidage (Guidance software development workshop)

D. CAIGNAULT, J. L. LEBRUN, SEXTANT Avionique, Villacoublay, FR

This paper discusses the design features of the VISA, GALA/GALI, and BVA software development support tools.

VISA (validation interactive de specifications avioniques) supports specification development. VISA has a graphic interface that simplifies its use and it can check for coherence of the specification through simulation. GALA (generation automatique de logiciel avionique) and GALI (generation automatique de logiciel d'interface) contains a "hands-off" automatic code generator that can go from the specification to code. BVA (le banc de validation avionique) supports the final validation phase. A fourth program, PALAS (production assistee de logiciel d'application structure) is a configuration management tool that manages the documentation and data throughout the development cycle.

Paper 7: Atelier de spécification/maquetage pour les logiciels de gestion du vol (Software development workstation)

H. ROBIN, J-C. MIELNIK, SEXTANT Avionique, Villacoublay, FR

The tools in the software development workstation described by the authors include requirements capturing and specification debugging (syntax correcting) methods that can support the validation process. The most interesting tool, however, is the rapid prototyping tool they are using. This tool uses the KEE object-oriented design language adapted to the guidance and control domain. They are enthusiastic about this approach, relating time savings from several hours on simple applications to several weeks for more complicated applications.

While in the process of designing and developing the specification and rapid prototyping tools, they considered a variety of many different methods and language environments. The paper contains an informative discussion of the pros and cons of the OOA, SART, ESTEREL, LUSTRE, and HMS specification integration methods and of the comparative usefulness of the Ada, LISP, KEE, Fortran, and C languages in the rapid prototyping tool.

Paper 8: AGLAE—Atelier de génie logiciel de l'aérospatiale, division engins (Aerospace software engineering works)

J. HAMON, F. BOIS, M. VAZEILLES, D. MOUSSEAU, F. Y. VILLEMIN, AEROSPATIALE, Service E/ETEL, Châtillon s/Bagneux, FR, CNAM-CEDRIC, Paris, FR

This paper describes the AGLAE software environment developed by Aerospatiale for designing aircraft control systems.

AGLAE is an automated system with an expert system to assist the user in designing a control system by traditional methods. The rule base of the AGLAE expert system contains rules reproducing the traditional knowledge of control system design as represented in the AEROSPATIALE E/ETEL. A "fact" base for the expert system contains:

a. A database about the characteristics of the hardware components to be used in the control system.
b. Facts about the control system being designed.
c. The requirement specification of the control system showing the detail of its loops and external inputs/outputs with all the detail of the algorithms of the system control laws.

Item "c" is then enriched in detail as the design progresses. At the end of this process, there will be sufficient detail about the cyclic tasks with their timeframes, the input/output functions, and the hardware

configuration chosen from components in "a" to move on to the coding phase of the development cycle.

AGLAE is hosted on a SUN 3/260 workstation and the UNIX environment with the support of the expert system generator KNOWLEDGE CRAFT V3.3. The facts base is represented as a set of classes and objects interconnected by inheritance relationships and grouped in workable contexts by KNOWLEDGE CRAFT. The base of rules, along with the inference engine, are software packages that have been developed using COMMON LISP V4.0.

The author of this paper divides the system life cycle into four main phases:

a. System concept definition and software requirement specification.
b. Preliminary and detailed design.
c. Coding, tests, integration, and validation.
d. Installation, commissioning, and maintenance.

The author is careful to point out that AGLAE only operates in the preliminary and detailed design phase. In the future, however, AEROSPATIALE hopes to be able to integrate the present expert system with other facilities and environments to extend the usefulness of the system into the other three phases.

Paper 9: Withdrawn.

Paper 10: Software design considerations for an airborne command and control workstation

P. KUHL, B. MUTH, P. KEILHORN, R. VISSERS, Dornier Luftfahrt GmbH, Friedrichshafen, GE

This paper describes a modular, multipurpose command and control workstation developed by Dornier Luftfahrt GmbH. The interesting features about the workstation are not so much in the applications, but in the design; particularly, of the software.

The workstation hardware includes a main processor, add-on processors (for additional specialized functions), a graphic engine with one or more display screens, operator input devices, and external system interfaces. The highly modularized approach divided the software into building blocks. Ada was used throughout as the programming language and ARTX (operating system) was used as well.

The workstation design features the following:

a. Open architecture/modularity—permits easy expansion of new modules to obtain additional features.
b. Reusability—almost all of the basic software contained in the basic system can be used for command and control domain applications.
c. Portability—system software is almost entirely in modular blocks, written in Ada (greater than 95% of the code).
d. Real-time processing capability/tasking—no detrimental problems in real-time processing have been experienced, which the authors attribute to the use of the ARTX operating system.

REVIEW OF SESSION III:

The three tools (VISA, GALA/GALI, and BVA) plus the PALAS configuration management program described in paper 6 are impressive in their scope of usefulness, but they are not unique. Their value is inherent in the fact that they exist and have been proven to be useful tools in the development of large software systems.

Paper 7 is similar to paper 6 in purpose and scope. Like paper 6, the authors of paper 7 described software development tools they have developed and used. However, SEXTANT Avionique went one step further; they have been working with an object-oriented design language that has been adapted to the guidance and control domain. This work is most promising, and should prove to be a very effective tool for developing guidance and control software.

The ALGAE system (paper 8) is a very promising approach to guidance and control software system design, and it is very important that this program be watched closely in the future. AEROSPACIALE has experienced time savings of 50% (on the average) during the preliminary and design phase and a better design has been created in the process. Extended to the other phases, ALGAE could become an even more powerful tool.

Paper 10 once again shows that it is quite possible to design an effective workstation for the development of guidance and control software (or other specific application domain).

3.4 SESSION IV: SOFTWARE REQUIREMENTS

Chairman: Mr. K. A. HELPS, (UK)

Paper 11: Utilisation industrielle de spécifications formelles pour les télémesures (Formal specification of satellite telemetry: a practical experience)

M. LEMOINE, J-M. HUFFLEN, ONERA-CERT/DERI, Toulouse, FR

This paper describes the use of formal algebraic specifications to provide a usable specification for processing telemetry information. The formalism, inherent in formal algebraic specification methods, was used as a means of removing ambiguity and developing a software family in which reusability was a primary aim.

The authors discuss the decision, made early in the project, to not use software that already existed, but, instead, to write a formal specification of the existing software before integrating the software into the larger system. The exercise was a failure because the writers of the formal specification were not telemetry specialists and could not properly interpret the telemetry requirements.

