AFWAL-TR-85-1016 VOL V

(2)

WRIGHT RESEARCH & DEVELOPMENT CENTER

EVALUATION & VALIDATION (E&V)

TEAM PUBLIC REPORT
Volume V

RAYMOND SZYMANSKI
E&V TEAM CHAIRMAN
AVIONICS LABORATORY
AIR FORCE WRIGHT AERONAUTICAL LABORATORIES
WRIGHT-PATTERSON AIR FORCE BASE, OHIO 45433-6543

October 1990

Interim Technical Report for Period

01 January 1989 - 01 September 1990

**APPROVED FOR PUBLIC RELEASE
DISTRIBUTION UNLIMITED**

PREPARED FOR:

ADA* JOINT PROGRAM OFFICE

3D139 (FERN ST/C107) PENTAGON

WASHINGTON, D.C. 20301

Unclassified
SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1a. REPORT SECURITY CLASSIFICATION Unclassified	1b. RESTRICTIVE MARKINGS
2a. SECURITY CLASSIFICATION AUTHORITY	3. DISTRIBUTION/AVAILABILITY OF REPORT Approved for Public Release; Distribution Unlimited
2b. DECLASSIFICATION/DOWNGRADING SCHEDULE	
4. PERFORMING ORGANIZATION REPORT NUMBER(S) AFWAL-TR-85-1016, Vol V	5. MONITORING ORGANIZATION REPORT NUMBER(S)
6a. NAME OF PERFORMING ORGANIZATION Wright Research and Development Center 6b. OFFICE SYMBOL (If applicable) WRDC/AAAF-3	7a. NAME OF MONITORING ORGANIZATION
6c. ADDRESS (City, State, and ZIP Code) Wright-Patterson Air Force Base, OH 45433-6543	7b. ADDRESS (City, State, and ZIP Code)
8a. NAME OF FUNDING/SPONSORING ORGANIZATION Ada* Joint Program Office 8b. OFFICE SYMBOL (If applicable)	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER
8c. ADDRESS (City, State, and ZIP Code) 3D139 (Fern Street/C107) Pentagon Washington, D.C. 20301	10. SOURCE OF FUNDING NUMBERS PROGRAM ELEMENT NO. 63226 · PROJECT NO. AJPO · TASK NO. 28 · WORK UNIT ACCESSION NO. 53

11. TITLE (Include Security Classification)

Evaluation and Validation (E&V) Team Public Report, Volume V

12. PERSONAL AUTHOR(S)
Raymond Szymanski, E&V Team Chairman

13a. TYPE OF REPORT	13b. TIME COVERED	14. DATE OF REPORT (Year, Month, Day)	15. PAGE COUNT
Interim Technical	FROM 1 JAN 89 TO 30Sep90	1990 October 31	454

16. SUPPLEMENTARY NOTATION

*Ada is a Registered Trademark of the U.S. Government (Ada Joint Program Office)

17. COSATI CODES

FIELD	GROUP	SUB-GROUP
09	02	

18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
Ada* Common APSE Interface Set (CAIS)
Evaluation Ada Programming Support Environment (APSE)
Validation

19. ABSTRACT (Continue on reverse if necessary and identify by block number)

Activities and accomplishments of the evaluation and Validation (E&V) Team are reported for FY1989 and FY1990. The purpose of the E&V Task, which is sponsored by the Ada Joint Program Office (AJPO), is to develop techniques and tools that will provide a capability to perform assessment of Ada Programming Support Environments (APSEs) and to determine conformance of APSEs to the Common APSE Interface Set (CAIS). As this technology is developed, it is being made available to Department of Defense (DoD) components, industry, and academia.

20. DISTRIBUTION/AVAILABILITY OF ABSTRACT ☐ UNCLASSIFIED/UNLIMITED ☒ SAME AS RPT. ☐ DTIC USERS	21. ABSTRACT SECURITY CLASSIFICATION Unclassified	
22a. NAME OF RESPONSIBLE INDIVIDUAL Raymond Szymanski	22b. TELEPHONE (Include Area Code) (513) 255-6548	22c. OFFICE SYMBOL WRDC/AAAF-3

DD Form 1473, JUN 86 Previous editions are obsolete.

SECURITY CLASSIFICATION OF THIS PAGE
Unclassified

TABLE OF CONTENTS

SECTION 1 - Project Technical Summary 1

APPENDIX A - NAECON Paper A-1

APPENDIX B - New E&V Reference System Flyer B-1

APPENDIX C - The E&V'ing News C-1

APPENDIX D - Issues and Strategies for Evaluation and Validation of CAIS-A Implementations D-1

APPENDIX E - Final Report: Independent Validation and Verification (IV&V) on the Ada Compiler Evaluation Capability (ACEC) E-1

APPENDIX F - E&V Task Presentations F-1

APPENDIX G - Abstract of Supporting Selection Decisions Based on the Technical Evaluation of Ada Environments and Their Components G-1

APPENDIX H - Tools & Aids Document H-1

APPENDIX I - Minutes of the E&V Team Meeting December 1988 . . I-1

APPENDIX J - Minutes of the E&V Team Meeting February 1989 . . J-1

APPENDIX K - Minutes of the E&V Team Meeting June 1989 . . . . K-1

APPENDIX L - Minutes of the E&V Team Meeting September 1989 . . L-1

APPENDIX M - Minutes of the E&V Team Meeting December 1989 . . M-1

APPENDIX N - Minutes of the E&V Team Meeting March 1990 . . . N-1

APPENDIX O - Minutes of the E&V Team Meeting May 1990 . . . . O-1

SECTION 1

PROJECT TECHNICAL SUMMARY

1.1 Introduction

This report is the fifth in a series of technical reports to be published by the Evaluation and Validation (E&V) Project. The purpose of the E&V Public Report is to provide an overview of the many technical accomplishments of the E&V Project and E&V Team during an appropriate time frame. This fifth report contains information resulting from E&V activities during December 1988 to September 1990 which is being made available for public review and comment. Contents of this report reflect an observation of the E&V Project and E&V Team progress during the specified time frame and should not be viewed as final representations of the technology being developed.

1.2 Background

In June 1983, the Ada Joint Program Office (AJPO) proposed the formation of the E&V Task and a tri-service Ada Program Support Environment (APSE) E&V Team, with the Air Force designated as lead service. In October 1983, the Air Force officially accepted responsibility as lead service on the E&V Task.

The Ada community, including Government, industry, and academic personnel, needs the capability to assess APSEs and their components, and to determine their conformance to applicable standards (e.g., DOD-STD-1838, the Common APSE Interface Set (CAIS) standard). The technology required to fully satisfy this need is extensive and largely unavailable; it cannot be acquired by a single Government sponsored, professional society sponsored, or private effort. The purpose of the Evaluation and Validation Task is to provide a focal point for addressing the need by:

(1) Identifying and defining specific technology requirements,

(2) Developing selected elements of the required technology,

(3) Encouraging others to develop some elements, and

(4) Collecting information describing elements which already exist.

This information will be made available to Department of Defense (DoD) components, other Government agencies, academic institutions, and industry.

1.3 E&V Project Summary for the Reporting Period

The E&V Project accomplished several tasks during the period 01 January 1989 through 01 September 1990 which are summarized in subsections 1.3.1 - 1.3.4. Additional details can be found later in this report.

1.3.1 Evaluation and Validation Reference System

The E&V Reference System is a set of two complementary documents, the E&V Reference Manual and the E&V Guidebook, which provides detailed information on Ada Programming Support Environments and their assessment. The E&V Reference System Version 1.1 was released in late November 1988. This version has been distributed to a growing number of DoD agencies and the worldwide Ada community. Shortly after this release, work began on Version 2.0. In order to better serve the community of E&V Reference System users, a questionnaire was mailed out to those individuals who received copies of Version 1.0. This feedback substantially affected Version 2.0, particularly in the development and inclusion of additional evaluation checklists in the Guidebook section of the Reference System.

Distribution of the E&V Reference Manual Version 2.0 commenced in November of 1989. The current list of recipients numbers well over five hundred and users include every major DoD contractor, many commercial software development companies, and a substantial number of DoD agencies.

1.3.2 Ada Compiler Evaluation Capability (ACEC)

The Ada Compiler Evaluation Capability is a set of tests, test scenarios, and support tools which enable the user to determine the performance and usability characteristics of an Ada compilation system. The distribution of ACEC Version 1.0 was begun in October 1988. Since that time it has been delivered to over ninety users. These include DoD program offices, DoD laboratories, DoD contractors, and Ada compiler vendors. A partial list of the users and their projects are contained in the Presentation Material section of this report. Although not shown on that list, many Ada compiler vendors use the ACEC to exercise their products before releasing them for production use.

In May 1990, ACEC Version 2.0 distribution was begun. Version 2.0 contains an additional 300 performance tests and assessors for the program library system, diagnostic messages, and symbolic debugger. Also included is a Single System Analysis Tool for the comparison of related tests executed on a single system and enhanced user documentation. Plans for ACEC Version 3.0 are now complete. Details of these plans are contained later in this report.

1.3.3 CAIS Implementation Validation Capability (CIVC)

CIVC Version 1.0 is a set of tests, test scenarios, and support tools which enable the user to determine conformance of a CAIS (DOD-STD-1838) implementation to the CAIS standard. CIVC Version 1.0 was released January 1990 and is being used by the North Atlantic Treaty Organization (NATO) Special APSE verification contractor. This version has been updated as an interim version for use with DOD-STD-1838A implementations. The interim CIVC-A version is also scheduled for use on the NATO Special APSE.

Development of a validation suite for DOD-STD-1838A implementations is expected to be completed in January 1992. This version, CIVC-A, will initially exploit the interim CIVC-A version and an interface coverage test selection criteria for its production. This version will also use the hypertext traceability technology developed for the CIVC effort. The hypertext tool will provide the CIVC-A developer/user with traceability between test requirements, test scenarios, and test cases, and can be used to determine test suite coverage.

1.3.4 Ada Compiler Quality Testing Service Procedures

In June 1988, the House Appropriations Committee directed the Ada Joint Program Office to "include efficiency testing in compiler validating procedures." In response to this the AJPO has developed a draft document titled "Ada Compiler Quality Testing Service Procedures" which would guide the Ada compiler evaluation process should a testing facility be established. The current draft of this document was significantly influenced by members of the E&V Team during and between E&V Team meetings. (Much of the Team discussion on this topic is reflected in the meeting minutes contained herein.) Although no formal action has yet been taken on establishing a Quality Testing Service, the Procedures document rates as one of the major accomplishments of the E&V Team and E&V Project for the reporting period.

1.4 E&V Meetings

E&V Team meetings are held on a quarterly basis. For the period covered by this report quarterly meetings were conducted on the following dates: 5-7 December 1988, 21-24 February 1989, 6-8 June 1989, 6-8 September 1989, and 4-7 December 1989.

1.5 E&V Team Organization

In order to coordinate all of the activities within the E&V Task, the E&V Team is partitioned into six working groups. The identification of these working groups, and their associated areas of responsibility, are delineated in the following sections. These working groups are subject to change during the life of the E&V Task. Each working group has a designated Chairperson and Vice-Chairperson. It is the responsibility of each working group Chairperson to coordinate the activities of the working group with the E&V Team Chairperson. In addition, each working group Chairperson is required to brief the status of the respective working group at every E&V Team meeting.

1.5.1 Directional Management Working Groups

1.5.1.1 E&V Requirements Working Group (REQWG)

The REQWG is responsible for the following tasks:

- Maintain an E&V Requirements Document against which the E&V Reference Manual will be developed.

- Provide analysis of requirements in the E&V Requirements Document to determine their adequacy, completeness, traceability, testability, consistency, and feasibility.

- Identify issues which may impact the development of E&V technology.

- Provide recommendations for acquisition of E&V tools and aids through the development of an E&V Tools and Aids Document.

- Prepare position papers through the duration of the E&V Task which address issues on E&V requirements.

1.5.1.2 E&V Standards Evaluation and Validation Working Group (SEVWG)

The SEVWG is responsible for the following tasks:

- Recommend specific areas of consideration for standards related to future evaluations and validations.

- Emphasize study on the CAIS.

- Review the development of the CAIS and identify areas of possible concern to E&V.

- Provide presentations to the E&V Team on the CAIS.

- Prepare position papers throughout the duration of the E&V Task which address particular aspects of the CAIS as relevant to E&V.

1.5.2 Technical Management Working Groups

1.5.2.1 E&V Ada Compiler Evaluation Capability Working Group (ACECWG)

The ACECWG is responsible for the following tasks:

- Provide a formal interface between the Ada community and the ACEC effort.

- Evaluate and critique aspects of the technical approach being employed on the ACEC effort.

- Evaluate and critique selected ACEC deliverables.

- Discuss and provide feedback on issues critical to the ACEC.

1.5.2.2 E&V CAIS Implementation Validation Capability Working Group (CIVCWG)

The E&V CIVCWG is responsible for the following tasks:

- Provide technical expertise to E&V chairman and team for review of CIVC contractors' products and activities.

- Provide to E&V chairman and CIVC project engineer recommendations regarding validation of CAIS.

- Coordinate regularly and closely with SEVWG concerning validation of DOD-STD-1838 implementations.

1.5.2.3 E&V Classification Working Group (CLASSWG)

The CLASSWG is responsible for the following tasks:

- Serve as focal point for analysis of Reference System (Reference Manual and Guidebook).
- Solicit information and recommendations regarding E&V technology.
- Classify E&V technology.
- Aid in the technology transition of the Reference System.
- Delineate whole APSE issues.
- Recommend new areas of investigation.

1.6 Conclusion

This E&V Public Report is being made available by the E&V Team in order to solicit comments from those individuals who are not actively involved in the E&V Task. All comments should be addressed to:

Raymond Szymanski
WRDC/AAAF-3
Wright-Patterson AFB, Ohio 45433-6543
(SZYMANSK@AJPO.SEI.CMU.EDU or
EV-TEAM@AJPO.SEI.CMU.EDU)

APPENDIX A

Evaluation and Validation of Ada Programming
Support Environments: 5 Years After

Raymond Szymanski
Wright Research and Development Center (WRDC/AAAF-3)
Wright-Patterson Air Force Base, OH 45433-6523

ABSTRACT

This paper provides information on the Evaluation and Validation (E&V) Task sponsored by the Ada Joint Program Office (AJPO). Included is a rationale for the program, a description of objectives, and a summary of active projects. With the recent proliferation of Ada Programming Support Environment (APSE) tools, compilers in particular, it is important to have the capability to select the proper tools for the intended application. To address this complex problem, the E&V Task uses several mechanisms including the E&V Team composed of Government representatives and distinguished reviewers from industry and several E&V-technology development contracts. These elements actively interact to ensure that all E&V products address the needs of the Ada E&V community.

INTRODUCTION

The Ada Joint Program Office was formed in December 1980. It is the principal Department of Defense (DoD) agent for development, support, and distribution of tools, common libraries, and coordination of Ada activities within the DoD. The AJPO coordinates all Ada efforts within the DoD to ensure their compatibility with the requirements of the services and DoD agencies, to avoid duplicative efforts, and to maximize sharing of resources.

In June 1983, the AJPO proposed the formation of the E&V Task and a tri-service APSE E&V Team with the Air Force designated as lead service. In October 1983, the Air Force officially accepted responsibility as lead service on the E&V Task with the Air Force Wright Aeronautical Laboratory (now known as Wright Research and Development Center) as the lead organization. Since June 1985, the E&V Task has been led by Mr. Raymond Szymanski of the Avionics Laboratory.

THE NEED FOR EVALUATION AND VALIDATION TECHNOLOGY

The Ada community including Government, industry, and academic personnel, needs the capability to assess APSEs and their components and to determine their conformance to applicable standards (e.g., DOD-STD-1838, the CAIS standard). The technology required to fully satisfy this need is extensive and largely unavailable; it cannot be acquired by a single Government sponsored, professional society sponsored, or private effort.

The purpose of the E&V Task is to provide a focal point for addressing the need by (1) identifying and defining specific technology requirements, (2) developing selected elements of the required technology, (3) encouraging others to develop other elements, and (4) collecting information describing existing elements. This information will be made available to DoD components, other Government agencies, industry, and academia.

Technology for the assessment of APSEs and APSE components (tools) is needed because of the importance of the decisions to be supported by these assessments and because of the difficulty of making these assessments. The importance of the decision to select an APSE (or the approach to incremental development of an APSE) is evident when one considers the large, critical, Ada-based systems to be developed in the coming years. The effectiveness, reliability, and cost of these systems will be strongly influenced by the environments used to develop and maintain them. From the point of view of a software developing organization, the decision to select an APSE can be an important investment decision with long lasting influence on a number of projects and the organization's methods of operation, training, and competitiveness.

The difficulty of assessing APSEs and tools is evident for several reasons. First, an APSE represents complex technology with many elements which can be assessed individually or in combination. Second, there is a confusing diversity of choice with respect to individual tools, tool sets, or "whole APSEs;" and third, there are a number of ways of viewing APSEs. The state of the art of APSE architecture and of some categories of tools is undergoing rapid change (e.g., graphic design tools). Finally, there is a lack of historical data relevant to APSEs, partly because of the general pace of technological change and partly because Ada is a relatively new implementation language.

In addition to the need for assessment technology, there is a need for information about this technology. Potential buyers and users of APSEs and tools need a framework for understanding APSEs and their assessment as well as information about specific assessment techniques. Similarly, vendors of tools and APSEs need to be aware of the deficiencies of current products as well as the criteria to be used in the assessment of future products. Such awareness by both producers and consumers of APSE products, expressed in a common terminology, will accelerate the development of better software environments.

OBJECTIVES AND ACTIVITIES

In order to accomplish the purpose of the E&V Task, several specific objectives have been identified. These are discussed next with descriptions of activities that will enable the E&V Task to meet these objectives.

1. Develop Requirements for E&V

As a prerequisite to the development of APSE E&V technology, E&V requirements must be specified. The development of E&V requirements will be based upon examination of APSE related issues such as life-cycle methodologies, human engineering aspects, software engineering practices, etc. The E&V requirements which are developed will be used to guide the E&V technical effort.

The currently defined set of E&V requirements are contained in the E&V Requirements Document, Version 2.0. This document is part of the E&V Team Public Report, Volume III, which is available through the Defense Technical Information Center as AD Number A196 164.

2. Develop APSE Evaluation Capability

An evaluation capability will be developed for some APSE components for which no formal standards exist (i.e., MIL-STD, ANSI, etc.). The evaluation capability for some components will be provided through established metrics; whereas, the evaluation capability for other components may be limited to a detailed questionnaire. As a first step toward achieving this objective, an Ada Compiler Evaluation Capability is being developed which will enable the user to compare the performance of different Ada compilers. Details of this effort will be presented later.

3. Develop APSE Validation Capability

A validation capability will be developed for the Common APSE Interface Set (CAIS), DOD-STD-1838, which has been developed under AJPO sponsorship. As other APSE related standards are established (i.e., 1838A), appropriate validation capabilities will be considered for development. Examination of the current validation procedures and the Ada Compiler Validation Capability

(ACVC) test suite utilized by the Ada Validation Organization (AVO), as well as procedures implemented by ANSI and ISO, will be used as a foundation. The CAIS Operational Definition (CAIS OD) work at Arizona State University will provide a baseline from which a validation capability may be evaluated. At present, a validation suite for DOD-STD-1838 is under development. Plans are also being established for a validation suite for proposed DOD-STD-1838A. The 1838 validation suite effort will be detailed later in the paper.

4. Develop Evaluation & Validation Tools and Aids

As the requirements for E&V are determined, various software tools/aids will be identified as essential to the E&V effort. Such tools/aids include test sets, test scenarios, data reduction capability, and other designated means of automated support. As these tools/aids become more clearly defined, an assessment will be made to include such capability. Existing tools/aids which are applicable to the E&V Task will be considered for use. A document titled "E&V Tools and Aids" details the E&V Team's deliberations and recommendations on this subject and is available in the E&V Team Public Report, Volume III.

5. Provide Initiative and Focal Point with Respect to APSE E&V

A focal point is needed for APSE developers and users with regard to information about E&V of APSEs. APSE E&V questions arise frequently within professional societies and user groups. A forum is needed in which APSE E&V questions can be addressed and discussed, and in which APSE E&V information can be disseminated throughout the Ada community.

The E&V Team, through its quarterly meetings, will provide a focal point for APSE E&V for the Ada community. Public reports on the results of this activity will be made available to professional organizations such as SIGAda and AdaJUG. This is in keeping with the AJPO philosophy of public dissemination of information. The E&V task is the lead DoD effort with regard to APSE E&V. In this respect, the E&V Team will participate in, and assist where possible, other programs technically related with APSE E&V. Such programs include the Ada Validation Organization and international development efforts. To ensure that its activities are relevant to the entire Ada community, the E&V task will continue to allow distinguished reviewers from industry to attend the quarterly E&V Team meetings.

6. Promote Community Use and Acceptance of the E&V Effort

Use of the E&V technology developed through this task will provide for an orderly progression of technology insertion into user environments. The E&V technology thus developed will be extendable to other software development efforts, thereby maximizing the economic benefits of the E&V task products and minimizing the cost within DoD and industry of doing E&V related work.

In addition to the E&V Team products mentioned above, the E&V Task is responsible for three major contractual efforts. These include the Ada Compiler Evaluation Capability (ACEC), the E&V Reference System, and the CAIS Implementation Validation Capability (CIVC). As part of each development effort, the E&V contractors provide quarterly briefings to E&V Team meeting participants. This information is generally used during the E&V Team's working group sessions during which the presentation issues are discussed in detail as a form of feedback from the E&V community. The following sections provide brief technical descriptions of the ACEC, the E&V Reference System, and the CIVC.

E&V TECHNOLOGY DEVELOPMENTS

APSE E&V REFERENCE SYSTEM

The E&V Reference System is a coordinated set of documents comprised of the E&V Reference Manual, Version 1.1, and the E&V Guidebook, Version 1.1. They provide information about APSEs and their assessment.

The E&V Reference Manual establishes common terminology and a framework for understanding APSEs. It includes a Life-Cycle Activities Index, a Tool Category Index, a Function Index, and an Attribute Index. Each index entry contains a definition, cross references to entries in the same or other indices, and pointers to relevant sections in the E&V Guidebook. As a stand-alone document, it is intended to help users find information about index elements and relationships among them. In conjunction with the Guidebook, it is intended to help users find criteria, metrics, and methods for assessment of APSEs and their components.

The E&V Guidebook provides descriptions of specific instances of assessment technology. These include evaluation (assessment of performance and quality) or validation (assessment of conformance to a standard) techniques. For each category of item to be assessed (e.g., compilation system, test system, whole APSE, etc.), there are descriptions of various techniques such as test suites, questionnaires, checklists, and structured experiments. The Guidebook also contains synopses of documents of general historical importance to the field of Ada environments and their assessment.

ADA COMPILER EVALUATION CAPABILITY (ACEC)

The Ada Compiler Evaluation Capability is a product which enables users to determine the performance characteristics of Ada compilation systems. The ACEC includes the ACEC Software Product and three supporting documents: the ACEC User's Guide, the ACEC Version Description Document (VDD), and the ACEC Reader's Guide.

The ACEC Software Product consists of both operational software and support software. The operational software is a suite of performance test programs which makes it possible to (1) compare the performance of several Ada compiler implementations, (2) isolate the strong and weak points of a specific system relative to other systems which have been tested, (3) determine what significant changes were made between releases of a compilation system, and (4) predict performance of alternate coding styles.

The ACEC tests provide assistance in measuring execution time efficiency, code size efficiency, and compile time efficiency. The test suite does not explicitly cover tests for usability, capacity, or existence of language features. However, in the course of exercising the test suite, these items may be covered. The support software consists of a set of tools and procedures which assist in preparing the test suite for compilation, in extracting data from the results of executing the test suite, and in analyzing the performance measurements obtained. The support software consists of the following tools:

INCLUDE -- assists in adapting programs to particular targets by performing source test inclusion;

FORMAT -- extracts timing and code expansion data; and

MEDIAN -- compares results of performance tests of various systems.

The ACEC Software Product was developed for uniprocessor, uniprogramming target systems and is distributed on one 9-track, 1600 bpi, VAX/VMS backup tape.

The ACEC User's Guide provides ACEC users with the information necessary to adapt and execute the ACEC Software Product. This guide explains how to use the support tools and how to deal with problems which may occur in the process of executing the ACEC Software Product. The ACEC Reader's Guide describes how users can interpret the results of executing the benchmark test suite, the statistical significance of the numbers produced, the organization of the test suite, and how to submit error reports and change requests. The ACEC Version Description Document describes the ACEC Software Product as contained on the distribution tape. This product includes the compilation units, programs, test problems, specific language features and optimizations, and sample data.

CAIS IMPLEMENTATION VALIDATION CAPABILITY (CIVC)

The goal of the CAIS is to promote interoperability and transportability of Ada software across APSEs used by the DOD. Those Ada programs that are used in support of software development and lifecycle maintenance are defined as "tools." The CAIS, more formally known as DOD-STD-1838, is a document produced under AJPO sponsorship that defines the Ada package specifications for interfaces to those services, traditionally provided by operating systems, that significantly impact tool transportability. A second evolutionary step towards a full, state-of-the-art interface definition is currently under Government review. This proposed standard, DOD-STD-1838A, is an upgraded and more complex set of interfaces with compatibility to DOD-STD-1838.

The objective of the "CAIS Implementation Validation Capability" (CIVC) is to develop usable and reliable validation test suites for CAIS and CAIS-A implementations. The purpose of this validation capability is to test conformance of an implementation of the CAIS to the standard. The rationale for such a capability is to increase the reliability, usability, and acceptability of such a standardized interface set.

The CIVC contract provides for the development and delivery of a CIVC test suite and associated support products. The contractor will develop a taxonomy suitable for evolutionary development of the validation capability and deliver an integrated hypertext-based requirements traceability product to facilitate assessment of the completeness of the validation capability.

The CIVC Test Administrator will provide a convenient and reliable user-interface to the validation capability. This feature will facilitate application of the CIVC by both CAIS developers and CAIS users. The hypertext product provides an interactive vehicle for analyzing the connection between (1) requirements (paragraphs in DOD-STD-1838), (2) test objectives (developed from the requirements), (3) test scenarios (test design definitions), and (4) actual test cases (Ada code).

The CIVC contract has recently completed critical design review (CDR) and has moved into the development phase where test cases, the Test Administrator code, and associated traceability documentation will be produced. The initial operational capability is scheduled for delivery in the fourth quarter of this year (4th Q, 1989).

E&V TASK OUTLOOK

To date, considerable progress has been made in the areas of E&V problem definition and creating solutions to those problems in the form of E&V technology developments. This author believes that many challenges in this area still exist and will exist for some time to come. The E&V task is scheduled to continue through 30 September 1991 when the last contractual effort terminates. Hopefully, a far-sighted organization within the Air Force will understand the importance of the E&V effort and will continue where the current effort leaves off.

OBTAINING E&V INFORMATION

The ACEC documentation described above is available in hardcopy form or as a package distributed in Latex format on one 9-track, 1600 bpi, VAX/VMS backup tape.

Please note that all ACEC products are subject to the DoD Directive 5230.25, Withholding of Unclassified Technical Data from Public Disclosure, which limits the distribution of unclassified export-controlled technical data to organizations certified as qualified contractors by the Defense Logistics Services Center (DLSC). It is not necessary for Government activities to be DLSC certified.

To order the ACEC software and documentation, please contact the Data & Analysis Center for Software at the following address:

Data & Analysis Center for Software
RADC/COED
Griffiss AFB NY 13441-5700

ATTN: Document/Dataset Ordering
(315) 336-0937

To obtain information concerning the availability of E&V products such as the E&V Reference System, the CIVC, or E&V Team documents, send your name and address (electronically preferred) to: szymansk@ajpo.sei.cmu.edu, or by regular mail to: Mr Raymond Szymanski, WRDC/AAAF, Wright-Patterson AFB OH 45433-6523.

About the author, Raymond Szymanski: Mr. Szymanski is currently the Program Manager for the Evaluation and Validation (E&V) of Ada Programming Support Environments (APSEs) Task sponsored by the Ada Joint Program Office. In addition to administering technical contracts for the E&V effort, Mr. Szymanski is also the Chairman of the E&V Team. He has given presentations at various technical forums including SIGAda, AdaJUG, Ada Europe (Edinborough), and the Ada Board (former member).

6/90

APPENDIX B

-- New Documents Available --

APSE E&V REFERENCE SYSTEM

The Ada Programming Support Environment (APSE) Evaluation and Validation (E&V) Reference System is a pair of documents developed, and periodically updated, by the APSE E&V Task, sponsored by the Ada Joint Program Office and led by the US Air Force Avionics Laboratory. The documents are entitled the "E&V Reference Manual" and the "E&V Guidebook."

APSE E&V Task Purpose -- The Ada community needs the capability to assess APSEs and their components, and to determine their conformance to applicable standards. The technology required to fully satisfy this need is extensive and largely unavailable. The purpose of the APSE E&V Task is to provide a focal point for addressing this need by (1) identifying and defining specific technology requirements, (2) developing selected elements of the required technology, (3) encouraging others to develop some of these elements, (4) collecting information describing existing elements, and (5) making E&V technology information available to government agencies, industry, and academia.

E&V Reference Manual -- The manual provides a framework for understanding APSEs and their assessment, and establishes common terminology. One chapter discusses an APSE as a whole and its assessment. Other chapters are indexes to APSE component characterization and assessment, organized by life cycle activities, APSE tool category, APSE function, and attribute to be assessed. An entry in an index consists of a description, cross references to other entries in the Reference Manual, and cross references to the "E&V Guidebook." The manual is intended to help a variety of users obtain answers to their questions. As a stand-alone document it is intended to help a user find useful information about index elements and relationships among them. In conjunction with the Guidebook, it is indended to help users find criteria and metrics for assessment of APSEs and their components.

E&V Guidebook -- The Guidebook provides descriptions of specific instances of assessment technology. These include evaluation (assessment of performance and quality) and validation (assessment of conformance to a standard) techniques. For each category of item to be assessed (e.g. compilation system, test system, whole APSE, etc.), there are brief descriptions of applicable tools and aids -- such as test suites, questionnaires, checklists, and structured experiments -- and references to primary documents containing detailed descriptions. The Guidebook also contains synopses of documents of general historical importance to the entire field of Ada environments and their assessment.

E&V Task Products and Schedule

E&V Reference Manual -- Version 2.0, DTIC (or NTIS) No. AD-A214 167; 3.0 (Nov 90)
E&V Guidebook -- Version 2.0, DTIC (or NTIS) No. AD-A214 166; 3.0 (Nov 90)
Ada Compiler Evaluation Capability (ACEC) test suite -- Version 2 (call DACS)
CAIS-A Implementation Validation Capability(CIVC) tests -- TBD

MAILING LIST FOR E&V PRODUCTS

If you would like to receive instructions for obtaining the E&V Reference System documents and other E&V products as they become available, attach your business card or fill in your name and address and send to Mr. Raymond Szymanski, WRDC/AAAF, Wright Patterson AFB, OH 45433-6523. .

Name ______________________________

Address ______________________________

____

APPENDIX C

E&Ving NEWS

Volume 1.0 October 1989

The E&V Task

The Evaluation & Validation (E&V) Task, under the direction of Ray Szymanski, WRDC, provides a focal point for addressing APSE assessment needs by (1) identifying and defining specific technology requirements, (2) developing selected elements of the required technology, (3) encouraging others to develop other elements, and (4) collecting information describing existing elements. This information is being made available to DoD components, other government agencies, industry, and academia. To meet these needs the E&V Task has identified six guiding objectives. These include (1) Develop requirements for E&V technology, (2) Develop APSE evaluation capabilities, (3) Develop APSE validation capabililities, (4) Develop additional E&V tools and aids, (5) Provide a focal point for APSE E&V, and (6) Promote community use and acceptance of E&V technology.

E&V REFERENCE SYSTEM

The E&V Reference System consists of two companion documents: the E&V Reference Manual and the E&V Guidebook. The purpose of the E&V Reference Manual is to provide information that will help users to: (1) Gain an overall understanding of APSEs and approaches to their assessment, (2) Find useful reference information (e.g., definitions) about specific elements and relationships between elements, and (3) Find criteria and metrics for assessing tools and APSEs, and techniques for performing such assessments.

The purpose of the E&V Guidebook is to provide information that will help users to assess APSEs and APSE components by: (1) Assisting in the selection of E&V procedures, the interpretation of results, and integration of analyses and results, (2) Describing E&V procedures and techniques developed by the E&V Task, and (3) Assisting in the location of E&V procedures and techniques developed outside the E&V Task.

All E&V procedures and techniques found in the E&V Guidebook are referenced by the indices contained in the E&V Reference Manual.

Initial versions of the E&V Reference Manual and E&V Guidebook were distributed in the Fall of 1988. In response to comments received via the E&V Reference System Questionnaire, Version 2 of the Reference System has been enhanced by the incorporation of a number of new checklists and additional references to emerging E&V Technology. Version 2 is scheduled for release this fall. Yearly updates of the E&V Reference System are planned. Constructive comments and pointers to E&V technology, not currently in the E&V Reference System, are always welcome.

CIVC

Version 1.0 of the CAIS Implementation Validation Capability (CIVC) is nearing completion. Formal testing for the CIVC begins in early December, 1989 with general availability in early 1990. The CIVC will provide a reliable and useful capability for validation of DoD-STD-1838. Included in the initial release will be over 200 individual test cases, an integrated and versatile test manager, and a hypertext product (the Framework) to provide traceability between the standard (DoD-STD-1838), the taxonomy, and the actual test cases. Future work under this contract activity will focus on development of a validation capability for MIL-STD-1838A.

ACEC

Version 1.0 of the Ada Compiler Evaluation Capability (ACEC) has been distributed by DACS to several dozen sites since its release in October 1988. Version 2.0 is scheduled to be delivered to the government in December 1989. 294 new performance tests are being added along with an assessment capability for three new functional areas; diagnostics, the debugger, and the program library system. It will also provide a tool to assist in analyzing the performance of a single compilation system. Problem reports received for Version 1.0 are being reviewed for inclusion in Version 2.0. ACEC users are encouraged to provide feedback on their experiences and submit error reports as appropriate to Mr. Raymond Szymanski at the address below.

FOR INFORMATION ON E&V PRODUCTS CONTACT:

RAYMOND SZYMANSKI
WRDC/AAAF-3
WPAFB, OH 45433-6543
PHONE: (513) 255-3947
NET: szymansk@ajpo.sei.cmu.edu

E&Ving NEWS

Volume 1.1 December 1989

The E&V Task

The Evaluation & Validation (E&V) Task, under the direction of Ray Szymanski, WRDC, provides a focal point for addressing APSE assessment needs by (1) identifying and defining specific technology requirements, (2) developing selected elements of the required technology, (3) encouraging others to develop other elements, and (4) collecting information describing existing elements. This information is being made available to DoD components, other government agencies, industry, and academia. To meet these needs the E&V Task has identified six guiding objectives. These include (1) Develop requirements for E&V technology, (2) Develop APSE evaluation capabilities, (3) Develop APSE validation capabililities, (4) Develop additional E&V tools and aids, (5) Provide a focal point for APSE E&V, and (6) Promote community use and acceptance of E&V technology.

E&V REFERENCE SYSTEM

The E&V Reference System consists of two companion documents: the E&V Reference Manual and the E&V Guidebook. The purpose of the E&V Reference Manual is to provide information that will help users to: (1) Gain an overall understanding of APSEs and approaches to their assessment, (2) Find useful reference information (e.g., definitions) about specific elements and relationships between elements, and (3) Find criteria and metrics for assessing tools and APSEs, and techniques for performing such assessments.

The purpose of the E&V Guidebook is to provide information that will help users to assess APSEs and APSE components by: (1) Assisting in the selection of E&V procedures, the interpretation of results, and integration of analyses and results, (2) Describing E&V procedures and techniques developed by the E&V Task, and (3) Assisting in the location of E&V procedures and techniques developed outside the E&V Task.

All E&V procedures and techniques found in the E&V Guidebook are referenced by the indices contained in the E&V Reference Manual.

Initial versions of the E&V Reference Manual and E&V Guidebook were distributed in the Fall of 1988. In response to comments received via the E&V Reference System Questionnaire, Version 2 of the Reference System has been enhanced by the incorporation of a number of new checklists and additional references to emerging E&V Technology. Version 2 was released in November 1989. Yearly updates of the E&V Reference System are planned. Constructive comments and pointers to E&V technology, not currently in the E&V Reference System, are always welcome.

CIVC

Version 1.0 of the CAIS Implementation Validation Capability (CIVC) is nearing completion. Formal testing for the CIVC begins in early December, 1989 with general availability in early 1990. The CIVC will provide a reliable and useful capability for validation of DoD-STD-1838. Included in the initial release will be over 200 individual test cases, an integrated and versatile test manager, and a hypertext product (the Framework) to provide traceability between the standard (DoD-STD-1838), the taxonomy, and the actual test cases. Future work under this contract activity will focus on development of a validation capability for MIL-STD-1838A.

ACEC

Version 1.0 of the Ada Compiler Evaluation Capability (ACEC) has been distributed by DACS to several dozen sites since its release in October 1988. Version 2.0 is scheduled to be delivered to the government in early 1990. Over 300 new performance tests are being added along with an assessment capability for three new functional areas; diagnostics, the debugger, and the program library system. It will also provide a tool to assist in analyzing the performance of a single compilation system. Problem reports received for Version 1.0 are being reviewed for inclusion in Version 2.0. ACEC users are encouraged to provide feedback on their experiences and submit error reports as appropriate to Mr. Raymond Szymanski at the address below.

FOR INFORMATION ON E&V PRODUCTS CONTACT:

RAYMOND SZYMANSKI
WRDC/AAAF-3
WPAFB, OH 45433-6543
PHONE: (513) 255-3947
NET: szymansk@ajpo.sei.cmu.edu

APPENDIX D

Issues and Strategies for Evaluation
and Validation of CAIS-A
Implementations

Based on the April 6, 1989

MIL-STD-1838-A

February 1, 1990

Prepared by:
The Standards Evaluation and Validation Working Group
of the APSE Evaluation and Validation Team

The Task for Evaluation and Validation of Ada Programming Support Environments (APSE's) is sponsored by the Ada Joint Program Office.

TABLE OF CONTENTS

Section . . . Page

1.0 INTRODUCTION . . . 3
1.1 Objective . . . 3
1.2 Background - E&V Team . . . 3
1.3 Standards Evaluation and Validation Working Group . . . 3
1.4 CAIS-A: A Transportability Platform for APSE Tools . . . 4

2.0 SCOPE . . . 5
2.1 Document Background . . . 5
2.2 Basis for Identifying Issues . . . 5

3.0 APPROACH . . . 6

4.0 CIVC-A COST ANALYSIS . . . 7
4.1 Additional Interfaces and Functionality . . . 7
4.2 Additional Functionality in Existing Interfaces . . . 7
4.3 The Number and Size of Anticipated Tests . . . 7

5.0 TEST SELECTION CRITERIA FOR CIVC-A . . . 9
5.1 Requirements of Test Selection Criteria . . . 9
5.2 Facilities that Implementors are Likely to Omit . . . 10
5.3 Facilities Most Critical to Transportability . . . 10
5.4 Facilities Most Likely to Apply to Both CAIS and CAIS-A . . . 11
5.5 A Random Sampling Across Facilities . . . 11
5.6 Facilities That Achieve Broadest Implementation Coverage . . . 12

6.0 REVIEW BOARD AND FAST REACTION TEAM . . . 13

7.0 CAIS-A EVALUATION CAPABILITY . . . 14
7.1 Performance Tests . . . 14
7.2 Alternative Approaches . . . 15

8.0 CAIS-A EVALUATION VALIDATION POLICY . . . 17

9.0 AUTOMATICALLY GENERATE TEST CASE CODE . . . 18
9.1 Develop a Scenario Description Language . . . 18
9.2 Scenario Dependency Analysis . . . 19
9.3 Translator from Scenario Descriptions to Ada . . . 19

10.0 DEVELOP A NEW TAXONOMY OR CIVC-A . . . 20

11.0 MAINTENANCE OF THE CIVC-A TEST SUITE . . . 22

12.0 SUMMARY RECOMMENDATIONS . . . 23

APPENDIX A - ACRONYMS . . . 25
APPENDIX B - SEVWG MEMBERSHIP . . . 26
APPENDIX C - REFERENCES . . . 27

1.0 INTRODUCTION

1.1 Objective

This document is intended to provide insights and guidelines for the analysis and validation of implementations of the Common Ada Programming Support Environment Interface Set, MIL-STD-1838A (CAIS-A) [CAIS-88]. In this document, the key issues to validating CAIS-A implementations are identified and discussed. Where possible, we identify approaches to resolve these issues. An earlier version of this document [IAS-88] identified issues relative to the January 1985 version of CAIS. Many of the issues discussed relative to January 1985 CAIS are relevant to CAIS-A, these are not restated herein; instead, we discuss only those issues relevant to the design changes resulting in CAIS-A.

The analysis reported in this document was performed by the Standards Evaluation and Validation Working Group (SEVWG) of the APSE Evaluation and Validation (E&V) Team, whose membership appears in Appendix B.

1.2 Background - E&V Team

In 1983 the AJPO formed the Task for Evaluation and Validation of Ada Program Support Environments (E&V Task) and a tri-service APSE E&V Team, with the Air Force designated as lead service. The overall goal of the E&V Task is to develop the techniques and tools which provide a capability to perform assessment of APSEs and to determine conformance of APSEs to relevant standards. As the E&V technology is developed, it is made available to the community for use by DOD organizations, industry, and academia as deemed appropriate by the respective organizations. The E&V Task is developing technology to evaluate specific APSE components, including CAIS-A. The soecific components and evaluators are enumerated in the E&V Team Requirements document [REQ-87] and they include components such as compilers, editors, command language interpreters, and debuggers.

The Air Force has been tasked as the lead service on this effort. Hence, the majority of E&V Team members are Air Force personnel. Air Force Wright Research and Development Center (WRDC) is lead organization for the E&V Task and the E&V Team Chairperson is a WRDC representative.

1.3 Standards Evaluation and Validation Working Group (SEVWG)

The E&V Team is divided into working groups. The Standards Evaluation and Validation Working Group(SEVWG) is chartered to provide a forum for the evaluation and validation of current, proposed and future Ada Programming Support Environment (APSE) related standards and their implementations. Included in this charter is the identification of issues relating to validating conformance to an APSE related standard and suggesting approaches for achieving conformance. Further, the SEVWG is concerned with evaluating all aspects of standards implementations. SEVWG considers both technical and non-technical aspects of APSE related standards.

1.4 CAIS-A: A Transportability Platform for APSE Tools

CAIS-A is a set of Ada package interfaces designed to enhance the transportability and interoperability of Ada software engineering environment tools and data. The scope of the CAIS-A includes the functionality affecting transportability that is needed by tools, but not provided by the Ada language. In addition to a general entity management system for APSE tools, the CAIS-A contains definitions for primitive entities for manipulating devices, files and processes. CAIS-A is based on an entity-relationship approach and it allows the user to define entities, in a limited way, by means of a typing mechanism. CAIS-A also includes functionality to support tools requiring transaction processing, a rudimentary triggering mechanism and explicit control over APSE distribution.

The CAIS-A was developed by SofTech under contract to the Naval Ocean Systems Center. CAIS-A is a design enhancement of the existing DOD Standard CAIS (1838) [CAIS-86.] CAIS 1838 was developed by the Kernel APSE (KAPSE) Interface Team and the KAPSE Interface Team for Industry and Academia(KIT/KITIA) as a first evolutionary step towards a full, state-of-the-art interface standard. CAIS-A is viewed as the next step in that evolutionary process.

The KAPSE Interface Team (KIT), a tri-service organization chaired by the Navy under the guidance of the AJPO, was established in late 1981 as the result of a Memorandum of Agreement signed by the Deputy Under Secretary of Defense and the Assistant Secretaries of the three services. The KIT completed its work in 1988. The KAPSE Interface Team from Industry and Academia (KITIA) was established in early 1982. The KITIA consisted of volunteer representatives from industry and academia who provided technical expertise and review capability to the KIT. The objective of the KIT/KITIA was to define a standard set of Kernel Ada Programming Support Environment (KAPSE) interfaces to ensure the interoperability of data and the transportability of tools between conforming APSE's. The CAIS DOD-STD-1838, developed by the KIT/KITIA, provides a common kernel interface for tools requiring device, file, and process manipulation.

In addition to the KIT/KITIA's development of the CAIS 1838, other efforts have contributed to the foundation of the E&V Task. One such effort was the formation of the Ada Validation Organization (AVO), under the direction of the AJPO. The AVO is responsible for the development of an Ada Compiler Validation Capability (ACVC) which is in use to determine that Ada compiler developers have consistently implemented the standard Ada language, ANSI/MIL-STD-1815A [ADA-83]. A second effort which contributes to the E&V task is the derivation of a taxonomy for an APSE, which systematically defines tool capabilities for a full APSE. A third effort, performed at the Air Force WRDC, provided an initial evaluation mechanism for Ada compilation systems; called, the Ada Compiler Evaluation Capability (ACEC). Finally, previous efforts sponsored by the AJPO, at Virginia Tech and Arizona State University have addressed various techniques for the validation of Ada software interfaces.

2.0 SCOPE

2.1 Document Background

The SEVWG is composed of a representative spectrum of potential CAIS-A users and implementors from academia, government, and industry. The diversity of users possess different perspectives on the CAIS-A which include:

- o Funding agencies and end user's of tools who are principally concerned with maximizing tool transportability and who are motivated by the need to obtain a reliable mechanism for encouraging and establishing the use of CAIS-A-based technology;

- o APSE and tool developers concerned with the flexibility, efficiency, and completeness of the CAIS-A standard and the ease or difficulty of using it as a means of achieving enhanced tool functionality; and

- o CAIS DOD-STD-1838 developers that are concerned with developing validation tests consistent with the intent of the current standard CAIS-A, current operational definition efforts, and anticipated future enhancements.

The earlier version of this document, which was released in 1988, covered the CAIS as it existed just prior to its standardization as DOD-STD-1838. That document raises issues and outlines approaches to validation and evaluation of CAIS, most of which still apply to the CAIS-A. This document has been produced as an increment to the earlier release [IAS-87], and it does not iterate those issues covered earlier.

2.2 Basis for Identifying Issues

This document addresses the analysis, evaluation, and validation of the CAIS-A. Consequently, sections in this document require access to and an understanding of the CAIS-A. This document enumerates many of the issues and problems that should be considered for validation and evaluation of the CAIS-A implementations, and potential solutions are presented as appropriate. This document does not provide a complete or comprehensive set of issues or solutions to these issues. The scope of this document includes issues arising from design decisions resulting in the CAIS-A. Included in these design changes are transaction mechanisms, typing, extensions to the node model, explicit control of distribution and a triggering mechanism.

3.0 APPROACH

The initial three chapters of this document present introductory material including some of the motivation for the E&V Team, the SEVWG, and the creation of this document. The last chapter contains summarized recommendations regarding resolving the issues discussed in the document. CAIS-A validation and evaluation issues are presented in separate chapters beginning with Chapter Four. Validation issues may be either technical or programmatic in nature. The topics included in the evaluation discussion include those necessary for determining the performance features of a given CAIS-A implementation as well as other aspects relevant to selecting a CAIS-A platform. The appendices of this document detail items such as acronyms, SEVWG membership and references.

This document is of interest to the designers or modifiers of the CAIS standard. It also provides limited insights to certain problem areas for those interested in implementing the CAIS-A. The CAIS-A validation contractor will also benefit from these preliminary investigations, as will those who are developing a prototype evaluation capability for entire APSE's. The first and foremost application of this document is the communication of this information within the E&V Team itself, and between the E&V Team and directly related activities and organizations. These include:

1. E&V Technical Support Contractor

2. CAIS-A Implementation Validation Capability(CIVC) Contractor

3. Government funded CAIS-A developers.

This document is also intended as a vehicle to communicate these issues to other interested organizations, which consist primarily of government agencies and contractors considering the utilization or development of CAIS-A implementations or CAIS-A-resident tool-sets.

4.0 CIVC-A COST ANALYSIS

An immediate concern to SEVWG upon receiving and reviewing the Proposed CAIS-A set of interfaces is the additional complexity and resulting additional effort generating, administering and executing a validation mechanism for this new set of interfaces. In total, the document has increased in size from roughly 600 pages for CAIS to roughly 1200 pages for CAIS-A. For various reasons, one of course cannot outright compare complexity of CAIS-A to CAIS based on document length. Further, we expect that any expansion in the document is attributable not only to added facilities and complexity, but also to clarifying and detailing existing functionality. Nevertheless, the expanded size and scope of CAIS-A over CAIS will certainly have an impact on the ability to develop, manage, execute and maintain a validation (and evaluation) capability for CAIS-A. In this section, we try to classify the differences in the evolution from CAIS to CAIS-A. In doing so, we indicate the impact on validation.

4.1 Additional Interfaces and Functionality

In an early draft of CAIS-A, the number of interfaces, excluding overloads, had increased well over 100. In the most general sense, the functionality added includes a permissive typing mechanism (including type definitions, type checking and alternative views of type identifiers,) transactions, attribute monitors, distribution, access control, a more general Input Output model and several changes in the basic node model. Of these, typing transactions and changes to the basic node model have the largest impact on validation.

4.2 Additional Functionality in Existing Interfaces

Several interfaces in CAIS-A were already in CAIS. These interfaces, however, often have additional functionality imposed by added features. Additional functionality isn't necessarily accompanied by additional parameters or exceptions. For example, addition of the typing model has made a modest increase in the number of interfaces, primarily to define and manipulate definition and view nodes. But, a majority of the functional changes are distributed across existing interfaces. The added functionality for typing can generally be referred to as the semantics of type checking. Other functional changes that alter existing interfaces includes adding multiple keys for relationships, manipulating unique identifiers for nodes, bidirectional relationships and case sensitivity. Now, CAIS-A comparisons are case insensitive, but case is preserved.

4.3 The Number and Size of Anticipated Tests

SEVWG has discussed an estimation of the number of validation tests that might be needed for CAIS-A based on the existing draft specification and on the estimated number of test objectives required for 1838 (roughly 8000.) We agreed that it would be counter productive to include this form estimation since we have based it on unreliable assumptions. SEVWG does, however, have two recommendations regarding the cost to develop and execute a validation mechanism for CAIS-A.

Several factors can be identified through the development of CIVC that could be very useful in estimating the cost to produce a validation mechanism for CAIS-A. These include; the number of test objectives identified per interface, the number of test objectives identified per specification paragraph, the number of scenarios per test objective, the number of scenarios per test case and the number of Ada/CAIS source statements per test case. SEVWG recommends that SofTech include in the development of CIVC, as one of its primary responsibilities, a gathering of this information as it is pertinent to formulating and estimating the needed resources for CIVC-A, as well as its future evaluation. A cost estimate for CIVC-A will be most accurate when it is obtained with information recorded by SofTech regarding the above factors. It would also be of benefit to record the resources required (personnel, machine, etc.) and directly attributable to the development of test objectives (identification, development, documentation, review, and configuration management), scenarios, framework, and the test cases themselves.

SEVWG also emphasizes the development of reusable tests in CIVC phase 1 development. Although it is currently apparent that this awareness exists, it would be helpful to document a transition plan for CIVC indicating how products associated with CIVC can be adapted to CIVC-A. That plan would further be useful in future evolution and maintenance of CIVC-A.

5.0 TEST SELECTION CRITERIA FOR CIVC-A

SEVWG views that one of the more important issues regarding the development of CIVC-A is the method used to select test objectives. CIVC limited its scope to Chapter 4 (General Requirements) tests. The decision to start with Chapter 4 was influenced by SEVWG as well as the desire to augment related efforts to generate test sets for CAIS, for example MITRE and Arizona State University. Chapter 4 contains functionality applicable throughout the CAIS, and no other efforts are working in this area. Included in Chapter 4 are relationship management, pathname syntax, access control mechanisms and basic node concepts. These facilities are required by process control, node management, input/output, etc. (virtually the remainder of CAIS). SEVWG agrees, however, that creating CIVC-A using such a narrow focus would only serve to encourage partial implementations of CAIS-A. Thus, this section looks at alternative approaches to selecting tests and considers how they should be implemented.

5.1 Requirements of Test Selection Criteria

Various requirements apply specifically to the context of generating a validation set for kernel interfaces such as CAIS-A.

1. Development of the test suite must employ several test selection criteria in order to balance test suite costs (development, administration and maintenance) with interface implementation objectives (transportability, implementation completeness and architecture suitability.)

2. Criterion must be capable of accommodating a prioritized development scheme in which criteria may be prior tized for greater emphasis.

3. The selection criteria must allow selection of tests from the same representation of CAIS-A (e.g., the SofTech Taxonomy.)

4. The test selection process must be amenable to assessing test coverage. Preferably, coverage should be assessable with respect to a named criterion as well as providing for a unified analysis of coverage which considers the entire suite.

5. Tests selected from different criterion must be capable of being managed together with those selected from other criteria.

6. Test selection criteria must be implementable. That is, there must exist in current technology the ability to select tests based on a criterion. Although implementability is in reality a scale, certain criterion are considerably less implementable than others. For instance, the criterion: "select tests best able to distinguish complete CAIS-A implementations" is currently not implementable. Without having several CAIS-A implementations and a history of successful and unsuccessful validations, one cannot determine which tests distinguish successful from unsuccessful implementations.

Several test selection criteria have been identified by SEVWG, most of which we believe have the potential of conforming to these requirements. These include (listed in no specific order) selecting tests for:

1. Facilities that encourage complete implementations,

2. Facilities most critical to transportability of CAIS-A tools,

3. Facilities in CAIS-A common to other tool interfaces,

4. A random sampling across facilities,

5. Facilities that achieve broadest coverage.

SEVWG has discussed each of these selection criteria in detail and has no single recommended criterion that we feel should be used. The following subsections discuss in order, methods we've discussed regarding techniques for implementing each criterion in selecting tests to be implemented in the suite, our recommendation as to which criteria should be used, and finally, a presentation of the remaining issues regarding how to balance the selection criteria with available resources for CIVC-A development.

5.2 Facilities That Encourage Complete Implementations

Some CAIS-A interfaces have small visible syntax, but require extensive implementation detail or have complex semantics. This might include, for instance, path name pattern matching in iterators, or other complex interfaces such as COPY_TREE. One aspect of this criterion is that there are certain facilities in CAIS-A which have only a small syntactic interface to the CAIS-A user, but which require extensive functionality to implement. Validation tests that exercise this functionality are more likely to distinguish among conforming implementations and encourage complete implementations. SEVWG supports focusing some validation tests in areas where CAIS-A semantics are complex. Well-chosen tests of facilities that are commonly used by tools, but for which there exist complex semantics can only aid in achieving the goals of CAIS-A.

Drawing on criterion used by testing services, one approach would be to incorporate tests most likely to predict success in the overall validation test suite. That is, supposing that there are 15,000 possible test objectives in CAIS-A and that only 5,000 are to be developed into test programs. Then, one would want to select the 5,000 revealing the most information. Those tests most often failed by non-conforming implementations would be most revealing regarding correct implementation of the standard. The primary problem with this approach as applied to CAIS-A is that it is difficult to implement. Empirical data is needed revealing the success rate of implementations on each test program. Test programs that are the most valuable are those that are failed by an implementation that fails very few others. That is, the most revealing test programs are those that in isolation predict the validation results. Thus, implementation of this technique tends to discard test programs from the validation suite that are not good predictors and incorporate new test cases that reveal more about the implementation. This concept is appealing, and is recommended by SEVWG for use in maintaining the suite rather than in original development.

5.3 Facilities Most Critical to Transportability of CAIS-A Tools

This criterion selects tests for interfaces that are deemed most critical to transportability. While the objective of CAIS-A is to enhance the transportability of tools, certain aspects of CAIS-A have a larger impact on transportability of most tools than other aspects of CAIS-A. For example, using this criterion, one might focus on Input and Output interfaces and on pathname manipulation within CAIS, knowing that these interfaces are most frequently used by tools important to transport.

One possible implementation technique for this criterion would be to study the literature and existing state of tools to determine the most frequently used facilities. One could examine tools that should be transportable, such as editors, source analyzers, target simulators, mail tools, and possibly cross compilers. The study would list, according to frequency, all system interfaces used by these tools. Next, each interface used by existing tools would be mapped to a corresponding functionality and interface(s) in CAIS-A. This may not be a trivial process. Although we expect that there are corresponding interfaces in CAIS-A for all or nearly all the interfaces used by existing tools, there may be a hierarchy of CAIS-A facilities needed to support that functionality. For example, CAIS-A typing, access control mechanisms, and CAIS constants would all be needed for nearly any functionality. Nonetheless, this process would result in a ranked listing of the facilities in CAIS-A most needed for transportability.

SEVWG, in considering this criterion, has discussed the likely interfaces that would appear in such a ranking. We suspect that many of the functions that would appear high in the priority listing would be input and output facilities. In particular, CAIS-A does not explicitly support windowing or graphical interfaces. The reason for this is that this area is changing dramatically. In the design process of CAIS-A, several potential interfaces were discussed such as X-Windows, QuickDraw, PHIGS, GKS, etc. It was felt that developing a binding for one (or more) of these systems for inclusion in CAIS-A would at this time negatively affect the potential success of the interfaces as a whole. Because of this decision, we expect that a whole class of input/output functions, deemed critical to transportability, do not even exist in CAIS-A. SEVWG feels that CIVC-A would be most effectively developed if it were to place all input/output interfaces at a low priority given the quickly changing technology in this area.

5.4 Facilities in CAIS-A Common to Other Tool Interfaces

AJPO is currently pursuing CAIS-A implementations, supporting tool development and related efforts, such as developing a validation mechanism. CAIS-A facilities refine and enhance CAIS adding important functionality. Yet, the composition of CAIS-A includes substantial functionality that is basic not only to CAIS, but also to tool support interfaces in general, such as process management, aspects of access control and path name manipulation. Validation tests developed for this basic set of functionality having more stable and better refined semantics and implementation paradigms. Refined functionality has higher potential for evolution and continued impact on tool transportability. SEVWG recommends that one criterion for selecting tests for CAIS-A be the extent to which the functionality being tested is common to general tool support interfaces.

5.5 A Random Sampling Across Facilities

Studies in testing have shown that a random approach to test selection is beneficial in certain situations [DURAN-84]. Where there is a limited ability to adequately cover the entire range of functionality, a random approach to selecting tests performs well.

There are areas of CAIS-A for which sufficient resources will not exist to allow full development of tests. A randomly selected set of tests should be developed to provide assessment in such areas. The only requirement necessary to apply this technique is the ability to enumerate potential tests. The SofTech taxonomy for CIVC (as modified for CAIS-A) should provide a basis for this enumeration since it lists all actions on entities that should be tested in validation.

An enhancement to this approach would be to predetermine the portion of tests to be developed in classes of tests in a CAIS-A area. For example, one may decide that the tests in access control be evenly distributed among the test types, exception processing, normal processing and static processing. An enumeration of the potential access control tests in each of these classes would allow a more finely tuned random selection process. SEVWG recommends that the CIVC-A contract investigate the cost of selecting a portion of the CIVC-A tests at random.

5.6 Facilities That Achieve Broadest Coverage

Another criterion might be to select tests whose execution involves interfaces minimally used by the validation suite. Execution of a single validation test program requires the use of several CAIS interfaces. Most test programs establish an initial context (input condition) by invoking other CAIS interfaces. They then perform some action against that context and complete by calling other CAIS interfaces to determine whether the proper effect is achieved (output condition.) In light of this, the best measure for broadest coverage of CAIS interfaces used is not determined by the test objectives for which test programs are developed. Instead, one must consider all the CAIS interfaces that are used by a test program for a scenario.

One way to do this is to develop several test objectives and scenarios. For each scenario, determine the interfaces needed to execute the scenario, and those other scenarios having common or overlapping input conditions. Assuming that a predetermined number of test programs could be developed, one could determine which test programs to develop in the following manner. Iteratively pick the scenario requiring the maximum number of unused interfaces. Develop a test program for that scenario and all others having an overlapping input condition. From this selection technique, a database could be developed revealing information such as relative importance of each test program, with respect to diversity of interfaces used, and traceability to other scenarios and test programs.

The important situation that makes this issue relevant to CIVC-A is that there are insufficient resources to construct a complete set of tests for CAIS-A. With the limited resources, this approach shows how to select the tests that produce a suite that best covers aspects of an implementation.

6.0 REVIEW BOARD AND FAST REACTION TEAM

Several issues have been raised by the SEVWG aimed at producing the highest quality test suite possible given the seemingly impossible constraint that there are currently no reasonable CAIS (-A) implementations on which to test the test suite. The current state of the implementations most likely to be used for testing the CIVC indicates that a significant amount of effort would be required in order to allow them to execute their test programs on a diversity of CAIS implementations. Seemingly, the problem will worsen with CAIS-A, as it has recently completed the standardization process and the complexity of the interfaces will surely require more time to develop solid implementations than for CAIS.

The SEVWG supports the creation of a specific review activity to evaluate the completeness, accuracy, applicability and evolution of the CAIS and CAIS-A test suites. The activity would include a review board, most likely consisting of members of the SofTech CIVC development team, E&V Team and SEVWG team members, CAIS-A design team members and possibly some members of the Government Review Team for the CAIS-A design. Among other responsibilities the activity and board would serve as the focal point for:

1. Reviewing and monitoring the test suite development process including:
   a. Determining whether the test objectives developed by SofTech were correct with respect to the standard and whether they fairly represented the test selection mechanism(s),
   b. Reviewing the scenarios to determine their accuracy with respect to the test objectives,
   c. Review the mappings from scenarios to test programs for appropriateness, and
   d. Determine whether the test programs themselves accurately reflect the scenarios.
   e. Review for appropriateness and accuracy all documentation developed for the CIVC-A.

2. Serve as a fast reaction team regarding the use and interpretation of tests in the suite.

3. Monitor and promote evolution of the suite. In particular, recommend and encourage areas where transportability would best be served by additional suite expansion, or where uses of CIVC-A and CAIS-A could best serve the interests of transportability.

4. Serve as the focus for the process by which the test suite is applied to CAIS implementations.

This activity would serve as the technical sounding board for the development of CIVC-A, in much the same manner as CIVCWG does for the development of CIVC. Further, it would provide the framework for both CAIS and CAIS-A that allows a new test to be introduced in a controlled way. This ongoing effort will also require an organization to review new test recommendations and to make decisions regarding any additions.

7.0 CAIS-A EVALUATION CAPABILITY

The Ada community has recognized the need to gather data on compilation systems. The information is useful when a project must select compilers and associated tools. In part, the need has been fed by existing compilers available on a multitude of host environments. A project intending to use a specific host frequently has several choices of Ada compilers for the host configuration. In addition to the plethora of compilers, we are seeing that compilation systems vary widely in their support of various Ada features; most often tasking, exceptions, and machine dependencies. As a step toward meeting this need, the E&V Task has sponsored the development of the ACEC (Ada Compiler Evaluation Capability) which is a suite of performance tests for Ada compilation systems.

SEVWG anticipates the same need for implementations of CAIS-A. For example, we expect to see wide differences in the performance characteristics of CAIS-A implementations. In particular, differences will most likely occur in basic node manipulation, as well as in distribution, support for transactions, type manipulation, type checking and attribute monitors. Collecting CAIS-A performance information, for example, is one aspect useful in evaluating a CAIS-A implementation for use on a project's support environment. Further, we expect wide variation among CAIS-A implementations on other quality factors ranging from supporting documentation to maturity and reliability. SEVWG's recommendations on CAIS-A evaluation recognize the need for a comprehensive approach to assessing the quality and suitability of CAIS-A implementations.

Although there are similarities with compiler evaluation, we do not anticipate many projects will be confronted with the problem of which CAIS implementation to select for a particular host. Because of the expense involved in developing a CAIS implementation and its tight binding with the host, there will be fewer implementations of CAIS for a host than there are implementations of Ada compilers for a single host. Further, time and space are typically less critical in a support environment than they are in an embedded application. These factors indicate that evaluating CAIS implementations will be different than evaluating compilers. For instance, since comparisons might be necessary across hardware, we see the need for measures of CAIS performance that are useful independent of hardware speeds. Two possible approaches are:

1. Normalization of wall clock results,

2. Adopting a priori measures by defining elementary CAIS operations and measuring performance in terms of these elementary operations. These operations might be actions such as checking access rights to a node, stepping a single relationship, etc.

7.1 Performance Tests

SEVWG has discussed various measures that should be obtained when collecting performance information about a CAIS implementation. These reflect CAIS characteristics apart from the tools residing on top of CAIS. The list is given as representative rather than exhaustive and includes:

1. traversing a path,

2. spawning a new process (creating a job,)

3. striking new relationships or creating/changing attribute values,

4. defining a given type and view definition structure,

5. referencing the type structure in "type checking,"
   a) for shallow inheritance/view structures,
   b) for deep inheritance/view structures,

6. performing mandatory and discretionary access checks,

7. opening nodes,

8. canceling a transaction/committing a transaction,

9. observing the performance effects of nested transactions,

10. initiating an attribute monitor,

11. importing and exporting node contents,

12. creating and assigning channels and devices.

Additionally, there are measures examining the space required for CAIS entities. These might include disk usage, virtual or physical memory usage for various kinds of nodes, relationships and other entities.

7.2 Alternative Approaches

While we have discussed collecting performance information in the traditional sense, SEVWG also considered other methods short of a suite of test programs. A minimal set of tools could be rehosted (rewritten) to take advantage of CAIS-A. In rehosting to use CAIS-A facilities, the tools could be instrumented so as to report on the efficiency with which they operate. Together with these tools one could develop a set of standard scenarios or scripts that would exercise the underlying CAIS-A implementation in a predefined fashion. Although the expense of rehosting tools to a CAIS-A implementation is probably in itself as expensive or more expensive than developing the performance suite, it has the following added advantages:

1. a usable minimal APSE running on top of CAIS-A would be available early for transition purposes.

2. the tool set could be used as the basis for CAIS-A design evaluation and use studies.

No one has yet determined that the CAIS-A will achieve its goal of increased transportability economically. Using a tool set as a performance measuring tool would require careful management to assure proper use. Users would need to know what portions of an implementation were being reported, and how extensively those portions were being exercised. Doing so correctly would require identification of test objectives being exercised by the tool set and scripts.

Another approach, which also draws considerable debate is the use of validation tests for performance measurement. The validation tests themselves could be instrumented with time and space gathering code and used for the dual purpose of validation and evaluation. Although this approach has the danger of doing neither validation nor performance collection correctly, if done properly it can provide a suite of tests that not only accomplish validation, but also report on the performance characteristics of an implementation in certain typical use scenarios. The validation tests themselves follow a general pattern in which a CAIS context (node structure most commonly) is constructed. One or more tests are then executed against the context to achieve some number of expected results. Finally, other CAIS services are called to determine that the context was modified as called for in the CAIS specification. The process of building the context could include performance gathering hooks without changing the validation process. Once again, management of these dual purpose tests is an important underlying concern.

8.0 CAIS-A EVALUATION AND VALIDATION POLICY

A draft CAIS validation policy has been discussed by the KIT and is included in the final report of that effort [KIT-88.] KIT recommends a multilevel validation policy that would be applicable to CAIS-A. Motivation for such a policy is that CAIS-A will be implemented on such a variety of machines having sufficiently differing purposes and use requirements that warrant having differing validation procedures. Adoption of a policy by the AJPO having multi-level validation requirements has potentially significant impact on the construction of CIVC-A.

9.0 AUTOMATICALLY GENERATE TEST CASE CODE

Another approach reducing the cost of generating validation tests for CAIS-A is to automatically generate the test case code from scenarios. SofTech, in producing CIVC, has adopted a three step process for generating test cases for taxonomy entries. First, Test Objectives are generated from the taxonomy. The Test Objectives describe actions that are to be validated at the level of CAIS entities and actions on those entities, for example, create a secondary relationship. Next, Scenarios are generated to accomplish the Test Objective. Scenarios provide more detail as to how the Test Objective is to be achieved. A Scenario is made up of an input condition, output condition and a CAIS service to be called. That is, the test is to set up the input condition, which is generally a node structure that must be created using some CAIS services, then call the CAIS service, and finally, determine that the output condition has been satisfied.

Often, more than one Scenario is needed to satisfy a Test Objective. The last step in generating the validation test case is to combine and code the Scenarios. Scenarios are combined with others having compatible input conditions to avoid the overhead or reestablishing the same context in several Test Cases. Coding is done using CAIS service calls as determined by the Scenarios.

SEVWG believes that by adopting a somewhat modified and more precise form for expressing Scenarios, that code can be automatically produced. This can be done in a manner that requires more time to generate scenarios from the Test Objectives, but that saves substantial labor in generating test cases themselves. SEVWG recommends that the CIVC-A contractor study the potential for adopting this modified approach. We see that three fundamental steps must be taken to shift to this approach.

9.1 Develop a Scenario Description Language

A new form must be developed for describing Scenarios. The form would be a cross between the current graphical notation and a well defined language having predefined syntax and semantics. Although a graphical representation could be used, we suggest that a notation similar to the Common External Form (lists) be adopted. This Scenario Language must be capable of being easily translated into Ada code, and more specifically, sequences of calls to CAIS interfaces. The language would retain many of the characteristics that SofTech is currently using to express Scenarios; namely, variables to name relations, relationships, nodes and attributes. The ability to express don't care situations in the language is critical to matching one scenario's precondition with those of other scenarios. The language would adopt a technique for expressing node structure specifying the types of nodes and the relations emanating from nodes. One manner to express such structures would be a predicate calculus employing existential and universal quantification. For example, to express that the scenario requires a structural node having an emanating secondary relationship (whose relation and key are unimportant) with the attributes a1, a2 and a3 the following predicate may be used: there exists a node n and relationship r such that n is structural and r is secondary and r has attributes a1, a2 and a3. In this particular case, it doesn't matter where the node n is located, so long as it is accessible. The language would also support naming and use of common descriptions in other descriptions. For example, the description given above may be named "having relationship attributes a1, a2 and a3." The language would allow for this naming and subsequent uses of the name would refer to the predicate. A translator (9.3 below) would be capable of translating the language statements, which would probably be expressed in general list format, into CAIS-A interface calls that generate the structure.

9.2 Scenario Dependency Analysis

The second step in adopting this approach would be to devise a mechanism capable of performing the dependency analysis that SofTech is currently doing manually. This analysis determines which Scenarios have compatible preconditions, and may thus be combined into the same test case. While this analysis minimizes the number of distinct node contexts that the validation suite must create, it does take considerable time to develop. We expect that nearly as good of a job can be done automatically using the Scenario Language. This involves matching the precondition specifications in separate Scenario Descriptions to determine commonality or matching don't care situations. While this tool would be a useful addition to this approach it is not critical to adopting the approach. In particular either the current manual approach could be used with the Scenario Language Descriptions or, at the expense of execution efficiency of the resulting validation suite, no minimization need be performed.

9.3 Translator from Scenario Descriptions to Ada

Finally, a translator must be constructed, as referenced in item 9.1, that converts Scenarios in the definition technique into test case code. This translator would use common compiler construction techniques to analyze a language far simpler than a general purpose programming language. We suspect that translation templates would be recorded for use by the translator. When the translator recognized a node structure component matching one of its templates it would simple emit the code needed to build that component. A majority of the overhead for developing the translator could be included in the syntactic description of the Scenario Language. Using a compiler-compiler tool such as YACC and LEX would greatly ease this effort. Although the resulting translator may not be optimized to run efficiently, its performance characteristics (aside from correct functionality) are irrelevant.

10.0 DEVELOP A NEW TAXONOMY FOR CIVC-A

To reduce the CAIS specification to a more manageable form, SofTech developed a taxonomy in which CAIS entities, such as nodes, attributes, relationships, processes, devices, etc. exist together with the actions that CAIS provides for those entities. The actions generally involve defining, accessing and removing CAIS entities. Entries in the SofTech taxonomy are CAIS entities representing the actions that may be taken on those entities. Thus the taxonomy can be taken as a requirement for the CIVC, in that, the validation mechanism must determine that the action can be made (as described in the specification) on the entity represented.

The taxonomy is used to populate test objectives in a given topical area. Test objectives are generated directly from the specification using the taxonomy as a checklist in a given area. While SEVWG sees the benefit and need for CIVC to adopt this approach, we also believe that it would be beneficial to have tests traceable to the specification itself. In the case of CIVC, the taxonomy provides a dense representation for the specification. Although the specification details how one creates an access relationship for example, the SofTech taxonomy abbreviates the specification by telling what actions are possible on access relationships, rather than detailing which interfaces build, reference or delete access relationships.

SEVWG discussions on the taxonomy indicate that the current approach needs modification to accommodate CAIS-A, and that the taxonomy can be more directly related to the specification. For example, with the introduction of typing within CAIS-A comes the potential to create entities and act on them, in a limited manner, which cannot be registered in the SofTech taxonomy. Using CAIS-A typing it is possible to create node types, for example, that are not one of those standardly defined in CAIS-A. The operations and this capability might be missed by the validation suite for CAIS-A if, strictly speaking, the same taxonomy approach is used for CAIS-A.

In a CAIS-A Appendix, the standard predefines a definition structure (a set of node, relation and attribute type definitions) for CAIS-A. While it is expected that users will enhance this definition structure, it provides the basis for elementary operations needed. This set of predefined definitions would make a better basis for the taxonomy than the current table approach. The primary benefit of doing so would be that the validation suite would be directly based upon the specification rather than an interpretation of the specification. This would allow test objectives to be derived directly from nodes in the predefined definition structure. SEVWG recommends that CIVC-A initially investigate the suitability of using this approach. In particular, there are questions which must be resolved if this approach is to be adopted. Will it work for all Test Objectives; what is not in the predefined structure that must be evaluated? SEVWG has identified exceptions and static semantics as example components that must be validated, but which do not appear in the predefined structure. Are there other examples?

Another question that must be answered is: What is the suitability of this approach to coverage analysis? One of the main benefits of the taxonomy developed by SofTech for CIVC is that coverage analysis of the taxonomy is quite straightforward. One can easily analyze the table entries in the taxonomy which have been developed into test programs. The same analysis may be available when basing test objectives on the predefined type structure in CAIS-A. Although the questions of completeness of the structure complicate coverage analysis, having a taxonomy based on the specification and also having coverage analysis are the two main benefits of this approach.

11.0 MAINTENANCE OF THE CIVC-A TEST SUITE

The CIVC consists of two software products, the Test Administrator and the Test Suite. Further, substantial electronic documentation, called the Framework, exists supporting the validation suite. With the exception of the Test Suite itself, the other electronic products do not have the strict requirement that they be CAIS conforming tools or data. That is, for example numerous CIVC data exist on special purpose software currently available on the Macintosh. This presents an issue regarding the long term maintenance of this information. While it would be desirable, from the standpoint of evolution of CIVC-A, to have all software and electronic data hosted on CAIS-A, current implementations of CAIS-A prohibit development of this information. Thus the issue that needs to be resolved is: How can we have a CIVC-A that will in time be maintainable on a CAIS-A environment that can be developed in the short term?

While this question is currently unanswerable, various approaches to its solution need to be identified and pursued. SEVWG recommends that the CIVC-A contractor, early in the development of CIVC-A, study and report on the feasibility of developing a minimal CAIS-A tool that allows Framework documentation to reside on a CAIS-A environment.

The Framework is one of the most distinguishing products of the CIVC. It provides users with a friendly user interface to the collection of documentation pertaining to the CIVC. The framework allows a user to navigate among the taxonomy, test objectives and scenarios. The framework, which is a collection of graphical and textual information, provides the relationships between different forms of the tests within CIVC.

The framework currently resides on the hypertext mechanisms available with the guide product on the Macintosh. SEVWG sees the significance of the framework and its user interface for the purpose of maintenance of the CIVC product. At least as important, however, is the proliferation of CAIS-A hosted tools. SEVWG believes that whenever possible CAIS related contracts should create tools that contribute to a CAIS based APSE. Further, maintenance accounts for such a significant portion of the cost of a system. This will surely be the situation with CIVC-A. SEVWG recommends that the CIVC-A contractor conduct a feasibility study on the possibility of implementing a tool on top of a CAIS-A implementation that could be used to host a version of CIVC-A's framework product. Even though the tool development to host such a product could be quite extensive, we believe that a tool possessing reasonable functionality could be constructed in a manner beneficial to the CIVC-A program and equally beneficial to the CAIS. Further, subtle but important improvements could be made over the Macintosh version of Framework currently used on CIVC. The first improvement is that the Framework could be made to easily reference the actual source code of the CIVC-A suite. This would provide a Framework in which all aspects of the test suite and its traceability back to the CAIS-A specification could reside. The second improvement would be to allow the Framework to be loaded in an automatic fashion. Currently, SofTech is providing the links between the components of the Framework in a manual fashion. This process makes it difficult to assure completeness, correctness and consistency of the relationships between test objectives, scenarios, the specification and the taxonomy. If a tool were constructed on top of CAIS-A, it could provide the capabiltiy for automatic loading of the links among objects.

12.0 SUMMARY RECOMMENDATIONS

In this section, we present in summary form the issues presented in this document. These are presented in the following format. This issue itself is briefly explained. SEVWG recommendation for resolving the issue is next given. This most often is done by identifying a specific product that will aid in deciding on the issue. Any dependencies with other issues and the appropriate activity for developing the product are also identified.

1. Size and Cost, reference discussion in Section 4. SEVWG recommends an initial study by the CIVC-A contractor to estimate, in more precise terms than SEVWG, the impact to the validation suite that the move to CAIS-A will have. We recommend that the study should include:

   a. Suitability of existing CIVC test objectives to CAIS-A.

   b. Impact of new topical areas introduced in CAIS-A (typing, transactions, attribute monitors, distribution, access control, Input/Output and basic node model changes) estimate:

   c. The number of new test objectives that must be developed for functionality in CAIS-A that currently exists in CAIS-1838. The purpose of this study is to provide the information needed to determinedirections that CIVC-A should take. The information will provide the basis for obtaining funding for CIVC-A. Further, it will provide the basis for technical direction on the development of the suite.

   d. The CIVC-A contractor should generate a summary report on the development of CIVC. The report would estimate or summarize information on the level of effort and machine resources needed to populate CIVC with tests. The report should include:

      o the number of scenarios generated per test objective in CIVC

      o the number of scenarios tested in a test class and the mean number of scenarios that were able to be combined through dependency analysis.

      o effort required to identify, develop, coordinate (register in framework), and review
        a. test objectives
        b. scenarios
        c. Ada code in test cases

2. Test Selection Criteria, reference Section 5. The CIVC-A contractor should develop a plan for the test selection criteria to be used in developing CAIS-A test objectives. The plan should be based on those criteria identified in Section 5 or argue

alternative methods. There exist many practical approaches to selecting tests that combine two or more of the several criteria listed in section 5. SEVWG suggests that those used to develop CIVC-A be reviewed by CIVCWG at the onset of the CIVC-A activity, and that portions of the total number of test objectives developed be derived from selected criteria.

3. CIVC and CIVC-A Review Board, reference Section 6. To aid in the use and maintenance of CIVC(-A) and to assure consistent and reliable results from its use, SEVWG recommends the creation of a CIVC review board. The board would be formed by AJPO and the E&V Task to perform activities as described in Section 6.

4. Automatically Generate Test Case Code, reference Section 9. SEVWG believes that by adopting a somewhat modified and more precise form for expressing scenarios, that code can be automatically produced. This can be done in a manner that requires more time to generate scenarios from the test objectives, but that saves substantial labor in generating test cases themselves. SEVWG recommends that the CIVC-A contractor study the potential for adopting this modified approach. We suggest that the three steps outlined in Section 9 be examined by the CIVC-A contractor with respect to feasibility and effort required to shift to this approach.

5. A New Taxonomy for CIVC-A, reference Section 10. SEVWG recommends that the CIVC-A contractor investigate alternative methods for representing CIVC-A test requirements. One method to achieve this might be to use the Appendix of CAIS-A which predefines types for the standard. Another might be to develop an index system allowing requirements existing in the standard to be individually named and referenced for completeness coverage.

6. Framework, reference Section 11. SEVWG recommends that the CIVC-A contractor, early in the development of CIVC-A, study and report on alternatives for hosting the CIVC-A framework.

APPENDIX A - ACRONYMS

ACVC	Ada Compiler Validation Capability
AJPO	Ada Joint Program Office
APSE	Ada Programming Support Environment
CAIS	Common APSE Interface Set (used generically to refer to characteristics shared by both references CAIS and CAIS-1838 below.)
CAIS-A	Common APSE Interface Set Revision A (See reference [CAIS-88].)
CAIS-1838	Common APSE Interface Set (See reference [CAIS-86].)
CIVC	CAIS-1838 Implementation Validation Capability
CIVC-A	CAIS-A Implementation Validation Capability
E&V Team	Evaluation and Validation Team
I/O	Input and/or Output
KAPSE	Kernel APSE
KIT/KITIA	KAPSE Interface Team (Government) / KAPSE Interface Team from Industry and Academia
REQWG	Requirements Working Group of the E&V Team
SEVWG	Standards Evaluation and Validation Working Group

APPENDIX B - SEVWG MEMBERSHIP

Dr. Tim Lindquist, Arizona State University - Chairman
Ms. Karyl Adams, cj kemp systems inc.
Mr. Lynn Chilson, SofTech Houston
Mr. Jeff Facemire, SPC (formerly SofTech Houston)
Mr. Jack Foidl, TRW San Deigo
Mr. Kurt Gutzmann, SofTech Houston
Mr. Kevin Hackett, SofTech San Diego
Mr. John McBride, SofTech Houston
Mr. Gary McKee, McKee Consulting
Mr. Lloyd Stiles, US Navy FCDSSA

APPENDIX C - REFERENCES

[CAIS-88] MIL-STD-1838A Military Standard Common Ada Programming Support Environment (APSE) Interface Set (CAIS) (Revision A) Vol's I thru III. April 6, 1989.

[ADA-83] ANSI/MIL-STD-1815A Reference Manual for the Ada Programming Language, February 17, 1983.

[CAIS-86] DOD-STD-1838 Common Ada Programming Support Environment (APSE) Interface Set (CAIS). October 6, 1986.

[RAC-85] DOD Requirements and Design Criteria for the Common APSE Interface Set (CAIS), KIT/KITIA, October 1986.

[MUN-88] Munck, et.al, An Overview of DOD-STD-1838A (proposed), The Common APSE Interface Set, Revision A. Proc. ACM Practical Development Environments, November 1988.

[SEV-85] Component Validation Procedures (CVP) Document, E&V Team (SEVWG), December 1985.

[SEV-87] Issues and Strategies for Evaluation and Validation of CAIS Implementations, Based on DOD-STD-1838, E&V Team (SEVWG) 1987.

[KOP-85] The Proposed CAIS Validation Policy Document, as presented to the E&V Team by Maj. Al Kopp 1985.

[REQ-87] Requirements for the Evaluation and Validation of Ada Programming Support Environments, E&V Team (REQWG) 1987.

[KIT-88] Final Report of the KAPSE Interface Team (KIT), KIT, 15 October 1988.

[IAS-87] Issues and Strategies for the Evaluation and Validation of CAIS DOD-STD-1838 Implementations Version 1.0, APSE E&V Team, 1987.

[DURAN-84] An Evaluation of Random Testing, IEEE Transactions on Software Engineering, SE-10, 4, July 1984.

APPENDIX E

REPORT

of

INDEPENDENT VALIDATION AND VERIFICATION

Ada Compiler Evaluation Capability

Prepared by:	Mike Burlakoff Rt 2, Box 324 Springfield, Mo 65802
Title:	Consultant, Computer Scientist
Contract:	F33615-88-C-1706
Task:	89-4-5
Air Force Technical Monitor:	Raymond Szymanski, WRDC/AAAF-3
Date of Report:	22 January 1990

ACKNOWLEDGEMENTS

I wish to first express my appreciation to Mr. Ray Szymanski who was responsible for my being assigned to this task. His technical guidance and direction added to the results of this project.

I also wish to thank the members of the Dayton Office of The Analytic Sciences Corporation (TASC). In particular, Mr. Larry Qualls and Mr. Michael LaFollette were helpful in providing the necessary administrative support throughout the period.

Appreciation is also extended to the members of the Evaluation & Validation (E&V) Ada Compiler Evaluation Capability Working Group (ACECWG) for their time and helpful recommendations and suggestions in the development of the Keyword Index. The following members were especially helpful in this effort: Nelson Weiderman (ACECWG Chairman), Dan Eiler, and Greg Gicca.

ABSTRACT

The activities of this project consisted of three separate (but related) efforts: 1) Review of the ACEC test suite for the purpose of developing a Keyword Index, 2) Performance of an Independent Validation and Verification of the Ada Compiler Evaluation Capability (ACEC) and 3) Technical reviews of several ongoing E&V and ACEC efforts. An overview of the first two areas is given below:

The ACEC is a suite of approximately 1,074 tests designed to evaluate the performance of Ada compilation systems. At present the suite of tests is not formally organized by test category. It was determined that an index which gives the primary, secondary and incidental purposes of the tests would be useful in later namings and/or categorization of tests. To complete this requirement, the test suite was reviewed and the tests were indexed according to a taxonomy which listed primary, secondary and incidental purposes.

The ACEC test suite and support software Version 1 was delivered to the Air Force by the Boeing Company ACEC contractor in the Summer of 1988. In response to user and Air Force requirements, a follow-on Version 2 of the system was under development. The Air Force determined that it would be desirable to perform an Independent Validation and Verification (IV&V) of this delivery. My task was to accomplish this IV&V. The emphasis was to evaluate requirements and design documentation and to perform and verify formal testing prior to system delivery.

I. INTRODUCTION

In 1975 the Department of Defense (DoD) High Order Language Working Group was formed with the goal of establishing a single high order language for use in DoD systems (in particular, in Embedded Computer Systems). Following establishment of technical requirements and international competition, the Ada language as currently defined in (1) was selected. One of the major goals of Ada is to reduce the rapidly increasing costs of software development and maintenance in military systems.

Early in the development process it was realized that the acceptance and benefits derived from a common language could be increased substantially by the development of an integrated system of software development and maintenance tools. The requirements for such an Ada Programming Support Environment (APSE) were stated in the STONEMAN (2) document. STONEMAN identifies the APSE as support for "the development and maintenance of Ada application software throughout its life cycle." (2)

In June 1983 the Ada Joint Program Office (AJPO) proposed the formation of the E&V Task and a tri-service APSE E&V Team, with the Air Force designated as the lead service. In October 1983 the Air Force officially accepted responsibility as the lead service for the E&V Task.

The purpose of the E&V Task is to provide a focal point for addressing the need to provide the capability to assess APSEs and their components and to determine their conformance to applicable standards, such as the Ada Language Standard (1). This will be accomplished by (1) identifying and defining specific technology requirements, (2) developing selected elements of the required technology, (3) encouraging others to develop some elements, and (4) collecting information describing existing elements. This information will be made available to DoD components, other government agencies, industry and academia (3).

The Ada Compiler Evaluation Capability (ACEC) is one of the technology initiatives of the E&V effort. The E&V team proposed the initial ACEC concept and has made valuable contributions in the guidance and direction of this technology. The Boeing Military Airplanes (BMA) Software and Languages Organization is the contractor responsible for the ACEC work. For a technical discussion of the ACEC from a user's perspective, refer to the documents referenced in (4) and (5).

This report summarizes the Keyword Index work in Section II and the IV&V activity in Section III. In addition to these efforts, a number of technical reviews relating to E&V and ACEC activities were accomplished. The results of these were usually written comments to the Air Force, E&V Team and/or the ACEC contractor.

A copy of the Keyword Index, significant IV&V comments/recommendations and other technical review results are included as attachments to this report. Appendix I gives a list of these attachments.

During the period of this effort, attendance at the quarterly E&V team meetings was required and was accomplished.

II. ACEC KEYWORD INDEX ACTIVITIES:

Following are the procedures that were used to develop the Index:

1. One of the more difficult tasks was to determine the methods and techniques for developing the Index. As an aid in making those decisions, informal technical papers from Boeing, the Air Force and members of the E&V team, as well as prior technical discussions were used. The decision was made to use a taxonomy approach which would index each test according to primary purpose in the form:

main_category.lower_level_category.lower_level...

2. An initial review of the Ada Language Reference Manual (LRM) (specifically ANSI/MIL-STD-1815A, see reference 1) was accomplished. The purpose was to index all LRM language features which (in the opinion of the investigator) should be tested by a performance test. Thus, the LRM was the basis for the initial Keyword Index taxonomy.

3. The entire suite of tests was then reviewed. Each test was assigned a unique Index which represented the primary purpose of the test. Some of the indexes were already listed from the LRM review. Many other indexes were added during the review. A LRM reference paragraph was also assigned to reflect the primary purpose of the test by LRM paragraph. The Keyword Index consists of three parts: Taxonomy Keyword, Test Name, and LRM Reference.

4. The completed Keyword Index resulted in a large amount of textual information. To aid in the presentation of the information, a set of support software was developed. This software enabled the presentation in three different orders and forms as follows: By 1) Keyword Index (with Test Name and LRM Reference), 2) Test Name (with LRM Reference and Keyword) and 3) LRM Reference (with Test Name and Keyword).

At the September 1989 quarterly E&V meeting, the Ada Compiler Evaluation Capability Working Group (ACECWG) reviewed the above Keyword Index and made several recommendations and suggestions. The most significant of these was that the Index should contain Secondary and Incidental purposes of the tests (if any). This recommendation was approved and a re-review of the test suite was accomplished and a new version of the Index was completed and delivered to the contractor, Air Force and members of the ACECWG. Another review of the Index was made by the ACECWG at the Dec. 89 quarterly E&V meeting. Several minor changes were recommended and implemented. A Dec. 89 version has been completed and delivered (6). It should be noted that the comments and recommendations of the ACECWG have been helpful and will result in a better final technical product.

III. ADA COMPILER EVALUATION CAPABILITY (ACEC) IV&V ACTIVITIES:

The initial phase of the Ada Compiler Evaluation Capability (ACEC) test suite and support software was delivered to the Air Force by the Boeing Company ACEC contractor in Summer 1988. In response to Air Force and user requirements, a Version 2 of the system is being developed. The Air Force determined that it would be desirable to perform an Independent Validation and Verification (IV&V) of this version.

The ACEC Version 2.0 development consists of additional performance tests and assessors for the following Ada Compilation support areas: Diagnostics, Debugger and Program Library System. In addition, a ACEC Single System Analysis capability is being developed.

The IV&V consisted of review and evaluation of the ACEC Version 2.0 requirements and design documentation, attendance at appropriate design reviews and performance and observation of formal testing prior to Air Force contractual delivery.

The Boeing Company developed a number of working papers outlining the proposed approach and design of the above systems. These were reviewed and comments were given to the Air Force and Boeing. The requirements, design, and testing documentation was also reviewed and written comments were delivered. The Preliminary and Critical Design reviews were attended. Prior to each of these, documentation was reviewed and appropriate technical discussion was given at the reviews.

Three separate visits were made to the Boeing Co. for the purpose of performing testing and observation of contractor testing. Following is a summary of that activity:

The completed set of new performance tests were compiled and executed. A sample of the Debugger Assessor scenarios were executed. The documentation describing the Debugger Assessor was reviewed as well as the template for recording testing results. A majority of the Library Assessor scenarios were executed and evaluated. The Library Assessor documentation and results templates were evaluated. A sample of the Diagnostics Assessor and associated user documentation was likewise executed and evaluated. The Single System Analysis system was still under development and could not be executed. However, the documentation was evaluated. In addition to testing and evaluation by the investigator, observation of a sampling of contractor Formal Qualification Testing (FQT) was accomplisted.

As a result of this IV&V, both verbal and written comments were given to the Air Force and Boeing.

APPENDIX I

Following is a list of attachments to this report.

NUMBER	TITLE	DATE
1.	ACEC Keyword Index	15 DEC 89
2.	Keyword Index Update	15 DEC 89
3.	Keyword Index Update	25 OCT 89
4.	ACEC Recommendations	18 DEC 89
5.	IV&V Summary (SEP 28, 29 & OCT 19, 20)	3 NOV 89
6.	IV&V Testing at Boeing, OCT 19, 20, 89	25 OCT 89
7.	Library Assessor Comments (IV&V at Boeing)	20 OCT 89
8.	Single System Analysis Comments (IV&V at Boeing)	19 OCT 89
9.	ACEC IV&V Testing	2 OCT 89
10.	ACEC Test Plan/Procedures Comments	13 OCT 89
11.	ACEC Support S/W SPS, 5 JUN 89	13 JUN 89
12.	Comments to ACEC Operational SW SPS, 5 JUN 89	16 JUN 89
13.	ACEC Support S/W SRS Comments	24 APR 89
14.	ACEC Single System Analysis Working Paper Cmts	1 APR 89
15.	ACEC Database Working Paper Comments	30 MAR 89
16.	ACEC Library Robustness Paper Comments	28 MAR 89
17.	ACEC Diagnostics Working Paper Comments	21 MAR 89
18.	ACEC WP 22 (Symbolic Debugger) Comments	17 MAR 89
19.	Topics for ACEC V3 and V4	22 NOV 89
20.	An Approach for an Input Data File for Median	15 JUN 89
21.	ACECWG/PC Comments	13 MAR 89
22.	ACEC/PIWG: An E&V'rs Viewpoint	11 MAR 89
23.	CLASSWG - Ref Man Tools Chapter 5	8 MAR 89

24.	PTS Procedures Comments	23 AUG 89
25.	Ada Compiler Eval P&G Comments	1 APR 89
26.	ACSH Comments	14 JAN 89

REFERENCES

1. DoD. Ada Programming Language, ANSI/MIL-STD-1815A. 22 JAN 83.
2. DoD. "Stoneman". Requirements for Ada Programming Support Environments. FEB 80.
3. DoD. Evaluation & Validation (E&V) Plan. Version 4.0. 4 JUN 87.
4. Boeing. ACEC Technical Operating Report. User's Guide. 10 JUN 88.
5. Boeing. ACEC Technical Operating Report. Reader's Guide. 10 JUN 88.
6. Burlakoff. ACEC Keyword Index. 15 DEC 89.