To correct this problem, the telemetry specialists wrote an informal but rigorous set of requirements. Then, the formal specification writers were able to prepare the formal algebraic specification.

The lessons learned from the project were:

a. A formal algebraic specification is the only way to remove specification ambiguities.
b. A formal specification is an efficient way to develop a reusable family of software.
c. A formal specification is an excellent way to express what a user wants which greatly simplifies the V&V process.

Experience has shown that the telemetry specialists accept the formal language more readily than software engineers. The authors concluded that writing a formal specification is becoming a feasible task for industrial applications.

Paper 12: Formal verification of a redundancy management algorithm

J. DRAPER, GEC Avionics, Rochester, Kent, UK

This paper describes an offline experimental application of formal techniques to capture and verify the redundancy management part of a flight control system. The system was part of a safety critical software avionic system.

The first stage involved the use of a specification language Z and verified the specification with handwritten, rigorous proofs. In the first part, the final result was a proof that alternated between a series of

formal steps and rigorous steps. The formal steps were very tedious to check (they were checked by hand) and the rigorous steps were hard to check as well, thus, undermining the confidence in the whole process. The second phase adopted a computer-aided tool to assist in the identification of many of the formal steps and to provide a record of the proof process.

The author concluded that using a formal specification highlighted ambiguities in the informal description and allowed for their removal before going into test and verification. The list of assumptions made during the formal specification development was useful both for the formal specification and for the proof process as well. An approach in which theorems are written with wide coverage originally and to which exceptions are added later in the proof stage appears to be advantageous, even when those exceptions are handled by informal analyses.

Paper 13: A methodology for software specification and development based on simulation

G. FERNANDEZ de la MORA, R. MINGUEZ, S. KHAN, J. R. VILLA, SENER, Madrid, SP

This paper discusses an experimental method for specifying and developing guidance and control software. The authors refer to this method as the "phased" or "simulation-based development" approach. The application case discussed in the paper was a small flight critical subset of the software used for the EJ-200 DECU (software used in the control of the EFA engine main metering valve and the main fuel metering unit).

Starting from a systems requirements document, the approach taken involved the capture of requirements for both "simulation embodied software" and "other simulation software". The former encapsulates the parts of the software which are common to the simulation and the flight software and the latter relates to the simulation of the environment. The approach basically proposes that the software (developed early in the system definition phase for requirements analysis and design definition) simulates the design, then becomes the flight software itself (simulation embodied software). This means that the software generated must be generic enough so that changes can be easily implemented as the design progresses and the software can still operate under real-time constraints. Iteration was used to mitigate this conflict.

The authors reported that errors, usually very costly to find and correct, were eliminated early in the program using this process. Errors generated while translating simulation software into the software requirements document were eliminated as well. The only errors that survived undetected through software testing were those due to software inaccuracies and errors in the environment simulation. Development time was observed to decrease due to a combination of factors; flight software prototypes were available early

in the program and the software for both the simulation and actual flight software was developed only once. However, more effort was required in the preparation of simulation and the embedded software was nonoptimal from an execution viewpoint.

REVIEW OF SESSION IV:

This session was devoted to the investigation of methods of capturing and documenting requirements and then developing a design specification having few or no ambiguities. Papers 11 and 12 approach the problem using formal language specifications. This approach will solve the problem of ambiguity and, additionally, offers the promise of being able to produce code automatically from the formal specification. However, there are a number of problems with formal specifications that need to be resolved before formal specifications can be used extensively. First, formal languages are normally based on mathematical notions (such as predicate calculus) and are not as readable as they need to be by both the applications engineers and their management. Second, there is still a lack of support tools for formal methods. Third, operational properties, such as timing and space constraints and quality are not currently within the expressive power of formal languages. It is possible that these deficiencies will be overcome in the future, but probably not in the near future.

Paper 12 proposed an interim method for getting around some of these problems. J. Draper believes that by using formal specifications only in certain critical elements of a system, where the use of formal specifications are truly needed, one can benefit from the rigorous, unambiguous nature of the method, and yet not be burdened by the method in parts of the system where it is not required.

The methodology presented in paper 13 uses simulation to aid in the development of requirements specifications. The simulation software is then used as a rapid prototyping tool to develop the embedded software. This process solves some of the problems normally encountered when using current approaches.

3.5 SESSION V: DESIGN METHODS FOR REAL-TIME SOFTWARE

Chairman: Dr. A. BENOÎT (BE)

Paper 14: Network programming: a design method and programming strategy for large software systems

L. SCHUBERTH, J. KUTSCHER, W-J. GRUNEWALD, Forschungsinstitut für Funk und Mathematik, Wachtburg-Werthhoven, GE

This paper describes a methodology for developing and supporting large data processing systems. The methodology, called network programming, is a decentralized approach which makes it possible for the various parts to be developed independently, and for different languages to be used. The key to the process is the construction of channels that are used to communicate among the various parts of the large system software. The channels also act as buffers, changing the parameters of the interface communication data as necessary to enable the parts to communicate without errors.

The authors point out that large software systems, like guidance and control embedded software, are subject to changes in the hardware configuration, in requirements, or even in design goals as the development progresses. Network programming helps to cope with changes because it:

a. Supports the implementation of well-defined processes.
b. Allows processes to be written in different languages to communicate effectively.
c. Makes processes communicate effectively that are run on different machines and under different operating systems.
d. Supports changing programs, adding processes to the system, and removing processes from the system at minimal cost in implementation and test time.

An Ada-oriented workbench has been developed that uses the network programming methodology. It provides the tools to handle communication channels and the generators for the necessary test environments. Both the channel and test environment are developed using Ada.

Paper 15: The data oriented requirements implementation scheme

C. M. THOMAS, British Aerospace (Dynamics) Ltd., Stevenage, HERTS, UK

This paper describes the data-oriented requirements implementation scheme (DORIS) methodology and tool set. Recognizing that there are many tools supporting different parts of a system's life cycle, but there is not one set of integrated tools which covers all phases of the life cycle, DORIS is an applied research project at British Aerospace designed to remedy this situation.

DORIS is a set of integrated methods and associated tools that have been developed to support the specification, design, and development of real-time embedded software systems. DORIS uses two existing methods: CORE (controlled requirements expression) which is used in the definition phase, and MASCOT (modular approach to software construction, operation, and test) which is used in the design phase. Both CORE and MASCOT have been extended and adapted to permit their integration into the DORIS scheme. The data interaction architecture (DIA) is used to implement the hardware design.

Paper 16: Process/object-oriented ADA software design for an experimental helicopter

K. GRAMBOW, Elektronik-System, GmbH, München, GE

This paper describes how software for an experimental helicopter avionic system was developed using a method based on the Ada tasking model and object-oriented design principles. The avionic system is controlled by a multiprocessor system of 68030 processor boards. The avionic system includes all of the basic elements of flight control, guidance, and pilot control and displays. The software for the system was designed using a methodology derived from the one developed by K. Nielsen and K. Shumate at Hughes Aircraft Company. This methodology follows five steps, as summarized below:

a. Determine all processes/interactions at the top level.
b. Connect external devices via handler processes.
c Construct a shallow leveled data flow diagram (DFD).
d. Refine the DFDs until the concurrent processes are identified (in steps "c" and "d", the main avionic application is decomposed using the Yourdon/De Marco DFDs, not for functional analysis of the project, but for the software design itself).
e. Determine the rendezvous direction, showing the result in Ada task graphs.

The author points out that most real-time avionic systems are typically constructed using a global cyclical scheduler. This design approach was taken to guarantee that critical time constraints could be met by certain software functions. With larger applications, these functional and periodicity requirements are difficult to meet. The author further states that by using the inherent tasking features in Ada, the language itself provides all the features necessary for real-time scheduling.

The basic concern commonly expressed about using the Ada tasking model is that its nondeterministic nature cannot be used to develop software needing deterministic, time critical features. However, studies at the Software Engineering Institute have proved that all tasks will meet their deadlines without knowing exactly when any given task will be running if:

a. The software program stays within certain bounds of central processing unit utilization.
b. A "rate monotonic scheduling algorithm" (which gives each task a fixed priority, assigning higher priorities shorter periodicies) is used.
c. A "priority ceiling protocol" (which prevents deadlock situations and unwanted priority inversions to occur) is used.

Another concern expressed is that present Ada compilers add an unacceptable amount of overhead on the software run-time schedules. SIGAda performance working group reports show that compilers now have good performance; a task switch can be completed in less than 50us, a null rendezvous less than 100us, and clock resolution less than 200us.

Paper 17: Code generation for fast DSP-based real-time control

H. HANSELMANN, A. SCHWARTE, H. HENRICHFREISE, dSPACE digital signal processing and control engineering, Paderborn, GE

This paper describes how a commercially available tool set (DSP-CITpro) is being used to automatically generate code for digital single-chip processors (DSP). Examples of DSP applications include servohydraulic actuators, active damping of flexible structures, and other motion controllers. The DSP-CITpro complements existing control design tools by closing the gap between the design and the implementation of the design. Because the DSP-CITpro can model the actual design, it can be used also for real-time system simulation (hardware-in-the-loop simulation).

Paper 18: Conception assistée par ordinateur du pilotage et du guidage de systèmes d'armes utilisant la technique de commande fonctionnelle prédictive PFC (Computer-aided design of weapon system guidance and control with Predictive Functional Control technique.)

D. CUADRADO, P. GUERCHET, THOMSON-CSF/DSE, Bagneux, FR; and S. ABU EL ATA-DOSS, ADERSA Verrières-le-Buisson, FR

This paper describes how the predictive functional control (PFC) technique has been used to design control laws in two simulation applications: (1) a high velocity missile operating in the initial, pursuit, and terminal phases, and (2) a turret homing device for a very short-range weapon system. The PFC technique, with an associated computer-aided design (CAD) tool, permitted the designers to not only improve the dynamic performance, but also to analyze and design into the system techniques for rejecting the effect of unscheduled perturbations, such as wind blasts and inertia variation. For example, the miss distance for the missile guidance law was reduced between 30% and 60%. For the turret control system, good performance and robustness was achieved under a wider set of operating conditions than the previous turret control system design.

Paper 19: Analyst Workbench

T. F. REESE, F. P. ARMOGIDA, Naval Weapons Center, China Lake, CA, US

This paper describes an analysis tool developed by the Naval Weapons Center for analyzing flight test and simulation test data to study the performance and effectiveness of missile systems. The tool takes normal test data (telemetry or otherwise) and stores the information (which can be large quantities of data) on hard disk drives. The stored data can then be interactively queried, analyzed, and manipulated through special support programs resident in the workstation.

This workstation has not only enhanced personnel productivity, but has enhanced the communication between analysts and also between the analysts and their management.

REVIEW OF SESSION V:

The network programming method described in paper 14 has many advantages. There are other systems similar to this one, such as the network computer system (NCS), that have been proven effective in the development of large software systems. However, if this kind of approach is used to develop guidance and control systems there is a basic problem; it is difficult, if not impossible, to control the timing of interacting processes, a function critical in many real-time guidance and control functions.

The DORIS system, described by C. M. Thomas in paper 15, is a significant prospect for the future. It is being designed to be language, host, and processor independent. Once the system is well integrated and proven, DORIS should provide excellent improvements in productivity.

Paper 16 (and papers 14 and 15) provide compelling reasons to believe that, with the proper environment of methods and protocols, Ada can be used effectively to design and develop guidance and control software programs.

Although there is little new in using a microprocessor-based code generator for small, embedded applications (as described in paper 17), it is interesting to note the effectiveness of the commercially available tool set for designing and developing software for closed-loop controllers found in guidance and control systems. Papers 18 and 19 did not address the issues of software design. However, paper 18 was an informative paper on a method of designing and rapidly prototyping complex control laws in guidance and control systems.

3.6 SESSION VI: ADA APPLICATIONS

Chairman: Dr. J. NIEMELA, (US)

Paper 20: Une expérience pratique de l'utilisation d'ADA pour le développement du logiciel embarqué. (Practical experience of ADA for developing embedded software)

C. GOETHALS, C. GRANDJEAN, DASSAULT ELECTRONIQUE, Saint Cloud, FR

This paper is a "lessons learned" report made by the RAFALE software development team. They used Ada throughout the avionic system, including most of the real-time executive system.

The RAFALE has two mission computers, three data buses, and standard tactical subsystems including; radar, HUD, INUs, and multifunction displays. Each mission computer contained about 1 megabyte of software. Part of the software was designed using the process-driven method, and the other part (the man-machine interface) of the software was designed using the object-oriented design method. The RAFALE design team used their own real-time executive that complied to rate monotonic scheduling principles.

Concerns about many of the aspects of Ada led the team to concentrate on assembling and adapting an impressive set of software support tools. For specification development support, they used the STP (software through picture) adapted to operate with several modeling techniques, including structured analysis (Yourdon/DeMarco), structured design (Constantine), structured real-time analysis (Hatley), and entity-relationship model (Chen). They used the KEYONE tool to support the detailed design and coding phase and they used DEVISOR for the software debug and test phase. DEVISOR was also used in the validation phase, as was the software validation bench (SVB). The SVB simulates the computer environment and was used to validate the complete operational software.