ACEC KEYWORD INDEX
(With LRM References and Applicable Tests)

December 15, 1989

Notes: 1) A test in parenthesis means that its sole purpose is for performance comparison with the preceding test.
2) Test names with no suffix, indicate that the PRIMARY purpose of this test is this feature. A "/s" means that this is a SECONDARY purpose while a "/i" indicates that this feature is INCIDENTALLY used in this test.

access.operations	3.8.2	ss154,ss155,ss256,ss257,ss648, ss746(ss744..45),ss748,ss805, ss161/s,ss162/s,ss163/s, ss164/s,ss165/s,ss166/s, ss167/s,ss739/s
application.avionics	1.1.2	arti_asum,arti_atan2,arti_cos, arti_fmod,arti_ifpm_control, arti_ifpm_init,arti_ifpm_io, arti_ifpm_rotors,arti_nairini, arti_nscni,arti_nutmini, arti_sin,ew,forward_euler1, forward_euler2
application.data_encryption_standard	1.1.2	des1,des2,des3,des4,des4a, des5,des5a,des6,des6a,des7, des7a
application.error_correcting_code	1.1.2	Reed_Solomon_0,Reed_Solomon_1, Reed_Solomon_2,Reed_Solomon_3, Reed_Solomon_4
application.integration	1.1.2	ss398,ss402
application.kalman_filter	1.1.2	Kalman
application.lag_filter	1.1.2	ss397,ss401
application.polynomial.coding_style	1.1.2	ss120,ss121,ss122,ss123
application.simulation	1.1.2	simulate_BMBAT, simulate_EMRPM, simulate_HMPROTO, simulate_KMDUMP, simulate_QMPITCH, simulate_RCWFRDET, simulate_rmkeying, simulate_UMNAV
application.symmetric_deadzone	1.1.2	ss399,ss403
application.symmetric_limiter	1.1.2	ss400,ss404
array.aggregates	4.3.1	ss775,ss778,ss764/s,ss765/s, ss766/s,ss767/s,ss768/s
array.dynamic	3.6	ss419(ss420)
array.operations	3.6.2	ss.17,ss.18,ss.19,ss.57,ss.77, ss.78,ss.79,ss.80,ss.81,ss301, ss645,ss646,ss647,ss758,ss759, ss760,ss761,ss762,ss763,ss774,

		ss776,ss777,ss.53/s,ss.54/s, ss.55/s,ss.58/s,ss.75/s, ss.76/s,ss120/s,ss168/s, ss169/s,ss170/s,ss172/s, ss173/s,ss174/s,ss175/s, ss192/s,ss193/s,ss194/s, ss235/s,ss243/s,ss246/s, ss258/s,ss259/s,ss284/s, ss285/s,ss309/s,ss388/s, ss429/s,ss430/s,ss511/s, ss512/s,ss518/s,ss519/s, ss520/s,ss553/s,ss554/s, ss405/i,ss406/i,ss409/i, ss410/i,ss411/i,ss419/i, ss420/i,ss428/i,ss432/i, ss433/i,ss434/i,ss435/i, ss436/i,ss437/i,ss438/i, ss439/i,ss442/i,ss443/i, ss477/i,ss508/i,ss509/i, ss516/i,ss517/i,ss535/i, ss536/i,ss541/i,ss542/i, ss542x/i,ss545/i,ss557/i, ss562/i,ss596/i,ss597/i, ss648/i,ss652/i,ss653/i, ss654/i,ss655/i,ss656/i, ss657/i,ss658/i,ss659/i, ss660/i,ss661/i,ss662/i, ss663/i,ss664/i,ss665/i, ss666/i,ss667/i,ss668/i, ss669/i,ss670/i,ss671/i, ss672/i,ss673/i,ss674/i, ss675/i,ss676/i,ss677/i, ss678/i,ss679/i,ss680/i, ss681/i,ss687/i,ss688/i, ss689/i,ss690/i,ss691/i, ss692/i,ss693/i,ss694/i, ss695/i,ss696/i,ss697/i, ss698/i,ss699/i,ss700/i, ss701/i,ss702/i,ss703/i, ss704/i,ss705/i,ss706/i, ss707/i,ss708/i,ss709/i, ss710/i,ss711/i,ss712/i, ss713/i,ss714/i,ss715/i, ss716/i,ss731/i,ss732/i, ss734/i,ss735/i,ss749/i, ss750/i
array.constraints	3.6.1	ss596(ss597),ss597
boolean.arrays.packed	4.5	ss337,ss338,ss339,ss340,ss341, ss342,ss343,ss344,ss345,ss347, ss348,ss349,ss524,ss525,ss526, ss764,ss765,ss766,ss767,ss768, ss346/i,ss353/i,ss500/i, ss501/i,ss502/i,ss506/i

boolean.arrays.unpacked	4.5	ss326,ss327,ss328,ss329,ss330, ss331,ss332,ss333,ss334,ss336, ss351,ss352,ss346/i,ss353/i, ss486/i
boolean.expressions	4.5	ss.72,ss101,ss177,ss228,ss229, ss486,ss487,ss488,ss489, ss492,ss499,ss686y,ss686x, ss.73/s,ss.74/s,ss176/s, ss227/s,ss230/s,ss231/s, ss232/s,ss280/s,ss326/s, ss327/s,ss329/s,ss330/s, ss331/s,ss332/s,ss333/s, ss334/s,ss335/s,ss336/s, ss337/s,ss338/s,ss339/s, ss340/s,ss341/s,ss342/s, ss343/s,ss344/s,ss345/s, ss346/s,ss347/s,ss348/s, ss349/s,ss350/s,ss351/s, ss352/s,ss353/s,ss500/s, ss501/s,ss502/s, ss145/i,ss146/i,ss147/i, ss314/i,ss315/i,ss316/i, ss317/i,ss318/i,ss323/i, ss464/i,ss598/i,ss599/i, ss602/i,ss604/i,ss805/i
boolean.record	3.5.3	ss682,ss683,ss684,ss685,ss717, ss718,ss719,ss720
classical_benchmark.ackermann's	1.1.2	acker1,acker2
classical_benchmark.cube_placing	1.1.2	puzzle
classical_benchmark.dining_philosophers	1.1.2	task.7,task.8,task.9,task10, task25
classical_benchmark.dhrystone	1.1.2	dhry1,dhry2,dhry3
classical_benchmark.eight_queens	1.1.2	queens
classical_benchmark.GAMM_measure	1.1.2	gamm,gamm2
classical_benchmark.numerical.CFA	1.1.2	auto,bmt,heapify,lu,runge, target
classical_benchmark.num.knuth_loops	1.1.2	loop.0,loop.1,loop.2,loop.3, loop.4a,loop.4b,loop.4c, loop.5,loop.6,loop.7,loop.8, loop.9,loop10,loop11,loop12, loop13,loop14,loop15,loop16, loop17
classical_benchmark.num.livermore_loops	1.1.2	kernel.1,kernel.2,kernel.3, kernel.4,kernel.5,kernel.6, kernel.7,kernel.8,kernel.9, kernel10,kernel11,kernel12, kernel13,kernel14,kernel15, kernel16,kernel16_goto, kernel17,kernel18,kernel19, kernel20,kernel21,kernel22, kernel23,kernel24

classical_benchmark.prime_number	1.1.2	seive
classical_benchmark.search	1.1.2	search,ssearch,ssearch2
classical_benchmark.sort	1.1.2	bsort1,bsort2,ciqsort,iqsort, qsort1,qsort2,shell1,shell2, merge1,merge2
classical_benchmark.whetstone	1.1.2	whet1,whet2,whet3,whet4
conversion.fixed	3.5.10	ss107,ss108,ss466,ss467,ss721, ss722,ss723
conversion.float	4.6	ss..2,ss..8,ss.13,ss289,ss290, ss283/s
conversion.integer	4.6	ss.12,ss233,ss234,ss300,ss468, ss277/s,ss282/s,ss303/s
conversion.null	4.6	ss241
conversion.packed_to_unpacked	4.6	ss335,ss346,ss353
conversion.unchecked_conversion	13.10.2	ss259(ss258),ss500,ss501, ss502,ss506
conversion.unpacked_to_packed	4.6	ss350
delay	9.6	ss455,ss458,ss459,delay.1, delay.2,delay.3,delay.4, delay.5,delay.6,delay.7, delay.8,delay.9,delay10, delay11,delay12,delay13, delay14
exception.constant.propagation	11.6	ss316,ss317,ss529
exception.handling	11.4	ss379,ss380,ss381,ss382,ss383, ss384,ss527,ss528,funcexp, ss543/s, ss598/i,ss599/i,ss602/i, ss604/i,ss638/i,ss741/i
exception.numeric_error	11.1	ss313,ss369
exception.raise	11.3	ss117,ss311,ss312,ss755/s, ss757/s
fixed.operations	3.5.10	ss109,ss110,ss460,ss461,ss462, ss463,ss464,ss465
float.operations	3.5.8	ss..1,ss..3,ss..4,ss..5,ss..6, ss211,ss286,ss287,ss288,ss302, ss308,ss315,ss324, ss591(592..4),ss592(593..4), ss593(ss594),ss594,ss643x ss.20/s,ss.21/s,ss.22/s, ss.23/s,ss.24/s,ss.25/s, ss.59/s,ss.60/s,ss.61/s, ss.62/s,ss.63/s,ss.64/s, ss.65/s,ss.66/s,ss.71/s, ss134/s,ss135/s,ss136/s, ss150/s,ss216/s,ss219/s, ss220/s,ss256/s,ss257/s, ss293/s,ss294/s,ss295/s, ss296/s,ss297/s,ss298/s, ss299/s,ss301/s,ss314/s, ss316/s,ss317/s,ss318/s, ss323/s,ss389/s,ss390/s,

		ss391/s,ss392/s,ss552/s, ss575/s,ss576/s,ss577/s, ss578/s,ss579/s,ss580/s, ss581/s,ss582/s,ss583/s, ss585/s,ss588/s,ss589/s, ss590/s,ss595/s,ss606/s, ss607/s,ss609/s,ss643/s, ss779/s,ss780/s,ss781/s, ss782/s,ss783/s,ss784/s, ss785/s,ss786/s,ss787/s, ss788/s,ss789/s,ss790/s, ss791/s,ss792/s,ss793/s, ss794/s,ss795/s,ss796/s, ss797/s,ss798/s, ss.67/i,ss.68/i,ss.69/i, ss.70/i,ss120/i,ss121/i, ss122/i,ss123/i,ss141/i, ss142/i,ss143/i, ss154/i,ss155/i,ss210/i, ss218/i,ss226/i,ss233/i, ss234/i,ss262/i,ss263/i, ss291/i,ss292/i,ss304/i, ss305/i,ss306/i,ss307/i, ss397/i,ss398/i,ss399/i, ss400/i,ss401/i,ss402/i, ss403/i,ss404/i,ss406/i, ss407/i,ss413/i,ss414/i, ss415/i,ss416/i,ss417/i, ss418/i,ss431/i,ss432/i, ss433/i,ss434/i,ss435/i, ss436/i,ss437/i,ss442/i, ss443/i,ss444/i,ss448/i, ss450/i,ss454/i,ss467/i, ss485/i,ss511/i,ss512/i, ss513/i,ss514/i,ss515/i, ss529/i,ss530/i,ss531/i, ss532/i,ss533/i,ss534/i, ss535/i,ss536/i,ss547/i, ss548/i,ss549/i,ss586/i, ss621/i,ss622/i,ss623/i, ss624/i,ss625/i,ss626/i, ss627/i,ss628/i,ss629/i, ss630/i,ss631/i,ss632/i, ss633/i,ss645/i,ss646/i, ss647/i,ss649/i,ss650/i, ss753/i,ss754/i,ss758/i, ss759/i,ss760/i,ss761/i, ss762/i,ss763/i
generics.subprogram	12.	ss148,ss149(151),ss150,ss478, ss621,ss622,ss623,ss624,ss625, ss626,ss627,ss628,ss629,ss630, ss631

integer.int_32.operations	3.5.5	ss270,ss271,ss272,ss273,ss274, ss275,ss276,ss277,ss278,ss280, ss282,ss283,ss284,ss282/s
integer.MOD	3.5.5	ss102,ss199/s,ss446/i
integer.operations	3.5.5	ss..7,ss..9,ss.10,ss.11,ss.46, ss201,ss202,ss203,ss268,ss269, ss281,ss561,ss729,ss744,ss745, ss.40/s,ss.41/s,ss.42/s, ss.43/s,ss.44/s,ss.45/s, ss.47/s,ss.48/s,ss.49/s, ss.50/s,ss.51/s,ss.52/s, ss.56/s,ss137/s,ss189/s, ss195/s,ss196/s,ss197/s, ss198/s,ss217/s,ss221/s, ss393/s,ss394/s,ss395/s, ss396/s,ss503/s,ss550/s, ss551/s,ss556/s,ss560/s, ss566/s,ss567/s,ss568/s, ss569/s,ss570/s,ss571/s, ss572/s,ss573/s,ss574/s, ss584/s,ss608/s,ss610/s, ss611/s,ss753/s,ss754/s, ss.95/i,ss.96/i,ss.97/i, ss.98/i,ss102/i,ss103/i, ss117/i,ss129/i,ss130/i, ss131/i,ss138/i,ss139/i, ss140/i,ss190/i,ss191/i, ss200/i,ss209/i,ss213/i, ss214/i,ss241/i,ss264/i, ss265/i,ss266/i,ss267/i, ss268/i,ss269/i,ss364/i, ss366/i,ss367/i,ss369/i, ss372/i,ss373/i,ss374/i, ss375/i,ss384/i,ss385x/i, ss386/i,ss423/i,ss424/i, ss425/i,ss426/i,ss427/i, ss428/i,ss429/i,ss430/i, ss431/i,ss440/i,ss441/i, ss445/i,ss446/i,ss447/i, ss449/i,ss451/i,ss466/i, ss468/i,ss469/i,ss470/i, ss471/i,ss472/i,ss473/i, ss474/i,ss475/i,ss476/i, ss490/i,ss491/i,ss500/i, ss501/i,ss502/i,ss506/i, ss507/i,ss511/i,ss512/i, ss558/i,ss559/i,ss563/i, ss564/i,ss565/i,ss612/i, ss634/i,ss635/i,ss636/i, ss637/i,ss638/i,ss639/i, ss640/i,ss651/i,ss652/i, ss752/i,ss755/i,ss756/i,

		ss757/i,ss774/i,ss775/i, ss776/i,ss777/i,ss778/i
integer.REM	4.5.5	ss103,ss447/i,ss204/s, ss276/i,ss362/i,ss363/i, ss447/i
IO.direct	14.2	IO11,IO12,IO13,IO14,IO15,IO16
IO.sequenctial	14.2	IO17,IO18,IO19,IO20,IO21,IO22, IO23
IO.Text_IO	14.3	IO.0,IO.1,IO.2,IO.3,IO.4,IO.5, IO.6,IO.7,IO.8,IO.9,IO10, ss537/i,ss538/i,ss539/i, ss540/i,ss686/i
IO.Text_IO.float.string	14.3.8	ss134,ss135,ss136
IO.Text_IO.integer.string	14.3.7	ss137,ss431
math.dependent.adx	4.5	ss810,ss807/s
math.dependent.intexp	4.5	ss809,ss806/s
math.dependent.setexp	4.5	ss811,ss808/s
math.function.asin	4.5	ss586
math.function.atan	4.5	ss.34,ss299
math.function.cos	4.5	ss.28,ss295
math.function.exp	4.5	ss.14,ss.31,ss296,ss308/i
math.function.ln	4.5	ss.32,ss297,ss.14/s,ss308/i
math.function.sgn	4.5	ss.35,ss267/i,ss268/i, ss269/i,ss413/i,ss414/i, ss562/i
math.function.sin	4.5	ss.27,ss294
math.function.sqrt	4.5	ss.33,ss298
optimization.algebraic_simplification	10.6	ss.44,ss.47,ss.48,ss.49,ss.50, ss.51,ss.61,ss.62,ss.63,ss.64, ss.65,ss.66,ss.67,ss.73,ss.74 ss218,ss220,ss221,ss319,ss320, ss321,ss322,ss432(ss433), ss433,ss434,ss435,ss436,ss437, ss560(ss561)
optimization.boolean_var_elim	10.6	ss176(ss177)
optimization.bounds_check	10.6	ss174,ss192,ss193,ss194,ss368
optimization.common_sub_expr_elim	10.6	ss.75,ss.76,ss172, ss210(ss211),ss406,ss428, ss508,ss509,ss530,ss533,ss553, ss554,ss643,ss644,common
optimization.data_flow	10.6	ss504,ss505,ss753(ss757), ss754(ss757),ss755(ss757), ss756(ss757)
optimization.dead	10.6	ss.56,ss.68,ss.71,ss225,ss226, ss427,ss638,ss639,ss640,ss641, ss642,ss649,ss650,ss651,dead
optimization.folding	10.6	ss.41,ss.42,ss.55,ss.60, ss.70(ss.69),ss185, ss189(ss190),ss216,ss217, ss219,ss227,ss230,ss231, ss232,ss239,ss285,ss303, ss304,ss305,ss306,

		ss314(ss315),ss318,ss325, ss362,ss421,ss532,ss537,ss538, ss539,ss540,ss556, ss558(ss559),ss561x,ss563, ss564,ss565,ss587(591..4), ss588(591..4),ss589(591..4), ss590(591..4),ss595,ss806, ss807,ss808,fold,ss..2/s, ss..8/s,ss.54/s,ss.83/s, ss591/s,ss594/i
optimization.inline	10.6	ss260,ss410(ss411)
optimization.jump_tracing	10.6	ss182,ss183,ss184,ss250,ss619, ss620
optimization.loop_flattening	10.6	ss405
optimization.loop_fusion	10.6	ss180(ss181)
optimization.loop_induction	10.6	ss236,ss237,ss409
optimization.loop_interchange	10.6	ss750
optimization.loop_invariant	10.6	ss212,ss222,ss429,ss430,ss536, ss749,ss752,invar
optimization.loop_rotation	10.6	ss385(ss385x),ss386,ss387
optimization.loop_unrolling	10.6	ss238,ss240,ss541, ss542(ss542x),ss105/s
optimization.machine_idiom	10.6	ss.40,ss.43,ss.45,ss.52,ss.59, ss173,ss196,ss197,ss198,ss199, ss200,ss204,ss205(ss206), ss207,ss208,ss214,ss215,ss323, ss385x,ss407,ss408,ss503, ss555,ss611,idioms,ss..7/s, ss.29/s,ss.30/s,ss115/s
optimization.merge_tests	10.6	ss175,ss178(ss179), ss440(ss441)
optimization.order_of_evaluation	10.6	ss413,ss414,ss415,ss416,ss417, ss418,ss545,ss546,ss547,ss548, ss549,ss550,ss551,ss552
optimization.redundant_code	10.6	ss195,ss261,ss376,ss377, ss.93/s
optimization.register_allocation	10.6	ss235,ss262,ss263,ss264,ss265, ss307,ss388,ss412,ss442,ss443, ss507,ss510,ss511,ss512,ss531, ss534,ss557,ss606,ss607,ss608, ss609,ss610,ss612
optimization.strength_reduction	10.6	ss.15,ss.16,ss188, ss213(ss422),ss279,ss291, ss423(ss424),ss425,strength, ss426/s
optimization.test_swapping	10.6	ss438,ss439
optimization.unreachable_code	10.6	ss543,ss751,unreach
package.overhead	8.	ss469,ss470,ss471,ss472,ss473, ss474,ss475,ss476,ss477,ss779, ss780,ss781,ss782,ss783,ss784, ss785,ss786,ss787,ss788
pragma.interface.language.assembly	13.9	ss747

pragma.pack	13.1	ss156,ss157,ss158,ss159, ss160,ss161
pragma.suppress.discriminant_check	11.7	ss613,ss614,ss615,ss616,ss617, ss618,ss242/s
pragma.suppress.division_check	11.7	ss444,ss445,ss446,ss447,ss448, ss449,ss450/s,ss451/s
pragma.suppress.index_check	11.7	ss.53,ss.54
pragma.suppress.overflow_check	11.7	ss450,ss451,ss444/s,ss445/s, ss446/s,ss447/s,ss448/s, ss449/s
pragma.suppress.range_check	11.7	ss117,ss168,ss169,ss170,ss171, ss363,ss364,ss365,ss366,ss367, ss372,ss373,ss374,ss375,ss757, ss242/s,ss252/s,ss254/s, ss255/s,ss758/s,ss759/s, ss760/s,ss761/s,ss762/s, ss763/s
record.aggregates	4.3.1	ss116
record.assignment	3.7.4	ss100,ss114
record.component.assignment	3.7	ss.21,ss115,ss244,ss156/s, ss157/s,ss158/s,ss159/s, ss160/s,ss161/s,ss215/s, ss724/s,ss725/s,ss736/s, ss737/s,ss738/s,ss407/i
record.discriminants	3.7.1	ss152,ss153,ss242,ss245,ss598, ss599(ss598),ss600(ss601), ss601,ss602,ss603,ss604,ss605
record.operations	3.7.4	ss513,ss514,ss515
record.overhead	3.7.4	ss789,ss790,ss791,ss792,ss793, ss794,ss795,ss796,ss797,ss798,
representation.attributes	13.7.2	ss730,ss731,ss732,ss734,ss735, ss736,ss737,ss738,ss739,ss740
representation.pack.unpack	13.1	ss652,ss653,ss654,ss655,ss656, ss657,ss658,ss659,ss660,ss661, ss662,ss663,ss664,ss665,ss666, ss667,ss668,ss669,ss670,ss671, ss672,ss673,ss674,ss675,ss676, ss677,ss678,ss679,ss680,ss681, ss687,ss688,ss689,ss690,ss691, ss692,ss693,ss694,ss695,ss696, ss697,ss698,ss699,ss700,ss701, ss702,ss703,ss704,ss705,ss706, ss707,ss708,ss709,ss710,ss711, ss712,ss713,ss714,ss715,ss716, ss724,ss725
scope.intermediate	8.3	ss.96,ss.97,ss.98
scope.local	8.2	ss.20,ss.95
statement.abs	4.5.6	ss.29,ss.30,ss266,ss293, ss368/s,ss431/i
statement.attributes	4.1	ss246
statement.block.overhead	5.6	ss.22,ss.23,ss.24,ss.25,ss544

statement.case	5.4	ss118,ss119,ss133/s,ss325/s, ss482/i,ss488/i
statement.catenation	4.5.3	ss113
statement.control.exit	5.7	ss354,ss355,ss356,ss357, ss182/s,ss183/s,ss184/s, ss250/s,ss376/s,ss377/s, ss386/s,ss612/s, ss406/i,ss427/i
statement.control.for	5.5	ss.58,ss104,ss105,ss181,ss422, ss424,ss516,ss517,ss518,ss519, ss520,ss535,ss542x,ss749, ss.57/s,ss.80/s,ss.81/s, ss106/s,ss171/s,ss180/s, ss225/s,ss236/s,ss237/s, ss238/s,ss239/s,ss240/s, ss387/s,ss409/s,ss423/s, ss425/s,ss525/s,ss536/s, ss541/s,ss542/s,ss651/s, ss749/s,ss750/s,ss752/s, ss776/s, ss120/i,ss163/i,ss164/i, ss165/i,ss166/i,ss167/i, ss212/i,ss213/i,ss428/i, ss431/i,ss438/i,ss439/i, ss440/i,ss441/i,ss442/i, ss443/i,ss472/i,ss473/i, ss477/i,ss490/i,ss491/i, ss511/i,ss512/i,ss654/i, ss655/i,ss659/i,ss660/i, ss664/i,ss665/i,ss669/i, ss670/i,ss674/i,ss675/i, ss679/i,ss680/i,ss686y/i, ss686x/i,ss689/i,ss690/i, ss694/i,ss695/i,ss699/i, ss700/i,ss704/i,ss705/i, ss709/i,ss710/i,ss714/i, ss715/i,ss741/i
statement.control.while	5.5	ss209,ss426, ss185/s, ss148/i,ss162/i,ss165/i, ss166/i,ss369/i,ss385/i, ss479/i,ss480/i,ss481/i, ss482/i,ss493
statement.exponentiating	4.5	ss191,ss222/s,ss121/s,ss188/s, ss291/s,ss304/s,ss305/s, ss306/s,ss307/s,ss595/s, ss643x/s
statement.goto	5.9	ss.26,ss261/s,ss385/s, ss619/s,ss620/s ss356/i
statement.hand_optimize	4.4	ss.69,ss190

statement.if.coding_style	5.3	ss.82,ss.83,ss.84,ss.85,ss.86, ss.87,ss.88,ss.89,ss.90,ss.91, ss.92,ss.94,ss186,ss187,ss223, ss224,ss490,ss491, ss494(ss495),ss495,ss496, ss497,ss498(ss499)
statement.if.condition	5.3	ss.93,ss129,ss144,ss179,ss206, ss292,ss441,ss559,ss207/s, ss208/s,ss324/s,ss328/s, ss421/s,ss438/s,ss439/s, ss440/s,ss504/s,ss505/s, ss507/s,ss508/s,ss509/s, ss510/s,ss558/s,ss561/s, ss644/s,ss649/s,ss650/s, ss686x/s,ss686y/s,ss751/s, ss800/s,ss801/s,ss802/s, ss128/i,ss132/i,ss144/i, ss176/i,ss177/i,ss178/i, ss205/i,ss214/i,ss227/i, ss228/i,ss229/i,ss230/i, ss231/i,ss232/i,ss262/i, ss263/i,ss264/i,ss311/i, ss312/i,ss313/i,ss319/i, ss320/i,ss321/i,ss322/i, ss339/i,ss355/i,ss356/i, ss385x/i,ss398/i,ss399/i, ss400/i,ss402/i,ss403/i, ss404/i,ss409/i,ss417/i, ss418/i,ss431/i,ss479/i, ss480/i,ss481/i,ss511/i, ss512/i,ss526/i,ss527/i, ss528/i,ss537/i,ss538/i, ss539/i,ss540/i,ss754/i, ss806/i,ss807/i,ss808/i, ss809/i,ss810/i,ss811/i
statement.null	5.1	ss..0,ss106,ss804,ss544/s, ss543/i
statement.overhead	5.2	ss634,ss635,ss636,ss637
statement.parentheseis	4.5	ss389,ss390,ss391,ss392,ss393, ss394,ss395,ss396
storage.reclamation	4.8	ss162,ss163,ss164,ss165, ss166,ss167,ss741, reclaim_heap_con, reclaim_heap_unc, reclaim_call_con, reclaim_call_unc
subprogram.external	6.4	ss.36,ss.37,ss.38,ss.39,ss632, ss768(ss769..73), ss769(ss770..73), ss770(ss771..73), ss771(ss772..73),ss772(ss773),

		ss773,ss641/s,ss642/s, ss236/i,ss237/i,ss365/i, ss385/i,ss386/i,ss387/i, ss516/i,ss517/i,ss518/i, ss519/i,ss520/i,ss546/i, ss547/i,ss548/i,ss549/i, ss596/i,ss638/i,ss639/i, ss640/i,ss730/i
subprogram.inline	6.3.2	ss142(ss144),ss411,ss633, ss563/s,ss564/s,ss565/s
subprogram.local	6.4	ss127,ss141,ss143,ss247,ss248, ss249,ss258,ss358,ss359,ss360, ss370,ss483,ss484,ss485,ss521, ss522,ss523,tak, ss260/s,ss596/s,ss748/s, ss236/i,ss237/i,ss379/i, ss380/i,ss381/i,ss382/i, ss383/i,ss384/i,ss486/i, ss487/i,ss492/i,ss598/i, ss599/i,ss600/i,ss601/i, ss603/i,ss604/i,ss605/i
subprogram.nested	8.3	ss361
subprogram.parameters	6.4.1	ss419(420),ss420,ss584,ss585
subprogram.parameters.default	6.4.2	ss124,ss125,ss126
subprogram.parameters.modes	6	ss138,ss139,ss140,ss145, ss146,ss147,ss378,ss562
subprogram.parameters.passing	6.4	ss566,s567,ss568,ss569,ss570, ss571,ss572,ss573,ss574,ss575, ss576,ss577,ss578,ss579,ss580, ss581,ss582,ss583, ss247/s,ss248/s,ss249/s, ss613/s,ss614/s,ss615/s, ss616/s,ss617/s,ss618/s
task.interrupt	13.5.1	INT_0,INT_1,INT_2,INT_3,INT_4, INT_5,INT_6,INT_7,INT_8,INT_9
task.language_feature_tests	9.	task.1,task.2,task.3,task.4, task.5,task.6,task11,task12, task13,task14,task15,task16, task17,task18,task19,task20, task21,task22,task23,task24, task26,task27,task28,task29, task30,task31,task32,task33, task34,task34_delta,task35, task35_delta,task36,task37a, task37b,task38,task39,task40, task41,task42,task43,task44a, task44b,task45a,task45b, task46,task46x,task47,task48, task49,task50,task51,task52, task53,task57,task58,task59, task60,ss740/s

task.rendezvous	9.5	task_num_.1,task_num_.5, task_num_10,task_num_15, task_num_20,task_num_25, task_num_30,task2_num_.1, task2_num_.5,task2_num_10, task2_num_15,task2_num_20, task2_num_25,task2_num_30
task_storage_size	9.9	task54,task55,task56
timing.calendar	9.6	ss453,ss454,ss456,ss457,ss799, ss800,ss801,ss802,ss803
timing.clock	9.6	ss452
type.character.operations ss479,ss480,ss481,ss482,ss493,	3.5.5	ss486/i,ss487/i,ss488/i, ss489/i,ss492/i
type.enumeration.attributes	3.5.5	ss128(129),ss130,ss131,ss251, ss252,ss253,ss254,ss255
type.enumeration.operations	3.5.5	ss132,ss133,ss309,ss310
type.erroneous.program	1.6	ss686
type.named_number	3.2	ss267(ss268..ss269),ss726, ss727,ss728, ss483/s,ss484/s,ss587/s, ss529/i,ss530/i,ss531/i, ss534/i
type.string.assignment	3.6.3	ss.99,ss111,ss112,ss151,ss243, ss371,ss113/s,ss149/s,ss151/s, ss370/s

```
Received: from VMA.CC.CMU.EDU by SMSVMA.BITNET (Mailer R2.05) with BSMTP id
 5331; Fri, 15 Dec 89 23:08:07 CST
Received: from CMUCCVMA by VMA.CC.CMU.EDU (Mailer R2.04) with BSMTP id 9677;
 Fri, 15 Dec 89 23:58:58 EST
Received: from ajpo.sei.cmu.edu by vma.cc.cmu.edu (IBM VM SMTP R1.2.1) with TCP;
 Fri, 15 Dec 89 23:58:55 EST
Received: by ajpo.sei.cmu.edu (5.54/2.2)
        id AA03027; Fri, 15 Dec 89 23:52:02 EST
Message-Id: <8912160452.AA03027@ajpo.sei.cmu.edu>
Received: from umrvmb.umr.edu by RELAY.CS.NET id aa14410; 15 Dec 89 21:16 EST
Received: from SMSVMA.BITNET by UMRVMB.UMR.EDU (IBM VM SMTP R1.2.1MX) with BSMTP
 id 0927; Fri, 15 Dec 89 21:16:32 CST
Received: by SMSVMA (Mailer R2.05) id 5041; Fri, 15 Dec 89 21:15:52 CST
Date:         Fri, 15 Dec 89 21:14:38 CST
From: Burlakoff <MIB413F%SMSVMA.BITNET@UMRVMB.UMR.EDU>
Subject:      Keyword Index Update
To: ACECWG <EV-TEAM%AJPO.SEI.CMU.EDU@relay.cs.net>
```

December 15, 1989

ACECWG, Team

Reference your comments and suggestions on the ACEC Keyword Index at the San Diego meeting.

Following are changes that have been implemented as a result of that meeting.

1. Category "classical" has been changed to "classical_benchmark".

2. Category "constant_propagation" was an error and was deleted. Those tests were already listed under "exception.constant.propagaton".

3. Category "expression" has been deleted and those tests have been placed under the "statement" category.

3. Category "generic" has been changed to "generics"

4. Subcategory "bigint" has been changed to "int_32".

5. Category "interface" has been deleted and placed under "pragma.interface".

6. Category "loop" has been deleted. Those tests have been placed under "statement" with the subcategory "control."

7. Category "math_dep" was an error. This has been changed to "math.dependent".

8. Category "parameters" has been deleted and placed as a subcategory under "subprogram".

9. Subcategory "numeric_error" under category "pragma" has been deleted. Those tests have been indexed under two new subcategories: "pragma.suppress.division_check" and "pragma.suppress.overflow_check".

As a results of these changes, the number of major categories have been reduced to 26 as follows:

access
application
array
boolean
classical_benchmark
conversion
delay
exception
fixed
float
generics
integer
IO
math
optimization
package
pragma
record
representation
scope
statement
storage
subprogram
task
timing
type

There were other comments and recommendations made at the meeting which were not implemented. In some cases they were not implemented because I did not fully understand the recommendation. In other cases, the suggestion could result in a fairly major change, and further discussion may be desirable to be certain that the correct implementation was made. I've listed these below (as well as I could from my notes).

1. The Keyword Index should reflect non_timing tests such as: Delay, Folding, Accuracy, Non_parentheses tests, etc.
2. Subcategories of "exception" may be missing.
3. Determine other "Generics" categories such as "packages", etc.
4. Look into the possibility of categories of accessing variables. (For example, is the category "package.overhead" correct)? The category "scope" may fit into this grouping.
5. Review the group of tests under "task.language_feature_tests" to determine whether this indexing could be improved.
6. Consider changing the Keyword Index output format to improve the output to show groups of primary, secondary and incidental tests.

Hard copies of the revised Keyword Index will be mailed to Nelson, Ray and Sam. A net copy will be sent to Gary (upon request). In addition, a tape will be mailed to Boeing. I should have these actions completed by Dec 20th. Upon request, I can send others a copy (preferably via net).

In summary, I'd like to thank the members of the ACECWG for their time and efforts in review of the Keyword Index. I'm certain that their recommendations will result in a better quality final product.

Regards,

Mike Burlakoff......

```
Received: from CMUCCVMA by VMA.CC.CMU.EDU (Mailer R2.04) with BSMTP id 4147;
 Wed, 25 Oct 89 02:07:55 EDT
Received: from ajpo.sei.cmu.edu by VMA.CC.CMU.EDU ; Wed, 25 Oct 89 02:07:51 EDT
Received: by ajpo.sei.cmu.edu (5.54/2.2)
        id AA16828; Wed, 25 Oct 89 02:09:22 EDT
Message-Id: <8910250609.AA16828@ajpo.sei.cmu.edu>
Received: from umrvmb.umr.edu by RELAY.CS.NET id aa28963; 25 Oct 89 1:06 EDT
Received: from SMSVMA.BITNET by UMRVMB.UMR.EDU (IBM VM SMTP R1.2.1MX) with BSMTP
 id 6192; Wed, 25 Oct 89 01:07:02 CDT
Received: by SMSVMA (Mailer X1.25) id 3978; Wed, 25 Oct 89 01:03:40 CDT
Date:         Wed, 25 Oct 89 01:02:10 CDT
From: Burlakoff <MIB413F%SMSVMA.BITNET@UMRVMB.UMR.EDU>
Subject:      Keyword Index Update
To: ACECWG <EV-TEAM%AJPO.SEI.CMU.EDU@relay.cs.net>
```

ACECWG, Team:

Reference your comments and suggestions on the Keyword Index at the Sep meeting.

The major categories (or subjects) for the index have been refined into the below list of 30 items. I hope this is the kind of a high level structure that you discussed (20-30 categories for naming purposes). This list corresponds to the primary purpose of the tests (the highest level taxonomy category) in the revised Keyword Index. (You may wish to withhold comments until you've had an opportunity to review the new version).

access
application
array
boolean
classical
conversion
delay
exception
expression

fixed
float
generic
integer
interface
IO
loop
math
optimization
package
parameters
pragma
record
representation
scope
statement
storage
subprogram
task
timing
type

The SS tests have been re-reviewed. I mostly left the primary purpose of the tests as in the original version (except for the above major category refinements). The major goal in the review was to determine "secondary" and "incidental" purposes of the tests. This was accomplished. The new version of the Keyword Index (with Test Name and LRM Reference) is now much longer (12 hard copy pages). Because of the size, I'll mail a copy to each ACECWG member by Nov. 1st.

Regards,

Mike.......

December 18, 1989

SUBJECT: ACEC Recommendations

ACECWG:

Reference recent discussion by Dan, Nelson, Phil, others? on ACEC recommendations. I'd like to comment on two areas: 1) The single number result and 2) The number of tests in the ACEC.

SINGLE NUMBER RESULT:

The developers of the ACEC attempted to remove "subjectivity" as much as possible, and therefore, used statistical techniques to present results of tests. Only a discussion of the "system factor" result has been given, and not of the other Median statistical results. Recall that Median produces other results such as "outliers", etc. which give the names of tests which are exceptional (very poor or very good). For the "system factor" results, my guess would be that systems which are rated lower, are actually poorer in "overall" performance than systems with a better "single number" rating.

NUMBER OF TESTS:

I believe that we should carefully consider any recommendations for reducing the current size of the ACEC from its current size (1200 or so in Version 2.0) to something in the order of 200. I doubt that the developers of the tests were paid on the basis of how many tests they could develop, and therefore, I feel that "most" of the tests have a purpose or they wouldn't be in the suite. It seems that the possibility exists that the very tests that were removed might be the one(s) that discover a serious error in a Ada system.

Rather than discussing the "size" of the ACEC, I'd prefer to review the coverage, and determine whether there are areas where performance testing "weaknesses" exist. During this review, if in fact "fat" is discovered in some areas, tests could be then be considered for removal.

Mike Burlakoff

November 3, 1989

Following is a summary of IV&V activities at Boeing on my Sep 28, 29 and Oct 19, 20 trips:

I. NEW PERFORMANCE TESTS:
1. The complete set of tests were compiled and executed on the VAX/780 using Tom L's *.COM files.
2. I spent a lot of time in getting all the support files I needed into my separate area (e.g. GLOBAL, INCLUDE,...) before Tom's *.COM files would run. Also, some of the COM files point to specific directories and these must be corrected.
II. DEBUGGER ASSESSOR:
1. A large notebook of documentation exists which describes how to run the scenarios and includes the source listings of the tests which are used for the scenarios. A Template for recording results has also been developed.
2. A few of the scenarios were successfully executed. There are a few support files and some setup needed to run the tests.

III. LIBRARY ASSESSOR:
1. Nineteen separate scenarios have been developed. Fourteen of these were executed usng Tom L's *.COM files. The other 5 scenarios are such that may consume large amounts of time, memory and disk space and were not attempted. Once all the files were copied into a separate area, there was no setup needed and all the 14 scenarios executed successfully.

2. A number of special programs have been developed for the scenarios. A template for recording results is also available. The documentation that I saw was directly in the *.COM files. A few comments and suggestions were given to Tom L.

IV. DIAGNOSTICS ASSESSOR:

1. The Diagnostic Assessor consists of a large number of programs with known source errors. A file (DIAGCMPR) documents the expected errors, etc. that should occur for each of the programs and allows the user to compare their output with the expected results. A Template is available for recording the results.

2. A sample of the Assessor was successfully executed. Once the correct files were copied into a separate area, there was very little setup.

3. Documentation describing the use of the system was still being developed.

V. SINGLE SYSTEM ANALYSIS SYSTEM:

1. There are 146 tables which contain example output of sets of related tests. Some of these have been manually generated and many have been generated in an automated manner. Some of the data in the tables is actual data, while much of it is sample data. Some of the tables are inconsistent and incomplete. The SSA documentation was also incomplete. The "real" software to generate these tables has not yet been completed. This is the software that would read ACEC output log files and generate actual result tables.

2. A high level review of the tables and much of the available documentation was completed. Several comments and recommendations were given to Kermit.

Mike Burlakoff

Mike Burlakoff
Rt 2, Box 324
Springfield, Mo. 65802
(417)-865-5422
October 25, 1989

SUBJECT: IV&V Testing at Boeing, October 19, 20, 1989.

TO: WRDC/AAAF-3
Mr. R. Szymanski
WPAFB, Oh 45433-6523

Ray:

Following is a summary of my activities at Boeing on Oct. 19, 20, 89. I'll also make a few comments regarding the updated Keyword Index.

1. A tape with the updated Keyword Index was delivered to Boeing on Oct. 19. The tape contains a file sorted by Keyword Index and also files sorted by Test Name and LRM.

2. A review was made of the Single System Analysis (SSA) documentation and tables. There have been approximately 146 tables generated for the SSA. These tables are examples (some contain actual data) of the output that can be expected from the SSA when the SSA software is completed. The tables and documentation are incomplete at this point. Also, as previously noted, the SSA software to read the log file, etc. and generate the "real" tables has not been completed. Several comments and recommendations on the documentation and tables were written and given to Kermit Terrell.

3. The Library Assessor system was reviewed. There are 19 scenarios developed. A template for recording results has also been developed. The majority of the scenarios (14) were executed on Boeing's VAX System. Except for possible weaknesses in the VAX Ada Library system, all of the scenarios executed successfully. A few comments and recommendations on the Library Assessor were written and given to Tom Leavitt.

As usual, I received helpful assistance from all of the Boeing staff. (Kermit for the SSA and Tom for the Library Assessor). Sam provided any other support that I needed.

Now for an update on the Keyword Index activities.

At the Sep E&V meeting, the ACECWG requested that the tests and the index be re-reviewed for:

a) Determining 20-30 categories that could be used for naming purposes and,

b) In addition to the Primary purpose of the tests (as was done in the initial version), determine any secondary and incidental purposes.

The re-review was completed on Oct. 15th. The tape delivered to Boeing contains the results of the above review. The Keyword Index information is much longer now (in my format) and requires about 12 pages in hard copy. As noted to my message to the Team/ACECWG on this date, I'll send the ACECWG members (info to you) a hard copy.

Regards,

Mike Burlakoff

Copy to: Sam Ashby

C O P Y

20 OCT 89

SUBJECT: Library Assessor Comments (IV&V at Boeing)

Following is a copy of the comments given to Boeing on this date.

1. Some of the assessor outputs are difficult to relate to the template. Consider reviewing the *.COM files and add additional comments just

prior to the expected output. (This was especially true for LIB02_4 through LIB02_17). A comment block just prior to the expected output may be useful. For example:

```
! ************* ... ******* ...
! LIB02_N  Unit not present ...
! ...
! ************* ... ******* ...
```

2. Consider moving LIB02_2 and LIB02_3 to LIB14. (Or note on the template that they are accomplished at that time).

3. In general, excellent comments are provided in the *.COM files to guide the user through the scenarios. As other systems are being tested, review of the comments should be made, and additional helpful comments added.

4. Consider summarizing all of the *.COM file comments into one document (A Library Assessor User's Guide). Otherwise, users of other systems may complain that the tests are VAX/VMS Ada oriented.

Mike Burlakoff

C O P Y

19 OCT 89

Subject: Single System Analysis Comments, (IV&V at Boeing)

Following is a copy of the comments given to Boeing on this date.

1. All of the present documentation needs to be reviewed and corrected as necessary. Many of the cross references and file names are incorrect and incomplete. See for example: OPT_FOLD.SSA.

2. All of the sub-groups need to be reviewed for completeness. That is, are major groups of tests missing? The paired comparisons of optimizations are useful, however, they are largely incomplete. Would most users want a complete report (table) which shows all optimizations related to say, Folding? The present paired comparisons only give a small portion of the complete set.

3. Review the names of all the Tables File Names and re-name all "SS..." names to be descriptive of the purpose of the group.

4. I'm not certain how useful the "Bar Chart" and "Similiar Groups" information will be. Consider using this space for other information. Possibilities include other statistical data from the log file, code size, compilation time, etc.

Mike Burlakoff

```
Received: by SMSVMA (Mailer X1.25) id 9556; Mon, 02 Oct 89 13:17:52 CDT
Date:         Mon, 02 Oct 89 13:15:58 CDT
From:         Burlakoff <MIB413F@SMSVMA>
Subject:      ACEC IVV Testing
To:           Szymanski <SZYMANSK%AJPO.SEI.CMU.EDU@RELAY.CS.NET>
cc:           MIB413F@SMSVMA
```

October 2, 1989

Ray:

Following is a summary of the testing performed at Boeing on September 28, 29, 1989.

1. The set of new performance tests were compiled and executed on the DEC/Ada Vax System. Boeing's .COM files were used. All tests in the given *.com files compiled and executed successfully. One observation that I had was that many of the new tests contained extensive comments describing the background and purpose of the test.

2. A sample of the Debugger Assessor was reviewed and several of the scenarios were successfully executed. The documenation describing the system and procedures for use has been prepared and is quite extensive.

3. A sample of the Diagnostics Assessor was reviewed and successfully executed. Documentation describing the system and its use is in the process of being developed.

I received helpful assistance from all of the Boeing staff. (Tom, Kermit for the new tests; Barbara Lindsey for the Debugger and Tom Lee for the Diagnostics Assessor).

Another trip is scheduled for October 19, 20. Planned activities will be to review the Single System Analysis system, and additional review and testing of the other Version 2 deliveries.

Regards

Mike Burlakoff

Copy to: Sam Ashby

```
Received: from CMUCCVMA by SMSVMA.BITNET (Mailer X1.25) with BSMTP id 5435;
 Fri, 13 Oct 89 14:44:26 CDT
Received: from CMUCCVMA by CMUCCVMA (Mailer X1.25) with BSMTP id 0894; Fri, 13
 Oct 89 15:38:25 EDT
Received: from ajpo.sei.cmu.edu by VMA.CC.CMU.EDU ; Fri, 13 Oct 89 15:38:19 EDT
Received: by ajpo.sei.cmu.edu (5.54/2.2)
        id AA04351; Fri, 13 Oct 89 15:36:28 EDT
Message-Id: <8910131936.AA04351@ajpo.sei.cmu.edu>
Received: from umrvmb.umr.edu by RELAY.CS.NET id aa28714; 13 Oct 89 14:33 EDT
Received: from SMSVMA.BITNET by UMRVMB.UMR.EDU (IBM VM SMTP R1.2.1MX) with BSMTP
 id 5110; Fri, 13 Oct 89 14:33:45 CDT
Received: by SMSVMA (Mailer X1.25) id 5314; Fri, 13 Oct 89 14:25:18 CDT
Date:         Fri, 13 Oct 89 14:23:24 CDT
From: Burlakoff <MIB413F%SMSVMA.BITNET@UMRVMB.UMR.EDU>
Subject:      ACEC Test Plan/Procedures Comments
To: EV-TEAM%AJPO.SEI.CMU.EDU@relay.cs.net
```

Following are my comments to the ACEC Test Plan and Test Procedures documents, both dated September 27, 1989.

Test Plan Document:

P.6. The first para includes the Assessors as part of the Opertional Software, but the third para states that the test suite is a set of performance tests. It should be stated that the bullets only apply to the performance portion of the test suite.

P12, para 9.1.1.5. It should be added that all variations of FORMAT produced anciliary data will be tested. As stated, it appears that only one set of MED_DATA_CONSTRUCTOR output will be tested.

P14-16. The plan only discusses testing relating to the performance tests and the support software. It needs to be updated to describe the testing plan for the assessors.

P15. para 9.2.2.1. Will all assessors be tested on all these hosts/ targets?

SUMMARY: The document primarily discusses testing of the performance tests and support software. Additional sections need to be added to describe the plan for testing each of the assessors.

Test Procedures Document:

P5. One bullet should be added to summarize the procedures for testing the assessors.

P7. Sec 4. A separate section should be added to discuss the procedures for testing the assessors. For example, the following should be included:

1. Test programs and documentation needed for the testing.
2. Set up procedures (compile GLOBAL, etc).
3. Execution procedures.
4. Method(s) of logging the results.
5. Procedures for analyzing the results.

P8, para 4.1.1. The Sperry 1631 is not included in this list. Also, will all assessors be tested on all of these systems? If so, the intro para should include assessors.

P8-9, Sec 4.1. Is a separate para needed to describe the method of performing correctness analysis for each assessor?

SUMMARY: As stated in the above comments for the Test Plan, this document also needs to be updated for testing of the assessors. To my knowledge, Boeing has (and will) expend a great deal of time and effort in the testing process. This should be reflected in these documents.

Mike Burlakoff

13 JUN 89

Comments - ACEC Support S/W, SPS, 5 Jun 89

P14,15, 3.1.2.2.2. INCLUDE PDL: Should mention that "Code Expansion Size" computation is initiated in the INCLUDE program and then "included" in STOPTIME2. (or mentioned on pages 35-40, 3.3.1.1.2)

P19. Parameters: in_file,out_file, inline, inlast are not described on P17,18.

P27-31, 3.1.6.3. SSA outputs should have some textual description (or references) (can't understand).

P38&42. A number of CONSTANT'S are in the source code for FORMAT and INCLUDE but are not listed here.

P42, 3.3.2.4.2. It should be stated that the time is "minimum" time.

P45, 3.3.3.1.2. Section ?? should be 3.3.3.1.1 (P44).

P45, 3.3.1.2. Proc 'Get_Time_Stamp' is missing from the chart.

P47,48. Variable names ending in '_time' are misleading. These could also be space size or compile time.

P60-62, 3.3.4.4.1. A number of CONSTANT's in the source are missing here. The same is true of the list of TYPE's. The variable list seems OK. Locals in procs are not included here.

P64,65. The chart on P.65 shows Proc 'large_enough' but no textual description is given on P.64. Also, no PDL is given for this proc.

P67. No info is given on the design of the statistical techniques used in proc 'do_comparison'. This is a major output of the report.

P64-72. Should a discussion be included on the changes needed to add new capabilities to the SSA? For example: Add to Method Type, Table_Array Type, etc.

Mike Burlakoff

16 JUN 89

Comments to ACEC Operational Software SPS, 5 JUN 89

General Comment: The document is not written at the design level but rather mostly includes material from the Users and Readers Guides and Working Papers and the Software Requirements Spec, dated April 17, 1989. Those documents have previously been reviewed and written comments were completed. It should be noted that the requirements for the ACEC additions (I/O, Debugger, Diagnostics, Library, etc) have been described in more detail. However, as previously stated, design information is generally not given.

Specific Comments:

P18. Last 3 '-'. The description of new capabilities are described as 'assessment procedure'. Would 'assessment capability' be more correct?

P42. 3.2.1.4.6.2.2. Why should selecting a carriage return keep the cursor in the same position?

P60. 3.3.1.17. Tracing. Would another useful test be to check whether selective (e.g. every nth value) and conditional tracing are permitted? This would prevent 10MBytes of traced output.
P95. 2nd and 3rd bullets. Do these require deleting the Subunit (are not independent and should not be bullets).

P99. Bullets 2, 4, and 6 should be removed. As in the previous comment, they imply an independent action.

P106,107. Figures 3 and 4 are outdated (do not include Med_Data_Constructor).

P119. 2nd para. Typo: 'symbolic debugger' should be 'program library system'.

Mike Burlakoff.......

```
Received: from CMUCCVMA by SMSVMA.BITNET (Mailer X1.25) with BSMTP id 8214;
 Mon, 24 Apr 89 20:54:54 CDT
Received: from CMUCCVMA by CMUCCVMA (Mailer X1.25) with BSMTP id 6986; Mon, 24
 Apr 89 21:53:13 EDT
Received: from ajpo.sei.cmu.edu by VMA.CC.CMU.EDU , Mon, 24 Apr 89 21:53:09 EDT
Received: by ajpo.sei.cmu.edu (5.54/2.2)
        id AA03781; Mon, 24 Apr 89 21:42:16 EST
Message-Id: <8904250242.AA03781@ajpo.sei.cmu.edu>
Received: from umrvmb.umr.edu by RELAY.CS.NET id aa01062; 24 Apr 89 21:07 EDT
Received: from SMSVMA.BITNET by UMRVMB.UMR.EDU (IBM VM SMTP R1.2) with BSMTP id
 5531; Mon, 24 Apr 89 18:40:56 CDT
Received: by SMSVMA (Mailer X1.25) id 7769; Mon, 24 Apr 89 18:38:56 CDT
Date:         Mon, 24 Apr 89 18:38:26 CDT
From: Burlakoff <MIB413F%SMSVMA.BITNET@UMRVMB.UMR.EDU>
Subject:      ACEC Support S/W SRS Comments
To: ACECWG <EV-TEAM%AJPO.SEI.CMU.EDU@relay.cs.net>
```

ACECWG, Team:

Following are my comments to the Support Software, Software Requirements Specification, dated 17 April 1989. Comments which were previously made on Working Papers on these topics are not duplicated herein.

P7, para 1.2., subpara 2. FORMAT. The sentence may be clearer if "for the SINGLE SYSTEM ANALYSIS program" were in parentheses.

P12, para 3.4.2, 1st para, last 2 lines. Can the user still optionally manually prepare FORMAT output (construct MED_DATA) for direct input to MEDIAN without using the MED_DATA_CONSTRUCTOR program?

P12, para 3.4.2, 3rd para, 2nd line. Typo: should be "systems".

P12, para 3.4.2.1. 1st sentence. I could not find any details in the Operational Software SRS regarding the file generated by the execution of the Operational Software. The only details I could find were in Sec 4 of the Readers Guide. This should also be corrected in the 1st sentence of para 3.4.2.2.
P13, 1st para. Add a sentence something like: "See Sec 4 of the ACEC Readers Guide for a complete description of the remainder of the fields".

P15, para 3.4.3.1, 2nd para, last line. It should be stated that only the last instance of duplicated test runs are considered (or refer to paras 3.4.3.2,3).

P17, para 3.4.3.3, last para. Typo: "Some of these The... "

P18, 2nd para. Either complete the reference of refer to Sec 2.2.

P21. Task Scheduling bullets. Could information on some of the Tasking implementation options be provided. For example, Select options (LRM 9.7.1), scheduling order of tasks of the same priority (LRM 9.8), etc.

P22, I/O bullets. This could be a large subarea. Would another option be to include these in the proposed set of ACEC I/O tests rather than in SSA?

P20-24, Sec 3.4.5.2. Most of the subareas/topics mentioned here are excellent (additional) ACEC analysis capabilities. Would it be worthwhile to review each of these to determine whether they might apply equally well to the "whole ACEC" rather than just SSA? That is, determine whether it might be useful to modify MEDIAN to output analysis between systems.

P26, para 3.6.5. Will the Users Guide contain specific examples on the use of all of the Support Software?

Mike Burlakoff

```
Received: from CMUCCVMA by SMSVMA.BITNET (Mailer X1.25) with BSMTP id 9283;
 Sat, 01 Apr 89 12:07:48 CST
Received: from CMUCCVMA by CMUCCVMA (Mailer X1.25) with BSMTP id 3187; Sat, 01
 Apr 89 13:07:29 EST
Received: from ajpo.sei.cmu.edu by VMA.CC.CMU.EDU ; Sat, 01 Apr 89 13:07:21 EST
Received: by ajpo.sei.cmu.edu (5.54/2.2)
        id AA18922; Sat, 1 Apr 89 13:01:21 EST
Message-Id: <8904011801.AA18922@ajpo.sei.cmu.edu>
Received: from umrvmb.umr.edu by RELAY.CS.NET id aa27842; 1 Apr 89 9:00 EST
Received: from SMSVMA.BITNET by UMRVMB.UMR.EDU (IBM VM SMTP R1.2) with BSMTP id
 3504; Sat, 01 Apr 89 09:00:53 CDT
Received: by SMSVMA (Mailer X1.25) id 8832; Sat, 01 Apr 89 07:57:38 CST
Date:         Sat, 01 Apr 89 07:55:38 CST
From: Burlakoff <MIB413F%SMSVMA.BITNET@UMRVMB.UMR.EDU>
Subject:      ACEC Single System Anal Working Paper Comments
To: ACECWG <EV-TEAM%AJPO.SEI.CMU.EDU@relay.cs.net>
```

ACECWG, Team:

Following are comments to Boeing's ACEC Working Paper titled: Single System Analysis:

1) Sec 1, Para 2. A basic capability of the tool should be to report timing/space results such as is presently being produced using the FORMAT/MEDIAN tools. If I were executing the ACEC on my system, my primary are of interest would be timing/space results of the individual tests (and of course results of tests which failed to compile/execute). I'd also like totals/summaries of these.

2) I must have had a misconception on the purpose of this tool. Somehow, I thought that the primary purpose was to output the results that are presently being done by FORMAT/MEDIAN (where it is appropriate to do so, such as time/space results). This would save the user the effort that is presently needed to set up outputs of other systems, even when comparisons are not desired. However, the capabilities that are being proposed in the paper

(Optimizations, Code Expansion Size, Compiler and Linker Issues, Run Time Issues and Language Features) seem to apply equally to ACEC testing/analysis regardless of whether the ACEC is being used for comparison with other systems of executed on a single system. In this regard, the comments that follow, are viewed as "whole" ACEC changes (in cases where the capability is not presently being performed).

3) The idea of grouping the ACEC into sets of related tests in a formal manner should provide a number of useful analysis areas for the users of the system. Could MEDIAN be enhanced to perform much of the analysis for comparison of performance of these sets between systems?

4) Sec 4, last para. Would capacity limits be more appropriate in the Diagnostic set of tests. (See p. 14,15 of the Diagnostic paper).

5) Sec 5, last bullet. Don't understand this? (Consistency in ...).

6) Sec 6. Note that comparison with other systems is recommended here. As mentioned in para 2) above, it just seems as though all of the recommendations apply regardless of the ACEC mode.

Mike Burlakoff

```
Received: from CMUCCVMA by SMSVMA.BITNET (Mailer X1.25) with BSMTP id 5077;
 Fri, 31 Mar 89 00:43:10 CST
Received: from CMUCCVMA by CMUCCVMA (Mailer X1.25) with BSMTP id 6916; Fri, 31
 Mar 89 01:42:51 EST
Received: from ajpo.sei.cmu.edu by VMA.CC.CMU.EDU ; Fri, 31 Mar 89 01:42:46 EST
Received: by ajpo.sei.cmu.edu (5.54/2.2)
        id AA26060; Fri, 31 Mar 89 01:37:22 EST
Message-Id: <8903310637.AA26060@ajpo.sei.cmu.edu>
Received: from umrvmb.umr.edu by RELAY.CS.NET id aa04630; 30 Mar 89 21:00 EST
Received: from SMSVMA.BITNET by UMRVMB.UMR.EDU (IBM VM SMTP R1.2) with BSMTP id
 3409; Thu, 30 Mar 89 21:01:46 CDT
Received: by SMSVMA (Mailer X1.25) id 4380; Thu, 30 Mar 89 19:58:35 CST
Date:         Thu, 30 Mar 89 19:56:59 CST
From: Burlakoff <MIB413F%SMSVMA.BITNET@UMRVMB.UMR.EDU>
Subject:      ACEC Database Working Paper Comments
To: ACFCWG <EV-TEAM%AJPO.SEI.CMU.EDU@relay.cs.net>
```

ACECWG, Team:

Following are comments on Boeing's Phase 3 Database Test Problems Working Paper 25, 16 September 1988.

1) Is there a possibility of including in the ACEC an actual DBMS that is written in Ada? If a portable system could be found (and permission to use it is granted), it may be that it could be a aid in determining efficiency of

serveral file I/O areas. The Ada IC Newsletter implies that several of these types of systems are under development. I realize that this is easier said than done: Is the system truly portable, is the implementation good, is it proprietary...? (But hopefully it will be worth the time to investigate this possibility).

2) Since these applications would likely execute on large, multi-user systems, we should also emphasize that the tests should be run during several typical workload periods. (This is briefly mentioned on p. 14). Median could then process the results from the different times and produce comparison evaluations.

3) P.1, 4th para. It should also be noted that in addition to the I/O hardware that the system software device drivers, I/O interrupt routines, I/O processing routines, scheduling algorithms (the operating system implementation) would all have an effect on the efficiencies.

4) P.4, 4th para. What is meant by B-tree Support Package? Is more than basic tree traversal intended?

5) P.10, Sec 4. Would a consideration be to use several of the well known basic sort algorithms and compare them for efficiency. Hopefully, the underlying I/O software may use different access methods for some of these.

6) P13, Sec 6. I'd agree with the recommendation. In particular, some of the operations needed for additional I/O pattern tests may directly (or closely) apply to B-Tree, Partial Match, and Sort. For example, operations such as: File creation/deletion, read/write the same record repetitively, read/write in sequential or random order, and variations of insertions and deletions could be similiar in those areas.

Mike Burlakoff

```
Received: from CMUCCVMA by SMSVMA.BITNET (Mailer X1.25) with BSMTP id 6126;
 Tue, 28 Mar 89 17:40:53 CST
Received: from CMUCCVMA by CMUCCVMA (Mailer X1.25) with BSMTP id 3209; Tue, 28
 Mar 89 17:58:14 EST
Received: from ajpo.sei.cmu.edu by VMA.CC.CMU.EDU ; Tue, 28 Mar 89 17:58:07 EST
Received: by ajpo.sei.cmu.edu (5.54/2.2)
        id AA04977; Tue, 28 Mar 89 17:50:40 EST
Message-Id: <8903282250.AA04977@ajpo.sei.cmu.edu>
Received: from umrvmb.umr.edu by RELAY.CS.NET id aa17416; 28 Mar 89 16:57 EST
Received: from SMSVMA.BITNET by UMRVMB.UMR.EDU (IBM VM SMTP R1.2) with BSMTP id
 3251; Tue, 28 Mar 89 16:58:02 CDT
Received: by SMSVMA (Mailer X1.25) id 5810; Tue, 28 Mar 89 15:54:56 CST
Date:         Tue, 28 Mar 89 15:53:34 CST
From: Burlakoff <MIB413F%SMSVMA.BITNET@UMRVMB.UMR.EDU>
Subject:      ACEC Library Robustness Paper Comments
To: ACECWG <EV-TEAM%AJPO.SEI.CMU.EDU@relay.cs.net>
```

ACECWG, Team:

Following are comments to Boeing's ACEC Working Paper on Library Robustness:

1) P.1, Sec 2. Other possible areas of evaluation could be: Completeness of capabilities, useablity and user documentation. Following initial design requirements, a list of desired standard capabilities could be developed. This list could then be used to assess these kinds of areas.

2) P.1, 2nd bullet, 2nd para. It should be noted that storage access may also depend on the implementation design.

3) P.3, 2nd para. As the development proceeds, it may determined that quite a large number of scenarios are needed for a complete evaluation.

4) P.7, para 11. What is meant by "consistency"?

5) P.8, para 12. To fully test the "version" capability, it may be necessary to construct a small sample with several versions.

6) P.8, para 13. The metric should be clarified: "Is the space from deleted units reused".

7) P.8, para 14. Does this item belong in the Linker test suite?

8) P.10, Sec 4. In addition, it may be desirable to provide a simple "usability" result.

9) For some of the timing measurement on multi-user systems, it may be desirable to submit batch jobs which run at several times during the day (peak, 3 am, etc), to determine the difference in access times.

Summary: The scenarios that are listed seem to be an excellent and varied set and may be complete. However, prior to final selection of the evaluation set, several "popular" vendors librarys should be reviewed to determine if additional scenarios are needed. The final set should then be prioritized.

Mike Burlakoff

```
Received: from CMUCCVMA by SMSVMA.BITNET (Mailer X1.25) with BSMTP id 0201;
 Tue, 21 Mar 89 22:51:43 CST
Received: from CMUCCVMA by CMUCCVMA (Mailer X1.25) with BSMTP id 8803; Tue, 21
 Mar 89 23:48:57 EST
Received: from ajpo.sei.cmu.edu by VMA.CC.CMU.EDU ; Tue, 21 Mar 89 23:48:06 EST
Received: by ajpo.sei.cmu.edu (5.54/2.2)
        id AA08395; Tue, 21 Mar 89 23:43:18 EST
Message-Id: <8903220443.AA08395@ajpo.sei.cmu.edu>
Received: from umrvmb.umr.edu by RELAY.CS.NET id aa15295; 21 Mar 89 23:42 EST
```

```
Received: from SMSVMA.BITNET by UMRVMB.UMR.EDU (IBM VM SMTP R1.2) with BSMTP id
 2942; Tue, 21 Mar 89 22:42:34 CST
Received: by SMSVMA (Mailer X1.25) id 0181; Tue, 21 Mar 89 22:39:42 CST
Date:          Tue, 21 Mar 89 22:37:26 CST
From: Burlakoff <MIB413F%SMSVMA.BITNET@UMRVMB.UMR.EDU>
Subject:       ACEC Diagnostics Working Paper Comments
To: ACECWG <EV-TEAM%AJPO.SEI.CMU.EDU@relay.cs.net>
```

ACECWG, Team:

Following are my comments to Boeing's ACEC Diagnostics Working Paper, along with the extension (Working Paper 23, 9 JUN 88).

1) Early in the paper, a discussion should be given which describes the specific approach for determining the test problems to be included in this suite of tests. There are a large number of potential problems discussed. Many of these seem important and are of the "hard" category for Compilers, Linkers and Run Time Systems. It is apparent that these were chosen based on varied experiences, and thus represent professional opinion of examples of needed diagnostics testing.

It would seem that the first step for this development would be for a team of experienced professionals to review documentation related to these systems. For example, a review of the LRM would probably point the need for several areas that require diagnostics testing. (e.g., semantics, not adequately covered by the ACVC, causing particular user difficulties, etc). For the Linker and Run Time Systems, an approach might be to review the documentation of several "well known" systems to determine whether desired diagnostic capabilities are lacking. Following these reviews, the recommendations should be listed in some priority order. The highest priority tests could then be completed (based on contractual agreements) in the initial phase of the effort.

2) P.1, list of requirements. Recommended additions:

a) The test problems should be chosen to improve APSE disgnostics quality and thus enhance the correctness of the resulting programs and aid programmer productivity.

b) To aid in user evaluation of the test results, each test problem will produce a textual description of the error and the expected diagnostic message.

3) P. 1, requirement 4. Should the term "APSE" be used here?

4) P. 2, para 2. I agree that the time to run the tests should be minimized. Even this will be difficult, since various error conditions on some systems may terminate the job and require re-start of the test suite. However, as stated in the paper, the analysis will largely be a manual process. I question whether the goal should be to complete the analysis in one day. This time will largely depend on the experience (and interest) of the user. As previously mentioned in 2b) above, clear textual information output to the user will be an aid in the analysis process.

5) P. 3,4. It may be the case that it will be necessary to use one or two of the suggested approaches (or combinations) to determine which provides the most useful results. In any case, it would seem that some form of the rating system recommended in the Presser/Benson paper is needed. (particularly for compiler evaluation).

6) P. 6. I don't believe that the paper should limit the scope of the effort at the outset. In a sense, this is a new area of technology in the evaluation process. It would seem to be difficult (at this point) to determine the scope of the effort.

7) P.8.9. Good discussion. It appears that point 4 on P.9 is the best alternative. It will probably be the case that it will not be necessary to run all of the tests through the editor.

8) P.10-28. As a further aid to user evaluation, each test problem should identify the LRM reference and state whether the error violates the LRM, is a poor (dangerous) practice, etc.

9) P.10-17, Excellent test examples. I believe that para 10 on P.17 is a matter of preference. It would seem that it would cause some additional processing time, and some users (myself) might prefer to have the error reported at each occurrence.

10) P.19, 3rd para. "Those that have made the extra effort...". How will the evaluation reward these "goodies".

11) P.21, program segment. Unless I'm missing something, the "K<1" and the accompanying explanation seem in conflict.

12) P.26, para 20. Should this test belong in the Linker section?

13) P.27, para 23. Should this para be combined with para 7?

14) P.31, para 12. Should this belong in the "Library Diagnostics" set.

15) P.32, para 1. This recommended test would seem to be in the "very hard" category. My understanding is that deadlocks are very difficult to detect, and that if a system could detect the deadlock, then it would correct the error and a deadlock would not exist. However, the test, as proposed, seems like a good exercise.

16) P.33-34. Are we proposing here to verify which of the four techniques a Runtime System might use? I don't understand the relationship of the four techniques to the planned diagnostic testing.

17) P.32-37. The excellent examples which are presented are largely limited to diagnostics for access types. This again points to the need for study of needed diagnostics in other areas.

Mike Burlakoff.....

```
Received: from CMUCCVMA by SMSVMA.BITNET (Mailer X1.25) with BSMTP id 4091;
 Sat, 18 Mar 89 11:51:08 CST
Received: from CMUCCVMA by CMUCCVMA (Mailer X1.25) with BSMTP id 4445; Sat, 18
 Mar 89 10:41:40 EST
Received: from ajpo.sei.cmu.edu by VMA.CC.CMU.EDU ; Sat, 18 Mar 89 10:41:32 EST
Received: by ajpo.sei.cmu.edu (5.54/2.2)
        id AA00847; Fri, 17 Mar 89 19:16:20 EST
Message-Id: <8903180016.AA00847@ajpo.sei.cmu.edu>
Received: from umrvmb.umr.edu by RELAY.CS.NET id aa01040; 17 Mar 89 18:51 EST
Received: from SMSVMA.BITNET by UMRVMB.UMR.EDU (IBM VM SMTP R1.2) with BSMTP id
 2768; Fri, 17 Mar 89 17:52:02 CST
Received: by SMSVMA (Mailer X1.25) id 3032; Fri, 17 Mar 89 17:49:22 CST
Date:         Fri, 17 Mar 89 17:47:44 CST
From: Burlakoff <MIB413F%SMSVMA.BITNET@UMRVMB.UMR.EDU>
Subject:      ACEC WP22 (Symbolic Debugger) Comments
To: ACECWG <EV-TEAM%AJPO.SEI.CMU.EDU@relay.cs.net>
```

ACECWG, Team:

Following are comments on ACEC Working Paper 22, 31 May 1988, Symbolic Debugger Evaluation:

1) In contrasting development of a Symbolic Debugger Evaluation Capability with Ada Compiler Evaluation suites such as the ACVC or ACEC, the major difference is that the LRM served as a basis for those developments (the ACVC in particular). Since there is no such standard for Symbolic Debuggers, the evaluation capibility will consist of tests which are believed to represent typical capabilities in Symbolic Debuggers. The Working Paper lists 19 Scenarios which are intended to be non-exhaustive and typical of tests to be included. It appears that these were gathered in no particular order, and represent capabilities that the implementor felt were most important. In this regard, the following recommendation is made:

a) Begin with an "approved" Symbolic Debugging Capabities list and insure that the test suite includes scenarios to evaluate each item on the list. It is understood that no such approved/accepted standard presently exists. The starting point might be the checklist Table 6.5-1 in the E&V Guidebook. This list could be refined during reviews and as the design becomes firm.

2) It is probably the case that most existing Symbolic Debuggers are multi-language, with Ada extensions. (the VAX/VMS for example). This raises the question of whether this capability should be "universal" or Ada only. In reviewing the 19 Scenarios in the Working Paper, I categorized 11 as "Universal" and 8 as "Ada only". Therefore, we could consider designing the evaluation tool in 2 parts: 1) Universal and 2) Ada extensions.

3) In addition to the test suite, the evaluator should include an E&V type of capability to permit evaluation of areas such as: Useability, Performance and Documentation. This could be something as simple as a one page evaluation list which is completed at the end of the execution sequence.

4) The majority of the scenarios seem to be essential, and several will likely challenge the facilities of most debuggers. Some of them (such as infinite loop detection and detection of deadlock) may be over and above the call of duty of typical debuggers.

Mike Burlakoff

```
Received: from VMA.CC.CMU.EDU by SMSVMA.BITNET (Mailer X1.25) with BSMTP id
 6874; Wed, 22 Nov 89 15:53:47 CST
Received: from CMUCCVMA by VMA.CC.CMU.EDU (Mailer R2.04) with BSMTP id 1139;
 Wed, 22 Nov 89 16:52:53 EST
Received: from ajpo.sei.cmu.edu by vma.cc.cmu.edu (IBM VM SMTP R1.2.1) with TCP;
 Wed, 22 Nov 89 16:52:47 EST
Received: by ajpo.sei.cmu.edu (5.54/2.2)
        id AA25133; Wed, 22 Nov 89 16:52:08 EST
Message-Id: <8911222152.AA25133@ajpo.sei.cmu.edu>
Received: from umrvmb.umr.edu by RELAY.CS.NET id aa03312; 22 Nov 89 15:48 EST
Received: from SMSVMA.BITNET by UMRVMB.UMR.EDU (IBM VM SMTP R1.2.1MX) with BSMTP
 id 8096; Wed, 22 Nov 89 15:49:12 CST
Received: by SMSVMA (Mailer X1.25) id 6808; Wed, 22 Nov 89 15:47:23 CST
Date:         Wed, 22 Nov 89 15:46:11 CST
From: Burlakoff <MIB413F%SMSVMA.BITNET@UMRVMB.UMR.EDU>
Subject:      Topics for ACEC V3, and V4
To: ACECWG <EV-TEAM%AJPO.SEI.CMU.EDU@relay.cs.net>
```

Ray, ACECWG, Team:

In reviewing your list, I found it necessary (in many cases) to consider/rate "line items" under your major categories. Also, I've made recommendatons for additions in a few areas.

1. Capacity testing: Grade C.
2. Systematic compile speed testing: Grade C.
3. Additional performance tests: Grade B.
    a. The Cache processing tests could also be useful for analysis on general virtual (paging) systems (even if they don't use cache memories).
    b. ADDED: Performance tests for each math function. This should give the user the execution time of each function, and also include the accuracy of the function (in decimal digits).
    c. ADDED: String applications.
4. Packaging:
    a. Where multiple test problems could preclude running the rest of the problems: Grade A.
    b. Two version discussion: Grade C.
5. Unreliable time measurements. This recommendation implies significant testing to determine whether measurements are actually more reliable: Grade B.

6. Documentation.

a. Analysis guide. Need more info on what is to be contained in the guide.

b. Indexes to the guides: Grade B.

c. Intro to Benchmarking: Grade C.

d. Non-performance discussion: Grade B.

e. ADDED. Much of the discussion of MEDIAN output is written at the level of understanding of a statistician. Additional examples/explanations of this output would aid the common user to easily understand (and make better use of) the MEDIAN results. I believe terms like "system factors, outliers, etc." could be better defined at the layman level.

7. MEDIAN.

a. I don't feel qualified to make recommendations on the first 3 items. (e.g. not familiar with ANOVA, how much overhead would be required to compute confidence intervals...)

b. The Single System Analysis Tool will provide analysis of a number of ACEC problem subsets. It may be that some of these will also be useful for analysis by MEDIAN. It should be possible to request analysis by MEDIAN of any desired ACEC subset: Grade A.

c. User interface. The recommendation that MEDIAN read the *.log files directly is a major change in the ACEC support software area. This implies that MEDIAN treats the raw output as data and does not require any other tools to produce the present results. Minor modifications to the present *.log formats/output will be needed: Grade A.

Check confidence levels between runs: Grade C.

d. Output: Grade B.

8. Continuing support: Grade A.

9. ADDED. Work on "Lessons learned from Versions 1 and 2". Considering the many valuable and useful inputs from individuals and groups (like the ACECWG), and the many resulting changes made for Version 2, I would expect similiar inputs as the ACEC is distributed and used more widely. In that respect, I would somehow list this as a separate (large) category. This is not "maintenance" as such, rather, something like "user requirements."

Mike Burlakoff

15 JUN 89

AN APPROACH TO AN INPUT DATA FILE FOR MEDIAN

There has been some recent discussion regarding ACEC Support Software overhead, and the desirablity of using a data file for processing test results. In this regard, the following is a possible approch.

PROPOSED SEQUENCE:

```
 __________        __________
! System 1 !      !Expanded !
!  Test    ! ->   !   Log   ! ->
! Execution !     !  File   !
!__________!      !_________!    ______________     __________
                                ! Reduced with !   ! Read and !
                                ! Editor and   !   ! Processed !
 __________        __________   ! Concatenated ! -> ! by MEDIAN ! -> Report
! System 2 !      !Expanded !   ! into one     !   !__________!
!  Test    ! ->   !   Log   ! -> ! Data File    !
! Execution !     !  File   !   !______________!
!__________!      !_________!
     .                .
     .                .     ->
     .                .
```

CONCEPT:

1. As tests are executed, the "log-file" would output:
   a) The present information
   b) An additional line (interspersed with the present information), identified with something like "+++" and containing: TIME, TIME_ERROR_CODE, and SPACE.
2. A user searches these log files (editor, etc.) for TEST_NAME, TIME, TIME_ERROR_CODE and SPACE and concatenates all files for all systems into one data file.
3. MEDIAN reads the data file into the arrays which are presently already declared in MED_DATA and produces the statistical report.

REQUIREMENTS/CHANGES:

1. In each test, change the present PUT("TEST_NAME,description...) to PUT("+++TEST_NAME,description...).
2. Each test which can produce an error code (in the test source code) will also set a GLOBAL variable TIME_ERROR_CODE.
3. In STOPTIME2, set TIME_ERROR_CODE for "err_unreliable_time" or for "not confidence_interval_within_tolerance (#)".
4. Prior to return from STOPTIME2, output a new_line with:
   "+++, TIME, TIME_ERROR_CODE, SPACE".
5. Modify MEDIAN to declare the present MED_DATA types and arrays and read the data file into the arrays. (The TIME_ERROR _CODE simply replaces the TIME for that test in MEDIAN). MEDIAN then produces the present report.

Mike Burlakoff

```
Received: from CMUCCVMA by SMSVMA.BITNET (Mailer X1.25) with BSMTP id 9425;
 Mon, 13 Mar 89 23:31:45 CST
Received: from CMUCCVMA by CMUCCVMA (Mailer X1.25) with BSMTP id 8465; Mon, 13
 Mar 89 18:22:40 EST
Received: from ajpo.sei.cmu.edu by VMA.CC.CMU.EDU ; Mon, 13 Mar 89 18:22:32 EST
Received: by ajpo.sei.cmu.edu (5.54/2.2)
        id AA23931; Mon, 13 Mar 89 18:15:20 EST
Message-Id: <8903132315.AA23931@ajpo.sei.cmu.edu>
Received: from umrvmb.umr.edu by RELAY.CS.NET id aa29266; 13 Mar 89 17:40 EST
Received: from SMSVMA.BITNET by UMRVMB.UMR.EDU (IBM VM SMTP R1.2) with BSMTP id
 2423; Mon, 13 Mar 89 16:29:57 CST
Received: by SMSVMA (Mailer X1.25) id 8465; Mon, 13 Mar 89 16:27:25 CST
Date:         Mon, 13 Mar 89 16:25:54 CST
From: Burlakoff <MIB413F%SMSVMA.BITNET@UMRVMB.UMR.EDU>
Subject:      ACECWG/PC Comments
To: Nelson <EV-TEAM%AJPO.SEI.CMU.EDU@relay.cs.net>
```

Nelson:

Following are some recommended additions to the questionnaire. I've preceded each of my added lines with a '*'.

Name, Phone, Organization, Address ...

Have you used the ACEC?

Do you plan to use it?

What compiler(s) are you evaluating?

*What computer system and operating system did you use?

*How long did it take you to install the ACEC on your system? Please discuss any particular problems/recommendations.

*How long did it take you to run the ACEC? Please discuss any particular problems/recommendations.

*Did you run the analysis tools (FORMAT and Median). If yes, did they provide the desired results. Please discuss any particular problems/recommendations and comment on the ease of use/understanding.

*Which ACEC documentation did you use? Was it complete and helpful? Please comment on the ease of use and overall quality of the documentation?

*Did the ACEC provide a desired efficiency analysis of any areas of concern in your system?

*Do you plan to publish the results of the use of the ACEC or of the evaluation of the compiler(s)? If so, where?

*Do you have any suggestions for improvements or extensions of the ACEC?
*Do you have any criticisms of the current ACEC?

*Are you interested in attending a Birds of a Feather session to discuss the ACEC at a SIGAda or AdaJUG meeting?

Please mail to (envelope enclosed):

Mr. Raymond Szymanski
WRDC/AAAF
Wright Patterson AFB, OH 45433-6523
szymansk@ajpo.sei.cmu.edu

Mike Burlakoff

```
Received: from CMUCCVMA by SMSVMA.BITNET (Mailer X1.25) with BSMTP id 5054;
 Sun, 12 Mar 89 18:48:40 CST
Received: from CMUCCVMA by CMUCCVMA (Mailer X1.25) with BSMTP id 2893; Sun, 12
 Mar 89 19:47:16 EST
Received: from ajpo.sei.cmu.edu by VMA.CC.CMU.EDU ; Sun, 12 Mar 89 19:47:08 EST
Received: by ajpo.sei.cmu.edu (5.54/2.2)
        id AA16106; Sun, 12 Mar 89 19:43:44 EST
Message-Id: <8903130043.AA16106@ajpo.sei.cmu.edu>
Received: from umrvmb.umr.edu by RELAY.CS.NET id aa29959; 12 Mar 89 19:45 EST
Received: from SMSVMA.BITNET by UMRVMB.UMR.EDU (IBM VM SMTP R1.2) with BSMTP id
 2325; Sat, 11 Mar 89 16:50:24 CST
Received: by SMSVMA (Mailer X1.25) id 3704; Sat, 11 Mar 89 16:47:57 CST
Date:        Sat, 11 Mar 89 16:47:21 CST
From: Burlakoff <MIB413F%SMSVMA.BITNET@UMRVMB.UMR.EDU>
Subject:     ACEC/PIWG: An E&V'rs Viewpoint
To: EV-TEAM%AJPO.SEI.CMU.EDU@relay.cs.net
```

Sandi, Ray, Team:

Reference the recent discussions on this topic. As a user of the ACEC the past year, I thought I'd offer my 1 1/2 cents worth as follows:

My familiarity with the PIWG, U. of Mich, AES, ASR and Aerospace suites is primarily from reading some material and talking with users of a couple of the test suites. The developers of these other systems (who were primarily volunteers) are to be commended for their splendid products and efforts. However, my conclusion is that none of those are in the same class as the ACEC and therefore do not measure up to the ACEC. I say this because of: 1) extent of coverage, 2) complexity of the issues that are covered, and in 3) overall quality of the ACEC. Following are points which I believe justify these statements.

1) The ACEC is a planned, systematic, formal development requiring person years for the development. The developers are recognized, skilled professionals.

2) The test suite coverage is extensive, and attempts to evaluate many of the more difficult runtime issues (task loading, interrupt processing, design trade-offs, exception handling, file I/O, ...).

3) A great deal of design/development effort has been expended to provide an accurate and usable test environment.

a. The timing code is extensive and provides information on the accuracy that was achieved (or not achieved).

b. The individual test template structure is such that it is easy for the user to construct and insert new tests into the environment.

c. The analysis tools provide excellent statistical comparisons and summaries.

4) The ACEC is an on-going effort that is being reviewed (E&V) and monitored for quality.

5) The documentation is complete, extensive and of high quality. For example, the Readers Guide contains excellent background material on a number of technical issues and trade offs relating to evaluations.

6) The test suite is sponsored and supported by the government.

In view of the above, I believe that the ACEC, (given an unbiased evaluation by users of the system) will become the industry standard, and should be promoted as such.

Mike Burlakoff

```
Received: from CMUCCVMA by SMSVMA.BITNET (Mailer X1.25) with BSMTP id 3234;
 Wed, 08 Mar 89 19:24:03 CST
Received: from CMUCCVMA by CMUCCVMA (Mailer X1.25) with BSMTP id 8785; Wed, 08
 Mar 89 16:52:56 EST
Received: from ajpo.sei.cmu.edu by VMA.CC.CMU.EDU ; Wed, 08 Mar 89 16:52:44 EST
Received: by ajpo.sei.cmu.edu (5.54/2.2)
        id AA20129; Wed, 8 Mar 89 16:47:23 EST
Message-Id: <8903082147.AA20129@ajpo.sei.cmu.edu>
Received: from umrvmb.umr.edu by RELAY.CS.NET id aa15084; 8 Mar 89 14:32 EST
Received: from SMSVMA.BITNET by UMRVMB.UMR.EDU (IBM VM SMTP R1.2) with BSMTP id
 2055; Wed, 08 Mar 89 12:20:37 CST
Received: by SMSVMA (Mailer X1.25) id 1048; Wed, 08 Mar 89 12:15:18 CST
Date:         Wed, 08 Mar 89 12:13:19 CST
From: Burlakoff <MIB413F%SMSVMA.BITNET@UMRVMB.UMR.EDU>
Subject:      CLASSWG - Ref Man Tools Chapter 5
To: Peter <FV-TEAM%AJPO.SEI.CMU.EDU@relay.cs.net>
```

Peter:

Following are my thoughts on a few of the references:

5.1.3 Code Generator (Back End)
Cross References:
Functions:

[Code Generation	7.1.6.7.x]

7.1.6.7.x Code Generation
Description:
Using a Compiler generated translation form such as an intermediate language to produce object code. The object code machine language may be for the host or a target computer system.
Cross References: (Same as 7.1.6.7?)
Guidebook References: HELP! (?Choose those that apply from 7.1.6.7)

5.2.1 Host-Target System Cross-Assembler

Same as 5.1.4 Assembler, except add to the description of 7.1.6.6 the following: "The translation may be to the host machine language or to a target computer systems machine language".

5.2.8 Host-to-Target Downloader

Functions:

[Downloading	7.1.6.15]

(Note: We may wish to discuss whether Linking/Loading belongs under 7.1.6, or whether a new category such as 7.4 (Data Communications/Data Transfer) is needed. For consistency at this time, I'll place Downloading/Uploading in this section).

7.1.6.15 Downloading
Description:
The process of moving programs or data from a host to a target computer system over a data communications medium. The transfer may be to main memory or to secondary storage.
Cross References: (?Same as 7.1.6.13)
Guidebook References: HELP! I suspect that a new checklist should be developed for Downloading/Uploading.

5.2.9 Target-to-Host Uploader
Functions:

[Uploading	7.1.6.16]

7.1.6.16 Uploading
Description:
The process of moving programs or data from target to a host computer system over a data communications medium. The transfer is generally to secondary storage.
Cross References: (?Same as 7.1.6.13)
Guidebook References: ?New Checklist needed.

5.9.2 Size Estimator
Functions:

[Sizing Analysis	7.3.1.30
Resource Utilization	7.3.2.12]

5.11.8 Input and Output Services
Functions:

[Input/Output Support	7.2.3.2]

Note: I don't believe that we should limit the 7.2.3.2 Description to "standard I/O devices". Most Embedded systems access non-standard I/O devices (sensors, etc). Also, because this is such a large support area, I'd consider expanding 7.2.3.2 to much more detail.

5.11.9 Performance Monitor
Functions:

[Timing	7.3.2.14
Tuning	7.3.2.15]

Mike

```
Received: from CMUCCVMA by SMSVMA.BITNET (Mailer X1.25) with BSMTP id 2659;
 Wed, 23 Aug 89 22:07:04 CDT
Received: from CMUCCVMA by CMUCCVMA (Mailer X1.25) with BSMTP id 8966; Wed, 23
 Aug 89 23:08:25 EDT
Received: from ajpo.sei.cmu.edu by VMA.CC.CMU.EDU ; Wed, 23 Aug 89 23:08:20 EDT
Received: by ajpo.sei.cmu.edu (5.54/2.2)
        id AA11313; Wed, 23 Aug 89 23:06:48 EDT
Message-Id: <8908240306.AA11313@ajpo.sei.cmu.edu>
Received: from umrvmb.umr.edu by RELAY.CS.NET id aa21822; 23 Aug 89 23:03 EDT
Received: from SMSVMA.BITNET by UMRVMB.UMR.EDU (IBM VM SMTP R1.2.1MX) with BSMTP
 id 1625; Wed, 23 Aug 89 22:02:59 CDT
Received: by SMSVMA (Mailer X1.25) id 2657; Wed, 23 Aug 89 22:01:02 CDT
Date:         Wed, 23 Aug 89 22:00:13 CDT
From: Burlakoff <MIB413F%SMSVMA.BITNET@UMRVMB.UMR.EDU>
Subject:      PTS Procedures Comments
To: Stripe <STRIPET%JALCF.WPAFB.AF.MIL@relay.cs.net>
Cc: EV-TEAM%AJPO.SEI.CMU.EDU@relay.cs.net
```

August 23, 1989

Following are my comments to the Performance Testing Service (PTS), 14 Aug 89, Preliminary Draft.

Pii, 1st para, last sentence. I can forsee cases whereby the primary purpose of the PTS is to evaluate (say a new) compiler's performance. (Not in conjunction with application needs, or to supplement any other process).

P1, 1.2, 1st sentence. Should "... the AJPO has recognized the need for information ... " be changed to "... the AJPO has recognized the need for evaluation ... ". This might emphasize that we are doing more than just providing information.

P2, 5th line. Shouldn't there be some sort of "grading" system? What is the purpose of all this if all compilers which are tested equally pass?

P2, 1.3, 1st sentence. It would seem that the primary objective of the PTS is to provide the means to evaluate compiler quality and then provide the results of the evaluation (rating summary, etc).

P7, 3.1. Should another paragraph be added to state that the AJPO issues a PTS certification number? (see p15, 5.5).

P9, 4.3. Although it is probably understood, I'd add to the assessment that the software must be well documented.

P10, 4.5. I'd strongly recommend that the customer NOT determine customization. This leaves the testing open to possible unethical manipulation. To my knowledge the developer of the ACEC went to great lengths to make the test suite portable. The only customization that I know of is in the math library. I would state that no test suite modifications are permitted without prior AJPO/PTF approval. Let all tests be run against all systems. The results should show any tests which should not apply to this compiler system.

P12, 5.1. Should the agreement show any proposed implementation dependent information which might affect the results?

P15, 5.5. Will AJPO issue a PTS certification number if 25% of the tests did not execute?

Major Comment. The document sidesteps all "pass/fail" and "grading" issues. I believe that there must be a certain point where if a compiler system fails a portion of the test system in terms of performance, usability, etc., that system would NOT be certified (even though testing was performed by a PTF). The document needs to specify rules and procedures in these cases.

Mike Burlakoff........

```
Received: from CMUCCVMA by SMSVMA.BITNET (Mailer X1.25) with BSMTP id 9293;
 Sat, 01 Apr 89 12:12:16 CST
Received: from CMUCCVMA by CMUCCVMA (Mailer X1.25) with BSMTP id 3197; Sat, 01
 Apr 89 13:10:36 EST
Received: from ajpo.sei.cmu.edu by VMA.CC.CMU.EDU ; Sat, 01 Apr 89 13:10:27 EST
Received: by ajpo.sei.cmu.edu (5.54/2.2)
        id AA18929; Sat, 1 Apr 89 13:02:06 EST
Message-Id: <8904011802.AA18929@ajpo.sei.cmu.edu>
```

```
Received: from umrvmb.umr.edu by RELAY.CS.NET id aa27887; 1 Apr 89 9:03 EST
Received: from SMSVMA.BITNET by UMRVMB.UMR.EDU (IBM VM SMTP R1.2) with BSMTP id
 3505; Sat, 01 Apr 89 09:04:22 CDT
Received: by SMSVMA (Mailer X1.25) id 8837; Sat, 01 Apr 89 08:01:06 CST
Date:         Sat, 01 Apr 89 07:58:48 CST
From: Burlakoff <MIB413F%SMSVMA.BITNET@UMRVMB.UMR.EDU>
Subject:      Ada Compiler Eval P&G Comments
To: EV-TEAM%AJPO.SEI.CMU.EDU@relay.cs.net
```

Team:

Following are my comments to the Ada Compiler Evaluation Procedures and Guidelines document, Version 1.0, dated 1 MAR 89. ("l" refers to line no.):

l5. The sentence seems incomplete. Would "performance of Ada compilation systems" be better?

l141. Typo. 5.1.11 should be "Prepare the Performance Summary".

l245-251. I don't agree. Formal evaluation results do imply the usability of Ada for particular applications. They do not warrant, but they do imply.

l308-309. Does the EAR include the Performance Summary? If so, then it should be stated.

l330. Would the term "procedures" rather than "process" be more correct?

l390-488. Many of the responsibilities of the AEO, ADMO and AEF conflict and are duplicated. For example, the AEF's and the AEO both review disputes and evaluate test plans. Would the AEO be qualified to participate in deciding on the withdrawal of test programs? The intent of the organizational structure seems to be three areas of responsibility: 1) Management/Admin, 2) ACEC Test Suite Maintenance and Distribution and 3) Evaluation Performance (as presently defined on lines 448-466). This section and also lines 654-850 should be reviewed and modified as necessary to insure that an efficient organization is proposed.

l510. "contains" may be a better term than "includes". (To prevent confusion with the INCLUDE process).

l511. "object code size" may be misleading. Object code could include header information, link item types, relocation info, etc. The ACEC measures execution size of the program in terms of machine language instructions and data segments. This is also mentioned on lines 515 and 838.

l511-523. It might be mentioned here that on some systems, sizing information and compilation times are difficult to obtain.

1568. Under the proposed organization, shouldn't problems be reported to the ADMO?

1661-849. The evaluation steps and procedures seem overly involved, complex and require a great deal of administrative overhead. Following is a high level summary of another approach to the steps and procedures.

1. The client obtains the ACEC. (Included are instructions for requesting an evaluation).

2. The client requests an evaluation by completing the Evaluation Test Plan. The client states whether they require assistance in customizing the support software and in running the test suite.

3. The AEF approved Evaluation Test Plan constitutes an Evaluation Agreement.

4. The client prepares the test suite for execution (with possible AEF assistance), executes the test suite and analysis tools and submits the results to the AEF. Included is client information on Evaluation Issues.

5. The AEF reviews the results and arranges to visit the client for the purpose of formal demonstration of test execution.

6. The test suite (or possibly a random sample) are re-executed at the clients facility. At this time, Evaluation Issues are discussed. Those that cannot be resolved will be noted in the final report for later resolution by the AJPO.

7. The AEF analyzes the results and prepares the final EAR and Performance Summary.

1874-889. I probably don't fully understand the definition of "derived" on lines 298-302 (seems vague). In this regard, I'm not certain that all "derived" implementations require re-evaluation. Does this matter need to be studied further?

11037-1041. It seems as though hardware and operating systems should be mentioned.

11044. Typo: "several"

11088. "weight or size". Poor examples?

Mike Burlakoff

January 14, 1989

Nelson:

Following are my comments to the ACSH. I'll try to make clear page/para references so my comments can be easily understood. I'd like to keep the draft copy for reference. (It won't be reproduced or distributed).

Reference the specific points that you asked us to consider. I won't have any specific comments on any of them. However, I hope that the following comments will give you information on my opinion of some of those.

P6, 1st answer, 2nd sentence: Do you mean "Errors in the runtime system ...", or something like "A poorly designed runtime system (excessive overhead, poor functional accuracy ..."?

P6, last answer: Could a point be made here that one of the objectives of Ada is to "share" or "reuse" code for cost savings ...

P7, 3rd answer: Another point that may be worth mentioning is that in the past, developers largely accepted the language/compiler and completed the development. (Then why do we need to evaluate Ada comilers today)?

P8, 1st question: Typo: "optionization"

P8, 2nd answer: I'd explain what I meant by optimization bugs. It also might be useful to add that debuggers themselves may contain bugs...

P9, 3rd answer, 4th line: Typo: "may great".

P9, last answer: Unless you saw the results of the AES evalualtions, I doubt if I would say it was a "notable" exception. Also, see my later comments regarding the other test suites (chapter 8).

P11, 1st answer: Typo: "of the shelf".

P13, 5th para. Typo. 1.11 should be 1.12.

P18, 4th para. I know that it is not your intent, but this could be misunderstood to read that for $500. an evaluation will be performed.

P18, para 3.2.4. I don't agree that a rehosted system suggests compile time emphasis. Primarily because rehosting largely deals with new operating system interfaces. I would not suggest only subset retesting for either rehost/retarget. In my view, you need the complete evaluation.

P18, para 3.2.5, 1st sentence. Rather than saying "for one user will not be acceptable to another...", would something like: "for one user will not meet the requirements of another...". For example, your evaluation of the real time performance is acceptable to me, but I need more emphasis on I/O...

P16-19, section 3.2. You discuss this in Section 9.1, but would it be useful if you gave a summary of your view of what a evaluation should consist of (as a summary of this section). For example:

1) A comprehensive "approved" benchmark test suite execution.
2) Approved checklist evaluation.
3) ...

P21, Intro para. It seems to me that the most important benchmark criteria is the skill, knowledge and experience of the benchmark designers/developers (as well as the users of the benchmarks).

P21, 2nd para, 4th sentence. Is this a problem with benchmarking or simply that one should be aware that benchmarks execute in the environment and reflect the environment.

P24, Operating and Runtime System bullet. Another major cause of results variation is the effects of a multiprogramming/multiprocessing environment.

P24, para 4.3. Somewhere, you might mention that some benchmarks use operating system computed CPU time and others will compute time using wall clock time.

P25, 3rd para. Typo: "A logic analyzer a".

P27. Bullets. It seems like all the points should apply, regardless of whether compile time or runtime.

P32, 2nd para. Typo: "Ada program...".

P32, 3rd para. Don't understand . "state of practice has raised expections ...". (typo?)

P32/33, bullets/questions. Another bullet might be whether more than one user can simultaneously invoke the compiler (and the consequences, if any). (or is that what is meant by the P33 1st bullet)?

P34, para 5.4. Would this be the proper place to emphasize the usefullness of "interactive help" information availability.

P35, para 5.4.3. One of the major documentation requirements should be a clear explanation of each compiler error message in an easy to find/read format (categorized by severity).

P37, 4th para, last sentence. I doubt if many compiler users would actually evaluate operating systems. However, as you've pointed out they need to be aware of capabilities and performance characteristics.

P38, Bare Machine Environment. Should the middle block include "Executive Functions" in the text description?

P39, sec 6.2.1, 1st para. I don't agree with the implication that much can be learned in a "short" amount of time by simply inspecting assembly code. It generally requires highly technical, experienced people (who are also familiar with compiler code generation) to analyze compiler code generation and the many available options to the compiler writer. This comment also applies to the last para on page 40 for sec 6.2.1.

P41, First 5 bullets. Should a bullet be added to read "Input/Output"?

P42, last bullet. Typo: "and redundent".

P43, 3rd para. I agree that it's usually difficult to determine the size of the runtime system. Is it because no one usually takes the time to write a program to automatically read the load map...?

P47, para 6.8. You may wish to mention that the ACEC contains several tests which test interrupt processing performance in a tasking environment.

P53, para 7.2.1. "more fully covered in chapter 6". I couldn't find the material in chapter 6 (but probably didn't spend enough time looking).

P54, para 7.2.3. In every project that I've been involved in, downloading/ uploading is a real ordeal (time consuming, complex, error prone, etc). (in more "modern" times, there may have been improvements). In any case, if you'd have some recommendations for developers/evaluators in this area, it would probably be useful info.

P55, 2nd para. Typo: "conduct on".

P54-56, Sec 7.3. My early experience with debuggers was that they were unreliable, contained errors themselves and broke under stress. (Hopefully they've improved). When using them I always wondered if the bug was in my program or the debugger. Therefore, I recommend emphasizing that the user be aware of the quality, history, etc. of these tools.

P56, last para, 1st line. Typo: "a more".

P56-57, sec 7.4. As with debuggers, users need to be aware of the quality, possible errors, etc. in simulators.

P57, last para. I understand your intention of "integration". But, I for one, believe that the worst possible APSE is one whereby all of the tools are so called "integrated". Those APSEs are large, complex, costly and never work properly.

P59-66, chapter 8.

My familiarity with the PIWG, U. of Mich, AES, ASR and Aerospace suites is primarily from reading your material and talking with users of a couple of the test suites. The developers of these other systems (who were primarily volunteers) are to be commended for their splendid products and efforts. However, my conclusion is that none of these are in the same class or measure up to: 1) the extent of coverage, 2) complexity of the issues that are covered, or in 3) the overall quality of the ACEC. I say this because:

1) The ACEC is a planned, systematic, formal development requiring person years for the development. The developers are recognized, skilled professionals.

2) The test suite coverage is extensive, and attempts to evaluate many of the more difficult runtime issues (task loading, interrupt processing, design trade-offs, exception handling, file I/O, ...).

3) A great deal of design/development effort has been expended to provide an accurate and usable test environment.

a. The timing code is extensive and provides information on the accuracy that was achieved (or not achieved).

b. The individual test template structure is such that it is easy for the user to construct and insert new tests into the environment.

c. The analysis tools provide excellent statistical comparisons and summaries.

4) The ACEC is an on-going effort (I don't have the information on the contractual status) that is being reviewed (E&V) and monitored for quality.

5) The documentation is complete, extensive and of high quality. For example, the Readers Guide contains excellent background material on a number of technical issues and trade offs relating to evaluations.

6) The test suite is sponsored and supported by the government.

In view of the above, I believe that the ACEC, (given an unbiased evaluation by users of the system) will become the industry standard, and should be promoted as such. My recommendation, therefore, is that a separate chapter be devoted to the ACEC description.

In reviewing ACEC documentation this past Summer, one of my major criticisms was that the documentation did not sufficiently describe the extent of coverage of issues. In this regard, I'd recommend greatly expanding the information that is currently on pages 61-63. I'd chose relevent material from the Users and Readers Guides and the VDD. For example, the VDD could be used to expand the present bullets listed on page 62. Following is an example:

o Classical tests such as Ackerman function, Dhrystone, Whetstone, ...

o Avionics application study

o Kalman filter tests

o Delay statements with various delays in continuity/non-continuity...

o Data encryption standards programs

o Interrupt timing tests

o I/O processing tests

o Memory reclamation tests

o Language feature tests (referenced to the LRM)

o Tasking applications tests

P68, recommended additional bullet: One of the major areas may be the determination of the maturity of the systems. (Used on large/small projects...).

P68, para 8. The third sentence seems confusing? (Checklist data ...).

P69, 2nd para, 4th sentence. Since the instruction set architechture is where the real work is done, my preference would be to select the desired target system and insist that the compilation system provide the quality of support that is needed.

P69, last para. Difficult to understand: "required for both evaluation...".