The RAFALE software development team members report that they experienced an overall gain in productivity of approximately 30%. However, they also report that the amount of effort spent in the architectural design phase increased significantly, but they do not quantify this increased effort.

Paper 21: The development of a requirement specification for an experimental active flight control system for a variable stability helicopter—an ADA simulation in JSD

G. PADFIELD, RAE, Flight Dynamics Division, Bedford, UK; R. BRADLEY, University of Glasgow, Dept. of Aerospace Engineering, UK; A. MOORE, LBMS, UK

This paper describes how an active control technology (ACT) system is being developed for an experimental Lynx helicopter by the Royal Aerospace Establishment. The development process starts with the capturing of the system requirements, progresses to the development of a system design specification, and ends with software code automatically generated for a PC-hosted simulation. The entire process is supported using the Jackson System Development (JSD) methodology and CASE tools. The features of the new ACT system include: a full authority fly-by-wire actuation system, a safety pilot with back-driven controls; a fail-operate/fail-safe hardware architecture coupled to a variety of sensors; and a pilot interface (sidestick controller, displays) providing inputs to the control laws.

Early in the development process, the ACT system and the supporting simulation environment were broken into functional elements (or modules) that would correspond to elements in the actual helicopter avionics system; or, in the case of the simulation environment, that would make it possible to rapidly change the simulation program. Descriptions of the elements were prepared, including the element TYPE (analog, digital, mechanical, etc.), FUNCTION (what the element does), OPERATION (narrative), PERFORMANCE (task timing and accuracy), INPUTS and OUTPUTS (signals), INTERFACES (unit to unit), TESTING (how function is to be verified), and FAILURE REPORTING AND RECOVERY (how errors are reported and how the system will recover). Once the basic specification was complete, a full JSD specification was developed which:

a. Provided a check on specification ambiguity, vagueness, and outright errors.
b. Made it possible to generate code automatically for the simulator.
c. Provided a "living design specification" for the ACT as the design was interactively tested and modified.
d. Eventually, generated proven code automatically for the flight software program.

Paper 22: Withdrawn

Paper 23: Software methodologies for safety critical systems

W. C. DOLMAN, A. M. ASHDOWN, Lucas Aerospace, Electronic Systems Division, Birmingham, UK; T. C. MOORES, Ministry of Defence, London, UK

This paper discusses a program sponsored by the Ministry of Defence in the United Kingdom to analyze and evaluate the suitability and the problems of using Ada language in the flight safety critical systems. The stated concern of the Ministry of Defence was that Ada was not yet ready for incorporation into full development of high integrity software based systems. The Ministry of Defence awarded a contract, called the High Order Language Demonstrator (HOLD), to Lucas Aerospace to study the issue.

The study called for implementation of Ada in an existing, flight validated, safety critical function, and:

a. Identifying those features of the Ada language which conflict with the requirements for a flight safety "critical" aeroengine control system.
b. Providing a critical assessment of the design and development methods that will provide the best possible application of the language to meet both performance and integrity requirements.
c. Assessing the efficiency of the executable code and the resulting system performance and integrity using an existing flight certified electronic engine control (EEC) as a benchmark.

The program is about 75% complete. However, the program has progressed enough for the authors to conclude that using the Yourdon methods is an improvement over using past development processes. Also, the Ada run-time system is completely compatible for use in future engine control systems.

Paper 24: CAMP: common ADA missile packages

B. MULLINS, Airforce Armament Laboratory, Eglin AFB, FL, US

The CAMP program was started in September 1984 to determine if sufficient commonality existed within the missile operational flight software domain to warrant the development of reusable software parts and, if so, to identify and establish methods and tools to use these parts in future missile development programs. In phase 1 of the program, it was proven that commonality does exist within the ARMONICS (armament electronics) domain (missiles). During this first phase, 219 reusable parts were identified and designed. In phase 2, reusable parts were built and tested. An "11th" missile was designed using reusable parts identified from 10 existing missile systems. This part of the program proved that, by using the reusable parts alone, a 15% increase in productivity could be gained. This program also proved that a 28% productivity increase could be gained if the reusable parts and the engineering tools and methods were both used. The program is now in phase 3. The purpose of this phase is to improve the reusable parts and the methods and tools.

Paper 25: Development and Verification of Softwre for Flight Safety Critical Systems

H. AFZALI, A. MATTISSEK, LITEF GmbH, Freiburg, GE

This paper discusses how two flight critical systems for the EFA are being developed using Ada. The paper also describes how system safety analysis techniques are being used to identify, assess, and eliminate

(or minimize) safety hazards. One of the applications is the pilot ejection control for the LITEF (a triple-redundant seat sequencer system), and the other application is a fail-op/fail-op flight control self-aligning attitude and heading reference system for the inertial measurement unit (a quad-redundant system) for the EFA flight control system.

The overall objectives of the safety analysis were "to identify hazards, to eliminate the hazards (if possible) through design or by reducing the associated risk to an acceptable level, and to minimize the (consequences of) the hazardous events." LITEF is using the traditional fault tree, scenario tree, and FMECA methods during the requirements analysis and detailed design phases: and the static and dynamic code analysis methods during the test and verification phase. The objective of the static code analysis is to identify the deficiencies in the data flow, control flow, and information flow. The objective of the dynamic code analysis is to verify that the test cases selected provide sufficient coverage to validate the software.

REVIEW OF SESSION VI:

Unlike the conclusions drawn in paper 16 where productivity increased due to using Ada, paper 20 concludes that increased productivity came from using good software development support tools and not as the result of using Ada. The two papers are complementary, however. In both instances, full Ada and good software support tools were used. The increased productivity was probably due to the combination of Ada and good tools.

The ACT system described in paper 21 has proven to be an excellent tool for designing and developing closed-loop flight control systems. While there were benefits in developing the system design specification, the greatest benefit was in code generation for the simulation program and flight software, if and when the program flight tests the experimental Lynx helicopter.

There was considerable debate about paper 23. First, there was a repeat discussion (from an earlier session) about the relative value of using formal methods and formal specifications. All agreed that formal methods would lead to unambiguous specifications, but everyone did not agree on whether or not formal methods should be used at all and, if so, to what extent. Most people agreed that, given the present status of formal methods, they should be used, but should not be imposed on all parts of a project. Paper 12 proposed this approach as well. A second debate centered (again) on using Ada in flight critical software. This question is basically the issue being investigated in paper 23. Most of the symposium attendees emphatically agreed that Ada is here and will be used. As one attendee said, "Ada is being used successfully and will continue to be used exclusively on the EFA." It must be pointed out, however, that this study will provide valuable insight into the problems of validating flight critical systems.