P70, last 2 sentences. I'd question any shortcut recommendations.

P72, para 9.7, intro para. In general, I'd agree with the statement for MIS applications. But is it true for embedded systems? Aren't we likely to upgrade our weapons systems? Wouldn't there be real cost savings if we could port some of the applications to the new target?

P74, 1st 3 bullets. Possible bullets to be added:

o Past contractual history (overruns, but no useful delivery)

o Maintenance response times

o Quality of documentation. Are there procedures to update the documentation as the system is updated.

In summary, the ACEC is a top quality product which contains a great deal of useful information. It should be a welcome source of information to managers (as well as technicians) as an aid in the evaluation process.

Mike Burlakoff

APPENDIX F

E&V Project Presentations

Raymond Szymanski, Project Manager

For a majority of the presentations listed below, presentation material was selected from a core set of approximately seventy-five viewfoils. The number of foils used per presentation varied from a mere few to over sixty. (Sixty-nine foils were used at the Naval Post-Graduate School (NPGS) lecture, December 1989.) Throughout the current reporting period many of the foils in the core set were updated to reflect the progress of the E&V Project, its technology developments and surrounding environment.

To reduce the volume of available presention material for this publication, much of which would be replication, only the material from the NPGS lecture (p. F-4), the Tri-Ada '89 presentation (p. F-39), and status foils for each of the contractual efforts (ACEC, p. F-52; CIVC, p. F-59; Reference System, p. F-65) are included herein. The NPGS lecture and contract status foils should give the reader some insight into the type of material which was used to represent the E&V Project during the specified reporting period and at the same time bring much of it up to date. The Tri-Ada '89 presentation material is included, along with the text, because its contents differs significantly from the core set of foils and also marks the first time that mandated use of E&V Project-developed technology was publicly debated.

In addition to the presentations listed below, the E&V Project Manager has given innumerable presentations/briefings internal to the management of the the Ada Joint Program Office and the US Air Force. This includes the Director and staff of the Ada Joint Program Office, Air Force Systems Command, Air Force Aeronautical Systems Division, Air Force Avionics Laboratory, and Air Force Wright Research and Development Center. Although much of the material used for these presentations was technical in nature it also included significant amounts of administrative and managerial detail which is not appropriate for this report. Therefore, this material is not included herein.

The E&V Project Manager apologizes, in advance, to any organization which received an E&V Project presentation for the specified reporting period but is not listed below. An earnest attempt was made to be thorough and complete in compiling the list. But, alas, after five years of managing the E&V Project and chairing the E&V Team meetings, the project manager is growing somewhat feeble and subject to an occasional error.

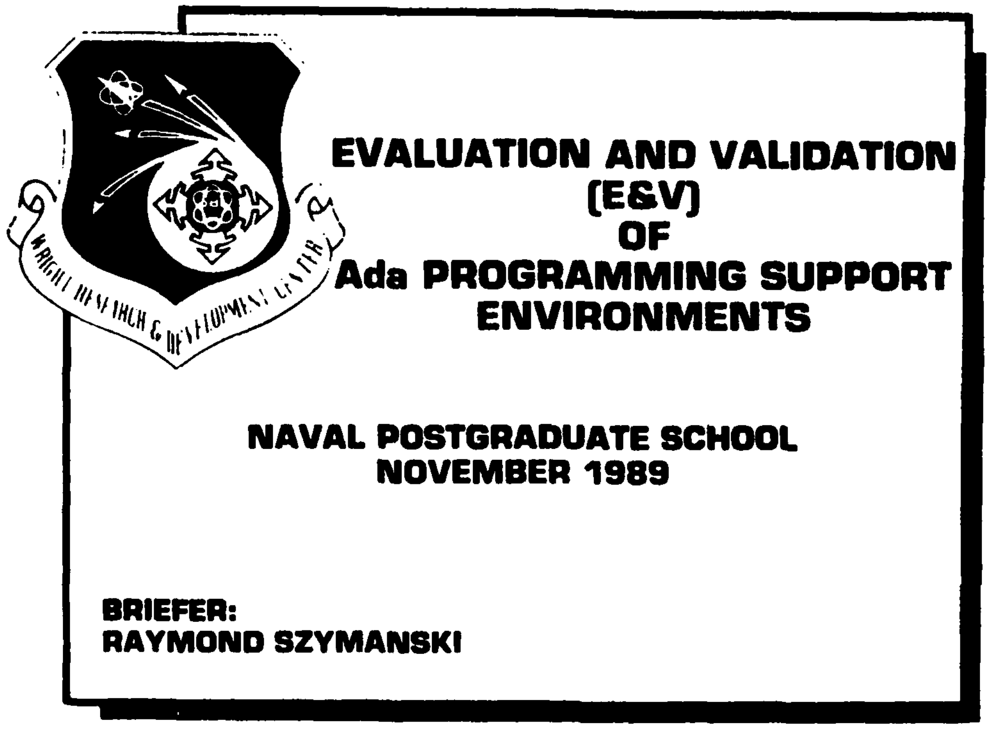
List of E&V Project Presentations for the period 1 December 1988 -- 1 August 1990

August 1990	Ada Executive Officials Meeting	Washington, DC
June 1990	Ada Board Meeting	Washington, DC
June 1990	SAF/AQK- Deputy Assistant Secretary for Comminications Computers and Logistics	Washington, DC

June 1990	Ada Executive Officials Evaluation and Validation Working Group	Washington, DC
June 1990	HQ USAF/SCTT	Washington, DC
June 1990	Ada Europe	Dublin, Ireland
May 1990	23rd Quarterly E&V Team Meeting	Pittsburgh, PA
May 1990	Space Defense Initiative Office	Washington, DC
May 1990	HQ USAF/SCTIA	Washington, DC
April 1990	Software Technology Support Center Conference	Salt Lake City
March 1990	1990 Air Force Avionics Symposium	Las Vegas, NV
March 1990	22nd Quarterly E&V Team Meeting	Denver, CO
February 1990	Ada Joint User's Group Meeting	San Diego, CA
December 1989	Standards Implmentation Policy Board	Washington, DC
December 1989	Naval Post Graduate School Guest Lecturer	Monterrey, CA
December 1989	21st Quarterly E&V Team Meeting	San Diego, CA
November 1989	Government Accounting Office	Washington, DC
October 1989	Tri-Ada Conference	Pittsburgh, PA
September 1989	IEEE Environments Working Group Meeting	San Francisco
July 1989	Ada Joint User's Group Meeting	Denver, CO
June 1989	Washington Ada Symposium	Washington, DC
June 1989	20th Quarterly E&V Team Meeting	Dayton, OH

May 1989	Wright Research and Development Center Ada Forum	Dayton, OH
May 1989	US Air Force Systems Acquisition School	San Antonio, TX
May 1989	National Aerospace Electronics Conference	Dayton, OH
March 1989	19th Quarterly E&V Team Meeting	Denver, CO
April 1989	Undersecretary of Defense for Research and Engineering	Washington, DC
April 1989	Electronic Industries Assoc. Computer Resources Committee	Dayton, OH
April 1989	Ada Joint Program Office	Washington, DC
February 1989	Automated Logistics Management Systems Activity (ARMY)	St. Louis, MO
December 1988	Special Interest Group/Ada Conference	Los Angeles, CA
December 1988	18th Quarterly E&V Team Meeting	San Diego, CA

MCS 119158

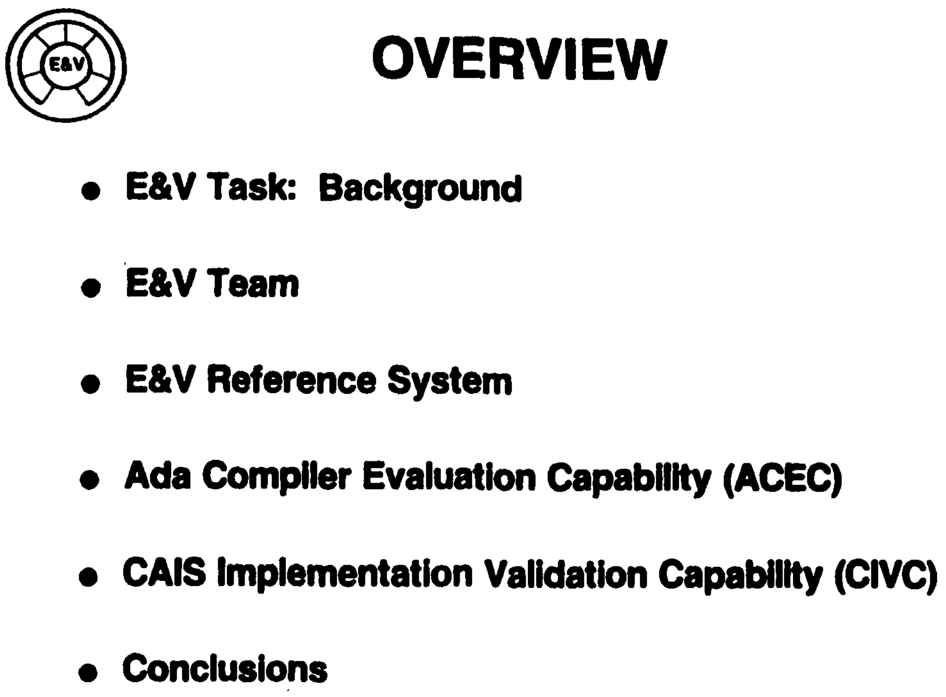


EVALUATION AND VALIDATION (E&V) OF Ada PROGRAMMING SUPPORT ENVIRONMENTS

NAVAL POSTGRADUATE SCHOOL
NOVEMBER 1989

BRIEFER:
RAYMOND SZYMANSKI

1-10372



OVERVIEW

- E&V Task: Background
- E&V Team
- E&V Reference System
- Ada Compiler Evaluation Capability (ACEC)
- CAIS Implementation Validation Capability (CIVC)
- Conclusions

T 16375

OVERVIEW

- E&V Task: Background
  - Definitions
  - Need for E&V
  - Task Purpose
  - Task Process
- E&V Team
- E&V Reference System
- Ada Compiler Evaluation Capability (ACEC)
- CAIS Implementation Validation Capability (CIVC)
- Conclusions

T 16375

E&V TEAM PURPOSE

- Provide DoD with a forum to discuss evaluation and validation issues
- Provide technical expertise in development of E&V technology

  Team products
  Task products
- Represent E&V point of view in the larger community

DEFINITIONS

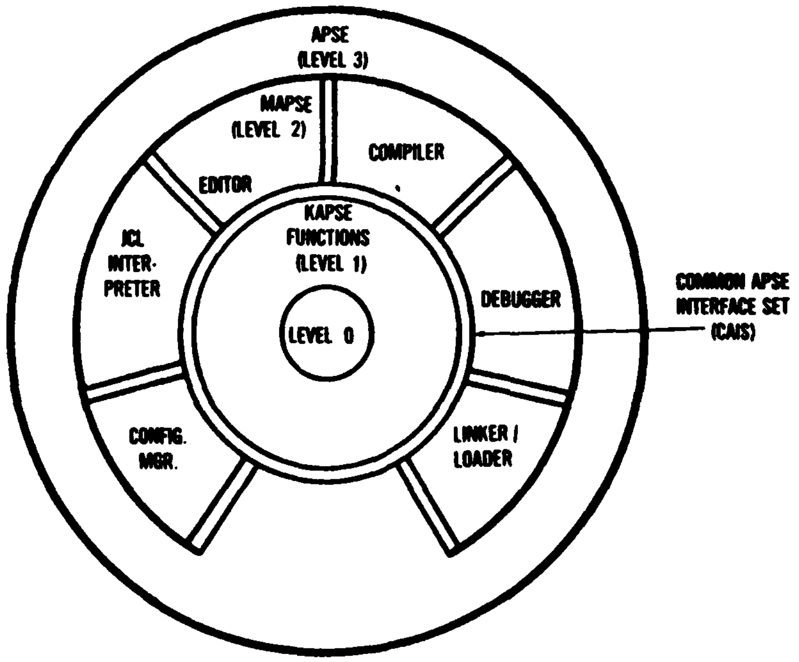
APSE -- Ada Programming Support Environments

E & V -- Evaluation and Validation

Evaluation -- Assessment of Performance and Quality

Validation -- Assessment of Conformance to a Standard

APSE DEFINITION



NEED FOR APSE E&V TECHNOLOGY

- **IMPORTANCE OF ENVIRONMENT DECISIONS**
  - Large, Critical Ada-based Systems
  - Major Investments for Software Developers
  - Major Influence on Software Maintenance

- **DIFFICULTY OF ASSESSING APSEs AND TOOLS**
  - APSEs are Complex Systems
  - Great Diversity of Choice and Viewpoints
  - Rapid Technological Change
  - Lack of Relevant Historical Data

E&V TASK PURPOSE

To Provide a Focal Point for Addressing Community Need for E&V Technology — Assess APSEs and Components

ACTIVITIES
1) **Identify and Define Requirements**
2) **Develop Selected Elements**
3) **Encourage Others to Develop Some**
4) **Collect Relevant Information**
5) **Disseminate Information**

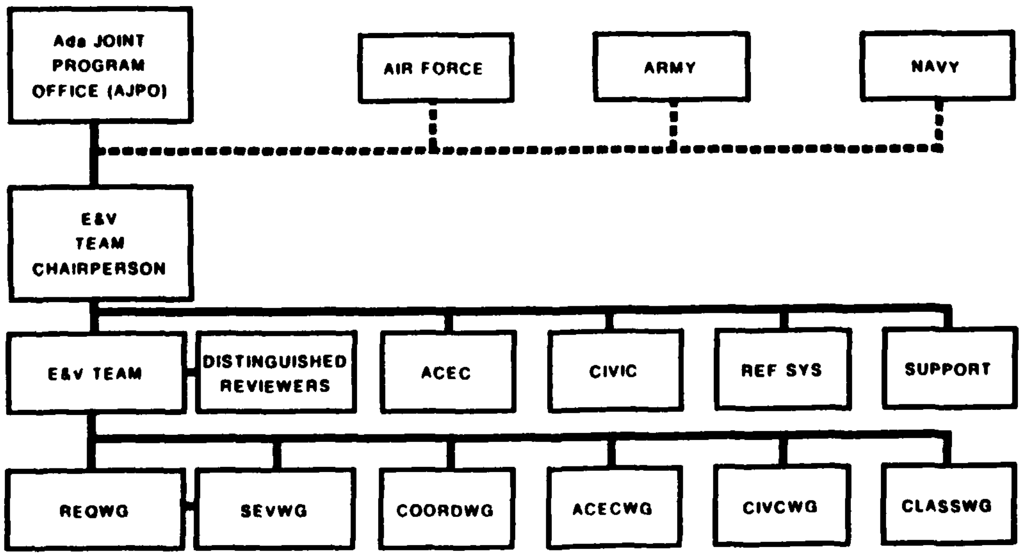
[illegible]

E&V TASK PROCESS

SPONSOR	Ada Joint Program Office
LEADER	Air Force Avionics Laboratory Mr. Raymond Szymanski, Chair
TECHNICAL ADVISORS	The E&V Team — Government, Industry and Academia Representatives
CONTRACTORS	TASC — Technical Support and Reference System Boeing — Ada Compiler Evaluation Capability (ACEC) SofTech — CAIS* Implementation Validation Capability (CIVC)

* Common APSE Interface Set -- MIL-STD-1838

E&V MANAGEMENT STRUCTURE



I 16374

OVERVIEW

- E&V Task: Background
- ➡ E&V Team
  - Team Purpose
  - Composition
  - Method of Operation
  - Team Products
  - Relation to the Acqusition Process
- E&V Reference System
- Ada Compiler Evaluation Capability (ACEC)
- CAIS Implementation Validation Capability (CIVC)
- Conclusions

E&V TEAM

- DIRECTIONAL MANAGEMENT WORKING GROUPS
- TECHNICAL MANAGEMENT WORKING GROUPS

E&V TEAM
(CONT'D)

- **DIRECTIONAL MANAGEMENT WORKING GROUPS**
  - **REQUIREMENTS WORKING GROUP**
  - **STANDARDS EVALUATION WORKING GROUP**
  - **COORDINATION WORKING GROUP**

F 16376

OVERVIEW

- **E&V Task: Background**
- **E&V Team**
- ➡ **E&V Reference System**
  - **Purpose**
  - **Organization and Use**
    - **Reference Manual**
    - **Guidebook**
  - **Relation to the Acqusition Process**
- **Ada Compiler Evaluation Capability (ACEC)**
- **CAIS Implementation Validation Capability (CIVC)**
- **Conclusions**

THE "TOOLS AND AIDS" DOCUMENT

- **A PRODUCT OF THE REQUIREMENTS WORKING GROUP (REQWG) OF THE APSE E&V TEAM**
- **REFLECTS EARLIER WORK* OF REQWG, DISCUSSIONS WITH ALL E&V TEAM MEMBERS, AND SURVEYS OF APPROPRIATE ARPANET-MILNET INTEREST GROUPS**
- **PURPOSE**

  To identify the community's E&V Technology needs — kinds of assessors to acquire

  To prioritize the needs

  To provide information to those willing/able to fund E&V Technology Development

*"*Requirements for the Evaluation and Validation of Ada Programming Support Environments, Version 2.0,*" REQWG, December 1986

E&V TEAM (CONT'D)

- **TECHNICAL MANAGEMENT WORKING GROUPS**
  - **E&V TECHNOLOGY CLASSIFICATION WORKING GROUP**
  - **Ada COMPILER EVALUATION CAPABILITY WORKING GROUP**
  - **CAIS IMPLEMENTATION VALIDATION CAPABILITY WORKING GROUP**

E&V TEAM PRODUCTS

- E&V TEAM PUBLIC REPORT
  - E&V TEAM REQUIREMENTS ANALYSIS DOCUMENT
  - E&V TOOLS AND AIDS DOCUMENT
  - E&V TEAM PUBLIC COORDINATION DOCUMENT
  - CAIS ISSUES AND STRATEGIES DOCUMENT
  - E&V TEAM MEETING MINUTES
  - E&V TEAM WHITE PAPERS
- UNLIMITED DISTRIBUTION
- AVAILABLE FROM NTIC/DTIC
- NORMALLY PRODUCED ON AN ANNUAL BASIS

I 16383

RELATION TO THE ACQUISITION PROCESS

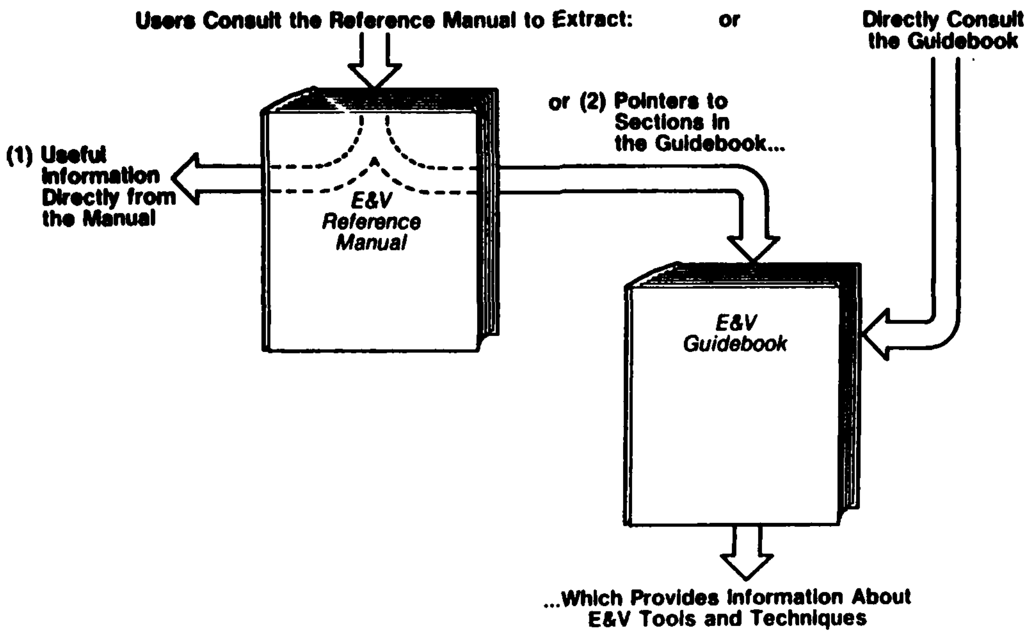
- **Opportunity to provide inputs to DoD E&V technology focal point**

  Increase DoD awareness of E&V technology needs

  Opportunity to impact E&V technology developments in process

  Educational opportunity

USE OF THE REFERENCE SYSTEM



Users Consult the Reference Manual to Extract:
or
Directly Consult the Guidebook
or (2) Pointers to Sections in the Guidebook...
(1) Useful Information Directly from the Manual
E&V Reference Manual
E&V Guidebook
...Which Provides Information About E&V Tools and Techniques

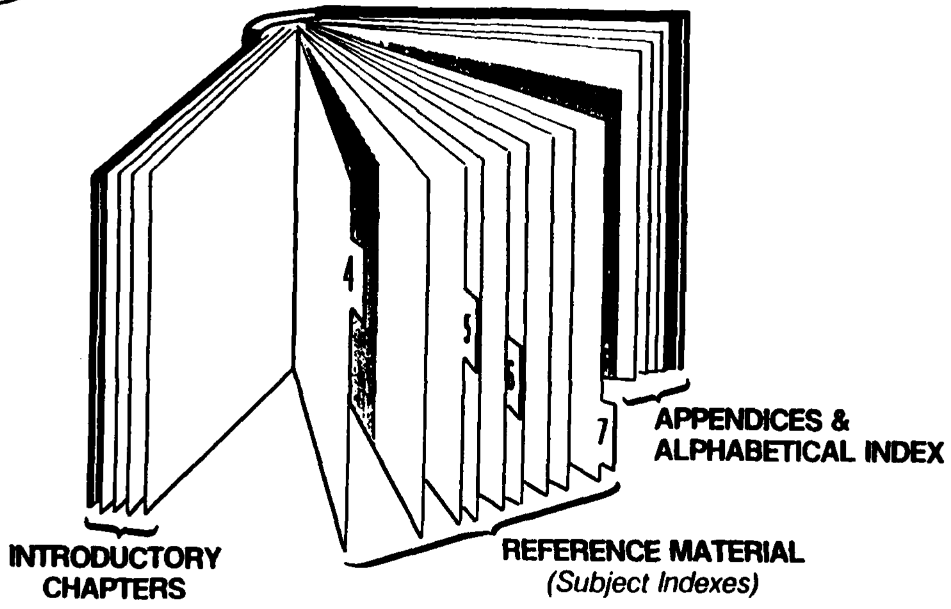
WHY USE THE E&V REFERENCE SYSTEM ?

THE E&V REFERENCE SYSTEM SHOULD HELP USERS TO:

- GAIN OVERALL UNDERSTANDING OF APSEs AND APPROACHES TO ASSESSMENT
- FIND USEFUL INFORMATION -- TERMINOLOGY, DEFINITIONS, RELATIONSHIPS
- FIND ASSESSMENT CRITERIA / METRICS AND "POINTERS" TO SPECIFIC EVALUATION OR VALIDATION TECHNIQUES
- FIND DESCRIPTIONS OF EVALUATION OR VALIDATION TECHNIQUES
- FIND GUIDANCE IN THE SELECTION, INTERPRETATION, AND INTEGRATION OF EVALUATION AND VALIDATION TECHNIQUES AND RESULTS

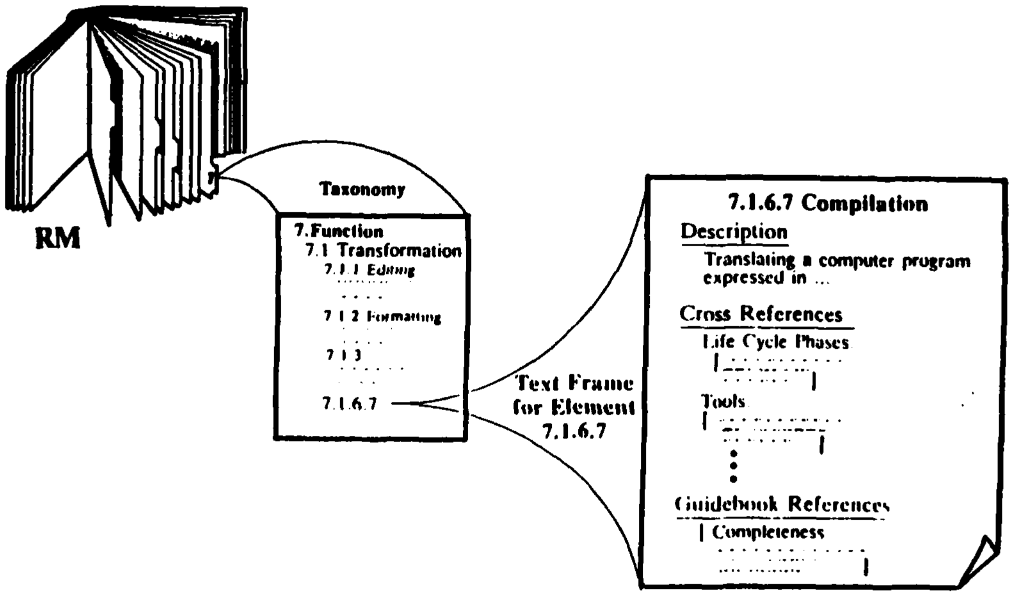
G 09879
10/26/88

REFERENCE MANUAL ORGANIZATION



G 09880
11/11/88

INDEXES AND TEXT FRAMES



EXAMPLE COMPILER TEST CAPABILITIES CHECKLIST *(PAGE 1 OF 2)*

5.11 ARTEWG RUNTIME ENVIRONMENT TAXONOMY

Purpose: Describes the basic elements of Ada runtime environments and provides a common vocabulary. The following excerpt is taken from the introduction to the Taxonomy section. "If a runtime environment for an Ada program is composed of a set of data structures, a set of conventions for the executable code, and a collection of predefined routines, then the question arises: what are examples of these elements, and moreover, what is the complete set from which such elements are taken when a particular runtime environment is built?...It should be noted that the dividing line between the predefined runtime support library on one hand, and the conventions and data structures of a compiler on the other hand, is not always obvious. One Ada implementation may use a predefined routine to implement a particular language feature, while another implementation may realize the same feature through conventions for the executable code. ... This taxonomy concerns itself primarily with those aspects of the runtime execution architecture which are embodied as routines in the runtime library. It does not treat issues of code and data conventions, nor issues related to particular hardware functionalities, in any great depth."

[@RM: Runtime Environment 7.2.3.5, @RM: Power 6.4.21]

Primary References:

[ARTEWG 1988] "A Framework for Describing Ada Runtime Environments." Proposed by Ada Runtime Environment Working Group (SIGAda), Ada Letters, Volume VIII, Number 3, May/June 1988, pp. 51-68

Vendors/Agents: [ARTEWG]

Method: Capabilities checklist

Inputs: Capability checklist (see Table 5.11-1)and runtime environment documentation.

Process. Check off capabilities demonstrated by the runtime environment or discussed in the documentation.

Outputs A list of capabilities performed by the runtime environment.

EXAMPLE COMPILER TEST CAPABILITIES CHECKLIST *(PAGE 2 OF 2)*

TABLE 5.11-1

RUNTIME ENVIRONMENT TAXONOMY

FEATURE	FOUND
Runtime Execution Model	
Dynamic Memory Management	
Processor Management	
Interrupt Management	
Time Management	
Exception Management	
Rendezvous Management	
Task Activation	
Task Termination	
I/O Management	
Commonly Called Code Sequences	
Target Housekeeping Functions	

T 11601

EXAMPLE COMPILER TEST SUITE SUMMARY
(PAGE 1 OF 3)

5.3 Ada COMPILER EVALUATION CAPABILITY (ACEC)

Purpose: The purpose of this test suite is best stated by the following quote taken from the introduction in the ACEC Reader's Guide "The ACEC ... consists of a portable test suite and support tools ... It contains test problems designed to measure the execution time and size of a systematically constructed set of Ada examples The support tools assist the ACEC user in executing the test suite and analyzing the results obtained " The scope of coverage provided by the test suite is shown by the following excerpts from the ACEC classification taxonomy:

II. Execution Time Efficiency
   A. Language Feature Efficiency
      1. Required (referenced by LRM section)
      2. Implementation Dependent (referenced by LRM section)
         - attributes (LRM Appendix A)
         - record representation clauses
         - interrupts
         - language interface
         - unchecked programming
   B. Pragmas
      1 Predefined
      2. Implementation Defined
   C. Optimizations
      1 Classical
      2 Effects of Pragmas
      3. Static Elaboration
         - aggregates
         - tasks
      4 Language Specific
         - Habermann-Nassi transformation for tasking
         - delay statement optimization
   D Performance Under Load
      1 Task Loading
         - task creation
         - task termination
         - task abortion
         - Dining Philosophers Problem
         - task starvation
         - task delay
      2 Levels of Nesting
         - static
         - dyna
      Param

T-11602

EXAMPLE COMPILER TEST SUITE SUMMARY

5.4 PIWG BENCHMARK TESTS

Purpose: Identification of performance characteristics of Ada compilers. The tests examine the performance of the compiler itself in terms of compilation speed, as well as the quality of the generated code for both processing and storage effectiveness. The test suite measures performance for both isolated language features and composites or mixes of language features (using the Whetstone and Dhrystone tests).
[@RM: Compilation 7.1.6.7, @RM: Processing Effectiveness 6.4.22; @RM: Storage Effectiveness 6.4.31]

Primary References:

Host/OS: Unrestricted

Vendors/Agents: [PIWG]

Method:
Automated test suite.
Inputs: PIWG source code, Ada compiler and runtime system, and host (and target) computer.
Process:
1. Obtain the latest PIWG tests
2. Compile and run tests according to the documentation.
Outputs: Reports on the outcome of each test run

I 16177

GUIDEBOOK ORGANIZATION

1. **Introduction**
2. **Structure and Use of the Guidebook**
3. **Integration of E&V Technology**
4. **Synopses**

Early Chapters

5. **Compilation System Assessors**
6. **Target Code Generation Aids and Analysis Assessors**
7. **Test System Assessors**
8. **Tool/Host Interface Assessors**
9. **Requirements/Design Support Assessors**
10. **Configuration Management Support Assessors**
11. **Distributed System Development and Runtime Support Assessors**
12. **Distributed APSE Assessors**
13. **"Whole APSE" Assessors**
14. **Adaption Assessors**

99. **Other Assessors**

Formal Chapters

I 16178

CHECKLISTS IN THE GUIDEBOOK

Compilation Capabilities Checklist	Real-Time Analysis Capabilities
Program Library Management Capabilities Checklist	Testing Capabilities Checklist
Runtime Environment Taxonomy	Configuration Management Capabilities Checklist
Assembling Capabilities Checklist	Text Editing Capabilities Checklist
Linking/Loading Capabilities Checklist	Database Management Capabilities Checklist
Import/Export Capabilities Checklist	Electronic Mail Capabilities Checklist
Emulation Capabilities Checklist	Requirements Prototyping Capabilities Checklist
Debugging Capabilities Checklist	Performance Monitor Capabilities Checklist
Timing Analysis Capabilities Checklist	Simulation and Modeling Capabilities Checklist
Tuning analysis Capabilities Checklist	File Management Capabilities Checklist

I 16179

EXAMPLE CHECKLIST ASSESSOR
(PAGE 1 OF 3)

5.8 COMPILATION CHECKLIST

Purpose: Evaluation of the power of compilation by developing a list of functional capabilities [@RM: Compilation 7.1.6.7, @RM: Power 6.4.2]

Primary Reference:
[@E&V Schema 1987: B.;
Classification Schema/E&V Taxonomy Checklists: 4.4]

Vendors/Agents: [E&V Team]

Method: Capabilities checklist

Inputs: Capability checklist (see Table 5.8-1) and compiler documentation.

Process: Check off capabilities demonstrated during compiler runs or discussed in the documentation.

Outputs: A list of capabilities provided by the compiler

I 16180

EXAMPLE CHECKLIST ASSESSOR
(PAGE 2 OF 3)

TABLE 5.8-1
COMPILATION CAPABILITIES CHECKLIST

FEATURE	FOUND
Conditional Compilation	
Debug Information Generation	
Enable/Disable Listing	
Errors Only Listing	
Error Identification	
Set Default Directory For Source	
Set Listing Width And Height	
Specify Different Program Library	
Specify Main Program	
Disable Use Of SYSTEM Library	
Suppress All Run-Time Checks	
Compile Multiple Files	
Language Sensitive Editor Support	
Specify Error Limit	
Enable/Disable An Error Category	
Specify Optimization Parameters	
Syntax Only Checking	
Symbol Table	
Variable Set/Use Indications (Cross-Reference)	
Object Code Listing	

T-16381

EXAMPLE CHECKLIST ASSESSOR
(PAGE 3 OF 3) (Cont.)

TABLE 5.8-1
COMPILATION CAPABILITIES CHECKLIST (Cont.)

FEATURE	FOUND
Object Attribute Map	
Code Statistics	
Unidentified Compiler Options (Pragmas)	
Uncontrolled Dynamic Storage	
Elaboration Control	
Inline Expansion of Subprograms	
Interface With Other Languages	
Specify Memory Size	
Pack Data Representations In Memory	
Priority Control of Concurrent Tasks	
Shared Variables	
Specify Storage Unit	
Specify Alternative System Characteristics	
Machine Code Mapping	
Machine Code Insertions	
Cross Compilation	
Error Reporting	
Exceptions List	
Identify Target Dependencies	

T 16382

THE "MODEL PROJECT/STRUCTURED EXPERIMENT" APPROACH TO WHOLE-APSE EVALUATION

- **Create model project — code and 2167A documents**
- **Build "Scenarios" around, for example**
  - Requirements changes
  - Test exercises
  - Version control
  - Transitions between phases
- **Evaluate APSE from whole-team/whole-project perspective**
- **Extend/add scenarios to address almost any phase/activity of interest**

L 16845
5 05 89

E&V TECHNOLOGY MATRIX

Assessment Techniques

Assessment Subjects	Evaluation: Checklists etc	Evaluation: Test Suites	Evaluation: Structured Experiments	Evaluation: Decision Suport Systems	Validation: Test Suites	Validation: Formal Methods
Compilation Systems	◑	◑			◕	
Target Code Generation Aids, etc.	◕					
Test Systems	◑		◷			
Tool/Host Interfaces	○		◷		○	
Requirements/Design Tools	○		◷			
CM Support Tools	◑		◷			
Distributed System Dev. Tools	○					
Distributed APSEs	○					
Whole APSEs	◑		○	◷		
Adaptation Features	◑					
Others	◔					

F-16384

OVERVIEW

- **E&V Task: Background**
- **E&V Team**
- **E&V Reference System**
- ➡ **Ada Compiler Evaluation Capability (ACEC)**
  - **Objectives**
  - **Why Evaluate Compilers?**
  - **Contents**
  - **Application to the Acquisition Process**
- **CAIS Implementation Validation Capability (CIVC)**
- **Conclusions**

ACEC OBJECTIVES

1. COMPARE THE PERFORMANCE OF SEVERAL Ada COMPILER SYSTEMS
2. ISOLATE THE STRONG / WEAK POINTS OF A SPECIFIC SYSTEM
3. DETERMINE WHAT SIGNIFICANT CHANGES WERE MADE BETWEEN RELEASES OF A SPECIFIC COMPILER
4. PREDICT THE PERFORMANCE OF DIFFERING Ada DESIGN APPROACHES

WHY EVALUATE COMPILERS ?
(ISN'T VALIDATION ENOUGH ?)

1. Ada COMPILER VALIDATION ALONE DOES NOT MEASURE COMPILER QUALITY
2. APPLICATION DESIGNERS NEED AN ESTIMATE OF A COMPILERS PERFORMANCE BEFORE DEVELOPMENT
3. SELECTION OF AN INADEQUATE Ada COMPILER CAN LEAD TO THE FOLLOWING:
   - COST OVERRUNS
   - REDUCED PROGRAMMERS PRODUCTIVITY
   - SCHEDULE DELAYS

ACEC APPROACH

THE ACEC MEASURES THE FOLLOWING TEST ATTRIBUTES:

1. COMPILE TIME EFFICIENCY
2. EXECUTION TIME EFFICIENCY
   - MULTIPLE CATEGORIES
3. CODE SIZE EFFICIENCY
   - CODE EXPANSION SIZE
   - RUN TIME SYSTEM SIZE

ALL DATA PRODUCED IS USED AS INPUT TO THE MEDIAN PROGRAM TO OBTAIN A STATISTICAL SUMMARY OF THE DATA

ACEC USERS
(WHO CAN BENEFIT)

- DoD SYSTEM PROGRAM OFFICES
- Ada COMPILER VENDORS
- Ada PROGRAMMERS
- SOFTWARE DEVELOPMENT MANAGERS

[illegible]

ACEC: CONTENTS

- **Suite of 1076 tests**
- **Support and analysis tools**
- **User Documentation**

EXECUTION TESTS CLASSIFICATION

1. INDIVIDUAL LANGUGAGE FEATURES
   - REQUIRED
   - IMPLEMENTATION DEPENDENT
2. OPTIMIZATIONS
3. PERFORMANCE UNDER LOAD
4. DESIGN TRADE OFF'S
5. OPERATING SYSTEM EFFICIENCY
6. APPLICATION PROFILES
   - CLASSICAL
   - Ada IN PRACTICE
   - IDEAL Ada

ACEC TESTING AREAS (OF MISSION CRITICAL SIGNIFICANCE)

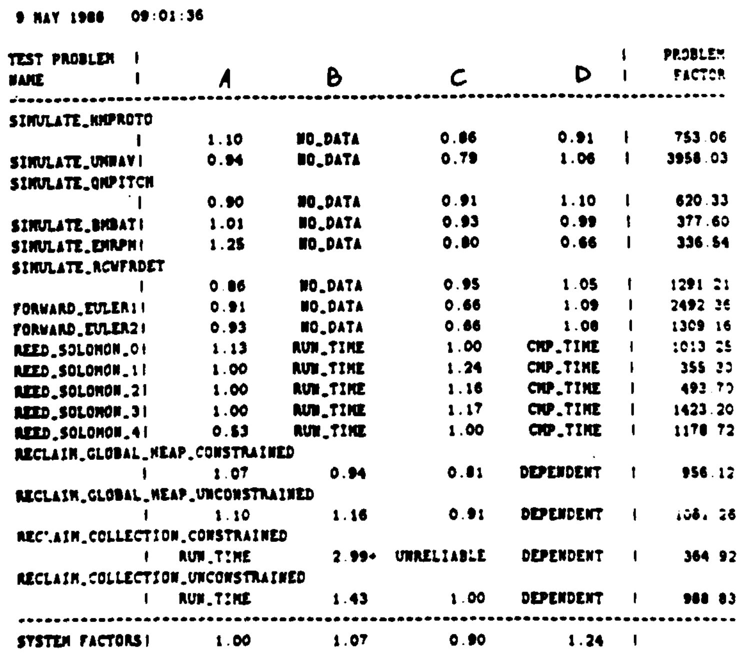
- MEMORY MANAGEMENT / STORAGE RECLAIMATION
- INTERRUPT HANDLING
- TASK PERFORMANCE
- RUN TIME CHECKING
- EXECEPTION HANDLING / PROPAGATION
- BIT MANIPULATION
- FLOATING POINT OPERATIONS

APPLICATION PROFILE TESTS

- DATA ENCRYPTION ALGORITHMS
- KALMAN FILTER ALGORITHM
- EW RECOGNITION AND TRACKING ALGORITHM
- E-3A SIMULATION
- ERROR CORRECTING CODE
- IN-FLIGHT PERFORMANCE MONITOR SYSTEM
- AL DATABASE APPLICATION

MEDIAN
ACEC OUTPUT TOOL

- MATRIX OF TIMING MEASUREMENTS
  - SYSTEM VS PROBLEM
- HISTOGRAM OF RESIDUAL VALUES
  - RELATIVE SYSTEM PERFORMANCE
  - DENOTES DEGREE OF OPTIMIZATION
- SUMMARY OF SYSTEM PERFORMANCE
- SUMMARY OF PROBLEM DIFFICULTY
- SUMMARY OF STATISTICAL DATA



```
9 MAY 1988   09:01:36

TEST PROBLEM  |                                                 |  PROBLEM
NAME          |       A          B            C           D     |   FACTOR
---------------------------------------------------------------------------
SIMULATE_HMPROTO
              |      1.10      NO_DATA       0.86        0.91   |   753.06
SIMULATE_UNHAV|      0.94      NO_DATA       0.79        1.06   |  3958.03
SIMULATE_QNPITCH
              |      0.90      NO_DATA       0.91        1.10   |   620.33
SIMULATE_BMBAT|      1.01      NO_DATA       0.93        0.99   |   377.60
SIMULATE_EMRPM|      1.25      NO_DATA       0.80        0.66   |   336.54
SIMULATE_RCVFRDET
              |      0.86      NO_DATA       0.95        1.05   |  1291 21
FORWARD_EULER1|      0.91      NO_DATA       0.66        1.09   |  2492 35
FORWARD_EULER2|      0.93      NO_DATA       0.66        1.08   |  1309 16
REED_SOLOMON_0|      1.13      RUN_TIME      1.00     CMP_TIME  |  1013 25
REED_SOLOMON_1|      1.00      RUN_TIME      1.24     CMP_TIME  |   355 33
REED_SOLOMON_2|      1.00      RUN_TIME      1.16     CMP_TIME  |   493 70
REED_SOLOMON_3|      1.00      RUN_TIME      1.17     CMP_TIME  |  1423.20
REED_SOLOMON_4|      0.53      RUN_TIME      1.00     CMP_TIME  |  1178 72
RECLAIM_GLOBAL_HEAP_CONSTRAINED
              |      1.07        0.94        0.81     DEPENDENT |   956.12
RECLAIM_GLOBAL_HEAP_UNCONSTRAINED
              |      1.10        1.16        0.91     DEPENDENT |  [illegible]
RECLAIM_COLLECTION_CONSTRAINED
              |   RUN_TIME       2.99+   UNRELIABLE   DEPENDENT |   364 92
RECLAIM_COLLECTION_UNCONSTRAINED
              |   RUN_TIME       1.43        1.00     DEPENDENT |   988 83
---------------------------------------------------------------------------
SYSTEM FACTORS|      1.00        1.07        0.90        1.24   |
16 MAY 1988   11:41:02
                                        E
                                ENTRIES
 SLOT         ACTUAL              IN
NUMBER        FACTOR      IND   SLOT                    HISTOGRAM
-----------------------------------------------------------------------------------
    0           0.01       <       0  |
    1           0.01       <       0  |
    2           0.01       <       0  |
    3           0.01       <       0  |
    4           0.02       <       0  |
    5           0.02       <       0  |
    6           0.03       <       0  |
    7           0.03       <       1  | *
    8           0.04       <       0  |
    9           0.05       <       0  |
   10           0.06       <       0  |
   11           0.07       <       1  | *
   12           0.08       <       0  |
   13           0.10       <       0  |
   14           0.12       <       0  |
   15           0.14       <       1  | *
   16           0.17       <       6  | ******
   17           0.20       <       4  | ****
   18           0.25       <       9  | *********
   19           0.30       -      13  | *************
   20           0.36       -       8  | ********
   21           0.43       -      11  | ***********
   22           0.52              35  | ***********************************
   23           0.63              24  | ************************
   24           0.76              35  | ***********************************
   25           0.91              30  | ******************************
   26           1.10              43  | *******************************************
   27           1.32              20  | ********************
   28           1.60              10  | **********
   29           1.93               5  | *****
   30           2.32       +      10  | **********
   31           2.80       +      20  | ********************
   32           3.37       +      23  | ***********************
   33           4.06       >      12  | ************
   34           4.90       >      12  | ************
   35           5.90       >      13  | *************
   36           7.12       >      14  | **************
   37           8.58       >       3  | ***
   38          10.35       >       0  |
   39          12.47       >       0  |
   40          15.04       >       2  | **
   41          18.13       >       0  |
   42          21.85       >       0  |
   43          26.34       >       0  |
   44          31.76       >       0  |
   45          38.28       >       1  | *
   46          46.15       >       0  |
   47          55.64       >       0  |
   48          67.07       >       0  |
   49          80.86       >       0  |
   50          97.48       >       0  |
   51         117.52       >       0  |
```

```
9 MAY 1988   09:01:36

               |                                             |  TOTAL
SYSTEM UNDER   |  VALID      NULL  UNRELIABLE                |  NUM OF
TEST           |  TIMES     TIMES     TIMES      OTHER       |  TESTS      PERCENT
------------------------------------------------------------------------------------
     A         |   750        11       224         91        |   1076      91.54%
     B         |   404         1        35        636        |   1076      40.89%
     C         |   795         4       271          6        |   1076      99.44%
     D         |   605         8         0        463        |   1076      56.97%
     E         |   366         0        29        681        |   1076      36.71%
```

ACEC DOCUMENTATION

- ACEC USER'S GUIDE
- ACEC READER'S GUIDE
- ACEC VERSION DESCRIPTION DOCUMENT

ACEC PHASE III
ADDITIONAL AREAS

1. ADDITIONAL EXECUTION TIME PERFORMANCE TESTS
2. DIAGNOSTIC MESSAGE EVALUATION
3. SINGLE SYSTEM REPORT
4. LIBRARY ROBUSTNESS EVALUATION
5. SYMBOLIC DEBUGGER PERFORMANCE EVALUATION

ACEC DISTRIBUTION
(HOW TO GET A COPY)

THE ACEC IS DISTRIBUTED BY:

DATA AND ANALYSIS CENTER FOR SOFTWARE (DACS)
RADC / COEE
BUILDING 101
GRIFFISS AFB, NY 13441-5700
ATTN: DOCUMENT ORDERING
(315) 336-0937

ADA COMPILER VENDORS USING ACEC FOR QUALITY ANALYSIS

TOTAL # OF VALIDATED ROOT COMPILERS = 210

VENDOR	# OF VALIDATED COMPILERS	%
VERDIX	47	22%
TELESOFT	39	19%
R&R SOFTWARE	11	5%
ALYSYS	8	4%
MERIDIAN	8	4%
RATIONAL	6	3%
GOULD	6	3%
IRVINE	4	2%
TLD	3	1%
DDCI	3	1%
INTERMETRICS	3	1%
DIGITAL EQUIPMENT	3	1%
CONCURRENT	3	1%
MIPS	1	.5%
	145	69%

ACEC USERS

USER	APPLICATION / FUNCTION
AF / McDONNEL ACFT CO.	ADVANCED TACTICAL FIGHTER
AF / MAC	F-15E AND F-15 MSIP COMPILER SELECTION
AF / GENERAL ELECT	FLIGHT CONTROLLER COMPILER SELECTION
AF / LEAR	ADVANCED INTEGRATED CONTROLS AND AVIONICS FOR AIR SUPERIORITY (ICAAS)
WRDC / AAAF-3	USABILTY OF ADA FOR EMBEDDED SYSTEMS
ARMY	INFORMATION SYSTEMS SOFTWARE CENTER COMPILER SELECTION
ARMY / SY TECH.	STRATEGIC DEFENSE COMMAND HIGH ENDOATMOSPHERIC DEFENSE INTERCEPTOR, XTV PHASE
NAVY	COUNTER MEASURES EVALUATOR SYSTEM COMPILER SELECTION
NAVY / SYSCON	IV&V OF THE ALS / N
MARINES	MARINE CORE TACTICAL SYSTEMS SUPPORT COMPILER SELECTION
DOD / NATO / BENDIX	MARK XV IFF COMPILER SELECTION
NASA / LOCKHEED	SPACE STATION FREEDOM COMPILER SELECTION
NASA / ROCKWELL	PERFORM REAL - TIME PERFORMANCE ASSESSMENT OF ADA

MCS 4 9 '01 [illegible]

APPLICATION OF ACEC TECHNOLOGY

"THE DIRECTIVE"

"HOUSE APPROPRIATIONS COMMITTEE REPORT (100-681), JUNE 10, 1988 DIRECTS Ada JOINT PROGRAM OFFICE TO "INCLUDE COMPILER EFFICIENCY IN VALIDATING PROCEDURES"

[illegible]

APPLICATION OF ACEC TECHNOLOGY

"THE RATIONALE"

"Ada HAS SOME TECHNICAL LIMITATIONS IN REALTIME AND DISTRIBUTED ENVIRONMENTS. PROJECT MANAGERS WORKING IN THESE ENVIRONMENTS WOULD BENEFIT FROM FINDINGS IN THESE AREAS."

[illegible]

APPLICATION OF ACEC TECHNOLOGY

"THE REPLY"

"ALTHOUGH BOTH EVALUATION AND VALIDATION ARE NECESSARY SEPARATION IS REQUIRED DUE TO THE NATURE OF THE TWO PROCESSES"

- VALIDATION TEST SUITE IS APPLICATION INDEPENDENT
- EVALUATION IS SUBJECTIVE AND APPLICATION SPECIFIC

"AJPO WILL ESTABLISH PROCEDURES AND GUIDELINES FOR FORMAL EVALUATION OF COMPILES AT EXISTING Ada VALIDATION FACILITIES BY MID-1989"

[illegible]

APPLICATION OF ACEC TECHNOLOGY

"THE PLAN"

Ada COMPILER EVALUATION PROCEDURES ~~AND GUIDELINES~~

- DEFINES COMMONLY USED FORMAL EVALUATION TERMS
- OUTLINES ORGANIZATIONAL STRUCTURE FOR MANAGEMENT, COORDINATION AND DIRECTION OF THE FORMAL EVALUATION PROCESS
- LISTS STEPS IN THE PROCESS
- PREVIOUS GUIDANCE TO DoD PROGRAM MANAGERS ON THE APPLICATION OF PERFORMANCE DATA IN THE ACQUISITION, USE AND MAINTENANCE OF Ada IMPLEMENTATIONS

MCS 18954

QUALITY TESTING SERVICE (QTS) -- 1

PURPOSE: "THIS SERVICE PROVIDES FOR THE COLLECTION OF COMPILER PERFORMANCE AND USABILITY DATA FOR ANALYSIS BY USERS AND PROGRAM MANAGERS IN EVALUATING Ada COMPILERS AND THE IMPLEMENTATION OF SPECIFIC LANGUAGE FEATURES." -- PTS, (VERSION I)

DOCUMENT DEFINES THE PROCEDURES FOR EFFECTIVE USE OF THE EVALUATION TECHNOLOGY BY THE DoD

- PAST $\rightarrow$ INITIAL DRAFT REVIEWED BY THE E&V TEAM AND OTHERS
- PRESENT $\rightarrow$ VERSION 1.0 IN REVIEW BY SELECTED EXPERTS
- FUTURE $\rightarrow$ APPROVAL OF PROCEDURES WILL RESULT IN AVAILABILITY OF STANDARDIZED EVALUATION CAPABILITY

MCS 10[illegible]

QUALITY TESTING SERVICE (QTS) -- 2

Ada COMPILER QTS PROCEDURES
(DRAFT)

- INTRODUCTION
- GLOSSARY
- ORGANIZATION AND RESPONSIBILITIES
- QTS TEST SET
- TESTING PROCEDURES
- PERFORMANCE AND USABILITY DATABASE

MCS 10856

QUALITY TESTING SERVICE (QTS) -- 3

QUESTIONS ABOUT THE QUALITY TESTING SERVICE (QTS):

- HOW VERSATILE IS THE ACEC ?
- WHO WILL OPERATE THE QTS ?
- HOW MANY INSTANCES OF THE QTS FACILITY WILL BE CREATED ?
- WHAT LEVEL OF EXPERTISE IS REQUIRED TO OPERATE THE QTS ?
- WHO WILL PAY FOR THE QTS ?

MCS 109

QUALITY TESTING SERVICE (QTS) -- 4

QUESTIONS ABOUT THE QUALITY TESTING SERVICE (QTS):

- WHAT IS THE BENEFIT TO THE GOVERNMENT ?
- WHAT IS THE ADVANTAGE TO THE VENDOR ?
- SHOULD THE VENDOR BE ALLOWED TO OMIT SOME TESTS ?
  - DOES THIS INVALIDATE THE RESULT ?
  - DOES IT PRODUCE MORE USEFUL INFORMATION ?

MCS [illegible]

E&V

APPLICATION OF ACEC TECHNOLOGY

"THE DREAM"

- AEF's EVALUATE Ada COMPILATION SYSTEMS (ACS) AND STORE DATA
- DoD PROGRAM OFFICES USE AEF DATA FOR ACQUISITION OF ACS's
  - SPECIFY ACS REQUIREMENTS
  - SPECIFY USE OF ACEC IN CONTRACTOR'S ACS SELECTION PROCESS

[illegible]

APPLICATION OF ACEC TECHNOLOGY

"THE GUIDANCE"

WHEN SELECTING AN ACS THE FOLLOWING SHOULD BE CONSIDERED:

- VALIDATION STATUS
- CODE SIZE
- EXECUTION SPEED
- COST
- COMPILATION SPEED
- EFFICIENCY OF Ada LIBRARY MANAGEMENT SYSTEM
- QUALITY OF DIAGNOSTICS INFORMATION
- AVAILABILITY / QUALITY OF DEBUGGERS AND OTHER SUPPORT TOOLS

APPLICATION OF ACEC TECHNOLOGY

"THE GUIDANCE (CONT'D)

STEPS FOR MAKING THE CHOICE

- INITIALLY USE "HIGH LEVEL" CHARACTERISTICS
- DETERMINE RELATIVE IMPORTANCE OF "LOW LEVEL" CHARACTERISTICS
- OBTAIN ACEC RESULTS FOR ALL CANDIDATES
- APPLY THE ACEC TO THE MOST PROMISING CANDIDATE CONFIGURATIONS
- IF LANGUAGE FEATURES ARE HEAVILY WEIGHTED USE APPENDIX V OF ACEC VDD. APPLY MEDIAN TO RESULTS

[illegible]

OVERVIEW

- E&V Task: Background
- E&V Team
- E&V Reference System
- Ada Compiler Evaluation Capability (ACEC)
- ➡ CAIS Implementation Validation Capability
  - What is CAIS
  - Who is using CAIS
  - What is CAIS IVC?
  - Application to the Acquisition Process
- Conclusions

MCS 49512

WHAT IS CAIS

C - COMMON
A - APSE (Ada PROGRAMMING SUPPORT ENVIRONMENT)
I - INTERFACE
S - SET

- CAIS IS A DOCUMENT WHICH:
  - DEFINES Ada PACKAGE SPECIFICATIONS FOR INTERFACES TO OPERATING SYSTEM SERVICES THAT SIGNIFICANTLY IMPACT TOOL TRANSPORTABILITY

CAIS IMPLEMENTATION VALIDATION CAPABILITY (CIVC)

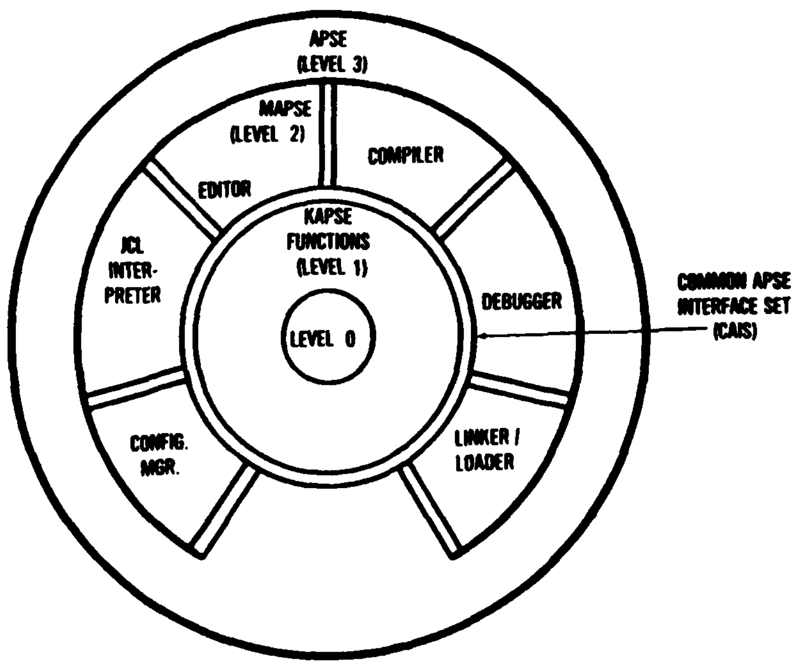
TECHNICAL OBJECTIVES

- DEVELOP A LIMITED VALIDATION TEST SUITE FOR THE COMMON ADA PROGRAMMING SUPPORT ENVIRONMENT (APSE) INTERFACE SUITE (CAIS)
- ENABLE DOD TO TEST CONFORMANCE OF APSE's TO CAIS
  - ENSURE TRANSPORTABILITY OF SOFTWARE TOOLS
  - ENSURE INTEROPERABILITY OF APSE DATA BASES

APPROACHES

- INITIAL CIVC BASED UPON DOD-STD-1838
- FOLLOW-ON VERSION TO EVOLVE TO DOD-STD-1838A
- WORK CLOSELY WITH CAIS IMPLEMENTATION GROUPS
- INCORPORATE DEVELOPMENTS OF OTHER ORGANIZATION INTO CIVC

APSE DEFINITION



WHO IS USING CAIS

NATO SPECIAL APSE DEVELOPMENT

- 10 COUNTRIES JOINTLY DEVELOPING
- APSE TOOLS TO BE CAIS-A BASED
- U.S. TO DEVELOP CAIS-A IMPLEMENTATION
- U.S. RESPONSIBLE FOR DEVELOPING "LIMITED" VALIDATION SUITE

I 16387

APPLICATION TO THE ACQUISITION PROCESS

- **No current DoD mandate to use CAIS**
- **Current environment research emphasizing CAIS**
- **CIVC/CAIS implementations analogous to ACVC/Ada compilers**

OVERVIEW

- E&V Task: Background
- E&V Team
- E&V Reference System
- Ada Compiler Evaluation Capability (ACEC)
- CAIS Implementation Validation Capability (CIVC)
- ➡ Conclusions

CONCLUSIONS

The software acquisition process is changing . . .

- "Word of Mouth" assessment unacceptable
- Cost of modern systems demands application of E&V technology of APSEs

MCS 10-9-47

COMPILER EVALUATION TECHNOLOGY PANEL

TRI-Ada CONFERENCE
PITTSBURGH, PA
25 OCTOBER 1989

PRESENTER: RAYMOND SZYMANSKI

MCS 10-9-48

E&V ACTIVITY AND TASK

EVALUATION AND VALIDATION OF Ada PROGRAMMING SUPPORT ENVIRONMENTS (a.k.a. E&V TASK)

SPONSOR ⟶ Ada JOINT PROGRAM OFFICE (AJPO)

CHARTER ⟶ DEVELOP THE TECHNOLOGY TO ASSESS APSE COMPONENTS

PROGRAM MANAGER ⟶ RAYMOND SZYMANSKI

PRODUCTS ⟶ Ada COMPILER EVALUATION CAPABILITY
CAIS IMPLEMENTATION VALIDATION CAPABILITY
E&V REFERENCE SYSTEM
E&V TASK ANNUAL REPORTS (VOL I-IV)

MCS 10-9-40

IN THE BEGINNING ...

OCT 1983 → DoD RECOGNIZES THE NEED FOR APSE EVALUATION AND INITIATES THE E&V TASKS

1985 → E&V TASK INITIATIVES EVALUATION TECHNOLOGY CONTRACTUAL ACTIVITIES

FEB 1987 → ACEC CONTRACT AWARD

JUNE 1988 → HAC (DIRECTED AJPO TO ...)

AUG 1988 → ACEC, VERSION 1.0 RELEASED

OCTOBER 1989 → DRAFT VERSION OF THE QUALITY TESTING SERVICE (QTS) PROCEDURES DOCUMENT DISTRIBUTED TO SELECTED REVIEWERS

DEC 1989 → (PLANNED) ACEC, VERSION 2.0 RELEASED

MCS 10 9 80

ACEC - STATUS

VERSION 1.0 → INITIAL RELEASE DATE - AUG 1988

CONTENTS - 1000+ TESTS THAT MEASURE COMPILER PERFORMANCE CHARACTERISTICS

VERSION 2.0 → INITIAL RELEASE DATE - DEC 1989

CONTENTS - LIBRARY ROBUSTNESS, DEBUGGER EVALUATION, DIAGNOSTIC MESSAGE EVALUATION, ADDITIONAL PERFORMANCE TESTS

ACQUISITION → DATA ANALYSIS CENTER (DACS)
PHONE # (315) 336-0937
(REQUEST THE "ACEC INFORMATION PACKET")

DEVELOPER → BOEING MILITARY AIRPLANE COMPANY
WICHITA, KANSAS

MCS 10-8-54

QUALITY TESTING SERVICE (QTS) -- 1

PURPOSE: "THIS SERVICE PROVIDES FOR THE COLLECTION OF COMPILER PERFORMANCE AND USABILITY DATA FOR ANALYSIS BY USERS AND PROGRAM MANAGERS IN EVALUATING Ada COMPILERS AND THE IMPLEMENTATION OF SPECIFIC LANGUAGE FEATURES." - PTS, (VERSION I)

DOCUMENT DEFINES THE PROCEDURES FOR EFFECTIVE USE OF THE EVALUATION TECHNOLOGY BY THE DoD

- PAST ⟶ INITIAL DRAFT REVIEWED BY THE E&V TEAM AND OTHERS
- PRESENT ⟶ VERSION 1.0 IN REVIEW BY SELECTED EXPERTS
- FUTURE ⟶ APPROVAL OF PROCEDURES WILL RESULT IN AVAILABILITY OF STANDARDIZED EVALUATION CAPABILITY

MCS 10-8-170

QUALITY TESTING SERVICE (QTS) -- 2

Ada COMPILER QTS PROCEDURES (DRAFT)

- INTRODUCTION
- GLOSSARY
- ORGANIZATION AND RESPONSIBILITIES
- QTS TEST SET
- TESTING PROCEDURES
- PERFORMANCE AND USABILITY DATABASE

MCS 10-8-56

QUALITY TESTING SERVICE (QTS) - 3

QUESTIONS ABOUT THE QUALITY TESTING SERVICE (QTS):

- HOW VERSATILE IS THE ACEC ?
- WHO WILL OPERATE THE QTS ?
- HOW MANY INSTANCES OF THE QTS FACILITY WILL BE CREATED ?
- WHAT LEVEL OF EXPERTISE IS REQUIRED TO OPERATE THE QTS ?
- WHO WILL PAY FOR THE QTS ?

MCS 10-8-57

QUALITY TESTING SERVICE (QTS) - 4

QUESTIONS ABOUT THE QUALITY TESTING SERVICE (QTS):

- WHAT IS THE BENEFIT TO THE GOVERNMENT ?
- WHAT IS THE ADVANTAGE TO THE VENDOR ?
- SHOULD THE VENDOR BE ALLOWED TO OMIT SOME TESTS ?
  - DOES THIS INVALIDATE THE RESULT ?
  - DOES IT PRODUCE MORE USEFUL INFORMATION ?

MCS 10-9-218

QUALITY TESTING SERVICE (QTS) - 5

CONCLUSION ⟶ FOR AVAILABILITY REGARDING THE PTS DOCUMENT, CONTACT.

DALE LANGE
ASD / SCEL
WPAFB, OHIO 45433

MCS 10-8-98

QUESTIONS FOR THE PANEL

ALL QUESTIONS SHOULD BE ADDRESSED IN THE CONTEXT OF "BARE MACHINES" ⟶

- WHERE ARE THE "QUALITY" COMPILERS?
- WHAT DO YOU THINK OF THE ACEC?
- WHAT ARE THE BENEFITS OF "FORMAL" COMPILER EVALUATION?

Text of presentation delivered by Raymond Szymanski, E&V Project Manager, at the Tri-Ada Conference, Compiler Evaluation Technology Panel, 25 October 1989.

(TITLE SLIDE, p. F-39)

Good Afternoon,

I'm Raymond Szymanski from the Wright Research and Development Center at Wright-Patterson Air Force Base in Dayton, Ohio. The projects that will be discussed here this afternoon include the Evaluation and Validation Task, the Ada Compiler Evaluation Capability, a.k.a. the ACEC, and the Ada Compiler Quality Testing Service, a.k.a. the QTS.

(E&V ACTIVITY AND TASK, p. F-39)

In October of 1983 the United States Department of Defense recognized the need for (APSE) evaluation technology by initiating the APSE Evaluation and Validation Task. This task is sponsored by the Ada Joint Program Office. The purpose of this task is to provide a DoD focal point for addressing DoD E&V technology needs. The E&V Task is responsible for 1) identifying and defining specific E&V technology requirements, 2) developing selected elements of the required technology, 3) collecting information on E&V technology, and 4) making E&V technology developments and information available to Government agencies, industry, and academia. The Program Manager of the E&V Task since 1985 is yours truly. Some of our current efforts include the ACEC, the CAIS Implementation Validation Capability, the E&V Reference System, and the E&V Team Public Report. For additional information on these items please obtain the E&V flyers that are available throughout the conference.

(IN THE BEGINNING, p. F-40)

The purpose of this foil is to provide some historical information which may help explain the presence of this panel today.

So, in 1983 the E&V Task is established.

In 1985, the first of three major contractual efforts was initiated. This first effort, which was awarded to The Analytic Sciences Corporation, has produced the E&V Reference System. This system consists of two coordinated documents, the E&V Reference Manual and the E&V Guidebook, for which Version 2.0 will soon be available.

In February of 1987, a contract was awarded to Boeing Military Airplane Corporation to produce a suite of Ada compiler performance tests and analysis support software.

In June of 1988, the House Appropriations Committee, the HAC, via HAC Report 100-681, directed the AJPO to "include compiler efficiency in validating procedures." The AJPO's response was that validation of compilers and evaluation of compilers were two tasks which should remain separate and that a test suite, independent of the validation suite, and known as the ACEC, was under development. They went on further to say that they, the AJPO, would develop a plan which would address the HAC's concern over compiler evaluation.

That plan is known today as the "Ada Compiler Quality Testing Service Procedures," the QTS Procedures.

This document was recently mailed to a list of professionals in the Ada community for review and comment. This list included, but was not limited to selected members of the following groups and organizations: this panel, the Ada Joint Users Group, the SEI, the STARS Program, major DoD program offices, the E&V Team, and over three dozen vendors of Ada compilers. Comments for this document are due on 17 November. A review and analysis of these comments will provide a basis for updating the draft and will result in an approved plan in the near future.

The final bullet on this slide announces the fact that ACEC Version 2.0 is scheduled to be accepted by the Government in December of this year and will be available for distribution shortly thereafter.

If you have not already guessed by now, the ACEC is scheduled to play a major role in the Ada Compiler Quality Testing Service.

In summary, a review of the events on this chart and their dates should make it obvious that the need for the ACEC was recognized long before the June 1988 HAC report.

(ACEC STATUS, p. F-40)

By now I'm sure many of you are asking "Just what is this ACEC?" Good.

The ACEC consists of the ACEC software product and three supporting documents; the ACEC User's Guide, the ACEC Reader's Guide, and the ACEC Version Description Document. The ACEC software product consists of both operational software and support software.

The operational software is a suite of 1,076 performance test programs which makes it possible to 1) compare the performance of several Ada compiler implementations, 2) isolate the strong and weak points of a specific system relative to other systems which have been tested, 3) determine what significant changes were made between releases of a compilation system, and 4) predict performance of alternate coding styles.

The ACEC tests provide assistance in measuring execution time efficiency, code size efficiency, and compiler time efficiency.

The support software consists of a set of tools and procedures which assist in preparing the test suite for compilation, in extracting data from the results of executing the test suite, and in analyzing the performance measurements obtained.

Version 2.0 of the ACEC is scheduled for release in December of this year. Along with an additional 300 performance tests, Version 2.0 will address several aspects of an Ada compilation system which usability, the Ada Library System, the Symbolic Debugger, and the Diagnostic Error Messages generated by the system will be tested.

The Diagnostic Message Assessor will include a set of erroneous programs which will trigger a variety of compiler, linker, and runtime error messages and warnings. The user will evaluate the diagnostic messages according to instructions included in the ACEC.

The Library System Assessor will consist of a set of programs and a set of scenarios which the user will perform using the programs. The user will follow the instructions included with the scenarios to evaluate both the functional capabilities and the performance of the Ada Library System.

Similarly, the Symbolic Debugger Assessor will include a set of programs, a set of scenarios describing operations to be performed with the symbolic debugger, and instructions for evaluating the performance of the debugger. Both the functional capabilities and the performance impact of the debugger will be tested.

In addition to the new performance tests and the introduction of usability tests, Version 2.0 will include a single system analysis tool. This tool will analyze the test results of one system and produce a report detailing system behavior.

The ACEC is currently being distributed by the Data Analysis Center for Software. The phone number is: area code 315, 336-0937. The caller should request the "ACEC Ordering Information Packet."

(QUALITY TESTING SERVICE --- 1, p. F-41)

In response to the stimulus provided by the June 1988 HAC Report, the AJPO has developed a draft document titled "Ada Compiler Quality Testing Service Procedures." The AJPO is currently giving consideration to establishing an Ada Compiler Performance Testing Service. This service would provide for the collection of compiler performance and usability data for analysis by users in evaluating Ada compilers.

The QTS procedures document defines the procedures for effective DoD use of evaluation technology, such as the ACEC. The initial draft of the document was reviewed by members of the E&V Team and a few other individuals. The current version is out for review by a considerably larger audience. When completed and approved, the QTS procedures will result in the availability of a standardized evaluation capability.

(QUALITY TESTING SERVICE --- 2, p. F-41)

The current draft of the QTS procedures document contains six major sections.

The obligatory introduction addresses the rationale behind the QTS, the goals of the QTS, and a concept of operation.

The glossary simply defines the terms used within the document.

The section on organization and responsibilities describes the roles and responsibilities of the organizations involved in managing, supporting, and executing the QTS and identifies potential QTS customers.

The QTS Test Set Section discusses the purpose of the test set, the initial test set composition, the criteria that additional test set components must meet, configuration management and software maintenance of the test set, customization of the test set and its availability.

The section on testing procedures covers numerous topical areas. It includes, but is not limited to, discussions of the following:

The projected testing schedule---from pre-payment for services to completion of the final report, the customer agreement---and what topics are addressed by it, pretesting activities, submission of results, what is a test issue?, what happens at on-site testing?, analysis of testing results---what support the QTF will provide in analysis, the compiler capability report---its contents and availability.

The last section addresses the performance and usability database---what it will contain and who has access to it.

(QUALITY TESTING SERVICE --- 3, p. F-42)

Shortly after accepting the invitation to participate on this panel I was informed by a reliable source that the topics for the Tri-Ada panels were selected not only for their potential to inform, but also for their potential of being highly controversial. To that end the next three foils contain several questions concerning the ACEC and the Quality Testing Service. The first two foils contain questions that have evolved during the creation of the QTS procedures document. The final foil lists those questions posed to each panelists upon their invitation to participate. Although I'm sure that some of the panelists will disagree with my responses and thus allow this panel to fulfill our pre-ordained destiny of controversy, I've also included questions that I do not yet have answers.

Question #1	---	How versatile is the ACEC?
Answer	---	Sufficiently versatile to contribute to the objectives of a Quality Testing Service.

Question #2 --- Who will operate the QTS?

Answer --- That has not yet been determined. However, several organizations have expressed interest in becoming quality testing facilities if and when the AJPO decides to institute QTFs.

Question #3 --- How many instances of the QTS Facility will be created?

Answer --- That has not yet been determined, although I suspect there is room for more than one.

Question #4 --- What level of expertise is required to operate the QTS?

Answer --- Various levels of expertise are required depending on the particular function being performed. The higher levels of expertise being required in the analysis of test results, especially the anomalies.

Question #5 --- Who will pay for the QTS?

Answer --- The user of the service.

(QUALITY TESTING SERVICE --- 4, p. F-42)

Question #6 --- What is the benefit to the Government?

Answer --- Save money while fielding compilers that meet our needs. The QTS will provide valuable information that can be used to determine which compilers meet our program requirements and which of those candidates is best suited for the task. The importance of having the proper compilation system cannot be overstated when considering the costly consequences of having the wrong one.

At present there are many Government organizations doing their own compiler testing, either under contract or in-house. Each of these organizations has had to pay the costs of learning how to do successful evaluations and interpreting the results. With centralized testing the learning curve costs to the DoD are significantly reduced and the probability that the task was done correctly is significantly increased as the experience base grows.

With centralized testing and its centralized database of compiler quality testing results it is reasonable to expect that the DoD would perform less compilation system evaluations than they would without the database. This is possible for two reasons. First, there is currently no reliable method for a Government agency to determine which compilation systems have been evaluated by other Government agencies or Government contractors. As such, I suspect that the same compilation systems have been evaluated by different agencies. A duplication of effort which I contend is a waste of money and could be avoided via a central database. The second reason is that once the database is sufficiently populated it is reasonable to expect that many users will find information there useful in identifying candidate compilers that meet their needs. In this case they have identified testing and have saved a significant amount of time and money.

Question #7 --- What is the advantage to the vendor?

Answer --- Two part answer. Part one, the ACEC is a tool which can assist the vendor in improving his product. One prominent vendor informed me that they use it as a Q/A tool to measure the performance differences of new releases and have used it to track down bugs in mature releases.

Part two, if a vendor has the results of his compiler testing in the database it is possible that a database user will select his compiler as a candidate based on the information contained therein. Also, for the purpose of advertising, the certified testing results will be held in higher regard by the user/buyer community than those not double checked by an independent agency.

Question #8 --- Should the vendor be allowed to omit some tests?

Answer --- From the perspective of a buyer I wouldn't want to allow test omission because it may remove an important data point. However, there may be some good technical reasons for omitting particular tests and in that case it should be allowed. Talk about you controversy.

(QUALITY TESTING SERVICE --- 5, p. F-43)

For information regarding the performance testing service draft document please contact the individual listed on this foil.

(QUESTION FOR THE PANEL, p. F-43)

The following three question are to be answered by all the panelists. I suspect that those following me will answer them in more detail than I.

Question #1 --- Where are the quality compilers?

Answer --- We at the Avionics Laboratory have some very good ideas where the quality compilers are since we have been using the ACEC to test these with. However, we are not inclined at this to announce our findings to the world. Also, we, like many other Government organizations doing in-house evaluations, are not chartered to do this type of work forever, and there will be much more of this work that needs to be done. Just another reason why we need a chartered quality testing service to continue this important endeavor.

Question #2 --- What do you think of the ACEC?

Answer --- As the E&V Task Program Manager and the current ACEC Project Manager, my opinion is sure to be looked upon as slightly biased. So, with your permission, I would like to mention just a few of the documented uses of the ACEC instead.

The F-15 System Program Office at Wright-Patterson Air Force Base is responsible for updating the F-15 Eagle and the F-15 Multistage Improvement Program On-board Computers and Operational Flight Programs. As a guide their contractor used the E&V Reference System for information on the process of selecting an Ada compiler and used the ACEC to "provide significant information to allow thorough evaluation and intelligent compiler selection."

Another example, the ACEC was used to identify, for a major compiler vendor, what was causing the code of a new version to execute slower than code from an older version. Without the ACEC this anomaly may not have been discovered and corrected until after the compiler reached the marketplace.

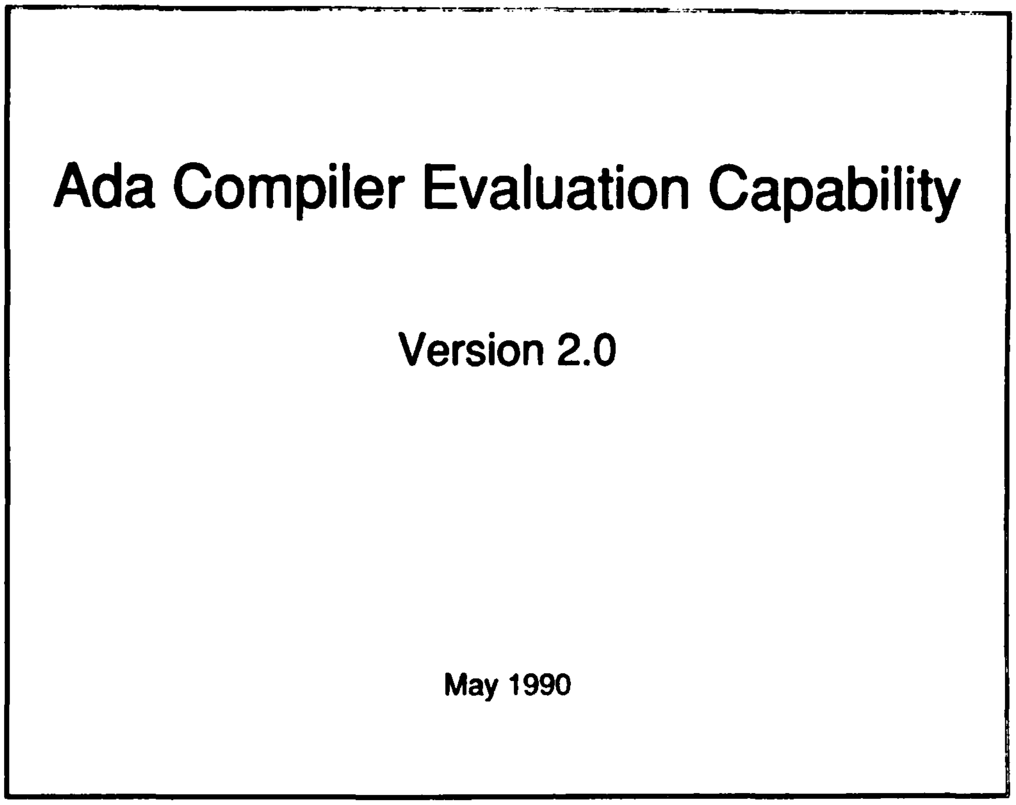
In another example, the ACEC was used at WRDC to discover that the program library for a particular compilation system continued to grow even after deleting the contents after each build, eventually filling a 20 meg disc. When a directory command was performed after the library was completely deleted, the disk still acted as if full. It was determined that the disk contained hidden files from the compiler.

The ACEC is currently being used by a NASA contractor to evaluate cross-compilers for the embedded real-time targets on the space station "Freedom." A paper given at the recent AIAA Conference indicates that this work is not yet completed and therefore I do not have a report card on their use of the ACEC. The point is that the ACEC is being used on this effort. In the paper they did say that use was made of the E&V Reference Manual and E&V Guidebook and that they were "excellent references."

Question #3 --- What are the benefits of "formal" compiler evaluation?

Answer --- Formal, as in institutionalized, not as in mathematical. Aside from the cost benefits to be realized from cradle to grave on a software project and simply having someone who knows how to do evaluations when they need to be done, there is the possibility that the QTS and its test suite will set a quality target for compiler vendors to strive for, thus elevating the quality of compilers across the board.

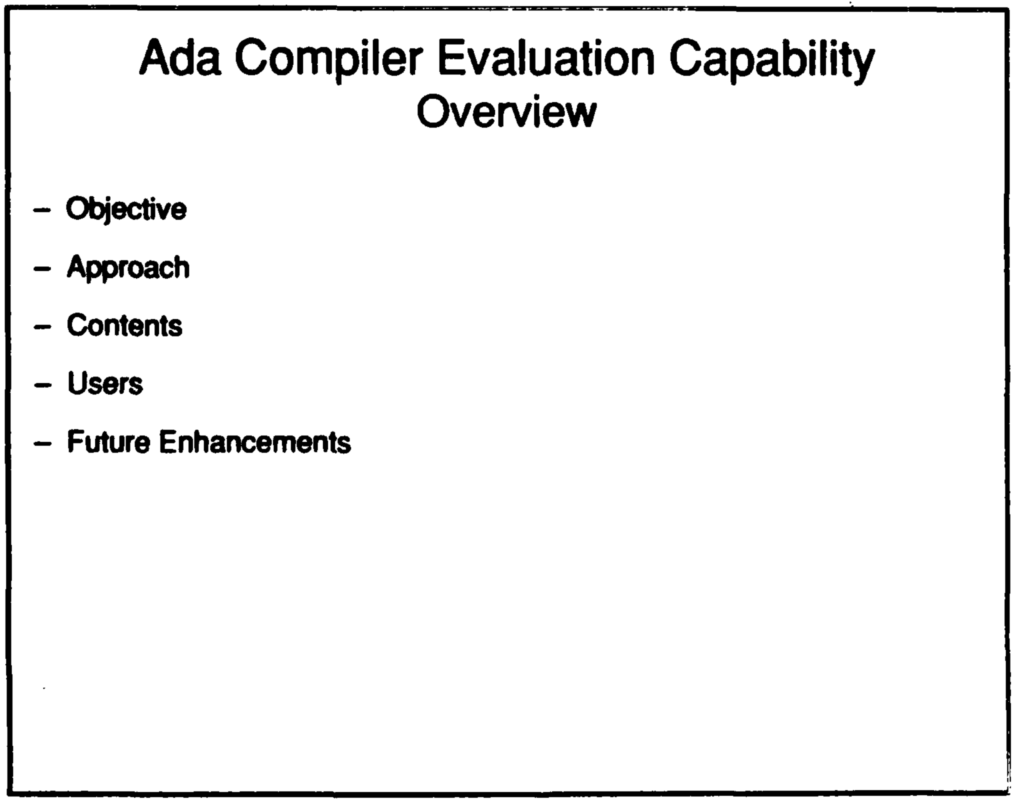
Thank you.



Ada Compiler Evaluation Capability

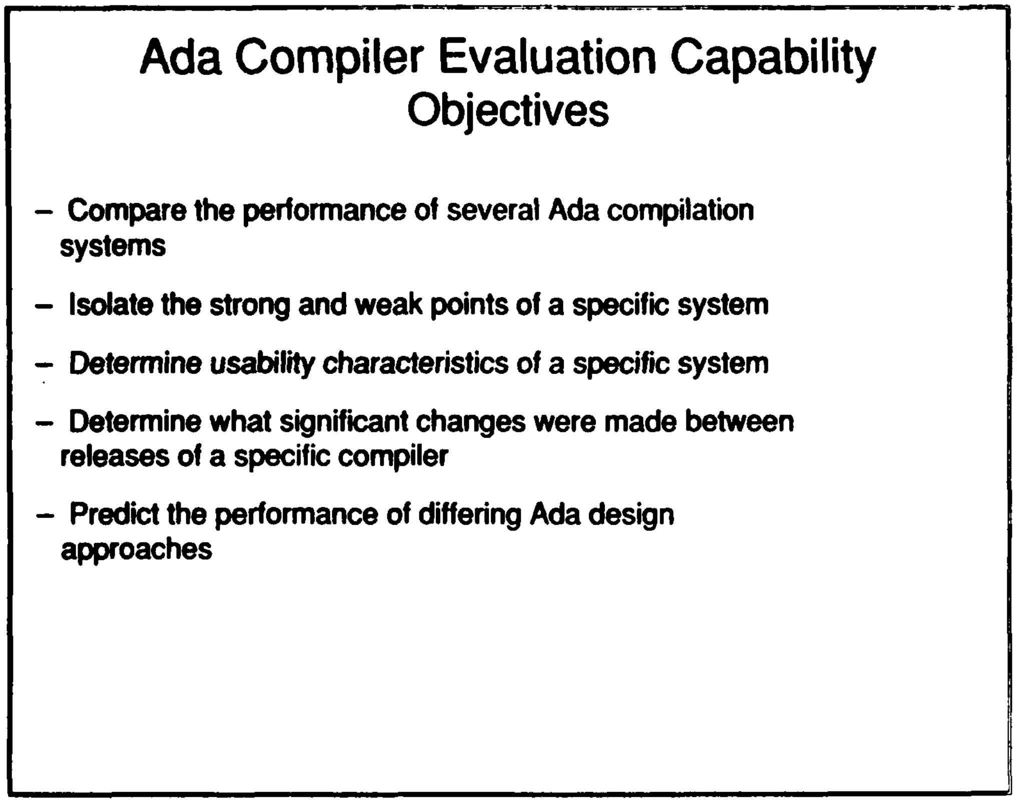
Version 2.0

May 1990



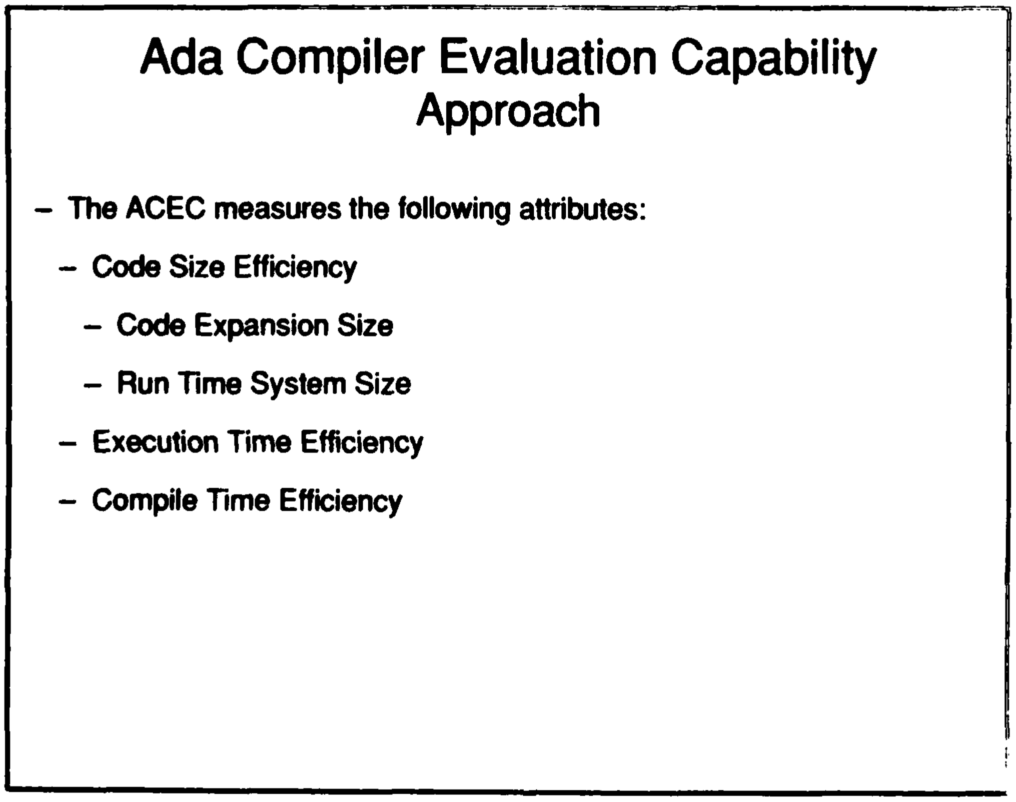
Ada Compiler Evaluation Capability Overview

- Objective
- Approach
- Contents
- Users
- Future Enhancements



Ada Compiler Evaluation Capability Objectives

- Compare the performance of several Ada compilation systems
- Isolate the strong and weak points of a specific system
- Determine usability characteristics of a specific system
- Determine what significant changes were made between releases of a specific compiler
- Predict the performance of differing Ada design approaches



Ada Compiler Evaluation Capability Approach

- The ACEC measures the following attributes:
  - Code Size Efficiency
    - Code Expansion Size
    - Run Time System Size
  - Execution Time Efficiency
  - Compile Time Efficiency

Ada Compiler Evaluation Capability
Approach (cont)

- The ACEC assists in the assessment of the following:
  - Diagnostic Messages
    - Clarity
    - Accuracy
  - Program Library System
    - Functional Capabilities
  - Symbolic Debugger
    - Functional Capabilities

Ada Compiler Evaluation Capability
Contents

- 1400 Performance Tests
- Usability Tests and Scenarios
- Support and Analysis Tools
- User Documentation

Ada Compiler Evaluation Capability
Performance Tests

- Individual Language Features
  - Required
  - Implementation Dependent
- Optimizations
- Performance Under Load
- Design Trade Offs
- Operating System Efficiency
- Application Profiles
  - Classical
  - Ada In Practice
  - Ideal Ada

Ada Compiler Evaluation Capability
Tests Of Mission Critical Significance

- Memory Management / Storage Reclamation
- Interrupt Handling
- Task Performance
- Run Time Checking
- Exception Handling / Propagation
- Bit Manipulation
- Floating Point Operations

Ada Compiler Evaluation Capability Assessors

- Sets of Tests and Scenarios to :
- Evaluate clarity and accuracy of system's diagnostic messages
- Determine wether the functional capabilities of a program library system are sufficient to accomplish a set of predefined scenarios
- Determine wether the functional capabilities of a symbolic debugger are sufficient to accomplish a predefined set of scenarios

Ada Compiler Evaluation Capability Analysis and Support Tools

- INCLUDE
  - Adapts programs to particular targets
- FORMAT and Med_Data_Constructor
  - Extract and format timing and sizing data
- Median
  - Compares results of performance tests
- SINGLE SYSTEM ANALYSIS
  - Compares results of related tests from a single system

Ada Compiler Evaluation Capability
Documentation

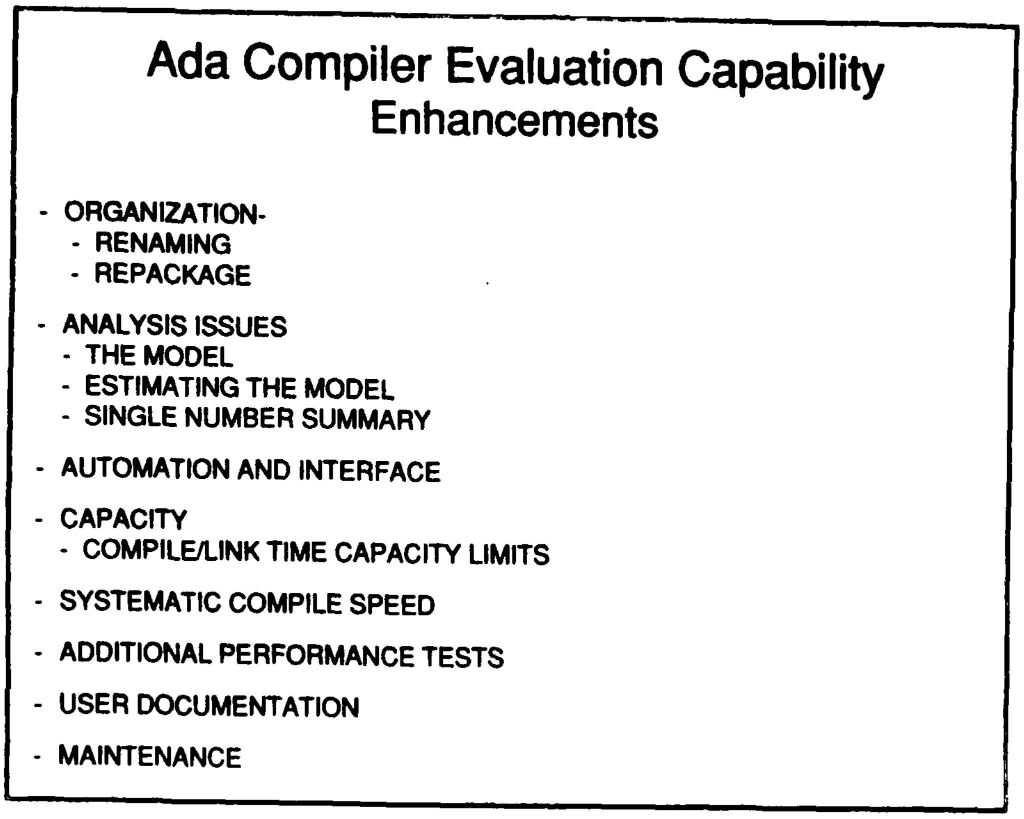
- ACEC User's Guide (109 pp)
  - How to adapt and execute the test suite
- ACEC Reader's Guide (153 pp)
  - Test Suite Organization
  - Interpretation of Results
- Version Description Document (295 pp)
  - Describes product as contained on distribution tape
  - Contains Indexes

Ada Compiler Evaluation Capability
Keyword Indexes

- Keyword Indexes Created to Enhance Usability
- Keyword Index 1
  - Primary Purpose -- 30 Categories
    - List of Qualified Tests
    - Lists of Secondary and Incidental Tests
  - Language Reference Manual Citations

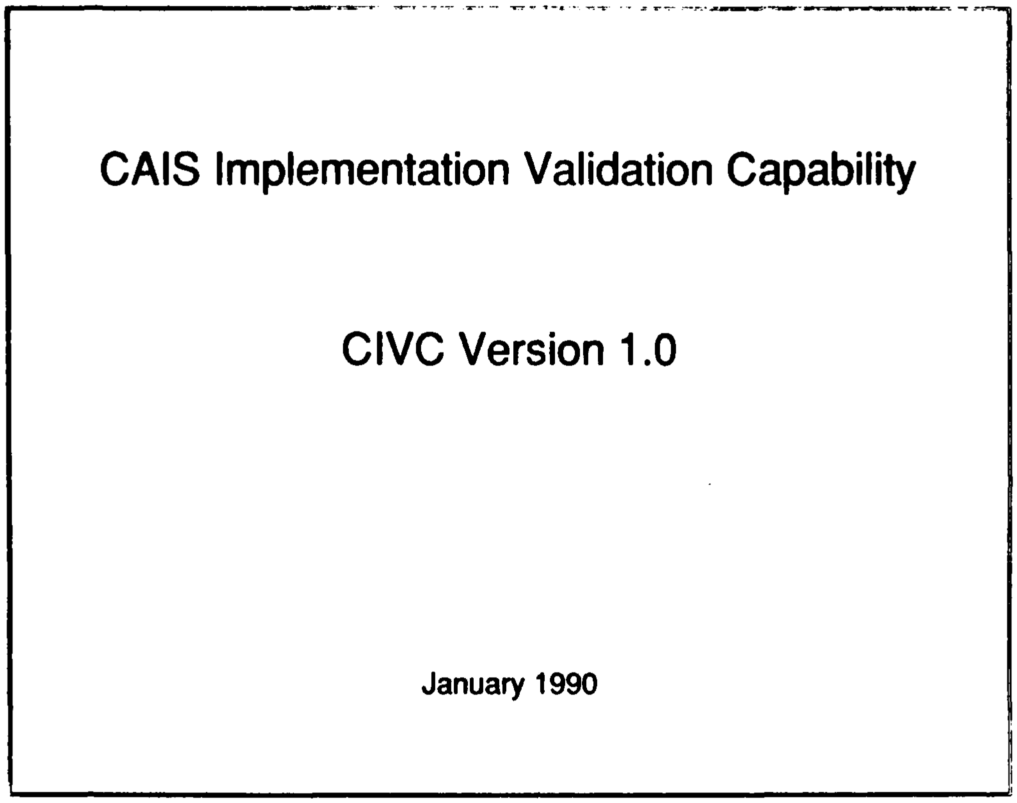
Example: subprogram.local 6.4

Primary: activation1, firth5, ss8, ...
Secondary: ss641, ss642
Incidental: ss236, ss237, ss365, ...



Ada Compiler Evaluation Capability Enhancements

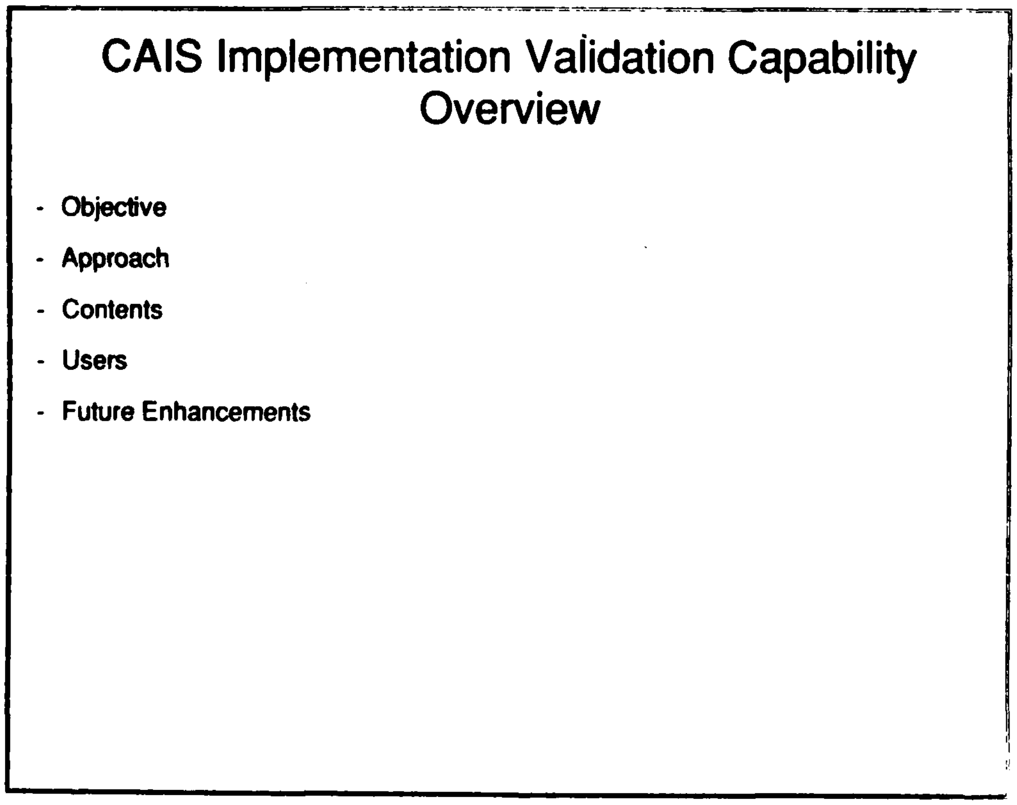
- ORGANIZATION-
  - RENAMING
  - REPACKAGE
- ANALYSIS ISSUES
  - THE MODEL
  - ESTIMATING THE MODEL
  - SINGLE NUMBER SUMMARY
- AUTOMATION AND INTERFACE
- CAPACITY
  - COMPILE/LINK TIME CAPACITY LIMITS
- SYSTEMATIC COMPILE SPEED
- ADDITIONAL PERFORMANCE TESTS
- USER DOCUMENTATION
- MAINTENANCE



CAIS Implementation Validation Capability

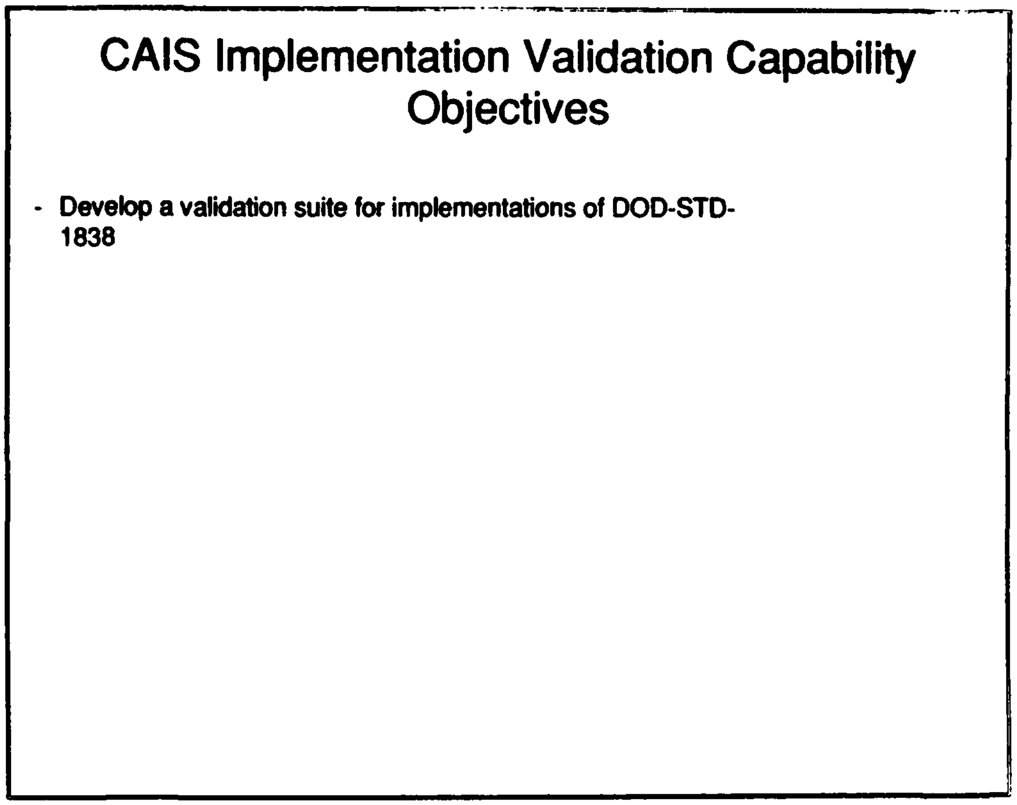
CIVC Version 1.0

January 1990



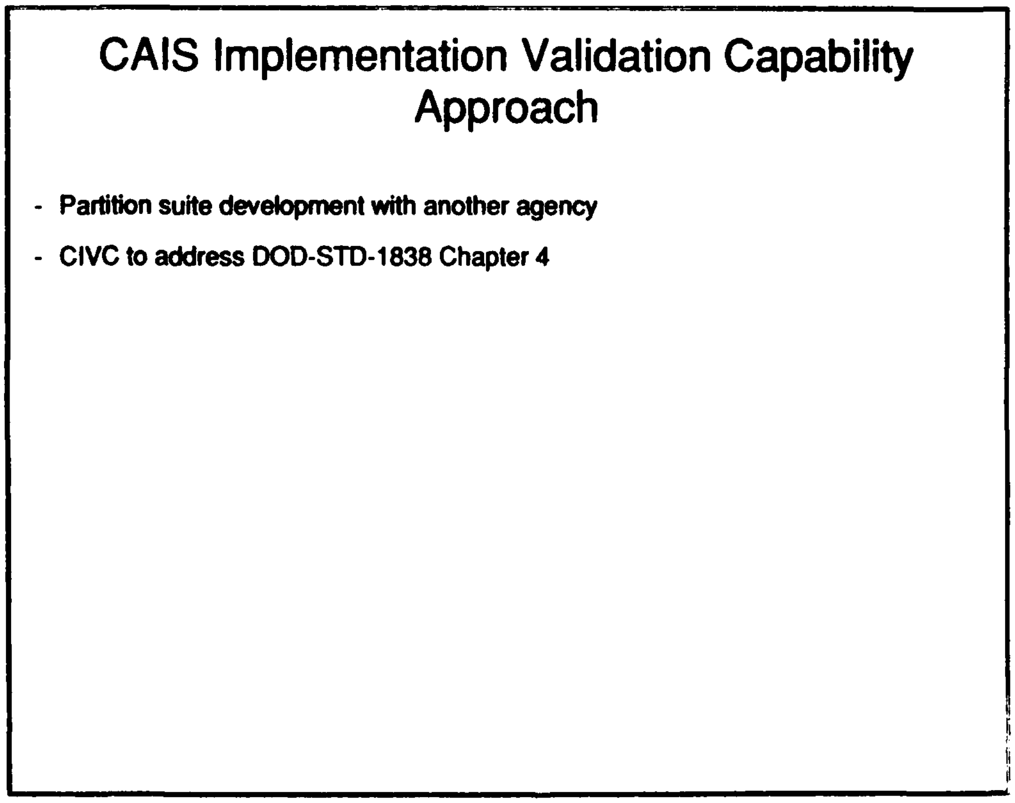
CAIS Implementation Validation Capability Overview

- Objective
- Approach
- Contents
- Users
- Future Enhancements



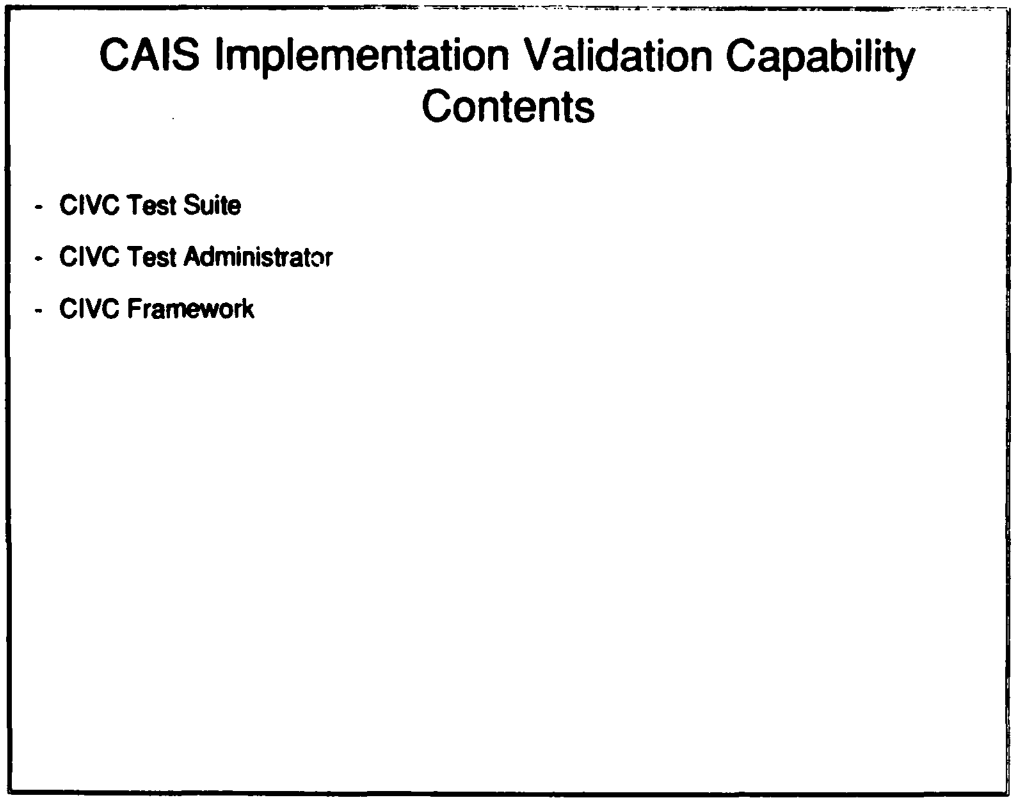
CAIS Implementation Validation Capability Objectives

- Develop a validation suite for implementations of DOD-STD-1838



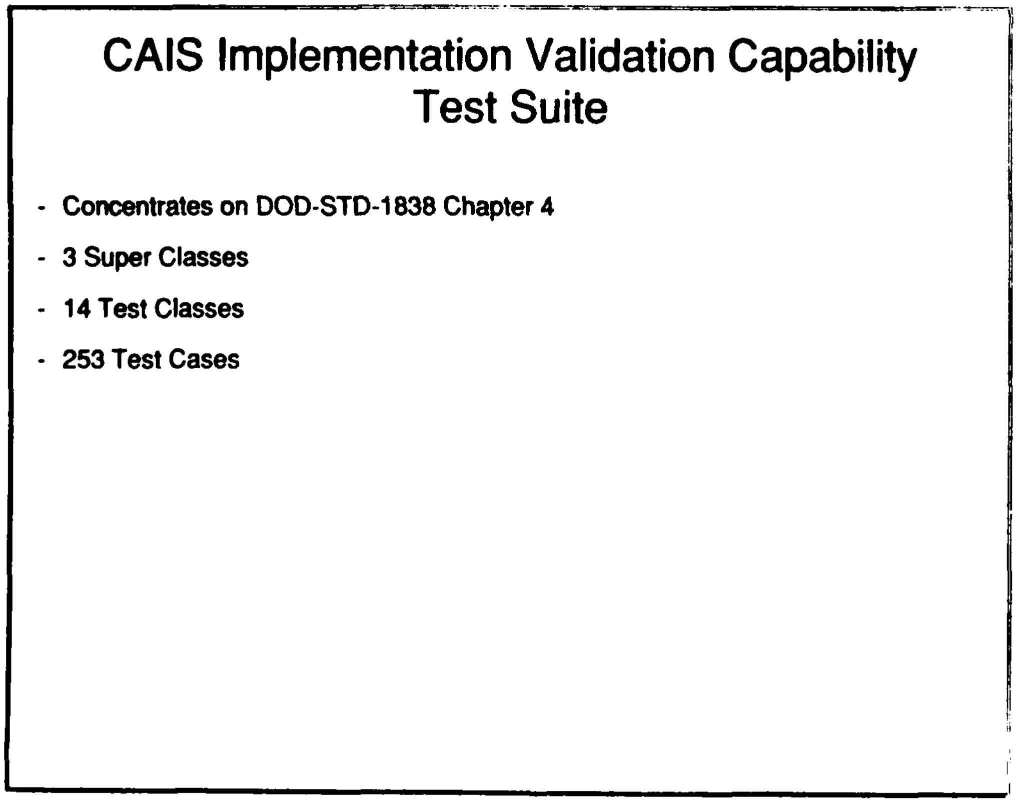
CAIS Implementation Validation Capability Approach

- Partition suite development with another agency
- CIVC to address DOD-STD-1838 Chapter 4



CAIS Implementation Validation Capability Contents

- CIVC Test Suite
- CIVC Test Administrator
- CIVC Framework



CAIS Implementation Validation Capability Test Suite

- Concentrates on DOD-STD-1838 Chapter 4
- 3 Super Classes
- 14 Test Classes
- 253 Test Cases

CAIS Implementation Validation Capability
CIVC Test Administrator

- Provides CIVC User Interface
- Encapsulates target environment dependencies for operating the CIVC
- Provides the mechanism for scheduling and executing the tests defined in the suite

CAIS Implementation Validation Capability
CIVC Framework

- Traces relationship between DOD-STD-1838, test objectives, scenarios, and the taxonomy
- Allows for understanding of DOD-STD-1838 interpretations and how implemented as test cases
- Implemented as on-line, hypertext information architecture

CAIS Implementation Validation Capability Documentation

- Test Suite Operators Guide
- Test Report Readers Guide
- Version Description Document
- Product Specification

CAIS Implementation Validation Capability Future Enhancements

- Develop validation suite for DOD-STD-1838A implementations
- Initial work will extract/update current CIVC suite for use on 1838A implementations
- Test selection criteria will guide further CIVC-A suite development

CAIS Implementation Validation Capability Users

- NATO Special APSE to contain 1838A Implmentation
- US Team to develop 1838A Implementation
- US IV&V Team to use CIVC and CIVC-A

E&V REFERENCE SYSTEM

Version 2.0

May 1990

E&V REFERENCE SYSTEM
Objective

THE E&V REFERENCE SYSTEM ALLOWS USERS TO

- GAIN AN UNDERSTANDING OF APSES AND APPROACHES TO THEIR ASSESSMENT
- FIND USEFUL INFORMATION -- TERMINOLOGY, DEFINITIONS, AND RELATIONSHIPS
- FIND ASSESSMENT CRITERIA/METRICS AND "POINTERS" TO SPECIFIC EVALUATION OR VALIDATION TECHNIQUES
- FIND DESCRIPTIONS OF EVALUATION OR VALIDATION TECHNIQUES
- FIND GUIDEANCE IN THE SELECTION, INTERPRETATION, AND INTEGRATION OF E&V TECHNIQUES AND RESULTS

THE E&V REFERENCE SYSTEM (1 OF 5)

Q. What is it?
A. Two Documents: Reference Manual(1) and Guidebook(2) (See Slide 2)

Q. Why is it Needed?
A₁. Importance of Decisions
A₂. Complex, New Technology
} (See Slide 3)

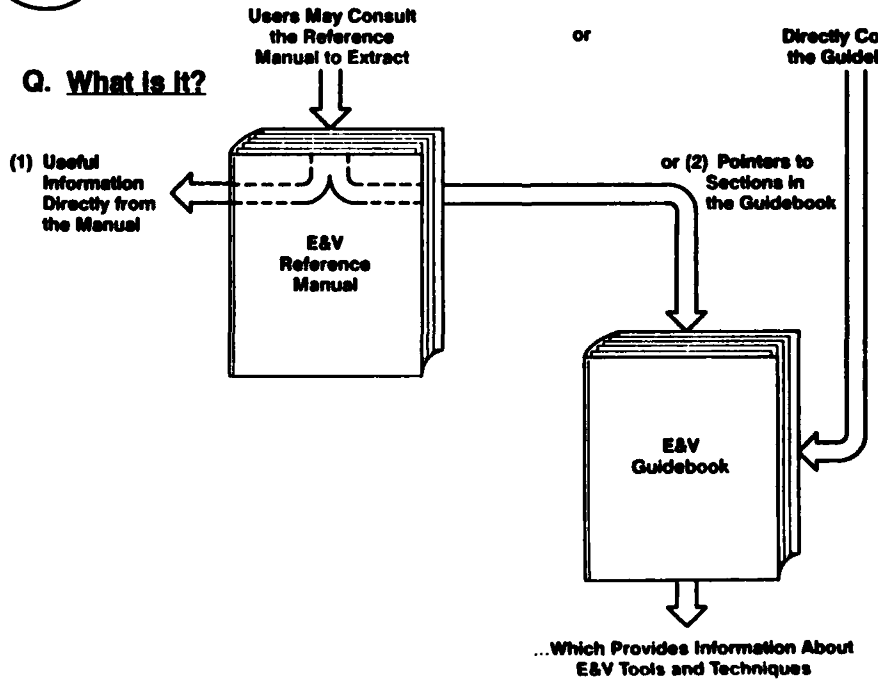
Q. How is it Used?
A. In Many Ways - (See Example, Slide 4)

Q. What is the Current Status of E&V Technology?
A. Much Exists; Much Still Needed -- (See Matrix, Slide 5)

(1) DTIC No. AD-A214 167
(2) DTIC No. AD-A214 166

THE E&V REFERENCE SYSTEM (2 OF 5)

Q. What is it?


Users May Consult the Reference Manual to Extract
or
Directly Consult the Guidebook
(1) Useful Information Directly from the Manual
or (2) Pointers to Sections in the Guidebook
E&V Reference Manual
E&V Guidebook
...Which Provides Information About E&V Tools and Techniques

THE E&V REFERENCE SYSTEM (3 OF 5)

Q. <u>Why Is It Needed?</u>

A_1. Importance of Decisions

- Large, Critical Ada-Based System to be Developed
- Quality and Cost of Systems Influenced by Environments
- ASPE and Tool Selections are Major Long-Term Investments

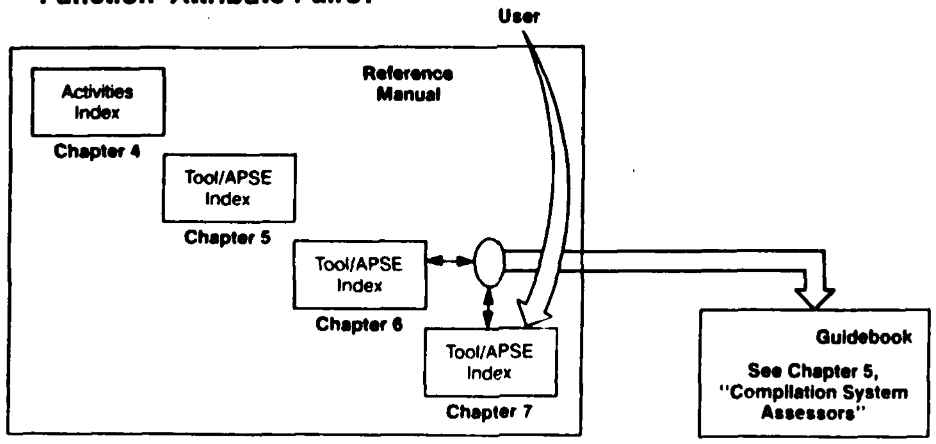
A_2. Technical Complexity

- Many Inter-Related Elements, Some with New Technology, Undergoing Rapid Change
- Diversity of Choices and Viewpoints
- Common Framework, Terminology, Defintions Needed

THE E&V REFERENCE SYSTEM (4 OF 5)

Q. <u>How Is It Used?</u>

A. Example: For the Function "Compilation", What Attributes are Important, and What Evaluation Techniques Apply to Relevant Function-Attribute Pairs?



E&V REFERENCE SYSTEM (5 OF 5)

Q. **What is the current status of E&V technology?**

Assessment Techniques

Assessment Subjects	Evaluation				Validation	
	Checklists etc	Test Suites	Structured Experiments	Decision Support Systems	Test Suites	Formal Methods
Compilation Systems	◑	◑			◕	
Target Code Generation Aids, etc.	◑					
Test Systems	◑		◷			
Tool/Host Interfaces	◷		◷		◷	
Requirements/Design Tools	◷		◷			
CM Support Tools	◑		◷			
Distributed System Dev. Tools	○					
Distributed APSEs	○					
Whole APSEs	◑		◷	◷		
Adaptation Features	◑					
Others	◔					

E&V Products
How to obtain

Ada Compiler Evaluation Capability

- Data Analysis Center for Software (DACS)
  - ACEC Ordering Information Package

E&V Reference System

- Defense Technical Information Center (DTIC)
  - E&V Reference Manual - # AD-A214 167
  - E&V Guidebook - # AD-A214 166

Ada Information Clearinghouse Bulletin Board <8DB,1SB,NP>

- (202) 694-0215 (301) 459-3865

DACS -- (315) 336-0937 DTIC -- (202) 274-7633

APPENDIX G

A Software Evaluation and Selection Framework

by
Major Patricia K. Lawlis
Air Force Institute of Technology

A key to the effective combination of the software evaluation and selection processes is a common evaluation framework. Without it, each evaluator uses different terminology and reports results in a different form, and each selector must either use only one evaluation result or else determine a way to convert all results into a common form. With it, a decision maker can readily collect any amount of evaluation data which seems appropriate, from any number of sources, and use it in the software selection process.

The evaluation framework outlined in this paper is intended as a pint of departure for further work in this area. It is by no means complete, but the important part is its organization. The framework is structured using two main concepts. First, there is a distinction between absolute and relative criteria, and second, the various criteria are organized in levels.

Many software characteristics must be specified in absolute terms. For example, what operating system must be used to run the product, what options are available with the product, what is the retail price of the product, etc. Although these are not normally the types of things thought of as software evaluation criteria, they are, nevertheless, important features of the software assessment. In an attempt to avoid confusion, these absolute characteristics of software are called **features**. In contrast to the features, the software characteristics which are assessed in relative terms, such as reliability, efficiency, etc., are called **criteria**.

Rather than attempting to consider all possible features and criteria at one level of abstraction in an assessment, the main categories of features and criteria are considered here as the highest level of abstraction. Then successive lower levels of abstraction are used to determine the details under each of these categories. The top level categories developed for both features and criteria are given in Figure 1. The feature categories have been put together as a composite of features identified in a number of sources [DoD 89, Firth 87, Foreman 87, Houghton 87, Lehman 87, Lyons 86, Weiderman 89]. The criteria categories, on the other hand, are fairly well agreed upon the area of software quality [Arthur 85, Bowen 85, DoD 89, Pressman 87]. The only one that has been added is vendor support, and this was deemed appropriate because it is often desirable to rate the quality of the support provided by the product vendor.

Figure 2 illustrates feature details filled in at the second level for some of the top level categories. These are a type of detailed features which either are or are not a part of a given product. The lists of software functions are general and may be applied to any software product. In Figures 3 and 4, many of the given feature details have numerical values or other values which may be enumerated. Default values for acceptability are given

Feature Categories	Criteria Categories
analysis functions	correctness
applied standards	efficiency
associated tool requirements	expandability
configuration requirements	integrity
contractual matters	interoperability
cost	maintainability
hardware control	reliability
management functions	reusability
numerics	survivability
options	transportability
security issues	usability
source code sizing	vendor support
timing requirements	verifiability
transformation functions	
user profile	

Figure 1 - Categories of features and criteria

analysis functions	applied standards
consistency checking	Ada (MIL-STD-1815A)
cross referencing	CAIS (MIL-STD-1838A)
data flow analysis	PCTE
mutation analysis	DIANA
regression testing	GKS
requirements simulation	PHIGS
statistical profiling	DOD-STD-2167A
traceability analysis	
management functions	**transformation functions**
configuration management	incremental compilation
cost management	editing
object management	formatting
performance monitoring	linking/loading
program library management	activities transformation
quality management	object transformation
resource management	program generation

Figure 2 - Detail features which are or are not present in a product

Feature	Default Value
configuration requirements	
host hardware	
target hardware	
host memory needed	<=4MB
host disk capacity needed	<=50MB
peripheral devices	
operating system	
support software	
distributed system	
contractual matters	
no restrictions on users	
number of users	
number of CPUs	
sale of derived software	
source code available	
support available	

user profile

Feature	Possible Values
skill level	novice (default), intermediate, expert
training	little or none (default) moderate, extensive

Figure 3 - General detail features with default values

Feature	Default Value
numerics	
bits in integer	>=16
max integer	>=32768
bits in float	>=32
bits in exponent	>=8
fixed point delta	<=0.0001
digits in float	>=8
long rep forms	
short rep forms	
source code sizing	
lines in unit	>=5000
units in compile	>=200
entries in task	>=20
elements in aggregate	>=100
discriminants in record	>=10
alternatives in case	>=25
alternatives in select	>=25
instantiations of generic	>=10
timing requirements	
compiling lines of code	>1000 lines/min
task rendezvous	<0.00001 sec
subprogram overhead	<0.00001 sec
exceptions overhead	<0.00001 sec
clock resolution	<0.000001 sec
max blocking time	<0.00002 sec

Figure 4 - Detail features with default values specific to Ada compilers

for each feature detail of this type. In Figure 3, the detail features are general, while the ones in Figure 4 are entirely specific to Ada compilers. Of course, for a different type of software product, each of the specific entries in these lists of feature details would have to change. It is entirely possible that in some instances both general and specific details could be applicable under one feature category. The detail feature lists in these figures are not exhaustive by any means, but they show how the lower levels can be organized.

In Figure 5, the second level of criteria is illustrated. Once again, the literature show a fair amount of agreement on these details. In some cases they are called metrics. Note that in many instances the same detailed criteria are listed for more than one category. Thus, although the feature categories are totally independent of one another, the crite‑ia categories are not.

There are many features and criteria which are important in almost every software selection. These should be a part of every evaluation performed. For many features, it is not so much a matter of evaluation as it is that information about the feature should be put in the evaluation report. The features which are almost always important involve such areas as product identification, configuration requirements, and contractual matters. These are listed in Figure 6. Many criteria should be seriously considered for evaluation and reporting by every individual and organization which provides evaluation data. In many ways, the evaluation data is not so easy to gather for these criteria as it is for the features, but it is every bit as important. These criteria of importance are given in Figure 7.

There is currently no general agreement on either the terminology of the definitions of the terms used in software evaluation, so a glossary of the terms as used here is provided at the end of this paper. Lower levels of detail can be filled in more completely with subsequent work in this area. The appeal of such an organization of features and criteria is that it can be very flexible, accommodating new ideas and new technology concepts as they arise, but at the same time the basic framework remains stable.

This paper has presented a framework which can be used for any type of software evaluation and selection scenario. This framework is a first step toward solving the problem of consistent reporting of evaluation data. furthermore, it provides a basis for developing a decision support system (DSS) which can then be used in the software selection process. Hopefully, it will also provide a basis for a common understanding of the evaluation and selection processes. This will make it possible for decision makers to insist on getting both complete and consistent data on which to base their decisions. Until this occurs, it is no wonder that the software crisis still exists.

correctness
- completeness
- consistency
- traceability

efficiency
- communication effectiveness
- processing effectiveness
- storage effectiveness

expandability
- augmentability
- generality
- modularity
- self documentation
- simplicity

integrity
- security
- standards compatibility

interoperability
- communication commonality
- data commonality
- modularity
- rehostability
- retargetability

maintainability
- augmentability
- communicativeness
- consistency
- modularity
- self documentation
- simplicity
- structuredness
- test availability

reliability
- accuracy
- completeness
- consistency
- fault tolerance
- modularity
- simplicity

reusability
- application independence
- generality
- hardware independence
- modularity

operating system independence
- self documentation

survivability
- autonomy

distributedness
- fault tolerance
- modularity
- reconfigurability

transportability
- hardware independence
- modularity
- operating system independence
- rehostability
- retargetability

self documentation
- support software independence

usability
- capacity
- ease of installation
- ease of use

maturity
- on-line help
- power
- tailorability
- user documentation

vendor support
- corporate health
- pricing policies
- reputation

support policies

verifiability
- communicativeness
- modularity
- self documentation
- simplicity
- standards compatibility
- structuredness
- test availability

Figure 5 - Criteria details

Product identification:
- Product name and version
- Vendor name

Configuration requirements:
- Host hardware
- Target hardware
- Operating system
- Minimum host primary memory
- Minimum host disk space

Contractual matters:
- Number of users
- Number of CPUs
- Is support available
- Basic software price
- Installation costs
- Other costs

Other required information:
- Applied standards
- Associated tool requirements
- Security level
- User skill level required
- User training required
- Functions supported

Figure 6 - Features which should always be recorded

- augmentability
- completeness
- consistency
- fault tolerance
- modularity
- simplicity
- ease of use
- user documentation
- tailorability
- corporate health
- reputation
- processing effectiveness
- storage effectiveness
- standards compatibility
- application independence
- hardware independence
- operating system independence

Figure 7 - Criteria which should always be evaluated

Glossary

The following definitions have been adapted from several sources. In cases where these sources provided different definitions for the same term, all definitions have been included. Each definition is given in one sentence. The first definition always expresses the sense in which the term is used in this paper. the definitions for features (absolute characteristics) are preceded by (f) and the definitions for criteria (relative characteristics) are preceded by (c) [Lawlis 89].

accuracy - (c) A quantitative measure of the magnitude of error expressed as a function of the relative error, with a high value corresponding to a small error. The precision of computations and control. Those characteristics of software which provide the required precision in calculations and outputs.

activities transformation - (f) A software function which performs a transformation on a product of one life cycle activity to produce a product for another activity.

Ada (MIL-STD-1815A) - (f) The standard which specifies the Ada language.

alternatives in case - (f) The maximum number of individual alternatives which can be defined in a case statement.

alternatives in select - (f) The maximum number of alternatives which can be defined in a select statement.

analysis functions - (f) Software functions which provide an examination of a substantial whole to determine both qualitative and quantitative properties.

application independence - (c) The extent to which software is not dependent on the support required for a particular application. Those characteristics of software which determine its nondependency on database system, microcode, computer architecture, and algorithms.

applied standards - (f) Standards to which software or its inputs or outputs conform.

associated tool requirements - (c) Tools which must be available and compatible with the software.

augmentability - (c) The extent to which software provides for expansion of capability for functions and data. Those characteristics of software which provide for expansion of capability for functions and data.

autonomy - (c) The extent to which software is not dependent on interfaces and functions. Those characteristics of software which determine its nondependency on interfaces and functions.

bits in float - (f) The total number of bits used for a float representation.

bits in exponent - (f) The number of bits used for the representation of the exponent (including its sign in a float representation.

bits in integer - (f) The number of bits used for an integer representation.

CAIS (MIL-STD-1838A) - (f) The standard which specifies the Common APSE Interface Set, a set of interfaces to the APSE kernel.

capacity - (c) The extent of the upper and lower limits of the functions implemented by a tool.

clock resolution - (f) The amount of time distinguishing (the difference between) two consecutive clock times.

communication commonality - (c) The degree to which standard interfaces, protocols, and bandwidths are used. Those characteristics of software which provide for the use of interface standards for protocols, routines, and data representations.

communication effectiveness - (c) The extent to which software performs its intended functions with a minimum consumption of communications resources. Those characteristics of the software which provide for minimum utilization of communications resources in performing functions.

communicativeness - (c) The degree to which the program provides feedback while it is operating to keep the user informed of the functions being performed.

compiling lines of code - (f) The number of lines of source code which are compiled in a minute (wall clock time).

completeness - (c) The extent to which a component provides the complete set of operations necessary to perform a function. The degree to which full implementation of required function has been achieved. Those characteristics of software which provide full implementation of the functions required.

configuration management - (f) A software function which establishes baselines for configuration items, controls the changes to these baselines, and controls releases to the operational environment.

configuration requirements - (f) Those specific components of system hardware and/or software which are required in order for the software to function correctly.

consistency - (c) The extent to which uniform design and documentation techniques have been used throughout the software development project. The use of uniform design and documentation techniques throughout the software development project. Those characteristics of software which provide for uniform design and implementation techniques and notation.

consistency checking - (f) A software function which determines whether or not an entity is internally consistent in the sense that it is consistent with its specification.

contractual matters - (f) Features determining the legal use of and support provided for software which may be specified in a contract with the vendor at the time of purchase.

corporate health - (c) The extent to which it is reasonable to assume that the vendor will remain in business with the ability to continue the current level of customer support.

correctness - (c) The extent to which software design and implementation conform to specifications and standards. The extent to which a program satisfies its specification and fulfills the customer's mission objectives. The extent to which software is free from design defects and from coding defects; that is, fault free. Agreement between a component's total response and the stated response in the functional specification (functional correctness), and/or between the component as coded and the programming specification (algorithmic correctness).

cost - (f) The total price associated with the purchase and productive use of the software (including the basic software price, training costs, installation costs, and any other ancillary costs associated with making the software a productive part of the user's facility).

cost management - (f) A software function which manages cost functions (such as the cost organization structure and the cost estimation methodology).

criteria - Characteristics of software which are used to make relative comparisons of similar software implementations.

cross referencing - (f) A software function which references entities to other entities by logical means.

data commonality - (c) The extent to which standard data structures and types are used throughout the program. The use of standard data structures and types throughout the program. Those characteristics of software which provide for the use of interface standards for data representations.

data flow analysis - (f) A software function which analyzes the formal requirements statements to determine interface consistency and data availability.

DIANA - (f) The standard which specifies a Descriptive Intermediate Attributed Notation for Ada, an abstract data type such that each object of the type is a representation of an intermediate form of an Ada program.

digits in float - (f) The largest number of decimal digits which may be represented by a float.

discriminants in record - (f) The maximum number of discriminants which can be defined for a single record type.

distributed system - (f) A system in which software functions are geographically or logically separated within the system.

distributedness - (c) The degree to which software functions are geographically or logically separated within the system. Those characteristics of software which determine the degree to which software functions are geographically or logically separated within the system.

DOD-STD-2167A - (f) The standard which establishes uniform requirements for software development that are applicable throughout the system life cycle.

ease of installation - (c) The relative ease with which a software product may be integrated into its operational environment and tested in this environment to ensure that it performs as required.

ease of use - (c) The relative ease with which a novice user can become an effective user of the program.

editing - (f) A software function which provides for selective revision of computer-resident data (the data may be textual, graphical, some internal representation, etc.).

efficiency - (c) The extent to which software performs its intended functions with a minimum consumption of computing resources. The amount of computing resources and code required by a program to perform its function. The relative extent to which a resource is utilized. The ratio of actual utilization of the system resources to optimum utilization.

elements in aggregate - (f) The maximum number of elements which can constitute an aggregate.

entries in task - (f) The maximum number of entries which can be defined in a single task.

exactness - The measure of assuredness that a component does no more than it was specified to do and does not contain malicious code.

exceptions overhead - (f) The execution overhead time which is attributable to the presence of exception handlers in the unit.

expandability (extensibility) - (c) The degree to which architectural, data, or procedural design can be extended. The relative effort to increase the software capability or performance by enhancing current functions or by adding new functions or data. The extent to which a component allows new capabilities to be added and existing capabilities to be easily tailored to user needs.

fault tolerance - (c) The extent to which the system has the built-in capability to provide continued correct execution in the presence of a limited number of hardware or software faults. Those characteristics of software which provide for continuity of operations under and recovery from non-nominal conditions. The protection of a component from itself, user errors, and system errors. The ability to recover and provide meaningful diagnostics in the event of unforeseen situations. The damage that occurs when the program encounters an error.

features - Characteristics of software which are used to specify absolute requirements for software implementations.

fixed point delta - (f) The smallest interval which may be used to distinguish among fixed point values.

formatting - (f) A software function which arranges data according to predefined and/or user-defined conventions.

generality - (c) The breadth of the potential application of program components. Those characteristics of software which provide breadth to the functions performed with respect to the application.

GKS - (f) The standard which specifies the Graphical Kernel System, a graphics system which allows programs to support a wide variety of graphics devices and which is defined independently of programming languages.

hardware control - (f) The ability of the software to control hardware directly (such as interrupts, bit manipulations, file servers, task scheduling, preemption, etc.).

hardware independence - (c) The degree to which the software is decoupled from the hardware on which it operates. Those characteristics of software which determine its nondependency on specific hardware. The degree to which hardware dependencies are isolated in a distinct library unit.

host disk capacity needed - (f) The combined storage size (in megabytes) required of the on-line disk units of the host hardware to ensure that the software will run properly.

host hardware - (f) The specification of the manufacturer and model of the computer hardware which will serve as the development platform for the software to be developed.

host memory needed - (f) The size (in megabytes) required of the primary memory of the host hardware to ensure that the software will run properly.

incremental compilation - (f) A software function which produces new object code for a particular source code unit from the previous object code for that unit and a set of specified changes to the source code which produced the original object code.

instantiations of generic - (f) The maximum number of times a single generic unit can be instantiated.

integrity - (c) The extent to which unauthorized access to or modification of software or data can be controlled. The extent to which the software will perform without failures due to unauthorized access to the code or data within a specified time period. The probability that the system will perform without failure and will protect the system and data from unauthorized access.

interoperability - (c) The degree to which an APSE may provide database objects and their relationships in forms usable by the components and user programs of another APSE without conversion. The extent to which two or more systems have the ability to exchange information and to mutually use the information that has been exchanged. The effort required to couple one system to another. The relative effort to couple the software of one system to the software of another system. The probability that two or more systems can exchange information under stated conditions and use the information that has been exchanged.

lines in unit - (f) The maximum number of source code lines which can be compiled in one compilation unit.

linking/loading - (f) A software function which creates a load/executable module on the host machine from one or more independently translated object modules or load modules by resolving cross-references among the object modules, and possibly relocating elements.

long rep forms - (i) The ability to specify a number (integer or float) which will be represented using more total bits than is used by numbers of the same base type without the "long" designation.

maintainability - (c) The extent to which a component facilitates updating to satisfy new requirements or to correct deficiencies. The effort required to locate and fix an error in a program. The ease of effort for locating and fixing a software failure within a specified time period. The ease with which software can be maintained. The probability that the system can be restored to a specified condition within a specified amount of time.

malicious code - operations which covertly damage or attempt to by-pass system security.

management functions - (f) Software functions which aid the management or control of system/software development.

maturity - (c) The extent to which a component has been used in the development of deliverable software by typical users and to which the feedback from that use has been reflected in modifications to the component.

max blocking time - (f) The maximum amount of overhead time used by the run-time system to block a task.

max integer - (f) The maximum number which may be represented as an integer.

modularity - (c) The extent to which software is composed of discrete components such that a change to one component has minimal impact on other components. The extent to which a component is implemented in a hierarchical structure in which identifiable functions are isolated in separate compilation units. The functional independence of program components. Those characteristics of software which provide a structure of highly cohesive components with optimum coupling.

mutation analysis - (f) A software function which applies test data to a program and its "mutants" (programs that contain one or more likely errors) in order to determine test data adequacy.

no restrictions on users - (f) Not disallowing or constraining the use of the software by a particular class of users (such as people not employed by the purchasing organization).

number of CPUs - (f) The total number of computers which may legally serve as the residence for a particular software component.

number of users - (f) The maximum number of users permitted simultaneous execution of a single purchased copy of the software.

numerics - (f) Software features which determine the computational capabilities of the software.

object management - (f) A software function which manages a collection of interrelated data (objects) stored together with controlled redundnacy, serving one or more applications and independent of the programs using the data (objects).

object transformation - (f) A software function which performs a transformation on a particular system object to produce another system object.

on-lin help - (c) The extent to which user documentation is readily available to the user from the program while it is operating.

operating system - (f) The specification of the name and version of the operating system under which the software will run.

operating system independence - (c) The degree to which the program is independent of operating system characteristics. Those characteristics of software which determine its nondependency on a specific operating system. The degree to which operating system dependencies are isolated in a distinct library unit.

options - (f) Software features whose specified values (each of which causes the software to execute i a somewhat different, yet controlled, manner) are set by the user.

PCTE - (f) The standard which specifies the Portable Common Tool Environment, a hosting structure defined by a set of program-callable primitives which support the execution of programs in terms of a virtual, machine independent level of comprehensive facilities.

performance monitoring - (f) A software function which monitors the performance characteristics of the finished product.

peripheral devices - (f) The hardware devices which are attached to and work with the computer but are not an integral part of it (such as printers, terminals, etc.).

PHIGS - (f) The standard which specifies the Programmers Hierarchical Interactive Graphics Standard, a sophisticated graphics support system that controls the definition, modification, and display of hierarchical graphics data.

power - (c) The extent to which a component has capabilities, such as default options and wild card operations, that contribute to the effectiveness of the user.

pricing policies - (c) The degree to which the vendor's prices for product support and upgrades are reasonable and in accordance with accepted practice within the software industry.

processing (execution) effectiveness - (c) The extent to which software performs its intended functions with a minimum consumption of processig resources. The run-time performance of a program. Those characteristics of the software which provide for minimum utilization of processing resources in performing functions. The choice between alternative algorithms based on those taking the least amount of time.

program generation - (f) A software function which provides the translation or interpretation used to construct computer programs (such as language translator generator, syntax analyzer generator, code generator generator, environment definition generator, user interface generator, etc.).

program library management - (f) A software function which performs the creation, manipulation, display, and deletion of the various components of a program library.

quality management - (f) A software function which manages the determination of the achieved level of quality in deployed software systems.

reconfigurability - (c) The extent to which software provides for continuity of system operation when one or more processor, storage units, or communication links fails. Those characteristics of software which provide for continuity of system operation when one or more processors, storage units, or communication links fails.

regression testing - (f) A software function which performs the rerunning of tests in order to detect errors spaqned by changes or corrections made during software development and maintenance.

rehostability - (c) The extent to which an APSE component may be installed on a different host or different operating system with a minimum of reprogramming. The ability of an APSE component to be installed on a different host or different operating system with needed reprogramming localized to the KAPSE or machine dependencies.

reliability - (c) The extent to which a component can be expected to perform its intended functions in a satisfactory manner over a specified period of time. The extent to which a program can be expected to perform its intended function iwth required precision. The extent to which the software will perform without any failures within a specified time period. The probability that software will not cause the failure of a system for a specified time under specified conditions. The probability that the system will perform as intended under stated conditions for a specified period of time.

reputation - (c) The degree of confidence expressed by program users in the vendor's willingness and ability to provide support for the program.

requirements simulation - (f) A software function which executes code-enhanced requirements statements to examine functional interfaces and performance.

resource management - (f) A software function which manages the resources attributed to an entity.

retargetability - (c) The extent to which an APSE component may accomplish its function with respoect to another target with a minimum of modification. The ability of an APSE component to accomplish its function with respect to another target.

reusability - (c) The extent to which a program (or parts of a program) can be reused in other applications. The relative effort to convert a software component for use in another application. The relative effort to adapt software for use in another application.

sale of derived software - (f) Disallowing or constraining the conditions under which some portion of the purchased software may be included in software provided by the purchaser to a third party.

security - (c) The extent of protection of computer hardware and software from accidental or malicious access, use, modification, destruction, or disclosure. The availability or mechanisms that control or protect programs and data.

security issues - (f) Features which affect the use of the software in a classified environment.

self documentation - (c) The degree to which the source code provides meaningful documentation. Those characteristics of software which provide explanation of the implementation of functions. The technical data, including on-line, documentation, listings, and printouts, which serve the purpose of elaborating the design or details of a component.

short rep forms - (f) The ability to specify a number (integer or float) which will be represented using fewer total bits than is used by numbers of the same base type without the "short" designation.

simplicity - (c) The extent to which the complexity of a system or system component (determined by such factors as the number and intricacy of interfaces, the number and intricacy of conditional branches, the degree of nesting, the type of data structures, and other system characteristics) is kept to a minimum. The degree to which a program can be understood without difficulty. Those characteristics of software which provide for definition and implementation of functions in the most noncomplex and understandable manner.

skill level - (f) The level of experience in using similar software.

source code available - (f) The possibility that the source code of the software can be purchased.

source code sizing - (f) The limits imposed on the size of selected components of the software.

standards compatibility - (c) The degree to which the program conforms to specific standards.

statistical profiling - (f) A software function which provides the analysis of a program to determine statement types, number of occurrences of each statement type, and the percentage of each statement type in relation to the complete program.

storage effectiveness - (c) The extent to which software performs its intended functions with a minimum consumption of storage resources. Those characteristics of the software which provide for minimum utilization of storage resources. The choice between alternative source code constructions based on those taking the minimum number of words of object code or in which the information-packing is high.

structuredness - (c) The degree to which the program is constructed of a basic set of control structures, each oe having one entry point and one exit.

subprogram overhead - (f) The overhead time involved in executing a subprogram call.

support available - (f) The possiblity of purchasing support for the software from the vendor on a continuing basis.

support policies - (c) The type and extent of support provided by the vendor for the software.

support software - (f) The specification of the anme and version of the support software required to work with the software in question to ensure proper functionality.

support software independence - (c) The degree to which the program is independent of nonstandard programming language features and other environmental constraints. Those characteristics of software which determine its nondependency on specific support software in the environment (utilities, input and output routines, libraries). The degree to which support software dependencies are isolated in a distinct library unit.

survivability - (c) The extent to which the software will performa and support critical functions without failures within a specified time period when a portion of the system is inoperable. The extent to which software will continue performing when a portion of the system has failed.

tailorability - (c) The extent to which the user interface of the program may be altered to conform to the preferences of the user.

target hardware - (f) The specification of the manufacturer and model of the computer hardware on which the software to be developed will be executed.

task rendezvous - (f) The overhead time required to accomplish a task rendezvous.

test availability - (c) The extent to which tests are available to verify that a program functions in accordance with its requirements. The extent to which tests are available to support the evaluation of a program's performance with respect to specific verification criteria.

timing requirements - (f) The limits imposed on the execution time of selected components of the software.

traceability - (c) The ability to trace a design representation or actual program component back to requirements. Those characteristics of software which provide a thread of origin from the implementation to the requirements with respect to the specified development envelope and operational environment.

traceability analysis - (f) A software functionw hich checks for internal consistency within the software requirements specification.

training - (f) The amount of trainig required to be able to use the software productively.

transformation functions - (f) Software functions which describe how the subject is manipulated to accommodate the user's need.

transportability (portability) - (c) The effort required to transfer the program from one hardware and/or software system environment to another. The relative effort to transport the software for use in another environment. The extent to which a component can be adapted for use in another environment. The extent to which a component may be installed on a different APSE without change in functionality.

units in compile - (f) The largest number of compilation units which can be involved in a single compile.

usability - (c) The extent to which resources required to acquire, install, learn, operate, prepare input for, and interpret output of a component are minimized. The effort required to learn, operate, prepare input, and interpret the output of a program. The relative effort for using software (training and operation). The probability that users can operate the system under specified conditions without user error given they have received specified training.

user documentation - (c) The extent to which documentation conveys to the end user of a system instructions for using the system to obtain desired results. The technical data which serve the purpose of elaboratig the design or details of a component to the user.

user profile - (f) Characteristics required of the user in order to use the software productively.

vendor support - (c) The extent to which a vendor is willing and able to provide the software user with assistance to ensure that the softwae performs desired functions and is willing and able to support the continuing maturation of the product.

verifiability - (c) The extent to which a component facilitates the establishment of verification criteria and supports evaluation of its performance. The effort required to test a program to ensure that it performs its intended function. The relative effort to verify the specified software operation and performance. The extent to which the specified system operation and performance determine the conditions and criteria for tests. The extent to which a component facilitates the evaluation of its correctness, completeness, and exactness.

References

[Arthur 85] L.J. Arthur, Measuring Programmer Productivity and Software Quality, New York: John Wiley & Sons, 1985.

[Bowen 85] T.P. Bowen, G.B. Wigle, and J.T. Tsai, Specification of Software Quality attributes, Volume II, AD A153989, Griffiss AFB, NY: Rome Air Development Center, February 1985.

[DoD 89] Department of Defense, E&V Reference Manual, Version 2.0, September 1989.

[Firth 87] R. Firth, V. Mosley, R. Pethia, L. Roberts, W. Wood, A Guide to the Classification and Assessment of Software Engineering Tools, CMU/SEI-87-TR-10, Pittsburgh: Software Engineering Institute, August 1987.

[Foreman 87] J. Foreman and J. Goodenough, Ada Adoption Handbook: A Program Manager's Guide, Version 1.0 CMU/SEI-87-TR-9, Pittsburgh: Software Engineering Institute, May 1987.

[Houghton 87] R.C. Houghton and D.R. Wallace, "Characteristcs and Functions of Software engineerig environments," Software Engineering Notes, vol. 12, no. 1, January 1987.

[Lawlis 89] P.K. Lawlis, Supporting Selection decisions based on the Technical Evaluation of Ada Environment and Their Components, Ph.D. dissertation, Arizona State University, 1989.

[Lehman 87] M.M. Lehman and W.M. Turski, "Essential Properties of IPSEs," Software Engineering Notes, vol. 12, no. 1, January 1987.

[Lyons 86] T.G.L. Lyons and J.C.D. Nissen, Selecting an Ada Environment, New York: Cambridge University Press, 1986.

[Pressman 87] R.S. Pressman, Software engineering, Second edition, New York: McGraw-Hill, 1987.

[Weiderman 89] N. Weiderman, Ada Adoption Handbook: Compiler Evaluation and Selection, Version 1.0, AD A207717, Software Engineering Institute, March 1989.

APPENDIX H

E&V TOOLS AND AIDS DOCUMENT

Version 3.0
September 1990

Prepared by

Evaluation and Validation Team
Requirements Working Group

for the
Ada Joint Program Office

The Task for the Evaluation and Validation of Ada Programming Support Environments (APSE's) is sponsored by the Ada Joint Program Office

CONTENTS

1 INTRODUCTION 3
- 1.1 Purpose of the Evaluation and Validation Task 3
- 1.2 The Need for E&V Technology 4
- 1.3 Purpose of the Tools and Aids Document 4
- 1.4 Scope 5

2 TYPES OF ASSESSORS 5
- 2.1 Requirements and Specifications 6
- 2.2 Guidelines 6
- 2.3 Metrics 6
- 2.4 Benchmarks, Tests, and Test Suites 6
- 2.5 Questionnaires 6
- 2.6 Decision Aids 6
- 2.7 Monitored Experiments 7

3 ASSESSOR CAPABILITIES 7
- 3.1 Compilation System Evaluators 7
  - 3.1.1 Compiler Evaluators 8
  - 3.1.2 Code Generation Evaluators 8
  - 3.1.3 Program Library Evaluators 8
  - 3.1.4 Runtime Evaluators 9
- 3.2 Target Code Generation Aids and Analysis Toolset Evaluators 9
- 3.3 Test Systems Assessors 9
- 3.4 Requirements/Design Support Assessors 9
- 3.5 Configuration Management Support Evaluators 9
- 3.6 Whole APSE Assessors 10
- 3.7 CAIS (Common APSE Interface Set) Evaluation and CAIS Validation Assessors 10
- 3.8 Distributed Systems Development and Runtime Support Evaluators 10
- 3.9 Distributed APSE Evaluators 10
- 3.10 Transportability Evaluators 10
- 3.11 Methodology Support Evaluators 10
- 3.12 Interoperability Evaluators 11
- 3.13 Multilingual APSE Evaluators 11

4 CONCLUSION 11

APPENDIX A ACRONYMS 12
APPENDIX B COMPILATION SYSTEM EVALUATORS 13
APPENDIX C TEST SYSTEM ASSESSORS 26
APPENDIX D REQUIREMENTS/DESIGN SUPPORT EVALUATORS 28
APPENDIX E WHOLE APSE ASSESSORS 30
APPENDIX F E&V TEAM REQUIREMENTS WORKING GROUP MEMBERSHIP 32

1 INTRODUCTION

The Tools and Aids Document is the result of deliberations of the Requirements Working Group (REQWG) of the Ada Programming Support Environment (APSE) Evaluation and Validation (E&V) Team concerning technology required to evaluate and validate APSEs and their components. This document is a reflection of the APSE E&V Requirements Document and the state of current APSE tools. It also reflects views on the subject which were obtained from a number of surveys conducted among the APSE E&V Team and appropriate ARPANet-MILNet Interest Groups.

1.1 Purpose of the Evaluation and Validation Task

The Ada community, including government, industry, and academic personnel, needs the capability to assess APSEs (Ada Programming Support Environments) and their components and to determine their conformance to applicable standards (e.g., DOD-STD-1838A, the CAIS standard). The technology required to fully satisfy this need is extensive and largely unavailable; it cannot be acquired by a single government-sponsored, professional society-sponsored, or private effort. The purpose of the Evaluation and Validation (E&V) Task is to provide a focal point for addressing the need by (1) identifying and defining specific technology requirements, (2) developing selected elements of the required technology, (3) encouraging others to develop some elements, and (4) collecting information describing existing elements. This information will be made available to DoD components, other government agencies, industry, and academia.

Validation is the process of determining conformance of an APSE or APSE component to existing standards. For example, Ada compilers are currently required to undergo validation by the Ada Validation Organization (AVO) to insure conformance to the Ada language standard (MIL-STD-1815A). In the future, validation may encompass additional standards such as the Common APSE Interface Set (CAIS).

Evaluation is the process of assessing characteristics or attributes of an APSE or APSE component for which there may or may not be standards. Examples of such attributes include usability, efficiency, and maintainability. In the absence of standards, such attributes are free to vary across different APSE implementations. Consequently, these attributes are of interest to users when selecting between APSEs because they contribute to, or detract from, overall APSE quality and suitability for different applications or methodologies. Even in cases where standards do apply to APSE components (e.g., MIL-STD-1815A and Ada compilers), evaluations will be used to supplement information gained during validation processes.

It is anticipated that the primary benefits of E&V will be to encourage the development of quality APSEs and APSE components, and to provide users and developers with a uniform and comprehensive means for assessing and selecting APSE's suitable for their specific applications and methodologies.

1.2 The Need for E&V Technology

Technology for the assessment of APSEs and APSE components (tools) is needed because of the difficulty in assessing APSEs and because of the importance of the decisions made based on these assessments. The importance of an APSE selection is evident when one considers the large, critical, Ada-based systems to be developed in the coming years. The effectiveness, reliability, and cost of these systems will be strongly influenced by the environments used to develop and maintain them. From the point of view of a software developing organization, the decision to select an APSE can be an important investment decision with long-lasting influence on a number of projects and the organization's method of operation, training, and competitiveness. From the point of view of a software maintenance organization, the environment used will strongly influence the organization's effectiveness, as well as the cost of its operations and training. Given the importance of APSE and APSE component selection, a technology to facilitate (or at least give some measure of quantification to the selection process) is required. Thus, the assessment technology addressed herein needs to be developed.

The difficulty of assessing APSEs and tools exists for several reasons. First, an APSE represents very complex technology with many elements, which can be assessed individually or in combination. Second, there is a confusing diversity of choice with respect to individual tools, tool sets, or "whole APSEs", and there are a number of ways of viewing APSEs (see Chapter 3 of the E&V Reference Manual). Third, the state of the art of APSE architecture and of some categories of tools (e.g., graphic design tools) is constantly evolving. Finally, there is a lack of historical data relevant to APSEs, partly because of the general pace of technological change and partly because we are dealing with Ada, a relatively new implementation language. E&V technology provides methods and techniques to overcome these difficulties and provides a basis for assessing performance and other attributes of APSEs.

In addition to the need for assessment technology itself, there is a need for information about this technology. Potential buyers and users of APSEs and tools need a framework for understanding APSEs and their assessment, as well as information about specific assessment techniques. Similarly, vendors of tools and APSEs need to be aware of the deficiencies of current products, as well as the criteria to be used in the assessment of future products. Such awareness on both sides, expressed in a common terminology, should speed up the evolution of better software engineering environments.

1.3 Purpose of the Tools and Aids Document

A critical need exists to support the Ada community with the selection, improvement, and development of APSEs and APSE components. This support extends not only to system developers but also to compiler and other APSE component builders, Ada users, educators, and managers. The information herein contained is presented for those who are willing and able to fund the continued evolution of E&V technology. Examples of such organizations are the Ada Joint Program Office (AJPO), Software Technology for Adaptable Reliable Systems (STARS), Joint Integrated Avionics Working Group (JIAWG), major program offices, the services and any other agency or group capable of providing the funding. To this end, this document identifies the communities'

E&V technology needs, provides definitions of those needs and prioritizes them.

In order to simplify the discussions, the term assessor is used to refer to those tools (e.g., Ada Compiler Evaluation Capability suite) and aids (e.g., checklists) for use in APSE and APSE component evaluation and/or validation. Types of assessors are discussed in Section 2 of this document. They include guidelines, checklists, and experimental tests and procedures. Acquisition of assessors includes incorporation of existing capabilities into the E&V assessment set, purchase of commercial off the shelf (COTS) products, or the development and implementation of needed technologies for assessment.

This document provides information and recommendations from the APSE E&V Team on the kinds of assessors to acquire, prioritized ordering of assessor acquisition and a rationale for those priorities.

1.4 Scope

The APSE E&V Reference Manual identifies the attributes and functionality of APSEs and APSE components which are perceived to require evaluation and/or validation (i.e., assessment). This document identifies the **kinds** of assessors needed to perform the assessment. This document is intended to provide the AJPO and other potential sponsors with a reference for use in the allocation of resources, RFP preparations, and source selection for assessors to support the tasks associated with APSE E&V.

The Tools and Aids Document is a pragmatic guide to assessor acquisition based on the APSE functions available which need evaluation and/or validation, and on the technologies and implementations of these technologies available as APSE function assessors. Through the prioritization of needs, this document emphasizes near-term acquisition of assessors.

Appendices B, C, D, and E provide guidance to tool and aid developers concerning the purpose of assessors in selected areas described in this document, the functionality to be assessed by the tools and aids developed in these areas as well as the attributes to be assessed, and possible approaches for accomplishing the desired assessment.

2 TYPES OF ASSESSORS

Assessors are the mechanisms for providing information about certain characteristics of APSE components, including functionality, performance, maturity, and the suitability of documentation.

Types of assessors include, but are not limited to, the following:

- Requirements and Specifications
- Guidelines
- Metrics
- Benchmarks, Tests, and Test Suites
- Questionnaires

- Decision Aids
- Monitored Experiments

Each assessor type may be implemented in a number of ways, such as automated tools, individual tests and batteries of tests, and manual procedures.

2.1 Requirements and Specifications

Requirements and specifications define the functionality, characteristics, and performance required of an APSE function or tool. These may include quantitative measures that may be made by other assessors and characteristics assessed by qualitative judgements only. As standards are adopted for various APSE capabilities, they will be included and used as the basis of validation for that capability.

2.2 Guidelines

Guidelines provide recommendations for the use or construction of an APSE function or component. Furthermore, guidelines may describe characteristics or qualities the tool should have.

2.3 Metrics

Metrics provide quantitative data about selected characteristics of an APSE or an APSE component.

2.4 Benchmarks, Tests, and Test Suites

Benchmarks are standard tests used to measure the execution performance or acceptability of an APSE function. Benchmarks may test one specific aspect of an APSE function, or may test a number of functions. Tests and Test Suites are instruments used to measure the performance, correctness, or other characteristics of APSE functions.

2.5 Questionnaires

Questionnaires are used to gather data not easily attainable by examination of the APSE or APSE component itself. Examples of such data might include historical information, typical usage scenarios, implementation strategies, enhancement perceptions, problems reports, etc.

2.6 Decision Aids

Decision aids allow a user to assess an APSE function from a particular point of view. Decision aids may combine the results of a number of assessors, each of which is weighted based on its usefulness for the view being considered.

2.7 Monitored Experiments

Monitored experiments, based on model projects involving an aggregation of APSE functions or tools, can be performed on APSEs or APSE components to gather data in a systematic and controlled manner. These experiments can be used for both qualitative and quantitative assessments of the functionality, usability, and performance, as well as other characteristics of APSEs.

3 ASSESSOR CAPABILITIES

A number of assessor capabilities have been identified as being important for providing an APSE E&V capability. Recommendations for near-term assessors are found below. The premise for near term attention is that E&V capabilities can be acquired by assembling existing assessors or by developing the assessors using existing, proven technology. They are ordered by acquisition priority determined by the E&V team. Priorities are based on the importance to the development of mission critical software, the availability of the APSE functions to be evaluated, the significance of the attributes to be evaluated, and the technical feasibility of developing the assessor. These assessors are:

1. Compilation System Evaluators
2. Target Code Generation Aids and Analysis Toolset Evaluators
3. Test Systems Evaluators
4. Requirements/Design Support Assessors
5. Configuration Management Support Evaluators
6. "Whole APSE" Evaluators
7. CAIS Evaluation and Validation Assessors
8. Distributed Systems Development and Runtime Support Evaluators
9. Distributed APSE Evaluators
10. Transportability Evaluators
11. Methodology Support Evaluators
12. Interoperability Evaluators
13. Multilingual APSE Evaluators

3.1 Compilation System Evaluators

This section includes Compiler Evaluators, Code Generation Evaluators, Program Library Systems Evaluators and Runtime Systems Evaluators.

For the purposes of this document, the compilation system is defined as those APSE components which are Ada-specific and are required for validation: the compiler, the code generator, the program library management system, and the runtime support system. While each of these components has characteristics which should be assessed individually, the assessment of their combined functionality will be more critical to the successful development of mission critical software.

The immediate criticality of assessor development for these four compilation system components is made evident by the many large-scale projects with requirements for the use of Ada which are presently being procured or are planned for near-term procurement. These large-scale projects include the

Strategic Defense System, the NASA Space Station, the STARS program, Army Tactical Command and Control System, Army WIS, and the ATF, ATA and LHX programs being evaluated for common avionics systems under the auspices of the JIAWG. The successful performance of these systems depends upon the quality/extent of code generation support and execution support found in the compilation system. APSE development teams are in the process of trying to determine which products are of sufficient quality to support the development of their complex systems. Tools to assist in these evaluations are needed now. See Appendix B for additional guidance for the tool developer in this area.

3.1.1 Compiler Evaluators

Compiler evaluators provide capabilities which measure areas such as compiler performance, code and/or space and/or time optimizations, implementation of real-time embedded programming features, usability, completeness of documentation, and completeness of configuration management and control practices. The issues being probed include how "good" are the compilers, and in what ways are they good.

The Ada Compiler Evaluation Capability (ACEC) provides only the initial evaluation technology required for Ada compilers. Available funding levels have restricted the scope of that effort to something significantly less than what is actually needed, so there is an immediate need to allocate additional funds for the acquisition of compiler evaluation technology which is not found in the ACEC. The ACEC Version 2, available since second quarter 1990, provides for object code efficiency, code expansion size, and assessors to determine the functional capabilities of symbolic debuggers, program library management systems, and compilation systems diagnostic messages. It also provides analysis tools to compare results of different systems and to analyze the performance results from one system.

Additional urgent requirements exist for additional assessment of compiler performance, real-time embedded programming features, usability, and other aspects of compilation that cannot be directly assessed through examination of object code.

3.1.2 Code Generation Evaluators

The generation of efficient code for embedded target processors is of prime importance in the compilation system. Assessors should evaluate both target and native host code generators for performance, efficiency, usability, modifiability, and completeness of documentation.

3.1.3 Program Library Evaluators

Program Library Management Evaluator Systems include evaluators to verify characteristics such as the completeness of documentation, performance, efficiency, functional capabilities, and usability of APSE supplied program library management systems, as examples.

3.1.4 Runtime Evaluators

Runtime evaluators are those which measure characteristics such as the performance, efficiency, and usability of the runtime system. These would also include evaluation of the completeness of documentation and configuration management and control practices of the runtime system.

Ada Runtime evaluation is needed to evaluate the performance of target runtime support systems (RTSS), typically a runtime executive and library of runtime services. Mission critical software is particularly sensitive to efficiency requirements as well as the amount of code needed for RTSS. The ability to make crucial decisions about the capability of a particular Ada RTSS to meet the demands of the application often determines the success or failure of a mission critical project. Providing sound evaluators for RTSS is essential to the success of both Ada and the mission critical systems to which it is applied. Since one of the evaluated measures will include the required space of the run time software, the ability to factor out unused run time services in order to reduce the support library size is an important consideration.

3.2 Target Code Generation Aids and Analysis Toolset Evaluators

These evaluators will provide the capability to evaluate host-target system cross-assemblers; host-based target linkers and loaders; host-based target system instruction-level simulators/emulators; host-based target-code symbolic debuggers; and host-based target system instrumentation interfaces which provide visibility into target processes during mission critical software execution.

3.3 Test Systems Assessors

These assessors will examine the ability of the APSE or APSE component to support and facilitate the planning, development, execution, evaluation and documentation of tests of mission critical software. See Appendix C for additional guidance for the tool developer in this area.

3.4 Requirements/Design Support Assessors

These evaluators will measure the suitability and effectiveness of various software definition, specification, and design tools. This will specifically include evaluators of Ada Program Design Language (PDL) implementations and/or guidelines in the use of Ada as a PDL. See Appendix D for additional guidance for the tool developer in this area.

3.5 Configuration Management Support Evaluators

These evaluators will examine the performance, usability, and completeness of the APSE or APSE component functionality related to controlling the contents of software systems. This will include monitoring the status, preserving the integrity of released and developing versions, and controlling the effects of changes throughout the lifetime of the software system.

3.6 Whole APSE Assessors

These assessors address the APSE macro characteristics, such as overall performance, efficiency, usability, and completeness of the APSE as a whole. Emphasis is given to the "integration services" provided by the APSE infrastructure, which makes the APSE more than a collection of tools. See Appendix E for additional guidance for the tool developer in this area.

3.7 CAIS (Common APSE Interface Set) Evaluation and CAIS Validation Assessors

CAIS validation assessors will determine if the CAIS is in conformance with the DoD Standard.

The CAIS evaluation assessment capability is to be developed to assure that the implementations of the CAIS will provide acceptable performance and other characteristics not covered by validation.

3.8 Distributed Systems Development and Runtime Support Evaluators

These evaluators will assess the ability of the APSE or APSE Components to support software development for distributed processing systems, and to provide runtime support for distributed processing systems.

3.9 Distributed APSE Evaluators

These evaluators will assess the ability of two or more distributed APSEs to communicate in cooperative ways in supporting the development of mission critical software at diverse geographical locations.

3.10 Transportability Evaluators

These evaluators assess the ease with which an APSE or APSE component can be moved to other specified hosts or APSEs without change in functionality. Transportability is measured as the degree to which this relocation can be accomplished without reprogramming.

3.11 Methodology Support Evaluators

These evaluators assess the extent to which the APSE or APSE components support software development methodologies.

3.12 Interoperability Evaluators

These evaluators assess the ability of an APSE to exchange database objects and their relationships with other specified APSEs in forms usable by APSE components and user programs without conversion. Interoperability is measured as the degree to which this exchange can be accomplished without conversion.

3.13 Multilingual APSE Evaluators

These evaluators assess the extent to which the APSE or APSE components support the analysis/development of mission critical software where multiple source languages are involved. Multiple source language support includes the construction of Ada programs which interface to units written in other languages; and/or the support for the maintenance of files of programs not written in Ada (such as documentation); and/or support for programs written completely in languages other than Ada (e.g., existing programs written in FORTRAN, Pascal, C, LISP, etc.).

4 CONCLUSION

While the E&V Team believes current assessment products to be the successful beginnings of APSE and APSE component assessor technology, they are by no means complete or mature. Coordinated efforts must continue to evolve the existing assessors and develop additional assessment technology. For the DoD to be successful in the continued development of MCCS, *assessment technology* must be infused into the mainstream of the software engineering discipline. Billions of dollars can be saved by selecting APSEs and APSE components based on a standard and controlled assessment of the myriad APSEs and APSE components which do/will exist. Protracted or repeated fits and starts due to APSE inadequacies can mean the difference between the ultimate success or failure of MCCS developments.

APPENDIX A

ACRONYMS

ACEC	Ada Compiler Evaluation Capability
AJPO	Ada Joint Program Office
APSE	Ada Programming Support Environment
AVO	Ada Validation Organization
CAIS	Common APSE Interface Set
E&V	Evaluation and Validation
JIAWG	Joint Integrated Avionics Working Group
KAPSE	Kernel Ada Programming Support Environment
KIT	KAPSE Interface Team
KITIA	KAPSE Interface Team Industry/Academia
NASA	National Aeronautics and Space Administration
PDL	Program Design Language
REQWG	Requirements Working Group
RFP	Request for Proposal
RTSS	Runtime Support System
STARS	Software Technology for Adaptable, Reliable Systems
WIS	WWMCCS Information System
WWMCCS	World Wide Military Command and Control System

APPENDIX B

COMPILATION SYSTEM EVALUATORS

Purpose:

These assessors examine the quality of a Compile System tool set. For evaluation purposes, an Ada compile system is delineated into 4 primary capabilities. These are: the compiler, code generator, program library manager, and target runtime system. Each are more thoroughly discussed below under these separate categories.

The attributes have been listed in prioritized order from highest to lowest ranking. The listings themselves are based upon the attributes listed within the E&V Reference Manual. The specific Reference Manual sections are listed next to each attribute. All Reference Manual attributes were initially considered for inclusion in all four of the four compile system capability sections. Where an attribute was unrelated to a capability or of a low priority, it was removed from the listing. Next, where the technology for the assessment of an attribute within a category existed and was covered within the E&V Guidebook it was also removed from the list. Thus the below lists denote those attributes that are important within a compile system capability area and require technology development.

Functionality to be Assessed:

All functions associated with an Ada Compile System will be assessed. In particular the compile system as defined in the E&V Reference Manual section 5.12. This tool set is then examined by its four primary capabilities of: Compiler, Code Generator, Program Library Manager, and target Runtime System.

COMPILER EVALUATORS

Purpose:

These assessors will examine the quality of the Compiler within a compile system tool set. These are assessors in addition to those already defined in E&V Reference Manual section 5.12 in general and 5.12.2 in particular. The Compiler part of a Compile System is only those aspects of the tool set whose primary task is syntax and semantics checking, and intermediate code generation. All issues involved in final program execution characteristics or other standard Compile System capabilities or characteristics are covered under one of the other Compile System categories.

Functionality to be Assessed:

All functions associated with Ada source compiling will be assessed. In particular the compiler as defined in the E&V Reference Manual section 5.12.2 and the E&V Reference Manual Function references listed in sections 7.1.6.7 on Compilation and 7.3.1.15 on Syntax and Semantics Checking.

Attributes to be Assessed:

1. Accuracy [RM 6.4.1]

   Possible Approaches:

   Method: Benchmarks, test and test suites, and monitored experiment.
   Input: Benchmarks, monitored experiment, compile system, and product documentation.
   Process: Execute benchmark tests and perform the experiment, noting failure results.
   Output: A measure of the accuracy of the compiler.

2. Document Accessibility [RM 6.4.13]

   Possible Approaches:

   Method: Questionnaire.
   Input: Questionnaire, compiler documentation, and vendor consultation.
   Process: Check off documents available and their quality on questionnaire.
   Output: A list of documents and a rough measure of their quality.

3. Anomaly Management, Fault or Error Tolerance, Robustness [RM 6.4.2]

   Possible Approaches:

   Method: Benchmarks, test and test suites, and questionnaire.
   Input: Compiler, test suites, and questionnaire.
   Process: Compile test suites, and note error handling/recovery, then complete questionnaire using test suite results.
   Output: Completed questionnaire for use in compiler comparisons or single product general rating.

4. Operability, Communicativeness [RM 6.4.20]

   Possible Approaches:

   Method: Questionnaire.
   Input: Compiler, documentation, and questionnaire.
   Process: Complete the questionnaire by both interactively executing the compiler and by finding questionnaire answers in the documentation.
   Output: A completed questionnaire useful for comparing compilers for capability existence and extent of operability offered.

5. System Compatibility [RM 6.4.34]

   Possible Approaches:

   Method: Questionnaire.
   Input: Questionnaire, compiler, product documentation, and applicable standards.

Process: Complete questionnaire by examining; compiler and documentation.
Output: Completed questionnaire for use in compiler comparisons or single product general rating.

6. Distributedness [RM 6.4.12]

Possible Approaches:

Method: Questionnaire.
Input: Questionnaire, compiler, and product documentation.
Process: Complete questionnaire by examining; product and documentation.
Output: Completed questionnaire for use in compiler comparisons or single product general rating.

7. Commonalty (Data and Communication) [RM 6.4.7]

Possible Approaches:

Method: Guidelines and questionnaire.
Input: Guidelines, questionnaire, compiler, product documentation, and applicable standards.
Process: Use guidelines to complete questionnaire. Compiler will be examined using both the actual software product and its documentation. The compiler will be examined for adherence to established data representation and communication standards.
Output: Completed questionnaire for use in compiler comparisons or single product general rating.

CODE GENERATION EVALUATORS

Purpose:

These assessors will examine the quality of the Code Generator generated' code, within a compile system tool set. These are assessors in addition to those already defined in E&V Reference Manual section 5.1 in general and 5.12.3 in particular. The Code Generator part of a Compile System is only those aspects of the tool set whose primary task is to generate executable code. In general these attributes examine the runtime performance characteristics of the generated code. All issues involved in runtime environment services, source translation, or source management are covered under one of the other Compile System categories.

Functionality to be Assessed:

All functions associated with a compiler's Code Generation capability will be assessed. In particular the code generator as defined in the E&V Reference Manual section 5.12.3 and the E&V Reference Manual Function references listed in section 7.1.6.7 on Compilation.

Attributes to be Assessed:

1. Accuracy [RM 6.4.1]

   Possible Approaches:

   Method: Benchmarks, test and test suites, and monitored experiment.
   Input: Self-checking Benchmarks, monitored experiment, compile system, and product documentation.
   Process: Execute self-checking benchmark tests to generate all possible machine instructions and perform the experiment, noting failure results.
   Output: A measure of the accuracy of the code generator.

2. Reconfigurability [RM 6.4.24]

   Possible Approaches:

   Method: Metrics and questionnaire.
   Input: Metrics, questionnaire, compile system product, and product documentation.
   Process: Examine software product to determine to what extent those aspects supporting software reconfigurability are supported by the product.
   Output: Completed questionnaire and metric rating for use in generated code comparisons or single product general rating.

3. Anomaly Management, Fault or Error Tolerance, Robustness [RM 6.4.2]

   Possible Approaches:

   Method: Benchmarks, test and test suites, and questionnaire.
   Input: Compiler, test suites, questionnaire, compile system product and product documentation.
   Process: Execute test suites and note error handling/recovery, then complete questionnaire using test suite results.
   Output: Completed questionnaire and metric rating for use in generated code comparisons or single product general rating.

4. Distributedness [RM 6.4.12]

   Possible Approaches:

   Method: Questionnaire.
   Input: Questionnaire, compile system product, and product documentation.
   Process: Complete questionnaire by examining; product and documentation.
   Output: Completed questionnaire for use in generated code comparisons or single product general rating.

5. Integrity [RM 6.1.2]

Possible Approaches:

Method: Questionnaire and monitored experiment.
Input: Questionnaire, monitored experiment, compile system product, and product documentation.
Process: Complete the questionnaire using information found in the documentation and information obtained by performing the monitored experiment.
Output: Completed questionnaire for use in generated code comparisons or single product general rating.

6. Document Accessibility [RM 6.4.13]

Possible Approaches:

Method: Questionnaire.
Input: Questionnaire, compile system documentation, and vendor consultation.
Process: Check off documents available and their quality on questionnaire.
Output: A list of documents and a rough measure of their quality.

7. Rehostability [RM 6.4.25]

Possible Approaches:

Method: Questionnaire and monitored experiment.
Input: Questionnaire, monitored experiment, compile system product, and product documentation.
Process: Complete the questionnaire using information found in the documentation and information obtained by performing the monitored experiment.
Output: Completed questionnaire for use in generated code comparisons or single product general rating.

8. System Compatibility [RM 6.4.34]

Possible Approaches:

Method: Questionnaire.
Input: Questionnaire, compile system product, product documentation, and applicable standards.
Process: Examine software product to determine to what extent those aspects supporting software reconfigurability are supported by the product.
Output: Completed questionnaire for use in generated code comparisons or single product general rating.

PROGRAM LIBRARY EVALUATORS

Purpose:

These assessors will examine the quality of the Program Library Manager within a compile system tool set. These are assessors in addition to those already defined in E&V Reference Manual section 5.12 in general and 5.12.5 in particular. The Program Library part of a Compile System is only those aspects of the tool set whose primary task is to manage or control an Ada program library system. All issues involved in source translation, final program execution characteristics or other standard Compile System capabilities or characteristics are covered under one of the other Compile System categories.

Functionality to be Assessed:

All functions associated with Ada Program Library Management will be assessed. In particular the Program Library Manager as defined in the E&V Reference Manual section 5.12.5 and the E&V Reference Manual Function reference listed in section 7.2.1.7 on Program Library Management.

Attributes to be Assessed:

1. Reliability [RM 6.1.3]

Possible Approaches:

Method: Benchmarks, test and test suites, and monitored experiment.
Input: Benchmarks, monitored experiment, compile system, and product documentation.
Process: Execute benchmark tests and perform the experiment, noting failure results.
Output: A measure of the reliability of the program library product.

2. Integrity [RM 6.1.2]

Possible Approaches:

Method: Questionnaire and monitored experiment.
Input: Questionnaire, monitored experiment, compile system, and product documentation.
Process: Complete the questionnaire using information found in the documentation and information obtained by performing the monitored experiment.
Output: Completed questionnaire for use in program library comparisons or single product general rating.

3. Accuracy [RM 6.4.1]

Possible Approaches:

Method: Benchmarks, test and test suites, and monitored experiment.
Input: Benchmarks, monitored experiment, compile system, and product documentation.

Process: Execute benchmark tests and perform the experiment, noting failure results.
Output: A measure of the accuracy of the program library manager.

4. Efficiency [RM 6.1.1]

Possible Approaches:

Method: Benchmarks, test and test suites, and monitored experiment.
Input: Benchmarks, monitored experiment, and compile system.
Process: Execute benchmarks and complete experiment noting the time to complete tasks with respect to difficulty in performing task.
Output: A measure of the efficiency of the program library product.

5. Maintainability [RM 6.2.2]

Possible Approaches:

Method: Questionnaire and monitored experiment.
Input: Questionnaire, monitored experiment, compile system, and product documentation.
Process: Perform experiment completing questionnaire with results, the ease of correcting failures.
Output: A measure of the maintainability of the program library product.

6. Granularity [RM 6.4.17]

Possible Approaches:

Method: Questionnaire.
Input: Questionnaire, compile system, and product documentation.
Process: Complete questionnaire noting the number of distinct capabilities offered.
Output: A measure of the granularity of the program library product. This measure should be paired with a measure of Power in product comparisons.

7. Operability, Communicativeness [RM 6.4.20]

Possible Approaches:

Method: Questionnaire.
Input: Compile system, product documentation, and questionnaire.
Process: Complete the questionnaire by both interactively exercising the program library and by finding questionnaire answers in the documentation.
Output: A completed questionnaire useful for comparing program library products for capability existence and extent of operability offered.

8. Verifiability [RM 6.2.3]

Possible Approaches:

Method: Questionnaire and monitored experiment.
Input: Questionnaire, monitored experiment, compile system, and product documentation.
Process: Adapt the tests within the experiment to test the current program library. Perform the experiment by both adapting the tests and executing them.
Output: A measure of the verifiability/testability of the program library product.

9. Distributedness [RM 6.4.12]

Possible Approaches:

Method: Questionnaire.
Input: Questionnaire, compile system, and product documentation.
Process: Complete questionnaire by examining; program library and documentation.
Output: Completed questionnaire for use in program library comparisons or single product general rating.

10. Reconfigurability

Possible Approaches:

Method: Metrics and questionnaire.
Input: Metrics, questionnaire, compile system, and product documentation.
Process: Complete the questionnaire noting those aspects of the program library that lend themselves to reconfiguration. Rate these characteristics using the metric algorithms.
Output: A measure of the ease of reconfiguration of the program library product.

11. Rehostability

Possible Approaches:

Method: Questionnaire and monitored experiment.
Input: Questionnaire, monitored experiment, compile system, and product documentation.
Process: Complete the questionnaire using information found in the documentation and information obtained by performing the monitored experiment.
Output: Completed questionnaire for use in program library comparisons or single product general rating.

12. Retargetability

Possible Approaches:

Method: Questionnaire and monitored experiment.
Input: Questionnaire, monitored experiment, compile system, and product documentation.
Process: Complete the questionnaire using information found in the documentation and information obtained by performing the monitored experiment.
Output: Completed questionnaire for use in program library comparisons or single product general rating.

RUNTIME EVALUATORS

Purpose:

These assessors will examine the quality of a Compiler's accompanying Runtime System within a compile system tool set. These are assessors in addition to those already defined in Reference Manual section 5.12 in general and 5.12.6 in particular. The Runtime System part of a Compile System is only those aspects of the tool set whose primary task is to support non-generated code language features for final program execution. The specific features and support provided will vary depending on the compile system. Specifically, features will vary between code generator and runtime environment depending on both the compile system and a particular target configuration. All issues involved in other characteristics of a Compile System are covered under one of the other Compile System categories.

Functionality to be Assessed:

All functions associated with an Ada Runtime System will be assessed. In particular the runtime system as defined in the E&V Reference Manual section 5.12.6 and the E&V Reference Manual Function references listed in sections 7.2.3.2 on Input/Output Support and 7.2.3.5 on Runtime Environment.

Attributes to be Assessed:

1. Accuracy [RM 6.4.1]

Possible Approaches:

Method: Benchmarks, test and test suites, and monitored experiment.
Input: Benchmarks, monitored experiment, compile system, and product documentation.
Process: Execute benchmark tests and perform the experiment, noting failure results.
Output: A measure of the accuracy of the runtime system.

2. Processing (Execution) Effectiveness [RM 6.4.22]

Possible Approaches:

Method: Benchmarks, test, and test suites.
Input: Benchmarks, runtime system, and product documentation
Process: Execute benchmarks and note results.
Output: A measure of the run-time performance and efficiency of the runtime system product.

3. Reliability [RM 6.1.3]

Possible Approaches:

Method: Benchmarks, test and test suites, and monitored experiment.
Input: Benchmarks, monitored experiment, runtime system, and product documentation.
Process: Execute benchmarks and perform the experiment, noting failure results.
Output: A measure of the reliability of the runtime system product.

4. Verifiability [RM 6.2.3]

Possible Approaches:

Method: Questionnaire and monitored experiment.
Input: Questionnaire, monitored experiment, runtime system, and product documentation.
Process: Adapt the tests within the experiment to test the current runtime system. Perform the experiment by both adapting the tests and executing them.
Output: A measure of the verifiability/testability of the runtime system product.

5. Capacity [RM 6.4.6]

Possible Approaches:

Method: Benchmarks, test, and test suites.
Input: Benchmarks, runtime system, and product documentation.
Process: Execute benchmarks and note limits reached. Also note any documented limitations.
Output: A measure of the capacity limits of the runtime system product.

6. Operability, Communicativeness [RM 6.4.20]

Possible Approaches:

Method: Questionnaire.
Input: Runtime system, product documentation, and questionnaire.
Process: Complete the questionnaire by both interactively executing the product and by finding Questionnaire answers in the documentation.

Output: A completed questionnaire useful for comparing runtime system products for capability existence and extent of operability offered.

7. Power [RM 6.4.21]

Possible Approaches:

Method: Questionnaire.
Input: Runtime system, product documentation, and questionnaire.
Process: Complete the questionnaire by confirming that documented capabilities do truly exist within the product.
Output: A completed questionnaire useful for comparing runtime system products for capability existence and extent of capabilities offered.

8. Reconfigurability [RM 6.4.24]

Possible Approaches:

Method: Metrics and questionnaire.
Input: Metrics, questionnaire, runtime system, and product documentation.
Process: Complete the questionnaire noting those aspects of the product that lend themselves to reconfiguration. Rate these characteristics using the metric algorithms.
Output: A measure of the ease of reconfiguration of the runtime system.

9. System Compatibility [RM 6.4.34]

Possible Approaches:

Method: Questionnaire.
Input: Questionnaire, runtime system, product documentation, and applicable standards.
Process: Complete questionnaire by examining; product and documentation.
Output: Completed questionnaire for use in runtime system comparisons or single product general rating.

10. Usability [RM 6.1.5]

Possible Approaches:

Method: Questionnaire and monitored experiment.
Input: Questionnaire, monitored experiment, and the runtime system.
Process: Perform the experiment, especially noting the ease with which the correct command/parameter is found and the ease of executing the capability. The time to perform the experiment is also important input to the questionnaire as it provides a rough measure as to the ease of use and power offered by the runtime system.
Output: Completed questionnaire for use in runtime system comparisons or single product general rating showing how "easy" this product was to use.

11. Correctness [RM 6.2.1]

Possible Approaches:

Method: Benchmarks, test, and test suites.
Input: Benchmarks, runtime system, and MIL-STD-1815A.
Process: Execute benchmarks noting failures. Of particular interest are results of large test executions and tests lasting large amounts of time. (This is in addition to ACVC)
Output: A measure of the general correctness of the runtime system product.

12. Commonalty (Data and Communication) [RM 6.4.7]

Possible Approaches:

Method: Guidelines and questionnaire.
Input: Guidelines, questionnaire, runtime system, and product documentation.
Process: Use guidelines to complete questionnaire. Runtime systems will be examined by both exercising the runtime system and examining the documentation. The runtime system will be examined for adherence to established data representation and communication standards.
Output: Completed questionnaire for use in runtime system comparisons or single product general rating.

13. Granularity [RM 6.4.17]

Possible Approaches:

Method: Questionnaire.
Input: Questionnaire, runtime system, and product documentation.
Process: Complete questionnaire noting the number of distinct capabilities offered.
Output: A measure of the granularity of the runtime system. This value should be paired with a measure of Power in runtime system comparisons.

14. Maintainability [RM 6.2.2]

Possible Approaches:

Method: Questionnaire and monitored experiment.
Input: Questionnaire, monitored experiment, runtime system, and product documentation.
Process: Perform experiment completing questionnaire with results, the ease of correcting failures.
Output: A measure of the maintainability of the runtime system.

15. Document Accessibility [RM 6.4.13]

Possible Approaches:

Method: Questionnaire .
Input: Questionnaire, runtime system documentation, and vendor consultation.
Process: Check off documents available and their quality on questionnaire.
Output: A list of documents and a rough measure of their quality.

16. Transportability [RM 6.3.4]

Possible Approaches:

Method: Questionnaire, metrics, and monitored experiment.
Input: Questionnaire, metrics, monitored experiment, runtime system, product documentation, and applicable standards.
Process: Examine runtime system for conformance to applicable interface standards. Execute experiment attempting to port the product to another similar platform.
Output: A measure of the transportability of the runtime system.

APPENDIX C

TEST SYSTEM ASSESSORS

Purpose

These assessors examine the ability of the APSE or APSE component(s) to support and facilitate the planning, development, execution, evaluation, and documentation of tests of mission critical software.

Functionality to be Assessed

Test System Capabilities including Static Analyzers, Tool Building Services, Test Building Services, Test Description and Preparation Services, Test Execution Services, Test Analysis Services, and Decision Support Services (see Guidebook (GB) Table 7.1-1 and Reference Manual (RM) Sections 5.14 and 7.3).

Attributes to be Assessed

1. Correctness [RM 6.2.1]/Verifiability [RM 6.2.3] of Test Building Services, Test Execution Services, and Test Analysis Services. •

Possible Approaches:

Method(s): "Test Suites" of Programs w/Seeded Errors
Input: Test Building Services, Test Execution Services, Test Analysis Services, Programs w/Seeded Errors
Process: Use Test Building Services to develop test data. Execute programs w/seeded errors on test data and use Test Analysis Services to determine whether or not tests were passed. Determine errors.
Output: List of errors found vs. seeded errors
Important Assessor Attributes: Seeded errors should represent typical or likely errors for applications of interest.

2. Survivability [RM 6.1.4]/Reliability [RM 6.1.3] of Test Execution Services.

Possible Approaches:

Method(s): "Test Suites" to Kill the System
Input: Test Execution Services and "Test Suites" including software and data.
Process: Execute test suites on data.
Output: Record of system "hangs and crashes".
Important Assessor Attributes: Test Suites should be designed to maliciously attack underlying system (e.g., overwrite memory, overload I/O channels, result in thrashing, etc.)

3. Efficiency [RM 6.1.1] of Static Analyzers, Test Building Services, Test Execution Services, and Test Analysis Services.

Possible Approaches:

Method(s): Test Suites/Benchmarks - wide variety of programs

Input: Static Analyzers, Test Building Services, Test Execution Services, Test Analysis Services, Test Suites
Process: Apply Testing Capabilities to Test Suites
Output: Performance measurements for testing capabilities on variety of programs
Important Assessor Attributes: Programs should vary in size, complexity, use of language constructs, etc.

4. Usability [RM 6.1.5]/Interoperability [RM 6.3.2] of Test System Components individually and as a "whole".

Possible Approaches:

Method(s): Monitored Experiments & Questionnaires
Input: Test System Components, Experiment Procedures
Process: Conduct Experiment as instructed by procedures
Output: Completed Questionnaire describing "ease of use" or lack thereof of test system when used as individual components and as a whole.
Important Assessor Attributes: Experiment should simulate intended use of test system -- typical users, typical software to be tested, etc.

5. Integrity [RM 6.1.2] of Test Execution Services, Test Analysis Services, and Decision Support Services.

Possible Approaches:

Method(s): ???
Input:
Process:
Output:
Important Assessor Attributes:

APPENDIX D

REQUIREMENTS/DESIGN SUPPORT ASSESSORS

Purpose

These evaluators measure the suitability and effectiveness of various software definition, specification, and design tools. This specifically includes evaluators of Ada Program Design Language (PDL) implementations and/or guidelines in the use of Ada as a PDL.

Functionality to be Assessed (Sections from E&V Reference Manual - RM)

- Strategic Planning (RM 4.1)
  - Enterprise Modeling (RM 5.9.3)
  - Identification of Strategic Systems Opportunities (RM ?)
  - Analysis (RM 7.3)
    - Goals and Problems (RM ?)
    - Technology Impact (RM ?)
    - Critical Success Factors (RM ?)
    - Simulation and Modeling (RM 7.3.2.3)
  - Documentation (RM 5.8, 7.1.2.3)
    - Operational Concept Document (RM
  - Interface to Requirements Engineering Tool(s) (RM 4.2, 4.3)

- Requirements Engineering (RM 4.2, 4.3)
  - Requirements Specification Language (RM 5.9.1, 5.9.2, 7.1.6.1, 7.1.6.2)
  - Allocation of Requirements to Hardware and Software (RM ?)
  - Process Modeling (RM 5.9.5, 7.3.2.3)
    - Data Flow
    - Control Flow
  - Data Modeling (RM 5.9.4, 7.3.2.3)
    - Data Structure
    - Entity-Relationship
  - User Interface Simulation/Prototyping (RM 5.9.6, 5.9.7, 7.3.2.1, 7.3.2.2)
  - Analysis (RM 7.3)
    - Data Flow (RM 7.3.1.3)
    - Functional (RM 7.3.1.4)
    - Requirements Traceability (RM 7.3.1.6)
    - Testability (RM 7.3.1.7)
    - Quality Measurement (RM 7.3.1.9)
    - Consistency/Completeness (RM 7.3.1.12, 7.3.1.13)
    - Maintainability (RM 7.3.1.18)
    - Auditing (RM 7.3.1.22)
    - Stability (RM ?)
    - Simulation and Emulation (RM 5.9.6, 7.3.2.1, 7.3.2.3, 7.3.2.13)
  - Documentation (RM 5.8, 7.1.2.3)
    - Requirements Specification(s) (RM 4.2.2, 4.3.2)
  - Interface to Design Engineering Tool(s) (RM 4.2, 4.4, 4.5, 7.1.7.1)

- Design Engineering (RM 4.2, 4.4, 4.5)
  - Hardware Design (CAD) (RM ?)
  - Process Modeling (RM 5.9.5, 7.3.2.3)

- Real-Time Support (RM 7.3.2.17)
  - Control Specification
  - State Transition
  - Timing Analysis (Dead Lock, Racing) (RM 7.3.2.14)
- Language-Specific Support (Booch/Buhr, Textual PDL)

Data Modeling (RM 5.9.4, 7.3.2.3)
- Normalization (RM ?)

Report/Screen Design (RM 7.1.1.3, 7.1.5, 7.3.2.4)

Analysis (RM 7.3)
- Interface (RM 7.3.1.5)
- Requirements Traceability (RM 7.3.1.6)
- Testability (RM 7.3.1.7)
- Test Condition (RM 7.3.1.8)
- Quality Measurement (RM 7.3.1.9)
- Complexity Measurement (RM 7.3.1.10)
- Consistency/Completeness (RM 7.3.1.12, 7.3.1.13)
- Reusability Analysis (RM 7.3.1.14)
- Maintainability (RM 7.3.1.18)
- Invocation (RM 7.3.1.19)
- Scanning (RM 7.3.1.20)
- Structured Walkthrough (RM 7.3.1.21)
- Auditing (RM 7.3.1.22)
- Type (RM 7.3.1.27)
- Units (RM 7.3.1.28)
- Formal Verification (RM 7.3.3)

Documentation (RM 5.8, 7.1.2.3)
- Design Document(s) (RM 4.2.2, 4.4.2, 4.5.2)
- Test Plan(s) (RM 4.4.3)
- Programmers' Manual(s) (RM 4.5.6)
- Users' Manual(s) (RM 4.4.6, 4.5.6)

Interfaces to Application Generator(s) (RM 4.6, 5.11, 7.1.7.3)
- Source Code (RM 5.11.1)
- Database Schema (RM 5.11.2)
- Report (RM 5.11.3)
- Screen (RM 5.11.4)

Requirements Reconstruction (RM 7.1.7.2)

Generic Interfaces (RM 6.3.2)
- Document Production System(s) (RM 5.8, 7.1.1, 7.1.2)
- Project Management System(s) (RM 5.5, 7.2.2)
- Configuration Management System(s) (RM 5.7, 7.2.2.7)
  - Configuration Control (RM 5.7.2)
  - Version Control (RM 5.7.4)
- Test System(s) (RM 5.14, 7.3)

APPENDIX E

WHOLE APSE ASSESSORS

Purpose

These assessors will examine the quality of an APSE, as a whole, in support of a project team across the entire life cycle of software development and maintenance, or in support of a project team as it performs a major "chunk" of activities.

Functionality to be Assessed

All functions associated with software development, but especially the "integration services" provided by the APSE infrastructure, which make the APSE more than just a collection of tools. These services support data integration, presentation integration, interoperability integration, process integration, coordination, and monitoring, as discussed in Section 3.3 of the E&V Reference Manual.

Attributes to be Assessed

1. Performance and Design attributes [GB 6.1,6.2] including Efficiency, Completeness, Integrity, and Usability of the APSE as a whole in support of an entire team across the entire life cycle, and Integration [GB 6.4.x].

Possible Approaches:

Method(s): Structured experiments built around model projects partially completed.
Input: Model project documentation and code, and experiment scenario scripts.
Process: Follow the scripted scenarios based on the model project and answer the questions posed.
Important Assessor Attributes: Efficiency, as influenced by the effort required of the evaluation team in establishing a valid basis for judging the performance of the APSE. Adaptability, as indicated by the ease with which the evaluators can expand and modify the scripted scenario to address aspects of performance that are of prime importance to them.

2. Adaptation attributes [GB 6.3] including Expandability, Augmentability, Interoperability, and Transportability.

Possible Approaches:

Method(s): Structured experiments built around model projects and a changing suite of tools.
Input: Model project documentation and code, and scenario scripts.
Process: Follow the scripted scenarios based on the model project and answer the questions posed.
Important Assessor Attributes: Efficiency, as influenced by the effort required of the evaluation team in establishing a valid basis for

judging the adaptation qualities of the APSE. Adaptability, as indicated by the ease with which the evaluators can expand and modify the scripted scenario to address aspects of adaptation that are of prime importance to them.

APPENDIX F

E&V TEAM REQUIREMENTS WORKING GROUP MEMBERSHIP

Becky Abraham	US Air Force
Jerry Brookshire	Texas Instruments Corporation
Mike Burlakoff	Southwest Missouri State University
Peter Clark	TASC
Bard Crawford	TASC
Dan Eilers	Irvine Compiler Company
Jay Ferguson	National Security Agency
Fred Francl	Sonicraft, Inc.
Greg Gicca	Sanders Associates
Marlene Hazle	MITRE Corp.
Alan Impicciche	US Navy
Elizabeth Kean	US Air Force
Pat Lawlis	US Air Force
Tom Leavitt	Boeing Military Airplanes
Ronnie Martin	Purdue University
Sandi Mulholland	Rockwell International
Helen Romanowsky	Rockwell International
Lloyd Stiles	US Navy
Nelson Weiderman	Software Engineering Institute

APPENDIX I

MINUTES

OF THE

EVALUATION AND VALIDATION TEAM

GENERAL SESSION

5-7 DECEMBER 1988

The task for the Evaluation and Validation of Ada Programming Support Environments (APSEs) is sponsored by the Ada Joint Program Office (AJPO).

TABLE OF CONTENTS

Section | Page

1.0 MONDAY, 5 DECEMBER 1988 3
- 1.1 Opening Remarks 3
- 1.2 Rockwell's Software Engineering Environment 4
- 1.3 Federal Mandate for Evaluation and Validation 4
- 1.4 AdaGEN: The CASE Tool for Ada on the PC 9

2.0 TUESDAY, 6 DECEMBER 1988 12
- 2.1 Evaluation and Validation (E&V) Reference System Update 12
- 2.2 Report on Software Development Environments Conference 13
- 2.3 Ada Compiler Evaluation Capability (ACEC) Update 15
- 2.4 CAIS Implementation Validation Capability (CIVC) Update 17

3.0 WEDNESDAY, 7 DECEMBER 1988 18
- 3.1 Announcements 18
- 3.2 Working Group Status Report 18
  - 3.2.1 CAIS Implementation Validation Capability Working Group (CIVCWG) Status Report 18
  - 3.2.2 Coordination Working Group (COORDWG) Status Report 19
  - 3.2.3 Requirements Working Group (REQWG) Status Report 19
  - 3.2.4 Ada Compiler Evaluation Capability Working Group (ACECWG) Status Report 21
  - 3.2.5 Classification Working Group (CLASSWG) Status Report 23
  - 3.2.6 Standards Evaluation and Validation Working Group (SEVWG) Status Report 24
- 3.3 Closing Remarks 25

LIST OF APPENDICES

10.0 APSE E&V Reference System Bulletin Board Announcement 26

20.0 List of Attendees 27

1.0 MONDAY, 5 DECEMBER 1988

1.1 Opening Remarks

Mr. George Robertson welcomed the Evaluation and Validation (E&V) Team to the FCDSSA facility. He gave a synopsis of the struggles encountered in the effort to promote Ada's acceptance. He emphasized that although the learning curve is steep and first attempts at using Ada will cost more and take more time, once a user has learned the language and is able to take advantage of its ability and receive maximum benefits, the final savings in both time and money is substantial.

Mr. Lloyd Stiles extended greetings to the team giving them a general outline of the facility and the surrounding San Diego area.

Mr. Ray Szymanski, E&V chairperson, formally convened the meeting by welcoming everyone to San Diego. The following announcements were made:

- With two stipulations, legal permission has been attained to have non-government personnel participate in the E&V activity:
    - Based on the original E&V Team charter, only government personnel are acknowledged as actual E&V Team members.
    - Distinguished reviewers cannot manage the E&V activity.
- Mr. Szymanski welcomed Dave Fitts from the Ada Joint Program Office (AJPO).
- Plans for AJPO moving to Systems Command are presently on hold.
- Several E&V Team presentations were given at AdaJUG. Mr. Szymanski's presentation covered the E&V efforts and the E&V Reference System. Kermit Terrell gave a presentation on the Ada Compiler Evaluation Capability (ACEC). An ACEC Birds of a Feather session was also held. Capt. Rebecca Abraham gave a presentation on the Software Technology for Adaptable Reliable Systems (STARS) activity.
- John Camp and Linda Elderhorst will not be returning to the team.