Paper 25 differed greatly from paper 23. This paper suggests that high integrity, flight critical systems can be developed and verified by using a restricted subset of Ada (a technique also suggested by others) and carefully applying proven, traditional safety analysis techniques.

3.7 SESSION VII: AUTOMATED SOFTWARE GENERATION APPROACHES

Chairman: Dr. E. B. STEAR (US)

Paper 26: Reusable software approach to software generation

A. P. DeTHOMAS, WRDC/FIGX,Wright-Patterson AFB, OH, US; D. DEWEY, Boeing Military Systems, Seattle, WA, US; S. WILSON, LOCUS Co., Fairfax, VA, US

This paper begins with a discussion of where and how the reusable software approach can be applied and what benefit the method could, theoretically, have on software generation productivity. A reusable software analysis model was used to estimate the cost benefits that might be expected. The model indicates that the development cycle costs could be reduced by 25% to 45%, and total life cycle costs would show an improvement of 11% to 26%.

WG10 made a survey of the current work going on within the NATO countries using the reusable software approach. Included in the survey were the CAMP, STARS, SCR, GRACE, CARE, ESPRIT, SPC, and the Eureka Software Factory (ESF) programs.

The authors conclude that the reuse concept can be applied across all phases of the design process (design through testing), but the reuse methodology must be built into the development process before a project is started and cannot be an afterthought. This means that the reuse system must possess certain characteristics or mechanisms to fully exploit software reuse. These mechanisms include:

a. Storage and retrieval of reusable objects (implying use of a library).
b. A specification process that provides an understanding of the object within the library.
c. Tools for tailoring reusable object to the current application.
d. Tools for integrating the objects into a new system.

WG10 believes that the lack of well-designed libraries of reusable software is a major limiting factor in the wide acceptance and use of reusable software. A second limiting factor and possibly the most important one, is how to handle the business aspects of reusing software. These issues include software developed by

one business group and used by another, who will bear the first-time development costs, and liabilities in terms of performance, safety, and maintenance.

WG10 concluded that there are no major technological breakthroughs required. However, several technology enhancements will speed the application of software reuse. Enhancements include: adopting existing development methodologies and standard interfaces, new tools for accessing and maintaining reuse libraries, and refinement of cost models to qualify reuse options. A strategy for infusing reuse technology was also provided. The three-stage process includes enhancement of the reuse technology (as previously discussed), demonstration of relevant GNC programs, and introduction of the results into the user community.

Paper 27. Fourth generation languages

P. CHINN, Marconi Defence Systems, Bucks, UK

This paper begins by tracing the evolution of the fourth generation language (4GL) concept from the first through the third generation languages. The 4GL appproach can be thought of as a combination and evolution of the rapid prototyping and integrated program support system concepts. Although it is difficult to consolidate the many definitions, 4GLs share three common characteristics. They are:

a. A close relationship to the task for which they are to be used (such as application- specific constructs).
b. An easy interface with the user.
c. Extensive use of modern support tools.

A 4GL approach to automatic software generation should provide the following benefits:

a. An improved product since the communication barrier between application engineer and the final code is removed.
b. Improved productivity, because the application-specific engineer can interact directly with the automated tools, using an application-specific language.
c. A greater consistency in the product.
d. Easier development and maintenance, which can be approached from a system engineering, rather than from a software engineering, viewpoint.

On the other hand, a 4GL approach has some limitations as well. For example, the user interface with the system is likely to be a proprietary workstation. If the intermediate level language is a standard engineering language, then support of the target types will depend on that implementation. Another potential

problem is that a 4GL system will have many components requiring periodic update or maintenance and keeping these compatible will be difficult.

Most of the technology and components for a 4GL system exists. However, some of the components are only in the embryonic stage. Overall, the members of WG10 are "cautiously optimistic" about the possibility that a full 4GL automatic software generator for guidance and control applications will one day be used. Some limited systems have been developed. However, pervasive use of this technology will depend on the stability of the application domain and hardware and projected use to warrent the investment.

Paper 28. Méthodes de transformation (Transformation Methods)

P. de BONDELI, Aerospatiale—STS/L, Les Mureaux, FR; M. LEMOINE, ONERA-CERT/DERI, Toulouse, FR

Transformational programming is defined in the WG10 report as a methodology of program construction by successive applications of transformation (conversion) rules. A transformation rule is a formal mapping between two programs (P and P') which preserves the initial functionality. The transformational programming process, as defined in the WG10 report, starts with a formal specification and ends with executable efficient code.

Transformational programming is being endorsed as a way to solve current software development problems. Ideal software development follows a waterfall model (B. Boehm), where one sequence follows another similar to the development cycle of an aircraft system. However, the process breaks down in the development of software because requirements are not fully known at the beginning of the program and they often change as the program progresses. If transformational programming is to succeed as a major element in automatic software generators, then it will be necessary to follow a more appropriate methodology.

There are a number of transformational programming initiatives at the present time. These include initiatives at the Information Science Institute of the University of Southern California (SAFE and TI), PSI (LIBRA, PMB, PECOS), Edinburgh (investigation of hybrid rules), Stanford (DEDALUS), the Technical University of Munich (CIP), and others. One of the most significant ongoing system projects using transformational programming is a software technology project sponsored by the ESPRIT program. Initiatives within this software technology project, such as the PROSPECTRA, TOOLUSE, RAISE, and REPLAY, emphasize transformational programming.

Some members of industry (such as, IBM Hursley Laboratory, Bull in France, ONERA/CERT/DERI (see paper 11), and Lockheed/LASC) have been experimenting with program transformations. There are many benefits to using program transformations including establishing proved programs, adoption and maintenance of programs, and synthesizing several algorithms from a single specification. Although limited experiences have shown that considerable improvements can be achieved, there are several areas which must be improved; such as, the development of more comprehensive support tools and training of personnel.

Paper 29: Knowledge-based approach to software generation

W. MANSEL, MBB, Deutsche Aerospace, Ulm, GE; H. ROSCHMANN, TST, Deutsche Aerospace, Ulm, GE

WG10 studied the use of expert systems in software generation to:

a. Support the software development process itself; this includes functions such as tutoring, user guidance, documentation, and information retrieval.
b. Identify and retrieve reusable software components; this includes functions such as pattern matching and deciphering semantic descriptions. Examples of this concept are the CAMP (see paper 24) and PRACTITIONER (an ESPRIT project).
c. Assist in the transformation process when applying formal methods. Examples of this function are STES and REFINE.
d. Be a part of the software generation and automated programming function itself.