Mr. Szymanski opened the floor for questions from the team.

Mr. Leavitt asked for an update on the status of the Congressional directive for including evaluation in the validation process. Mr. Szymanski stated that he had not been formally tasked to do anything in regard to this.

Major Patricia Lawlis made a request for any team members to please complete her software evaluation questionaire as she needed the feedback for her Master's dissertation. Major Lawlis thanked those team members who had already taken the time to respond to her questionnaire.

At this time Mr. Szymanski introduced the first speaker, Ms. Sandi Mulholland.

1.2 Rockwell's Software Engineering Environment

Sandi Mulholland
Rockwell

Ms. Sandi Mulholland gave the first presentation, the details of which cannot be publically released at this time.

1.3 Federal Mandate for Evaluation and Validation

John Stanton
Gemma Corporation

Mr. Stanton gave a two-part presentation focusing on the Federal perspective of processing standards, their evaluation and validation, and implementing testing services for standards testing. Mr. Stanton was involved in a project for the National Bureau of Standards and drew on this experience for the first part of his presentation concerning federal standards for software.

The Federal Information Processing Standard (FIPS) includes programming languages, data base languages, graphics languages, and operating systems. FIPS is mandated by the Department of Commerce for the National Institute of Standards and Technology and the National Computer and Telecommunications Laboratory. A Gemma Corporation contract supports the various FIPS standards validation groups. Gemma Corporation has 12-14 people who test most of the testable FIPS standards such as environment, data base, interfaces, and languages.

The Gemma Corporation tests more processing standards with a wider range of implementations than any other organization in the world. Testing standards is taking groups of tests out in test suites and running those against the standard. Testing standards is purely a validation process with the test results published in the Certified Compiler List.

Some of the FIPS standards include COBOL85, Basic, C, FORTRAN, PASCAL, and Ada. The C language has already become a FIPS, and according to Mr. Stanton, this has dangerous implications. The C language is linked with POSIX which is the standardized UNIX environment. The impact being that as a federal agency under the FIPS guidelines, if users are not mandated to use Ada, then they are free to use POSIX and C.

The Federal Government promotes standardized technical knowledge to protect its investment, and testing standards is a result of the Federal Government's desire to protect its inventory. The standards are enforced through the procurement process. Procurements are won or lost based on the conformance of the FIPS standards compiler implementations.

However, under certain mission critical circumstances, the Department of Defense (DoD) can invoke a unique feature called the Warner Amendment. The Warner Amendment excludes various mission critical systems and acquisitions from the Federal Information Processing Standards. Although the FIPS guidelines apply all the way through the DoD, the Warner Amendment guards against Federal imposition under certain mission critical circumstances.

The Federal interpretation of evaluation is that "the test method shall be used solely for the purpose of assessing conformance to the FIPS. The test method should not include tests or procedures for assessing other product characteristics such as performance, standard refinement, or product quality."

When issueing a FIPS, Federal acquisitions regulations do not mandate a compiler be evaluated prior to its sale. However, for this agency's purposes, the RFP specifies that the ACEC test must be run on compilers under consideration, and the award will be based on those results. Performance requirements must be clearly and fairly quantified in the RFP.

FIPS languages are so well established that the do not need to be specified in the RFP; they are assumed. However, if a standard language is not specified and a non-standard language is used, millions of dollars can be wasted on the effort. Although the FIPS are not driving the competitive standards, the procurement process is becoming more controlled and constrained due to the nature of the competitive environment. Procurements are overturned when a company using a standard language challenges the company using a non-standard language.

The government can initiate and support technology. However, the DoD has no mandate to be running an evaluation service and is not a profit-making organization and unless no company is available to provide an evaluation service, the government should only fill in the gap until the market pressures demand commercial industry step in and provide the service. Technology flourishes when it is allowed free flow and access beyond government control. A compromise is needed between allowing the technology to flourish unrestricted and using it as a bureaucratic building block for an organization within the government. A third party test will not be possible until this issue is resolved. The Ada industry has to grow faster and larger, and then the third-party testing issue on the ACEC will take care of itself.

The ACEC is distributed through DACS and is subject to government control. Although distribution of ACEC is restricted, acquiring a copy is not a complicated process. Restricting ACEC is an attempt to protect the government from unscrupulous vendors who would develop software based on ACEC and sell that software back to the government. However, it is acceptable for the vendor to develop the software and charge the government for running it because the vendor is providing a service which the government would otherwise have to perform itself.

Addressing the issue of appending the Ada Compiler Evaluation Capability (ACEC) to the Ada Compiler Validation Capability (ACVC), Mr. Stanton named the five standards which apply:

a. American National Standard Insitute (ANSI),

b. Federal Information Processing Standard (FIPS),

c. International Standard Organization (ISO),

d. North Atlantic Treaty Organization (NATO), and

e. Military Standards (MIL).

Providing the DoD is correctly testing Ada, the Federal E&V does not want to be involved. However, the Federal E&V will have a problem with appending the ACEC to the ACVC due to a conformance testing policy and procedure they issued in August 1988 which states that in no way is an evaluation requirement to be associated with a validation certificate.

The second part of Mr. Stanton's presentation focused on implementing testing services for standards testing. This part of the presentation covered six areas:

a. Goal setting,

b. Policy Issues and Lasting Solutions,

c. Testing Service Support Products,

d. Testing Service Support Environment,

e. Things That Go Wrong and Solutions, and

f. Conclusions and Recommendations.

Goal Setting

What do you want to accomplish? Areas to be considered include:

- Successful technology insertion
- Government and vendor support
- Conformance testing
- Financially self-sustaining and responsive service
- Fairness to industry - low controversy

Policy Issues and Lasting Solutions

- Build an iterative policy as experience with testing the standard grows.
- Establish a clear-cut definition of conformance that can be clearly understood and defended as it changes.
- Establish clear and responsive lines of communication and authority for policy exceptions and changes.

Testing Service Support Products

- Policy
  - Procedures
  - Testing methods
- Automated tools
  - Analysis
  - Reporting
  - Suite manipulation
  - Integrity checking
- Handbooks
- Checklists
- Quality control procedures

Testing Service Support Environment

- Timely interpretations
- Reporting of certification lists
- Management of support products and exception conditions
- Customer support
- Resourcing

Things That Go Wrong and Solutions

- Understanding the level of resources required
- Investing in the development of the support environment and products
- Documented definitions in:
  - Policy
  - Conformance
  - Procedures
  - Certification criteria
  - Applicable tests
  - Reciprocal certification
  - Resolution cut-off
- Inappropriate and excessive reporting requirements
- Modeling the testing of a new standard based solely on a few existing techniques for other standards

- Assuming high availability of vendor machine and financial resources

- Lack of corporate experience

- Not anticipating or adjusting policy, products, and environment to change

- The variety of different architectures that will adopt the standard and its impact on:
  - Policy
  - Procedures
  - Conformance definition
  - Test Suite
  - Applicable tests

- "Holes" in the standard:
  - Feedback into revision
  - Testing effects

- Informative testing
  - Quality
  - Performance
  - Range
  - Attribute

- Timely decisions during changing times and effectively communicating those changes to:
  - Implementations
  - Users
  - Government
  - Testers
  - Support environment
  - Support products

Conclusions and Recommendations

- Even if you have a final standard and model implementation(s), you are only 50% towards a complete testing service.

- Formal standards testing has been done successfully, by a few, for many years. Readily available corporate knowledge in the area is almost non-existant.

- Many individuals may know the standard and perhaps a few have some testing experience, however, complex standards testing in a regulatory environment requires experienced management and staff.

- Successful testing services thrive because they consistently recognize the need for:
  - High-level management commitment and resourcing
  - An engineering approach to building support products and support environments

- Politics will reign and must be dealt with, however, the support environment must be successfully buffered from those politics.

- Be ready to move fast, change policy and reorient when the service is first introduced and as it matures.

- Be ready to resource previously blind areas.

- Have an idea for balancing tradeoffs between standard conformance and technology insertion.

1.4 Adagen: The CASE Tool for Ada on the PC

Herm Fischer
Robert Karag
Mark V Systems Limited

Mr. Szymanski introduced Mr. Herm Fischer and Mr. Robert Karag from Mark V Systems. Mr. Karag's presentation covered the Mark V Systems company, their orientation, mission, future plans and past achievements. Following Mr. Karag's presentation, Mr. Fischer ran a demonstration of Adagen.

Mark V Systems has been in the software development and consulting business for over 14 years. Mark V Systems' product, Adagen, is their speciality and they emphasize Ada support for both government and industry. They have been involved with Ada programs since its beginning, and today they are very active in object-oriented software development methodologies. Mr. Karag emphasized that Mark V Systems made public domain of software development methodologies as opposed to private proprietary methodologies.

Mark V Systems prides itself in supporting its customer's environment by providing a good tool which is easy to use, as well as providing training on a regular basis. Mark V Systems offers upgrades on Adagen and to its customer's training program.

Adagen is a window-based, tailorable object-oriented development tool. It includes Ada specific design graphics support, automatic code generation, reverse engineering, and support for 2167A. Adagen has been on the market since September 1986 and has accumulated much user feedback. The Naval Weapons Center has a site license for ten copies and ten users of Adagen, and is available for references.

Mr. Karag stated that Mark V Systems was trying to help in three areas of Ada development:

1. Ada specific graphics will support requirements analysis, as well as virtually any diagramming technique. Ada design supports the two popular and public domain editors, Booch and Buhr.

2. Automatic code generation will allow the user to generate legal Ada into compilable Ada from Ada specific graphics. Adagen will automatically generate compilable Ada from Ada specific graphics, from the detailed design about to be generated, and it will be suitable for verification and can be included with the documentation.

3. Reverse engineering will allow the user to go back from Ada text to Ada specific graphics. When working with embedded systems, changes need to be synchronized, and as team members are added, communication needs to be efficient.

Having both code generation and reverse engineering capabilities facilitates the use of Ada because Adagen is used not only as a production tool, but also as a training tool.

Future Use and Enhancements

For the first quarter of 1989, Mark V Systems will continue their strategy of developing Adagen as a window-based tool, and will help their users integrate their analysis activities. The long-term plan for Adagen is to enhance its analytical and documentation capabilities for 2167A.

In January 1989, a real-time interaction within the code will be provided which will allow the user to see not only the topology and compilation dependency, but also real-time interactions will be depicted using interaction lines by Buhr.

In relation to 2167A, a document generation system based on hypertext will be added. The database itself will be used to link together the different graphical representations of the logical objects in the design, and to drive the production of chapters and sections of the books from the graphed structure representing the ideological entities in the design.

For analytical tools, Ada dynamics simulation and state transition will be added, as well as reverse transition from design back to the requirements.

Concerning interfaces, the database being developed is CAIS A evaluated. Mark V Systems is concerned with database subset interfaces. One advantageous commonality betweeen Adagen and Rational is the use of Diana trees, which facilitate the integration of Rational and Adagen for graphics and reverse engineering. The CAIS structure database will be used to generate on paper the equivalent of the closely interrelated views of the same objects that are seen on screen.

The following summarizes the responses to questions and comments made from the floor:

- Adagen on Mac 2 will be available April 1989.

- The cost of the system, per user until January is $4,500.00 plus annual maintenance of $900.00.

- Approximately 100 systems are presently in use, and this number may double effective January 1st.

- Mark V is profitable today. Adagen, with two major stockholders, involves no venture capital; the company's plans are financed through capital funds, and some customers have made future commitments.

- Adagen involves both the specification and generation of package specs.

- An estimate of the 1989 Adagen cost will be between $7,000.00 and $9,000.00, including the Mac version.

- The data dictionary is part of the database due to the special dictionary requirements that Ada provides, and will be added in the first quarter of 1989.

- Mark V Systems plans to buy addendums to their generated documents, but at the present time, no addendums of the 2167A documents are provided. Currently, Adagen interfaces with the user's publishing system for 2167A.

- The key item of reverse engineering is that it builds the tree that will drive setting up the data base so the pieces of the program are in the data base in the tree or graph of objects. In the future, the data base will be coupled with different diagrams, which will be a more complete way of iterating between the reverse and the forward.

- With reverse engineering and procedural calls, the attempt was to provide the Buhr charts showing the hierarchy and producing declarations diagrams as an initial step to producing Buhr charts. Many users have suggested a C port showing data references, global data use which would be a very valuable form of charting. The reverse shows where the major data instantiations are but not the defined structure underneath.

- Mark V Systems plans to expand by 20 people in the next few months to facilitate implementing their plans.

- The implementation is C and its Prologue. There are no plans to reimplement in Ada as Adagen is so closely tied to windowing packages.

- Adagen is an existing product that came into use with the object-oriented design product from underlining nouns and verbs. This is a good approach and would be used on the new C development.

- The new development is mostly on the reverse, which is in Prologue and on the database. The development is a low level C type.

Mr. Karag concluded his presentation saying that Adagen is to support Ada object-oriented software development. Adagen is only for Ada but is not written in Ada. Mr. Herm Fischer took the floor for the Adagen demonstration.

2.0 TUESDAY, 6 DECEMBER 1988

2.1 Evaluation and Validation (E&V) Reference System Update

Dr. Bard Crawford
The Analytic Sciences Corporation (TASC)

Dr. Crawford briefed the team on the status of the Reference Manual and the Guidebook. The events of the past three months were discussed along with subsequent publicity.

Version 1.0 of the Reference Manual was approved last spring and has now received a Defense Technical Information Center (DTIC) number. Version 1.1 of the Reference Manual includes the correct references to the revised Guidebook. Version 1.1 was reviewed by the Classification Working Group (CLASSWG) at the September E&V meeting. These changes were incorporated into the document before it was sent to the Avionics Laboratory. It was then reviewed by Mr. Szymanski, Captain Abraham, Mr. Clark, and Dr. Crawford. It was revised accordingly and delivered in October. Mr. Szymanski approved it in November and it has now been delivered to AJPO for approval.

Dr. Crawford stated that two steps are involved in the approval process. The document is in the second step which involves approval by the Public Affairs Office and is waiting to be signed.

There was no Version 1.0 of the Guidebook. When it was revised, it became Version 1.1 to correspond with the Reference Manual. Version 1.1 of the Guidebook was approved in September by AJPO. The Air Force then obtained the proper signatures. The initial mailing to the team occurred in November 1988. The current mailing list, outside of the team, includes over 70 names.

The two documents have been publicized at several recent events including the Ada Expo, the TriAda Expo, a conference on Methods and Tools for Real-Time Systems, AdaJUG, and the Ada Information Clearinghouse Bulletin Board. Pamphlets primarily dedicated to the Reference System were also distributed at these events.

There are plans to release information to other bulletin boards and to the Ada Information Newsletter. Future events include the SIGAda/AdaJUG presentation and the National Security Industrial Association (NSIA) panel.

Dr. Crawford displayed a slide containing the bulletin board announcement draft (see Appendix A). Also shown was an outline of the Guidebook divided into the early chapters and the formal chapters. The early chapters include:

1. Introduction
2. Structure and Use of the Guidebook
3. Integration of E&V Technology
4. Synopses

The formal chapters are as follows:

5. Compilation System Assessors
6. Target Code Generation Aids and Analysis Assessors
7. Test System Assessors
8. Tool/Host Interface Accessors
9. Ada Design Support Accessors
10. Configuration Management Support Assessors
11. Distributed System Development and Runtime Support Assessors
12. Distributed APSE Assessors
13. "Whole APSE" Assessors
14. Adaption Assessors

99. Other Assessors

Dr. Crawford commented on a paper given by Hank Sumi from the Naval Air Development Center (NADC) on Ada design support assessors. The paper outlined a model project approach to evaluation of four competing front-end tools which were requirement analysis tools. A typical Navy avionics program was used four different times for four tools with emphasis on matters such as how real-time requirements are handled, and performance analysis aspects of the system. One of the tools selected could not run; therefore, only three tools were evaluated. This evaluation was performed by Software Productivity Solutions (SPS) in Florida. A follow-on for some framework analysis will require a complete APSE evaluation which should be tracked and summarized upon completion. A copy of the first paper is available.

2.2 Report on Software Development Environments Conference

Peter Clark
The Analytic Sciences Corporation (TASC)

Mr. Clark attended the Software Development Environments (SDE) Conference in Cambridge, Massachusetts. He briefed the team on the sessions and demonstrations, and provided a brief summary.

The following is a list of the sessions held at SDE:

- Research Software Development Environments (SDEs)
- Industrial SDEs
- Case Studies of Actual Environments
- Panel: Major National and International SDE Programs
- SDEs and the Software Development Process
- SDE Integration Mechanisms
- SDE Technology Elements
- Version Management
- Panel: Data Management for SDEs
- Public Tool Interfaces: CAIS and Portable Common Tool Environments (PCTE) with discussion
- Progress in Ada SDEs

Of these sessions, Mr. Clark summarized the Industrial SDEs and the Public Tool Interfaces presentations. Regarding the industrial software development environments, Tom Stralich (Government Research Corporation) presented a paper on the software lifecycle support environment, an environment framework modeled to 2167A. An Entity/Relationships/Attributes (ERA) model of 2167A is done in *Standard Query Language (SQL). The environment includes 45 tools in* nine different categories. The model has approximately 200 entity types and 300 relationships, and is a fairly comprehensive model of 2167A. GRC claims to do cover to cover document generation. In addition, there is some knowledge-based support for linking various tools together in a common user interface.

One of the subschemas for the database is an environment/tools subschema in which the user describes the tool, the interface of the tool to the environment, and, provided the database does not have its own interactive environment (such as an editor), a menu or screen is displayed where the user enters the data in the same user interface as the rest of the environment. The environment initializes the process, formatting the data in whatever format the tool would describe. Basically this environment is a preprocessor, with the underlying database describing the tool interface.

Presentations on public tool interfaces were given by Bob Munk of MITRE, on the Common APSE Interface Set (CAIS) and Mr. Clark with Tom Hughes of Honeywell/Bull on the Portable Common Tool Environment (PCTE). A discussion followed where Rich Thull of SofTech represented the CAIS and Meyer Morin of ICL represented the PCTE. The objective third party was Bill Paceman of Atherton which has the software backplane, a commercial effort attempting to develop a tool interface similar to the CAIS and PCTE. At the attendees' request for a recommendation, Mr. Paceman stated that for DoD purposes, CAIS is the better choice over PCTE.

The following demonstrations were given at the conference:

- IBM: RPDE^3 Environment Framework
- Saber: C Language Environment
- IDE: Software Through Pictures
- Apollo: Domain SEE
- Rational
- Siemans: BIIN, Classic Ada
- DEC/Wayne State: Visual Interactive C (VIC)
- Orsay: Wish, Window Icon Shell for UNIX
- Aachen: IPSEN Integrated and Incremental/SDE
- Compass: DAPSE
- Columbia: Marvel Knowledge-Based SDE for Code, Test, Maintenance Mercury Generator for Distribution Language-Based Environment
- Brown - Field: SDE for UNIX
  Anim: Algorithm Animation System
- Sun: NSE, Network Software Environment
- Darmstadt: PSG Generator for Interactive Language-Based Environment

Mr. Clark summarized by stating that there was a lot of discussion at the conference concerning object management. The research efforts appear to be directed towards object-oriented design and away from the Entity Relationship (ER) models. The more practical environments are still very much ER models but many university efforts are object-oriented.

Dr. Lindquist explained that the difference between the ER and object-oriented models is that the ER models have failed to associate any type of activity with entities, while the object in the object-oriented approach has a state and may have activities, as well as responds to messages and operations.

According to Bill Paceman (Atherton) the original implementation of the software was an ER approach having 450 procedural interfaces. When redesigned using an object-oriented approach, it was reduced to 17 interfaces; therefore, the object-oriented approach may yield a simpler interface.

2.3 Ada Compiler Evaluation Capability (ACEC) Update

Tom Leavitt
Boeing Military Airplanes

Mr. Leavitt briefed the team on the Ada Compiler Evaluation Capability (ACEC) status.

Presentations were made over the last quarter at the SAE conference, AdaJUG, and Ada Expo. A tutorial will be given at the next SIGAda/AdaJUG meeting which will center on a practical approach to using the ACEC; how to run it and how to interpret the results. In addition, preparations are being made for a Preliminary Design Review (PDR) in February 1989.

About 18 copies of ACEC's first release are being distributed. Of those who have received the ACEC, two users have given feedback. One user had a problem with the integer counter overflowing on a fast machine. Another user complained about PRAGMA_INCLUDE (the mechanism used to put the timing loop in the ACEC) as not being a valid pragma because it changes the legality of the program.

Phase 3 activities have been limited over the last quarter due to a schedule extension for user feedback. However, library package measurement techniques have been studied. The problem was that the elaboration of the library package is done only once before the main program is entered. Because the elaboration is only performed once and not repeated, the normal timing loop cannot be used to measure it. To measure the elaboration, a clock measurement was placed inside the source associated with the package body. The user can read the clock and specify the elaboration pragmas. The result is a unique order of library package elaborations that can be measured at the end of the sequence of elaborations. The same technique as a normal clock loop can be used so this method is not totally based on the revolution of a clock. The clock measurement was coded up and tried on two different systems of 8,000 characters and seems to be workable. The error bounds of this method are much courser than the error bounds of a normal timing loop.

The results obtained on both the systems that were used to elaborate library packages and obtain dynamic data were as expected. These results were comparable to the results that would be obtained if equipment allocators had been used for the objects being allocated.

However, it is much faster to use domestic packages on a stack basis limitation, and it is not surprising that on a system that has slow allocators that a package of elaborations will be correspondingly slow. In light of the results from the measurement schemas, the overhead associated with being exclusively a library package is not particularly significant.

Three arrays were elaborated inside a package. On DEC Ada, when performed with dynamic objects in a library or in an exclusive allocator, allocation took approximately 6 milliseconds, adding package also took 6 milliseconds. When these are nested packages or they are blocked, the time increases to approximately 200 milliseconds.

Studies of the I/O intensive tests primarily focused on patterns of access in terms of random, sequential, and cyclic patterns, and observing disk caching. The basic problem is that because of the variability in the language specification which states that it provides FORM strings for giving implementation-dependent tuning parameters and information to the system for running the problem, some of the parameters set which specify FORM strings can not have an impact on the performance whether or not allocations, rewrite, or any amount of buffering is done. Many things can be specified on some systems which would impact performances. There is no guarantee that the default settings used between the two systems, if not specified, are comparable.

The current approach to the problem is to provide a duplicate set of tests for the patterns, and to instruct the user to construct FORM strings matching the expected use of the system.

The other major activity in the last quarter was the submittal of a contract proposal on 11 November 1988. The contract is currently under review. The proposal covers diagnostics, symbolic debuggers, library systems, and a single system analysis tool.

Mr. Szymanski stated that an attendee at the AdaJUG Birds of a Feather session illustrated the need for ACEC. The attendee's company's compiler met both their expectations and needs, but although it had been validated, they were not using the compiler in the same mode in which it had been validated. According to the attendee, validation guarantees very little and that performance is the number one issue. The attendee also mentioned that they had used a prerelease version of the ACEC in choosing a compiler. However, the choice between two comparable compilers was not based upon performance tests exclusively, but also on the vendor's ability to respond to customer needs.

Dr. Crawford attended an AdaJUG presentation by Robert Firth of SEI concerning compiler evaluation. Even in respect to the code produced from performing simple routines, the performance of current cross compilers is disappointing. Dr. Crawford asked Lt. Marmelstein to comment on ACEC's sensitivity.

Lt. Marmelstein said that some simple language feature tests can be made, and the amount of code generated would be small enough to study. However, the ACEC has no automated portion to do this; it must be done manually. Users can write their own tests to do this, however.

Mr. Leavitt concluded his presentation by stating that some of those tests will possibly be automated in a future contract.

2.4 CAIS Implementation Validation Capability (CIVC) Update

Jeff Facemire
SofTech

Mr. Facemire began his presentation with a review of the May status report and proceeded to bring the team up-to-date on the CAIS Implementation Validation Capability (CIVC).

A Pre-Critical Design Review (Pre-CDR) was held in May at Arizona State where Softech initially indicated that they would use the Ministry of Defense (MOD) test harness as part of the test administrator. In addition, the details concerning the interaction between the MOD and CIVC and the requirements for the cooperation between the two systems, as well as an initial view into some of the design details were presented.

During the May to August time frame, procurement problems resulted in the contract suffering some setbacks. Although the work continued, attendance at meetings and conferences was not attempted, and therefore, SofTech did not attend Ada Europe.

Prior to the September E&V Team Meeting, CIVC had started a strength-building phase and had begun to recover from their setbacks. But although the CAIS Implementation Validation Capability Working Group (CIVCWG) was conferred with informally during the meeting, a formal presentation at the general session was not possible.

In mid-September, the AJPO changed their position on the usage of the MOD test harness and recommended that all references to the MOD test harness be removed and no use of it be made in CIVC's design. This resulted in a complete redesign of the test administrator portion. The test suite was only minimally affected. The scheduled presentation of the CDR materials in August was postponed. Requirements had to be redefined throughout the high-level design of the test administrator.

A Technical Interchange Meeting was held in November. The overall high-level design of the CIVC was reviewed and included both the test administrator and the test suite. The low-level design CDR material for the test suite was reviewed as well.

In December, CIVC had recovered most of the low-level design on the test administrator. SofTech had continued to develop test objectives and scenarios throughout this time period.

A few levels of reviews are yet to be completed. Of the 694 test objectives developed, 244 have been reviewed in San Diego. Of the 218 scenarios developed, 141 have been reviewed in Houston, while 102 scenarios have been reviewed in San Diego. The 72 test cases developed will not be reviewed in this manner. Internal code reviews will be done in Houston only. As a result of the setbacks, the Phase 1 final delivery schedule has been adjusted from early February to May or June.

3.0 WEDNESDAY, 7 DECEMBER 1988

3.1 Announcements

Mr. Szymanski extended the team's gratitude to Mr. Lloyd Stiles for hosting this quarterly meeting of the E&V Team and to Mr. George Robertson for the use of the facilities.

3.2 Working Group Status Report

3.2.1 CAIS Implementation Validation Capability Working Group (CIVCWG) Status Report

Gary McKee, chairperson of CIVCWG, presented the working group's status report to the team.

Attendees:

Gary McKee, chairperson
Dr. Tim Lindquist
Jeff Facemire
Tracy Holmes
Lloyd Stiles
John McBride
Shawn Fanning
Jack Foidl

Guests in attendance were: Ray Szymanski, Ronnie Martin, and Denise Connor, working group reporter.

Deliverables Due This Quarter: None

Accomplishments:

- Attended the CIVC Technical Interchange Meeting on 10 November 1988 in Houston, TX., SofTech brought back five issues which were resolved in this session of CIVCWG.
- Discussed the Operator's Guide.
- Discussed the simulator paradigm.
- Discussed test selection criteria in a joint CIVCWG/SEVWG meeting.
- Discussed quality factors.

Action Items:

- Mr. Szymanski. Investigate data rights for CIVC documents.
- Mr. McKee. Review and suggest changes to the quality factors.
  Mr. McKee. Generate a NET message regarding control generic.
- Mr. Foidl. Send an updated copy of the validation policy to the working group.
- Mr. McBride. Send various documents to Mr. McKee.
- Mr. McBride. Send independent verification and validation information to Mr. McKee and Dr. Lindquist.
- Mr. Szymanski. Consider adding the Operator's Guide to the contract.

3.2.2 Coordination Working Group (COORDWG) Status Report

Mr. Patrick Maher presented the COORDWG status report to the team in the absence of the chairperson, Mr. Don Jennings.

Accomplishments:

- Reviewed the September E&V General Session Minutes.
- Met with the Requirements Working Group (REQWG) to consider issues pertaining to public relations (PR).

Projected Work:

- Produce a status report or newsletter.
- Develop PR strategy and the action items designated by REQWG. Information is needed from team members attending Ada related conferences.
- Review the December E&V General Session Minutes.
- Determine which members still need the third annual report.

3.2.3 Requirements Working Group (REQWG) Status Report

The chairperson of REQWG, Major Patricia Lawlis, presented the working group's status report to the team.

Attendees:

Maj. Patricia Lawlis, chairperson
Capt. Becky Abraham
Peter Clark
Dr. Bard Crawford
Dan Eilers
Fred Francl
Greg Gicca
Alan Impicciche
Tom Leavitt
Don Mark
Ronnie Martin
Sandi Mulholland
Nelson Weiderman
Barbara Rhoads, recorder

Deliverables Due This Quarter:

- Tools and Aids Document, Version 2.0

Accomplishments:

- Prepared the "Bulletin Board" announcement
- Upgraded slide presentation

Key Issues:

- E&V work should be based on requirements
- Public relation efforts
- Re-examination of E&V technology and priorities

Projected Work:

- Continue PR
- Work on next version of the Tools and Aids Document

Presentations Planned:

Conferences

- WAdaS
- TriAda

Organizations

- JIAWG Software Group
- MASA Software Engineering Group
- STARS
- DARPA
- CECOM
- RADC
- JLC
- NASA SSE

Action Items:

Carry Over

- Mr. Brookshire/Ms. Mulholland. Expand the distributed APSEs/ cross-development attributes to a checklist for use in the Guidebook.
- Ms. Mulholland. Life cyle support from whole-APSE view.
- ACECWG courtesy item. Capture lessons-learned information from the initial ACEC release (suggested source INFO-Ada).

New

- COORDWG courtesy item. Develop an E&V newsletter (resurrect Status Report but with a name change) to incorporate such items as the Bulletin Board announcement.
- COORDWG courtesy item. Coordinate a more detailed summary of E&V activities and products which is suitable for PR distribution.
- AAAF courtesy item. Develop a procedure for automatic distribution of approved E&V (hardcopy) material to team members.

Mr. Leavitt. Send a copy of Ada Expo material on the ACEC to Ms. Wills for reproduction.

- Ms. Wills courtesy item. Reproduce and distribute Ada Expo material.
- COORDWG courtesy item. Establish a file on the AJPO system to keep track of conference attendance by team members (includes establishing a format for the information in the file).
- COORDWG courtesy item. Establish a format for members to use in contributing to the conference attendance list.
- COORDWG courtesy item. Establish a procedure for updating the file monthly and sending a copy of the new version of the file to all team members.
- COORDWG courtesy item. Distribute the Bulletin Board material to periodicals (listed in the previous Status Report) when Reference Manual, Version 1.1 is approved.
- REQWG. Distribute Bulletin Board material to appropriate bulletin boards when the new version of the Reference Manual is approved.
- Mr. Szymanski courtesy item. Send information on the ACEC tutorial at the AdaJUG/SIGAda to those who have ordered the ACEC.
- Mr. Szymanski courtesy item. Make hard copies of both the new E&V slide presentation and accompanying commentary and distribute to team members.
- COORDWG courtesy item. Investigate the extent of the electronic AdaIC information.
- COORDWG courtesy item. Determine the procedure AdaIC could use for updating documents from the E&V team.
- COORDWG courtesy item. Check on possible problems with electronic copies of "official" documents.
- Dr. Crawford. Produce ASCII form of the Reference Manual and the Guidebook.
- COORDWG courtesy item. Deliver ASCII form of the Reference Manual and the Guidebook as appropriate.
- Ms. Mulholland. Develop a first draft of a quality factors document and develop its connection with the work of the E&V team.
- Mr. Eilers/Ms. Mulholland. Develop a white paper defining the requirements for compilation systems in more detail.
- Maj. Lawlis. Put December REQWG Status Report on the NET.

3.2.4 Ada Compiler Evaluation Capability Working Group (ACECWG) Status Report

The ACECWG chairperson, Nelson Weiderman, presented the working group's status report to the team.

Attendees:

Nelson Weiderman, chairperson
Dan Eilers
Greg Gicca
Alan Impicciche
Tom Leavitt
Don Marks
Lt. Robert Marmelstein
Sandi Mulholland
Ray Szymanski (ex officio)

Deliverables Due This Quarter: None

Accomplishments:

- Provided feedback on the recently released version of the ACEC.
- Reviewed a list of "showstopper" issues not presently covered.
- Reviewed the ACEC relative to the issues raised by R. Firth at AdaJUG.
- Reviewed the ACEC relative to the issues by the Uniformity Rapporteur Group.

Key Issues:

- Need to flag tests for which "fast is good" may not apply.
- Need better indexes for the tests.
- Need more compiler expertise at project reviews.
- Need checklists to monitor the coverage of the ACEC.
- Need requirements for the general areas of testing to be reviewed by ACECWG.
- Need test objectives to be reviewed by ACECWG.

Projected Work:

- Review Boeing/AFWAL proposals for Phase 3 of the contract.
- Review proposals for evaluation service.

Deliverables Due Next Quarter: None

Presentations Planned:

- A more complete presentation of Phase 3 test plans.
- Presentation of the feedback from the ACEC user community.
- Presentation on results of the Preliminary Design Review (PDR) in early February.

Other Significant Information: None

Action Items:

- Mr. Weiderman. Put the December ACECWG status report on the NET.
- Lt. Marmelstein. Get team input on test objectives before PDR.
- Lt. Marmelstein. Issue corrections on the first ACEC release.
- Lt. Marmelstein. Give consideration to the "showstopper" issues and provide a report.
- Mr. Eilers. Look at the coverage issues relative to other checklists.
- Lt. Marmelstein. Distribute white papers on Phase 3 to ACECWG members.

3.2.5 Classification Working Group (CLASSWG) Status Report

Ronnie Martin, chairperson, presented the CLASSWG status report to the team.

Attendees:

Ronnie Martin, chairperson
Capt. Becky Abraham
Peter Clark
Dr. Bard Crawford
Maj. Patricia Lawlis
Fred Francl
Debra Kent, new member

Deliverables Due This Quarter: None

Accomplishments:

- Reviewed the released Version 1.1 of the Reference Manual and Guidebook for enhancements to be done prior to the next release.
- Discussed mappings of tools to functions in the new Reference Manual, Chapter 5.
- Added a new checklist to the Guidebook for emulation capabilities.
- Determined additional references to E&V technology for inclusion in the Guidebook.

Key Issues:

- The CLASSWG needs assistance from other team members on the following:
  - Program Library Management Capabilities Checklist
  - Import/Export Capabilities Checklist
  - Timing Analysis Checklist
  - Tuning Analysis Checklist
  - Real-Time Analysis Capabilities Checklist

Projected Work/Action Items:

- Ms. Mulholland/Mr. Brookshire. Further elaborate whole-APSE assessment issues and transform customization issues in a checklist.
- Ms. Martin. Send any current information on the above referenced customization issues to Ms. Mulholland.
- Ms. Mulholland. Further elaborate whole-APSE assessment issues/address life cycle support issues.
- Ms. Martin. Review whole-APSE usability assessment per Ms. Hazel.
- Mr. Clark. Review the whole-APSE assessment issues work to date for inclusion in Version 2.0 of the Reference Manual.
- Mr. Martin. Review Chapter 4 of the Reference Manual, Life Cycle Activities, for its treatment of testing-related products and functions.
- Mr. McKee. Review/refine tools and mappings to functions to ensure coverage of CAIS implementations in the Reference Manual.

- CLASSWG. Review the Reference Manual Chapter 6, Introduction of Attributes for Appropriateness to APSE E&V as opposed to Weapons Systems Evaluation.
- Mr. Martin. Review attribute definitions in response to Ms. Hazel's message.
- Mr. Szymanski. Determine if the Aerospace Compiler Evaluation Document is available.
- Capt. Abraham. Determine if the WIS documents are available.
- Capt. Abraham. Determine if any recent STARS documents should be synopsized in the Guidebook.
- Mr. Clark. Determine if the synopses can be further referenced in the Guidebook.
- Mr. Francl. Generate an Instruction Level Simulation Capabilities Checklist for inclusion in the Guidebook.
- Ms. Martin. Determine any modifications to the Test System Assessors Checklist needed to highlight traceability.
- Ms. Martin. Review the RADC Tools Directory for checklist inputs.
- Mr. Clark. Review/enhance the Runtime Environment Taxonomy.
- Ms. Martin. Enlist help from experts on the areas identified above.

Deliverables Due Next Quarter: None

Presentations Planned: None

Other Significant Information: None

3.2.6 Standards Evaluation and Validation Working Group (SEVWG) Status Report

Dr. Tim Lindquist, chairperson, presented SEVWG's status report to the team.

Attendees:

Dr. Tim Lindquist, chairperson
Jeff Facemire
Tracy Holmes
Jack Foidl
Gary McKee
Lloyd Stiles
Ray Szymanski, de facto member

Accomplishments:

- Reviewed existing Issues and Strategies Document
    - Test Selection Criteria
    - Cost estimation for development of CIVC A
    - Elaboration of CIVC (A) review board, evolution, clarification activity.

Key Issues:

- The following are not yet included in the Issues and Strategies Document:
    - Coverage assessment
    - CAIS based tool for Validation Test Management
    - Format for recommendations

Deliverables Due Next Quarter:

- Draft of Issues and Strategies to the team at the February meeting.

Action Items:

- SEVWG. Submit any comments on the existing draft of the Issues and Strategies Document on the NET, including input on the remaining unresolved issues.

3.3 Closing Remarks

Dr. Lindquist briefed the team on details concerning the February meeting.

The December session of the E&V Team was then adjourned.

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX A

10.0 APSE E&V REFERENCE SYSTEM BULLETIN BOARD ANNOUNCEMENT

Two coordinated documents produced by the E&V (Evaluation and Validation) Task, under AFWAL leadership and AJPO sponsorship, are now available for distribution. They are: The "E&V Reference Manual, Version 1.1" and the "E&V Guidebook, Version 1.1." Considered together, they are known as the "E&V Reference System." They provide information about APSEs (Ada Programming Support Environments) and their assessment.

The "E&V Reference Manual" establishes common terminology and a framework for understanding APSEs. It includes a Life-Cycle Activities Index, a Tool Category Index, a Function Index, and an Attribute Index. Each index entry contains a definition, cross references to entries in the same or other indexes, and "pointers" to relevant sections in the "E&V Guidebook." As a stand-alone document it is intended to help users find useful information about index elements and relationships among them. In conjunction with the Guidebook, it is intended to help users find criteria, metrics, and methods for assessment of APSEs and their components.

The "E&V Guidebook" provides descriptions of specific instances of assessment technology. These include evaluation (assessment of performance and quality) or validation (assessment of conformance to a standard) techniques. For each category of item to be assessed (e.g., compilation system, test system, whole APSE, etc.) there are descriptions of various techniques--such as test suites, questionnaires, checklists, and structured experiments. The Guidebook also contains synopses of documents of general historical importance to the entire field of Ada environments and their assessment.

To obtain copies of the documents and/or to get on the E&V Mailing List (to ensure notification of future E&V products) send your name and address electronically (preferred) to: szymansk@ajpo@sei@cmu@edu, or by regular mail to: Mr. Raymond Szymanski, WRDC/AAAF, Wright Patterson AFB, OH 45433-6523.

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX B

20.0 LIST OF ATTENDEES

Abraham, Capt. Rebecca
AFWAL/FDCL
WPAFB, OH 45433-6543

Clark, Peter
TASC
55 Walkers Brook Drive
Reading, MA 01867

Crawford, Dr. Bard
TASC
55 Walkers Brook Drive
Reading, MA 01867

Eilers, Dan
Irvine Compiler Corp.
18021 Sky Park Circle, #L
Irvine, CA 92714

Facmire, Jeff
SofTech, Inc.
1300 Hercules Drive
Suite 105
Houston, TX 77058

Fanning, Shawn
SofTech, Inc.
16875 W. Bernardo Drive
San Diego, CA 92127

Fitts, David
AJPO
Rm 3D139
(Fern St/C107)
The Pentagon
Washington, D.C. 20301-3081

Foidl, Jack
TRW
Systems Division
Suite 205
9265 Sky Park Court
San Diego, CA 92123-4213

Francl, Fred
Sonicraft, Inc.
8859 S. Greenwood
Chicago, IL 60619

Gicca, Greg
Sanders Associates
MER24-1283
95 Canal Street
Nashua, NH 03061

Holmes, Tracy
GTE Government Systems
1 Federal Street
Billerica, MA 01821

Impicciche, Alan
Naval Avionics Center
NAC Code 826
6000 E. 21st Street
Indianapolis, IN 46219

Kent, Deborah
SAIC

Lawlis, Maj. Patricia
AFIT/ASU
3318 E. Dry Creek Road
Phoenix, AZ 85044

Leavitt, Tom
Boeing Military Airplanes
P.O. Box 7730, MSK80-13
Wichita, KS 67277-7730

Lindquist, Dr. Tim
Computer Science Dept.
Arizona State University
Tempe, AZ 85287-5406

McBride, John
SofTech, Inc.
1300 Hercules Drive
Suite 105
Houston, TX 77058

McKee, Gary
GARINCAR
P.O. Box 3009
Littleton, CO 80161-3009

Maher, Patrick
Motorola, Inc.
8220 E. Roosevelt
Mail Drop R1208
Dept. PZ511
Scottsdale, AZ 85252

Mark, Don
RADC/COEE
Griffiss AFB, NY 13441-5700

Marmelstein, Lt. Robert
AFWAL/AAAF-3
WPAFB, OH 45433-6543

Martin, Ronnie
Software Eng. Research Center
Dept. of Computer Science
Purdue University
West Lafayette, IN 47907-2004

Mulholland, Sandi
Rockwell International
400 Collins Road, NE
Cedar Rapids, IA 52498

Rhoads, Barbara
Oneida Resources, Inc.
3578 Kettering Blvd.
Suite 300
Dayton, Ohio 45439

Stanton, John
GEMMA Corporation
1111 Jefferson Davis Highway
Crystal Gateway North
Arlington, VA 22003

Stiles, Lloyd
FCDSSA, San Diego
200 Catalina Blvd.
San Diego, CA 92147

Szymanski, Ray
AFWAL/AAAF-3
WPAFB, OH 45433-6543

Weiderman, Nelson
Software Eng. Institute
Carnegie-Mellon University
Pittsburg, PA 15213

Wills, Betty
CCSO/XPTB
Tinker AFB, OK 73145

APPENDIX J

MINUTES

OF THE

EVALUATION AND VALIDATION TEAM

GENERAL SESSION

21-23 FEBRUARY 1989

The task for the Evaluation and Validation of Ada Programming Support Environments (APSEs) is sponsored by the Ada Joint Program Office (AJPO).

TABLE OF CONTENTS

Section . . . . . Page

1.0 TUESDAY, 21 FEBRUARY 1989 3
  1.1 Opening Remarks 3
  1.2 Survellance, Targeting, Attack Radar System (STARS) Presentation 3
  1.3 Chairman's Comments 4
  1.4 The Xinotech Program Composer 5
  1.5 Effect of Tasks and Exceptions on Execution Time 12
  1.6 Prototype Update 17

2.0 WEDNESDAY, 22 FEBRUARY 1989 18
  2.1 E&V Reference System Status Report 18
  2.2 Ada Compiler Evaluation Capability (ACEC) Status Report . . 22
  2.3 CAIS Implementation Validation Capability (CIVC) Status Report 22

3.0 FRIDAY, 24 FEBRUARY 1989 24
  3.1 General Discussion 24
  3.2 Working Group Status Reports 27
    3.2.1 Requirements Working Group (REQWG) Report 27
    3.2.2 CAIS Implementation Validation Capability Working Group (CIVCWG) Status Report 29
    3.2.3 Standards Evaluation and Validation Working Group (SEVWG) Status Report 30
    3.2.4 Classification Working Group (CLASSWG) Status Report . 31
    3.2.5 Ada Compiler Evaluation Capability Working Group (ACECWG) Status Report 33
    3.2.6 Coordination Working Group (COORDWG) Status Report . . 34
  3.3 Closing Remarks 35

LIST OF APPENDICES

10.0 Ada Technology Assessment 37
20.0 List of Attendees 39

1.0 TUESDAY, 21 FEBRUARY 1989

1.1 Opening Remarks

Captain Rebecca Abraham opened the February meeting on behalf of Ray Szymanski the E&V Team Chairman. She welcomed the team to Phoenix and thanked Dr. Tim Lindquist for making the arrangements.

Dr. Lindquist welcomed the team and gave a brief orientation.

1.2 Surveillance, Targeting, Attack Radar System (STARS) Presentation

Capt. Rebecca Abraham
WRDC/FDCL, Wright-Patterson Air Force Base

When STARS was originally conceived, the Department of Defense (DoD) perceived the high cost of software as the most important problem. Other problems included the rising requirements for mission-critical software, low productivity rates, poor management controls, and the inability to predict and budget software tasks. The requirements for future weapons systems, such as adaptability and reliability; new and enhanced capabilities; and large software systems, were also important considerations.

The DoD software initiative comprises a triad of Ada (language), STARS (technology), and the Software Engineering Institute (SEI) (transition). The STARS goal is decreased costs and increased reliability. This is to be accomplished by automated and integrated tools and methods, and with reusable parts.

Capt. Abraham summarized four aspects of the STARS program:

1. STARS-funded contractors developed 32 different tools to enhance reusability. These tools are available now.

2. IBM, Unisys, and Boeing are prime contractors creating a worthwhile repository and developing a software engineering environment completely hardware and operating systems independent.

   The three prime contractors have several subcontractors who are ready to move into their next phase of development. The first six months was devoted to developing prototypes of the repository and of other items under development.

3. The Shadow program involves paralleling an ongoing contract's software with Ada. Ideally, the Shadow program will yield research results as well as an opportunity to demonstrate the benefits of using Ada by comparing it to the "shadowed" software. Two Air Force Shadows are in progress: the Advance Millimeter Wave Seeker Missile at Eglin Air Force Base and the F-1 11 program.

   The purpose of the Shadow program is to promote Ada's acceptance in the SPO environment. Specifically, SPO is developing a digital flight control system in JOVIAL for the F-1 11. Through the Shadow program, Ada was paralleled with JOVIAL, and after analyzing the flight control system design, SPO found better ways to optimize and make the system more efficient by using Ada.

4. Several research programs were funded by STARS supported funding as well as several other sources. This research is being performed at Wright Research Development Center (WRDC) at Wright-Patterson Air Force Base (WPAFB), Ohio.

Capt. Abraham then opened the floor to questions.

Dr. Lindquist stated that in the original design of STARS, SEI, and Ada a much closer relationship appeared to exist between SEI and the STARS program than what there now appears to be. SEI was originally intended to be a showcase for a model software engineering environment providing technology insertion. Dr. Lindquist asked that in view of the preceding description of STARS, how does the mission fit with what the STARS program is intending to do. Capt. Abraham replied that the SEI is still working very closely with STARS by acting as consultants to the Shadow programs. There is an official agreement where SEI is supporting the STARS initiative.

Dr. Crawford followed up Dr. Lindquist's question asking if one DoD office oversees the three programs. Capt. Abraham stated that the STARS program is currently housed out of the Defense Advanced Research Projects Agency (DARPA) and is no longer a part of the same organization as AJPO. However, STARS, SEI, and Ada are still sharing one Program Element (PE) at the SEI.

1.3 Chairman's Comments

Ray Szymanski
WRDC/AAAF-3, Wright-Patterson Air Force Base

Upcoming events include:

- The next quarterly meeting of the E&V Team is scheduled for the first Wednesday in June and will be a two and a half day meeting.

- There will be an Ada Compiler Evaluation Capability (ACEC) tutorial at the upcoming Ada JOVIAL User Group (AdaJUG)/Special Interest Group Ada (SIGAda). Mr. Kermit Terrell will be a presenter. There will also be an E&V briefing. Mr. Szymanski will present an overview of the E&V effort detailing the Reference System.

Recent events include a direct mailing. In one case, the mailing was to the holders of the ACEC informing them about the tutorial which will give them a chance to speak firsthand with the developers. An additional mailing of over 100 letters went to the holders of the pre-ACEC to inform them that a new product is on the market.

Because the AJPO is being reorganized and is in a period of transition, the directorship will be rotated every two years among the services beginning with the Army. Apparently the plan involves each service providing a full-time deputy who will rotate into the position of the director. This plan may be effective within the next six months but no sooner. Possibly Dr. Solomond (Army) will be appointed as the first director.

Concerning contracts, an effort is underway to bring Mike Burlakoff on the ACEC team as the IBMC agent. The CAIS Implementation Validation Capabilities (CIVC) program recently completed a Critical Design Review (CDR).

Another current issue is the stringent Air Force requirements on the ACEC. Mr. Szymanski has been tasked by AJPO to investigate this matter. It has been discovered that since the ACEC is an Air Force developed program, it is subject to the strict Air Force requirements as opposed to the Ada Compiler Validation Capability (ACVC) which was not subject to the same restrictions. The AJPO would like to have the ACEC distributed along the same lines as the ACVC, and an investigation is being conducted to determine how this can be done. However, Mr. Szymanski estimates that the distribution problem will not be solved for several months because the National Technical Information Service (NTIS) is the distributor for the ACVC, and it takes several months to enter an item into their inventory and get it distributed.

Mr. Szymanski and Lt. Marmelstein attended an E&V briefing at the Army Material Command in St. Louis, Mo. and presented a briefing on the ACEC. The Army is interested in the ACEC as they are trying to select compilers for their entire inventory of computer hardware.

The public report Volume IV is at the printers.

Mr. Szymanski's and Dr. Lindquist's papers for Ada Spain were rejected. An E&V paper has been accepted by the National Aeronautical Electronics Conference which will be held in Dayton, Ohio in May 1989. Currently, a paper on E&V has been submitted to the Washington Ada Symposium (WAdaS) but acceptance is still pending.

Mr. Szymanski introduced a new member of the team: Jay Ferguson from the National Security Agency. He is now the head of their Ada program and is currently a member of the Ada board.

1.4 The Xinotech Program Composer

Todd Armstrong
Xinotech Research

Xinotech Research is developing a tool called the Xinotech Program Composer which is a syntax-directed editor used for the creation of a variety of programs including Ada. Mr. Armstrong discussed issues to be considered when evaluating a front-end tool for program construction, and gave a basic demonstration of the Composer.

Currently, the tool typically being used in this stage of development is a text editor. A new class of tools emerging which would be a more intelligent editor is the syntax-directed editors or the language-sensitive editors which tend to have slightly different attributes. The criteria to be used in evaluating such a tool requires a significant departure from the suggested criteria in the E&V Guidebook for evaluating a text editor. This new criteria must assist the evaluator in determining the amount of knowledge that such a tool has about the language that a programmer is working with and must help the evaluator to determine what that knowledge enables the tool to do. These tools also tend to cross some of the typical boundaries that one might think of in using a text editor for program creation. It is then also appropriate to examine the other capabilities besides program editing that may be provided by such a tool.

When looking at tools used for program construction, it is important to determine the underlying technology of such a tool which will determine to a large extent how much knowledge the tool has about the language and its capabilities in relation to that knowledge. Most of the intelligent or language sensitive editors available today are simply text editors to which various functions for template generation, auto intention, or background parsing (which is the capability that can be invoked within the editor without going to a compiler to check for syntax errors, correct them, and then request a compilation), have been added. These text-oriented tools contain an inherent shortcoming in that they lack a real knowledge and understanding of the language, and therefore, lack the ability to make this knowledge continuously available to programmers during the program creation process.

The Composer is a departure from today's intelligent text-oriented editors as it is an abstract tree-oriented editor. The Composer manipulates programs based on its understanding of the language that the program is written in. The Composer is able to understand the language because it is fed an external BNF-like definition which is processed through a separate tool to give it the capability to parse the program while it is being entered. When manipulating a program within the Composer, what is actually being manipulated is the abstract tree which is displayed by unparsing it with the appropriate syntax.

The Composer is a language-independent as well as a language-based tool. Besides working with Ada, the Composer also works with any number of other languages. All that is required is that the external language definition for that language be made available to the Composer in the required form. There are a number of advantages to this even if the only interest is in programming in Ada. For example, language independence allows for embedded design language such as ADADL or Byron to be added to a language definition and to be recognized by this tool. Language independence also allows setting programming guidelines. For example, the language definition can be modified to require at a specific site that all parameters be passed by associations or all loops at one exit loop.

The ability to have a tool be both language-independent as well as language-based is based on the fact that the representation of the knowledge of the language comes externally. The Composer is written in a language-independent fashion and uses the external language definitions to determine how to create templates and how to display the program. The tool is language independent and it becomes language dependent when a specific language definition is read in, because at that point the information of the language has been fed into the tool and it has been determined as an Ada program composer.

The Composer runs on the PC, XTA, and TPS family under MS-DOS. In addition, it runs on the SUN workstation, on the Apollo workstation, and on the VAX family of computers on the VMS.

Depending on how the Composer is configured, the initial installation under MS-DOS requires around 1.5 to 2 MBs. However, after initial installation, it is possible to remove whatever language definitions are not intended to be used or whatever compiler interfaces will not be used. The minimum configuration is estimated between 1 and 1.5.

Since the Composer is language independent, the process of customizing or updating the language definitions is limited to modifying the external language definition. Languages other than Ada that are currently defined are Ada with ADADL, Ada with Byron, the Byron template language, ISO Pascal, Turbo Pascal, VAX Pascal, MicroSoft Pascal, Modula-2, CMS-2, CMS-2M Plus, ANSI 74 Cobol, and ANSI 85 Cobol.

Dr. Lindquist asked if the Composer is written in C. Mr. Armstrong replied that the Composer is written in Modula-2. Basically the Composer manipulates programs in abstract parse trees. In abstract parse trees, the only information maintained is the user-supplied information about the various components. The abstract parse tree does not contain the syntactical elements of keywords, reserve words, delimiters, etc.

In the language definition, the grammar of the first language and the view of the first language allows the user to get the language in and manipulate it. The user adds to that a representation of the second language which maps as much as possible of the first language into the second language. This is only feasible to the extent that the first language is a super-set, or the second language is a super-set of enough of the first language that it makes sense financially to do so. If it is grammatically correct in the second language, the user can read back into the Composer and edit. If it is not correct in the second language, the user has two options. One option is to create an intermediate language. The other option is to use a text editor as an intermediate transition tool to remove the language for something that is correct.

C Plus adds explicit tape transfers. It still has typecasts, but Xinotech is in favor of using explicit type transfers and various other items that might replace some of the items that are context sensitive in C.

Other problems attributed to C are macros, from the point of view that it only makes sense to define macros that are logical units. They have not been a problem because they can simply be read in as an identifier in that position. Initially, C Plus seems to be a language the Composer will work well with. However, the language has not been defined for it as there has not been any impetus from outside the company to do so.

At this point, Mr. Armstrong gave a demonstration of the Composer.

The basic components of a procedure are the name, the parameters, the declarations, the statements, and the exception statements. Various components can be moved through using the Composer's text structural cursor command and "previous" can be used to move backward.

When doing cursor traversal, the Composer uses an area cursor to highlight the constructs of interest. This is because the Composer always manipulates logical constructs within the constructs of the program. Therefore, the area cursor enables the users to highlight the construct that they are working with.

The "in" command allows users to go into the construct that is contained within another construct. The user can go into the first statement and can go in again to reach the expression in the "if" condition. The user can also go in again to move to the first term of the relation, and when doing cursor traversal in the relation, the Composer will move from one side of the relation null expression to the other. This demonstrates the advantages that structural program manipulation gives the user in terms of understanding program structure. The user is able to reach any construct within the program using simple structural positioning operations. More visually oriented commands allow the user to go to the first token in a construct or to the last.

The Composer is always attempting to help clarify the program structure, and one way it clarifies is by abstracting unnecessary levels of detail. The "in" command can be used to open a summarized procedure and when moving through the procedure, it leaves the current level of interest, and the Composer will summarize it again. These types of abstractions can also be used when one is traversing through a program and going out. An infinite level of abstraction is maintained dynamically. The dynamic summarization feature can also be used to help the users better understand the scope that they are working in.

Mr. McKee inquired if the color to highlight changes automatically. Mr. Armstrong replied that a separate program is used for configuring the number of items, and then it automatically chooses the summary. When an item gets summarized, it is automatically put into a different font with three dots after the summary so that one can immediately see that it is summarized.

Dr. Fainter asked about the system supplying its own indentation. Mr. Armstrong said that the indentation is specified in the language definition which is currently provided with the Composer. In the near future, the langurage definition sources will be available to users, and they will be able to change all of that type of information.

The Composer also uses placeholders which are empty indicators of positions that may be refined later as a way of classifying program structure. The Composer has both optional and required placeholders which are displayed in different fonts. The highlights are referred to as fonts rather than as colors, as the system also runs on monochrome and uses different monochrome fonts to achieve the same highlight effect as would using colors.

At this time the presentation turned to a syntax-related discussion.

With the Composer it is not possible to have a syntactically illegal source code on the screen. However, if a user enters, for example, an interdict construct, the Composer automatically enters the key words of that construct. Also, it is not currently possible to read in a syntactically incorrect program. The Composer parses as it reads in the program, and if it reaches a point that is syntactically incorrect, the Composer will inform the user what is wrong at that position and the user will need to exit the tool and correct the problem. Xinotech views this as a shortcoming of the tool and is addressing the problem.

Required placeholders will appear in the user's output text file unless "no placeholders" is specified, and basically the Composer will output a form of a program that is not correctly compiled. In Xinotech's opinion if the user has not finished filling in the placeholders, the syntax has not been completed. Users can compile a program that does not have all the placeholders completed; however, they will face the same problems as if they attempted to compile this program and it was written in a text editor and they did not have the type name--it will not compile. The user will receive a message, however, that the program will not compile because the placeholder is an invalid type name or the type name is not there.

Placeholders are displayed locally when the user moves through a construct, or they can be displayed globely. The user can specify that all or none of the placeholders are to be displayed. If no placeholders are to be displayed, then no placeholders would show up in the file. Normally, the Composer will display required placeholders all of the time.

The Composer, in addition to being language independent, is representation independent meaning that the user can define multiple, alternate ways of viewing a program written in a given language by breaking the language definition into two separate parts. The first part is called grammar, and is A, B, and F definitions of the language. The second part, called the views, is the external representation of the program that is used in the process of unparsing the abstract parse tree, and the Composer allows the user to have multiple views for any given grammar.

Multiple representations allow users to perform a number of tasks. For example, it allows users to define multiple formatting schemes for use in a project development. Users can define one viewer representation which contains the project standard layout for all of the constructs and how everything should be formatted, and multiple alternate views can be defined which allow things to be formatted differently so programmers can use whichever formatting scheme they feel most comfortable working with. With the Composer's multiple representation capabilities, users can switch between one representation and another by using a command that unparses the tree using the representation.

Multiple representation schemes also allow the user to define views that may enhance readability and that can be used for conversion of programs from one structural level to another.

The most obvious advantage of the Composer when manipulating programs is it prevents the user from making syntax errors. The problem of being able to modify a program in such a way that it becomes invalid by doing deletions or cut-and-paste operations does not arise because the user cannot position on an end loop clause and delete the clause to cause it to become syntactically invalid.

Deletion is a special case of the cut-and-paste operation. The same cut is performed, but without specifying a destination for cut-and-paste. The Composer enters an explicit cut-and-paste menu that allows the user to collect other items into the buffer, as well as perform various other operations until the user chooses to empty the buffer.

When collecting items, users position on the item to be collected into the buffer. The user does not have to be concerned about getting the entire item, provided he is positioned on the desired item. Similarly, if a user attempts to do an invalid cut-and-paste, the Composer will explain why it cannot be done. The Composer uses the vocabulary of the language that users are working with to clearly indicate to them exactly what the problem is with what they are trying to do.

The Composer keeps an unlimited history buffer that will allow users to undo any of the commands that they have performed since their most recent save, and if desired, allows them to redo those commands.

Insertion is accomplished in the Composer in a similar manner to cut-and-paste in terms of positioning. One factor to keep in mind when doing insertion on the Composer is that the amount of knowledge that the Composer has about the language enables it to provide the user with a large amount of help with the language. The Composer can significantly speed up the insertion process by providing the user with templates and help menus. It can also understand to some extent what users are trying to do based on the context they are working in.

Another form of insertion that is available is nest insertion. This allows the user to basically perform the same type of insertion, but adds the capability of placing new constructs around existing constructs.

Mr. Armstrong distributed the following material.

- A brochure about the Composer and its abilities.

- An evaluation criteria received from the Naval Air (NAVAIR) Systems Command software support activities group that was used in evaluating off-the-shelf commercially available syntax-directed editors for use in a programming environment.

- A functional specification checklist for the Composer which describes the basic functionality of the Composer, omitting the basic functionality that one would expect in any type of editor whether a text editor or a syntax-directed editor.

- A comparison of the Composer's functional specification to DEC's Language-Sensitive Editor (LSE) was created by going through the Composer's functional specifications and determining which items are available and to what extent in LSE. This is intended to demonstrate the value of a checklist in determining what tool features are available in the process of evaluating a tool.

The functional requirements that NAVAIR specified in their syntax-directed program editor specification were separated into different categories. Basically, the language management category specifies that the tool should be able to work with more than one language. NAVAIR's main interest was that they are intending to use such a tool for both CMS-2 and Ada. They are also concerned about being able to enforce standards such as DoD 2167A and being able to automatically ensure that these standards are followed.

Multiple formatting capability was also specified, as well as the ability to handle design and documentation languages such as Byron or ADADL. Xinotech feels that a lot of these language management issues cannot be appropriately addressed by a text-oriented editor because it does not have the underlying comprehension of the language that an abstract tree-oriented editor has.

In relation to the integration into the programming environment, NAVAIR specified a user definable compiler interface allowing the syntax-directed program editor to get all compiler related information from the compiler and make it available during the program editing to the user. They also specified that project directory management should be included in the tool to allow the tool to automatically store and retrieve files from specified directories. The tool is also specified to have a directory system available from within the tool. The Composer implements this by defining language definition which mirrors the tree structure of most directory systems and reading the directory on the host into a program that is written in that language and allowing the user to traverse the directory as a program tree. The user can then do deletion operations on the program tree, and eventually will be able to perform other operations on the program tree such as move or copy.

The Composer on the PC allows manipulation of programs of any size by implementing its own virtual memory segmentation scheme. This allows any of the summarized constructs to be swapped out if they are not currently being viewed. Mr. Armstrong turned the floor over to questions from the team.

Dr. Crawford asked for a summary of major advantages. Mr. Armstrong stated that the syntax error prevention is definitely one of the major advantages. The ability to understand the language is the basic advantage. Multiple representations is an important feature as it allows the program to be represented in a number of different ways depending on what the user wishes to do with the program for any specific use.

Mr. Weiderman asked if there was any empirical evidence that it is beneficial to prevent the user from entering incorrect programs. Mr. Armstrong stated that although there is no empirical evidence, according to user feedback, the Composer saves a lot of time compared to doing things with the text editor.

Ms. Hazle asked how long the product had been on the market. Mr. Armstrong said the Composer itself has been on the market since approximately 1 January 1988. The development of the product started in 1983 and various data test versions of the product were used at a number of large sites in limited use for at least a year or two before release.

Ms. Mulholland stated that the Composer is achieving wide distribution throughout the Navy. A major purchase was made by the Naval Defense Center and is being distributed to the seven naval weapons centers.

Mr. Ferguson asked if more compilers would be integrated in. Mr. Armstrong replied that the process of integrating a compiler to the Composer is relatively straightforward. When a compiler is integrated to the Composer in-house, it takes a couple days to do. The error message test and position information that is produced by the compiler is converted into a compiler independent format. An external command procedure is used to invoke the compiler so that users can easily customize what that unit does.

Mr. Francl stated that one of the problems with buying a system like this is that the coding base is such a small part of the development item. He asked if Xinotech is planning on developing an integrated graphical design tool of some kind that will interface with this. Mr. Armstrong answered that Xinotech's main interest as far as graphical design tools is more along the lines of being able to properly integrate the tool with other people's graphical design tools. Xinotech has a number of graphical design tools which they believe are fairly good products, and rather than Xinotech trying to create another one, it would be more in their best interest to interface properly with those people's products.

1.5 Effect of Tasks and Exceptions on Execution Time

Dr. Robert Fainter
Arizona State University

Between June and December of 1988, Dr. Fainter was involved in an effort for McDonnell Douglas under contract with McDonnell Douglas Helicopter Company dealing with the evaluation of an Ada compiler. His presentation focused on the results of that evaluation.

The ACEC suite was not used in this effort as it was not yet available. A suite was developed to look at the effects of the presence of tasks and exceptions on execution types of programs. The compiler evaluated was the Verdix Version 5.5 Ada compiler running under VADS, APSE, and MicroVAX which ran 4.7 of VMS. The hardware used was the Delta 5 1750 architecture simulated with a real-time applications interactive debugger called RAID, and running a MicroVAX II with a DEC NET environment.

The programs were actually run on the VAX but the RAID debugger gave instruction times appropriate to the Delco system. The report provides execution time of programs in nanoseconds. These execution times are computed by the debugger; they are not actually measured from the hardware.

Nine experiments were run which generated a number of different programs for some of the experiments. Some of the experiments only took a couple of programs, but in order to look at the overhead of task presence, a control program was used. This included three different programs declaring one, five, and nine tasks and looked at the execution time of those programs.

The control program was the null program. The experimental programs themselves did nothing except declare and elaborate the task which had only the null statement in it. So only the task start-up overhead was being measured.

Experiment two involved measuring the cost of task scheduling, and was very similar to the first experiment, except experiment two included some code in the test programs themselves. In the control program, a number of Ada statements were sequentially executed. In each of the task programs, those statements were divided up and executed in the tasks which resulted in the same effect after the entire program ended, with the same values computed the the statements were distributed into the tasks.

Experiment three examined the cost of parameters in rendezvous. This involved looking at the numbers, types, and modes of parameters.

Experiment four studied the cost of the numbers of entries into a task. This was to determine if increasing the number of entries caused an increase in execution time.

Experiment five looked at the effect of the delay statement. This dealt with how much overhead it took to have a delay statement there and to get an idea of whether or not the delay statement was really delayed the duration specified.

Experiment six looked at selective wait.

Experiment seven involved timed entry calls and conditional entry calls. In combination, experiments six and seven studied the effect of the select statement. For selected wait, the task itself was examined; while in the case of timed entry calls and conditional entry calls, the calling procedure was studied.

The first seven experiments dealt with tasks. The last two experiments dealt with condition handlers. Experiment eight measured the effect of the presence of conditional handlers. Some programs were written with handlers but no conditions were raised to determine how much extra processing time would be needed to have handlers in the programs.

Experiment nine examined how much it cost to handle a condition. The interest here was to determine the length of time it took to handle a condition when the condition was handled remotely from where it was raised. It looked at nesting depth, handling a condition that is raised within a package.

The remainder of the presentation covered the results of running these experiments. These results were presented in a series of graphs. The indication was that execution time goes up steeply with the increasing number of tasks. The inference from this data is that the relationship appears to be linear.

Lt. Marmelstein inquired if the timing was just of execution time. Dr. Fainter replied that the program receiving an executable image was compiled with Verdix disassembly tool under VADS. The resulting assembly code enables the location of particular instructions. There are two branches to get to the first executable instruction. In the case of both tasks and programs with exception, a few instructions are executed in the program and then branches back into the run-time code to set up the tasks or exceptions. Dr. Fainter found the first executable instruction and began timing at that point. He then found the last executable instruction which is the program's final return into the run-time system for termination and stopped timing at that point. Therefore, the timing is from the first user's instructions to the last user's instructions.

The time for task elaboration is included in the timing measurement. The entire image is from linking to the run-time environment, and the branches that the run-time system takes can be traced. The tasks do not begin elaboration until after the first instructions of the user program. Therefore, the timing result includes the elaboration time.

In responding to a question concerning the tasks, Dr. Fainter said the tasks did nothing other than start and end. The single source instruction was the null statement because of the overhead of start-up. Dr. Fainter was asked if this was fully optimized. He replied that in examining what was generated, it was very clear that tasks were elaborated. A good optimizing compiler might be able to see a null statement and just optimize it out. This compiler did not.

Dr. Fainter was asked to explain the difference between the two lines on the graph. He stated the difference between the two lines is the amount of time that it took to execute the code in experiment two. The graph is a summary of a comparison of experiments one and two. In experiment one, all of the programs had the null statement only. In experiment two, some representative code was taken, which was navigation computation used in some of the helicopter systems, compiled and executed then divided and split into the programs.

The important point is the fact that adding the code did not cause any divergence in the time that the two test cases took. Another point which can be derived from this is that there is no extra penalty for using tasks, but when computation is done, the scheduling of the tasks does not cause any extra penalty in this particular compiler.

Mr. McBride asked what was actually being measured in the interpretation of the five tasks that came in just over 90 milliseconds. Dr. Fainter replied the measurement was of the execution time of the program that declared five tasks of the total program duration.

Dr. Crawford stated that the main conclusion would be comparing five with nine. The extra costs of starting up the four additional tasks is about seven milliseconds. The main factor is the cost to start up tasks. The cost of starting the task is the only cost incurred for using tasks, and once the task is started there is no additional cost.

Mr. McBride asked if this was really elaboration time. Dr. Fainter said this was total user program time from the first instruction of the user's program. Mr. McBride asked if this was expected to be linear. Dr. Fainter answered that it was. In various compilers the relationship once above one task could be logarithmic which is acceptable; linear which is probably the most common case and is acceptable; exponential in which probably the tasks would not be used with that compiler. The emphasis is, that once the code is actually put in and had a task program that was doing something without rendezvous, there was not any extra cost other than start-up.

Ms. Mulholland stated that the implication is that it is static. Dr. Fainter agreed. Ms. Mulholland asked if a family of tasks, task types, or straight tasks were used. She wondered if the features would have a difference. Dr. Fainter replied that there may have been a difference. They did not look at task types or families of tasks.

A graph was shown representing what happens when one task with one entry and the parameters to that entry were of type integer. All the parameters used were type integer. The results indicated that the number of parameters clearly causes increasing execution time. It is no surprise that the number of parameters cause the execution time to go up because of the way parameters are typically handled. In groups of one parameter, comparing "in", "out", and "in/out", in each case the "in/out" parameters take proportionately longer; the "out" parameters take the least time, and the "in" parameters take an intermediate length of time. Dr. Fainter stated that the one entry and the one task with parameters of those modes and numbers is called only one time.

Mr. McKee asked if there could be a standardization by having a rendezvous call with no parameters. Dr. Fainter stated that was not done in this case.

Dr. Lindquist asked if any other kinds of parameters were used aside from integer time. Dr. Fainter said both an enumeration type and an array type were used.

Dr. Lindquist asked if the numbers were exactly the same and if the array type was considerably different. Dr. Fainter answered that the numbers were not the same. The array type was different. The enumeration type and the integer type were exactly alike which was expected since the enumeration type was probably implemented as an integer. The array was being passed by a sort of reference mechanism or that the mechanism was being used for the various modes when an array was used. It was not the same for integers and enumerated types. This is the case in the enumeration time. If comparing this to an integer, it is exactly the same. The numbers are exactly the same even down to the nanosecond, but with the array, there is no variation at all across the mode, but there was the expected variance across the numbers of the parameters.

Mr. Francl said this seemed to be a series of experiments based on purely empirical measurements. Some of the questions could be answered by looking at the generated code. Dr. Fainter said that to an expert the questions could be answered by looking at the generated code. An underlying reason for the effort was the company's interest in getting a suite of programs that would allow the comparison of several compilers. The compilers could be compared by looking at the generated code but it was felt that it would take longer and involve more work than with a set of test programs where execution time could be seen.

This compiler was validated under one of the validation suites. The report points out that this compiler has been validated. That is the criterion the company uses that a compiler is validated according to a suite which is as far as the validation went. The types evaluated are the scaler, array, and user. "User" in this context is used as equivalent to enumeration. The array evaluated the same according to the modes. The scaler- and the enumeration-type cases receive exactly the same numbers.

Another graph had one task with a varied number of entries and two parameters per entry. Tasks one and five increased in the execution time of the program which declared one task and five tasks. Although there was a difference in that, there was no detectable difference between the programs that had five entries and nine entries. Looking at the numbers of entries on the horizontal axis and between one and five entries, there was an increase in execution time, but between five and nine entries in the task, there was no increase in execution time.

In looking at the delay statement in the next experiment, there was a program that did not have a delay statement and one that had a delay zero. There was a rather large difference in the execution time of the two. A team member stated that while looking at the ACEC it was suggested that a delay zero should not be a schedule point. It is not a language requirement. Dr. Fainter replied that the language manual does not require a delay to cut off at the duration. It merely has to delay at least that long.

There were programs with zero, one, five, and nine tasks that did not rendezvous. The duration is of zero, one, five, and ten seconds. Basically, there did not appear to be an interaction between the number of tasks and the number of delays, or the length of the delay statement. For this compiler, it was not believed that an interaction between the numbers of tasks and the duration of delay statements occurred.

The next graph covered the results of using a select statement. It represented the time of putting a selective wait into a task, and in the case of no selective waits there was a program with a task and an entry. The entry call was made and the length of time was measured. In this case, there was a selective wait statement in the task and there were two branches in the selective wait. The entry call was made to one of the branches and the time was measured. The difference in time was the overhead of having the selective wait.

In using the select statement for timed and conditional entry calls, there was a task that had an entry, and in this case, called that entry without a timed entry call or a conditional entry call. The conditional entry call took more time than the timed entry call. The timed entry call was set up with a ten-second delay in the delay leg of the select statement. It could have waited up to ten seconds for the entry to be accepted. The design of the program was such that the task was waiting when the entry call was made. There is no real explanation at this point for this without looking at the code generated as to why the timed entry call took less time. There was a feeling that more overhead should be associated with the timed entry call with no sort of condition at all around it. The surprise is not in the magnitude of the difference but in the fact that the difference is negative.

The final two graphs represented condition handlers. Once an exception handler is put in, there are no other penalties paid for including that handler providing none of those exceptions are raised.

Mr. Gicca asked if levels of nesting were also investigated. Dr. Fainter answered yes along with investigating the cost of the handler being without the condition raised. Ideally, the program should not raise the condition. Many times the handler will be there and never get executed.

A condition was declared in a main procedure and then within that procedure nested other procedures, one, five, and nine levels deep. Then at the deepest nesting, the condition was raised with the handler at the global area. It would appear no matter how deeply nested, at least for levels one, five, and nine, there were no increases in execution time as it processed through the extra levels of nesting.

A package was written that declared and exported a condition. From a procedure within that package, the condition was raised, but the procedure in the package did not handle the condition; it merely raised it. In that case, it took less time to handle that condition when the condition was raised and declared in the package and handled in the program that "withed" the package. It took less time to handle that condition than it did to handle conditions that were nested and were actually local to that declaring program.

For all the examples the same user declared exception was used. Mr. McKee stated that a key question is what is the cost of propagated exceptions. Dr. Fainter answered that if the exception is raised within the structures of this compiler, the differential cost is zero for the case tested.

Mr. Eilers suggested looking at the generated code to provide a better confidence level in the conclusions. Dr. Fainter agreed. Time constraints was one reason it was not previously performed, and another reason was the company wanted a tool which could easily compare several compilers. If looking at just one compiler, however, the code should be examined line by line. Dr. Fainter then closed his presentation.

1.6 Prototype Update

Major Patricia Lawlis
Arizona State University

Major Lawlis provided the team with a brief overview of the prototype form which is a first version of the prototype. The team was asked to review it and provide some feedback.

The first version tests the concept only and is very basic. It is not particularly efficient but it does not have to be. The main factor is it does not require a very sophisticated user; it is intended for a program manager type. It is also intended to be meaningful enough that it can be expanded for a more sophisticated user providing more information to anyone who wants it.

The prototype has a reasonable user interface and shows the basic selection process. It has only six compilers in the database at this point and the detailed levels are missing. The basic process idea is still there which is enough to provide the user with an idea of how more details could be added. There is also the ability for additions so that the user can receive more information from the system at the end. Some documentation on the prototype is available: two user's manuals are available on each system, in addition to some on-line documentation.

Mr. Szymanski thanked Major Lawlis for the update. He asked the team to consider the basic premise that there are no more Ada technology problems and, therefore, groups such as the team should be disbanded. He asked the team to be ready to give their opinions on this the following day. He then dismissed the General Session for the first day.

2.0 WEDNESDAY, 22 FEBRUARY 1989

2.1 E&V Reference System Status Report

Dr. Bard Crawford
The Analytic Sciences Corporation (TASC)

Dr. Crawford updated the team on the Reference System status, and model project/whole-APSE evaluation, and provided an example of a new Guidebook entry. Recent activities of the Reference System included final approval of Version 1.1 of the Reference Manual by AJPO and the Public Affairs Office. The document has been sent to all team members and is now being distributed to others on the mailing list. Mr. Clark has conducted some experiments regarding placement of a version of the Guidebook on electronic bulletin boards. The only technical problem at present is with the tables. Checklists, an important element of the Guidebook, were discovered not to automatically translate from the interleaf system into an ASCII set.

A one page announcement was placed on the Ada Information Clearinghouse Newsletter in January. At the beginning of February, the same announcement could be found on other bulletin boards. Major Lawlis placed the announcement on Ada Info and Alan Impicciche placed it on the JIAWG bulletin board. This effort resulted in a surge of document requests.

The total number of names on the current mailing list is 180 and includes 30 just recently submitted since the bulletin board announcements went out. There are 17 requests from contractors, 6 from Government (mostly Air Force), 4 from universities, 1 miscellaneous, and 2 from overseas.

Fliers were distributed at Dale Gaumer's seminar on Ada tools and environments. They will also be handed out at SIGAda/AdaJUG and at the NSIA meeting. Dr. Crawford attended the seminar on Ada tools and environments by Dale Gaumer. One interesting area discussed was evaluation of compilers. The paper included some material that was in the Magnavox proposal for the ACEC contract. Mr. Gaumer has been tentatively invited to speak at the June meeting.

Version 2.0 of the reference documents should be publicly available this summer. The plan is to provide some material to the Classification Working Group (CLASSWG) at this session with additional information to follow in June. The sum of the two meetings will be all the new items intended for Version 2.0 so the working group can do a thorough review. Immediately after the June meeting it will be formatted for the start of the approval process.

A number of improvements in reorganizations and additions are planned in various places. Gary McKee, for example, carefully reviewed the Reference Manual and the Guidebook relative to its discussion of CAIS types, CAISs, interface systems, and operating systems. Some good suggestions for improvements were made and will be on the agenda for this session.

One general theme which will continue to receive close attention is the whole-APSE issues as reflected in both documents. Among the questions asked are: What is an APSE?; What are the important issues about a whole-APSE?; What technology is needed and what is available in the form of checklists?

Model project activity was discussed in relation to whole-APSE issues, and has been discussed in the two or three REQWG meetings. Some activity has begun in this area under the current contract. [illegible] reason for starting the model project activity was that an appropriate individual who is experienced in using Ada and familiar with the difference between 2167 and 2167A became available to work on it

The Inertial Navigation System Simulation which SEI is already using was picked for the model project. This system comprises a couple of programs. A subset will be selected and run on one computer making it a model project.

The first job will be to become familiar with the 2167A format, and write the corresponding documents plus one or two others. A software test plan will be included. At a minimum, the beginning of a model project will be the basis for whole-APSE evaluation efforts in the future. The idea is to begin building by defining some assessment scenarios.

Included in Chapter 13 of the Guidebook will be the example summarizing in two pages an answer of E&V technology based on work sponsored by the Naval Air Development Center (NADC). The focus is on the early phases of the life cycle, particularly requirements in analysis and process of generating the requirements, software requirements specification, and evaluation of several CAIS tools. An experiment was built around the project and three tools were evaluated: CASE 2,000 by Naste which supports the Ward-Mellon extension of the Gordon construction design method; Teamwork by Cadre which supports the Hatley extension; and TAGS.

Initially a fourth tool was also chosen, DCDS by TRW. An initial list of 130 tools was formed to evaluate what might be applicable. An initial survey was done based on advertising literature and what could be learned and found by going to Ada shows and meetings, etc. The list of tools was narrowed; demonstrations were given; four tools were selected; and then the experiment was conducted. It was then discovered that the DCDS would not work. Therefore, after surveying to pick four tools, one of the four would not run on the system. The fourth tool was then dropped and the other three were evaluated. The interest in running the four tools was in the real-time critical area and capturing the requirements.

The areas of concern were divided into three major areas: method, automation, (how well the tool supports the method), and the vendor itself (how well is the product supported, how good is the documentation, etc.). In each of these areas was a number of evaluation criteria. A list of question with a yes or no answers was made, with a positive answer indicating the feature was there. An interesting issue for E&V is a tool cannot be evaluated sensibly without also admitting to evaluating the method that the tool is attempting to support.

One example of displaying results was the quality of documentation. This consumer report type of display was used to summarize the results. If a tool did not have any feature at all, it received a zero, but if it had a feature it received a score of 1-4. This provided a quick visual look comparing any one of the 13 areas. The tool's comparisons were compiled into one summary; all three of the tools received a score of two out of four. The end result was more than just the summary papers; write-ups summarizing the answers to the questions and the reasons for the scores were also produced.

As mentioned previously, the first interesting factor was that one of the four tools did not work. The second factor was that in all three cases there were instances of the tool not performing as was advertised. One of the lessons was that there is no substitute for getting the item in-house and having the people who will be using it later check it out. The third interesting factor was that in the final analysis, TAGS was deemed the best system; however, the reason was not because it was necessarily a better system, but simply that the TAGS system's method was appropriate for the application made. In time-critical systems which is a major element in the system to be built, Hatley and Ward-Mellon extensions to the traditional structure design way of doing things is not the answer in the opinion of the people who conducted this investigation. The methods that were implemented by TAGS were more appropriate being that those methods involved a different kind of portrayal of the stimulus response of a system and what its quantitative timing requirements are. Because that essential difference existed from the beginning of the experiment and was weighed very strongly in the final analysis, the TAGS system was much more appropriate thatn either of the other two systems.

Mr. McKee asked if what Dr. Crawford was saying was that they started off with the assumption that TAGS methodology was the best one and then proved it right. Dr. Crawford replied that they started off with the assumption that it was very important to capture the time-critical nature of the system requirements and carry that through. Dr. Crawford found that all three tools came out roughly equal in the second and third areas. They did a reasonably good job of supporting the method support advertised although there were holes in the CASE.

Separating tools from methods is a difficult task and an important factor to keep in mind. This activity should not be turned into one that evaluates methods and software processes, but there is difficulty in evaluating any tool. Some words on this issue may have to be put in the Reference Manual or Guidebook.

Ms. Mulholland stated that the purpose of E&V technology is to be truly independent of prejudice, truly supportive of the best technologies and that the items put in the Reference Manual and the Guidebook should not be everything that is available. Only those studies that the team has evaluated and feels are non-prejudice or good technical discussions of the technology should be included. Based on this, she would oppose putting the SPS reference in until the team evaluates it. The team needs to evaluate the study before referencing it to demonstrate as much as possible that there was no incredible bias or, although they had a nice approach to the study, that they were lacking in capability on this particular area which biased their results. Some filtering criteria is needed.

Dr. Crawford replied that inevitably in the choices of what to report and what not to report, the team will have to reflect their own opinions. The group working on this model project stated very clearly that the results obtained were due to their emphasis on the time critical factors. The results might be very different for another type of application. The group was very up-front about the intended application, and they communicated that in the report.

Mr. McKee said that given that this is a fairly competent company with respected people who came up with such an obvious methodology slant, maybe a section should be added in the reference manual saying that here are some characteristic problems when doing evaluation and discuss the impact that methodologies have on whether the tool works or not. Rather than denying the possibility of bias, it might be more appropriate to warn the Reference Manual users about the potential problem and provide the team's opinion.

Mr. Szymanski said that one has to look at this document and the Reference Manual from the point of balance where the other reports do not emphasize so much methodologies. The team owes it to the readers to present all approaches to this and because this approach is somewhat different maybe it deserves recognition in the Reference System.

Ms. Martin stated that so far other items included in the Guidebook have not undergone a careful evaluation by members of this group to determine whether or not they are good, sound studies. No set criteria has been used to objectively determine whether or not a the job was performed well and whether or not to report that. She expressed concern that unless a very good criteria is used there will be exclusion of items. If the team starts making these decisions without a very clear criteria, it will create a problem. She agreed that it is a very controversial area that has not really been addressed. So far the criteria has been to include everything.

Ms. Mulholland stated that the amount of items included, even if everything is included, is still going to be a small subset of what is available. She does not want the team's small subset to be the bad subset of the available technology. She thinks it is important that the team forms opinions as to what is to be put into the Guidebook because of the situation.

Mr. McKee asked that if at this point any possible entries have been eliminated from the guidebook, or if there have been any entries received that Dr. Crawford does not intend to place there.

Mr. McKee stated a preference for the idea of including everything that comes to the team. It seems as though filtering would not be a good idea at this point.

Ms. Martin stated that at this point the group has really been begging for people to provide items for inclusion. There is no overabundance of items to report. Most of the Guidebook consists of checklists the working group is creating because there is nothing there otherwise. She stated a preference for the idea of reporting all entries.

Mr. Terrell said that as a user rather than a producer he would prefer to see a reference to biases, etc., than to see no reference at all.

Mr. Szymanski suggested tabling this discussion for this day's schedule and to let CLASSWG continue with it and come up with a few recommendations.

2.2 Ada Compiler Evaluation Capability (ACEC) Status Report

Kermit Terrell
Boeing Military Airplane Company

ACEC has had its first release and work is presently being done on the second release which will come in two parts with the first part being extensions which are mainly additional tests. Other extensions will also be done covering compiler diagnostics, debuggers, library robustness, and system analysis. This contract has not yet been signed but an agreement has been reached and it is expected to start by the middle of March. The second release of the ACEC will be pushed back a couple of months and will probably occur around November.

Lt. Marmelstein stated that they had just finished with part one of the Preliminary Design Review (PDR) because there was no contract for the extensions. Part two of the PDR is in progress and should be ready to present about the first of April in Wichita, Kansas.

Ms. Hazle inquired as to the type of feedback received from the users of the ACEC. Mr. Terrell stated that there had not been a great deal of feedback. Some reports indicated that there were very few problems. Some people have run the ACEC and not had any major problems with the results. Very few people have said that this is a bad test or that something should have been done differently. There was a problem on the very fast machines with the timing loop because 16-bit integers were used to make the test as supportable as possible. There was an objection about PRAGMA_INCLUDE as it was thought there should not be a new pragma. Mr. Szymanski asserted that the upcoming tutorial would provide a good opportunity to obtain feedback as ACEC's owners will be able to meet with the developers.

About 40 organizations have the ACEC, but not all of them are using it. Mr. Terrell stated that the assumption was that if the users were having trouble using the ACEC, they would hear about it. The fact that they have heard nothing either means that the users have not tried the ACEC or are not having a lot of trouble using it.

2.3 CAIS Implementation Validation Capability (CIVC) Status Report

Jeff Facemire
SofTech

At the time of the previous status report, SofTech was having to revise its approach due to the decision not to use the Ministry of Defense (MOD) test harness. Prior to the December meeting, part one of the Critical Design Review (CDR) was held in Houston covering everything on the CIVC except the test administration facilities.

Many of the resolutions for part one of the CDR have been completed. In the last two months, SofTech has been concentrating on the approach of the second part of the CDR.

Recent events include:

- Updating the Software Requirements Specification (SRS) to reflect the removal of the MOD test harness. The new requirements were added for the test administrator. This SRS was delivered to the Government.

- Updating test plans to reflect both the changes in the requirements and the use of what is being called a CAIS response simulator. It was found that all of the testing of the CIVC could not be based solely on CAIS implementations because both the normal processing capability of the tests and the error conditions or the negative testing need to be tested. The CAIS implementation is typically going to provide one of those results and so a response simulator has been developed. This is really just a script-driven simulator that will guarantee the expected results so that the tests can be checked.

- Completing a detail design of CIVC test administrator.

- Presenting a CDR in Denver. SofTech hopes to close out the CDR for CIVC pending the resolutions of some issues raised at that meeting.

At the present time, SofTech is in the implementation phase of the CIVC and following resolution of the CDR issues will be implementing the test administrator.

In the development of the actual tests, there are an estimated 1,000 test projections. About 13 tests have been reviewed in-house in Houston and 239 have been approved by San Diego. Since the last report, the San Diego review numbers on the test objectives have written approximately 120. The internal review of the test objectives have written about 100 and the internal development of test objectives has risen about 600. Approximately 100 new scenarios have been developed since the last time but not much review has been done in that area. The concentration has been either on the development design phase of the CIVC as far as the test administrator or the development of new test objectives.
In the test case area, the numbers have decreased. Approximately 100 test cases have been developed with dependencies. Prior to this, tests were independent units, and not a lot of work had been done in the area of dependency analysis between tests. The 50 tests developed are current test class dependencies.

3.0 FRIDAY, 24 FEBRUARY 1989

3.1 General Discussion

Mr. Szymanski gave a synopsis of the background issues which prompted his request on Tuesday for the team member's opinions and input to build a case for or against the existence of Ada technology problems. Mr. Szymanski's local management is no longer supportive of the E&V activity believing that Ada has existed long enough that the technology problems have been solved and only the language needs to be maintained. Although his management believes compiler evaluation is important, they would prefer it if Mr. Szymanski would perform that task in-house.

Major Lawlis passed out copies of "Ada Technology Assessment" (Appendix A), a list of problems and suggestions which she prepared from the REQWG's session. This list was used as a basis for the team's discussion.

Mr. Stiles began the discussion stating that the Ada repository at present is just a place where software is stored, but no evaluation of that software in terms of quality is performed. There is no configuration management of that software for an organization to acquire a definitive report on both the software's quality and its maintenance.

In addition, to determining the quality of the reusable elements, finding out what is available is also important. Tools are needed to support the creation of the repository, such as ways to search through it to find relevant items.

Ms. Adams said that people who work entirely with embedded systems are concerned that they do not have a way to assess the merit of what is available. Their interest lies in reuse and how it affects Ada or how Ada affects it, but they have no way to assess what is available. Therefore, they cannot and will not commit to using Ada. They are looking for some organization to take a visible lead in assessing what is available.

Major Lawlis stated that E&V technology can lead to the use of software standards and the use of reusable software as people will not use it unless they can assess it. In a sense, certain standards are needed in order to establish reusable software which provides a lead-in for management to understand a little more of the importance of the CAIS, and this was the reason REQWG suggested that tools could be moved to a new platform, so management could understand the kind of productivity improvements such a concept could provide.

Mr. Francl suggested explicitly indicating some of the products that the team has developed to demonstrate the value being contributed. Examples of the teams' contributions are the tools which can be moved to new platforms (CAIS), the tools to assess software (ACEC), and the Reference System.
Mr. McKee suggested compiling a report of items left to be done; an incompleteness report rather than a completeness report.

Dr. Crawford stated that fast changing technology is implicit and could be made more explicit. The technology for evaluation and assessment of items of fast changing technology is immature and must continuously change. In other words, the team's role is just beginning as it is to assess items that are themselves rapidly changing. Ms. Hazle said this creates an even greater need for the ability to assess the technology which is changing even as it is being used.

Because of the negative connotation associated with maintenance in Mr. Szymanski's branch, the continuation of ACEC needs to be addressed from a technology development or research approach which would differentiate ACEC from maintenance. Another suggestion addressed the maintenance issue from a different angle. Because one of the roles of Mr. Szymanski's lab is to develop technology for future use over the life-cycle of a system, it is important to distinguish between the lab performing maintenance versus the lab developing technology to improve the productivity of the maintainers.

If new methods such as object-oriented design are a part of the development phase with Ada, there has to be important downstream influences on what that will do to the maintainer's job. They are going to be maintaining systems, and they will have to understand a whole new way of using new tools, including CAIS tools which will have to be developed and evaluated. The transition between the development phases through maintenance has to be dealt with on an integrated basis and is a very important aspect of the overall problem. Therefore, if Mr. Szymanski's branch ignores the issues that the team is dealing with, they will not be prepared in the future to anticipate their customer's needs.

The assessed technology can influence vendors as well as improve the Ada files. Mr. Weiderman agreed that there is a great potential for influence. At present he is concerned about the distribution and use of the ACEC as only five problem reports have been received. This is an indication that it is not getting to the people who need it, or they are not using it as fast as the team would like them to use it. The compiler vendors are not going to spend a lot of time trying to get it and use it unless programs demand its use.

Although JIAWG is not currently considering tools for assessment, Mr. Gicca considers the JIAWG a great vehicle. It is unto itself a tool that may fit into their environment and that would be very useful, partially for advertising. Mr. Brookshire said one reason the ACEC has not been a topic is because compilers are not a topic. It had been previously agreed upon that compilers will be independent of the JIAWG selection of tools because they are too processor dependent.

Mr. Szymanski asked what was the potential use by the Ada Validation Facility (AVF) of the ACEC towards the construction of a formal evaluation facility. The fact is that if and when that happens the ACEC is going to be at least one of the tools that is going to be used for the evaluations. Mr. Gicca said there is a possible need for the groups to define exactly what an ACEC should do. This should be done in terms of taking the broad perspective and saying this is what should be evaluated and then dropping down into some detail across the general categories already defined which will point out what has already been done and, more importantly, point out where the current test suite may be lacking. The future enhancements and technology needs need to be defined to complete the suite to make it more useful.

Mr. Szymanski asked if ACECWG could assess what the team has and create a brief report stating what top level areas are not currently addressed. Mr. Weiderman stated that was an ongoing activity which is done at every meeting.

Mr. Gicca thought that ACEC providing information to REQWG may be productive. ACECWG could analyze the fine points to determine what is missing. Although the global structure concerning the overall direction of ACEC is now a general topic, in the future it would probably be something REQWG would examine. Major Lawlis replied that REQWG is looking at it.

Mr. Weiderman placed the following questions on the view screen. He stated that the focus has to be more on the team than on the technology issues. The first question is: What are the costs and benefits for the team as a whole and for the individuals on the team? One of the obvious costs is the cost of conducting the quarterly meetings which has to be evaluated with the benefits derived from the meetings. This seems to be the number one issue.

The second question deals with the inputs and outputs of the team. What are the team's products? What are team members putting into the mix? What are team members getting out of the mix as individuals, as a team? What are the contractors costing in terms of putting dollars into contracts? What is the team receiving in terms of products from those contracts? There are some very strong arguments there in terms of cost sharing from Boeing, but the team has to look at what is going in and what is coming out.

The third question is: Who are the customers and are they receiving what they need? The customers are the people in the Department of Defense (DoD) or the projects, the programs in the DoD, that are producing weapons systems.

The fourth question is: Where is the team's sponsor and is the sponsor providing the necessary morale and dollar support to the effort. Are they providing the necessary backing?

The fifth question is: Who is making the agenda? Is the agenda being created by the team, or is the agenda being created by an external person?

Six, are the justifications for the team being based on actual needs of specific DoD programs? This relates back to the third question.

Seventh, is the E&V task or team the only way to accomplish this goal? Is it the best way to accomplish whatever goals the team has set? Are there other ways to do it cheaper, faster, or better, or is this the best or only mechanism for achieving the goal? There are times that the government has sponsored APSE development efforts and those APSE development efforts have gone away. Why did they go away? Was it because the need for APSEs went away? No. Someone felt that those efforts were not the best way to accomplish the goal or that private industry could take over those efforts. There is a need to justify that not only is the team doing something good, but it is the best organization to accomplish that good.

Mr. Szymanski stated the team's sponsor is a key element. All financial support comes from AJPO. Virginia Castor is currently the support person even though she is no longer the director. Although it cannot be predicted what will happen if the new plan comes into effect, Mr. Szymanski stressed that the team has established momentum.

Mr. Szymanski then addressed the following key questions. What are the cost and benefits for the team and for the individuals? It is because the team networks which is a good way of getting information. Every one has their own agendas and some are looking for future business. It is definitely costly, but the individuals have to determine whether or not it is appropriate. Justification for the team should be considered. Also, is it the only way to accomplish the task? This is the only ongoing effort at present. The team's charter says it is the DoD focal point for the validation of evaluation technology.

3.2 Working Group Status Reports

3.2.1 Requirements Working Group (REQWG) Status Report

Attendees:

Major Patricia Lawlis, Chair
Capt. Rebecca Abraham
Jerry Brookshire
Peter Clark
Dr. Bard Crawford
Dan Eilers
Jay Ferguson
Fred Francl
Greg Gicca
Marlene Hazle
Don Mark
Ronnie Martin
Sandi Mulholland
Nelson Weiderman

Deliveries Due This Quarter: None

Accomplishments:

- Decided to update the Requirements Document to reflect the newer attribute definitions.
- Decided to update the Tools and Aids Document with the appendices giving more details on the requirements specified in the document.
- Began work on a sample Tools and Aids Document appendix.
- Produced an Ada Technology Assessment paper.
- Began work on a Quality Factors paper.
- Established a name and scope for the E&V newsletter.
    - The E&Ving News.
    - Articles on:
        - ACEC.
        - CIVC.
        - Reference System.
        - Why should I care about/what is E&V?
        - What do you think/need?
        - Other pertinent E&V issues/activities.
    - Schedules of:
        - Product deliveries.
        - Talks.
        - Tutorials.
        - Other pertinent activities.
    - Points of contact.

Key Issues:

- Continue to emphasize the use of prioritized requirements.
- Continue to emphasize the technology transition (PR).

Projected Work:

- Continue work on the next version of the Tools and Aids Document.
- Continue work on the Quality Factors paper.
- Establish the scope of ongoing technology transition (PR) efforts.
- Plan for the future of the E&V effort.
- Look at how or if the team might be able to influence the timely production of Validation Summary Reports.

Deliverables Due Next Quarter:

- Version 2.1 of the Requirements Document.

Presentations Planned: None

Other Significant Information: None

Action Items:

Carried Over

- Mulholland. Life cycle support from the whole-APSE view.
- COORDWG. Coordinate a more detailed summary of E&V activities and products that are suitable for PR distribution (possibility: Ray's paper?).
- COORDWG. Check procedure AdaIC could use for updating documents from the E&V team.
- Crawford/Clark. Produce ASCII form of the Reference Manual and Guidebook.
- COORDWG. Deliver ASCII form of the Reference Manual and Guidebook to AdaIC.
- Mulholland. Develop a first draft of a quality factors document.

New

- Ferguson. Place the E&V bulletin board announcement on the MASA bulletin board.
- Crawford. Give a copy of the bulletin board announcement to Jay Ferguson.
- COORDWG. Place Jay Ferguson's NET address on the E&V team list.
- ACECWG/CIVCWG/CLASSWG Chairs. Prepare a short item on the product status and give it to COORDWG at the end of each meeting for the E&V newsletter.
- Team. Submit the appropriate items for the E&V newsletter to COORDWG.

COORDWG. Check with the Ada !etters to see if the E&V newsletter can be published there.

- COORDWG. Prepare the first newsletter by the next meeting.
- Lawlis. Place a reminder on the NET for everyone to send Betty Wills the information on any conferences - specify format (name, date, conference, location, and comments).
- Mulholland. Update the Requirements Document with a pointer to the Reference Manual attribute definitions (also update the date and version on the title page and remove the trademark references).
- ACECWG. Produce a questionnaire for feedback on the ACEC.
- Crawford. Give a copy of the Reference System questionnaire to the ACECWG.
- Martin. Draft an appendix on the Test Assessors for the Tools and Aids Document.
- COORDWG. Update the EV-Info directory.
- Lawlis. Place the Ada Technology Assessment paper on the NET.
- Lawlis. Place the working group status report on the NET.

3.2.2 CAIS Implementation Validation Capabilities Working Group (CIVCWG) Status Report

Attendees:

Gary McKee, Chair
Karyl Adams
Jeff Facemire
Robert Fainter
Dr. Tim Lindquist
John McBride
Lloyd Stiles

Deliveries Due This Quarter: None

Accomplishments:

- Completed/approved discussions on the CAIS Implementation Validation Capabilities (CIVC) Critical Design Review (CDR) and System Requirements (SRS).
- SofTech status report.
- Discussed configuration management issues for the CIVC contract and problem approaches.

Projected Work:

- NET traffic.
  - Implicit tests.
  - Format definition of CAIS.

Deliverables Due Next Quarter: None

Presentations Planned: None

Action Items:

- Working Group. NET traffic on the implicit tests and the formal definitions issues.

- McKee. Get an early draft of the February CIVCWG minutes to John McBride.
- Szymanski. NET access for Karyl Adams and Bob Fainter.

3.2.3 Standards Evaluation and Validation Working Group (SEVWG) Status Report

Attendees:

Dr. Tim Lindquist, Chair
Karyl Adams
Jeff Facemire
Robert Fainter
John McBride
Gary McKee
Lloyd Stiles

Deliveries Due This Quarter: None

Accomplishments:

- CAIS-A Evaluation: "Low Cost Performance Evaluation from a Set of Validation Tests."
- A CAIS tool to support the CIVC(A) framework. Issue: Graphics support.
- A minimum set of tools for "validation" that are also development tools.
- Evolving CIVC to CIVC-A.

Projected Work:

- June: the final SEVWG review of IAS for CAIS-A.
- September: the first team review of IAS presentation.
- December: the final team review.

Deliverables Due Next Quarter: None

Presentations Planned: None

Other Significant Information: None

Action Items: None

3.2.4 Classification Working Group (CLASSWG) Status Report

Attendees:

Ronnie Martin, Chair
Capt. Rebecca Abraham
Peter Clark
Dr. Bard Crawford
Major Patricia Lawlis
Fred Francl

Deliveries Due This Quarter: None

Accomplishments:

- Reference Manual Enhancements:
  - Chapter 4, Life Cycle Activities, discussed support for alternate paradigms.
  - Chapter 5, APSE Tools Categories, reviewed/refined tools and mappings to functions to ensure the coverage of CAIS implementations.
  - Chapter 6, Attributes, reviewed introduction of attributes for approoriateness to APSE E&V as opposed to weapons systems evaluation. Also reviewed the attribute definitions in response to Marlene Hazle's message.
- Guidebook Enhancements:
  - Determined if the synopses can be further referenced in the Guidebook.
  - Reviewed/enhanced the Runtime Environment Taxonomy.
  - Discussed the full screen/"modern" editors.
  - Determined the modifications to the Test System Assessors Checklist needed to highlight traceability.
  - Discussed the Timing Analysis Checklist.
  - Generated the Instruction Level Simulation Capabilities Checklist.
  - Reviewed the RADC Tools Directory for checklist inputs.
- Whole APSE Assessment Issues:
  - Reviewed the updates to the Guidebook, Chapters 3 and 13.
  - Reviewed the whole APSE assessment issues work to date for the inclusion in Version 2 of the Reference Manual.
  - Reviewed the whole APSE usability assessment per Marlene Hazle.
  - Discussed the potential for checklists based on Dale Gaumer's seminar.
  - Discussed the potential for checklists based on Lyon's book.
- Other Topics:
  - Discussed document availability/appropriateness - Aerospace Compiler Evaluation Document, WIS Documents, and STARS Documents.
  - Reviewed and suggested enhancements to the Draft Reference System Usage Questionnaire.
  - Discussed procedure for determining the inclusion/exclusion of the information in the Guidebook.

Key Issues:

- The CLASSWG needs help from experts with the following:
  - Program Library Management Capabilities Checklist.
  - Import/Export Capabilities Checklist.
  - Real-Time Analysis Capabilities Checklist.

Deliverables Due Next Quarter: None

Presentations Planned: None

Other Significant Information: None

Action Items:

- Martin. Write a paragraph on the life cycle/2167A independent view of life cycle activities for Chapter 4 of the Reference Manual.
- Martin. Review Chapter 4 of the Reference Manual, Life Cycle Activities, for its treatment of testing-related products and functions.
- Clark. Improve the Attribute Definitions and Introduction per discussions.
- Martin. Send the reference for Quality Factor Definitions to Peter Clark.
- Lawlis. Review the Nissen book for detailed technology and forward the information to Peter Clark.
- Clark. Determine how to sprinkle references to Lyon's book throughout the Guidebook (put out a message for help).
- Clark. Place the Enhanced Runtime Environment Taxonomy Checklist on the NET for comment, ATTN: Dan Eilers and Jerry Brookshire.
- Clark. Enhance the Editor Checklist for the more modern capabilities.
- Clark. Continue work on the Timing/Tuning Checklist using the DACS Tools Directory examples.
- Martin. Review the simulation coverage based on the DACS Tools Directory examples.
- Francl. Develop/refine the Time Critical Applications Support Checklist for the Front End Tools based on the Dale Gaumer materials.
- Martin. Review/refine Debugger and In Circuit Emulation Checklists based on the Dale Gaumer materials.
- Clark. Describe the actions to be taken on Gary McKee's recommendations on the NET (except Whole-APSE Assessment Issues).
- Crawford. Describe the actions to be taken on Gary McKee's recommendations concerning Whole-APSE Assessment on the NET.
- Crawford. Provide a mapping of the Requirement/Design Tools to Functions for Chapter 5 of the Reference Manual.
- Francl. Provide a mapping of the Reference Manual 5.2.8, Host/Target Downloader, and 5.2.9, Host/Target Uploader, Tools to Functions.
- Clark. Place a message on the NET asking for volunteers to provide a mapping of the Reference Manual Chapter 5, Tools to Functions.
- Abraham. Monitor STARS documents for the References/Synopses in the Guidebook.
- Crawford. Update the Reference System Usage Questionnaire per discussion.
- Martin. Develop procedures for the submission and review of the new Guidebook entries and handling of the user guidelines.
- Mulholland/Brookshire. Further elaborate the whole APSE assessment issues. Transform the customization issues (distributed APSEs, runtime support) into a checklist.
- Mulholland. Further elaborate whole APSE Assessment issues/address life cycle support issues.
- TASC. Include a paragraph in the Guidebook referencing the element of bias inherent in all evaluation techniques.

3.2.5 Ada Compiler Evaluation Capability Working Group (ACECWG) Status Report

Attendees:

Nelson Weiderman, Chair
Jerry Brookshire
Mike Burlakoff
Dan Eilers
Jay Ferguson
Greg Gicca
Marlene Hazle
Don Marks
Lt. Robert Marmelstein
Sandi Mulholland
Ray Szymanski
Kermit Terrell

Deliveries Due This Quarter: None

Accomplishments:

- Reviewed the new tests for Phase III.
- Discussed capturing lessons-learned.
- Discussed response to the feedback from the previous meeting.
- Discussed the future of ACECWG.

Key Issues:

- Some of the key areas are not covered by ACEC.
- Documentation issues which remain (principally indices and math library).
- Getting timely information to the working group.
- No significant user feedback.
- Some of the new tests may be inappropriate (corrections, I/O).
- Data and Analysis Center for Software (DACS) mailing list may not reach ACEC users (as opposed to POCs).

Projected Work:

- Review the white papers for Phase III extension.

Deliverables Due Next Quarter: None

Presentations Planned:

- More complete presentation of Phase III test plans (holdover).
- Presentation of the feedback from the ACEC user community (holdover).
- Presentation on the results of Part 2 of the Preliminary Design Review (PDR) (holdover).

Other Significant Information: None

Action Items:

Carried Over

- Eilers. Look at the coverage issues relative to the other checklists.
- Szymanski/Marmelstein. Distribute the white papers on extensions to the working group.

New

- Weiderman. Place the working group status report on the NET.
- Working Group. Place corrections to the December working group minutes on the NET by March 15.
- Boeing. Provide a rationale for including the tests which are seemingly correctness issues.
- Boeing. Provide a rationale for including the I/O tests which are seemingly test operating system and disk system.
- Boeing. Determine whether there is still a problem with the Alsys/Apollo file on the distribution tape.
- Szymanski/Marmelstein. Take up the issue of fixing the documentation (indices) with Boeing.
- Boeing. Check the coverage of the discriminates within the records (page 11 of the December minutes).
- Szymanski/Marmelstein. Get the auxiliary documentation (test report and interim report) or Defense Technical Information Center (DTIC) numbers to the working group.
- Boeing. Create a file giving the test descriptions.

3.2.5 Coordination Working Group (COORDWG) Status Report

Attendees:

Don Jennings, Chair
Pat Maher
Betty Wills

Deliveries Due This Quarter: None

Accomplishments:

- Reviewed the December 1988 General Session minutes.
- Met with REQWG to respond to courtesy action items.
- Drafted a format for the newsletter.
- Established an EV-Conference file on EV-Info.

Projected Work:

- Review the February General Session minutes.
- Continue the development of the public relations (PR) strategy.
- Follow through on the courtesy (required) action items from REQWG.

Deliverables Due Next Quarter:

- Newsletter

Presentations Planned: None

3.3 Closing Remarks

Mr. Szymanski closed by thanking Dr. Tim Lindquist for arranging this quarter's meeting of the E&V team. The February session of the E&V team was then adjourned.

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX A

10.0 ADA TECHNOLOGY ASSESSMENT

Ada technology problems are not solved

- Ada 9x effort points out the need for improvement, particularly in areas such as real-time applications
- compiler technology is being pushed beyond traditional concepts
  Example: use of concurrency and generics
- current compilers are not adequate for embedded systems
- current compilers are not supporting distributed/parallel systems
- Ada technology goes beyond compilers to the entire environment to be used for system development
  - good environments are almost non-existent, particularly for real-time development
    Example: they don't include tracing time-critical requirements through the entire development process
- first wave of Ada use is defining the need for the second new wave of technology

What can E&V do?

- E&V focus is not just compilers, but environments
- E&V technology is a special part of Ada technology which
  - can influence the development of APSEs and tools which address areas such as real-time development support
  - can fill the need for tools to assess software
  - can lead to the use of software standards
    - software reuse
    - tools could be moved to new platforms
  - gives the manager a base from which to choose
    Example: for distinguishing systems adequate for real-time use
  - covers broad issues and requires a team of experts representing a broad spectrum of interests

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX B

20.0 LIST OF ATTENDEES

Abraham, Capt. Rebecca
WRDC/FDCL
WPAFB, OH 45433-6543

Brookshire, Jerry
Texas Instruments
P.O. Box 869305
Plano, TX 75086

Burlakoff, Mike
Southwest Missouri State
  University
Computer Science Department
Springfield, MO 65804

Clark, Peter
TASC
55 Walkers Brook Drive
Reading, MA 01867

Crawford, Bard
TASC
55 Walkers Brook Drive
Reading, MA 01867

Eilers, Dan
Irvine Compiler Corporation
18021 Sky Park Circle
Irvine, CA 92714

Facemire, Jeff
SofTech, Incorporated
1300 Hercules Drive
Suite 105
Houston, TX 77058

Francl, Fred
Sonicraft, Incorporated
8859 S. Greenwood
Chicago, IL 60619
Houston, TX 77058

Gicca, Greg
Sanders Associated
MLR24 1283
95 Canal Street
Nashua, NH 03061

Hazle, Marlene
MITRE Corporation
Burlington Road
Bedford, MA 01730

Lawlis, Major Patricia
AFIT/ASU
3318 E. Dry Creek Road
Phoenix, AZ 85044

Lindquist, Dr. Timothy
Computer Science Department
Arizona State University
Tempe, AZ 85287-5406

Maher, Patrick
Motorola, Incorporated
2100 E. Elliot Road
P.O. Box 22050
Mail Drop T4052
Department PZ511
Tempe, AZ 85282

Mark, Don
RADC/COEE
Griffiss AFB, NY 13441-5700

Marmelstein, Lt. Robert
WRDC/AAAF-3
WPAFB, OH 45433-6543

Martin, Ronnie
Software Engineering Research
  Center
Department of Computer Science
Purdue University
West Lafayette, IN 47904-2004

McBride, John
SofTech, Incorporated
1300 Hercules Drive
Suite 105
Houston, TX 77058

McKee, Gary
GARINCAR
P.O. Box 3009
Littleton, CO 80161-3009

Mulholland, Sandi
Rockwell International
400 Collins Road NE
Cedar Rapids, IA 52498

Rhoads, Barbara
Oneida Resources
3578 Kettering Blvd.
Suite 300
Dayton, OH 45439

Stiles, Lloyd
FDCSSA, San Diego
200 Catalina Blvd.
San Diego, CA 92147

Szymanski, Raymond
WRDC/AAAF-3
WPAFB, OH 45433-6543

Weiderman, Nelson
Software Engineering Institute
Carnegie-Mellon University
Pittsburg, PA 15213

Wills, Betty
CCSO/XPTB
Tinker AFB, OK 73145

APPENDIX K

MINUTES

OF THE

EVALUATION AND VALIDATION TEAM

GENERAL SESSION

7-8 JUNE 1989

The task for the Evaluation and Validation of Ada Programming Support Environments (APSEs) is sponsored by the Ada Joint Program Office (AJPO).

TABLE OF CONTENTS

Section Page

1.0 WEDNESDAY, 7 JUNE 1989 3
1.1 Chairman's Corner 3
1.2 Software Life Cycle Support Environment (SLCSE) 4
1.3 Ada Compiler Evaluation (ACE) Procedures and Guidelines (P&G) 7
1.4 Evaluation and Validation (E&V) Reference System Status . . 13
1.5 CAIS Implementation Validation Capability (CIVC) Program Status 14
1.6 Ada Compiler Evaluation Capability (ACEC) Program Status . 16

2.0 FRIDAY, 8 JUNE 1989 17
2.1 Working Group Reports 18
2.1.1 Ada Compiler Evaluation Capability Working Group (ACECWG) Status Report 17
2.1.2 Classification Working Group (CLASSWG) Status Report 19
2.1.3 Requirements Working Group (REQWG) Status Report . . . 21
2.1.4 CAIS Implementation Validation Capability Working Group (CIVCWG) Status Report 23
2.1.5 Coordination Working Group (COORDWG) Status Report . . 24
2.1.6 Standards Evaluation Validation Working Group (SEVWG) Status Report 24
2.2 Discussion of ACE P&G 25

LIST OF APPENDICES

10.0 LIST OF ATTENDEES 31

1.0 WEDNESDAY, 7 JUNE 1989

1.1 Chairman's Corner

Raymond Szymanski
Wright Research and Development Center
Wright-Patterson Air Force Base

Mr. Szymanski brought the Evaluation and Validation (E&V) Team up to date on his activities since February. His efforts have centered on presentations and contracts.

The Ada Compiler Evaluation Capability (ACEC) was awarded a change in specification.

The CAIS Implementation Validation Capability (CIVC) contract is being modified to the specification; to date, the technical evaluation has been finished.

The Analytic Sciences Corporation (TASC) contract is also undergoing a specification modification. The effort will be extended through June 1991 to include creation of an Ada model project.

Most of the changes in these three contracts are in response to input provided by the team to modify the focus of the project or to continue in a certain area for each of those efforts.

Concerning procurement, ACEC will terminate in ne 1990; CIVC and the E&V Technical Support are negotiating to terminate in June 1991.

Presentations:

- Mr. Szymanski gave a briefing in mid-April in Dayton, Ohio to the Electronics Industry Association G-34 Committee where the theme was software reliability.
- On 9 May 1989, he was a guest lecturer at Brooks Air Force Base, Air Force Systems Command at a computer resources acquisition course.
- On 25 May 1989 a presentation was given at NAECON. This resulted in an invitation to speak to Centaur.
- On 15 and 16 May 1989, Mr. Szymanski co-chaired the Wright Research and Development Center (WRDC) forum. There were 36 programs briefed. The synopses of those efforts will be processed for public release.
- At the end of June, Mr. Szymanski will chair a round table discussion at the Washington Ada Symposium (WAdaS). He will be joined by Dr. Bard Crawford and Gary McKee on that panel.
- At the Ada JOVIAL User Group (AdaJUG) in Denver, Colorado in July, Dr. Bard Crawford is giving a presentation on tools. Mr. Szymanski will provide a summary of the WRDC Ada forum.

Mr. Szymanski then introduced the first speaker of the day, Elizabeth Kean.

1.2 Software Life Cycle Support Environment (SLCSE)

Elizabeth Kean
Rome Air Development Center

Ms. Kean's presentation covered an environment developed at Rome Air Development Center (RADC) over the last several years called the Software Life Cycle Support Environment (SLCSE). Concerning the background on its development, in past years the Air Force had some major disappointments in trying to develop Ada environments. For example, the Ada Integrated Environment (AIE) resulted only in a compilation system rather than a complete environment. A major problem was trying to develop the entire operating system from the ground up and place it on a bare machine.

With SLCSE, the concentration is on having the user interface and data base on top of an existing operating system. Portability was not a concern. The objectives for developing SLCSE included the ability to develop mission-critical software and to be able to support the entire software life cycle. Depending upon project development requirements, SLCSE supports the addition of new tools.

SLCSE is supported in each individual phase of the software life cycle in addition to a project management or global tools such as schedule and milestone tools and the types of functions that go on through the life cycle. A goal is to add in application-specific tools and to be able to integrate existing tools which have already been tested or used by other organizations, or software development projects.

The user's role in the operational concept is a key factor in SLCSE. The type of function performed defines the action of the user. The common user interface is tailorable which means that constantly used tools can be brought up to the top of the screen. They can then be moved and rearranged. The interface is menu or key word driven, and SLCSE runs on any VT-100 compatible machine and is key oriented. It is not currently mouse driven; however, that will be a future enhancement. A mechanism to put tools in a folder or directories is being developed along with a set of Ada tools for a risk computer: an Ada compiler, and debugger/simulator. The folder would be set up so the sets of tools are visible rather than all tools being visible at all times.

The system's settings are invocable allowing the user to select any role based upon the project manager's organization. When setting up a software development project, the project manager can determine individual roles for each user. Multiple roles are possible as most users take on several roles in a normal software development project. Users can scroll through the roles determining the role they want to perform and observe the associated tools and data base objects.

A total of 74 tools is included in the tools menu. However, the number of tools available depends on the user's role; a systems analyst has 60 tools available. To make the system easier to use, a capability for multiple window displayed on the screen allows the user to bounce between windows. One of the most helpful features of SLSCE is it allows the user to select and remember the set of attributes that was chosen to perform a desired function.

The key part of SLCSE is the database which is an entity relationship (ER) model that generates Sequential Query Language (SQL). The underlying data base is transparent to the user. Users do not realize they are using strictly VAX/VMS files; they view only the objects displayed based upon their chosen role. Elements of an entity relationship model include domain and range. The SLCSE supports a stripped-down version of the ER model. Inheritance is not included so as not to overly restrict the performance. Traceability and data sharing can be done from the requirements phase through the design, code, and unit testing. The ER interface to the SLCSE life cycle data base contains a set of nine ER models in a network. The user can view the models or schemas in the data base at any point in time.

The SLCSE database is designed to support document generation with a tool that automatically generates the 2167A documents provided the database has been populated. However, populating the database may not be entirely automated, and the user may have to view the database and manually populate it via a template.

SLCSE contains a tool for the user to collect manually quality metric information. This tool is being enhanced to automatically generate the metric information and place it in the SLCSE data base.

The data base application interfaces sit on top of a SMARTSTAR data base management system. The ER model is written in a formal language which is compiled and generates SQL queries which are placed into the data base management system. One of the key factors is that the data base is created using VAX/VMS files, and the relations are only pointers into the VAX/VMS files. Therefore, SLCSE supports attributes of any size.

The ER model is written in a formal language, and although making changes to the model would involve recoding, it is possible to make desired changes. For example, initially the database had been created according to MIL-STD 2167 and has since been updated to satisfy the 21F7A standard.

The initial tool set capabilities include DEC tools which run on the VAX/VMS and new tools may be integrated into the environment based upon a particular project's requirements. SLSCE supports four languages: COBOL, JOVIAL, Ada, and FORTRAN.

There are two ways to integrate new tools into the environment. Tightly coupled tools are those that have an interface package to both the data base and the user interface. Loosely coupled tools have an interface package to the user interface only. For the most part, generating the interface package to the user interface is an automated process, whereas generation of the data base interface package is a manual process.

The user interface of the documentation generation tool is a template that the user fills out. The tool extracts information from the data base, places it in the correct format and outputs it in LATEX format. The user fills out the template by stating which project is being worked on and the rest is then automatically filled in with the needed information. It is then output into the DGL file, and the information is extracted and output in the correct format.

Once there is a successful Preliminary Design Review (PDR) or Critical Design Review (CDR), the notification is put into SLCSE. SLCSE will automatically lock all of the database entities that are associated with the newly baselined document to prevent any changes.

The Automated Measurement System (AMS) has been integrated into the SLCSE and does the non-automatic data collection discussed previously. A tool, Quest, is being developed which will be an automated analysis tool.

Ada Life Cycle IMPACT Analysis (ALiCIA) is an automated life cycle change impact analysis tool developed by RADC. The tool performs deterministic algorithms to analyze the impact of changing a certain module, requirement, or design feature in relation to the whole project.

The RADC hardware configuration has a VAX 8600 with the Britton-Lee IDM-500, VAX 11/780, and VAX Station II. The IDM-500 is used for the data base but the only items stored are the relationships and all of the pointers to the data base.

The SLSCE concentrates on the common user interface with its ability to access the tools in a menu driven format and on the data base. The entity relationship model is working quite well; SLCSE is currently being BETA tested and is doing very well.

The presentation concluded with a question and answer session. The following matters were discussed:

- The off-the-shelf tools required to run SLCSE:
  - VAX/VMS License
  - DEC RDB License or Britton-Lee (Data base machine)
  - SMARTSTAR License
- Concerning data base integrity, the analyst is responsible for determining the validity of the relation. There are tools within the document generator that create a traceability matrix. Then the analyst would have to determine its validity. There is also a tool which identifies requirements having no corresponding design and/or code modules and all the various permutations. This tool is used basically in identifying holes in the project.
- It is not true that tools were incorporated based solely upon cost. Cost was a factor only in that if the tool existed and satisfied the requirements and was either Government owned or off-the-shelf then this route was taken as opposed to redesign and reimplementation.
- DEC supported tools are easily integrated into SLCSE; it is almost automatic.

1.3 Ada Compiler Evaluation (ACE) Procedures and Guidelines (P&G)

Phil Brashear, SofTech
Dale Lange
Aeronautical Systems Division (ASD)/Ada Validation Facility (AVF)

Mr. Szymanski stated the purpose of this briefing was to discuss the P&G document, listing and presenting the major issues. He introduced the speakers; Dale Lange who works at the Language Control Facility under Bobby Evans and is involved in the AVF work, and Phil Brashear of SofTech, Dayton which supports Bobby Evans and the work being done on compiler validations. This session was to be devoted to making constructive recommendations for the Ada Joint Program Office (AJPO) to study before making their final decisions. Recommendations were also sought from the Team as to what other groups should review the document. He then turned the floor over to Dale Lange.

Mr. Lange stated that the initial draft document of the P&G was developed with support from Mr. Szymanski and WRDC, and SofTech on behalf of the Ada Joint Program Office. Speculation is that the document is in direct response to the House Appropriations Committee (HAC) report. The Ada Compiler Validation Capability (ACVC) P&G was used as a guide. There are some unanswered questions regarding whether evaluation is going to be required, as validation is. The primary emphasis of this session is to go over the Team's comments and determine the significant issues so they can be prioritized. Mr. Szymanski then addressed these issues specifically.

In reviewing the scheduling issues, the initial schedule did not call for a public review of the P&G document and one of the more significant comments was that a public review was necessary. The additional review bodies will include Government agencies, Government Advisory boards, and non-Government reviewers.

Another item concerning scheduling is the implementation of evaluations. First is the purported incompleteness of ACEC. Because of the different features that are tested by the ACEC, it gives the impression that there are significant differences between the implementations whereas they may be comparable. The second issue is training plans. The technical expertise necessary for evaluation will probably be more difficult than for validation. There was uncertainty over who will do the evaluations and how they will be trained. This should be addressed and made publicly available so that all can understand that those doing evaluations will have the necessary training. The third issue is the overly optimistic schedule. The evaluations will probably not occur by midsummer, as originally planned.

Policy issues included foreign dissemination and public availability of evaluation results. If there is significant negative results about an implementation, the evaluation could possibly hurt a company's business; therefore, public availability is an AJPO decision. Another policy issue is required evaluations. AJPO's response to Congress regarding the HAC report implies that this will not be mandatory and evaluation will not be part of validation.

There were five other significant issues. The first issue was that a concept of operation was needed for the whole procedure of doing evaluations. The second issue concerned pointers to other evaluation resources; there were many comments that too much was put into the P&G itself instead of referring to other relevant documents. A third issue was the timeliness of reports. The validation summary reports are usually out before the next validation but evaluation reports must be produced in a much timelier fashion to be useful. The fourth issue was the level of coverage that the ACEC provides. The fifth issue is the maintenance/upgrades for the ACEC. After Version 2 comes out this fall, there are no current plans for upgrading the ACEC for successive versions.

At this point Mr. Szymanski took the floor to lead the discussion and prioritize issues. The following issues to be addressed were suggested by team members:

- Who will do the actual evaluations? (management structure)
- Who has the legal right to perform evaluations? (legal basis for evaluation)
- What is the cost of evaluation? Who will pay for the cost?
- Is the overall idea of evaluation a good idea? What is the motivation for evaluation? Is it valid or an impulsive reaction to a request in the HAC report?
- What are the real objectives behind evaluation; what are we trying to accomplish?
- Tailoring evaluations to user requirements.
- The Concept of Operation issue was not explained in great detail in the P&G document. Need more details for issues such as:
    - Will evaluations be required and what possible conditions will exist under that requirement?
    - Will evaluation be performed after validation?
    - Does the compiler have to be validated prior to evaluation?
    - Who requests evaluation? Vendor? Government?

Because the Concept of Operation issues were considered a top level priority by the team members, they concentrated on this issue in an attempt to establish and clarify their goals.

Dr. Lindquist attempted to respond to the request for information concerning the rationale and concept of operation of evaluation in relation to the HAC report. Gathering evaluation data for a given configuration is an expensive proposition. The rationale is to avoid replicating that expense. Also, evaluation is administering a test and gathering test results. When ACVC is compared with evaluation, or compiler validation and compiler evaluation, the issues are completely different. Ideally, the ACVC is an objective set of tests and the ACEC is much more subjective; however, the evaluation is performed by the person who determines which compiler he will purchase, and this is not the intended use of evaluation.

Mr. Szymanski gave his personal opinion to help clarify the issues. He believes that the Government needs a way to give the programmers who will choose a compiler extra data from which to base their selection. Programmers need considerably more data to work with before committing a compiler to a multi-million dollar program. AJPO agreed with Congress' intent that validation alone does not provide enough information to make these choices. Therefore, AJPO decided to respond to Congress' request to provide performance data as well as the validation.

Ms. Mulholland recommended that a requirements definition be developed to define the evaluation process. The ACEC is one tool by which the requirements can be met; however, the requirement of the ACEC as a forced evaluation on all products should be eliminated. Develop a requirements definition for the P&G document stating the specifications for evaluating a product.

Mr. Szymanski believes that policy decisions will drive the final content and structure of the P&G document, and that he would push to disassociate evaluation with validation and to not make evaluation mandatory.

Mr. Brashear commented that consistency needs to be kept at the center of the concept of operation. To be able to reuse the results from one program to another, tests must be performed in a consistent fashion which is an argument for having a rigid procedure.

Concerning the discussion of the overall rationale, Dr. Crawford discussed the impact on compiler vendors. The development of the first version of the ACEC makes it easy to acquire; defense contractors can run it and learn from it. This will impact compiler vendors because of the discussion that will take place between potential users, buyers, and vendors. That is an argument for having the technology developed. Questions to be considered are: What is the argument for making the ACEC mandatory as part of some Government-run test which is tacked onto the ACVC? What will be the effect on compiler vendors? Will the effect be positive or negative, and why?

Mr. Ferguson stated that these issues could be addressed in terms of who will pay for the cost. What would be the cost to the compiler vendor? Placing evaluation on top of validation will impact the vendors financially. On the other hand, it could be argued that it has to be done once the results are available so that all the different Government organizations do not have to run their own evaluation. An argument could be made for the Government to help pay for some of the evaluations.

Dr. Crawford was concerned not with the cost in dollars, but what the effects would be for the vendors. Will it help the vendors solve the wrong problems or the right problems as a result of running tests in that manner?

Mr. McKee's difficulty with standardized evaluation tests run by the vendor is that the vendors can normally afford better hardware for those tests than the contractors that use the compilers. Understanding the hardware and optimizing it to make evaluations run effectively takes away the elements of the results that are valuable to an actual contractor using normal hardware. Therefore, there will always be a need for Government organizations to run the tests on the actual delivered hardware no matter what the vendors do.

Mr. Gicca stated his thoughts about the benefit in doing a single evaluation which can be dispersed to different contractors and about the timeliness of ACVC and ACEC reports. The compilers are going to be outdated by the time the reports come out, and he does not think there will be any great savings to the different contacts or contract offices by doing a single evaluation.

Mr. Mills said the biggest cost to compiler vendors is the time spent running those tests which is probably more than the cost of the actual validation itself. Even if the Government funded these validations it would have a major impact on the vendor. On the other hand, many small contractors have only a small amount of code to write and need to buy a compiler, but do not have resources to do a big evaluation on several compilers.

Concerning policy issues, Mr. McKee stated that it would be wise and would increase the usability of this product if the cost of using it could be decreased. He suggested adding to the concept of operation or to the policies some revision of relevant prices.

Mr. Eilers suggested that it would be useful to have a vendor addendum to the evaluation reports.

Ms. Hazle stated that the bottom line in an evaluation is how the vendor can protect himself. Mr. Szymanski replied that under the current plan there is not any protection but the issue would be addressed, perhaps using the vendor addendum.

Mr. McKee felt that it would be appropriate for a letter to be sent to the vendor saying that his product has been submitted under this configuration.

Mr. Gicca, referring back to Dr. Crawford's question on a positive or negative impact on compilers, found that the P&G says the results will be compared to a single baseline compiler. The approach of comparing all compilers to a single compiler is likely to have a negative impact if the compiler dealt with is a generic model that is average in all things.

Mr. Szymanski stated that there will be more than one compiler to compare against that base. A problem would occur if there was only a singular compiler. Version 2 with a single system capability will address that.

Mr. Brashear stated that the baseline referred to is not a compiler. It is a set of data that is distributed with the ACEC. It is a composite set of data, an average of six or seven different compilers in Version 1.

Mr. Leavitt informed the team that the major reason for including the baseline data was to do some analysis when there was only one system. If comparing more than the two systems, the raw data should be compared and reanalyzed.

Mr. Gicca finds this coming down to a problem with the ACEC and not with the P&G. The problem is with having to have multiple results to produce a report, and the fact that a single system report will not be directly comparable then to a multi-system report. The only way to do that is to use two compilers from which to produce the report.

Mr. Weiderman referred back to the question of whether performing evaluation is an overall good idea. The HAC report was composed a year ago before the ACEC was out. The problem that was addressed was with the validation process which is supposed to test conformance to a standard but the solution is not producing good compilers. The compilers have errors in them, and they are not satisfying the requirements. The response to the HAC report is to perform more extensive tests than validation can provide through formal evaluation.

It is important to go back to the root causes and ask what the proper solution is to this problem. What is the root cause and how do we arrive at a cost effective solution? How should we go about improving the quality of compilers? The root question is whether the ACEC Procedures and Guidelines document is the solution and whether it is the most cost effective way.

A basic alternative is for the Team to proceed along its existing path. The technology has improved in the last two years and more test technology is now available. There are now more guidelines and a reference manual. The question is, can the people who need compilers use the available technology in a cost effective manner to solve the root problem rather than putting into place this proposed organization and bureaucracy? The answer is yes, but with the underlying premise that third-party evaluations cannot be done. The party that needs the product has to do the product evaluation. They have to ensure that they are using the right technology to answer the questions for their requirements. By providing these procedures and guidelines, a general purpose system is being created that solves the problem for everybody, but also dilutes the system and no one will get exactly what they need in terms of an evaluation.

Maj Lawlis concurred that an individual evaluation is probably a better evaluation than a third-party one. On the other hand, it is impractical to think that every organization can do all its own evaluations. The idea that an organization like the Government can direct this and be able to compile information on evaluation is not palatable to many people. But it is still the most practical method for an evaluation process. Some important points have to be considered. Many issues brought up by the team are not addressed in the P&G and that is one of the main concerns. The Government point of view must be considered; their needs must be met. The vendor's viewpoint must also be considered. The process will inevitably occur and such evaluations are going to have to be done. Therefore, there needs to be some guidelines and considerations of all the various points of view and it needs to be done in a timely manner.

Dr. Crawford stated that the Government people feel a need to have this done by the Government. It should be done in such a way that two other points are taken into account. One, people should be encouraged to see that there are many things out there besides this particular test. Second, people should be encouraged to get the ACEC themselves as it should be available to anyone and not just Government contractors.

Mr. McKee found that part of the reason for this effort is to compare different compilers on the same target hardware for contract selection purposes. The other purpose is to evaluate a compiler's performance for its intended use.

Mr. Francl stated that by trying to save money by having one organization perform both the ACVC and ACEC involves the trap of timeliness. In the ACVC, much of the value which is received by going through the validation is made available immediately. But in the ACEC almost the entire value is in the report itself, which is a major difference. It may be better to go through an entirely different approval cycle to shorten the time involved rather than save money by using the same approval cycle and not change the procedures.

Mr. Szymanski asked what the group thought was a reasonable level of training to ensure that the tests are executed correctly. The consensus was that significant training will be necessary. Users need to be knowledgeable in assembly language. Systems people and people who understand compilers, target chips, chip architecture, and how runtime support interacts with the operating system will be needed. Training addresses the whole concept of the technology and is therefore very significant.

Mr. McKee said a policy strategy could be added stating that a vendor is always permitted to have a representative on site for evaluations. This provides a very valuable compiler and vendor hardware resource that can be used.

Mr. Szymanski asked for comments on the issue of availability of data. There are comments that the Government should not be obtaining this data and then keep it; it should be widely distributed. Mr. Gutzmann said that you do not always know the circumstances under which the data was collected so the quality of data is not always known.

Mr. Szymanski solicited Mr. Eilers opinion as a vendor. Would he willingly and freely distribute the results of his test suite? Mr. Eilers replied that timeliness is a very big factor in this situation. A vendor does not want his report sitting on a bookshelf for a year or two. He would not want a competitor to know what areas are strong and which ones are weak. The vendor wishes the customer to come to him. Therefore with these considerations, vendors are not likely to want to publish results.

Mr. Ferguson thought that public availability of the results could foster competition among vendors to produce a better product.

Ms. Mulholland asked what would be wrong with holding the results within the agency? It could be distributed to all the other agencies. The information that a compiler has been evaluated could be published along with the information to contact the vendor.

Mr. Ferguson recommended, that when the results are produced but before being made public, the vendor should have some visibility into the results and be allowed to comment.

Mr. Szymanski asked for suggestions on additional Government review bodies. Mr. McKee suggested the NET for notification, the Software Repository, the Ada mailing list, and presentations at AdaJUG and SIGAda.

Mr. Szymanski summarized that the massive holes in the P&G need to be addressed before another draft is produced. The most important issue will be examining the document structure which is dependent upon policy formulated by AJPO.

Ms. Mulholland asked what the role of the E&V team is going to be in this process and what will be the role of the Ada Compiler Evaluation Capability Working Group (ACECWG) or the E&V team in the continuance of the ACEC. Mr. Szymanski suggested that the ACECWG produce a strategies document for ACEC Version 3. Mr. Weiderman said there is a problem in not having all the information concerning exactly what is covered. On the issue of timeliness of reports, Mr. Brashear said if the review cycle can be shortened then a complete report can be produced within a month. A summary report with just the bare numbers can be done in considerably less time.

Major Lawlis reiterated that in the P&G document the wording should state that these results should be out no later than a certain time. This places the burden on the organization by saying it must be out in a specific length of time.

Mr. Weiderman suggested providing a money-back guarantee. If vendors pay for information but do not receive it in a timely fashion, they do not pay for it.

Ms. Mulholland commented that the one month schedule is unrealistic for evaluation. In evaluation, test results have to be researched for anomalies and a month will be used trying to understand the results. Following that will be the report generation and the proving cycle. Evaluation is more time consuming than validation.

Mr. Szymanski closed the discussion thanking everyone for their comments.

1.4 Evaluation and Validation (E&V) Reference System Status

Dr. Bard Crawford
The Analytic Sciences Corporation (TASC)

Dr. Crawford briefed the team on the E&V Reference System in terms of the Reference Manual and the Guidebook, future plans and a review of community interaction. He also presented a short report on the Model Project activity.

The Reference Manual was officially released for the first time in March 1988 and was assigned a Defense Technical Information Center (DTIC) number. Version 1.1 of the Guidebook (no version 1.0 was issued) was released August 1988; it now has a DTIC number. DTIC numbers are needed for this pair of documents so they can be produced and mailed out.

A draft of Version 2 of both documents is ready for review by the Classification Working Group (CLASSWG). This version of these documents should be released as officially approved documents in September.

Version 2.0 contains some new Guidebook elements. The first four chapters are of an introductory nature. Beginning with Chapter 5 the order follows the Tools and Aids Document so it is basically a prioritized order. The new elements of the Guidebook have been tabulated. The "others" category is Chapter 9, and is a collection of items that are useful but do not fall into any other category. Items from Chapter 9 will migrate in future editions to new chapters.

An E&V Technology Matrix displayed showed the categories of things to be assessed.

To summarize, two versions of the Reference Manual have been delivered to the public along with one version of the Guidebook. Version 2.0 of each document should be delivered in September. If the negotiation progresses, and an extension is awarded for two more years, Versions 3 and 4 of each document will be produced.

A panel discussion concerning community interaction was held at the National Security Industrial Association (NSIA) conference. Mr. Szymanski was an invited speaker at the Computer Resources Acquisition Course in San Antonio, Texas and also at NAECON. Fliers were handed out at all events.

The questionnaires passed out at the February meeting have been turned into a postcard. These postcards will be sent out with a letter asking that the questionnaire be completed. A copy of the flier is also included with the postcard.

A panel discussion will be held at the WAdaS and a presentation will be given at AdaJUG. Mr. Szymanski is giving a presentation to the ASD Council. Fliers will be distributed at these events also.

The general idea of the model project is to get at APSE evaluations which could mean whole-APSE altogether or some major portion of activity in a broad scale manner. Rather than a test that measures some particular performance attribute of a particular tool, the idea is to test an entire APSE during a life cycle or a portion of a life cycle by using a model project and then build scenarios around that model project.

The model project chosen is an Inertial Navigation System (INS) simulation which is a Software Engineering Institute (SEI) "Ada artifact" based on an earlier system from the Naval Weapons Center.

The first job is to create a subset of the 2167A documents, the system specification, software requirements specification, software design specification, and software test plan. Portions of these documents are being written. Each will be written and formatted to include a top level view and a detailed level view. A top level view and portions of the detailed level view following a thread approach is currently being designed. Portions of these documents are being written where certain threads through the system are addressed in detail. Then scenarios will be built around those threads; a few initial scenarios and the concentration will be on some test type scenarios. There are many ways to build on this sort of approach to test various aspects of an APSE.

Mr. Szymanski thanked Dr. Crawford for his presentation and introduced Jeff Facemire.

1.5 CAIS Implementation Validation Capability (CIVC) Program Status

Jeff Facemire
SofTech

Jeff Facemire presented the CIVC progress report. All CDR issues have been finalized. The AJPO agenda was defined; AJPO decided in February to provide funding for CAIS-A. The North Atlantic Treaty Organization (NATO) arena has also made positive steps towards adopting CAIS-A.

A revised System Requirements Specification (SRS) was delivered based on the use and nonuse of the Ministry of Defense (MOD) test harness. All references to the MOD test harness were deleted from the SRS, and it was revised along with the test plans.

The design phase is completed and the coding phase has begun. The coding phase is due to be completed this summer. Positive steps have been taken toward automated configuration management tools. Work has been started on a coverage analysis tool. This tool is data base oriented.

SofTech underwent some programmatic changes, and the analysis team has changed. John McBride will be transitioning out; Jeff Facemire will be acting as the program manager; and a new development team lead, Kurt Gutzmann, was hired.

SofTech is currently completing Engineering Change Proposal (ECP) 3. The ECP provides releases of CIVC Version 1. The first release will cover CAIS Chapter 4. Subsequent releases will cover portions of Chapter 5 which continues hypertext requirements traceability. The schedule reflects the use and nonuse of the MOD test harness.

The new schedule includes the following:

- The CDR was 30 January 1989. The test harnessing equipment or code will be developed and is due to be completed in July. The test case development will be done throughout the summer.
- The final preliminary qualification testing (PQT) will be done in November 1989.
- The formal qualification testing is due in December 1989 and Version 1 of the CIVC is due to be completed at the end of 1989.
- Phase 2 and CIVC Version 2 will commence in January 1990 and conclude in June 1991.

Mr. Eilers asked Mr. Facemire to comment on how the coverage analysis tool works. Mr. Facemire stated that the CAIS has been condensed down to entities. It is SofTech's view that in dealing with a CAIS environment there are nodes, attributes, and handles to nodes which are called taxonomy entities. A CAIS interface must act on one of those entities in some way. In essence, SofTech is creating a large data base with taxonomy entities on one scale and CAIS interfaces on another. A count is being kept by the actual code that is written.

The test requirements are derived from the specification. The test objective is allocated to some particular part in the taxonomy. When test codes are written, some list management might have to be done to prove that the process node was created correctly. The code is being annotated so that, even though the primary reason for testing is to test process nodes, lists are touched upon and created, attributes are deleted, etc. Those numbers are placed in the data base. There is much implicit testing done when writing the physical code that was not traced. This provides some idea of what type of coverage over and above the test objective is allocated to a particular taxonomy.

Mr. Eilers commented that the coverage analysis tool is really a data base rather than a tool that is analyzing the source code and finding things.

Mr. Szymanski asked how the data is called out. Mr. Facemire replied that it is all in a data base. It is a normal off-the-shelf data base that is queried so information can be acquired with relative ease. The code is being modified so data can be processed out and placed in the data base.

1.6 Ada Compiler Evaluation Capability (ACEC) Program Status

Tom Leavitt
Boeing Military Airplane Company

PDR was held in May 1989. Five items were discussed: the status of the ongoing baseline activity contract, the diagnostic assessor, symbolic debugger assessor, the program assessor, and the single system analysis tool. The last four items were added to the contract on the basis of the recently completed contract modification.

There was no significant technical change on the baseline contract activity based on the last briefing. There will be some modifications of the MATH package. The test suite and tools are going to be modified to be compatible with a subset of the NUMWG recommendations. The MATH package is a new GENERIC_ELEMENTARY_FUNCTIONS (Real). The User's Guide will discuss interfacing with the library. The restrictions on NUMWG in GEN_MATH include no hyperbolics, no BASE parameter in LOG, no CYCLE parameter on trigonometric functions, and treatment of a constrained type as an actual parameter. There is a change in the MATH test and double MATH test functions. The MATH test will report when errors exceed accuracy suggestions.

The diagnostic assessor will be a set of compilation units and evaluation instructions. It will be a template of questions with yes or no answers concerning each problem. The tool will analyze the completed template and generate a summary.

The set addresses both error conditions and warning conditions. This is significant for validation which does not require any warnings. The warnings vary significantly for different systems and not all warning messages are helpful.

In the debugger assessor, there will be a number of programs to execute. It is a collection of scenarios. The user will have to adapt each scenario to the debugger system which is to be evaluated. The major result will be whether or not it is possible to adapt the scenarios to the system. A template will be filled in to indicate if the indicated operations were done. The primary emphasis is to determine the functional capabilities of the system being examined. Some scenarios will measure performance with conditional breakpoints set, watchpoints set, and no debug command active.

The program library management assessor will have a set of units to compile and a set of scenarios to attempt. Users will need to adapt these scenarios to the compilation system that is being run. A template will be filled in to indicate whether or not the user was able to perform the operation. The principal emphasis is on functional capabilities. There will be some scenarios that look for execution time and disk space usage. Some testing is for suitability in porting large programs. Some scenarios will compile multiple units into the library, observing time intervals between compiling and linking.

The Single System Analysis tool was the other additional capability added by the contract modification. This tool is noncomparative. The Median tool will report on various numbers of related test problems. In examining the results of the related test problems in one system, various sets of reports are produced based on items such as optimization techniques, coding styles, and language features. There are statistical tests to determine whether significant differences exist between related test problems. Any ancillary information generated as byproducts of running the test suite will be collected and reported.

The Single System Analysis tool processes the same input as Median taking the same aggregate. The aggregate will have to be extended to support this since more information will be added which will be carried over as comments in the aggregate. The tool will be extensible; any user will be able to add additional test problems or additional analyses of existing problems.

Mr. Leavitt then opened the floor to questions. Mr. Gicca asked what sort of categories show up under coding styles and what does the output look like. Mr. Leavitt replied that the coding style basically is a set of test problems. It gives the name of the problem with a brief description along with times, histogram, and an indication about whether or not there were significant differences between the measurements on those and different problems.

Mr. Francl inquired concerning the single system title whether that is a host resident debugger or if a target resident debugger is used with those tests. Mr. Leavitt stated that the debugger tests are tests for functional capabilities in the debugger. That is independent of whether it is host based or target based. Mr. Francl asked whether there were any problems with timing measurements. Mr. Leavitt replied that the emphasis on the debugger is not on timing measurements. The primary goal of the debugger assessor is determining functional capabilities. Mr. Leavitt then concluded his presentation.

2.0 FRIDAY, 8 JUNE 1989

2.1 Working Group Reports

2.1.1 Ada Compiler Evaluation Capability Working Group (ACECWG) Status Report

Attendees:

Nelson Weiderman, chair
Sam Ashby, visitor
Mike Burlakoff
Dan Eilers
Jay Ferguson
Greg Gicca
Marlene Hazle
Alan Impicciche
Tom Leavitt
Lt. Robert Marmelstein

Sandi Mulholland
Lloyd Stiles, new member
Ray Szymanski
Betty Wills

Deliverables Due This Quarter: None

Accomplishments:

- Reviewed all 26 of Dan Eilers IV&V net messages.
- Discussed results of the 05 May 89 PDR and the minutes of the PDR.
- Discussed single system analysis and the main report.
- Discussed the status of all outstanding action items along with suggestions made at the February meeting.

Key Issues:

- Some key areas are not covered by ACEC (holdover).
- Documentation issues (principally indices and test naming).
- Getting timely information about test objectives to the working group.
- No significant user feedback (holdover).
- Overall quality of ACEC must be raised by Boeing's internal QA and by IV&V.
- Impact of ACEC Procedures and Guidelines on ACEC.

Projected Work:

- IV&V

Deliverables Due Next Quarter: None.

Presentations Planned: None.

Other Significant Information: None.

Action Items:

Carry Over
- Dan Eilers. Look at coverage issues relative to other checklists.

New
- Nelson Weiderman. Place status report on the net.
- Dan Eilers. Come up with a keyword list for the VDD for indexing tests.

- Ray Szymanski/Lt. Robert Marmelstein. Make sure that the responses to the 26 IV&V net messages are put on the net and cross-referenced to the original message number.
- Working Group. Review the Single System Summary Report and put comments on the net.
- Ray Szymanski. Review ACEC questionnaire and send out as appropriate.
- Nelson Weiderman. Send copies of SRS to any working group member that requests it.

2.1.2 Classification Working Group (CLASSWG) Status Report

Attendees:

Ronnie Martin, chair
Capt. Rebecca Abraham
Peter Clark
Dr. Bard Crawford
Fred Francl
Maj. Patricia Lawlis
Patrick Maher, new member
Honorary: Mike Burlakoff
Greg Gicca
Gary McKee
Sandi Mulholland

Deliverables Due This Quarter: None

Accomplishments:

- A paragraph written on the life cycle/2167A independent view of life cycle activities for Chapter 4 of the Reference Manual.
- Reviewed Chapter 4 of the Reference Manual, Life Cycle Activities, for treatment of testing.
- Reference for Quality Factor Definitions sent to Peter Clark.
- Reviewed Nissen book for detailed technology.
- Determined how to insert references to Lyon's book throughout the Guidebook.
- Described actions to be taken on Gary McKee's recommendations on the net (except whole-APSE assessment issues).
- Described actions to be taken on Gary McKee's recommendations concerning whole-APSE assessments on the net.
- Provided mapping of Requirements/Design Tools to Functions for Chapter 5 of the Reference Manual.
- Provided mapping of Reference Manual 5.2 and 8-9 Tools to Functions, Host/Target Downloader and Target/Host Uploader.
- Placed message on the net asking for volunteers to provide mapping of Chapter 5 of the Reference Manual, Tools to Functions.
- Placed Enhanced Runtime Environment Taxonomy Checklist on the net for comment.
- Enhanced Editor Checklist for more modern capabilities.
- Continued work on timing/tuning checklist using DACS Tools Directory Examples.
- Reviewed simulation coverage based on DACS Tools Directory Examples.
- Developed/refined Time Critical Applications Support Checklist for Front-End Tools based on Dale Gaumer's materials.
- Reviewed/refined Debugger and In-Circuit Emulation Checklists based on Dale Gaumer's materials.
- Updated the Reference System Usage Questionnaire per discussions.
- Determined appropriate mailing procedures for questionnaires.
- Developed procedures for submission and review of new Guidebook entries and handling of "User Guidelines."
- Pat Maher's comments discussed.

- Reviewed updates to the Reference System.
- Addressed Greg Gicca's comments.

Open:

- Improve attribute definition and introduction per discussions. Due prior to Version 2.
- Monitor STARS Documents for references/synopses in the Guidebook.

In Progress:

- Whole-APSE assessment issues:
  - Customization
  - Distributed APSEs
  - Runtime Support
  - Life Cycle Support
- "Warning" paragraph in the Guidebook on bias in all techniques. Expanding on negative aspects of bias. Due prior to Version 2.

Key Issues:

Need help:
- Mapping of the Reference Manual Chapters 5 to 7.
  - Configuration Management
  - Distributed Systems
  - Distributed APSEs
- Extract technology/references from Nissen's book and from Lyon's book.
- Review Runtime Environment Taxonomy checklist.

Projected Work:

- Guidebook Checklists: Power or Completeness.
- PM Toolset Document.
- Comments from originators of technology.

Deliverables Due Next Quarter: None.

Presentations Planned: None.

Other Significant Information: None.

Action Items:

<u>New</u>
- Pat Lawlis. Send reference for spiral model article to Peter Clark. Due prior to Version 2.
- Ronnie Martin. Generate/locate definitions for new testing terms and determine placement in Chapter 7 of the Reference Manual.
- Ronnie Martin. Review test related definitions.
- Peter Clark. Reorder Reference Manual Chapter 5 Toolsets into Life Cycle Order. Due prior to Version 2.
- Ronnie Martin. Continue determining references to Lyon's book in the Guidebook.

- Working Group. Read Version 2 to determine what next.
- Peter Clark. Contact Nelson Weiderman referencing attributes addressed by AES and SEI handbook. Due prior to Version 2.
- Ronnie Martin. E&Ving News Article.
- Tom Leavitt. List of debugger and library management capabilities being evaluated by CLASSWG.
- CLASSWG. Add Tim Lindquist's operational definition of CAIS and MITRE tests to the Guidebook.

2.1.3 Requirements Working Group (REQWG) Status Report

Attendees:

Maj. Patricia Lawlis, chair
Capt. Rebecca Abraham
Sam Ashby, new member
Mike Burlakoff
Peter Clark
Dr. Bard Crawford
Dan Eilers
Jay Ferguson
Fred Francl
Greg Gicca
Marlene Hazle
Alan Impicciche
Liz Kean
Tom Leavitt
Ronnie Martin
Mike Mills
Sandi Mulholland
Lloyd Stiles, new member
Nelson Weiderman
Barbara Rhoads, recorder

Deliverables Due This Quarter: None.

Accomplishments:

- First drafts of appendices for Tools and Aids Document, Test System Assessors and Compilation System Assessors.
- Coordination with COORDWG on public relations activities.
- Suggestions for ACEC P&G concept of operations.
- Draft of Quality Factors paper.
- Start on plan for the future of the E&V task.
  - Product concerns:
    - Reference System - hypertext on-line
      - model project work
    - CIVC - CAIS A development
      - who will do the testing?
    - ACEC - input to plans for testing service
      - management of future development
  - General concerns:
    - Contact with new software group at Hill AFB.

        - possible future transition?
        - contact by next meeting
        - representation on team
    - Should general focal group on E&V continue? How?
- Continuing topics on new technology needed.
    - how to deal with them?

Key Issues:

- Technology transition
- Requirements prioritization and expansion

Projected Work:

- Continue work on appendices to Tools and Aids Document.
- Continue work on technology transition (PR) efforts.
- Continue work on Quality Factors paper.
- Continue work on ACEC P&G concepts.

Deliverables Due Next Quarter:

- Version 2.1 of the Requirements Document.

Presentations Planned: None.

Other Significant Information: None.

Action Items:

Carry Over

- Sandi Mulholland. Life cycle support from whole-APSE view.
- COORDWG courtesy item. Coordinate a more detailed summary of E&V activities and products that is suitable for public relations distribution.
- COORDWG courtesy item. Check procedure AdaIC could use for updating documents from the E&V team.
- Dr. Bard Crawford/Peter Clark. Produce ASCII form of the Reference Manual and Guidebook.
- COORDWG courtesy item. Deliver ASCII form of the Reference Manual and Guidebook to AdaIC.
- Nelson Weiderman/Gary McKee/Ronnie Martin courtesy item. Prepare a short item on product status and give to COORDWG for newsletter.
- COORDWG courtesy item. Check with Ada Letters to see if E&V newsletter can be published there.
- Sandi Mulholland. Update the Requirements Document with a pointer to the Reference Manual attribute definitions (also update date and version on title page and remove trademark references).
- Ray Szymanski courtesy item. Get out questionnaire for feedback on ACEC.

New

- Marlene Hazle. Produce RED-YELLOW-GREEN poster on relationships among assessors, tools, and executing software.

- Maj. Pat Lawlis. Place status report on the net.
- Greg Gicca. Place draft of appendix on compilation systems on the net.
- Jay Ferguson. Place draft of Quality Factors paper on the net.
- Ray Szymanski courtesy item. Get copy of latest version of Requirements Document to Sandi Mulholland.

2.1.4 CAIS Implementation Validation Capability Working Group (CIVCWG) Status Report

Attendees:

Gary McKee, chair
Jeff Facemire
Jack Foidl
Kurt Gutzmann
Tim Lindquist
Ray Szymanski
Lisa Antolini, translator
Denise Conner, recorder

Deliverables Due This Quarter: None.

Accomplishments:

- Review of CIVC status, SofTech presentation.
- Discussion of automatic test case generation concepts.
- Evaluation of the Hypertext framework status.
- Discussion of TRW CAIS status, NATO CAIS plans.

Key Issues: None.

Projected Work:

- IV&V of CIVC framework and test cases.

Deliverables Due Next Quarter: None.

Presentations Planned:

- CAIS tutorial at the ASEET symposium.

Other Significant Information: None.

Action Items:

- Jeff Facemire. Send updated 'framework' to Gary McKee by end of July '89.

2.1.5 Coordination Working Group (COORDWG) Status Report

Attendees:

Patrick Mayer, chair
Elizabeth Kean
Betty Wills

Deliverables Due This Quarter: None.

Accomplishments:

- Reviewed February General Session Minutes.
- Discussed format and content of newsletter.

Deliverables Due Next Quarter:

- Newsletter.

Presentations Planned: None.

Other Significant Information: None.

Action Items:

- Cleanup past courtesy action items.
- Release volume 1.0 of the newsletter.
- Update directory of conference information.

2.1.6 Standards Evaluation Validation Working Group (SEVWG) Status Report

Attendees:

Dr. Tim Lindquist, chair
Kurt Gutzmann
Jeff Facemire
Jack Foidl
Gary McKee
Kathy Kirkbride, recorder

Deliverables Due This Quarter: None.

Accomplishments:

- CAIS appropriateness study.
- Current version of IAS reviewed.

Key Issues:

Issues and Strategies for E&V of CAIS-A.
Outline:
1. Introduction
2. Scope

3. Approach
4. CIVC-A Size Analysis
5. Test Selection Criteria
   - Upgradeable from 1838
   - Most frequently used facilities
   - Facilities critical to transportability
   - Broadest coverage
   - Tests that distinguish conformance
   - Random sampling
   - High depth in a few areas
6. CAIS-A/CIVC-A Review Board and Fast Reaction Team
7. CAIS-A Evaluation
8. CAIS-A E&V Policies
9. Automatic Generation of CIVC-A Test Programs
10. CIVC-A/CAIS-A Taxonomy
11. Hosting CIVC-A Framework on CAIS-A
12. Summary Issues and Strategies

Projected Work: None.

Deliverables Due Next Quarter: None.

Presentations Planned: None.

Other Significant Information: None

Action Items:

- Jack Foidl. Mail copies of validation policy and KIT final.
- Working Group. Mail CAIS related materials to Jack Foidl.
- Tim Lindquist. Comment on granularity to CAIS-A.
- SofTech/Capt. Abraham: STARS contact regarding their use of CAIS-A and relationship to CIVC-A.
- Gary McKee. Tool transitions to CAIS-A (Hypertext).
- Tim Lindquist. Review and incorporate May 1988 SEVWG minutes in IAS (CAIS-A design team inputs to IAS).
- Gary McKee. Write up on validation strategies for common external form.
- Tim Lindquist. Hardcopy of IAS to SEVWG for markup by 1 August.

2.2 Discussion of ACE P&G

Ronnie Martin
RADC

A subgroup from REQWG met to discuss the objectives addressed by the ACE P&G. Ms. Martin informed the team of their conclusions. The group also discussed the pros and cons of the suggested approach. A brainstorming session proposed an alternative response.

The first objective of the ACE P&G was a response to Congress. The second objective was to improve the compilation systems that are going to be used for mission critical systems. The third objective was to provide the Government with standard information that can be used to judge compilers in particular.

The fourth and final objective was to reduce the cost of evaluation to the Government.

Next, a list comprising of positive and negative items of using the ACE P&G to try to accomplish the objectives was compiled. The first objective was not addressed as it does respond to Congress.

For the second objective, on the positive side this will tend to improve standard features of compilation systems. Negatively, it was felt that following this approach will focus the efforts of the vendors on the required evaluation tests rather than responding to project needs.

Concerning the third objective, it was considered as providing standard information by definition as a basis for comparison. The negative side was a concern whether that information is worth while because the results apply only to specific configurations and variations of that configuration. It is not clear whether or not the standard information produced can be reused again in a valuable way.

The point on required evaluations is important as it was unclear in reading the P&G when this was required or even if it was required. There was no statement on this. The ambiguity made it very difficult to have a clear discussion on what the effects would or would not be. The group's reactions were based on the assumption that this would be required in some way. The first concern was that the results which only apply to specific configurations and variations can be significant. There was also a concern about the project size. It was felt that the ACEC could be reasonable for a certain project size.

The next item was that standard test suites cannot be used to evaluate implementation dependent features. This was a concern as there are some items where the user has to know the internals of the compiler in order to be able to test it and know whether or not there is any difficulty. So the concern was that the standard test suite could not test everything that needs to be evaluated.

The next item was that information utility decays very quickly. The working group felt if the ACEC is looked at as being used to help select a compiler then it is very important that the information be current.

The fourth objective was to reduce cost of evaluation to the Government. On the positive side it was felt that the cost of development of evaluation capabilities would be reduced along with the cost of application. The reuse of evaluation results may reduce costs, but it is not known how often the user will be in the situation where the information really applies. On the negative side, there was a concern that the cost may end up being borne by someone other than the users of the information. It is not clear in the document who is intended to pay. It is not appropriate that people other than those using the information are going to end up paying for it. It was felt that needless evaluations may be done. The group is concerned about evaluations having to be done when there was not an identified consumer for the information. Another negative aspect is that the cost of extracting information may be high.

Ms. Martin then put forth the group's response to the objectives:

1. The ACEC is already available for use by programs. Therefore, the charge from Congress to be in a position to do these evaluations is overcome by events.

2. A mandatory evaluation should be included based on the use of ACEC or a subset, or other things which is what the P&G was viewed as describing.

3. There is technology available, such as the Reference System, the SEI terms, and various other benchmarks in the ACEC.

4. It is impractical to try to do formal evaluations.

5. Requirements should be defined for contractors to provide evaluation results making those results widely available.

6. First cut tests can be defined which provides some minimal level of quality or performance quality in a compiler.

7. Performance measurement could be done on the ACVC.

8. Production quality compiler requirements should be defined and checked against the standard.

9. A Government expert organization should be created to do tailored evaluations.

10. A private expert organization should be facilitated to do tailored evaluations.

11. Programs should be allowed to select appropriate subsets of standard tests such as the ACEC.

The recommendations are proposed in two groupings which are not totally independent of each other. One group is composed of numbers one, three, and four. This is the key motivation if the most important objective is the response to Congress. The idea is to inform Congress that formal evaluation is impractical. This is an important area where there is already technology available. This technology will continue to improve.

The other group consists of a combination of 5, 9, 10, or 11. The idea is to share information where possible, but the selection of information is driven by project needs. Either Government or private organizations can be formed as experts at evaluation. They can serve as consulting services. This set is viewed by the review group as being consistent with the ACVC P&G except for the idea of being mandatory in every case.

The final topic discussed by the review group was future direction. The following suggestions were formulated for Mr. Szymanski to carry out.

- First, try to slow down the implementation of the P&G.
- Second, convey the team's concerns about the ACE P&G to AJPO enumerating the basic objectives that the team thought were being addressed.
- Third, convey some of the alternatives.
- Fourth, lobby for serious consideration of those alternatives.
- Fifth, plan for the implementation.

Ms. Martin then asked for questions. Mr. Szymanski asked Mr. Eilers as a vendor what would be his biggest objective concerning evaluation. Mr. Eilers replied that the major objectives are already identified in the presentation. Controlling the agenda of every vendor and forcing every vendor into the same agenda is a very serious problem.

Dr. Crawford stated he liked the second group of options as it puts a Government team of experts in place. It also encourages programs to do a lot of their own evaluations. As additional negative, one of the results of a mandatory policy would be that in the long run there will be a lot less evaluation occurring which would save money on the Government side.

Mr. Szymanski asked if the ACECWG could compose a set of requirements for a potential Version 3. Mr. Eilers responded that it would be very useful to outline what is missing, but this is prevented by the lack of organization in the ACEC. Mr. Szymanski stated that in the forthcoming weeks there will be a listing of items to be handled and in what order they should be handled. He would be willing to apply people as consultants to correct these items. It is obvious that one of the first items will be coverage. He asked if three or four weeks prior to the next E&V meeting would be sufficient time for ACECWG to reconvene on Version 3. Mr. Eilers indicated that it was.

Mr. Gicca added that another point is the actual requirements of what should be evaluated for the data compiler system. The problem is the goal would be to evaluate the ACEC to obtain some direction. This still would not provide a handle on what ACEC does. Both efforts would be needed to decide a future direction.

Mr. Ferguson questioned the time for turn around on the reports. A year is too long a wait for evaluation. The timeframe should be no more than a month for results because you are already one-sixth of the way into the process of a new compiler version coming out after that point. Mr. Brashear said that his group is very conscious of this matter. Some tentative procedures have been drawn up for carrying out the P&G. One of the items emphasized at every stage is timeliness.

Mr. Ferguson asked what was the definition of client in the P&G and the compiler vendor's role in reviewing prior to publication of the evaluation. Mr. Brashear replied that it is not in the P&G.

Dr. Crawford asked what was the typical usage of the word "client." Mr. Szymanski stated it was anyone who has the money to pay for the evaluation. The client was basically meant to be the vendor as the interpretation was that evaluations were going to be required. Mr. Brashear said that some paragraphs clearly state that the client can be anybody who brings in the money and supplies the hardware and software. Other paragraphs only make sense if it is the vendor.

Ms. Martin said that part of the P&G states how everything has to be set up internally and a dry run take place before the evaluation. If this is viewed as a required process, it is not saving anything by getting these people to come in and do it. This is a totally different view than the one described by the team in terms of having an expert organization that is there to help. There was no expectation in the team's discussion that the client would have to have this up and running and be certain it is ready to go through this evaluation before the experts come in and do it. Mr. Brashear sees it as halfway between with the client proposing the tailoring and doing the work and then the facility approving it.

Mr. Ferguson proposed that the expert organization be created and do the work so it is not incumbent upon the client to do anything but supply the equipment. Mr. Szymanski thought that the rules changed depending upon the definition of the client.

Mr. Clark asked was the next version to be draft or final. Mr. Szymanski had discussed this with Mr. Evans and the outcome was that a new draft would be prepared to include the comments from the team. This new draft would have a wider area of review. He said that based upon the information received at this meeting he would lobby hard for some of the recommendations put forth. Mr. Brashear stated he would recommend to Mr. Evans to salvage the good portions but to essentially start over and redesign the document.

Mr. Szymanski thanked everyone for their participation and adjourned this session of the E&V team.

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX B

10.0 LIST OF ATTENDEES

Abraham, Capt. Rebecca
WRDC/FDCL
WPAFB, OH 4543-6543

Antolini, Lisa
Oneida Resources, Inc.
3578 Kettering Blvd.
Suite 300
Dayton, OH 45439

Ashby, Sam
Boeing Military Airplane Co.
P.O. Box 7730, MSK80-13
Wichita, KS 67277-7730

Brashear, Phil
SofTech, Inc.
3100 Presidential Drive
Dayton, OH

Burlakoff, Mike
Southwest Missouri State University
Computer Science Dept.
Springfield, MO 65804

Conner, Denise
Oneida Resources, Inc.
3578 Kettering Blvd.
Suite 300
Dayton, OH 45439

Clark, Peter
TASC
55 Walkers Brook Drive
Reading, MA 01867

Crawford, Dr. Bard
TASC
55 Walkers Brook Drive
Reading, MA 01867

Eilers, Dan
Irvine Compiler Corp.
18021 Sky Park Circle, #L
Irvine, CA 92714

Facemire, Jeff
SofTech, Inc.
1300 Hercules Drive
Suite 105
Houston, TX 77058

Ferguson, Jay
National Security Agency
ATTN: T303
9800 Savage Rd.
Ft. Meade, MD 20755-6000

Foidl, Jack
TRW, Systems Division
Suite 205
9265 Sky Park Court
San Diego, CA 92123-4213

Francl, Fred
Sonicraft, Inc.
8859 S. Greenwood
Chicago, IL 60619

Gicca, Greg
Sanders Associates
NCA1-2232
95 Canal Street
Nashua, NH 03061

Gutzmann, Kurt
SofTech, Inc.
1300 Hercules Drive
Suite 105
Houston, TX 77058

Hazle, Marlene
MITRE Corporation
Burlington Road
Bedford, MA 01730

Impicciche, Alan
Naval Avionics Center
NAC Code 826
6000 E. 21st Street
Indianapolis, IN 46219

Kean, Elizabeth
RADC/COEE
Griffiss AFB, NY 13441-5700

Kirkbride, Kathy
Oneida Resources, Inc.
3578 Kettering Blvd.
Suite 300
Dayton, OH 45439

Lange, Dale
ASD
WPAFB, OH 45433

Lawlis, Maj. Patricia
AFIT/ASU
3318 E. Dry Creek Road
Phoenix, AZ 85044

Leavitt, Tom
Boeing Military Airplane Co.
P.O. Box 7730, MSK80-13
Wichita, KS 67277-7730

Lindquist, Dr. Tim
Computer Science Dept.
Arizona State University
Tempe, AZ 85287-5406

McKee, Gary
GARINCAR
P.O. Box 3009
Littleton, CO 80161-3009

Maher, Patrick
Motorola, Inc.
8220 E. Roosevelt
Mail Drop R1208
Dept. PZ511
Scottsdale, AZ 85252

Marmelstein, Lt. Robert
WRDC/AAAF-3
WPAFB, OH 45433-6523

Martin, Ronnie
Software Engineering Research Center
Dept. of Computer Science
Purdue University
West Lafayette, IN 47907-2004

Mills, Mike
ASD-AFALC/AXTS
WPAFB, OH 45433

Mulholland, Sandi
Rockwell International
400 Collins Rd., NE
MS124-211
Cedar Rapids, IA 52498

Pina, Tracy
Oneida Resources, Inc.
3578 Kettering Blvd.
Suite 300
Dayton, OH 45439

Rhoads, Barbara
Oneida Resources, Inc.
3578 Kettering Blvd.
Suite 300
Dayton, OH 45439

Stiles, Lloyd
FCDSSA
200 Catalina Blvd.
San Diego, CA 92147

Szymanski, Ray
WRDC/AAAF-3
WPAFB, OH 45433-6523

Weiderman, Nelson
Software Engineering Institute
Carnegie-Mellon University
Pittsburgh, PA 15213

Wills, Betty
CCSO/XPTB
Tinker AFB, OK 73145

APPENDIX L

MINUTES

OF THE

EVALUATION AND VALIDATION TEAM

GENERAL SESSION

6-8 SEPTEMBER 1989

The task for the Evaluation and Validation of Ada Programming Support Environments (APSEs) is sponsored by the Ada Joint Program Office (AJPO).

TABLE OF CONTENTS

Section . . . . . Page

1.0 WEDNESDAY, 06 SEPTEMBER 1989 3
 1.1 Chairman's Corner 3
 1.2 Evaluation and Validation (E&V) Reference System Status Report 3
 1.3 Software Technology for Adaptable Reliable Systems (STARS) Briefing 5
 1.4 Ada Compiler Performance Testing Service (PTS) Procedures . 10
 1.5 CAIS Implementation Validation Capability (CIVC) Status Report 13
 1.6 Compiler Evaluation Capability (ACEC) Program Status Report 14
 1.7 Issues and Strategies for CAIS-A 15

2.0 THURSDAY, 07 SEPTEMBER 1989 22
 2.1 Perspectives on Selecting and Evaluating Ada Tools and Environments 22

3.0 FRIDAY, 08 SEPTEMBER 1989 26
 3.1 Working Group Status Reports 26
  3.1.1 Standards Evaluation and Validation Working Group (SEVWG) Status Report 26
  3.1.2 Classification Working Group (CLASSWG) Status Report . 27
  3.1.3 Ada Compiler Evaluation Capability Working Group (ACECWG) Status Report 29
  3.1.4 Requirements Working Group (REQWG) Status Report . . . 30
  3.1.5 CAIS Implementation Validation Capability Working Group (CIVCWG) Status Report 32
 3.2 Closing Remarks 33

LIST OF APPENDICES

10.0 LIST OF ATTENDEES 35

1.0 WEDNESDAY, 06 SEPTEMBER 1989

1.1 Chairman's Corner

Raymond Szymanski
Wright Research and Development Center (WRDC)
Wright-Patterson Air Force Base (WPAFB)

Mr. Szymanski welcomed everyone to the September quarterly meeting of the Evaluation and Validation (E&V) Team. Mr. Szymanski made the following announcements:

- Jeff Facemire is leaving SofTech and will no longer attend the E&V meetings.
- The following introductions were made:
    - Mr. Bruce Taylor from Intermetrics which was recently awarded an Independent Verification and Validation (IV&V) contract for the North Atlantic Treaty Organization (NATO) special Ada Programming Support Environment (APSE), particularly the CAIS implementation that is being built for that special APSE.
    - Mr. Kevin Hackett from SofTech, San Diego which is responsible for developing the CAIS implementation for the NATO special APSE. He is taking the place of Shawn Fanning.
    - Kermit Terrell is representing Tom Leavitt at this session. He is accompanied by Sam Ashby, Ty Startzman, and Richard Stiverson.
- Issues concerning the December meeting in San Diego included:
    - The information sent to the team by Lloyd Stiles.
    - A proposed five day schedule. This change from the normal agenda would allow Mr. Szymanski to sit in on all the contractual meetings throughout the week.
- Due to the change in the schedule, status reports will be done on the NET as opposed to having a wrap-up.
- Accomplishments since the previous meeting:
    - Meetings were conducted with all the contractual people over the summer. In the June timeframe all of the contractual efforts were newly negotiated and signed. This has taken longer than expected; the SofTech contract change has not yet been completed. Signatures are expected before the end of the fiscal year.

Mr. Szymanski introduced Dr. Bard Crawford as the first presenter of the day.

1.2 Evaluation and Validation (E&V) Reference System Status Report

Dr. Bard Crawford
The Analytic Sciences Corporation (TASC)

In a brief overview, Dr. Crawford stated that the Reference System has been the major activity but the presentation will also discuss a new activity, the Model Project status and plans which supports the APSE evaluation. Some miscellaneous items were also discussed.

Concerning the Reference System, 250 questionnaires were sent out with approximately 40 going to team members. There were 23 returned including two from team members which was a 10% return excluding the Team. Of those 21, it was generally agreed that the Reference System is a useful exercise. One response was that a thorough job had been done but its usefulness was uncertain. Two people replied that the Guidebook was definitely useful but they were not sure of the Reference Manual. All other responses indicated that the documents were useful. The major theme in the suggestions was to provide more checklists in the Guidebook.

These responses were to Version 1.1 of both documents. A major difference between Versions 1.1 and 2.0, which is about to be delivered, is that there are more checklists included in Version 2.0 and existing checklists have been extended.

Version 2.0 of the Reference Manual and Guidebook are completed; both drafts were reviewed at the June meeting. The Classification Working Group (CLASSWG) will review suggested changes by Rational, Inc. A quick turnaround is expected in incorporating these changes and the delivery to the Ada Joint Program Office (AJPO) is still set for September.

Ms. Mulholland asked if Rational is being used as an IV&V source on this system. Mr. Szymanski replied that they are not an IV&V contractor. The Reference Manual and the Guidebook handle mainly classical APSEs while Rational's system is very non-classical. When reviewed it became apparent that the Guidebook did not address many of those items contained within the Rational system and that should be in the checklist. Rational expressed interest in receiving copies of the next version of the Reference Manual and the Guidebook so that they could continue to provide feedback to the team.

Versions 3.0 of the Reference Manual and Guidebook have a deliverable date of September 1990. The main focus will be the enhancement of the Guidebook which will include more checklists and whole-APSE evaluation material.

The idea of the Model Project was to build scenarios around requirement changes, test exercises, version control, and transitions between phases. Its purpose is to evaluate APSEs from a whole-team/whole-project perspective. The project selected was from the Software Engineering Institute (SEI), an Inertial Navigation System (INS) simulation effort. Their functional requirements document discussed six major functional areas, seven subsystems, and seven other areas corresponding to these areas.

The idea developed last spring in consultation with Ronnie Martin was to treat the system development as though it were developed in five builds, one thread at a time. In the first build the message processing would be done, building the part of the architecture required to satisfy the message processing function and some of the operator interface and initiation. A thread could then be run through the actual system and a portion of the system could be tested. Other pieces could then be added. This is a good way to build a system and is also a useful device to create scenarios.

So far the effort involves creating a basis to perform other work by using some of the documents describing this project from SEI, rather than from 2167A-type standards. Internal software requirement specifications have been written and work has started on a design document and software test plan. By November, these documents should be completed at all high levels and for two of the five threads at a low level. By December, the documents should be ready to present to the Team.

The Software Life Cycle Support Environment (SLCSE) was considered for use as an APSE to evaluate scenarios. SLCSE is designed with the assumption that the user started at a requirement analysis or even earlier in the process and began entering data in a specific pattern. To evaluate SLCSE, these documents would have to be input along with the finer bits of information such as requirements, so that everything would be traceable through the system. This indicates another attribute of an APSE which may be important, that is, whether the APSE can be used to pick up a project midstream and incorporate all the data; if this cannot be done, it may be a weak point of the APSE itself.

Mr. Szymanski added that the Avionics Lab is deferring plans to evaluate SLCSE. The two individuals selected to do the work are no longer available. There was also the problem of spending $40,000-$50,000 to secure the additional memory units required to run SLCSE.

1.3 Software Technology for Adaptable Reliable Systems (STARS) Briefing

Teri Payton
UNISYS

The Software Technology for Adaptable Reliable Systems (STARS) briefing was given by Teri Payton from UNISYS which is a STARS prime contractor. Ms. Payton began by emphasizing the last three words in the STARS title, Adaptable Reliable Systems, which provide a focus for the effort. The three primes are UNISYS, IBM, and Boeing. From the system perspective, a joint study is being conducted among these three primes involving the Software First process for systems in the large development and the processes for creating systems in the large architecture.

The starting point was the study of the B-1 bomber, applying an Information Object Modelling technique. A consultant along with a team from each prime has investigated how to apply interviewing techniques and work with a generic architecture for the application domain. UNISYS is concentrating on the Naval tactical command and control, Boeing is concentrating on avionics, and IBM is concentrating on Air Traffic Control. UNISYS is studying the method of extending the types of techniques already in process.

Other research areas include aspects of software prototyping. In STARS, the final system requirements are not defined until the fourth year of the program. Up until that point, the basic development is prototypes. The Software First model follows an incremental development process. A reuse model is a key part in Software First.

In the second increment, an emphasis will be on starting a paperless presentation model. In the third increment, UNISYS will incorporate more of the essential information into a single essential form embedded in a program, and will use a type of hypertext.

Emphasizing reliable systems is one of the ways in which STARS is different from other efforts. One STARS' area that contributes to a reliable system is the creation of a multilevel security environment. This supports mandatory and discretionary access control functions. The goal is to demonstrate in a distributed environment a multilevel security environment with a minimal B-2 level of confidentiality and address the integrity of the component or who can modify the component. With respect to that reliability level, multilevel security is addressed under the automated configuration control. This is not a traditional configuration control; it addresses security aspects. Secure development methods on tools are being investigated to integrate more of an Ada syntax. This syntax defines a formal verification language and provides a mapping of pre-existing languages and techniques to the Ada form specified. UNISYS is doing some experimentation with Ada-based security development methods. This is being coordinated with programs at ORA Research Associates who are working on the Penelope language and systems for Ada verification. Another aspect of reliability is addressing reusability and qualification of parts. Some hardware components are reliable as they are standard parts to be used and reused, and the intent is to develop the same level of faith in the reliability of software components.

Under IBM's program, Harlan Mills Company is focusing on other aspects of reliability following cleanroom techniques. With statistically certifiable software, the same statistical quality control which is used in manufacturing can be applied so that a limited number of tests can be run and the test results will provide insight on the quality of the software. Harlan Mills is experimenting with this as it applies to software manufacturing. This holds some promise in the qualification and certification of software components. During the second increment, Harlan Mills is applying some of the cleanroom techniques which they developed to create a better method of software development with zero defects.

Next, one of the main adaptability aspects of the program is the creation of machine independent applications. At the end of five years, everything within STARS is to be machine independent. In other aspects of adaptability, one goal is to develop an environment and a repository that is tailorable to application domain. UNISYS is developing a technology base with a built-in tailorability mechanism. UNISYS has two main ways of accomplishing this, using knowledge-based systems to be able to tailor tools and the repository having different application domains.

The next aspect, rapid recertification, applies to reliability as well as adaptability in terms of applying reliability to trusted components. Recertifying a system is a lengthy process because is involves the entire system. Incremental certification would allow a piecemeal certification for components that had been precertified. If a specific methodology is followed for putting these components together, the entire system may not require recertification, thus saving considerable time in the recertification process.

Another aspect of adaptability is the use of application generators tailored towards different aspects of the application domain. It is also in frequent use within the environment itself to foster portability. Very small application generators were used in the CAIS A port to Unix that would supportSofTech's specifications. This would create both sides of the client interface that UNISYS has developed as part of the port.

Dr. Lindquist asked if the hypertext was implicit or generated, and where it would be hosted. Ms. Payton replied that this is more a hypertext server model where it would be easily usable in different types of applications. It would first be used on an intermediate representation for Ada programming which is being experimented with IRIS. The intent in the hypertext server is that it could be used elsewhere. The UNISYS server will probably be coupled under the CAIS-A model. Mr. McKee asked for the status on implementation of CAIS-A. Ms. Payton stated that the functional prototype was developed by SofTech for CAIS-A. UNISYS ported it to Unix and is being used by current subcontracts. It is not yet a complete CAIS-A implementation.

UNISYS is working on the definition of a five year plan for the object management system. UNISYS ported the SofTech November delivery of CAIS-A and had a functional prototype with which to start experimenting.

While straight access control will be supported, it will not be integrated with a trusted operating system until future increments. The UNISYS intent is not for STARS to be a CAIS implementor. STARS will use the results produced by SofTech and NATO. Access control falls within the second increment. Transactions and monitors are scheduled for inclusion by the third increment.

Other types of capability that will be added later include the use of long duration processes, a user-oriented DDL processor and user controlled distribution and instrumentation. This will show how the user is utilizing the underlying environment or the particular objects that seem to be undergoing constant change.

In the first increment, the CAIS port is integrated with the Ada command environment which was done under a STARS foundation. Harris, the test tool developer, integrated the test tool suite using CAIS-A nodes as objects. They utilized a DIANA intermediate representation. The issue is how to best structure tools as fragments so they can provide maximum advantage for reuse. UNISYS is still trying to break down the view where the tool builder creates a large monolithic tool making some use of CAIS-A overall, but everything else is extraneous to the tool. STARS' approach is to incorporate a fragment or capability more tightly into the environment.

Mr. Eilers asked if all tool fragments must work off the same intermediate representation. Ms. Payton replied that was the goal. A selection has to be made between DIANA and IRIS of which one would be the more common standard intermediate representation. Mr. Eilers asked for a comparison of DIANA and IRIS. Ms. Payton replied that IRIS provides a more concise notation for the intermediate representation. When working off DIANA nodes, the end result is a very large case statement as each node is specialized and it has to be processed separately. IRIS is much more concise in browsing information. The main advantage of IRIS is that with DIANA a tool specific attribute can be represented as either a comment or an annotation within the comment. For a tool to process information, it reads all the comments searching for something special. The main emphasis of IRIS is that each item can be made separately knowledgeable leaving nodes in the tree which can be processed as special entities by whatever tool works.

Mr. Weiderman asked what would happen if the questions at each phase are not answered affirmatively. He also wondered what sort of contingencies are in the later phases. Ms. Payton said that if the questions were not answered affirmatively with respect to CAIS-A, then the search would begin for an alternative object management system.

Mr. McBride asked what IBM is using as a base for their system. Ms. Payton stated that IBM is not currently working on the object management system per se. At present, IBM is creating low level types of interfaces such as what is available in CAIS-A.

Mr. McKee wondered where the CAIS-A interfaces would be placed. Ms. Payton said they would predominately be placed in the object management system. UNISYS is representing objects as nodes within the CAIS.

In the architecture, UNISYS has adopted CAIS-A as the object management system. CAIS-A has not been adopted for the virtual operating system to provide portability across the platform. UNISYS is using the virtual operating system to provide easy pathways to different machines. POSIX and Mach are the two portability platforms planned by UNISYS. CAIS will still provide for the portability of tools, POSIX and Mach will support portability of the CAIS-A based environment across a wide variety of computers. Currently, CAIS-A interfaces are used directly.

Located in the object structure layer are basically all of the virtual interfaces that a tool builder would use. The Ada unifying layer while being a virtual layer is not a physical layer. One of the currently perceived problems is that each of the different standards committees create their own Ada bindings to language standards. The intent is to present this common Ada binding mechanism which is still being researched for flexibility. UNISYS is exploring the feasibility of creating a standard way of doing Ada bindings.

Mr. McKee asked what would be the value of the GKS layer if the POSIX interface was below it and the STARS unique layer above it, and it could not be seen. Ms. Payton said the GKS layer will be seen. UNISYS intention in developing this layer is to experiment with it within the STARS environment and then it will go to the standard committees. Currently, UNISYS is involved with the GKS standard committee, SQL, and X. UNISYS has asked that the X consortium look at the Ada bindings and plans which have been submitted to them. The committee feels that this is premature but they are willing to talk whenever an implementation has been achieved. It is thought that the defacto bindings will be purely C based. The current concern is with X.

Mr. McBride asked what the rationale was for selecting POSIX on the commercial operating system layer as compared with using CAIS-A itself on top of an Ada operating system. Ms. Payton replied that industry is adopting POSIX on all types of machines. With the use of POSIX, UNISYS sees a more widespread use of CAIS-A based environments.

Mr. Gutzmann stated that there appears to be two operating systems, POSIX and CAIS-A. Ms. Payton said that POSIX is just a set of mappings to other operating systems. UNISYS is interested in what POSIX does to performance on those operating systems. Basically, UNISYS would be replacing the Unix interfaces by the POSIX interface. The question is whether POSIX is the native operating system or the virtual operating system.

One of the concerns about tools accessing the operating system or POSIX directly is that UNISYS is developing a secure environment. There has to be the ability to trap any access to data and this must be through CAIS. There must be the ability to envelope what goes to the compilers to ensure the user has access rights to the data before allowing the transaction to take place.

Mr. Ferguson asked what the CAIS does for the operating system compatibility. Ms. Payton stated that one of the tasks is to define the security architecture for the security environment. An issue that will be dealt with is integrating CAIS-A's mandatory and access control on a trusted OS. In CAIS there is no direct label for integrity so experiments will be performed using the access control mechanism to try to support integrity.

Mr. McBride inquired where the reuse library fits into the prototype. Ms. Payton replied that all three contractors had responded in their proposals that a repository can be integrated on top of the environment using the environmental framework. This repository would then use the object management system from the environment. In this increment, one of the tasks is to take the Reusability Library Framework (RLF) that was prototyped under the STARS Foundations and either integrate it to existing data base management systems to achieve persistence of objects outside of RLF or integrate it on the CAIS-A framework. Currently the operational repositories use straight file structure with no underlying mechanism for taxonomy; the structure is by CDRLs. In the next increment, IBM and Boeing will be working on a faceted structure. UNISYS is working on the next generation version of the repository, a knowledge-based repository library system.

Ms. Mulholland had heard of some studies on the repository which were very negative in terms of the licensing problems, the accessors, or availability problems that were not addressed. Also, there were items of reusability that are machine dependent and are being written into reusable application programs. Ms. Mulholland asked how these items were being addressed. Ms. Payton said that availability of components is an issue involving the legality protection on reuse. In terms of getting access to the components, Colonel Green has an assistant, Bill Bercaw whose primary task is releasing items where the initial release notice will be to DoD and DoD contractors so items will get beyond the boundaries of the STARS contractors.

Mr. Gutzmann asked how the document's designs are stored in the operational repository. Ms. Payton said that graphics are limited to items that can be drawn with small dashes.

The RLF is a knowledge-based system where some underlying knowledge was developed. A semantic network and a rule-based system was done in Ada. The architecture was done in terms of abstract data type to create the types of capability that is needed by an Ada knowledge representation system.

The library mechanism uses a rule-base to drive and focus a search across the semantic network mechanism. A semantic taxonomy structure was used for the library. Under the STARS foundation, this was prototyped on a sample library for an Ada benchmark.

Ada benchmarks were used for a couple different reasons. UNISYS had the expertise available in-house and could also reuse the knowledge base from the test plan assistant and Ada. The benchmark domain was feasible to accomplish in a short period of time, and a large body of components was already available. In the initial benchmark library, 75 components were modeled. This type of library facilitated the modeling of generators, as well as components. UNISYS was able to model the generator and components of generators rather than just being able to model specific components.

GADFLY's knowledge-based application uses the Hybrid Knowledge Representation Systems. The basic library system is another knowledge representation system. When both are used together and GADFLY is placed in front of the library, the result is termed the qualifying library. Components must pass a certain amount of tests including stress testing and black-box testing to enter.

Mr. Weiderman asked how UNISYS was planning to measure portability. Ms. Payton said that every delivery has to go through two distinct Ada compilers to establish portability. There are also initial portability guidelines in the reusability context. One of the recommendations is the importance of establishing portability guidelines to be used across projects so that the code is not tightly coupled to a particular compiler.

Mr. Weiderman inquired if there were plans to develop tools to check software for portability. Ms. Payton, in closing, said that currently UNISYS has not developed any such tools, but the possibility for development exists.

1.4 Ada Compiler Performance Testing Service (PTS) Procedures

Dale Lange
ASD/AVF

The Performance Testing Service (PTS) Procedure was rewritten with the viewpoint that evaluation should be a complete and total service. The experts involved would either have knowledge concerning application user requirements or have the ability to quickly learn to complete an evaluation or to do standardized tests. The approach chosen was to use standardized testing which anyone could access and from which other organizations could use the results.

The goals of rewriting the document were to:

- clarify objectives,
- address unresolved policy,

- describe concept of operation,
- provide for "public" review,
- define realistic schedule,
- specify availability of results, and
- allow for expansion.

The people involved will be experts at evaluating the data based on a complete knowledge of the application. These experts will not be involved in analyzing the anomalies unless requested to do so by the customer. The customer will be given the data along with the vendor's analysis and will be responsible for analyzing the data.

Ms. Mulholland commented that even with all the information the vendor will have a difficult time making a determination on the cause of the anomaly. That would be a problem with this plan. Mr. Lange stated that is the reason for the statement on the vendor addendum. Mr. Ferguson said that there should be at least one expert to review and write up the report. Mr. Szymanski stated that a clarification statement is needed in the plan on the level of expertise available.

Mr. Brashear commented that the document clearly states that the customer can request expert assistance even to the point of running the entire operation instead of depending on the customer's level of expertise. Mr. Lange replied that would be provided by the senior level people. Mr. Szymanski stated that each customer would require a different level of expertise and help would be provided accordingly.

Mr. Lange informed the team that the current comments indicated that the service is cost reimbursable. The AJPO commitment includes seed funding. In providing a grade, it was commented that there is not enough of a grade provided. AJPO will have to decide whether to include the grade. Mr. Lange feels it should be up to the customer.

Mr. Szymanski interjected that a single number is not a good indicator of a compiler, and the customer should be informed of this. Mr. Lange continued saying that the procedure implies that the customer is a vendor. Quality equals performance plus usability is the formula used in the testing. A problem occurred because usability was added later due to the fact that the Ada Compiler Evaluation Capability (ACEC) only does performance testing and later versions would perform usability testing. This provides some inconsistency in the document as well as in the title. It could be renamed Quality Testing or Performance and Usability Testing. Any recommendations from the Team will be passed on to AJPO.

It was commented that the testing is described as unbiased and impartial in the collection results. Although the testing itself is unbiased, the data is not. This deals primarily with the process and it was incorrectly assumed that the tests themselves provided impartial data. Mr. Szymanski stated that some of the scripts included were DEC oriented and removal has been recommended. The problems that have been identified have been studied and given due consideration. Mr. Eilers said that the stated goal should then be that not only the collection process but the final result is unbiased.

Mr. Szymanski said that steps should be taken to ensure that no intentional biasing of the suite for or against a particular compilation system occurs. Also, there is still not enough experience with the individual tests to determine the extent of bias that is present in the test suite. Mr. Lange said the resulting implication need only be supplemented by the application requirements. This is referred to in the procedures document in the same paragraph dealing with the reference set in the reference system.

There is a paragraph in the PTS that describes the result of these procedures in association with application requirements which can be used for compiler evaluation selection. The same paragraph refers to the E&V Reference System as a source for procedure in the process of compiler evaluation.

Ms. Mulholland would like to see a procedures definition written and delivered along with the results describing step by step the procedures for the customer to follow in analyzing the data. Also, a discussion should be included on how to further evaluate the other aspects of the system through the documents. She sees a need for a PTS user's guide. Mr. McKee suggested having guidelines for interpretation. The guidelines would be delivered with the results and would provide whatever level of information needed in order to use the data. A summary of this type of information would be useful. Mr. Lange said there were several comments on the definition of the Ada compiler. There was too much information included. The definition had been taken right out of the validation procedures which defined an Ada compilation system. It may be preferable to reference the performance testing of an Ada compilation system rather than just an Ada compiler.

With issues on customization, there was some concern that tests might be arbitrarily omitted if they did not perform well. Another suggestion with the PTS is that AJPO should approve all customization as there was some concern that compilers might be adapted so that all tests could be run. Mr. Weiderman asked if the intention was to allow the customer the right to customize the evaluation. Mr. Szymanski responded that since evaluations are not mandatory the customer should be allowed to have what he wants. The suite should be flexible enough for that although the results may not be available to other government agencies.

Mr. Lange stated that more detail is wanted in the definition of configuration. One comment stated that a definition of selection should be included as well as the existing definition of evaluation. It was commented that the projected testing schedule was overly optimistic. The schedule was arranged to achieve results as soon as possible. While it is optimistic, it is still attainable.

The comments on the CCR included that all modified tests should be identified; the vendor should be able to release the draft which is an AJPO policy question; and the contents of the result summary should be described. Some responses were that this data not be reported at all, just the raw data should be provided; it is an AJPO decision.

Mr. McKee asked what was the motivation for vendors releasing drafts. Mr. Lange said this referred back to the schedule being optimistic; if the vendor could release the draft, the unattainability of the schedule would not be as relevant.

With the PTS certification number, there is concern about relating it to a specific configuration. The configuration data could be used in the manner of the result data. This is another AJPO policy decision. Concern was expressed by the team on the use of the certification number in advertising by the vendors. Mr. Szymanski stated that a lot of people see certification as a seal of approval but it is not. It merely tells the buyer that the tests results have been double-checked by an independent agency.

The PTS data base is primarily items that have to be added by AJPO. These items concern contents, who would have access, and the legality of having this type of information in a data base.

The presentation closed with a discussion on which version of the ACEC document to use. The recommendation by the team was to go with Version 1.1 which would include the fixed problems.

1.5 CAIS Implementation Validation Capability (CIVC) Status Report

Kurt Gutzmann
SofTech

As Jeff Facemire is now with the Software Productivity Consortium, Kurt Gutzmann has been designated the project engineer for CIVC. John McBride is currently performing the duties of program manager.

Progress was reported on the following items:

- The Test Administrator is undergoing its final testing.
- The CAIS response simulator is being used successfully to test Test Classes and the Test Administrator.
- The coverage analysis tool was completed and was used to provide some coverage data on the completed test classes.
- The Software Product Specification is ready to be delivered.
- The Software Test Procedures are in development for the Formal Qualification Test (FQT) which is set for 08 December 1989.

Mr. Szymanski asked when the Software Product Specification will be delivered. Mr. McBride responded that it is ready to go but was held up due to contract problems.

Ms. Martin wondered how it can possibly be ready for delivery if the tests are not completed. Mr. Gutzmann stated that there are two versions of the document. The first delivery is without any listings. When testing is completed, the second version will be delivered. While the first version has preliminary and detail design documents, it is without any package codes or other codes.

Ms. Martin said it seems that the assumption is that no problems will be found requiring a change of design. Mr. Gutzmann replied that the testing is almost completed and any changes will be included in the second delivery. He then closed the presentation.

1.6 Ada Compiler Evaluation Capability (ACEC) Program Status Report

Sam Ashby
Boeing Airplane Company

The second release of the Software Product Specification is scheduled to go to Ray Szymanski in December. The following is a listing of what is contained in the second release.

- There were originally 1,076 performance tests. This was increased in version 2.0 by 294. The number of anticipated tests is 1,360.
- The diagnostic assessor tests had 76 scenarios based on input received at the Critical Design Review (CDR). There are 81 different scenarios describing the different diagnostics coming out of the compiler.
- The debugger assessor had 29 scenarios.
- There were 19 scenarios in the library system assessor.
- The single system analysis (SSA) tool is included in the second release. The Median tool is used to compare compilers A and B. The SSA tool compares the internal capabilities of a single compilation system.

Mr. Eilers asked if the PTS would use the single system Version 2.0. Mr. Szymanski stated that there was an obvious need to use the SSA which would have to be given some consideration.

Mr. Ashby continued the listing:

- There is currently some manual work being done along with some automation building via a MED_DATA constructor.
- There is a portable math library. The objective is to create a standardization of math libraries which can be used from A to B to C.
- Timing loop modifications are also included.

Ms. Mulholland inquired about the plans for developing an overall requirements definition. This definition would be populated by the scenarios being developed. It would give an indication of the percentage of completeness of functionality. She suggested doing a search and recommendation for consultants to help with those definitions. These definitions come under the Tools and Aids Document. If this document was finished, the requirements could be stated along with the information of where those requirements are met, but there is no overall definition as yet.

Dr. Crawford stated that in the problems discussed there is an impression that they will be fixed before the end of the calendar year. Can this be characterized? Is it going to be fixed by withdrawing large numbers of tests or is everything that Dan Eilers and Nelson Weiderman have been talking about going to be fixed?