WG10 has conducted a review of the current status within the main areas listed above and has identified some of the problems and issues requiring further research. While there is a real need to use knowledge-based approaches, there is a continuing problem of verification and validation. A rapid prototyping facility needs to be incorporated into the process and advanced user interface and rrepresentation methods are required in future 4GL systems.

PANEL REVIEW AND ANALYSIS OF SESSION VII:

After WG10 completed their report, a panel of selected members of the WG was convened with Prof. J. T. Shepherd as panel moderator.

There was only one comment concerning the reusability report. An observation was made that standard software modules and interfaces are already occurring (de facto) in the personal computer world (the inference was that this practice may spread into the guidance and control domain).

A concern was expressed about how 4GL (which are usually, by nature, domain specific) will be used and combined in software generators for complete aircraft systems. It was suggested by a WG10 panel member that an interface would be required similar to that designed by Pragma program to interface other languages with Ada. A simple transformation also might work in some cases, but certainly not in those instances where a formal method is being applied.

Again, problems about the transformation methods were discussed by the audience and WG10 panel members. A strong point was made and agreed upon by those discussing the issue that, while transformation methods have desirable features and are valuable tools in certain applications, it will be necessary to have an informal reading accompany the formal specification.

Since the software generation methods discussed are heavily dependent on the development and use of tools, a concern was expressed on flight safety issues. WG 10 stated that the tools must be verified to the same level as the flight critical requirement to enable its use in the V&V and certification process.

Mr. K. Helps, a member of the WG10 panel, made a provocative statement in the discussion period; he pointed out that, even though the panel working group found promising ways to increase software development productivity by as much as two times using automatic software generators, this fell far short of the hoped-for goal of ten times improvement. It is believed, however, that combinations of the methods investigated (expert systems combined with reusability, for example) will provide significant increases in software productivity. WG 10 took a conservative approach in estimating payoffs. The real proof of the payoffs from these methods will be the demonstration and quantification of results.

4.0 CONCLUSIONS AND RECOMMENDATIONS

4.1 CONCLUSIONS

The suitability of using Ada in high-integrity, flight critical applications continues to be debated. However, most people at the symposium appeared to support the position that Ada, or at least Safe Ada, is not only usable in flight critical applications, but has desirable features for these applications (papers 2, 16, 20, and 25 all supported the position that Ada was suitable in these applications). W. M. Fraefrich, in paper 2, referred to the Eurofighter study which demonstrated that the reliability of Ada programs is comparable to that of assembler programs (if not greater) if Safe Ada is strictly adhered to and the static code analysis at source code level is made. Paper 25 approached the problem differently; H. Afzali and Dr. A. Mattissek suggest (as did paper 2) that flight critical systems can be developed by using Safe Ada, but added the qualifier that traditional safety analysis techniques must be carefully applied to ensure integrity. Concerns about the suitability of Ada in time critical, deterministic applications (such as flight control) were addressed by W.

Fraedrick. In his paper 16, he refers to a study conducted by the Software Engineering Institute that proves all tasks will meet their deadlines if certain design principles are followed. C. Goethals and C. Grandjean do not claim benefits or penalties from using Ada, but they do say that, with superior support tools, they were able to design a high-integrity system with increased productivity. Paper 23 authored by W. Dolman, A. Ashdown, and T. Moores, reported on an ongoing program sponsored by the United Kingdom Ministry of Defence to resolve some of the remaining questions about the applicability of Ada in G&C applications.

A second conclusion is that there have been significant advances recently in the evolution of software development support tools and methods for G&C applications. Most notable are the CORE/EPOS, HOOD, SPARK examiner, TESTBED, IPSE, AGLAE, and DORIS. The successes being achieved using the object-oriented design method (HOOD, DORIS, etc.) are most encouraging.

The third conclusion is that it is evident there is a shift occurring in the overall design process approach. It has been established dogma that the standard "V" process was to be strictly followed if complex, high-integrity software was to be developedon time and on schedule. However, many people are beginning to question the wisdom of this rule. D. F. Thewlis captures this idea in paper 1, and presents compelling arguments for considering a simultaneous top-down and bottom-up approach. G. Fernandez de la Mora, R. Minguez, S. Khan, and J. Villa are proposing a "phased" or "simulation-based development" approach, where the software requirements and design are iteratively analyzed throughout the process. The software used to simulate the design during the analysis process then becomes the flight software system. Most rapid-prototyping tools enable the software designer to iteratively progress up and down the "V" chart process. This capability will probably bring a change in the overall development process logic.

4.2 RECOMMENDATIONS

There needs to be a serious look at the possibility of establishing standard interfaces among the various Ada support tools. Good interface definitions are required before the various individual process support tools can be linked together to support all phases of G&C software development. Efforts to link together a full set of support tools (like the DORIS system described by C. Thomas) requires good interfaces, but standard interfaces are needed if there is to be a rapid evolution of effective software development tools and processes. Efforts like DORIS should be carefully followed as they mature and reported to the G&C design community. The AGARD GCP could be instrumental in providing an effective communications base for possible Ada ASPE standards.

Formal methods for developing high-integrity, flight safety software for G&C systems have great potential. However, there are a number of problems that must be overcome before they can be used effectively. Further research is recommended in this promising technique.

WG10 concluded that automated software generators based on reusable software, expert systems, 4GL, or transformation methods alone would provide an increase in productivity. However, it was concluded that significant improvements (perhaps twice the productivity or more) could only be achieved in an approach that combined two or more of the concepts. It would be very useful if WG10 or some other GCP group would quantify the productivity improvements using the combinational approach.

APPENDIX: PARTICIPANT EVALUATION OF THE SYMPOSIUM

At the end of the symposium, each participant was asked to fill out an evaluation form and make comments on their opinions about the symposium. From 31 evaluation forms returned, the responses are summarized below:

Question 1: In your view did the papers presented meet the published objectives of the meeting?

Answer:	Most 26	About half 5	Very few 0

Question 2: Were the topics selected for presentation of interest to you?

Answer:	Most 16	About half 13	Very few 2

Question 3: Was the general level of the papers:

Answer:	Too deep 0	Satisfactory 25	Too superficial 6

Question 4: Were the speakers effective in presenting their topics?

Answer:	Most 16	About half 14	Very few 1

Question 5: Time was allowed for discussion after each presentation and at the end of the meeting. Was this:

Answer:	Too little 7	About right 24	Too much 0

Question 6: Was the meeting effectively organized (location, joining instructions, duration, audiovisual equipment, refreshments, etc.)?

Answer: Yes 26 (However, some said "but"...viewgraphs were hard to read; some transportation problems from hotel to symposium.)
No 5 (Same comments as above.)

Question 7: Did language cause a problem?

Answer: Yes 4 (Two people said that good viewgraph projectors would have alleviated the problem.)

No 27

Question 8: Please add any other comments you may have:

1) Four people suggested that a list of attendees, prepared and distributed after checking, would be helpful in getting acquainted with other attendees.

2) It was suggested that a small book on the mission of AGARD and the various groups sponsored by AGARD be handed out at the beginning of the symposium.

3) Two people suggested a get-acquainted party on the first evening of each symposium.

Question 9: Overall, what was you assessment of the meeting?

Answer: Excellent 0 Very good 14 Good 15
Satisfactory 2 Poor 9

Question 10: Please add here any suggestions you have for followup activity in this field, or for AGARD lecture series, courses, symposiums, etc. in other fields.

Suggestions: Almost all of the suggestions offered for future discussion centered around the issues of software certification in flight safety critical systems and verification and validation of software development tools. Clearly, these issues still have not been resolved and will continue to be a major concern in the future.

REPORT DOCUMENTATION PAGE

1. Recipient's Reference	2. Originator's Reference	3. Further Reference	4. Security Classification of Document
	AGARD-AR-302 1/11	ISBN 92-835-0647-2 2/08	UNCLASSIFIED

5. Originator Advisory Group for Aerospace Research and Development
North Atlantic Treaty Organization 1/14
7 rue Ancelle, 92200 Neuilly sur Seine, France

6. Title TECHNICAL EVALUATION REPORT ON THE GUIDANCE AND CONTROL PANEL 52ND SYMPOSIUM ON SOFTWARE FOR GUIDANCE AND CONTROL. 1/18

7. Presented at

8. Author(s)/Editor(s)
~~Donald E.~~ Dewey D.E. 2/09

9. Date
December 1991 1/12 1/13

10. Author's/Editor's Address
Boeing Military Airplanes
Box 3707, MS 4C-15
Seattle, WA 98124, United States

11. Pages
38

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

Guidance and control
Flight control
ADA — programming language
Formal methods
Expert systems
Software support environments
Programming languages
Design
Real time operations
Computer systems programs

14. Abstract

Technical Evaluation Report on the Guidance and Control Panel's 52nd Symposium held at the Helexpo, Thessaloniki, Greece from 7th to 10th May 1991.

In all, 27 papers were presented covering the following headings:

— Tools and methods for a user's viewpoint;
— General requirements on software;
— Integrated programmes support environments;
— Software requirements;
— Design methods for real-time software;
— ADA applications;
— Automated software generation approaches.

AGARD Advisory Report 302
Advisory Group for Aerospace Research and Development, NATO
TECHNICAL EVALUATION REPORT ON THE GUIDANCE AND CONTROL PANEL 52ND SYMPOSIUM ON SOFTWARE FOR GUIDANCE AND CONTROL
by Donald E. Dewey
Published December 1991
38 pages

Technical Evaluation Report on the Guidance and Control Panel's 52nd Symposium held at the Helexpo, Thessaloniki, Greece from 7th to 10th May 1991.

P.T.O.

AGARD-AR-302

Guidance and control
Flight control
ADA — programming language
Formal methods
Expert systems
Software support environments
Programming languages
Design
Real time operations
Computer systems programs

AGARD Advisory Report 302
Advisory Group for Aerospace Research and Development, NATO
TECHNICAL EVALUATION REPORT ON THE GUIDANCE AND CONTROL PANEL 52ND SYMPOSIUM ON SOFTWARE FOR GUIDANCE AND CONTROL
by Donald E. Dewey
Published December 1991
38 pages

Technical Evaluation Report on the Guidance and Control Panel's 52nd Symposium held at the Helexpo, Thessaloniki, Greece from 7th to 10th May 1991.

P.T.O.

AGARD-AR-302

Guidance and control
Flight control
ADA — programming language
Formal methods
Expert systems
Software support environments
Programming languages
Design
Real time operations
Computer systems programs

AGARD Advisory Report 302
Advisory Group for Aerospace Research and Development, NATO
TECHNICAL EVALUATION REPORT ON THE GUIDANCE AND CONTROL PANEL 52ND SYMPOSIUM ON SOFTWARE FOR GUIDANCE AND CONTROL
by Donald E. Dewey
Published December 1991
38 pages

Technical Evaluation Report on the Guidance and Control Panel's 52nd Symposium held at the Helexpo, Thessaloniki, Greece from 7th to 10th May 1991.

P.T.O.

AGARD-AR-302

Guidance and control
Flight control
ADA — programming language
Formal methods
Expert systems
Software support environments
Programming languages
Design
Real time operations
Computer systems programs

AGARD Advisory Report 302
Advisory Group for Aerospace Research and Development, NATO
TECHNICAL EVALUATION REPORT ON THE GUIDANCE AND CONTROL PANEL 52ND SYMPOSIUM ON SOFTWARE FOR GUIDANCE AND CONTROL
by Donald E. Dewey
Published December 1991
38 pages

Technical Evaluation Report on the Guidance and Control Panel's 52nd Symposium held at the Helexpo, Thessaloniki, Greece from 7th to 10th May 1991.

P.T.O.

AGARD-AR-302

Guidance and control
Flight control
ADA — programming language
Formal methods
Expert systems
Software support environments
Programming languages
Design
Real time operations
Computer systems programs

In all, 27 papers were presented covering the following headings:

— Tools and methods for a user's viewpoint;
— General requirements on software;
— Integrated programmes support environments;
— Software requirements;
— Design methods for real-time software;
— ADA applications;
— Automated software generation approaches.

ISBN 92-835-0647-2

In all, 27 papers were presented covering the following headings:

— Tools and methods for a user's viewpoint;
— General requirements on software;
— Integrated programmes support environments;
— Software requirements;
— Design methods for real-time software;
— ADA applications;
— Automated software generation approaches.

ISBN 92-835-0647-2

In all, 27 papers were presented covering the following headings:

— Tools and methods for a user's viewpoint;
— General requirements on software;
— Integrated programmes support environments;
— Software requirements;
— Design methods for real-time software;
— ADA applications;
— Automated software generation approaches.

ISBN 92-835-0647-2

In all, 27 papers were presented covering the following headings:

— Tools and methods for a user's viewpoint;
— General requirements on software;
— Integrated programmes support environments;
— Software requirements;
— Design methods for real-time software;
— ADA applications;
— Automated software generation approaches.