Mr. Ashby replied that prior to the CDR all of the available software problem reports were prioritized and put into various categories, such as mandatory fixes and possible fixes. The problems in the mandatory category are being fixed. There is no intention of dropping any test at this point.

Mr. Szymanski stated that some of the problems are cost and schedule impacts which would mean a contractual modification, but those items which need to be fixed will be fixed. There are a few problems that are not in the "highly desirable" category which should probably be fixed first, as they are cost and schedule impacts that cannot be absorbed. The plan was to use the first months of maintenance to fix the Version 1.0 items. The Version 2.0 reports will be fixed as they come in.

Mr. Eilers said that one of the major items was reorganizing and renaming tests as part of the PTS procedures which is concerned with getting reports in half a dozen areas, i.e., floating point or tasking. Are the tests going to be reorganized? Is there any coordination going on in being able to get the number of categories limited as it is for the CIVC?

Mr. Szymanski stated that Mike Burlakoff's work on the keyword index is the stop gap measure. Mr. Eilers replied that the PTS is pretty specific about being able to get categories for tests. Mr. Terrell replied that category is the second item; there are other pullout features. The easiest one is tasking. Mr. Lange said he was under the impression that a description was included in the PTS of what features were used in each test and that it was easy to pick out this test. A team member replied that while that is true for some features, it is not true for everything. Mr. Eilers affirmed that was not true for the items listed.

Ms. Mulholland asked how the individual review of each test was being achieved. Mr. Ashby answered that there was a checklist.

Mr. Szymanski closed out the presentation suggesting the discussion continue in the Ada Compiler Evaluation Capability Working Group (ACECWG).

1.7 Issues and Strategies for CAIS-A

Dr. Tim Lindquist
Arizona State University

Dr. Lindquist's presentation described the current draft version of the Issues and Strategies Document. The following is a list of context items from the Standards Evaluation and Validation Working Group (SEVWG) discussions:

- The uses of CIVC-A will be minimal initially. The number of implementations is expected to grow at a steady pace.
- Short term interests of CIVC-A are overshadowed by the concern for the long term quality of CIVC-A which is the CAIS Implementation Validation Capability.
- When compared to either CAIS or Ada Language Reference Manual (LRM), CIVC-A is large and complex. The first validation set contained about 1,600 tests but six years later there were over 4,500.

- The resources for constructing CIVC-A are very limited.
- Lessons learned from the CIVC effort were applied to CIVC-A.

Recommendations on the discussion issues include:

- Cost and Size Concerns. It is recommended that the CIVC-A contractor initially spend time providing information to the working group or Mr. Szymanski in order for the money to be spent appropriately.
- Test Selection Mechanisms for CIVC-A. These should be studied further and the generation of CIVC-A tests should be based upon one or more tests selection mechanisms or a combination of such.
- CAIS-A Evaluation Approaches. This needs to be studied. The transaction to Ada would have been sped up if the first version of performance tests had been available five years ago. Currently the movement is to other aspects of evaluation.
- CAIS-A Validation Policy. This will be discussed later.
- Automatically Populate the Test Set with Code from Scenarios. CIVC has produced scenarios which are graphical pictures of CAIS node structures that design input context for a test. The test cases then establish the input context invoking some CAIS interfaces which investigate whether or not the appropriate action was performed by the CAIS interface. The recommendation is that scenarios be developed in a more formal specification technique and that the transaction from that scenario to the source code could actually be done automatically.
- Taxonomy for CIVC-A. A new taxonomy should be developed or at least a modification of the current taxonomy.
- Framework. The framework for CIVC-A should be developed on a CAIS-A tool or at least based upon a CAIS-A tool. The framework allows movement from the specification test objectives to scenarios and also to the taxonomy or in any direction. This is currently implemented on Hypertext media, Hypertext Guide produced on the Macintosh.

One of the changes in CAIS-A is a new functionality which includes the following:

- Transactions. This allows several operations. Those operations are either committed to be registered into a CAIS data base or those changes are aborted as a single unit.
- Typing Mechanism. A way of defining the types of different nodes of attribute relationships has been added along with viewing those types as appropriate for different tools. These capabilities were not included in CAIS.
- Attribute Monitors. This allows association with certain attributes and their values, operations would be invoked in CAIS. These values were placed as particular constraints.
- Explicit Distribution. This enables the process to be created where the physical resources exist on different machines.
- Access Control. This is required in 1838A.
- Input/Output Model. The model has changed dramatically.
- Basic Node Model. There are also some changes to this node as examples.

The challenges that functionality presents, in terms of the CIVC-A test suite, include added bulk as there are more functions to generate the tests for. There are two aspects to the additional functionality: (1) New interfaces to support, and (2) The mapping into existing functions which existed in CAIS have changed to reflect the semantics of the new capability.
Carry Over challenges from CIVC include:

- Input/Output. This was placed at a low level of priority in terms of developing tests.
- Concurrency and Distribution. The same problems exist in CAIS-A and the same solutions are appropriate for CAIS-A.
- Data Access Synchronization. This area has been mainly ignored in CAIS-A.
- Pragmatics and Required capacity. This exists in CAIS-A in the same form that it did in CAIS.

Shortly after the beginning of the CIVC contract, the number of estimated required tests was between 7,000 and 8,000. There are several different scenarios necessary to establish a given test objective which could then be developed into CAIS test cases. In terms of 1838A there are approximately 15,000 test objectives. This number is based upon the number of interfaces in 1838.

SEVWG decided that tests should be picked in a manner allowing the group to achieve the objectives of a validation suite requiring 15,000 test objectives. The set of requirements for test selection include the following:

- The ability to use several different test selection criteria in conjunction to balance suite costs with interface objectives.
- The ability to prioritize different test selection criteria.
- The test selection criteria must apply test objectives from the same representation of CAIS-A.
- Test selection criteria must allow assessment of coverage relative to a specific criterion or overall different test selection criteria.
- Tests selected from different criteria must be manageable together.
- A test selection criteria must be implementable.

SEVWG recommends the CIVC-A contractor do the following very early as a way of balancing the cost and size aspect. First, study the suitability of existing CIVC test objectives for CIVC-A and the existing test objectives. Second, report via Topical Area the impact that the new mechanisms have on the number of test objectives. Third, estimate the number of test objectives to cover areas where CAIS and CAIS-A overlap. Fourth, develop a summary report on the effort expended to develop CIVC which could be in mythical man months or whatever is appropriate. This report should include summary statistics relating to the number of scenarios actually generated for test objectives and also the number of scenarios that will actually be molded together into a single test program.

In selecting test objectives for CIVC-A, the following example of a test objective was given. "Demonstrate that the predefined node attribute OBJECT_CLASSIFICATION is assigned to each root process node at node creation."

The process is that two or three scenarios are generated for a test objective and those scenarios are analyzed to see how they can be joined together in a single program. The reasons for a test selection mechanism include:

- Cannot select all test objectives,
- Want to maintain the suite over a long period of time,
- Want to enhance the suite over again during a long period of time, and
- The suite should reflect certain purposes and there is a need to have a test suite that addresses those particular purposes.

The following is an example of the test selection criteria which SEVWG has looked at.

- Facilities that implementors are likely to omit.
- Facilities more critical to transportability.
- Facilities most likely to apply to both CAIS and CAIS-A.
- A random sampling across the facilities.
- Facilities that achieve the broadest coverage.

Mr. Eilers asked if the distinction was made between intentional and unintentional for "likely to omit." Dr. Lindquist replied that there are two different categories in that particular area which may be intentional and unintentional.

A team member stated that some items are likely to be omitted simply due to the structure of the interface set or because they are hard to implement. Dr. Lindquist said these particular test selection criteria have a sketched implementation approach or an approach that could be used to actually select tests.

SEVWG has investigated two different areas with respect to trying to select tests that are likely to be omitted by implementors. First, there is very little syntax that is viewable by the user but has a large underlying functionality. Some examples are copy_tree, delete_tree, and pathname matching. The second is onion skin functionality. This example is representative of process management or at least routine in process management which uses node_management, typing, access_control, and list_management. The outer level of the onion skin is likely to be omitted. How are tests selected based upon this criteria? A functional dependence chart of CAIS-A is developed and then selected from the high levels on that chart. Third is a continuation of support for only those tests that have a history of distinguishing among implementations. This is not "likely to omit." Although this could not be used in the initial development of the validation suite, it certainly can be used in the long term to continue to maintain tests, to drop tests that are bad predictors, and to retain tests that are good predictors. This particular approach assumes that a point has been reached where use of CIVC-A exceeds some resource limit.

Transportability is one of the primary objectives of CAIS-A. All CAIS interfaces are critical to porting, but some interfaces are only used to complete a function.

To select tests that will focus on transportability, SEVWG suggests to:

- Pick a set of tools most likely to be ported,
- Isolate kernel facilities used by those tools,
- Map those facilities to CAIS-A functions, and
- Develop tests for those functions.

In discussions of this particular selection mechanism, SEVWG agreed that input and output facilities are critical for transportability and have a high priority.

For facilities that apply both to CAIS and CAIS-A, it is recommended that tests selected for CAIS-A still use this particular selection criteria. There is no existing policy requiring use of either CAIS or CAIS-A. It is too early to establish the feasibility of either CAIS or CAIS-A with respect to use in tool construction, maintenance of tool sets, transportability of tool sets, and implementability on a broad class of machines.

Flexibility is essential at present. If it is possible to develop suites that are going to be equally applicable to either of the interfaces, then it should be done. This could be accomplished by listing the functionality that is common to both sets and selecting from that list.

Random sampling is a very effective way of selecting tests. A random selection of test objectives is appropriate under the assumption that they are limited resources. The keys to allowing this type of selection mechanism are a complete taxonomy, linearize entries, and randomly select from linearization. Enhancement can occur in the following two respects. First, random selection can be done by CAIS-A topical area. The taxonomy can be completed in access control, enumerated, and then randomly selected. Second, random selection can be done by test type, normal execution, exception processing, and static tests. Random selection can be done by any one of these types or a combination of the two as long as the ability to linearize remains. SEVWG recommends the use of more than one method.

Under the broadest implementation coverage, the typical test case establishes a context by means of CAIS-A calls. In one test, 10 to 15 CAIS calls may be needed to establish a context. The next step would be to invoke one or more interfaces that are being tested against the initial context. The possible results are examined by calling other CAIS-A interfaces.

There are two different measures of broadest coverage. One measure is the number of interface tests. There is a mathematical relationship between that number and the number of test objectives. The second measure is called the broadest implementation coverage. It maximizes the number of different CAIS interfaces invoked in steps 1, 2, and 3 of the typical test case. To exercise the interface set as broadly as possible, one should call as many different interfaces as one can. How is this done? One, test objectives and scenarios are developed. Two, the scenarios are analyzed for dependency and combined as appropriate. Three, repeat. The test case code is developed for the remaining scenario that invokes the largest number of unused CAIS interfaces. These steps are repeated until there is no more money for developing tests. This technique assures maximal coverage of CAIS interfaces.

SEVWG recommends that the CIVC-A contractor should develop a plan for the test selection criteria to be used in developing CIVC-A test objectives. This plan should state the selection criteria to be used, the prioritizing of criteria, and the portions of the suite to be derived from each criteria.

SEVWG has also discussed standards for a review board and fast reaction team. This would be a technical sounding board for the development and administrator of CIVC/CIVC-A. The board would advise on direction and content for the development of CIVC-A. It would be a fast reaction team regarding the use of CIVC/CIVC-A and would monitor and promote the evolution of the suite. This may be done through the CIVC contract.

In discussing the CAIS-A implementation evaluation capability. SEVWG concluded that ACEC's schedule was seven years later than it should have been. The movement from performance tests to a more comprehensive evaluation data gathering should have occurred five years ago. The same mistakes should not be made with CAIS-A implementation evaluation.

There are some differences between evaluating the CAIS implementation and the Ada implementation. In general, one implementation per host is expected for the CAIS. This is not essentially true for Ada. Time and space in a development environment have a different meaning than an embedded system or a hard real-time application. Performance differences in CAIS-A implementations will likely occur in a few isolated areas of functionality and in fact those differences will arise from a smaller number of different implementation techniques.

The following items of Performance Tests are covered in the Issues and Strategies Document:

- Traversing a path,
- Spawning a process/job,
- Striking relationships,
- Creating attributes and values,
- Defining type and view definitions,
- Referencing the type structure for type checking,
- Performing access checks,
- Opening nodes,
- Canceling and committing transactions,
- Initiating attribute monitors,
- Importing and exporting nodes, and
- Manipulating channels.

SEVWG spent a considerable amount of time on alternate approaches. These are actually issues that need to be raised as possibilities and pursued further. First, rehost a minimal tool set onto CAIS-A. The tools should be instrumented for performance gathering purposes. The CAIS-A should be configured for use by the tool set into the framework for evaluation. There should be the same type of a framework for evaluation as for validation which applies to the compiler evaluation as well. A standard set of scripts or scenarios of tool use should be supplied that can be used to gather performance information. Advantages are MAPSE and CAIS-A design evaluation and use studies. Second, the validation suite should be used to collect performance information in addition to validation information. If the evaluation framework is properly configured, it can be made to work providing the configuration persists over time. As changes occur in the validation suite, the implementation on evaluation also needs to be considered.

For automatically generating test case code, the CIVC step from the scenario to the Ada/CAIS source code is very straightforward. The test case code has a fairly consistent format, initialize the CAIS context, exercise the facility, and examine for intended results. The SEVWG recommendation is to raise the level of formality in the expression of scenarios so they may be automatically analyzed to produce the source code directly. To accomplish this, develop a scenario description language, automate a dependency analysis, and develop a translator from scenario descriptions into Ada with CAIS calls.

SEVWG found that certain revisions need to be made to the taxonomy for CIVC-A. First is to remove the basis for identifying test objectives from an interpolation of the specification, meaning CAIS-A, directly to the specification itself. Second, the current coverage analysis must be performed relative to the taxonomy rather than the specification. Third, the current taxonomy is not directly applicable to CAIS-A without modification.

The next recommendation is to host the CIVC-A framework on a CAIS-A based tool. This would describe the framework as the Hypertext portion of SofTech's effort. The following is a list of advantages and reasons for doing this.

- Extensibility of the test suite is a critical issue.
- The framework is the key to coverage assessment, extensibility, and maintainability of CIVC-A. This is a very important product and one of the more innovative efforts.
- The framework for CIVC is currently hosted on the Macintosh Hypertext. It would be more appropriate for it to be on a CAIS-A environment.
- The framework for CIVC should be modified to provide the following:
    - accessibility through scenarios
    - associations for source code
    - automatic loading of associations
- A straightforward CAIS-A tool could be developed with better functionality for the framework with the exclusion that one would not expect user interface.

The presentation closed with a listing of SEVWG's summary of recommendations.

- Review cost and size concerns to allow for adequate planning of CIVC-A.
- Develop and review the test selection mechanisms and select the appropriate ones.
- Form a review board and fast reaction team.
- Develop a more precise syntax for expressing scenarios that automatically translate scenarios into code.

- Modify the taxonomy for CIVC-A to allow assessment relative to the specification.
- Host the framework for CIVC-A on a CAIS-A tool to allow for proper maintenance of the suite.

2.0 THURSDAY, 07 SEPTEMBER 1989

2.1 Perspectives on Selecting and Evaluating Ada Tools and Environments

Dale Gaumer
Magnavox Electronic Systems Company

Mr. Gaumer's objectives for this presentation were to provide his impressions on the state of technology of Ada tools and environments, their prospects, and the selection and evaluation process from a contractor's point of view.

The Ada language standard was developed in an unusual manner. Generally languages are tested in a draft experimental form and then develop gradually. The reverse of this was true for Ada. The Ada language is very complex; the development was done in an iterative fashion. Not all systems can afford this luxury. Most systems will not generate sufficient interest to make this work. For a successful environment to be specified, it has to go through the complete cycle of implementation, usage, and evaluation.

Of the current systems, UNIX is becoming the de facto standard by default, even for target and real-time systems. TRON is a Japanese operating system which may be mandated for use on all microcomputer systems sold in Japan. The effect would be to block out all American suppliers.

The Portable Common Tool Environment (PCTE) is off to a good start. It is being thoroughly funded and many companies are involved. The method of operation achieves a reasonable level of consensus with no apparent determined detractors. Mr. Szymanski recalled comments from an Ada Europe meeting where there were several evaluations of PCTE. Some of those evaluations pointed out that PCTE was not as good as the CAIS standard and it would not be anywhere near as good as the CAIS-A. So there were some detractors. The conclusion now is that PCTE is so weak, the movement is to PCTE Plus. Mr. Gaumer responded that PCTE Plus will better serve the military. He pointed out that his reference was not to technical detractors but political or cultural detractors. This is more prevalent in the United States primarily against Ada. Mr. McKee asked what was meant by political detractors in the United States. Mr. Gaumer replied that involves everyone from the individual to the contractor. There are contractors who seem determined to defeat Ada and have enlisted members of Congress.

A chart from a market survey was displayed which had been done for stock market investors. The chart showed information on operating systems and projections for the next few years. Mr. Gaumer stated that he sees the Macintosh growing rather that staying constant as indicated on the chart. UNIX is moving into Macintoshes and other machines.

Certain standards are needed. The software design graphics, actual pictorial elements describing a software design, need to be standardized. Currently, every tool manufacturer is designing their own elements or using someone else's, resulting in a fair amount of variation.

In many cases, a functional similarity exists where tools can easily convert from one to the other achieving a different representation. A consistent representation would help to encourage reuse of tool fragments within a single tool company or between tool companies. It appears that the Booch icons, although aesthetically appealing, have too low a density for sizable systems. The Buhr icons are much more complete with a consistent point of view represented along with a thorough taxonomy. Firesmith's icons are very compact; they have the best density. They are basically rectangles into which text can be put which identifies the object or the processing entry point in a very compact way.

In the compilation systems, many items need to be either standardized or regularized more. The Ada library system is one of these. There is a need for good capability in the library system. There is no way to characterize all of the different features and functions in the library facilities of compilation systems. Standardization would help when changing compilers in mid-project. It is common to do most of the development on one compiler which has advantages for debugging a host machine and then later switch to a target compiler. Another aspect of standardization is that it will help in reuse of very large pieces of software. It would be an advantage to have some of the reusable pieces of large portions of systems set into a particular library structure.

Mr. Terrell asked if standardization meant the ability to compile into one library and then use that library to load and compile. Mr. Gaumer responded no; the user would have a library structure and a few simple diagrams about library structures would be compiled into that and when the user switches to a different compiler, he puts the same source and compiles it into the same library structure. One library is not taken and placed into a different compiler. Mr. McBride asked what aspect Mr. Gaumer was suggesting standardization would help. Mr. Gaumer replied he was speaking of a system library.

Sublibraries are essential to large projects. The different development teams must work independently, use different historical versions at the same time, and merge those versions gracefully to make a consistent build. For the reuse effects, sublibraries may aid in design of boundaries for reuse.

It is important to have multiple compilers on the same project. A project may require delivery of the system for maintenance on multiple hosts, a host debug compiler different from the target compiler, or a better compiler may become available. When setting up the library structure the user should anticipate all possible compilers to be used. The library structure should be set up for the least common denomination of those compilers. Other items to consider are document generators and utility tools. One project used the least common denominator approach between two major compilers. This represents an Engineering Development Library Structure where sublibraries are used to permit independence between individual team members.

Various CASE tools have methods of protecting the temporary invalidity that occurs when changes are made. The changes are protected from other changes made in a different time period and then they are merged together.

Another interest is lateral tool interfaces. CAIS and PCTE stress vertical tool interfaces with a standard interface of a tool to an underlying system. There is a need for standardization for horizontal interfaces from one tool to another on a successive development activity.

Mr. Crawford asked for an example of the type of interface being referenced. Mr. Gaumer said there can be a data base that has different tools manipulating different activities in the projects. The syntax and semantics have some arbitrary definition so that each tool has to deal with a complete arbitrary definition. There needs to be more coordination possible between the different tools for different activities.

Mr. McBride cautioned that horizontal interfaces may be interpreted as tool-to-tool communication. There have been systems where the tools communicate directly with each other without any use of integrated data base technology. The protocols between those systems then become very tool dependent. The issue of how the information is actually represented still has to be dealt with.

Mr. Gaumer said that in the area of readiness for major software projects, SEI is working on the software development process models, standards, and procedures. The idea is to have companies place their software business on some regular basis. Ways to prepare managers are by technology management seminars which are specifically oriented toward Ada and software engineers. There are Ada development courses which are especially beneficial to lower level managers. Also, there is always a need for better project administration in traditional management roles.

Client readiness is another factor affecting Ada's acceptance. Most customers relate little to the technology; software is a minor issue for them. Congress is a major client for defense contractors. And Congress tends to impose inappropriately rigid expectations. The GAO has also become involved with the Ada issue. The GAO originally performed an accounting function but are now conducting system analysis, system engineering, and technical feasibility studies. The GAO has become more than a tool of Congress. GAO was tasked to study 160 companies. Although the study was discontinued, the resulting thesis was that Ada should be terminated as a technology because contractors had to go from one version of a compiler to another. Because no one knew how to build Ada compilers and GAO did not relate to the technolgy, it was concluded that Ada was not a worthwhile investment.

General impressions concerning the Department of Defense (DoD) Program Offices that software generally has a low priority. Because they are familar with an old software method and there is not sufficient time for training there is very little appreciation for Ada, software engineering, or modern methods. The problem is too little time on station. No one wants their project to pay either the cost or the risk to improve the maturation of a new technology. Bureaucracies force functionaries to take the safest course by improving all possible standards, even non-compatible combinations, all possible Data Item Descriptions, and all possible quality and management indicators.

Developers need to be ready for advanced software technology. In tool selection, the selection process is often primitive or even non-existent. Some users rely on validation alone. In compiler selection, there is also a general lack of appreciation for other aspects such as robustness, capacity, library facilities, error reporting, runtime systems, etc.

Initially there is an excess of faith as the tool is expected to solve a multitude of problems. There is a lack of appreciation for tool continuity between multiple development activities. Other aspects include tools used inappropriately, assistance sources, and the need for a good taxonomy. There is also the need for a more general understanding of a compatible progression through the life cycle requiring a complete model of the development process, a better understanding of newer methods, and tool flexibility to fit process and methods. Advanced tools, for example, can span several activities and give the appearance of skipping major activities in the development process.

The library and sublibrary systems play an important role in data base management. Different parts of the system develop asynchronously. There were some serious problems found in one project. A data base management system (DBMS) was required on the system on which the application programs were printed. In one case interfaces of the DBMS had to be changed and no one knew how to make the changes to the DBMS without a forced recompilation of the entire project. This was avoided by one of the interim releases being made with some sublibraries with different application programs using different copies of the data management software. Another complication was that the entire main project library could not be placed on one disk even though it would fit. It was split up across different disks on the VAX.

Mr. Weiderman asked if Mr. Gaumer had any knowledge of organizations using the ACEC and how they viewed its effectiveness. Mr. Gaumer replied he did not know of anyone using it to any meaningful degree. Mr. Szymanski said that only one problem report had been received from outside the agency. Mr. Gaumer replied that being the data analyzer for the Performance Issues Working Group (PIWG), he understood the effort involved in gathering data. Most of the results received about PIWG in the last year came from the compiler vendors who use it for their own purpose. Mr. Szymanski asked if there was any certification process to guarantee that the resulting data is true. Mr. Gaumer stated that was a weakness for PIWG as there is no independent verification.

Mr. Szymanski asked Mr. Gaumer's opinion on the idea of establishing a centralized testing service for compilers. Mr. Gaumer stated that a government agency would make it more difficult for those inside the company to understand what was involved in the evaluation. Although a centralized testing service seems to be a shortcut, the risks outweigh the benefits because users would be encouraged to rely on the agency's recommendation and would become disillusioned when the product that performed best according to the agency's criteria does not perform well in relation to the user's criteria.

Mr. Szymanski thanked Mr. Gaumer. Mr. Gaumer informed the team that he had some PIWG results. Any comments from the team concerning style and format would be welcome.

3.0 FRIDAY, 08 SEPTEMBER 1989

3.1 Working Group Status Reports

3.1.1 Standards Evaluation and Validation Working Group (SEVWG) Status Report

Attendees:

Dr. Tim Lindquist, Chair
Karyl Adams
Kurt Gutzmann
Kevin Hackett
John McBride
Gary McKee
Teri Payton
Bruce Taylor
Jerry Thomas

Deliverables Due This Quarter: None

Accomplishments:

- Reviewed draft Issues and Strategies for the evaluation and validation of CAIS-A.
- Gave a presentation of the Issues and Strategies Document to team.

Key Issues:

- Explored the ideas of interaction with STARS and their CAIS-A work.
  - Working group contributed several questions with respect to the use of CAIS-A and the use of tools.
- Reviewed comments on the Issues and Strategies Document concerning the following:
  - Test Selection and Rule-Based Test Selection. Bruce Taylor suggested looking at a rule-based test selection technique.
  - Framework and Alternatives. A suggestion was made for the Issues and Strategies Document that the framework be rehosted onto a CAIS-A tool.
  - Taxonomy and the form needed to accommodate CAIS-A.
  - Alternative languages for automated code generation and the form the scenario descriptions would have to take in those languages.

Deliverables Due Next Quarter:

- Final version of the Issues and Strategies Document.

Action Items:

- Bruce Taylor. Compose a white paper on a rule-based test selection mechanism.

- Team. Comments welcomed on the Issues and Strategies Document.
- Tim Lindquist. Revise Issues and Strategies Document based on comments.

3.1.2 Classification Working Group (CLASSWG) Status Report

Attendees:

Ronnie Martin, chair
Capt. Rebecca Abraham
Peter Clark
Bard Crawford
Fred Francl

Honorary Members:
Sandi Mulholland
Gary McKee

Deliverables Due This Quarter: None

Accomplishments:

- For the Reference Manual enhancements the following action items were completed:
  - Maj. Pat Lawlis. Send References for Spiral Model Article to Peter Clark.
  - Peter Clark. Reorder toolsets in Chapter Five of the Reference Manual into "Life Cycle Order."
  - Peter Clark. Improve the Attribute Definitions and Introduction per discussions.
  - Ronnie Martin. Determine placement of the new testing-related terms in Chapter Seven of the Reference Manual and provide definitions.
- For the Guidebook Enhancements, the following action items were completed:
  - Peter Clark. Contacted Nelson Weiderman referencing attributes addressed by AES and the SEI Guidebook.
  - Bard Crawford. Enhanced the paragraph in the Guidebook referencing the element of bias inherent in all evaluation techniques.
  - CLASSWG. Reviewed new checklists prepared by Peter Clark for inclusion in Version 2.0.

- For the Whole APSE Assessment Issues, there was progress on the following:
  - Sandi Mulholland. Continue transforming customization issues (distributed APSEs, runtime support) into the checklist.
- Other Accomplishments:
  - CLASSWG. Read Version 2.0 of the Reference System and came to the meeting prepared to discuss suggestions for improvement. (Reviewed comments from Fred Francl and Rational in the working group session.)
  - CLASSWG. Reviewed feedback cards.

Key Issues:

- CLASSWG is seeking help over the following topics:
  - Configuration Management
  - Distribute Systems and Runtime Support
  - Distributed APSEs Support
  - Program Library Management Capabilities Checklist
  - Import/Export Capabilities Checklist
  - Real-Time Analysis Capabilities Checklist

Projected Work:

- Discuss Guidebook checklists: Do they address power or completeness?
- Determine usefulness of Figure x - 1, x >= 4 in the Reference Manual.
- Provide descriptions of life cycle activities for the Reference Manual.
- Determine if availability should be an attribute in Chapter Six of the Reference Manual.
- Discuss obtaining comments from the originators of technology.

Deliverables Due Next Quarter: None

Presentations Planned: None

Other Significant Information: None

Action Items:

- CLASSWG. Read Version 2.0 of the Reference System and come to the next meeting prepared to discuss suggestions for improvement. (This only applies to those members who did not do this for the June meeting.)
- Ronnie Martin. Review the Reference Manual's Testing-Related Definitions.
- Nelson Weiderman. Respond to the message referencing attributes addressed by AES and the SEI for entries in the Guidebook.
- Ronnie Martin. Review RCS manual pages for input to the Configuration Management checklist.
- Ronnie Martin. Find an example of a Change Request Form.
- Ronnie Martin. Contact Tim Lindquist and Gary McKee in reference to adding entries to the Guidebook describing Dr. Lindquist's CAIS Operational Definition and the MITRE Tests.
- Peter Clark. Review the list of debugger and library management capabilities being evaluated by Boeing.
- Fred Francl. Review the Program Management Toolset Document provided by Sandi Mulholland.
- Sandi Mulholland. Continue transforming the customization issues (distributed APSEs, runtime support) into a checklist.
- Sandi Mulholland. Further elaborate the Whole APSE Assessment Issues/Address Life Cycle Support Issues.
- Ronnie Martin. Determine appropriate Guidebook references/synopses in the Guidebook.

- CLASSWG. Search for new opportunities to improve the Guidebook; new checklists (addressing function-independent attributes), and references to E&V technology.
- Ronnie Martin/Capt. Rebecca Abraham. Bring electronic mail descriptions to the next meeting to use as input to the electronic checklist.

3.1.3 Ada Compiler Evaluation Capability Working Group (ACECWG) Status Report

Attendees:

Nelson Weiderman, chair
Sam Ashby
Mike Burlakoff
Jay Ferguson
Greg Gicca
Alan Impicciche
Elizabeth Kean
Sandi Mulholland
Ray Szymanski
Kermit Terrell
Betty Wills

Deliverables Due This Quarter: None

Accomplishments:

- Discussed the June minutes. Clarified the level of summarization required by the working group.
- Heard the ACEC contract status; reviewed release dates and deliverables.
- Reviewed SPRs. There are 50 reports with 600 individual items. They are classified as: critical, extremely desirable, desirable.
- Passed two resolutions:
  1. ACEC Version 1.0 will not be used as a basis for PTS.
  2. The following SPRs are critical and must be fixed before ACEC can be used for PTS:
     - #3 (51 problems)
     - #6 (20 problems)
     - #7 (62 problems)

     plus others that were agreed to.
- Discussed the ACEC/AES combination and the SEI recommendations to AJPO.
- Discussed possible work for ACEC Version 3.0.
- Improve quality/usability of ACEC first.
  - Address compilation system issues before the APSE issues.
  - Address compiler performance and capacity.

- Discussed Mike Burlakoff's index. There were two major issues:
  1. 20-30 categories for naming.
  2. Keywords for cross-reference.
- Discussed issues from previous meetings:
  - Naming of OOD tests.
  - Tests to include in system factor.

Key Issues:

- Ray Szymanski is interested in receiving more advice on Version 3.0 if it is funded.
- Impact of PTS on ACEC.
- Limited use and feedback from Version 1.0.
- Impact of export control on use of ACEC was not discussed.
- Overall quality is a continuing issue comprised of: documentation, organization, and the usability of the test suite.

Projected Work:

- Review of Version 2.0 issues.
- Review of Version 2.0 documents.
- Review of quality checklists.

Deliverables Due Next Quarter: None

Presentations Planned: TriAda

Other Significant Information: None

Action Items:

- Nelson Weiderman. Place status report on the NET.
- Sandi Mulholland. Synopsize the transcript from the June meeting.
- Elizabeth Kean. Investigate the availability of the Ada Tests and Verification System (ATVS) and program generator for capacity testing.
- Mike Burlakoff. Continue working on the ACEC index and place on the NET for comment.
- Greg Gicca. Continue working on the compile systems assessor document and take it to one more level of detail.
- Alan Impicciche. Develop checklists from the compiler systems assessor document.
- Ray Szymanski. Provide the Boeing checklists and the Version 2.0 documentation to those of the working group committed to reviewing it.
- Nelson Weiderman. Provide Ray with a copy of the ACEC questionnaire.
- Ray Szymanski. Review and send out questionnaire.

3.1.4 Requirements Working Group (REQWG) Status Report

Attendees:

Capt. Rebecca Abraham, substitute chair
Peter Clark
Bard Crawford
Dan Eilers
Jay Ferguson
Fred Francl
Greg Gicca
Alan Impicciche
Ronnie Martin
Nelson Weiderman

Betty Wills
Barbara Rhoads, recorder

Deliverables Due This Quarter: None

Accomplishments:

- Approved minutes.
- Composed a recommendation to AJPO for the PTS implementation.
- Established procedures for constructing Guidebook appendices to assessors.
- Combined with COORDWG.

Key Issues:

- PTS concerns.
- Keeping work within scope of E&V.

Projected Work:

- Continuation of work on the following:
  - Appendices to Tools and Aids Document.
    - need for volunteers
    - inclusion of Quality Factors Paper as appendix
  - Public Relations efforts:
    - E&Ving News published in time for TriAda
    - Consolidate information for standard press releases

Presentations Planned: None

Other Significant Information: None

Action Items:

<u>Carried Over</u>

- Sandi Mulholland. Life cycle support from Whole-APSE view.
- Betty Wills/Elizabeth Kean. Coordinate a more detailed summary of E&V activities and products suitable for public relations distribution.
- Betty Wills/Pat Maher. Verify the procedure that AdaIC could use to update the E&V team documents.
- Bard Crawford/Peter Clark. Produce an ASCII form of Version 2.0 of the Reference System.
- Betty Wills/Pat Maher. Deliver ASCII Version 2.0 to AdaIC.
- Gary McKee. Prepare a short item on the product status and give it to Betty Wills for the E&Ving News.
- Betty Wills/Elizabeth Kean. See if there is a grace period for the Ada Letters submissions.
- Sandi Mulholland. Update the Requirements Document with a pointer to the Reference Manual attribute definitions; also update date and version on title page and remove the trademark references.
- Ray Szymanski. Send out the questionnaire for feedback on ACEC.

- Bard Crawford/Marlene Hazle. Produce RED-YELLOW-GREEN poster on relationships among assessors, tools, and executing software.
- Greg Gicca. Continue work on the Tools and Aids appendix for the compilation systems; place new draft on the NET.
- Jay Ferguson/Lloyd Stiles/Sandi Mulholland. Continue work on the Quality Factors paper. NEW ITEM: Tailor this information to fit the assessors and to be included as an appendix to the Tools and Aids Document.

<u>New</u>

- Peter Clark. Send Betty Wills a copy of Ray Szymanski's NAECON paper for a guide in writing the public relations blurb.
- Elizabeth Kean/Betty Wills. Place the text of the E&Ving News on the NET by 25 SEP 89 for review by the team; go final 02 OCT 89.
- Elizabeth Kean/Betty Wills. FAX Ray Szymanski a copy of the draft and final version of the newsletter to review the overall look.
- Ray Szymanski. Establish a procedure for printing and distributing the E&Ving News.
- Barbara Rhoads. Transcribe "Appendectomy Procedures" and send to Ronnie Martin.
- Capt. Rebecca Abraham. Place Nelson Weiderman's recommendation for the PTS implementation on the NET.
- Capt. Rebecca Abraham. Put REQWG status report on the NET.

3.1.5 CAIS Implementation Validation Capability Working Group (CIVCWG) Status Report

Attendees:

Gary McKee, chair
Karyl Adams
Kurt Gutzmann
Kevin Hackett
Tim Lindquist
John McBride
Ray Szymanski

Visitors:
Dave Carney, IDA
Bruce Taylor, Intermetrics
Jerry Thomas, NOSC

Accomplishments:

- Review of CIVC contract status.
- Discussed the framework product.
- Discussed the strategy for PQT/FQT.
- Discussed the strategy for the coordination with the CAIS-A IV&V activity.

Projected Work:

- IV&V activities by Tim Lindquist and Gary McKee.
- Acquisition of the TRW CAIS and the SofTech CAIS-A implementations.

Action Items:

- SofTech. Send test cases and documentation to Tim Lindquist and Gary McKee for IV&V.
- SofTech. Send the framework to Gary McKee for review by 01 OCT 89.
- Bruce Taylor. Produce a white paper on the rule-based software for test selection.
- Jerry Thomas. Facilitate the acquisition of the CAIS/CAIS-A implementations.

3.2 Closing Remarks

Mr. Szymanski provided the team with information for the December meeting and thanked them for a productive and successful meeting. The September meeting was then adjourned.

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX A

10.0 LIST OF ATTENDEES

Abraham, Capt. Rebecca
WRDC/FDCL
WPAFB, OH 4543-6543

Adams, Karyl
C.J. Kemp Systems, Inc.
3318 E. DryCreek Road
Phoenix, AZ 85044

Ashby, Sam
Boeing Military Airplane Co.
P.O. Box 7730, MSK80-13
Wichita, KS 67277-7730

Brashear, Phil
SofTech, Inc.
3100 Presidential Drive
Dayton, OH

Burlakoff, Mike
Southwest Missouri State University
Computer Science Dept.
Springfield, MO 65804

Carney, David
IDA

Clark, Peter
TASC
55 Walkers Brook Drive
Reading, MA 01867

Crawford, Dr. Bard
TASC
55 Walkers Brook Drive
Reading, MA 01867

Eilers, Dan
Irvine Compiler Corp.
18021 Sky Park Circle, #L
Irvine, CA 92714

Ferguson, Jay
National Security Agency
ATTN: T303
9800 Savage Rd.
Ft. Meade, MD 20755-6000

Francl, Fred
Sonicraft, Inc.
8859 S. Greenwood
Chicago, IL 60619

Gicca, Greg
Sanders Associates
NCA1-2232
95 Canal Street
Nashua, NH 03061

Gutzmann, Kurt
SofTech, Inc.
1300 Hercules Drive
Suite 105
Houston, TX 77058

Hackett, Kevin
SofTech, Inc.
16875 West Bernardo Drive
San Diego, CA 92127

Impicciche, Alan
Naval Avionics Center
NAC Code 826
6000 E. 21st Street
Indianapolis, IN 46219

Kean, Elizabeth
RADC/COEE
Griffiss AFB, NY 13441-5700

Kirkbride, Kathy
Oneida Resources, Inc.
3578 Kettering Blvd.
Suite 300
Dayton, OH 45439

Lange, Dale
ASD
WPAFB, OH 45433

Lawlis, Maj. Patricia
AFIT/ASU
3318 E. Dry Creek Road
Phoenix, AZ 85044

Lindquist, Dr. Tim
Computer Science Dept.
Arizona State University
Tempe, AZ 85287-5406

McBride, John
SofTech, Inc.
1300 Hercules Drive
Suite 105
Houston, TX 77058

McKee, Gary
McKee Consulting
P.O. Box 3009
Littleton, CO 80161-3009

Martin, Ronnie
Software Engineering Research Center
Dept. of Computer Science
Purdue University
West Lafayette, IN 47907-2004

Mulholland, Sandi
Rockwell International
400 Collins Rd., NE
MS124-211
Cedar Rapids, IA 52498

Payton, Teri
UNISYS Corporation
12010 Sunrise Valley Drive
Reston, VA 22091

Rhoads, Barbara
Oneida Resources, Inc.
3578 Kettering Blvd.
Suite 300
Dayton, OH 45439

Startzman, Ty
Boeing Military Airplane Co.
P.O. Box 7730, MSK80-13
Wichita, KS 67277-7730

Stiverson, Rich
Boeing Military Airplane Co.
P.O. Box 7730, MSK80-13
Wichita, KS 67277-7730

Szymanski, Ray
WRDC/AAAF-3
WPAFB, OH 45433-6523

Taylor, Bruce
Intermetrics Inc.
733 Concord Ave.
Cambridge, MA 02138

Terrell, Kermit
Boeing Military Airplane Company
P.O. Box 7730, MSK80-13
Wichita, KS 67277-7730

Thomas, Jerry
NOSC

Wills, Betty
CCSC/XPTB
Tinker AFB, OK 73145

APPENDIX M

MINUTES

OF THE

EVALUATION AND VALIDATION TEAM

GENERAL SESSION

4-7 DECEMBER 1989

The task for the Evaluation and Validation of Ada Programming Support Environments (APSEs) is sponsored by the Ada Joint Program Office (AJPO).

TABLE OF CONTENTS

Section Page

1.0 TUESDAY, 05 DECEMBER 1989 3
1.1 Chairman's Corner 3
1.2 The UK Ada Evaluation System 4
1.3 Supporting Selection Decisions Based on the Technical Evaluation of Ada Environments and Their Components . 7
1.4 CAIS Implementation Validation Capability (CIVC) Project Update 10
1.5 Ada Compiler Evaluation Capability (ACEC) Update 12
1.6 Reference System Update 13
2.0 WORKING GROUP STATUS REPORTS 15
2.1 Ada Compiler Evaluation Capability Working Group (ACECWG) Status Report 15
2.2 CAIS Implementation Validation Capability Working Group (CIVCWG) Status Report 17
2.3 Classification Working Group (CLASSWG) Status Report . . . 18
2.4 Requirements Working Group (REQWG) Status Report 21
2.5 Standards Evaluation and Validation Working Group (SEVWG) Status Report 22

LIST OF APPENDICES

10.0 LIST OF ATTENDEES 25

1.0 TUESDAY, 05 DECEMBER 1989

1.1 Chairman's Corner

Raymond Szymanski
Wright Research and Development Center (WRDC)
Wright-Patterson Air Force Base (WPAFB)

Mr. Szymanski welcomed the Evaluation and Validation (E&V) Team to San Diego. He introduced Lloyed Stiles, the host of this quarter's meeting, who provided the team with a brief orientation. Mr. Szymanski then informed the team of an agenda change. The first speaker, John LeBaron from the Army Center for Software Engineering, was unable to attend. Mr. Walter Marks was attending the meeting in his place.

The following introductions/announcements were made:

- Lynn Chilson from SofTech Houston is the new program manager for the CAIS Implementation Validation Capability (CIVC).
- Karla Nohalty from The Analytic Science Corporation (TASC) in Reading, Massachusetts will give a presentation to the Requirements Working Group on the model project.
- Mr. Phil Brashear is now a member of the E&V team. He was formerly with the Ada Compiler Validation Facility with SofTech Dayton and is now with CTA, Incorporated.
- Tom Leavitt from Boeing will not be attending the meeting due to illness.

Activities since the last meeting:

- Mr. Szymanski gave a presentation at the TriAda conference in late October concerning the Ada Compiler Evaluation Capability (ACEC) and the Quality Testing Service (QTS). As a result, Mr. Szymanski was invited by the Government Accounting Office to answer questions on the QTS.
- Mr. Szymanski met with Mr. Joe Batz who is involved with technology on an international scale. The meeting covered future possibilities of the ACEC and the QTS.
- Mr. Szymanski was a guest lecturer at the Navy Post-Graduate School in Monterey. He spoke on the E&V activity.
- Nelson Weiderman gave a presentation at TriAda on Ada and the selection method for compilers.
- The next E&V Team meeting will be March 6-8.

Mr. Szymanski opened the floor for questions from the Team.

Mr. Weiderman asked what the schedule was for getting the ACEC out. Mr. Szymanski replied that three copies had already been shipped. A meeting was held over a year ago between Joe Batz and the U.K. representative who asked for the ACEC. After a few delays, three copies were shipped on November 1. Copies were shipped to the U.K., the Danish government, and Spain, respectively.

Dr. Crawford asked if this was an indication that the ACEC will be shipped from government to government in one copy as opposed to sending it to foreign firms. Mr. Szymanski stated the Clearinghouse Newsletter would soon carry an article on the process for nondomestic entities receiving the ACEC. The ultimate authority for releasing the ACEC is ASD/XRID at Wright-Patterson Air Force Base.

Mr. Weiderman asked what restrictions will be used. Mr. Szymanski answered that the basic restrictions apply. Dr. Crawford questioned if the receivers were basically defense contracts as in the U.S. Mr. Szymanski explained that the basic method was for the embassy to make a request to Air Force Systems Command headquarters and then to the ASD/XRID.

Mr. Weiderman felt the reception at the meeting on the ACEC and QTS was fairly mild, although some criticisms and questions were somewhat hostile.

Mr. Szymanski said that the chief technical investigator from the Government Accounting Office (GAO) stated he had heard only good things about the ACEC. The investigator asked for an explanation why some people were saying that the QTS was unnecessary. Mr. Szymanski felt the reasons were a lack of participation in the audience (from known supporters) and his inability to provide details of the QTS document on the intended uses and details due to time constraints. Since the plan was late in being sent out, most of the audience had probably not seen a copy. The negative reaction was attributed to fear of the unknown.

Dr. Crawford felt that the presentation was generally well received. He thought the most critical comments were from panel member Dan Eilers. Dr. Crawford believed it would create a good impression that the most severe critic was a team member who would help to ensure development of a quality product. Mr. Weiderman asked if anymore information or dates was available on the QTS. Mr. Szymanski indicated that he would be meeting soon with Dr. Solomond to discuss these issues.

Mr. Szymanski then introduced Nelson Weiderman as the first speaker of the day.

1.2 The UD Ada Evaluation System

Nelson Weiderman
Software Engineering Institute (SEI)

Mr. Weiderman briefed the team on the latest version of the Ada Evaluation System (AES), Version 2.1. BSI announced the release of the latest version of AES in a September 1989 newsletter. This product contains approximately 300,000 lines of Ada code. The cost of AES, Version 2.1 is 1,200 pounds sterling which is 200 pounds sterling more than the last version.

The AES function has been expanded to cover the minimal APSE. The expanded areas are the evaluation of the Program Library System, the linker/loader, and a debugger. The methodology has also been expanded. The former AES was based only on tests and a questionnaire. The new AES uses scenarios which are scripts of possible actions to take when using a debugger or Program Library System. Revisions have been made to the questionnaire to make it more objective. Because of problems with portability, the porting guide has also been revised making it more concise and user friendly.

Desktop publishing technology was used to produce the Reader's Guide for the AES and a report on one of the systems evaluated last spring. The Reader's Guide for the AES is a 180 page report covering all aspects of the evaluation process. A 150 page product report discussing every compilation system evaluated is included. The reports provide a general idea of the overall quality of a compilation system and help readers select a compiler suited to their project requirements. The product report focuses on the individual compiler's strengths and weaknesses, but makes no comparison among compilers. The cost of the individual reports has stayed the same, 250 pounds, and the subscription service has continued.

The Reader's Guide has three parts: Part one is general background information; Part two provides the breakdown of eight different categories of evaluation: Compilation Phase, Execution Phase, Run-time Support, Language Features, Program Library System, Linker and Loader, Debugger, and Commercial Considerations; and Part three describes the 24 categories of tests. Part three describes in a general way the tests included in those categories and their intended accomplishments. The new test categories are for debugger behavioral tests, debugger capacity tests, Program Library System scenario support tests, erroneous execution tests, incorrect order dependent tests, and Program Library System support units.

The second volume contains the actual evaluation information for a particular compilation system. The introduction instructs the reader on interpretingthe reports and provides some of the icons, etc. that are used in the report. It contains a management summary and some product information. Another level of the summary called the Per-chapter Summary contains eight chapters: The Compiler, The Execution Phase, Run-Time Support, Language Features, The Program Library System, The Linker and Loader, The Debugger, and Commercial Details. These correspond to Part 2 of the Reader's Guide and are cross-referenced to it. Also included is a series of four appendices: Configuration Details, Running of Tests, Example of Outputs, and Vendor's Comments.

Numerous tables are given in the Evaluation Reports to provide visual graphical information as well as numbers. All of the sections are keyed back to the Reader's Guide by section number and cross-references are given to the applicable test in the Reader's Guide.

Weighting factors are given for the questions, and the database contains two numbers. The first number is the importance factor of the question and the second number is the weight given to the question in the evaluation summary. These numbers are user selectable with the idea of basing them on the user application. There is a heavy use of icons in the reports. These are placed in the margins to indicate information that is special in some way. Extra or deduced information is clearly marked. A bar in the margin indicates that this is not part of the automatically produced report. The primary information in the right-hand margin contains extraneous information such as assesser's comments. Non-relevant questions are omitted. If a section of questions does not apply, it is not included in the report. The automated production of the report is not available in the do-it-yourself version. An estimated 80% of the report is produced automatically from the database. The successor's comments have to be inserted manually, so it is a fairly automated process to produce these reports for new compilation systems.

Mr. Burlakoff wondered if the vendor reports should be sold to anyone willing to pay for them. Mr. Weiderman replied that it was the same for the AES as for the ACEC; the availability of the reports is going to improve the quality of the compilers as it will encourage those with good products to get the information out while discouraging those having poor products.

Mr. Weiderman's evaluations included several strengths and weaknesses. The strengths were that this volume contains a tremendous amount of useful information. The organization and the cross-referencing were excellent, with the tests organized into convenient categories.

One of the weaknesses is the weighting scheme. Its use was not readily understood and there was no evidence of the weighting scheme in the reports. He suggested tnat the weighting system, if used at all, be for the do-it-yourself person who can assign weights based on their application, but it does not seem worthwhile in a general report. Another weakness is a lack of confidence in the UNIX timings. The description of dual benchmarks and the problems involved with them were insufficient. It was not analyzed correctly and the interpretation was not presented in the Reader's Guide. A third weakness was that the cross-referencing is only for the tests and it does not include the questionnaire and scenarios. For any evaluation information, there should not only be a pointer back to the test that was used, but if it is a test, the pointer should go back to the number. If it is based on a scenario, there should be some way of referencing the scenario step that was used to get the answer to this question. The final weakness mentioned was that the chapter summaries are weak. Although they try to highlight information, the summaries do not provide enough information to be useful.

Mr. Burlakoff asked Mr. Weiderman to comment on the depth and breadth of coverage. Mr. Weiderman said he focused on the reports based on the two volumes. He had not commented on the coverage but he felt it was fairly good. The depth of coverage was not as complete as the ACEC but the breadth was good. In some areas such as optimization tests, AES does not go as far as the ACEC does.

Dr. Crawford asked for a general comparison of the ACEC and AES and if they were complimentary. He wondered what Mr. Weiderman thought of an ultimate merging of the two. Mr. Weiderman stated that they were complimentary and that much could be gained by combining the best of each. He was not, however, optimistic about that occurring.

Dr. Lindquist agreed with Mr. Weiderman about the confusion over the weighting factor and the importance. He could see some situations where importance may be worthwhile including in the report, but the weighting factor on the other hand should be completely disregarded. He felt that some of the other summary information was appropriate.

Mr. Gutzmann wondered what the coverage was on errors. He had noticed a number of items that related to correctness. Mr. Weiderman replied that several categories of tests deal with errors. He then closed his presentation.

1.3 Supporting Selection Decisions Based on the Technical Evaluation of Ada Environments and Their Components

Major Patricia Lawlis
AFIT/ENC
WPAFB

Major Lawlis' presentation covered her proposal, research activities, and analysis for her dissertation. The proposal was divided into three areas: the problem, reason for its importance, and the proposed action. The problem is the difficulty in the selection of software for system development. The decision is usually made by a technical manager while evaluations are done by the technical staff or an outside organization. So those decision-makers not having performed the evaluations do not necessarily understand all of the voluminous data which comes out of the evaluation situations.

As to the reason for its importance, large software projects are very expensive and many fail due to software tools or the inadequacy of the evaluation technology. Software engineering as a young discipline has no standard methods for software development, no standard software tools, and no standard methods for evaluating software. Major Lawlis proposed a vehicle which can assist a decision-maker in selecting the appropriate software. This vehicle would consider all the important evaluation aspects and maintain consistency.

The premise is that any one evaluation by its very nature is biased and is not comprehensive. The ACEC and AES as an example address only a very small portion of the whole spectrum of evaluations that can possibly be made on software. The idea was to define a computerized decisions support system which would be able to present things in a consistent manner, maintain consistency, and be able to get a decision-maker through the weighting process, etc. A prototype was implemented covering these aspects.

The research activities are very similar to those that go on with software development and include preliminary, requirements, design, and prototype. The preliminary efforts were the proposal itself and the development and distribution of a questionnaire. The questionnaire was directed to anyone on the E&V Team willing to participate. The questionnaire results covered the information to be contained in a decision support system, the methods for development, etc. The results affirmed the initial ideas but also provided an emphasis. A requirements document was started along with some of the prototype development. Provision was made for verification in the requirements document and in the development of a test plan. It provided test descriptions and cross references from the requirements to the design and tests.

Design documentation was prepared covering the documentation for each module, diagrams of each level, interfaces for each package, and PDL for lower level subprograms. The subsystem coverage included decision logic and part of the knowledge base. The subsystem coverage was developed in four subsystems using the basic structure of a typical system support decision system. This is a complete high level design. The Knowledge Base Subsystem contains both the knowledge of the system and the basic data from the evaluation. The Knowledge Acquisition Subsystem would be an automated method of bringing in the data from the files and putting it into the knowledge base in a manner that it can be used by the system. The User Interface Subsystem is the basic model and the Logic Subsystem controls everything and provides a method of acquiring knowledge in the system.

The main area covered was the decision logic part of the system. For the Knowledge Base Subsystem, the structures and functions were designed going into the knowledge base used by the system as it pulled information out of the database. Nothing was done to the Knowledge Acquisition Subsystem. Information for the User Interface Subsystem was done manually which was the reason for including all the tools.

The prototype was proof of concept and demonstrates the system's capabilities. Version 1.0 was basically a storyboard without much function. It provided feedback to the requirements. For Version 2.0, one level of detailed specifications was entered to specify the types of things to be considered in the selection process. Although the prototype does work with the compilation system, it tends to do a great deal more than perform the evaluation process. A user's manual was also developed.

Accomplishments include the development of a general evaluation framework for APSEs which applies the framework to software as well as APSEs. The attempt was to make the framework flexible and expandable, and because of the knowledge base structure, the knowledge base can be easily modified for a more general application. The viability of the concepts was demonstrated through the use of the prototype on the compilation systems. The calculations used were based on the concept of decision theory.

The evaluation framework takes information with the data that comes from other sources and attempts to place it into the database to be used in assisting a decision-maker in selecting software. The problem of organizing the data has been dealt with in a number of ways. A considerable amount of work has been done in this area and is included in the E&V Reference Manual. This provided help in determining the high level organization. Absolute criteria and relative criteria is a good breakdown of the two different types of information that would go into evaluation. The terminology of the Reference Manual was not used as there was no commonality.

Absolute characterized features are so called as they are typically features of the software. The criteria of the relative characteristics are called quality factors in some places but have different names in different places, and are used to make comparisons. A complete glossary was developed including all of the various definitions derived from using the features and criteria as a reference point.

The features include functions that are in the software and configurations. The range of value requirements include items such as cost restrictions on the compilation system and memory specifications. The range of value requirements can be set up in other forms, for example an "absolute requirement" is a requirement that cannot be violated. This was the most difficult part of the research because it is an area that has not received extensive work in organization. The E&V Reference Manual deals with it by combining many different areas with relative areas. Here they were separated. The top level features or categories of various absolutes from the evaluation process include some items from the E&V Reference System in terms of functions. These include analysis functions, management functions, and transformation functions. Many others were added.

Cost was included separately as it is such an important area. This is not the actual cost of purchasing the software but the cost of training and other items considered in making a selection decision.

Of the various kinds of features considered, a decision-maker could choose those which are important and the system will then provide reasonable default values for the remaining features. The decision-maker can make appropriate changes to the default values. It can also be used by someone more management oriented who may not be familiar with many detailed areas and may prefer to have a lot of defaults already set. The intent was to make this usable by management-oriented individuals as well as decision-makers.

The criteria or relative characteristics include all factors which determine the quality of the software. Much of the work had already been done in this area. The top-level criteria are also called quality factors or attributes while the detail criteria are sometimes called metrics. In the top level criteria, vendor support is extremely important in any type of selection.

The most difficult problem for decision-makers is how to organize the evaluation and information to arrive at a reasonable decision. The system has to be able to arrive at recommendations. The data must be compared and the easiest way to do that is by using common terminology and some type of ordinal function for comparisons. The system then comes up with a numerical result for the evaluations made based on what the user has indicated is important.

One significant digit from one to ten, with ten being high, was used for the weighting process. In the feature area, a feature absolutely required is given a ten. Anything lower than that is highly desired. As far as the criteria are concerned, a ten indicates maximum importance. A rating of two is considered good. For a feature, two indicates the feature is complete. A rating of one is acceptable, or in the case of a feature, it partially implements that feature. Zero is unacceptable, or the feature is not contained. Combinations are done in relation to the calculations of the ratings and the weights.

From any particular area, linear added functions are a part of what might be in the system for the overall feature rating. The feature categories could include user profile, hardware control, configuration requirements, and cost. Out of the user profile, two lower level items were chosen which would be the training and skill level required of the user. For configuration requirements, there is operating system, target hardware, and host memory as specific requirements. Particular hardware and particular host memory are also requirements.

The system provides the decision-maker with a list of items in the database to be considered, a list of items acceptable based on these rating, and another list of those items that were unacceptable. It can also provide a list of those items that were not even considered because they did meet the stated requirements. The list is created according to the ratings, but only if the user requests it. It will also explain how the ratings were determined and give the user an opportunity to make some changes to some of the basic parameters.

Of the lessons learned, the most important one is that it is not easy to acquire evaluation data. Although in-house evaluations are important, they are not sufficient; evaluation organizations are important. It would be a good idea to have a repository where data can be gathered to provide a better overall evaluation than using pieces of evaluations from various places not covered as well as what a lot of different evaluations could cover. An obvious problem with this is how to keep the database current. Vendor information is a very important type of evaluation data for this database but it should not be the only type. Ultimately, the user should be able to weight how much stock to put in the vendor data as opposed to other sources as well.

Technical and managerial personnel have different expectations; the attempt was made to have ASSIST deal with whatever would be comfortable for either one, where a more managerial-oriented person would not have to get into the low level technical details. There is also the possibility of accessing the low level technical details for those who were interested.

As this is only a prototype, more development will be required for it to become a valuable tool for the selection process. The design must be completed and fully implemented.

The concept and design of ASSIST is a significant contribution to software evaluation technology. It is a basis for comprehensive evaluation framework and furnishes a foundation for further advances. The evaluation framework provides a guide for more complete software assessments. The prototype supplies a vehicle for expanding the framework. ASSIST is a blueprint for practical use which was the major goal in developing the prototype. Major Lawlis closed her presentation with a brief question and answer session.

1.4 CAIS Implementation Validation Capability (CIVC) Project Update

Kurt Gutzmann
SofTech, Inc.

Mr. Kurt Gutzmann presented the CAIS Implementation Validation Capability (CIVC) status report. The following documents have been delivered: the Software Test Plan, Software Test Procedures, Test Report Reader's Guide, the Implementor's Guide, Software Product Specification without the listings, and the review version of the hypertext framework which was delivered to the Independent Verificaiton and Validation (IV&V) consultants. The 253 test cases implemented in the CIVC1 Test Suite are organized as 14 test classes and three superclasses. SofTech is applying a Coverage Analysis Tool to the Test Suite. The results of that test will be in the Phase 1 report that will show the number of phases covered. The Test Administrator was integrated with the Test Suite and is working well.

The Operator's Guide is prepared and will be delivered as an appendix to the Test Report Reader's Guide. The Version Description Document and the Software Product Specification with listings are ready for delivery. The CAIS issues are prepared; 38 issues are identified for communication to the CAIS Editorial Board. The Phase 1 Report is in preparation. CIVC is ready for the Formal Qualification Test (FQT).

The following program changes have occurred:

- Lynn Chilson is the new CIVC Program Manager.
- Kurt Gutzmann is the CIVC Project Manager.

The CIVC hypertext traceability framework is a Macintosh-based tool. The object-oriented taxonomy of CAIS entities is to aid in coverage analysis. The Guide product from OWL is the hypertext software. With the OWL's guides for the PC and MS DOS compatible file names, the Guide files will work on the PC version of the Guide also.

It is not necessary to have Guide to exercise the product on the Macintosh. An envelope function will provide a read-only copy which can be driven around in a read-only mode. Problems might occur in producing that envelope as the memory size must be equivalent to the multiple size of the envelope to create. Benefits of the hypertext traceability include:

- A quick and easy examination of traceability relationships through several different documents.

- Assistance in the verification and validation of traceability correctness.

- Generation of reports and metrics on the hypertext network for requirements flow-down and coverage analysis.

- Extension of the test case code along with other documents having ultimately the whole project in hypertext management.

An ER diagram was shown including the 1838 specification and test objectives. The test objectives are related to one or more scenarios. These are not stored in the framework products but can be added easily. The taxonomy is a description of the object or entities existing in the interfaces. Test objectives and scenarios are tagged onto these in the framework.

A slide was displayed showing the hypertext links navigable by the user. Mr. Gutzmann then gave a demonstration on the Macintosh of Guide. The envelope framework is one file, 4.3 megabytes. This size does not prohibit running it on a Macintosh with a small memory.

Mr. McKee asked if the hypertext files are folded and if they are required to be in there. Mr. Gutzmann replied that everything needed is in that file. The envelope is the read-only version and is all that is needed by the user.

Mr. Brashear asked if any way had been discovered to automate the creation of the links or were they created one at a time. Mr. Gutzmann responded that it is a manual process. OWL will not reveal the internal file. Some alternatives are being investigated that will allow easier ways to construct it, to modify its representation, or display formalities as chosen. Currently, there is a dependency on the goodwill of OWL. The envelope product is no longer supported.

Mr. McKee said that information had to be place in two different locations, in the CM and under Guide. He wondered if there was a type of vehicle for sitting a pointer into the Guide product that points to a file outside of the Guide pilot. Mr. Gutzmann replied no; the test cases are keyed in a configuration management tool of which the user must check in and out. Other project objects can also be connected in hypertext frameability. All of the project documents and other related information, i.e., design information, for the test add more code.

Mr. Szymanski stated that was an enormous amount of manual effort. Mr. Gutzmann replied he did not see any way to avoid human inspection. It will be very difficult to automate verification in QA in traceability networks.

Dr. Crawford asked what was the primary reason for use of this product. Mr. Gutzmann responded that it served a dual purpose. It would help the ultimate user in analyzing the kinds of tests that can provide implementation, and it helps to manage the relationships of project objects for the effort in a more maintainable fashion than a paper traceability tape.

Mr. Gutzman concluded by listing all the test objectives needed in the framework. The user could then go straight into a particular test objective. Currently, the user must go through 1838 which is not always the direct way.

1.5 Ada Compiler Evaluation Capability (ACEC) Update

Sam Ashby
Boeing Military Airplane Company

Mr. Ashby stated that Boeing is going through a significant organization effort. Boeing Military becomes a division of Boeing Commercial as of January. There are currently two unapproved plans for the ACEC. Boeing Military's work on the ACEC could be combined with some of the other work in the other operating companies. There is a great deal of effort involved with Ada; NASA is involved with both Ada and Boeing. As of now, the Ada team will remain intact and the work will continue.

In November the development work was completed on Release 2 of ACEC. FQT was started with Ray Szymanski, Gary McKee, and Mike Burlakoff. This will be done on five systems: DEC Ada, Telesoft, Apollo, the Air Force System Compiler (AIMS), and Silicon Graphics which is a mips processor and is self-hosted and targeted using the Verdix compiler.

Release 2 has in excess of 300 new tests for the ACEC. It has assessors for the Diagnostic System, the debugger and Library Assessor, and the Single System Analysis. These were found to be more difficult to use than envisioned. To run the Test Suite, the user had to have knowledge of the compiler and how to get the assessors to run. The difficulty is increased as there are no standards for these assessors.

Boeing was asked by SofTech to submit a proposal for training. A proposal has been submitted and this is tentively scheduled for the first of the year. This comparison training would take place in Dayton, Ohio.

Release 2 of the ACEC is scheduled to be released in December 1989. An optimistic schedule for revision is in January 1990.

Mr. Szymanski questioned some problems Boeing was having with the systems. Mr. Ashby replied that it was possible that not everything would be completed on every system, but if not, it will be close to completion.

Dr. Crawford asked if there was a tentative schedule for Release 3 and how different would it be from Release 2. Mr. Szymanski answered that although Dr. Solomond supports the idea of producing Release 3, but no further action has been taken.

Mr. Weiderman asked if in the future there would be an RFP for a QTS. Mr. Ashby replied that in his discussions with SofTech it had been affirmed that was the Air Force directive. This led to SofTech's request for Boeing to prepare an execution of the ACEC. Mr. Szymanski added that if there was a QTS, SofTech may be making a change in the specification to the contract which would enable them to do compile evaluations along with compiler validations. He assumed the reasoning was in preparation for a positive decision by John Solomond to implement this. If so, someone within the Government would be trained to do that initially before letting the contract go out to separate the two entities. The quickest way would be to have it done under the initial contract. As far as being prepared, it is probably a minimal investment at this point.

1.6 Reference System Update

Dr. Bard Crawford
The Analytic Sciences Corporation (TASC)

Dr. Crawford's status report consisted of two parts. The first part is the Reference System which comprises the E&V Reference Manual and the Guidebook. Version 1.2 of both documents has been recently released. This will be discussed further along with the plans for Version 2. The second part of the status report deals with the whole-APSE evaluation by a model project.

Version 2.0 of the Reference System has been completed and approved by the Air Force. It has been released by the Department of Defense for unlimited distribution. The distribution of the mailing list is presently underway. The electronic versions of the Reference System is presently being created. An ASCII file out of the publications department will be sent to the Clearinghouse. It is also being put in WordPerfect format for easier updating in later versions. For Version 3.0, checklists will continue to be added and existing checklists expanded. The checklists will correspond to public feedback as acquired from the questionnaires. The current focus is the conversion of 3.0.

The CASE tool assessment area comprises CASE Research Corp., U.S. Army CECOM, and Methodology support - CECOM. Some effort will be made to expand on it. There is the notion that the real benefit of Ada should come in the whole life cycle, particularly in the area of software maintenance. Attention should be given to the maintenance people such as FCDSSA and others in the Army and Navy.

In community interaction under the Reference System, the following occurred since the September meeting.

- Mr. Szymanski gave a presentation at the IEEE Working Group on CASE Evaluation Standards.

- A panel discussion at TriAda focused on the ACEC and compiler evaluation. Mr. Szymanski's presentation covered E&V as a whole and mentioned the Reference System.

In the development activities, a Model Project has been selected which is an Ada project developed by the SEI. In addition, three other activities are occurring simultaneously. One has to do with the source code which does compile in the computer. Slight modifications are being made to the code in addition to testing it as part of the first scenario. Along with that is the creation of other projects, such as a Model Project documentation using 2167A format, taking documents and putting them into the document format. Otherwise, documentation has to be written from scratch. The evaluation scenarios are being developed simultaneously and will be tested and documented.

The first project is called scenario number 1. The projections come from the SEI which was produced from the Naval Surface Weapon Center. It has two major parts running on two different computers, one in the INS simulator and external computer. The focus is on the INS. To build early scenarios running on one computer, modifications have to be done to the interface codes to simulate what happens in that interaction.

Mr. Ashby asked about the size of the code and if it has recently been completed. Dr. Crawford replied it is 10,000 lines of code and has been used for other purposes at the SEI to support other projects. One of the reasons for selection was all of the Ada work already done. This meant a lot of time would not have to be spent in creating or obtaining permission for release of proprietary material.

The Model Project itself will be contained in the code as well as in the six documents. The Software Requirements Specification for the first scenario is about 50% completedted. The lower level details need to be provided in some areas. The System/Segment Specification is nearly completed. None of these documents have been delivered.

The scenario will be tested and then packaged into one group of documents representing the incomplete Model Project but complete enough to support another scenario. There is also a draft of the scenario document.

Mr. McKee asked when Version 3.0 of the Reference System would be released. Dr. Crawford stated that the plan is to repeat the cycle. A nearly complete draft will be ready for the cursory review in June. It will be finalized between June and September. Version 3.0 should be released next fall. It is expected that Version 4.0 will be developed the following year which will probably be a small upgrade.

2.0 WORKING GROUP STATUS REPORTS

2.1 Ada Compiler Evaluation Capability Working Group (ACECWG) Status Report

Attendees:

Nelson Weiderman, Chair
Sam Ashby
Phil Brashear
Mike Burlakoff
Bard Crawford
Dan Eilers
Jay Ferguson
Greg Gicca
Liz Kean
Pat Maher
Gary McKee
Lloyd Stiles
Ray Szymanski
Kermit Terrell
Betty Wills

Visitors:
Brian Riegel

Deliverables Due This Quarter: None

Accomplishments:

- Reviewed the minutes and decided to accept them "as is"; the action items from the last meeting were reviewed.
- Reviewed the ACEC contract status including the release dates and deliverables. Also reviewed lessons learned from the recently completed FQT, the status of documentation (particularly with respect to new assessors), the status of IV&V items (in general), and the status of the AIFRAME program, the naming of the OOD tests, and basketball/golf score issues in particular.
- Reviewed feedback on the ACEC received through telephone conversations with Ray Szymanski, a letter from TeleSoft, and through a report from a subcontractor to FCDSSA.
- Reviewed the keyword index generated by Mike Burlakoff and provided feedback on its enhancement.

- Reviewed eight suggestions for Version 3 and Version 4 of the ACEC as supplied on the NET by Ray Szymanski, added six more suggestions and took a straw poll of the importance of the resulting 14 categories.

Key Issues:

- Further advice needed by the Government on ACEC Version 3.0 and Version 4.0.
- Completion of the indexing system for Version 2.0.
- Restoration of funding to Boeing so that work can continue and the product can be delivered.
- Impact on ACEC of QTS (not discussed this time).
- Limited use and feedback of ACEC Version 1.0 (couple of dozen copies distributed, but only a dozen problem reports from other than team members).
- Overall quality of ACEC:
  - Documentation
  - Organization
  - Usability

Projected Work:

- Review of Mike Burlakoff's indexes.
- Review and setting of priority for Version 3 and Version 4 issues.

Deliverables Due Next Quarter: None

Presentations Planned:

- Lloyd Stiles: Feedback on ACEC's use to evaluate ALS/N.

Other Significant Information: None

Action Items:

<u>Carried Over</u>

- Liz Kean. Investigate availability of Ada Test and Verification System (ATVS) and program generator for capacity testing.
- Ray Szymanski. Review and send out questionnaire.

<u>New</u>

- Nelson Weiderman. Put status report on the NET.
- Mike Burlakoff. Revise ACEC index and put on the NET (second time).
- Ray Szymanski. Make sure that announcement for Version 2 and ordering information gets into the Ada Information Clearinghouse Newsletter well before availability.
- Liz Kean. find out if DACS will put announcement in DACS Newsletter and get information to Ray Szymanski.
- Lloyd Stiles. Report at March meeting on experiences of FCDSSA with ACEC for evaluating Navy's ALS/N compilers.
- Ray Szymanski. Put the voting for the Version 3/Version 4 follow-on work on the NET and call for a new round of voting.

2.2 CAIS Implementation Validation Capability Working Group (CIVCWG) Status Report

Attendees:

Gary McKee, Chair
Karyl Adams
Phil Brashear
Lynn Chilson
Jack Foidl
Kurt Gutzmann
Kevin Hackett
Tim Lindquist
John McBride
Ray Szymanski
Bruce Taylor
Jerry Thomas
Valerie Rodriquez, recorder

Deliverables Due This Quarter:

- Presentation by SofTech on the status of the CIVC contract and on preparations for the Formal Qualification Testing (FQT)/Functional Configuration Audit (FCA) from the 13th to the 15th of December.

Accomplishments:

- Discussion of the CIVC status, based on the SofTech presentation. Developed a plan for Tim Lindquist to evaluate the CIVC test suite at Arizona State University. Discussed issues and ideas related to the transition of the CIVC contract from CAIS to CAIS-A in January 1990.
- Detailed discussion of the pre-FQT software analysis kit for test-class #8 and of several documentation or implementation issues related to test-class #8. Discussed schedules for FQT/FCA (13-15 DEC 89) and for PCA (mid-January 1990).
- Reviewed and discussed a white paper on expert systems produced by Bruce Taylor. This is in regards to finding ways to automate the setup, execution, and maintenance of the CIVC.
- Discussed taxonomy issues for CAIS-A and CIVC-A. Developed a list of changes that were needed in the Framework product for CIVC-A.
- Received and discussed a presentation by Jack Foidl on his current project, "The CAIS Appropriateness Study." This study is to evaluate the actual usability of the CAIS by porting tools over to the CAIS and evaluating their effectiveness.

Key Issues:

- The use of the SofTech "Coverage Analysis Tool" provided in support of the CIVC requires the use of Microsoft File, a commercial program that is not delivered as part of the contract.

- Completion of the FQT/FCA and the PCA are necessary before the CIVC contract can transition to the CIVC-A development. Some time needs to be spent in JAN-MAR 1990 to determine what aspects of the CIVC-A project needs to be different from the CIVC project.

Projected Work:

- Status report from SofTech at the March meeting.

Deliverables Due Next Quarter: None

Presentations Planned: None

Other Significant Information: None

Action Items:

- SofTech. Develop a prioritization for the 1838A test selection criteria taking into account the Issues and Strategies document produced by the SEVWG. (By the March meeting)
- CIVCWG. Put comments on the NET concerning the functional requirements and capabilities for an Ada language Hypermedia product to support CIVC-A and related efforts. (By 15 FEB 90)
- Ray Szymanski. Add Kurt Gutzmann and Bruce Taylor to the E&V mailing list.
- CIVCWG. Put comments on the NET concerning ideas and changes desired in the next phase of the CIVC contract. (By the March meeting)

2.3 Classification Working Group (CLASSWG) Status Report

Attendees:

Ronnie Martin, Chair
Captain Rebecca Abraham
Peter Clark
Dr. Bard Crawford
Fred Francl
Major Patricia Lawlis
Pat Maher
Walter Marks

Honorary Members:
Karyl Adams

Deliverables Due This Quarter: None

Accomplishments:

Reference Manual Enhancements - The following action items were completed:

CLASSWG:	Discussed the usefulness of Fig x-1, x-4 in the Reference Manual. Determined that availability should not be an attribute in Chapter 6 of the Reference Manual.

Discussed obtaining comments from the originators of technology.

Guidebook Enhancements - The following action items were completed:

Nelson Weiderman	Responded to a message referencing Attributes addressed by AES and the SEI for the entries in the Guidebook.
Fred Francl	Reviewed the Program Management Toolset Document provided by Sandi Mulholland.
R. Martin/ R. Abraham	Brought e-mail descriptions to the meeting and reviewed them to determine possible enhancements to the e-mail checklist.
Peter Clark	Reviewed the Guidebook Checklists - Do They address Power or Completeness???.
Karyl Adams	Provided a synopsis of the Lyons book for inclusion in the Guidebook.

Other Accomplishments:

CLASSWG	Discussed the possibility of conducting a survey of the Evaluation Service Organizations.
CLASSWG	Discussed the future enhancements for Version 3 of the Reference System.
B. Crawford/ P. Lawlis	Drafted worksheets for use by REQWG when generating appendices to the Tools and aids Document.

Key Issues:

- The CLASSWG needs help from experts in the following categories:
  Configuration Management
  Distributed Systems and Runtime Support
  Distributed APSEs Support

  If you know anything about these topics please provide cross references between the appropriate elements in the APSE Tool Categories Lists (Chapter 5 of the Reference Manual) and the elements in the functional taxonomy (Chapter 7 of the Reference Manual).

- If you are knowledgeable about Runtime Environments, Peter Clark has developed a new checklist that needs review.

- The CLASSWG also needs help with the following topics:
  Program Library Management Capabilities Checklist
  Import/Export Capabilities Checklist
  Real Time Analysis Capabilities Checklist

  If you know anything about these topics, please review the existing checklists in the Guidebook and let CLASSWG hear your comments/suggested enhancements.

For more information, inquire through e-mail.

Projected Work: See Action Items

Deliverables Due Next Quarter: None

Presentations Planned:

- Brian Nejmeh (Peter Clark will arrange)

Other Significant Information: None

Action Items:

- CLASSWG. Read Version 2 of the Reference System and come to the next meeting prepared to discuss suggestions for improvement. (Only applies to those members who have not yet done this.)
- Becky Abraham. Determine how to implement Change Request Form inclusion in Reference System documents (also review form included in SEI documents).
- B. Crawford/P. Clark. Propose a solution to the dreaded Fig. 1-1, x-4 in the Reference Manual.
- Peter Clark. Extract descriptions of the life cycle activities from DOD-STD-2167A for inclusion in the Reference Manual.
- Ronnie Martin. Review the Reference Manual Testing Related Definitions.
- Ronnie Martin. Review Karyl Adams' writeup of the Lyon's book.
- Peter Clark. Review the List of Debugger and Library Management Capabilities Being Evaluated by Boeing.
- Ronnie Martin. Review the Reference Manual pages for input to the Configuration Management Checklist.
- Ronnie Martin. Refine e-mail checklist based on CLASSWG discussions.
- Fred Francl. Refine the proposed Tracking Checklist based on CLASSWG discussions.
- Ronnie Martin. Send SERC Technical Report on Evaluating the User Interface Management Systems to Peter Clark.
- Peter Clark. Prepare the Guidebook entries describing CECOM documents.
- Ronnie Martin. Determine appropriate Guidebook References to Lyon's book, Chapters 8, 9, 15, 16 and 17. Are any other references needed?
- Sandi Mulholland. Continue Transforming customization Issues (Distributed APSEs, Runtime Support) into a Checklist.
- Sandi Mulholland. Further elaborate Whole APSE Assessment Issues/Address Life Cycle Support Issues.
- Becky Abraham. Monitor STARS documents for References/Synopses in the Guidebook.
- CLASSWG. Search for new opportunities to improve the Guidebook ... new checklists (addressing function independent attributes, for instance), and references to E&V Technology ...
- Peter Clark. Make a few phone calls referencing existing Evaluation Services.

2.4 Requirements Working Group (REQWG) Status Report

Attendees:

Major Patricia Lawlis, Chair
Captain Rebecca Abraham
Mike Burlakoff
Peter Clark
Dr. Bard Crawford
Jay Ferguson
Fred Francl
Greg Gicca
Liz Kean
Pat Maher
Walter Marks
Ronnie Martin
Karla Nohalty
Lloyd Stiles
Nelson Weiderman
Betty Wills
Barbara Rhoads, recorder

Deliverables Due This Quarter: None

Accomplishments:

- Progress in producing an ASCII form of the E&V Reference System.
- Update to E&Ving News.
- Better definition of the format to be used for appendices to the Tools and Aids Document.
- Work on the Tools and Aids appendices.
- Work on refining the model project concepts.

Key Issues:

- Exchange of information with the E&V community.
- Get Tools and Aids Document to a state where it will be useful to agencies in a position to fund E&V technology.

Projected Work:

- Continue work on:
  - appendices to the Tools and Aids Document
  - public relations efforts

Deliverables Due Next Quarter: None

Presentations Planned:

- Major Lawlis will demonstrate ASSIST at the next meeting of the IEEE Standards Working Group on CASE tool evaluation.

Other Significant Information: None

Action Items:

Carried Over

- Sandi Mulholland. Life cycle support from whole APSE view.
- Betty Wills/Pat Maher. Verify the procedure that the Ada Information Clearinghouse could use to update the E&V team documents.
- Bard Crawford/Peter Clark. Produce an ASCII form of Version 2.0 of the Reference System.
- Betty Wills/Pat Maher. Deliver ASCII Version 2.0 to the Ada Information Clearinghouse.
- Sandi Mulholland. Update the Requirements Document with a pointer to the Reference Manual attribute definitions; also update date and version on title page and remove the trademark references; update table of contents as necessary.
- Ray Szymanski. Get out questionnaire for feedback on Ada Compiler Evaluation Capability.
- Jay Ferguson/Lloyd Stiles/Sandi Mulholland. Continue work on the Quality Factors paper; tailor this information to fit the assessors and to be included as an appendix to the Tools and Aids Document.
- Greg Gicca/Mike Burlakoff/Tom Leavitt/Fred Francl. Prepare the appendix on Compilation Systems Evaluators.
- Ronnie Martin. Prepare the appendix on Test Systems Assessors.
- Alan Impicciche/Peter Clark. Prepare the appendix on Requirements/ Design Support Evaluators.
- Bard Crawford/Patricia Lawlis. Prepare the appendix on the Whole APSE Assessors.
- Ronnie Martin. Put the "appendectomy" procedures on the NET.

New

- Bard Crawford. Produce Assessor Requirements/Coverage Worksheet and distribute to the team.
- Liz Kean. Get E&Ving News updated and to Ray Szymanski.
- Ray Szymanski. Get E&Ving News to Ada letters prior to 31 DEC 89.
- Patricia Lawlis. Put status report on the NET.

2.5 Standards Evaluation and Validation Working Group (SEVWG) Status Report

Attendees:

Tim Lindquist, Chair
Karyl Adams
Lynn Chilson
Kevin Hackett
Duston Hayward
Kurt Gutzmann
John McBride
Gary McKee
Ray Szymanski

Deliverables Due This Quarter: None

Accomplishments:

- Final review of the Issues and Strategies Document for CAIS-A by SEVWG. SEVWG reviewed changes that Dr. Lindquist had made in response to team comments at the September review. These included comments on the writeup in test selection criteria, CAIS evaluation, general editorial comments and the final section on recommendations.
- A discussion on the other relevant APSE standards rendered an outline for a SEVWG position paper. Covering lessons learned from CIVC/IAS, the paper will review important interface standards in light of the lessons SEVWG has learned regarding tools for generating and maintaining validation test sets.
- Duston Hayward brought SEVWG up to date on the Joint European efforts. It seems that people (U.S. side coordination by Currie Colket at NADC) are planning a combination of PCTE+ and CAIS-A (called PCIS). This appears to be a generalization (in the inheritance sense) of the two interface standards. Planning for such an effort includes a schedule calling for development of a validation suite and existance of production quality implementations by the mid-90s and environments populated with tools by the late 90s.

Key Issues:

- All issues regarding the Issues and Strategies Document have been resolved (or more appropriately elaborated in an acceptable form in the paper).
- It appears that a "join" operation should occur with Peter Clark and company regarding reviews of other standards. This should be elevated to a joint session of SEVWG and CLASSWG in March.
- The work on PCIS mentioned above underscores the importance of viewing CIVC and CIVC-A as a maturing evolving process rather than as a single product that is released and used henceforth. To that end, the Issues and Strategies Document recommends a more tool oriented approach to developing CIVC-A. (By tool oriented SEVWG means tools to aid in the development and maintenance of the validation suite.)

Projected Work:

- Dr. Lindquist will be making a final editing pass through the Issues and Strategies Document for CAIS-A in the near future (before March). SEVWG agreed to generate a draft of the white paper before the June meeting.

Deliverables Due Next Quarter:

- Issues and Strategies Document for CAIS-A. Reviewed changes since last meeting, final editorial changes will be made and copy submitted to Ray Szymanski.

Presentations Planned: None

Other Significant Information: None

Action Items:

- Tim Lindquist. Revise the Issues and Strategies Document, place status report on the NET, send current copies of the Issues and Strategies Document to Kurt Gutzmann and Phil Brashear, coordinate the joint session's agenda with Peter Clark.
- Tim Lindquist. Ask Ray Szymanski for whom the Issues and Strategies Document is being prepared. (This is for the title page).
- Karyl Adams. Place on the NET a listing from the Reference Manual and the Guidebook of checklists that offer a baseline to build up checklists for CAIS-A evaluation.
- Barbara Rhoads. Mail out a draft of the white paper discussion to Tim Lindquist and Gary McKee.

APPENDIX A

10.0 LIST OF ATTENDEES

Captain Rebecca Abraham
WRDC/FIOC
WPAFB, OH 45433-6543

Sam Ashby
Boeing Military Airplane
Company
P.O. Box 7730, MSK80-13
Wichita, KS 67277-7730

Mike Burlakoff
Southwest Missouri
State University
Computer Science Dept.
Springfield, MO 65804

Peter Clark
TASC
55 Walkers Brook Drive
Reading, MA 01867

Dan Eilers
Irvine Compiler Corp.
18021 Sky Park Circle, #L
Irvine, CA 92714

Jack Foidl
TRW, Systems Division
Suite 205
9265 Sky Park Court
San Diego, CA 92123-4213

Greg Gicca
Sanders Associates
NCA1-2232
95 Canal Street
Nashua, NH 03061

Kevin Hackett
SofTech, Inc.
16875 West Bernado Drive
San Diego, CA 92127

Elizabeth Kean
RADC/COEE
Griffiss AFB, NY 13441-5700

Tim Lindquist
Computer Science Dept.
Arizona State University
Tempe, AZ 85287-5406

Karyl Adams
c.j. kemp Systems, Inc.
3318 E. Dry Creek Road
Phoenix, AZ 85044

Phil Brashear
CTA, Inc.
5100 Springfield Pike
Dayton, OH 45431

Lynn Chilson
SofTech, Inc.
1300 Hercules Drive
Suite 105
Houston, TX 77058

Bard Crawford
TASC
55 Walkers Brook Drive
Reading, MA 01867

Jay Ferguson
National Security Agency
ATTN T303, 9800 Savage Road
Fort Meade, MD 20755-6000

Fred Francl
Sonicraft, Inc.
8859 S. Greenwood
Chicago, IL 60619

Kurt Gutzmann
SofTech, Inc.
1300 Hercules Drive
Suite 105
Houston, TX 77058

Duston Hayward

Major Patricia Lawlis
AFIT/ENC
WPAFB, OH 45433

Patrick Maher
P.O. Box 1417, M/D H8175
8201 E. McDowell Road
Scottsdale, AZ 85252

Walter Marks
CECOM-CSE
Fort Monmouth, NJ 07703

John McBride
SofTech, Inc.
1300 Hercules Drive
Suite 105
Houston, TX 77058

Karla Nohalty

Valerie Rodriquez
Oneida Resources, Inc.
3578 Kettering Blvd.
Suite 300
Dayton, OH 45439

Bruce Taylor
Intermetrics, Inc.
733 Concord Ave.
Cambridge, MA 02138

Jerry Thomas

Betty Wills
CCSO/XPTB
Tinker AFB, OK 73145

Ronnie Martin
Software Engineering
Research Center
Dept of Computer Science
Purdue University
West Lafayette, IN 47907-2004

Gary McKee
McKee Consulting
P.O. Box 3009
Littleton, CO 80161-3009

Barbara Rhoads
Oneida Resources, Inc.
3578 Kettering Blvd.
Suite 300
Dayton, OH 45439

Lloyd Stiles
FCDSSA
200 Catalina Blvd
San Diego, CA 92147

Kermit Terrell
Boeing Military Airplane Company
P.O. Box 7730, MSK80-13
Wichita, KS 67277-7730

Nelson Weiderman
Software Engineering Institute
Carnegie-Mellon University
Pittsburg, PA 15213

APPENDIX N

MINUTES

OF THE

EVALUATION AND VALIDATION TEAM

GENERAL SESSION

7 - 10 MARCH 1990

The task for the Evaluation and Validation of Ada Programming Support Environments (APSEs) is sponsored by the Ada Joint Program Office (AJPO).

TABLE OF CONTENTS

Section | Page

1.0 WEDNESDAY, 07 MARCH 90 3
1.1 Chairman's Corner 3
1.2 CAIS Implementation Validation Capability Update 4
1.3 Evaluation and Validation Technical Support Update 8
1.4 CLASSWG/SEVWG Joint Session Status Report 10
1.5 Ada Compiler Evaluation Capability Update 12
1.6 Closing Remarks 16

2.0 WORKING GROUPS STATUS REPORT 16
2.1 Ada Compiler Evaluation Capability Working Group (ACECWG) Status Report 16
2.2 CAIS Implementation Validation Capability Working Group (CIVCWG) Status Report 18
2.3 Classification Working Group (CLASSWG) Status Report . . . 19
2.4 Requirements Working Group (REQWG) Status Report 22
2.5 Standards Evaluation and Validation Working Group (SEVWG) Status Report 23

LIST OF APPENDICES

10.0 LIST OF ATTENDEES 27

1.0 WEDNESDAY, 07 MARCH 90

1.1 Chairman's Corner

Raymond Szymanski
Wright Research and Development Center,
Wright-Patterson Air Force Base

Mr. Szymanski stated that the Evaluation and Validation (E&V) Team in its present form will terminate at the September 1990 meeting. The task itself will continue with those who are contractually obligated to do so until the end of the contract. No decision has been made as to whether or not the quarterly group meetings will continue.

Mr. Szymanski is creating a list of those people that the Department of Defense will honor for their service to the E&V task over the last several years. Current team member were asked to submit the names of those people who have significantly and positively had an impact on the E&V task but are no longer highly active members of the Team but are deserving of recognition.

Dr. Solomond has publicly stated that the ACEC/AES merger has been negotiated, signed, and delivered. Mr. Szymanski has discussed this with Dr. Solomond who indicated that all questions on the subject should be directed to him.

Ada Compiler Evaluation Capability (ACEC) Version 2.0 was recently delivered to the Air Force and is currently in an in-house review. All the documentation is being reviewed, including the Version Description Document, the User's Guide, and the Reader's Guide. Mr. Szymanski is anticipating that the Data and Analysis Center for Software (DACS) will have the suite and the documentation within the next four weeks.

SofTech has finished their CIVC deliverable.

Activities since the last meeting:

- Most of Mr. Szymanski's time has been involved with the following items: ensuring funding for all the contracts; changing the specification on the ACEC contract; and getting a particular individual to the meeting on guest orders.

- With respect to the RFP, Boeing was asked for a proposal to do enhancements to the suite as per recommendations from the ACEC working group. They are currently working on this.

Publicity:

- Mr. Szymanski provided an update to the E&V'ing News.

- Articles were submitted to the following:
  - Ada Information Clearinghouse Newsletter
  - Software Technology Support Center (STSC) Newsletter
  - Language Control Facility Newsletter

- Mr. Szymanski gave an AdaJUG presentation covering an update on the E&V task effort. The basic message was the lack of funding from AJPO next year.

- Mr. Szymanski participated on a panel with Dr. Solomond, as well as the Ada representative from the Navy, Marshall Potter, and other Ada executives from the Air Force and Army.

- Mr. Szymanski briefed SENTAR, an organization within the Aeronautical Systems Division. They asked questions pertaining to technology regarding due date, its exploitation, and interested programs. SENTAR is to review the technology at certain points in the program. Because the focal point and the panel has voted to support the technology within WRDC and the ASD community, they will be doing some of the broadcasting for the E&V effort. Mr. Szymanski provided them with pamphlets which they redirected to some of the project offices. As a result, Mr. Szymanski was asked to provide a write up on the ACEC as that technology seems to have made the broadest impact.

- Mr. Szymanski will brief the STSC spring conference on the ACEC evaluation. He will also be giving a presentation at the 1990 Avionics Symposium in Las Vegas, Nevada. This presentation will be concern the E&V task. One of Mr. Szymanski's final messages to that audience will be that this activity has made some in-roads, but it has a long way to go before completion; it will be in the Department of Defense's (DoD's) and the Air Force's best interest to continue this type of work.

The next meeting of the E&V Team will be in Pittsburgh, Pennsylvania beginning after the Memorial Day holiday. Mr. Weiderman stated that the meeting will be held at the SEI.

Mr. Szymanski commented that he would like the working groups, particularly the Standards Evaluation and Validation Working Group (SEVWG) and the Requirements Working Group (REQWG) as they are due to disband first, to begin seriously working on their portion of the final report according to the format, contents, and style of the KAPSE Interface Team (KIT) report.

Mr. Crawford requested that everyone begin to think about their recommendations as the working group chairs of the different committees need input from everyone. Mr. Szymanski stated that this is a document that will definitely be used after the Team disbands.

1.2 CAIS Implementation Validation Capability Update

Lynn Chilson
Kurt Gutzmann
SofTech

Mr. Chilson stated that the Functional Configuration Audit (FCA) and Physical Configuration Audit (PCA) have been completed and CIVC-1 was delivered to

Ray Szymanski. The CIVC-1 activities are almost closed out. The Phase 1 report is currently in progress.

Some important activities that are currently being conducted are porting the applicable test cases out of the CIVC-1 test suite to CAIS-A. SofTech will be delivering a beta test release of the port in July of this year.

The reason for the July release is to fold some CIVC validation capability into the CAIS-A work that is being preformed. The contract was more or less overcome by events as the deliverables and the end of the contract is March 20, 1992 and a lot of the early CAIS-A implementation work is going to be completed and in the hands of users by that time. SofTech cannot go back and do the contract and the work in zero time. The first solution was to take the CAIS work that was done and upgrade that where possible to CAIS-A and release that as a beta test suite. To support the new efforts in CAIS-A, SofTech had to make some hardware and software upgrades to the JMS5.2 and DECAda 2.0.

The kick-off meeting for the CIVC-2 Phase III, CAIS-A validation, was held on the 7th and 8th of February 1990 in San Diego, California. At that time, Duston Hayward and Currie Colket spoke about the CAIS-A/PCTE (Portable Common Tool Environment) merger and the Portable Common Interface Set (PCIS).

Kurt Gutzmann's presentation to the Team was provided by Currie Colket at the Ada Joint User Group (AdaJUG). The Team will be given an overview of the PCIS merger activity or what is forecast at this time. Currie Colket is currently in charge of the North Atlantic Treaty Organization (NATO) effort to merge the CAIS-A and PCTE. Duston Hayward is program manager in charge of the implementation of CAIS.

Mr. Chilson then turned the floor over to Mr. Gutzmann. Mr. Gutzmann stated that his material was provided by Currie Colket, Duston Hayward, and John Solomond at the AdaJUG.

This presentation focused on the activity with the interface sets and their merger. The outline of that briefing included the importance of Interface Technology, history of CAIS-A/PCTE+ convergence effort, CAIS-A, PCTE+, highlights of convergence solutions, PCIS program (preliminary), available documentation, and a summary. Mr. Gutzmann focused on solutions and the PCIS program.

In interface technology, the DoD tends to drive Ada and CAIS and standards activities, but the PCTE is working more out of the European commercial interest. This technology will provide an increased commercial interest in developing wider-based and better quality integrated tools, and tool, data, and personnel portability with improved project inter/intra operability and a shorter learning time.

CAIS-A is sponsored by the Ada Joint Program Office (AJPO). The goals listed include:

- Ada Programming Support Environment (APSE) Interface
- Based on RAC requirements
- Independence from existing proprietary systems
- Upwards compatible with CAIS-1
- 90%/10% goal

The idea is to obtain tools independent from vendors. Vendor hardware and operating system independence would be something to obtain through a standard interface set and that is a key driver for a lot of customers. Concerning upward compatibility, SofTech is currently investigating porting the CIVC test suite from CAIS to CAIS-A.

Mr. Hackett interjected that upward compatibility was one of the major design requirements for CAIS-A.

Mr. Gutzmann stated that the PCTE+ is a European sponsored program and he did not know how they are set up to do this work. Their APSEs (Ada Programming Support Environments) are called IPSEs (Integrated Project Support Environments). They have a comparable set of goals which include:

- IPSE interface
- Industry input and acceptance
- Based on EURAC requirements
- Independence from proprietary systems
- Upward compatible with PCTE
- Complete interface goal

Aside from a few differences, it is basically the same notion of layering the operating system with the interface set and have the tools using it. It's the same idea of conversions to common concepts.

Two different meetings and workshops were conducted in Laughlin and Winnersh. The experts from both sides discussed the similarities and differences between the CAIS-A and PCTE+ and found a lot of common ground.

The differences in CAIS-A and PCTE language binding include names, parameter order and number, and exception reporting mechanisms. The proposed solution is to allow incompatibilities in details. The concepts are the same. It is not certain what PCIS would be if it had a brand new interface set.

The data model is basically the same. There are terminology differences but the concepts are the same.

In typing and views, the semantics of deletion and relationships are different between these two interface sets. CAIS-A has stronger typing than PCTE+ and the visibility control was different between these two. The proposed solution is to use the PCTE+ deletion and relationship semantics, use CAIS-A strong typing, PCTE's view-oriented visibility control; the best of both. The syntax for pathnames is different; the solution is to support both forms.

The differences in entity monitoring include CAIS-A's use of a single event queue, PCTE+'s support of multiple queues and of event selection based on event type. PCTE+'s more comprehensive mechanism for entity monitoring will be used. There are no major incompatibilities in the area of processes.

For execution monitoring, CAIS-A has no direct support while PCTE+ has very detailed support. PCTE+ is more tightly bound to tools. It has an execution monitoring function where a program can be monitored as it runs. CAIS-A does not explicitly have that capability, but uses some interprocess communication facilities to achieve this. They would like to reduce the dependency of PCTE+ or PCIS on elements like the compiler so execution monitoring can be performed without needing a particular compiler.

Both interface sets and provides access control functions. According to the Winnersh report, PCTE+ had better support for access control, security and integrity functions. The proposed solution is to adopt a more comprehensive model.

Both distribution models have a system model concept. CAIS-A provides the user with some idea of what system architecture is included. These interfaces recognize distribution explicitly. The user selects the particular processor and launches the process. The idea is to combine these. CAIS-A does not provide as much guidance on how to distribute processes in the system.

Mr. Hackett commented that the CAIS implementation does not have to support multiple processors or a robust resource model but is allowed to. Possibly PCTE mandates certain kinds of distribution. That may be the difference.

The resource model, which is called a system model for CAIS-A, contains the other elements of resources available in this system: printers, other types of devices, processors. They both support that sort of thing. They wanted to get a unified representation for it.

The common external form in CA'S-A allows the user to export an environment to some other system. The user can pick up a project from one contractor and move it to another. If the contractor finishes and someone else wanted to use that work, it makes it fairly easy to pick up the actual work products rather than just the papers. It is a very important feature. PCTE+ does not have a mechanism for exporting environments. The solution is a user interface. PCTE+ describes one and have called for manipulating bit-mapped devices. CAIS-A just has the simple I/O packages.

The Winnersh working group thought it was a feasible idea to go ahead and start working on a convergence of these interface sets. The U.S. is also interested in expanding its own economic opportunities.

The idea of the APSL was originally sponsored by the DoD to support them in their development of software for defense purposes. They find that a lot of those ideas are useful in commercial business now. A lot of large companies are using Ada; therefore, that technology is moving out to the commercial sector.

In the AJPO/IEPG TA-13 meeting, they reviewed/released the Winnersh Report; agreed in principle to pursue the program; and identified the following five principles to govern its conduct:

- Spans both defense and commercial applications,
- Retains maximum possible investment from CAIS and PCTE,
- Timed to derive maximum benefits from PCTE and CAIS,
- Provides natural evolution of the PCTE and CAIS standards, and
- Ultimately be submitted for ISO standardization.