ISBN 92-835-0647-2

AGARD
NATO OTAN
7 RUE ANCELLE · 92200 NEUILLY-SUR-SEINE
FRANCE
Téléphone (1)47.38.57.00 · Télex 610 176
Télécopie (1)47.38.57.99

DIFFUSION DES PUBLICATIONS
AGARD NON CLASSIFIEES

L'AGARD ne détient pas de stocks de ses publications, dans un but de distribution générale à l'adresse ci-dessus. La diffusion initiale des publications de l'AGARD est effectuée auprès des pays membres de cette organisation par l'intermédiaire des Centres Nationaux de Distribution suivants. A l'exception des Etats-Unis, ces centres disposent parfois d'exemplaires additionnels; dans les cas contraire, on peut se procurer ces exemplaires sous forme de microfiches ou de microcopies auprès des Agences de Vente dont la liste suite.

CENTRES DE DIFFUSION NATIONAUX

ALLEMAGNE
Fachinformationszentrum,
Karlsruhe
D-7514 Eggenstein-Leopoldshafen 2

BELGIQUE
Coordonnateur AGARD-VSL
Etat-Major de la Force Aérienne
Quartier Reine Elisabeth
Rue d'Evere, 1140 Bruxelles

CANADA
Directeur du Service des Renseignements Scientifiques
Ministère de la Défense Nationale
Ottawa, Ontario K1A 0K2

DANEMARK
Danish Defence Research Board
Ved Idraetsparken 4
2100 Copenhagen Ø

ESPAGNE
INTA (AGARD Publications)
Pintor Rosales 34
28008 Madrid

ETATS-UNIS
National Aeronautics and Space Administration
Langley Research Center
M/S 180
Hampton, Virginia 23665

FRANCE
O.N.E.R.A. (Direction)
29, Avenue de la Division Leclerc
92320, Châtillon sous Bagneux

GRECE
Hellenic Air Force
Air War College
Scientific and Technical Library
Dekelia Air Force Base
Dekelia, Athens TGA 1010

ISLANDE
Director of Aviation
c/o Flugrad
Reykjavik

ITALIE
Aeronautica Militare
Ufficio del Delegato Nazionale all'AGARD
Aeroporto Pratica di Mare
00040 Pomezia (Roma)

LUXEMBOURG
Voir Belgique

NORVEGE
Norwegian Defence Research Establishment
Attn: Biblioteket
P.O. Box 25
N-2007 Kjeller

PAYS-BAS
Netherlands Delegation to AGARD
National Aerospace Laboratory NLR
Kluyverweg 1
2629 HS Delft

PORTUGAL
Portuguese National Coordinator to AGARD
Gabinete de Estudos e Programas
CLAFA
Base de Alfragide
Alfragide
2700 Amadora

ROYAUME UNI
Defence Research Information Centre
Kentigern House
65 Brown Street
Glasgow G2 8EX

TURQUIE
Millî Savunma Başkanlığı (MSB)
ARGE Daire Başkanlığı (ARGE)
Ankara

LE CENTRE NATIONAL DE DISTRIBUTION DES ETATS-UNIS (NASA) NE DETIENT PAS DE STOCKS DES PUBLICATIONS AGARD ET LES DEMANDES D'EXEMPLAIRES DOIVENT ETRE ADRESSEES DIRECTEMENT AU SERVICE NATIONAL TECHNIQUE DE L'INFORMATION (NTIS) DONT L'ADRESSE SUIT.

AGENCES DE VENTE

National Technical Information Service
(NTIS)
5285 Port Royal Road
Springfield, Virginia 22161
Etats-Unis

ESA/Information Retrieval Service
European Space Agency
10, rue Mario Nikis
75015 Paris
France

The British Library
Document Supply Division
Boston Spa, Wetherby
West Yorkshire LS23 7BQ
Royaume Uni

Les demandes de microfiches ou de photocopies de documents AGARD (y compris les demandes faites auprès du NTIS) doivent comporter la dénomination AGARD, ainsi que le numéro de série de l'AGARD (par exemple AGARD-AG-315). Des informations analogues, telles que le titre et la date de publication sont souhaitables. Veuiller noter qu'il y a lieu de spécifier AGARD-R-nnn et AGARD-AR-nnn lors de la commande de rapports AGARD et des rapports consultatifs AGARD respectivement. Des références bibliographiques complètes ainsi que des résumés des publications AGARD figurent dans les journaux suivants:

Scientifique and Technical Aerospace Reports (STAR)
publié par la NASA Scientific and Technical
Information Division
NASA Headquarters (NTT)
Washington D.C. 20546
Etats-Unis

Government Reports Announcements and Index (GRA&I)
publié par le National Technical Information Service
Springfield
Virginia 22161
Etats-Unis

(accessible également en mode interactif dans la base de données bibliographiques en ligne du NTIS, et sur CD-ROM)

Imprimé par Specialised Printing Services Limited
40 Chigwell Lane, Loughton, Essex IG10 3TZ

AGARD
NATO OTAN
7 RUE ANCELLE · 92200 NEUILLY-SUR-SEINE
FRANCE
Telephone (1)47.38.57.00 · Telex 610 176
Telefax (1)47.38.57.99

DISTRIBUTION OF UNCLASSIFIED AGARD PUBLICATIONS

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 Aérienne
Quartier Reine Elisabeth
Rue d'Evere, 1140 Bruxelles

CANADA
Director Scientific Information Services
Dept of National Defence
Ottawa, Ontario K1A 0K2

DENMARK
Danish Defence Research Board
Ved Idraetsparken 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)

LUXEMBOURG
See Belgium

NETHERLANDS
Netherlands Delegation to AGARD
National Aerospace Laboratory, NLR
Kluyverweg 1
2629 HS Delft

NORWAY
Norwegian Defence Research Establishment
Attn: Biblioteket
P.O. Box 25
N-2007 Kjeller

PORTUGAL
Portuguese National Coordinator to AGARD
Gabinete de Estudos e Programas
CLAFA
Base de Alfragide
Alfragide
2700 Amadora

SPAIN
INTA (AGARD Publications)
Pintor Rosales 34
28008 Madrid

TURKEY
Milli Savunma Başkanlığı (MSB)
ARGE Daire Başkanlığı (ARGE)
Ankara

UNITED KINGDOM
Defence Research Information Centre
Kentigern House
65 Brown Street
Glasgow G2 8EX

UNITED STATES
National Aeronautics and Space Administration (NASA)
Langley Research Center
M/S 180
Hampton, Virginia 23665

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 Headqu
Washington D.C
United States

Government Reports Announcements and Index (GRA&I)
published by the National Technical Information Service
Springfield