The goal is to have a new interface specification available in mid-1994.

Shortly, they will start working on the definition and the specification effort. A demo implementation will be done about the same time. If they can reuse a lot of existing elements from PCTE+ and CAIS-A, a lot of time will be saved.

Available documentation includes the Waltham Report, NATO Requirements and Design Criteria, the rationale for the NRAC, and the Winnersh Report. All documentation can be provided by Currie Colket.

The AJPO and the IEPG TA-13 which is a NATO group have agreed to develop this new interface standard with Dr. Solomond supporting it.

The NASA software support environment contractors have CAIS-A and are working it into that effort right now. Several third party people like Charles McKay and the NASA people are also urging CAIS-A for the software support environment for the space station.

Mr. Burlakoff asked if anyone had any ideas on how many person years have been invested in CAIS-A and PCTE. Mr. Gutzmann replied that it added up to about 20 man years for the CAIS-A implementation at SofTech San Diego. Mr. Burlakoff asked if PCTE was implemented. Mr. Gutzmann stated that they are implemented and running.

Dr. Lindquist commented that at the time 1838 was being finished, PCTE was already enjoying implementation. They actually announced an Ada compiler that had been hosted on top of PCTE. He said they are quite a ways from seeing a CAIS-A implementation that has an Ada compiler that is integrated with that information.

Mr. Gutzmann concluded saying that there will probably be a validation capability required for PCIS at some point in the future. It is an opportunity for the E&V Team to bring its experience and knowledge to that effort.

1.3 Evaluation and Validation Technical Support Update

Bard Crawford
The Analytic Sciences Corporation (TASC)

Dr. Crawford's presentation focused on a standard summary of the Reference System status. He stated that in the absence of Peter Clark, Dr. Tim Lindquist would be giving the summary of the joint meeting between SEVWG and CLASSWG.

The E&V Reference System project began in the mid-1985. The first six or seven months of the project were dedicated to producing a draft version of the three documents: Classification Schema, Reference Manual, and the Guidebook. Draft versions were delivered to Mr. Szymanski but none of them were sent out to the general public.

In early 1987 a consensus was reached on the E&V Reference System structure and how to present it to the public in such a way that the documents can stand-alone and be easily used.

The one official version of the Classification Schema was based on that consensus. It was finished as a final Classification Schema report and used as a baseline for the Reference Manual. The Reference Manual became a stand-alone document. The Reference Manual has been updated regularly and the Schema has changed in minor respects since the last version.

Version 1.0 of the Reference Manual was delivered and publicly released. At the same time, there was a major revision of the Guidebook and so Version 1.0 was never delivered. Version 1.1 of the Guidebook was issued in mid-1988. Version 1.1 of the Reference Manual was simply updated to make it consistent on all the reference work. Version 2.0 came out last fall, and Version 3.0 will be delivered this fall. There will be a Version 4.0 which probably will not be a major upgrade.

WRDC approved both documents of Version 2.0 simultaneously. Approximately 150 copies have been mailed out. From this point on, the documents can be ordered directly through Defense Technical Information Center (DTIC), but the number of copies that have been sent out will not be available until DTIC provides that feedback.

The electronic version of both documents were sent to the Ada Information Clearinghouse. The Guidebook is available; a list of chapters is provided and the user can download it chapter by chapter. Although it is stated that the Reference Manual is there and a listing of chapters is shown, the information downloaded will be that of the Guidebook and not the Reference Manual. They have the entire ASCII version of both documents equivalent to the written documents except for some of the figures in the Reference Manual.

Contact has been made with the Ada Europe Working Group on environments relative to their June meeting. It is thought that a status update will be asked for of the E&V activities which will probably focus on the Reference System.

On the model project status, a fairly extensive project report was given to the REQWG by Karla Nohalty. At the end of that meeting, it was indicated that she would return to report on the completed scenario number 1. That work has been delayed for two reasons: 1) funding, and 2) she has been busy on other projects. The first thing she had to do was convert the inertial navigation program from a VAX/VMS program. She had to make some changes and they were fairly sensitive to get it all working under VAX/VMS. This has been done. The inertial navigation system and EC are two separate processes. One terminal is in Karla's office and one terminal is in the next office. It can probably be done with a different kind of terminal with two windows so the program is seen running properly. That does not mean that every detail is checked out and tested, but the appropriate screens come up. It is a fairly sizable program. The inertial navigation portion is two-million lines of code, and EC has 9,000 lines. It is being used as a test bed for whole-APSE and other kinds of evaluations. The concentration is on test analysis activities as well as whole-APSE activities and scenarios. There has not been much work done. Karla has done some work with diagrams that correspond to the program. She is expected at attend the May meeting in Pittsburgh.

Version 3.0 will include a comment form which means the change request form, a Government-type form, will be the first page of each document. There will be a continual reaction to any public feedback as from the survey previously conducted. Inputs are continually sought for distributed APSE evaluation and distributed system support evaluation. There is not much there now and they are important.

Work is being done on CASE tool evaluation and on the whole APSE area. Better understanding is needed in these areas for reflection in the Reference System.

1.4 CLASSWG/SEVWG Joint Session Status Report

Tim Lindquist
Arizona State University

The Standards Evaluation and Validation Working Group (SEVWG) and the Classification Working Group (CLASSWG) met jointly to discuss whole-APSE evaluation and its relationship to standards. The meeting began with a presentation by Brian Nejmeh which focused on integratable tools. The three approaches to integratable tools are (1) strategic alliances between tool vendors; (2) standards; and (3) open tool interfaces which was the focus of his presentation. He defined 13 characteristics of off-the-shelf tools that actually promote integration among tools and by means of open tool interfaces.

He had them ordered in a scale of simplest to most difficult, on a scale of 1 to 13, the ideas become more complex and the implementation more difficult.

The first characteristic is location independence. The tools did not assume a specific location in an environment, and did not depend on pathnames. Another characteristic was noninterference; the execution of a tool did not affect performances of other tools. Data import and data export alluded to the ability for tools to essentially produce and use output from other tools.

The next three related characteristics (schema extension, conceptual information exportation, and operation extension) are fairly complex. These characteristics allow the tool to adapt to the specific needs of the environment. For example, in the Macintosh environment, these characteristics provide the ability to cut and paste information and to modify the types or schematics that relate what information has been cut.

X-windows was discussed as one of the 13 characteristics. This characteristic concerns adherence to standards. Adopting standards that are common, the end result will be tools that are more highly integratable. Also mentioned were hardware transparency and reasonable licensing scheme. Those are the issues that actually inhibit the ability to integrate tools and to use tools from different hosts.

This sets the stage for whole-APSE evaluation as those characteristics are not just of a single tool, but several tools used in conjunction to compose an environment.

Two different areas were produced from his discussion; APSE model presentations and APSE-related standards. For APSE model presentations, Peter Clark tried to fit existing standards into a model for an APSE to determine where they existed or where they could be appropriately placed.

Mr. Clark presented that model to the group which immediately started considering other models and other views. For example, Sandi Mulholland provided a run-down of the National Institute of Standards Technology (NIST) model for an APSE. Activities then ranged between the importance of having a model and its relevance to evaluation or assessment of whole-APSE characteristics and what that model ought to be.

Standards were also discussed from three different perspectives: the standards that exist, those being created, and those that are needed. The group tried to see how they fit into different models.

Peter generated as a CLASSWG activity a listing of APSE-related standards and passed out copies. The group discussed the list at length.

Certain relationships exist among the three perspectives of whole-APSE assessment, APSE models, and APSE standards. Some time was spent talking about the relationships and the needs and the drawbacks, for example, of adopting a single model and basing assessment on a model as opposed to other paradigms for doing that.

The group believes that whole-APSE evaluation or assessment is an important aspect of an APSE evaluation and validation. Different components of the group believe that certain products and services are based upon this idea of an APSE model, a TASC view of assessment, have to exist in an environment.

SEVWG is working on a white paper that surveys applicable standards and indicates the importance of examining the evaluation and validation of those standards and trying to promote the idea that a standard that is applied to an APSE implementation is not really worthwhile without some form of validation, if not evaluation.

The group discussed new homes for E&V. The group believes that the September deadline imposes a barrier that may inhibit the continuance of these activities. Software Technology Support Center (STSC) was suggested as a home for E&V.

Dr. Lindquist concluded his presentation with questions from the Team.

1.5 Ada Compiler Evaluation Capability Update

Tom Leavitt
Boeing

Mr. Leavitt briefed the Team on the status of the ACEC contract, a review of the Formal Qualification Testing (FQT) activities, lessons learned, and the candidates considered for the next release.

The second release has been delivered and includes the Software Product, Version Description Document, User's Guide, Reader's Guide, and Software Test Report. Work is being done on scheduling the delivery of the technical report; Software Product Specification is pending. ECP for the third release has been received; some work has been started on that.

The FQT was performed on five systems: DEC Ada and TeleSoft Ada which were both VAX/VMS self-targeted; ALSYS Ada which is an Apollo Aegis self-targeted; the MIPS processor which is a Silicon Graphics compiler; and a 1750A cross compiler which is VAX/VMS hosted and targeted to Sperry 1631.

The operation of the tools of the 1750A compiler was not appropriate, and therefore was not done. The operation for those tools were done on the host for that system. Three tools were not run on the Apollo. The reasons, associated with timing, were that availability to the borrowed system was lost, equipment was returned, the person doing the testing on the Apollo was transferred, and funds were exhausted.

An error prevented the SSA and the Silicon Graphics from running. Some problems on that compilation system under this tool and some other places in the testing indicate that the process of exception handling is not completely reliable. The exception of the problem in the SSA and Silicon Graphics was that in the process of running the performance test the program was not modified. Occasionally, some errors were found in the system. Concerning tools, there was some adaptation required, particularly for the math packages similar to the testing in the first release.

In the second release products, it was found that a representation was developed in an independent version of math dependent which will extract the numbers without relying on the bits. That was run without modification. All but one system, the 1750, tested on it and that system caused the compiler to crash with an internal error.

The results from the MEDIAN factors test were displayed on a viewgraph. Although the DEC compiler and TeleSoft were VAX targeted, they were run on different processes, on different machine models. So the actual machine factors included both the difference in underlying hardware and software systems.

Some analysis was done of the actual test measurements. It was found that 85 problems had zero or near-zero times. Optimization was expected in 27 of those. Another set of problems actually only occurred on the Silicon Graphics which reported zero execution times and had non-zero code sizes and was anomalous. This was investigated; however, some things that were not entirely repeatable.

Also investigated were problems flagged as outliers by MEDIAN. The majority of those problems are I/O test problems. Large differences in those systems were anticipated.

During testing, 60 software problem reports were written on the ACEC software. Before testing was completed, all but one of those were fixed and that one was a suggested enhancement to rename the scan problems which was deferred since everythng in Version 3.0 will be renamed. The test report was written on the data from the first FQT run-through.

In a review of the type and number of problems, a number of tests were associated in a bad format meaning that some automating tools were used to do preprinting, reformatting, spelling check, and moving the comment blocks. In some cases, those tools lost units, and introduced errors. To correct the errors, a set of problems was found that was susceptible to loop invariant optimizations and these were modified. Some errors were found in that particular program by running it. A system was finally found that was reliable and produced some results.

Two basic problems of the AIFRAME program were corrected. One was allocating structures with access types inside a record; when it was releasing the whole record, it did not release the space for the subordinate structure which would cause space to exhaust any available memory. Another problem was that a couple of routines were deallocating a pointer after it was already deallocated. AIFRAME successfully ran on four different systems, including the DEC Ada and TeleSoft Ada on the VAX, and Silicon Graphics system. They all worked on those systems without exhausting heaps or crashing.

Looking through the run-time errors that have been reported on the different systems, there were a couple of patterns of interest. There was some pattern to problem language features or areas that systems are failing on in more than a random fashion. There was a set of problems associated with aliasing. These new problems in the second release caused several failures on two different systems.

The implicit storage reclamation tests which contain structures which implicitly allocated space was done repetitively to see if it exhausts memory. When these were run, it was found that a couple of systems failed. There might be slightly more failures on the one system because in the process of running it was found that it would easily crash and destroy data, directory structures, corrupting systems. There was a reluctance to try any more tests on a system where it might cause sever problems. So there might be a slightly higher failure count than that.

Several systems were found that had some problems with the bit manipulation problems associated with packed Boolean arrays operation. There was one pattern associated with allocation and deallocation of access types that failed on one system. There were several examples of problems that exercised allocation and deallocations failing on that system. Preemptive priority scheduling did not work at all on two systems.

The findings in the second release show that for the single system analysis tool the output was larger than hoped; it automatically compares results of related tests.

There was some modification to the timing loop code in the second release which moved a lot of the code from STOPTIMEO into a procedure call. One of the effects saves about 95 lines per test problem. The total code passing through the compiler after being included for the second release is about the same as it was in the first release, even though 380 more problems were added. The time display was changed to put out an exponential format. When having a small or large number, this is helpful because there are cases, particularly on the faster system, when results need more precision than tenths of microseconds. A change of one digit in the least significant place is much more of a fractional change than measurements that are added. A minor change was made to the timing code to check for small values and small timing measurements. If the code expansion size is zero and measurements show that the code expansion measurement times are working and a small timing value is reported, it will force that to zero or indicate an unreliable error message if it is too large. A test was put in that reports that there are some operating systems which always returned a unique value to the time which is not related to the actual clock. Essentially, a ten millisecond clock was used and the value returned is the clock time plus the number of calls made on the timer routine since the last clock update. The Apollo system actually does that.

The MED_DATA_CONSTRUCTOR program was one of the tools developed for the second release. It was found to be helpful in eliminating or simplifying a lot of tasks associated with preparing reports and output from test problems and getting to the analysis programs. It does a lot of checking for duplicate values and missing values and actual building aggregates or taking the aggregates and building programs.

Another change made was to the MATH library; it is compatible with a subset of the NUMWG implementation. This has been substantiated on the vendor libraries. The rest of the ACEC test suite analysis tools were run on that and did not uncover any problems. There is a MATH dependent version which does not require users to do any representation clause, bit manipulation, or system dependent code adapted to extract the exponent field of a floating point number. That version has been checked out and multiple implementations are working. The MATHTEST program was run. One of the modifications made to that in the NUMWG recommendations was to have it put out some suggested accuracy tolerances for the actual functions. Those tests were incorporated for those limits in MATHTEST programs so that it observes errors larger than the recommended error bounds. In running on vendor libraries, error bounds were found on functions, which was not surprising. One case was losing about half the significant bits.

One of the major activities new to the second release was the assessors which provide some structured methods for evaluating libraries, debuggers, and diagnostic systems. There is a large amount of effort required in running the assessors.

This is an inherent problem in the sense of asking someone to assess a debugger because that system's debugger will have to be learned which takes a large amount of effort to do if the tester has not used the system before. The background of the testers is important in how long it actually takes to run the test through. There is no portable standard for how a debugger or library system should work, so there is going to be user adaptation to take the scripts and adapt them to whatever implementation systems are provided on the different systems. This is not as much a problem in the diagnostics in the sense that operationally, the diagnostic has simply compiled some source and looked at the results. Unfortunately, it means that the user has to go through and look at the results.

In major findings on the different assessors, it was disappointing that most of the libraries seem to be more fragile than would be desirable for production use. They crashed a lot. The different systems have taken different approaches on how to provide debuggers, particularly the diagnostics. There will be some judgment required of the assessors in actually interpreting what a system response is going to mean. At least one case where a particular type of error response was anticipated, one system came up with something different, but reasonable. In that particular case, the problem was revised to make it less ambiguous, an error condition to be diagnosed, but the general problem is unavoidable.

The following list was taken from the requested-for-change list Boeing received from the Air Force.

1. Capacity tests
2. Systematic compile speed
3. Runtime memory size
4. Additional performance tests
5. Reliability tests
6. Weighting
7. User interface
8. Documentation
9. Automation
10. Organization
11. Names
12. Packaging
13. Grouping
14. Indexes

A response is being prepared to propose a new release to incorporate these items which will be covered in depth in the ACECWG.

Mr. Leavitt closed with a short session of questions from the Team.

1.6 Closing Remarks

Raymond Szymanski
WRDC

As the ATF has come forward with some funding, they asked Mr. Szymanski to meet with the people working their compiler efforts. The purpose was to find out if there is anything that the Joint Integrated Avionics Working Group (JIAWG) want the E&V Team to work on in the ACEC and enhancement. Mike Mills attended the JIAWG meeting for Mr. Szymanski and is working on this task. He has a three page report for TI along with some other information from McDonnell Douglas.

Mr. Szymanski was also asked to investigate whether or not the Ada9x prototype implementations would be available in time for inclusion by the ACEC. According to Chris Anderson, it will not be available. In light of the numerous activities that the E&V Team wants to accomplish, Mr. Szymanski decided not to spend time on this when it may never make it into the language. The ATF people were very much interested in having this done as they had provided many of the recommendations. The JIAWG put in many of the Ada9x recommendations. Mr. Szymanski feels that they are satisfied with the explanation that it is not possible due to the schedule.

Mr. Szymanski then thanked all those involved in the meeting and asked if there were any questions. He said that if possible he would try to minimize the reports from the contractors at the June meeting which would give the Team an opportunity to have one of the more distinguished SEI people to speak. Mr. Szymanski then adjourned the March General Session.

2.0 WORKING GROUPS STATUS REPORT

2.1 Ada Compiler Evaluation Capability Working Group (ACECWG) Status Report

Attendees:

Nelson Weiderman, Chair
Sam Ashby
Phil Brashear
Mike Burlakoff
Dan Eilers
Greg Gicca
Tom Leavitt
Gary McKee
Sandi Mulholland
Lloyd Stiles
Ray Szymanski
Kermit Terrell
Berry Wills
Valerie Rodriguez, Court Reporter

Visitors:

Stephanie Hellen

Deliverables Due This Quarter: None

Accomplishments:

- Reviewed the minutes of the previous session. Several working group members redlined copies of the draft and gave them to Oneida for final editing.

- Reviewed the ACEC contract status. Release 2 of the ACEC has been delivered to the Government under Boeing risk money.

- Reviewed the results of FQT, at which time five compilers were evaluated using the ACEC. These included compilers written by Digital, TeleSoft, ALSYS, Verdix, and Intermetrics, and included four self-hosted compilers and a cross compiler.

- Reviewed feedback on the ACEC from FCDSSA as presented by Lloyd Stiles and as communicated by Mike Mills from Texas Instruments and McDonnell Douglas.

- Spent time reviewing and prioritizing the suggestions for changes and additions proposed for Version 3.0 of the ACEC. These were boiled down into nine major categories. The consensus ranking of the nine categories was as follows:

  1. Organizational Issues
  2. Interface/Automation
  3. Documentation
  4. Analysis Issues
  5. Capacity Tests
  6. Runtime Memory Size
  7. Systematic Compile Speed
  8. Additional Performance Tests
  9. Reliability

- Edited the Information Sheet on ACEC to be distributed with Version 2.0 of the ACEC.

- Held a short discussion on the ACECWG final report.

Key Issues:

- Further advice needed by the Government on ACEC Version 3.0 and 4.0.

- Funding for Versions 3.0 and 4.0 of the ACEC.

- Impact on the ACEC of QTS (not discussed this time).

- ACECWG final report and recommendations.

Projected Work:

- Review of high level design work on Version 3.0 by Boeing.

- ACECWG final report.

Deliverables Due Next Quarter: None

Presentations Planned: None

Other Significant Information: None

Action Items:

- Nelson Weiderman. Put status report on the NET.

- Liz Kean. Investigate availability of Ada Test and Verification System (ATVS) and program generator for capacity testing (held over).

- Ray Szymanski. Review and send out questionnaire (held over).

- Nelson Weiderman. Ask for input on the new name for MEDIAN.

- Greg Gicca. Determine whether there is a capacity testing program generator of the Ada Software Repository.

- Dan Eilers. Contact Deer Island about availability of benchmarks.

- Ray Szymanski. See whether FQT results can be sanitized for release to researchers so that important information can be disseminated in generic form.

2.2 CAIS Implementation Validation Capability Working Group (CIVCWG) Status Report

Attendees:

Gary McKee, Chair
Phil Brashear
Lynn Chilson
Jack Foidl
Kurt Gutzmann
Kevin Hackett
Tim Lindquist
Ray Szymanski
Jerry Thomas
Kathy Kirkbride, Recorder

Deliverables Due This Quarter: None

Accomplishments:

- CIVC contract status review
- Discussion on Framework/Hypertext options
- Discussion on test selection criteria
- CIVC Change Control Board (CCB) meeting
- Presentation of the RBS tool for coverage analysis and selection criteria

Key Issues: None

Projected Work:

- CIVC TIM scheduled for April in Dayton with Ray.
- Acquisition of TRW CAIS and SofTech CAIS-A implementations.

Deliverables Due Next Quarter: None

Presentations Planned: None

Other Significant Information: None

Action Items: None

2.3 Classification Working Group (CLASSWG) Status Report

Attendees:

Ronnie Martin, Chair
Peter Clark
Bard Crawford
Fred Francl
Major Patricia Lawlis

Honorary Members:

Sandi Mulholland

Deliverables Due This Quarter: None

Accomplishments:

Reference Manual Enhancements - The following action items were completed:

- Bard Crawford. Proposed a solution to the dreaded Figure x-1, x>=4 in the Reference Manual.

- Peter Clark. Entity-relationship diagrams will appear in Version 3.0 draft.

- Peter Clark. Extracted descriptions of life cycle activities from DOD-STD-2167A for inclusion in the Reference Manual - will appear in Version 3.0 draft.

Guidebook Enhancements - The following action items were completed:

- Ronnie Martin. Reviewed Karyl Adams write-up of the Lyon's book - recommended the merging of Karyl's write-up and existing description.

- Fred Francl. Refined proposed Tracking Checklist based on CLASSWG discussions.

Other Accomplishments:

- Becky Abraham. Determined how to implement Change Request Form inclusion in Reference System documents.

- Ronnie Martin. Sent SERC Technical Report on evaluating User Interface Management Systems to Peter Clark.

- Peter Clark. Made a few phone calls referencing existing Evaluation Services.

- CLASSWG. Met with the Standards Evaluation and Validation Working Group (SEVWG) to discuss standards issues. Peter Clark arranged for special guest Brian Nejmeh to address the group.

Key Issues: None

Projected Work: See Action Items.

Deliverables Due Next Quarter: None

Presentations Planned: None

Other Significant Information: None

Action Items:

- Pat Lawlis. Design Change Request Form and place on the NET for comment, forward to Ray Szymanski for action.

  Ray Szymanski. Get Design Change Request Form approved per Becky Abraham's information for inclusion in Version 3.0 of the Reference System.

- Sandi Mulholland. Upgrade functions lists for early life cycle activities in the Reference Manual, Chapter 4.

- Ronnie Martin. Review Reference Manual Testing - Related Definitions.

- Peter Clark. Review List of Debugger and Library Management Capabilities Being Evaluated by Boeing - talk to Ray Szymanski or Boeing to determine status of lists ... are they part of a deliverable that can be referenced? If not, decide what to do.

- Ronnie Martin. Contact Tim Lindquist and Gary McKee in reference to adding entries to the Guidebook.

- Ronnie Martin. Contact Tim Lindquist and Gary McKee referencing adding entries to the Guidebook describing Tim's CAIS Operational Definition and the MITRE Tests.

- Ronnie Martin. Review RCS man pages and TASC checklist for input to the Configuration Management checklist.

- Ronnie Martin. Refine E-mail checklist based on CLASSWG discussions.

- Peter Clark. Review SERC Technical Report on evaluating User Interface Management Systems.

- Peter Clark. Prepare the Guidebook entries describing CECOM documents.

- Ronnie Martin. Determine appropriate Guidebook References to Lyon's book, chapters 8, 9, 15, 16, and 17 ... any other needed references?

- Sandi Mulholland. Continue transforming distributed APSEs Issues (Configuration Management, Distributed Databases, Shared Common Memory, Maintenance Support) into a checklist.

- Peter Clark. Split Sandi Mulholland's inputs into appropriate separate checklists.

- Fred Francl. Prepare Distributed APSEs Security Issues Checklist.

- Peter Clark. Put definition of vendor support on the NET along with the first cut at relevant factors for group discussion.

- Peter Clark. Conduct NET survey of existing Evaluation Services.

- Bard Crawford. Talk to DEC User of Reference System about how it has been used and any comments or suggestions for improvement.

- Becky Abraham. Monitor STARS documents for references/synopses in the Guidebook.

- CLASSWG. Search for new opportunities to improve the Guidebook ... new checklists (e.g., addressing function-independent attributes), and references to E&V Technology ...

2.4 Requirements Working Group (REQWG) Status Report

Attendees:

Major Patricia Lawlis, Chair
Sam Ashby
Mike Burlakoff
Peter Clark
Bard Crawford
Fred Francl
Greg Gicca
Stephanie Hellen
Tom Leavitt
Ronnie Martin
Sandi Mulholland
Lloyd Stiles
Betty Wills

Deliverables Due This Quarter: None

Accomplishments:

- ASCII form of the E&V Reference System now on AdaIC system
- E&Ving News sent to Ada Letters.
- Work on Tools & Aids appendices.
- Work on E&V Final Report.

Key Issues:

- Completing the Tools & Aids Document.
- Developing a legacy.

Projected Work:

- Tools & Aids appendices.
- Update to the Requirements Document to maintain consistency of definitions.
- Inputs to the Final Report.

Deliverables Due Next Quarter: None

Presentations Planned: None

Other Significant Information: None

Action Items:

Carried Over

- Sandi Mulholland/Lloyd Stiles/Jay Ferguson. Continue work on the Quality Factors paper.
- Greg Gicca/Mike Burlakoff/Tom Leavitt/Fred Francl. Prepare the appendix on Compilation Systems Evaluators.
- Ronnie Martin. Prepare the appendix on Test Systems Assessors.
- Alan Impicciche/Peter Clark. Prepare the appendix on Requirements/Design Support Evaluators.
- Bard Crawford/Pat Lawlis. Prepare the appendix on the Whole-APSE Assessors.

New

- Sandi Mulholland. Send Betty Wills the location of the Requirements Document.
- Betty Wills. Update the Requirements Document with a pointer to the Reference Manual attribute definitions; also update date and version on the title page and remove the trademark references; update table of contents as necessary.
- Pat Lawlis. Put draft of REQWG part of the Final Report on the NET.
- Pat Lawlis. Put the status report on the NET.
- Ronnie Martin. Develop the part of the Final Report which deals with the Requirements Document.
- Pat Lawlis. Put generic outline of a working group section in the Final Report on the NET.

2.5 Standards Evaluation and Validation Working Group (SEVWG) Status Report

Attendees:

Tim Lindquist, Chair
Lynn Chilson
Jack Foidl
Kurt Gutzmann
Kevin Hackett
Gary McKee
Ray Szymanski
Kathy Kirkbride, Recorder

Deliverables Due This Quarter: None

Accomplishments:

- Rundown of CIVC-A kick-off meeting held in San Diego February 1990.

- Discussed CIVC-A test selection process, taxonomy, and framework.

- Discussed SEVWG white paper contents.

- Held a joint meeting with CLASSWG regarding the impact of standards assessment on whole-APSE evaluation.

SEVWG met jointly with CLASSWG to discuss the relationships between Whole-APSE Assessment and Standards Assessment. The activities included:

- Presentation by Brian Nejmeh on Integratable Tools and Their Characteristics.

- APSE architectural model presentations and discussions.

- APSE-related standards. Handout by Peter identifying many standards.

- Discussed relationships between a single model, standards, and whole-APSE assessment.

- At meeting close, there seemed to be a movement toward a task or process view of an APSE in place of a model. This view was augmented with the need to identify certain services that are present in the APSE (presumably responding to certain standards). There was no discussion on whether the two working groups felt that this was worthy of jointly developing (any comments via NET are welcome).

Key Issues:

- PCIS (Portable Common Interface Set) which is being constructed to merge CAIS-A and PCTE+ was briefed for CIVCWG at the February kick-off meeting. Currently, PCIS is scheduled as a nine month effort to begin early in 1993. SEVWG/CIVCWG have raised the issue that a nine month effort is inadequate to accomplish progress toward a test suite.

- Kurt Gutzmann presented one potential approach to establishing a taxonomy for CIVC-A. The approach is rule-based (Prolog) and would aid in automating coverage analysis at the cost of integration with the framework.

- SEVWG also discussed avenues for recording implicit test objectives. When a validation test calls "other" CAIS interfaces to establish an initial context or examine results, implicit test objectives may be satisfied. Currently, we have no mechanism to record these tests.

- SEVWG briefly discussed CAIS evaluation. We concluded that a mechanism should be developed to evaluate CAIS-A implementations.

- White Paper entitled "APSE Related Standards: A Survey and Recommendations for Assessors." The paper will cover various APSE standards, starting with the list generated by Peter Clark and distributed at the joint CLASSWG/SEVWG meeting. Existence, policy for use and actual use are all three important. Any developing standards should also be considered. The paper will recommend which standards to pursue with assessment. The paper will also recommend adoption of a single process for developing and managing validation suites. SEVWG/CIVCWG would like to elevate a concern for subjects such as test selection, suite maintenance, coverage analysis, and usability to the software engineering community in general. SEVWG would like to encourage the philosophy that most standards need an implementation validation mechanism, which must be managed and maintained in the same sense as the standard itself.

Projected Work:

- White paper: "APSE Related Standards: A Survey and Recommendations for Assessors."

Deliverables Due Next Quarter:

- SEVWG plans to have a draft of the white paper ready for the June meeting.

Presentations Planned: None

Other Significant Information: None

Action Items:

- Ray Szymanski. Pulse Currie Colket to send copies of the Winnersh Report.

- Ray Szymanski. Draft a letter to Currie Colket regarding schedule for PCIS validation.

- Kurt Gutzmann. Put Prolog rules and facts onto the NET for comment.

- Gary McKee. Pulse Duston for Intermetrics Performance Analysis Report 5.1 and pulse Jack Foidl for IDA CAIS-A Usage Model Report.

- Tim Lindquist. Send Kurt Gutzmann a copy of the in-draft paper to be submitted to the December APSE conference.

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX A

10.0 LIST OF ATTENDEES

Ashby, Sam
Boeing Military Airplane Company
P.O. Box 7730, MSK80-13
Wichita, KS 67277-7730

Brashear, Phil
CTA, Inc.
5100 Springfield Pike
Dayton, OH 45431

Burlakoff, Mike
Route 2, Box 324
Springfield, MO 65802

Chilson, Lynn
SofTech, Inc.
1300 Hercules Drive
Suite 105
Houston, TX 77058

Clark, Peter
TASC
55 Walkers Brook Drive
Reading, MA 01867

Crawford, Bard
TASC
55 Walkers Brook Drive
Reading, MA 01867

Foidl, Jack
TRW, Systems Division
Suite 205
9265 Sky Park Court
San Diego, CA

Francl, Fred
Sonicraft, Inc.
8859 S. Greenwood
Chicago, IL 60619

Gicca, Greg
Sanders Associates
NCA1-2232
95 Canal Street
Nashua, NH 03061

Gutzmann, Kurt
SofTech, Inc.
1300 Hercules Drive
Suite 105
Houston, TX 77058

Hackett, Kevin
SofTech, Inc.
10875 Rancho Bernardo Road
San Diego, CA 92127

Hellen, Stephanie
Hughes Aircraft Company
P.O. Box 92426
MS RE/R11/10046
Los Angeles, CA 90009

Lawlis, Patricia Maj
AFIT/ENC
WPAFB, OH 45433

Leavitt, Tom
Boeing Military Airplane Company
P.O. Box 7730, MSK80-13
Wichita, KS 67277-7730

Lindquist, Tim
Computer Science and Engineering Dept.
Arizona State University
Tempe, AZ 85287-5406

Martin, Ronnie
Software Engineering Research Center
Department of Sciences
Purdue University
West Lafayette, IN 47907-2004

McKee, Gary
McKee Consulting
P.O. Box 3009
Littleton, CO 80161-3009

Mulholland, Sandi
Rockwell International MS124-211
400 Collins Road, NE
Cedar Rapids, IA 52498

Nejmeh, Brian
Instep, Inc.
13526 Copper Bed Road
Herndon, VA 22071

Szymanski, Raymond
WRDC/AAAF-3
WPAFB, OH 45433-6523

Thomas, Jerry
Naval Ocean Systems Center (NOSC)
San Diego, CA 92152-5000

Wills, Betty
CCSC/XPTB
Tinker AFB, OK 73145

Stiles, Lloyd
FCDSSA San Diego
200 Catalina Blvd.
San Diego, CA 92147

Terrell, Kermit
Boeing Military Airplane Company
P.O. Box 7730, MSK80-13
Wichita, KS 67277-7730

Weiderman, Nelson
Software Engineering Institute
Carnegie-Mellon University
Pittsburgh, PA 15213

APPENDIX O

MINUTES

OF THE

EVALUATION AND VALIDATION TEAM

GENERAL SESSION

31 MAY - 01 JUNE 1990

The task for the Evaluation and Validation of Ada Programming Support Environments (APSEs) is sponsored by the Ada Joint Program Office (AJPO)

TABLE OF CONTENTS

Section | Page

1.0 WEDNESDAY, 30 MAY 1990 3
1.1 Chairman's Corner 3
1.2 CAIS Implementation Validation Capability Update 5
1.3 Ada Compiler Evaluation Capability Update 6
1.4 Reference System Update with Conference Summaries 9
1.5 Final Report Discussion 12

2.0 THURSDAY, 31 MAY 1990 15
2.1 Discussion on the Recommendations for the Final Report . . 15
2.2 ACEC: A User's Perspective 19

3.0 WORKING GROUP STATUS REPORTS 27
3.1 Ada Compiler Evaluation Capability Working Group (ACECWG) Status Report 27
3.2 CAIS Implementation Validation Capability Working Group (CIVCWG) Status Report 29
3.3 Classification Working Group (CLASSWG) Status Report . . . 30
3.4 Requirements Working Group (REQWG) Status Report 33

LIST OF APPENDICES

10.0 LIST OF ATTENDEES 35

1.0 WEDNESDAY, 30 MAY 1990

1.1 Chairman's Corner

Raymond Szymanski
Wright Research and Development Center (WRDC)
Wright-Patterson Air Force Base (WPAFB)

Raymond Szymanski, chairman of the Evaluation and Validation (E&V) Team, opened the May meeting by updating the Team on the significant events which have occurred since March.

Conference Presentations:

- Mr. Szymanski was invited to speak at the 1990 Avionics Symposium held 12-15 March 1990. His subject was the Evaluation and Validation Task. The Avionics Symposium generated several other opportunities for briefings at Wright-Patterson Air Force Base and the Pentagon.

- Mr. Szymanski gave a briefing and participated in a panel discussion at the Software Technology Support Center, 23-27 April 1990. He spoke on Ada Compiler Evaluation and Selection. Mr. Szymanski participated on the Ada Compiler Quality Testing Service Panel which included Dr. Solomond.

- A briefing given at the Mission Critical Computer Resources Meeting on 20 April 1990 was a direct result of the Avionics Symposium. The E&V Task has been granted acceptance into SENTAR which is connected with the Aeronautical Systems Division (ASD). It is now Mr. Szymanski's task to find a home for the Ada Compiler Evaluation Capability (ACEC), the CAIS Implementation Validation Capability (CIVC), and the E&V Reference System. This has also created a potential for additional funding.

- Another briefing generated by the Avionics Symposium was the HQ/USAF SCTIA held on 8 May 1990. Mr. Szymanski covered the E&V Task and the Ada Compiler Quality Testing Service.

Contractual Efforts:

Ada Compiler Evaluation Capability:

- The Software Engineering Institute (SEI) and the Wright Research and Development Center have a Memorandum of Understanding. The SEI has promised to provide the E&V with feedback on their efforts with the ACEC.

- Version 2.0 was officially accepted by the Air Force on 23 May 1990. This includes the documentation and the software. A pre-release was sent to Data and Analysis Center for Software (DACS). They will soon be shipping copies of the ACEC Version 2.0.

- Version 3.0 is currently in negotiation. The deadline of mid-June will be met. This opens the way for Boeing to officially start working on the next version of the ACEC.

- Independent Testing Activities:

    - WRDC/AAAF are testing the ACEC in-house at Wright-Patterson.

    - McKee Consulting has been testing a Meridian compiler.

    - A USAF Reservist Project is currently ongoing on the ACEC.

E&V Reference System:

- A Technical Interchange Meeting (TIM) was held on 9 May 1990. It covered a detailed analysis of Version 3.0 changes.

CAIS Implementation Validation Capability:

- A TIM was held 17-18 April 1990. The key issue was the Test Selection Criteria for the CAIS-A specification. Another topic of discussion concerned information that NASA is seriously considering scrapping the current work on their environment for the space station and moving to a CAIS-oriented environment.

Upcoming Events:

- At the direction of Dr. John Solomond, Ray Szymanski has become a member of the Ada Executive Officials Evaluation and Validation Working Group. Meetings will be held 25 June 1990, 16 July 1990, and 20 August 1990.

- Mr. Szymanski will be giving an E&V Task overview at the Ada Board Briefing on 29 June 1990.

- ACEC Version 3.0 Kick-off Meeting will be held 2-3 July 1990.

- Mr. Szymanski will present an E&V Task overview at the Ada Executive Officials Briefing on 6 July 1990.

- Ada 9X Evaluation and Validation Meeting is tentatively scheduled for 23-27 July 1990.

- Summer SIGAda ACEC Presentation will be held 22 August 1990.

- The E&V Team Final Meeting will be held 11-14 September 1990. Mr. Szymanski will recognize those people who have made a contribution to the E&V Task.

Mr. Szymanski concluded his remarks and introduced the first presenter of the day, Kurt Gutzmann.

1.2 CAIS Implementation Validation Capability Update

Kurt Gutzmann
SofTech

Mr. Gutzmann displayed a slide of the CAIS Implementation Validation Capability (CIVC) status indicating three items; the Beta Test Suite, the Test Selection Process, and the Hypermedia Trade Study. He explained that SofTech's major focus has been on porting the test suite for use with CAIS-A. This resulted in approximately 160 test cases in the Beta Test Suite. Delivery is scheduled for 17 July 1990. An operator's guide will accompany the Beta Test Suite.

The Test Selection Process reduced all test possibilities into a set of tests providing the most effective use of the budget. A Technical Interchange Meeting (TIM) was held on that subject and a white paper was produced.

A trade study of Hypermedia products was performed to help determine which current product would be most effective. Approximately ten different Hypermedia products were evaluated. Apple's HyperCard Version 2. with the OSHA Sticky Text properties and programmability appeared to be the best choice. A technical report was produced summarizing the study.

Dr. Crawford asked for a brief description of Sticky Text. Mr. Gutzmann explained Sticky Text as the ability to associate some function with only a region of text in a text field rather than the whole text object.

Mr. Szymanski asked if it would be beneficial to revisit the Test Selection Process issue in more detail. He was concerned whether the Test Selection Process was an Area A contrast or if it was coordinated with the Issues and Strategies Document. Mr. Gutzmann stated the objective of the Test Selection Process was to obtain a test suite that optimizes the available resources while providing maximum validation and capability. Estimations of possible tests vary from 10,000 to one million or more.

The CIVC used the Taxonomy to select tests. The Taxonomy enumerated possible arrangements of entities in the CAIS. It did not, however, specify function or operation; therefore, it really did not guide the selection of tests for CAIS. So, the Phase I test suite was largely based on linear processing of the specification.

Tim Lindquist has identified several criteria, in the Issues and Strategies Document, for selecting tests. These include facilities to enhance transportability, facilities difficult to implement, areas of broadest coverage, and facilities giving more implicit information on the quality of the interface set. These are the proposed selection criteria for Phase III.

In addition to the criteria, a more rigorous approach to obtaining information was desired. An item called a Metamodel was devised as a real entity relationship model to explain the CAIS-A. The Metamodel will guide the test selection process within the aforementioned criteria. It will also help assessment of the test coverage.

A suggestion was made to look into applying repository technology to test selection. There was general agreement that that might be worthwhile investigating.

Ms. Mulholland inquired as to the manner in which assessment of evaluation coverage will be handled, or how certain areas will be prioritized. It was explained this would fall under the criteria of maximization of coverage. Some sort of function or supporting information model would be developed to evaluate coverage from which optimization could readily be accomplished.

The issue of Portable Common Interface Set (PCIS) involvement in CAIS-A was discussed. According to Mr. Szymanski, development of a validation suite for CAIS-A will continue on its present course without a lot of effort towards PCIS. However, effort will be made to ensure that the validation for CAIS-A will be compatible with PCIS. After completion, the CAIS-A validation suite could then be used as a foundation for the PCIS validation suite.

Mr. Szymanski indicated that some thought is being given to which organizations might be most interested and qualified to head up the PCIS effort. Details of this investigation would be forthcoming in a white paper.

1.3 Ada Compiler Evaluation Capability Update

Sam Ashby
Boeing Military Airplane

Mr. Ashby stated that his organization would be relocating from Wichita, Kansas to Seattle, Washington. This meant the Ada Compiler Evaluation Capability (ACEC) contract would move as well. Focus of the discussion then shifted to the status of each of the ACEC versions and to future direction.

There were 67 problem reports received against Release 1.0. Many of which were resolved in Release 2.0 before that effort was closed. The Software Problem Reports (SPRs) were satisfactorily addressed, however. Mr. Ashby stated that any new problem reports received will be addressed in Release 3.0 and the door would then be closed on Release 1.0 issues.

Release 2.0 was accepted by the Government and is to be available from the Data and Analysis Center for Software (DACS) by early June. This release contains some additional tests plus assessors for debuggers, in addition to a diagnostic assessor, a library assessor, and the single system analysis.

There was some concern over the exact titles of the various releases (Version 2.0, Release 2.0, etc.). Although the number is the significant element, the titles should be consistent.

An ACEC tutorial is anticipated by Boeing at TRI-Ada '90. It will cover using the ACEC in relation to running the test suite and analyzing the results.

Discussion then turned to the proposal for Release 3.0 which included several topics.

- Organizational issues involve renaming and grouping all the tests.

- Analysis issues are concerned with how results are presented after analysis. A number will be generated for each group of 20-30 tests in order to weight that group as well as each individual test.

  Also included in this area is the intention to review the statistical analysis model and to determine if there is not a better model to use for analysis.

- Attempts will be made to improve the user interface of the test suite and the tools, as well as tool automation.

Mr. Ferguson asked about the removal of Digitial Equipment Corporation (DEC) biased batch suites and if those were part of the interface automation or of the interface modification. Mr. Ashby explained that had to do with DEC Control Language and the Interface to Starlet. These are still included in the new version.

- Capacity testing will be included in Release 3.0. This will provide the capacities of both compile time and run time. Run time will be in terms of the sizes of various things and how they affect the system itself.

- Systematic compile speed is concerned with the amount of time required to compile some of the tests resulting in the ability to analyze the compilation time.

- The primary issue concerning additional performance tests has to do with systems having cache and the ability to determine how that affects the performance. Variations observed in execution time may be relative to cache in some cases.

- Compilation system reliability was originally proposed for inclusion in Release 3.0, but after further review, it was determined this would be too great an undertaking and was subsequently canceled.

- Improvements in user documentation are intended. A preliminary summary will be included to give the user an overall view of the ACEC without having to go too deeply into the documentation.

  A troubleshooting list will be included to help determine what might be wrong when problems occur on large and intermediate systems as well as smaller systems.

  It was suggested an index of some sort be included to assist the user in tracing entities and relationships between entities. It was also suggested a Hypermedia product might prove to be useful in compiling or generating the index.

- Run-time memory size will be investigated in the new version, particularly on embedded systems. Technical constructions as well as general will be considered.

- Comparison with related test suites is another item proposed but will not be included.

- The maintenance issue proposed has to do with corrections relative to SPRs. There will not be any new capabilities, but simply correction of problems in a new point release to Release 2.0. It is intended to also do this for Release 3.0.

  Duration of the Release 3 update and maintenance is expected to be about 21 months, July 1990 through February 1992.

  Although it was not definate whether the point release will be distributed automatically to users possessing the original, or only a notice will be sent indicating a new release is available, the general consensus was it would not be fair to users to have to remit full price for a small update.

Discussion of proposals for Release 3.0 concluded and Mr. Ashby then turned to the anticipated future for ACEC work.

- It is hoped the quality testing service will take place to help instruct users in how to use the test, how to analyze results, and what to do with the results.

- There was discussion on not trying to do everything serially relative to Ada 9X, but doing it parallel. No further information was available, a meeting was scheduled for the end of July on the issue.

- There was interest in the ACEC getting into other APSE tools and what the ACECWG is doing in that area. The intention was to go beyond just those tools within the compilation system. This brought up the suggestion to not put this issue under the term ACEC, but to present it at the REQWG.

- Finally, the future of the ACECWG organization was of concern and discussion on that issue would be welcome.

Mr. Ashby then turned the floor over to Mr. Szymanski who reminded everyone that the ACEC program was funded only through 30 September 1990. He indicated, however, there were several potential sources of funding including the Ada Joint Program Office. The level of funding required was approximately ten person years. Mr. Szymanski also indicated that if funding is not available, there would be no reason to continue the ACEC program.

Mr. Szymanski took an action item, with Boeing's permission, to excerpt some portion of their technical proposal to illustrate in greater detail what was planned in the next release.

Mr. Szymanski concluded by briefly explaining why the comparison with related test suites and the compilation system reliability issues was dropped from the Release 3.0 proposal. The assumed level of funding for Release 3.0 was used as a gauge when measuring the proposal and the cost of several items was weighed against the return of investment.

Two of the items (comparison with related test suites and compilation system reliability) were determined to be sufficiently low in priority, and in one case excessively high in cost, and were therefore cut from the proposal to meet the estimated budget.

1.4 Reference System Update with Conference Summaries

Dr. Bard Crawford
The Analytical Sciences Corporation (TASC)

E&V Reference System

Dr. Crawford's presentation covered the Reference System Status and a few technical issues of which the most important is integration. He defined integration as a characteristic of an APSE which distinguishes it from a simple collection of tools. Another issue concerns the categories of E&V technology. A third issue is that of standards and how they relate to APSEs and integrated APSEs. There is much confusion about the issue of all the different standards that people are considering and how they relate to one another and to environments.

Reference System, Version 3.0 will be delivered to Ray Szymanski in September. The following chapter by chapter listing provides an overview of how this version will differ from previous versions.

Reference Manual, Version 3.0:

- Chapter 2: Some of the overview pictures of the scheme have been revised.

- Chapter 3: Some minor wording changes were made. Section 3.3 Whole-APSE Functions was added, and Section 3.4, Whole-APSE Attributes, has been revised.

- Chapter 4: Figures are being updated. Cross-references of functions to products and products to functions will be added.

- Chapter 5: A new figure has been added along with a Maintenance Tools Set.

- Chapter 6: A new figure has been added. Other additions include a Vendor Support Attribute, new references based on new Guidebook entries, an integration attribute, and the Document Accessibility definition.

- Chapter 7: New functions have been added under the Requirements/ Design category and to Maintenance Tools. There will be a new reference based on the new Guidebook entries.

- The Change Request Form is being revised.

- Notation was changed from little circles and links with arrows, to the entity-attribute type of notation. Corresponding to this, changes have been made to the diagrams appearing at the beginning of each chapter.

Guidebook, Version 3.0:

- Chapter 3: Some material has been revised in light of the emphasis on integration.

- Chapter 4: Some new synopses have been added. This chapter contains important background documents. One example being the new PIWG special issue of Ada Letters. Documents on the activities at the STSC at Hill Air Force Base have been synopsized. Also included will be a revision on CAIS/CAIS-A. Peter Clark is working on the Information Resources Data Standards (IRDS), and there have been a few upgrades.

- Chapter 5: There have been revisions on ACEC, PIWG, and the SEI Compiler Evaluation. An addition is Hartstone which was a paper written by Nelson Weiderman and associates on the Hard Real-time Test Suite which was included in the PIWG special issues.

- Chapter 6: Updates and improvements have been made to some of the checklists.

- Chapter 7: An example has been included of one of the items from the STSC work. Another entry from the Lyon's book has been added; the Ada Europe Environment Working Group text that was released a couple years ago.

- Chapter 8: The CIVC-A information has been revised.

- Chapter 9: Changes were made to the STEM/Draper, Evaluation Services, and CECOM reports.

- Chapter 10: The checklist has been revised.

- Chapter 11: Changes include two papers from the PIWG special issue.

- Chapter 12: A new checklist has been included.

- Chapter 13: Changes were made to the Integration Characteristics of Tools checklist and the Integration Services of APSE checklist.

- Chapter 15: The E-Mail checklist has been revised.

- Chapter 99: This is the catch-all for those items omitted from earlier chapters. Included are the Tracking checklist, Scheduling checklist, and STEM/TRW - Documentation Tools Evaluation.

Section 3.3 of the Reference Manual is a recent draft and is an adaptation of some of the material covered by Brian Nejmeh in his presentation last March. He emphasized that there are two responsibilities in developing a truly integrated APSE. First is the responsibility of the APSE itself with the infrastructure and services it has to provide, and second is the responsibility of the tool.

Mr. Nejmeh described the following four dimensions of integration:

- Data Integration Services includes four subcategories: (1) have the output Tool-1 go to the input of Tool-2; (2) the creation of relationships; (3) centralized configuration management; and (4) multiple manipulation of the same data.

- Presentation Integration Services.

- Interoperability Integration Services.

- Process Integration Services.

- Coordination Services.

- Monitoring Services.

A matrix covering these items was generated and is in the current draft of Version 3.0. It was established through this matrix that these functions are covered in the document. A large hole was discovered in the area of attributes. Mr. Nejmeh says the key characteristics of an integrated APSE are integration, scope, and openness. The topic of scope is covered under completeness. By openness, Mr. Nejmeh meant the ability to add new tools, expandability, augmentability, etc. This has been covered with just a little wording change.

The area of integrating services which is a functional area is most important. Dr. Crawford is proposing the addition of integration at a fairly high visibility level in the attribute taxonomy. Looking at integration from the tool responsibility side, a checklist was created based on the characteristics of the tools that are integratable in APSEs.

Conferences

Dr. Crawford summarized the conferences recently attended by either himself or Peter Clark. The first was the Software Technology Support Center (STSC) Conference, which covered the Software Tool Evaluation Model (STEM), Ada language and Ada compiler evaluation, software management and specific tools, and HQ USAF/SC selected presentations.

Peter Clark attended the first annual Symposium on Environments and Tools for Ada (SETA1). The areas of interest were methods and tools for design, specification, and reuse; Ada libraries, configuration management, and version control; building, debuging, and testing real-time and distributed systems; and environment architectures and frameworks. Mr. Clark also attended an International Workshop on CASE. Different standards were reviewed and characterized in terms of attributes.

Dr. Crawford commented that a software engineering system from Hewlett Packard was mentioned in all three meetings. Ms. Mulholland said that the NIST working group had reviewed this model and rejected it. At one of the conferences, a speaker from DEC stated many people want a vendor independent standard. Because of the current confusion in the area of standards, they are trying to make DEC a standards-independent vendor so that tools can be written to their standard.

Whole-APSE Evaluation/Model Project

The model project is a two-step process. The products include the source code and documents. This can be visualized as being surrounded by an APSE and its tools which creates and modifies most of the middle items. This in turn is surrounded by the structured experiment which has a series of steps and questions. This begins by assigning roles such as the supervisor or administrator of the test and some evaluators. They then follow the steps in the outer layer in using the APSE and its tools to operate on the inner layer which produces the answers to the questions in evaluation.

This "Super Scenario" on software test evaluation includes the following "mini-scenarios": documentation, Unit Test A, Unit Test B, CSC Test, and Integration Test. It requires partial versions of several documents including the design document. A variety of elements of the APSE are used, including documentation tools, update tools, design tools, etc. This idea can be built upon to focus on other areas in testing.

Dr. Crawford closed his presentation with a videotape of a product called PLAN. The video covered usability testing of the product.

1.5 Final Report Discussion

Tim Lindquist
Arizona State University

Dr. Lindquist led a discussion on the E&V Team's Final Report. A draft outline, modeled on the KAPSE Interface Team (KIT) Final Report, was distributed. Section 4, Issues and Recommendations, has been added.

Each of the working groups has a suggested outline for their specific report. This includes a preliminary section giving information about that working group including the working group's charter, a summary of the products and a section describing the documents produced by that group. The final section on Futures would cover a brief description of what the working group felt was appropriate for future consideration, remaining issues, and recommendations.

There is a slight conflict concerning whether or not recommendations should be contained within each of the different working group reports or whether they should be within a separate section. In the KIT Final Report, there are two separate perspectives, one from industry and one from Government members as well as the individual working group efforts. The individual working group reports can be either of a local nature or very focused if the group has a specific problem domain.

Dr. Lindquist raised two points. First, in the KIT report the types of recommendations that appeared were very specific to the given working group. These should not be brought out to the level of team recommendations. On the other hand, there may be some recommendations important enough to include in working group sections and summarized in a separate section, such as Section 4. There should be an identification between the individual working group recommendations and recommendations that would derive from the entire team.

Dr. Lindquist enumerated the different categories currently existing under the recommendation section. The first is Tool Assessors; some sentences in this area concern the Tools and Aids Document which should be embodied in one set of recommendations. The Team needs to decide what the specific recommendations are for the Team level. Those recommendations may be flushed out by one of the working groups. The second category includes Whole-APSE Assessors; third is the Standards Validation; fourth is the Compiler Evaluation; fifth is the Automated Reference System; sixth is the Department of Defense (DoD) Coordination of APSE Assessment Activities; seventh is the Importance of Evaluation; and eighth is the Need for an Evaluation Service.

The final item in the outline is a message from Bard Crawford with comments on the Final Report organization. The first point is that the areas for making recommendations should be based upon the specific working groups involved in that area. The second point is that the Final Report should be fashioned after the KIT Final Report.

A discussion arose on whether to include the Tools and Aids Document in its entirety in the Final Report. Dr. Lindquist proposed that an abstract of the products be included but not the entire document as that would be a replication. Major Lawlis stated that only a preliminary version of the Tools and Aids was published. The final version of the document would be used for the Final Report. This would be the only place the document would be published.

Mr. McKee suggested keeping the Final Report concise and referencing any other important documents. Ms. Mulholland disagreed saying the reader would not take the time to search out a separate document. The main document of the Final Report should be short, concise, and abstract. It would be there for the managers to read and if more information is wanted it should be there in association with the main document.

Dr. Crawford said that a Requirements Working Group (REQWG) conclusion had been to write a fairly brief front document and then include both the Tools and Aids Document and the CAIS-A Issues and Strategies Document as appendices.

Dr. Lindquist does not support having the Issues and Strategies Document in the Final Report. The level of recommendations in the Issues and Strategies Document is not appropriate for the Final report. Dr. Lindquist feels this also applies to the Tools and Aids Document. There should be a minimum number of recommendations in the Final Report which would actually incorporate or abstract ten different recommendations so that one recommendation is a recommendation on tool assessment. It could even call out specific tools in the description of the recommendation but would not go to any deeper level other than just referencing the appropriate document. Dr. Crawford stated that Ray Szymanski is the one responsible for making this decision because the Final Report is his report to the Ada Joint Program Office (AJPO). The Team can make recommendations but Mr. Szymanski is the one to make the final decision.

Mr. Weiderman asked if it is Mr. Szymanski's report or the Team's report. He wondered if it was the Team making the recommendation to AJPO or the Team making the recommendation to Mr. Szymanski who has the right to modify it any way he sees fit. This area needs to be clearly defined.

Mr. Szymanski will respect the Team's opinion. Even if the Team included something that was unpopular with him, he would not want to edit it. However, there are always one or two people who have very strong opinions which are the only ones voiced which does not really represent a Team perspective. Mr. Szymanski will be very careful in those recommendations to ensure that there is not an individual who has a particular axe to grind within the confines of the Final Report. Mr. Szymanski stated he would probably not individually sit down and review the final recommendations, but will probably call on a few select people to help consider each recommendation.

Mr. Ferguson sees the Final Report as being Mr. Szymanski's report to the sponsor, assisted in preparation by the Team. If a recommendation is made, Mr. Szymanski needs to be able to defend it to the sponsor. Captain Abraham and Ms. Mulholland disagreed with this view. Mr. Szymanski assured everyone that the Team's work would be well represented in the Final Report.

Dr. Crawford commented that he sees no justification for writing Section 4. It is good to have a Team discussion about what the final overall recommendations might be from the chairman, but time should be spent in the working groups concentrating on the recommendations. Mr. Szymanski said that instead of synopsizing from all the different working group areas and putting them in Section 4, the synopsis should be up front.

Dr. Lindquist stated that there are two motivations for including Section 4. One, the recommendations need to be abstracted to a preselected minimal number. Two, the entire Team needs to be provided the opportunity to contribute to all of the recommendations. Section 4 is the best vehicle for doing this.

Mr. Ferguson asked that if the recommendations are a represented consensus, had there been any discussions about representing minority points of view. Mr. Szymanski replied the key would be to identify any points of agreement with the disagreeing parties and indicate the point of divergence.

Major Lawlis suggested a slight reorganization by making the overall recommendations, which would be a summary form, Section 3 and then making the working group reports, which would be more detailed, Section 4.

Dr. Lindquist asked if the executive summary of the recommendations should come at the beginning or the end of the report. It is a matter of choice whether the reader would look first at the beginning or the end of the document. Mr. Szymanski replied that hopefully they would look first at the table of contents. The Team then adjourned for the day.

2.0 THURSDAY, 31 MAY 1990

2.1 Discussion on the Recommendations for the Final Report

Major Patricia Lawlis
Air Force Institute of Technology, WPAFB

Major Lawlis led a Team discussion on the Final Report, focusing mainly on the Team recommendations. She commented that there were not many changes made. The executive summary was added to the document. The working group reports were changed to cover past accomplishments, contractual efforts, and a brief description of accomplishments with highlights of specific recommendations mentioned in Section 4.

REQWG contends that all the recommendations should be together at the end of the report in Section 4, and they should be brief. There should be no more than ten final recommendations. The reader can read the document, the executive summary, and then proceed directly to the end and read the recommendations. If the reader wants more information, they can go to the middle of the document for the details of the report. REQWG is seeking the Team's reactions concerning the organization of the report.

Mr. Ferguson said having just ten recommendations in Section 4 is a good idea, but other valid recommendations should be included. The minority viewpoint could be included in Section 4 which would raise the limit above ten, or could be placed in the working group sections. Mr. McKee suggested putting the minority recommendations in Section 3 along with a statement that these recommendations are from the working groups.

Major Lawlis said that Section 3 could be expanded any way the individual working group would wish. REQWG wants to have the main recommendations placed at the end of the document in bullet format. Major Lawlis passed out some material which was not in Dr. Lindquist's draft outline. She felt this information should probably go into the introduction section, but its placement would be up to Mr. Szymanski. Ms. Adams suggested also including a new executive summary.

REQWG had a brainstorming session on ideas for the recommendations to be included. The recommendations put forth for discussion are:

1. Develop DoD policy addressing assessment technologies.

2. Raise public awareness toward the importance of E&V to reducing risk in software intensive programs.

3. Do not continue E&V Team effort.

4. Continue to improve and enhance ACEC (reference the Tools and Aids appendix).

5. Do not establish QTS.

6. Do not mandate evaluation.

7. Develop systems development evaluation capability.

8. Promote use of team-developed products. (There would be sub-bullets here.)

9. Incentivize use of E&V technology.

10. Build an existing E&V technology--buy off-the-shelf.

11. Task SEI to include E&V technology in graduate and undergraduate degree curricula.

12. Promote collection of E&V data in a repository.

13. Develop an automated E&V reference system.

14. Do not internationalize the E&V effort.

15. Continue the emphasis on Ada and software engineering.

16. Emphasize the distinction between the evaluation and validation.

17. Encourage use of existing standards.

18. Push a validation effort for each standard activity.

Major Lawlis detailed a review schedule for the Final Report. The first draft will be on the Net by 8 June 1990; the second draft, based on Team comments from the first draft, will be due 13 July 1990. All final comments are to be submitted by 17 August 1990. This will provide a draft for Ray Szymanski in September 1990. During a discussion over deadlines for the working group reports, Mr. Weiderman said he thought putting Section 4 on the Net was more important. The recommendations should take precedence as opposed to the working group reports. The working group reports are fairly routine, but the recommendations will stimulate some discussion and so should be out on the Net at an early date. Major Lawlis agreed that the recommendations are clearly the most important part of the draft to produce first. The list put forth by REQWG is open to amendment and additions by the Team. It was thought especially appropriate that there were recommendations such as to not establish the QTS. The proposed recommendations are for inclusion in Section 4 and will be placed on the Net for Team comments.

Major Lawlis stated the remainder of the discussion should concentrate on the recommendations, to weed out those that are redundant or combine those covering the same issue. Dr. Crawford commented that he visualizes Section 4 with the recommendation stated in bold face along with a brief explanatory paragraph. Major Lawlis agreed.

A discussion ensued over the meaning of "consensus" for inclusion of a recommendation. Mr. Weiderman pointed out that the dictionary definition is unanimous consent. Mr. McKee stated it should be on the order of a two-thirds to three-quarters majority. Objections from one or two people on a certain recommendation are acceptable, but five or six people objecting would be too many.

Major Lawlis reminded everyone that the purpose of the discussion was to capture as much in writing as possible. This would help the REQWG in producing the draft. Mr. McKee said it is easier to offer a minority opinion in a discussion group than to offer it to the Team in writing. Therefore to some degree, objections on the Net have a larger degree of credibility because they require additional thought. Net activity will play an important part in filtering out minority opinions.

Major Lawlis asked if there were any additions to the list of recommendations. Mr. Munck stated, that while he did not have anything to add, he has been uncomfortable with the separation between the KIT work, the work by the E&V Team, and the formal specifics of CAIS-A, etc. Since the Government is about to participate in another effort, Mr. Munck would like to see some recommendation towards integration. Mr. McKee suggested a recommendation along the lines of E&V technology being integrated into the PCIS activity. Mr. Munck agreed but stated it should not be an entirely separate effort. Mr. Weiderman contended that he perceived Mr. Munck as saying that evaluation and validation have to be integrated more, not only for PCIS, but for figures across the board in the technology. Mr. Munck agreed saying that he is suspicious of formal specifics and formal methods, but suggested trying that in order to be part of the effort also.

Mr. Weiderman thought the idea was that the "E" should be integrated with the "V." Validation started out first on a separate track. The E&V Team then came along and kept the validation separate from the evaluation as far as the Ada standard is concerned. This should be better integrated. Mr. Munck agreed saying there should be more integration between the "E" and the "V" for designing.

Dr. Lindquist commented that another aspect of Mr. Munck's remark is the design and benefit by looking at the E&V requirements. The Team has not taken as much advantage of that in the past as it should have.

Mr. Szymanski asked for a clarification on recommendation 2. When talking about public awareness toward the importance of the E&V, does this mean the people with the funding or the policy makers in Washington? Who really is the public? Mr. Ferguson said number 2 was addressing the users. Mr. Szymanski then asked if this was the program managers. Mr. Weiderman said it also included the contractors. Mr. McKee read number 2 as applying to both the low level engineers and the money makers and managers. Mr. Impicciche said it was a very general idea to make people more aware. Dr. Lindquist said that recommendations two, nine, and eleven seem to be different aspects of one recommendation.

Mr. Terrell asked for a clarification on recommendation 7. Mr. Ashby replied that in recent experience the system was designed before the trace studies were done of the hardware and software. The tools used had already begun to use the database, some structure design, and so forth. If this is not carried forward, the work is thrown away and started anew. In number 2, there are users and designers, and Mr. Ashby was thinking of the designers.

Mr. Ferguson asked if in number 2 the contractors are the people that led the contracts or those to whom the contracts are awarded. Mr. Szymanski said it would be to whom the contracts are awarded. It could work both ways; you could either impose the test suites on them or you could be the user of it. Mr. Szymanski commented that the listed recommendations are in a general mode. Those from the working groups such as the CIVCWG will be very specific.

Dr. Lindquist disagreed with the appropriateness of including number 15. Major Lawlis said she had questioned it herself. Mr. Impicciche stated the recommendation was his. He feels the emphasis needs to be continued. He thinks there is a weakness out there starting in academia all the way through to the work environment of the software engineering practices which includes using Ada in that environment. Dr. Lindquist agreed but does not see that it should be a recommendation. It would be perceived as patronizing, but the people reading it will agree with that recommendation. Captain Abraham stated it re-enforces the Team's support that the effort needs to continue. This tells the AJPO that they still have a mission or function to do. Many agencies are negligent of this. If they obtain a stronger voice, they may eventually be successful in killing the emphasis to continue these practices. It was part of the discussion idea that there is enough negative force out there and the Team needs to re-enforce the positive side. While Captain Abraham agrees with this, she is not sure if this is the appropriate vehicle for raising this concern. The appropriate people will not be reading the Final Report. If they do read it, they will wonder what this has to do with E&V. She feels it takes away credibility by placing it in the recommendations.

Mr. Gutzmann commented that the issue is very nonspecific, and no one would read and act upon it. Mr. Impicciche disagreed saying that people are acting on that right now.

Mr. Szymanski said that the reader is Dr. Solomond and his staff. The recommendations are focused to indicate that AJPO as a program office ought to take these recommendations and act upon them. The software engineering is outside their daily jurisdiction. In light of the statement of pushing a validation effort for each standards activity, it could also be said to push an evaluation effort for each standards activity in the sense of the CAIS standards. The AJPO could say by having funded the CIVC it ought to fund the CIVC evaluation suite. That would be very specific. Something like continuing the emphasis on Ada is something that they do every day; their job is to make Ada a success. In light of Kurt Gutzmann's statement, that is not something they can act upon.

Mr. McKee agreed that it was obvious that recommendation 15 is both patronizing and inactive. He suggested taking out all the recommendations that do not have a specific action. Every recommendation is going to have a rationale and every one of those rationales should have a specific action that AJPO can take. If a specific action cannot be found, then the recommendation should not be in Section 4. Number 15 is one of those. Major Lawlis stated that REQWG may be able to combine some of the more specific recommendations with some others that might achieve the expected impact.

Mr. Ferguson said that the Ada effort is taking some negative publicity. Any time there is a chance for the Team to state its full support of the Ada effort, the emphasis on Ada and software engineering, the Team should do that and write a rationale for it, whether or not they can call out the specific idea for the AJPO to take.

Captain Abraham suggested taking number 15 and putting it in the executive summary as part of the Team position. The statement should be included that the E&V Team feels strongly about this. Mr. Weiderman suggested putting number 15 into the introduction section rather than as a recommendation. Mr. Szymanski said the recommendations on the emphasis of Ada and software engineering are directly out of the software master plan. The Team could state their support of the position of the software master plan and not include these in the list of recommendations.

Mr. Baker questioned numbers 16 and 19 being separate recommendations. Captain Abraham said that they are different but the recommendations would not have to remain separate as long as the difference is understood.

Mr. McKee commented that REQWG is making good progress on the Final Report. The Team then adjourned to their individual working groups.

2.2 ACEC: A User's Perspective

Gary McKee
McKee Consulting

The General Session reconvened at the beginning of the Thursday afternoon session. Gary McKee presented his research findings on the ACEC Version 2.0.

Introduction

The topic of Gary McKee's presentation to the E&V General Session concerned the usability (performing the intended job) and ease of use (performing the intended job with little frustration and anxiety) of the ACEC Version 2.0.

Mr. McKee worked on a MacIntosh II with a 40 megabyte internal drive and two 40 megabyte external cartridge drives, allowing him 120 megabytes of usable data space. Because the ACEC aggressively examines the compiler for bugs, three versions of compilers (Adavantage Versions 3.2, 4.0, and 4.1) were needed to get through the process of running performance tests, which indicates that the ACEC is worthwhile. All versions of the compiler had been previously validated through the ACVC which locates undetected errors in compilers.

Philosophy

The four kinds of users are ACEC developers, compiler vendors, evaluation experts, and application users.

The ACEC developers have a certain skill level and thoroughly know the system. The compiler vendors who use the ACEC get immediate information about their product from every test that fails or succeeds. Evaluation experts have experience with different evaluation test suites and understand evaluation without regard for a particular product. Application users have a series of questions and expectations.

Application users have certain expectations regarding the ACEC. Users expect the ACEC product to answer their questions and that the ACEC should be reliable and error-free. If users are evaluating a compiler with this tool, they have to trust the tool to work; otherwise, when a problem occurs, they will not know whether the problem is the tool or the compiler. A high level of trust with the ACEC is very important so users can assume that when a problem occurs it is a compiler problem.

Application users have questions regarding the quality, correctness, and speed of a compiler, as well as the reliability of a compiler product. In addition, they want to know the status of the product they have in-house and whether or not they should buy a new compiler to upgrade this product, or simply buy a maintenance product and assume that what they have is adequate.

Typically, application users will apply the ACEC prior to contract awards to determine what compiler product is needed to achieve a certain job for a certain proposal, as well as the cost involved in developing and maintaining a product for a contract. They want to know about quality, size, what the timing simulations are, and whether the compiler is easy to use.

Self-training is another issue for application users. In part, the ACEC is intended to be used for self-training which enables the user to learn to use the compiler, the environment, and the operating system.

Acquisition of the ACEC

After acquistion of the ACEC, Boeing sent Mr. McKee a stack of MacIntosh disks with 687-689 source files on them which he downloaded to the MacIntosh. Approximately 20 megabytes of empty space is required to download the system with the initial 20 megabytes of source data. Of the three 40 megabyte cartridges, two were archived and zeroed, allowing 80 megabytes of available space. The internal drive was alloted for the operating system, word processing, spread sheet programs, and analysis tools. Mr. McKee allocated 80 megabytes of storage and used one disk for the actual evaluation, allowing 40 megabytes for working space and 40 megabytes for archivable information.

The approximately 700 source files were sent in one physical directory with no information in the documentation or command files to facilitate separating those files into usable chunks. Mr. McKee set up one directory exclusively for the accumulation of test programs that he wrote during setup, and no data was placed in multiple places.

Mr. McKee broke up the 1,384 tests (262 programs) by taking a listing of the source files and partitioning them alphabetically. He created script programs to find those files. Any very large source files were pulled out and placed into one of six other directories. Mr. McKee made a hardcopy of the command files which were essential to execution and were very valuable in setting up the subdirectories.

Because setting up subdirectories and partitioning the source files into the subdirectories is time consuming and complex, Mr. McKee recommends that Boeing consider setting up command files and a subdirectory structure to simplify this task. Likewise, the user documentation was no help to Mr. McKee in categorizing or separating the subdirectories, and therefore needs to be improved.

Preparing the Environment

Some major decisions must be made before the performance tests can be run, the most important of which is choosing a math package. After looking at the major math packages (such as Deptest, Math-Dependent, PORTABLE_MATH), Mr. McKee tried PORTABLE_MATH which is the most general purpose solution. Although this math package works on the VAX, it was too large for his compiler.

Mr. McKee wrote a test program to determine that his compiler used 'ADDRESS. The compiler produces addresses for objects or labels. However, Version 3.2 did not do arithmetic on those objects. Version 4.0 did do the arithmetic, but, 'ADDRESS only worked for objects and not labels.

Mr. McKee solved the problem by taking the assembly line code that Boeing delivered to get ADDRESS which is an equivalent workaround. This is used for calculating code expansion size, so it is absolutely essential to everything that occurs in all future tests. The compiler "thought" that 'ADDRESS worked, so it was only the link that failed.

The problem is in the Meridian compiler, not in the Boeing test suite. Boeing handled the problem effectively by offering a way to use 'ADDRESS and get ADDRESS code. Boeing gave two alternatives and documented them adequately. Mr. McKee determined which one would work on the Meridian compiler.

The documentation recommends that whenever possible, the supplied math package should be used. However, the Meridian-supplied math package does not supply all of the essential functions. Therefore, PORTABLE_MATH, in its entirety, was the next alternative. When it did not compile, Mr. McKee modified PORTABLE_MATH to use the Meridian math package whereby supplying the essential function or allowing the use of existing codes. The program was reduced by two-thirds and small enough to compile.

The resulting workaround was a instantiated PORTABLE_MATH as modified for TYPE_FLOAT, and then only for the two different data types: real and double (real is DIGITS six, double is DIGITS nine, and TYPE_FLOAT is DIGITS 12). The workaround solved the type-double and type-real problems by using one instantiation of PORTABLE_MATH for type float. But because the only workaround that the compiler allowed may have invalidated some of the tests, the quality of the product was already in question. Therefore, Mr. McKee recommends that the manual indicate which tests are predicated on the math packages.

Compilation Process

The compilation process consisted of sorting out 16 directories and then running a series of *.a files through the INCLUDE program which resulted in a series of *.ada files. The *.ada files were compiled to get Library units. The Library unit was run through the linker which resulted in an executable program. Boeing designed their command files to never save these *.Ada files. A file is compiled, built, and deleted.

Merdian performed all of the validations without using any of the optimization switches. The compiler does not complete with optimization on, and there is no indication as to which tests are only relevant if optimization is on.

According to Boeing, some of the tests are comparisons between optimized and nonoptimized code, and although they would show no effects, there is no way to determine if that is their only impact. Without this certainty, it is unclear how to explain or interpret the results. However, nothing in Boeing's documentation indicates that these tests are only good if you have a compiler that optimizes, and the appendix in the Version Description Document (VDD) has a list of optimizations and tests that apply to the documentation. Therefore, the only information gained from running these tests was that the compiler did not do any optimization.

Disk Space Requirements

Disk space needs to be managed efficiently. The programs prior to execution require about 20 megabytes. Another 10 megabytes are needed for the source data. At least 30 megabytes are needed to get to the point of running the data. Saving the *.ada ACEC files yields about 40 megabytes of data. Including the command files, the documentation files, and assessors adds another 6 megabytes. Saving the library files adds an additional 8-10 megabytes.

The range of storage used for programs can be anywhere from 10-30 megabytes, with the average being 20 megabytes. Individuals need to document their experience so others have some data point for reference.

INCLUDE Program

INCLUDE expands all of the actual tests by a factor of about four. This has an impact on analysis because the code is so large that it becomes difficult to determine why a test fails. Four files are included on every program, and another three to six files are included on some programs. The documentation gives no indication as to what is included where.

Boeing solved the problem of how to instrument tests in a standardized manner by using PRAGMA_INCLUDE and INCLUDE processor. As a general solution to software development this would not be helpful. But because the primary use of the INCLUDE program is to add instrumentation to the test, for this particular purpose the overall strategy was acceptable.

Unfortunately, INCLUDE performs functions other than adding instrumentation codes. Mr. McKee's conclusion is that INCLUDE is inefficient in terms of both space and time, and is not well documented.

Compilation Times

In general, compilation times averaged from three to four minutes to compile and link a correct Ada program. Compiling a single *.A file took 31 seconds, and the *.Ada file took about a minute longer. The INCLUDES added an additional minute per program. Approximately one third of the compilation time could be reduced if the INCLUDE files were modified. One INCLUDE, Inittime.txt, is 14 pages of listing and is included in every Ada program. Due to location dependencies, 11 or 12 of those pages are not needed.

Three of the INCLUDE files start the timing data as part of a block, code exists, and the rest of the timing data is inserted at the end of the block so that the whole block is tested and measured for size. However, the Inittime.txt file's purpose is to do null loop analysis. It has a great deal of commentary to indicate what is being done while going through the code.

The important point is that the actual compilation time with INCLUDE, but not including setup and analysis time, took almost 18 hours. If INCLUDE is managed more efficiently, this time can be reduced by one third.

Execution Cycle

Once Mr. McKee separated the tests in different directories and was ready to execute, he created a command file for each directory to log the information. The command files contain the information concerning what is to be done and are essential to success.

Mr. McKee ran three sample programs to avoid the possibility of one running successfully by accident. He took the first three programs and test-set number one. He compiled, linked, and ran them and received logged results. Nothing in the results indicated that anything had gone wrong, but he had no way of knowing for certain because no output examples are in the documentation. However, since he made some progress, Mr. McKee ran the first directory of 32 programs. Every third program failed, and after a while every program failed.

The analysis led to the conclusion that it was a design error of the ACEC that caused the problem. Mr. McKee determined that he would have to either fix the problem or stop altogether. There was no workaround to this problem because every test would fail sooner or later. Therefore, he decided to resolve the problem.

Mr. McKee's first consideration was that he had no software documentation, and therefore, he did not know if he understood Package Global well enough to know if he had the correct solution to the problem. The second consideration was that Inittime.txt is 14 pages long and he did not know if his changes would fix the problem or only move it. The third consideration was that Package Global is very large, and he did not know what other problems he would create by making his changes.

Mr. McKee's analysis of the problem revealed that an incorrect engineering decision had been made. Part of what the ACEC does with these INCLUDE.TXT files is to use them to analyze the code expansion for different tests by putting a label before and after the code and use "pick_value" or "get_address" to get the address of those labels and calculate the difference. However, the data objects wherein those addresses were stored before the arithmetic were of type INT. INT is specified as being within the range of -32K to +32K. Theoretically, INT can be stored in 16 bits. Mr. McKee's compiler was so efficient that it only used 16 bits. Whereas, if a 3 megabyte address space is used, sooner or later it will run out of storage location. But for the ACEC, UNCHECKED_CONVERSION was used to store a value of type SYSTEM.ADDRESS in an object called "start_address" or "stop_address" of type INT. If the compiler is efficient and uses 16 bits for that storage, then part of your address is lost or overwritten with other data space.

On the MacIntosh, the programs are loaded into memory sequentially, and therefore, as more programs are generated than have ~~been~~ executed, more and more memory is used until the 3 megabyte space is reached and then it starts over with zero and moves forward again. Any one test executed in manual mode works because it is in the lower end of memory. As more and more are put in automatically, eventually the 64K limit is reached, and it falls apart. Therefore, every third program ran failed because after each failure Mr. McKee started over again which would lower the memory capacity allowing him to run three more programs before it got up high enough to fail again.

To remedy this, Mr. McKee went into Package Global and changed the object definitions for "start_address" and "stop_address" to use a 32-bit integer number called "double_INT". Then he went into the INCLUDE file, Inittime.txt, and performed TYPE_CASTING instead of UNCHECKED_CONVERSION. After Mr. McKee changed TYPE_CASTING to double integer; did the arithmetic; and then changed TYPE_CASTING back into single integer, the problem was resolved.

The large size of both Inittime.txt and Package Global was a negative factor for Mr. McKee, and he recommends reducing the size of both as much as possible.

After Mr. McKee recompiled the entire batch, he was ready to run the execution cycle again. This time the first six programs ran successfully, at this point, however, the system began to crash on every program. The problem was "Memory_Full error" and the operating system crashed. The problem was that the heap space used by the Ada compiler was exhausted by a certain class of programs, and the compiler was unable to recover from a "Memory_Full error." Meaning that when objects are created on the heap, and enough are created that the available memory is exhausted (in this case 3 megabytes or some x number of megabytes), the compiler is not robust enough to recover after that particular program fails. Instead, the program fails, the MPW shell fails, and the operating system crashes. This problem, however, may not be in the compiler, but in the MacIntosh operating system. The documentation shows no evidence that the user should expect these kinds of problems. According to the VDD some systems run these programs successfully, while some fail consistently.

In most cases, however, the operating systems of those systems do not crash. The MacIntosh operating system does not operate in supervisory mode on the 68020. It was not designed that way, and therefore, does not have the power that a good operating system can have. The MacIntosh is vulnerable to program failure because it does not have a supervisory mode.

The quarantine list in the VDD indicates that these tests failed on at least one system, but not all of them; some systems do this job correctly, some do not. At less than halfway through the execution cycle, Mr. McKee dropped 43 tests as not being doable. His frustration level was high because every time the system crashed, all the log files for the tests that did pass were also lost. Each time the system crashed, he had to restart from the whole set. To combat this, Mr. McKee modified his command files to completely save off a log file every five tests, and the system did not fail.

Mr. McKee proceeded until he got to the KERNEL tests. Progress was slow, so he eliminated the KERNELxx tests at Directory 4. Everything ran smoothly for

Directories 5-15. All of the real frustration and anxiety of running the ACEC occurred at the beginning of the process. The latter half of the ACEC was easy for Mr. McKee to use and he had no problems. He received the log files he expected, and the performance tests completed as expected.

Mr. McKee's main criticism has to do with the interactive tests. Four interactive tests require keyboard input and should be extracted and isolated in Version 3.0. Also, as a typical user, Mr. McKee had difficulties throughout because the documentation was not clear. Another criticism was that three or four Disk IO tests that ran correctly gave him failure results. The tests ran correctly; they provided correct log file results; and the results were "failed" because "this test requires 10 megabytes of free disk space." Because Mr. McKee did not have 10 megabytes left of his 80 megabytes, the test stopped. Nothing in the documentation indicates that 10 more megabytes of disk space is needed.

The total execution time for the CPU was 24 hours and 43 minutes, averaging about 7 minutes per test. Specifically, 63 percent of the total 326 source files ran correctly and did not fail.

All link failures were due to a single reason: the Meridian compiler uses an existing Macintosh Linker that has a 32K address space limitation. Any library unit has a maximum 32K of address space. Any program that is larger than 32K will fail. This will be fixed in Version 4.1.

The compilation failures occurred for a number of different reasons, some of which will be submitted to Boeing as probable errors.

Setup in the null loop took about four minutes of execution time per test. The setup in the null loop is executed by Inittime.txt. The average program took seven minutes, and the shortest program took four and a half minutes. If each test could be reduced by four minutes, some of these tests would run very rapidly and all would be at least 50 percent faster. The 24 hours Mr. McKee spent could be reduced by five hours if he had not had to perform the Inittime.txt operations. Therefore, Inittime.txt needs to be re-examined. Is Inittime.txt correct? Is it a more effective way of performing diagnosis? Is it cost effective? Also, Inittime.txt produces approximately a quarter to a half of the log files text output. Moving this log data would reduce the cost of human review by possibly 50 percent.

Mr. Terrell explained that Inittime.txt is included in all programs because they expected that programs were being compiled with different compiler options. In particular, some of them should be compiled with optimize. Others were expected to be compiled with different suppression pragmas. If those are changed, the execution time of the null loop may also be changed, meaning that a different set of Inittime.txt values are needed to get accurate timing.

It is possible to perform Inittime.txt for whatever set of compiler variations is desired. But it must be done very carefully to be done accurately; otherwise, it could result in inaccurate timing with no indication that the numbers are inaccurate.

Data Analysis Cycle

The data construct package was used to construct Med_Data package. First, Mr. McKee read through the manual to understand FORMAT. The FORMAT program ran the first time with no difficulties. Next, the Med_Data_Constructor program was run, but Median would not work without the Median package. However, the Med_Data_Constructor program failed to compile because it is one large library unit with at least six nested subprograms. After several tries at attempting to adapt it, Mr. McKee could not solve the problem and the compilation did not work. However, he believes the problem is solvable and recommends that the package be separated from Med_Data_Constructor into one file, but with a series of library units.

Conclusions

Mr. McKee only completed the performance testing phase. The Library, Diagnostic, and Debugger assessors still need to be assessed. Also, the analysis programs (Med_Data_Constructor, MEDIAN, and Single Systems Analysis (SSA)) need to be compiled and executed.

Large programs should be a class of testing, and not a prerequisite for success. Most of the failures occurred because the programs were too large, and he suggests resolving this issue for Version 3.0 by examining ways to proceed if large programs fail.

The ACEC product is generally useful and worthwhile for performance testing and very useful for finding compiler bugs. Areas that need to be improved are the product organization, the user interface, and the process involved in setting up and doing the setup analysis.

Through running the performance testing process, Mr. McKee found four or five serious program problems, only one of which did not have a workaround. Out of slightly under 20 megabytes of program that ran and produced results, there was only one difficult problem and three or four workaround problems. This indicates that the ACVC is a useful product and that the compilers are very healthy. It also indicates that Boeing is pushing the far edge of compiler technology by producing tests that are very difficult for any compiler to pass. Mr. McKee expressed confidence and faith in the ACVC and believes that the ACEC goes a step further in the right direction. A number of significant improvements are possible with respect to the user interface. The criticisms of Meridian and Boeing in this presentation are minor. Although it is possible that Mr. McKee received better support from Meridian because he was reporting to the Government, he has had an Ada compiler for about two years and Meridian has always responded to him.

Mr. McKee estimated the performance analysis effort takes no less than six weeks from when it is started. If the completion of the data acquistion phase is included, the effort may take nine weeks, while understanding and analyzing the data will take an additional two weeks.

However, for someone who performs the analysis on a regular basis, the work could be done in three weeks with another three weeks on the analysis. The difficult part of this job is understanding what is to be done and getting past the anxiety to be able to move faster.

Because the first encounter with the ACEC involves a good deal of time, anxiety, and frustration, Mr. McKee recommends investing in the development of a good training guide to assist users, rather than making available a hot line of experts.

Suggestions

1. Re-evaluate the Inittime.txt strategy.

2. Document the 16-bit storage problem in a one- or two-page letter to Version 2.0 users, and fix the problem with 'ADDRESS values.
3. Redesign the Med_Data_Constructor to be MODULAR.

4. Identify the resource requirements for the test, especially disk resources.

5. Identify and isolate the interactive IO programs.

6. Provide more support and analysis during the setup phase.

Suggestions 1 and 2 need to be addressed to ensure a credible product. The remaining four suggestions increase the product's flexibility, but do not change the credibility. Suggestion 6 is a large step toward user friendliness because it reduces anxiety, and it puts more analysis in the front end with qmore description and explanation of how to use that analysis. These provide strong benefits in making the product user friendly.

Three of the six suggestions pertain to documentation which reinforces its importance. This is a key example of documentation being as much of a problem as the actual technology.

Mr. McKee concluded his presentation by stating that he hoped to have the assessor analysis completed by September, allowing the group to move forward at that time.

3.0 WORKING GROUP STATUS REPORTS

3.1 Ada Compiler Evaluation Capability Working Group (ACECWG) Status Report

Attendees:

Nelson Weiderman, Chair
Sam Ashby
Barbara Decker-Lindsay
Dan Eilers
Jay Ferguson
Greg Gicca
Alan Impicciche
Gary McKee
Ray Szymanski
Kermit Terrell

Deliverables Due This Quarter: None.

Accomplishments:

- Gary McKee gave a presentation on his experience running the ACEC, Version 2.0 against his Meridian Ada compiler.
- Sam Ashby gave a status report on ACEC during the general session.
- Kermit Terrell gave a short report on the status of all SPRs submitted for Version 1.0. Discussed the maintenance of Version 1.0 in light of the new versions.
- Discussed at some length Gary McKee's suggestions for Version 3.0 based on his experience with Version 2.0.
- Spent some time discussing the recommendations that relate to ACEC for the final report.
- Reviewed the ACEC contract status. Version 2.0 of the ACEC will be shipped in June. Kickoff for Version 3.0 will be the 2nd and 3rd of July. PDR will be held around the middle of August. No commitments to funding beyond 1 October have been received. John Solomond is to make a decision on QTS in the next four months.
- Assignments were made to work on draft recommendations and the Chairman agreed to draft the ACECWG section of the final report.
- Meeting time was cut short by plenary discussions of the Team final report.

Key Issues:

- Further advice needed by the Government on ACEC Version 3.0 and Version 4.0 after the Team disbands.
- Funding for Versions 3.0 and 4.0 of the ACEC.
- Status and implementation of QTS.
- ACECWG final report and recommendations.
- Training materials for ACEC.

Projected Work:

- Review of high level design work on Version 3.0 by Boeing.
- ACECWG final report.

Deliverables Due Next Quarter: None.

Presentations Planned: None.

Other Significant Information: None.

Action Items:

- Nelson Weiderman: Put status report on the Net.
  Nelson Weiderman: Put paragraph for E&Ving News on the Net for Liz Keane.
- Nelson Weiderman: Put drafts for recommendation 3 (develop and refine the technology) on the Net.
- Nelson Weiderman: Put draft for recommendation 7 (international availability) on the Net.

- Nelson Weiderman: Put draft for the working group final report on the Net.
- Working Group: Respond to Compile Systems Assessors appendix.

3.2 CAIS Implementation Validation Capability Working Group (CIVCWG) Status Report

Attendees:

Gary McKee, Chair
Karyl Adams
David Baker
Jack Foidl
Kurt Gutzmann
Tim Lindquist
Robert Munck
David Remkes
Ray Szymanski
Jerry Thomas
Kermit Terrell

Deliverables Due This Quarter: None.

Accomplishments:

- CIVC contract status review (with respect to the Beta Test Suite, release data).
- Discussion on Hypermedia Trade Study.
- Discussion on alternate methodologies that may influence the CIVC-A (in particular, spiral model of the software development and software repository methods).
- Discussion of the Test Selection Criteria Trade Study and the use of Prolog to automate the process of analysis.

Key Issues: None.

Projected Work:

- Review PDR package to be received from SofTech (24 August 1990).
- Development of the recommendations and CIVCWG section for the final report.

Deliverables Due Next Quarter: None.

Presentations Planned: None.

Other Significant Information: None.

Action Items:

- SofTech: PDR package to be mailed by 24 August 1990 to Tim Lindquist and Gary McKee.

- Gary McKee: Copy of the IDA Tool Usage Study to Karyl Adams and Kurt Gutzmann.
- Jack Foidl: A courtesy item to send a copy of the actual tool usage numbers to SofTech to facilitate test selection analysis.

3.3 Classification Working Group (CLASSWG) Status Report

Attendees:

Peter Clark, Acting Chair
Captain Rebecca Abraham
Bard Crawford
Major Patricia Lawlis
Patrick Maher
Ronnie Martin (via Fed Ex)

Deliverables Due This Quarter: None.

Accomplishements:

Reference Manual Enhancements - The following action items were completed:

- Ronnie Martin: Proposed changes to several testing-related definitions, Chapter 7.
- Peter Clark: Drafted definition and factors for new attribute vendor support, Chapter 6.
- Bard Crawford: Discussed solution to Figure x-1, x>=4 in the Reference Manual. E-R diagrams will appear in the Version 3.0 draft.
- Bard Crawford: Drafted new section on the Whole-APSE functions, Chapter 3.
- Peter Clark: Discussed descriptions of life cycle activities from 2167A, no descriptions to extract, Chapter 4.
- Peter Clark: Drafted new section on Maintenance Tools, Chapter 5.
- Peter Clark: Renamed and revised section on Document Accessibility to Documentation Quality, Chapter 6.
- Peter Clark: Discussed new functins based on the maintenance tools and the requirements/design work for the REQWG Tools & Aids appendix, Chapter 7.

Guidebook Enhancements - The following actions items were completed:

- Ronnie Martin: Revised the Configuration Management Checklist, Chapter 10.
- Ronnie Martin: Revised the e-mail checklist, Chapter 15.
- Ronnie Martin: Drafted the new sections based on Lyon's book, Chapters 6, 7, 10, 13, and 99.
- Fred Francl: Drafted Distributed APSEs Security Issues Checklist, Chapter 12?.
- Peter Clark: Conducted a phone survey of existing Evaluation Services, Chapter 9.
- Bard Crawford: Drafted the new sections on the PIWG Ada Letters Special Issue, Chapters 4, 5, and 11.

- Bard Crawford: Drafted new sections on the STSC/STEM project, Chapters 4, 7, 9, and 99.
- Greg Gicca: Updated the cross-reference to the compilation assessors, Chapter 5.
- Nelson Weiderman: Updated the cross-references to AES and SEI compilation assessors, Chapter 5.
- Peter Clark: Discussed new assessors from AIM and ASCI, Chapter 8.
- Peter Clark: Drafted the new section on the Tool Integrability Characteristics from Brian Nejmeh, Chapter 13.
- Peter Clark: Discussed renaming the Adaptation Assessors, Chapter 14, to something like "Assessors for All Tools," adding all the sections that are independent of tool functions, and moving the new chapter to precede the current Chapter 5.
- Peter Clark: Discussed published tool evaluations and comparisons, which will appear in the later Version 3.0 draft.
- Peter Clark: Talked with Boeing regarding the Debugger and Library Management Capabilities Lists, which are a part of ACEC Version 2.0, Chapter 5.

Other Accomplishments:

- Pat Lawlis: Distributed and discussed sample Change Request form.
- Becky Abraham: Discussed STARS documents for References/Synopses in the Guidebook.
- Working Group: Discussed WG recommendations to team final report.
- Bard Crawford: Dry run of presentation to Ada-Europe Environments Working Group.

Key Issues:

- Further advice needed by Government on Reference System after the Team disbands.
- CLASSWG final report and recommendations.

Project Work:

- Review of Version 3.0 draft of the Reference System.
- Direction for Version 4.0 of the Reference System.
- CLASSWG final report.

Deliverables Due Next Quarter: None.

Presentations Planned:

- Ada-Europe Environments Working Group, 6-14-90, Dublin, Ireland.

Other Significant Information: None.

Action Items:

Carry Over

- Ray Szymanski: Get the Change Request form approved per Becky Abraham's information for inclusion in Version 3.0 of the Reference System.
- Sandi Mulholland: Upgrade functions list for early life cycle activities in the Reference Manual, Chapter 4.
- Ronnie Martin: Contact Tim Lindquist and Gary McKee in reference to adding entries to the Guidebook describing CAIS OD and MITRE Tests.
- Peter Clark: Draft a section on Evaluating User Interface Management Systems based on the SERC Technical Report, Guidebook.
- Peter Clark: Prepare Guidebook entries describing CECOM documents, Guidebook Chapter 9.
- Peter Clark: Split Sandi Mulholland's inputs into the appropriate separate checklists, Guidebook.
- Bard Crawford: Talk to the DEC user of the Reference System about how it has been used and any comments or suggestions for improvements; also talk to Marlene Hazle, Sandi Mulholland, and Greg Gicca.
- Working Group: Search for new opportunities to improve the Guidebook such as new checklists and references to E&V technology.

New

- Peter Clark: Put status report on the Net.
- Ronnie Martin/Peter Clark: Put draft for the working group final report on the Net.
- Bard Crawford: Draft new section(s) on the Integration Attribute(s), Reference Manual Chapter 6.
- Bard Crawford: Check the Guidebook to see if the "disclaimer" is mentioned prominently enough and make recommendations.
- Bard Crawford: Lok into adding labels to the external binding of the Reference Manual and the Guidebook so they are visible on a book shelf.
- Bard Crawford: Combine CAIS and CAIS-A synopses, Guidebook Chapter 4.
- Peter Clark: Recommend to Ray Szymanski that synopses on PCIS and the DoD Software Master Plan be added, Guidebook Chapter 4.
- Peter Clark: Draft synopsis for IRDS, Guidebook Chapter 4.
- Peter Clark: Track framework efforts and add synopses as appropriate NIST, STARS, NASA, ECMA, etc., Guidebook Chapter 4.
- Peter Clark: Revise the ACEC section and have Boeing/ Ray Szymanski review it, Guidebook Chapter 5.
- Peter Clark: Draft sections on ACPS and PQAC (see Nelson Weiderman) and drop the section on CAP32, Guidebook Chapter 5.
- Peter Clark: Draft section for Nissen's book, Guidebook Chapter 5.
- Peter Clark: Put together new chapter on Assessor for All Tools (see above) and send the new Guidebook organization out on the Net.

- Peter Clark: Draft the new sections for the AIM Benchmars and the ACSI product, Guidebook Chapter 8.
- Peter Clark: Draft the new sections for the IEEE and ACM SIGSoft CASE tool evaluations, Guidebook Chapter 9.
- Bard Crawford: Review vendor/agent components for consistency and make a recommendation, Guidebook Chapters 5 through 99.
- Peter Clark: Change Guidebook Chapter 99 to be the last chapter.
- Peter Clark: Contact Sandi Mulholland regarding the availability of the CATALYST document, Guidebook Chapter 99.
- Peter Clark: Draft new sections for published tool evaluations and comparisons, Guidebook.
- Working Group: Review the working group final report over the Net.
- Working Group: Review Version 3.0 draft of the Reference System.
- Working Group: Review the Change Request form over the Net.

3.4 Requirements Working Group (REQWG) Status Report

Attendees:

Major Patricia Lawlis, Chair
Captain Rebecca Abraham
Sam Ashby
Peter Clark
Bard Crawford
Barbara Decker-Lindsay
Dan Eilers
Jay Ferguson
Greg Gicca
Alan Impicciche
Patrick Maher
Sandi Mulholland
Karla Nohalty
Nelson Weiderman
Lisa Antolini, Recorder

Deliverables Due This Quarter:

- Requirements Document, Version 2.1

Accomplishments:

- Considerable work on the E&V Final Report: final organization; draft recommendations; and assignment of responsibilities for completion.

Key Issues:

- Tools & Aids Document.
- Final Report.

Project Work:

- Three drafts of the Final Report by the next meeting.
- Final draft of the Tools & Aids Document by the next meeting.
- Final version of E&Ving News ready by the next meeting.

Deliverables Due Next Quarter: None.

Presentations Planned: None.

Other Significant Information: None.

Action Items:

- Becky Abraham: Provide a draft of the Final Report section 5.1.
- Karyl Adams: Provide a draft of the Final Report section 5.2.
- Nelson Weiderman/Gary McKee/Bard Crawford: Provide a draft of the Final Report section 5.3.
- Jay Ferguson: Provide a draft of the Final Report section 5.4.
- Alan Impicciche: Provide a draft of the Final Report section 5.5.
- Jay Ferguson: Provide a draft of the Final Report section 5.6.
- Nelson Weiderman: Provide a draft of the Final Report section 5.7.
- Tim Lindquist: Provide a draft of the Final Report section 5.8.
- Bard Crawford: Provide a draft of the Final Report section 5.9.
- Gary McKee: Provide a draft of the Final Report section 5.10.
- Becky Abraham: Incorporate all the inputs and put the second draft of the Final Report on the Net by 8 June 1990.
- Team: Comment on the second draft of the Final Report on the Net by 13 July 1990.
- Team: Comment on the third draft of the Final Report on the Net by 17 August 1990.
- Nelson Weiderman: Put a draft of the ACEC blurb for the final version of the E&Ving News on the Net.
- Liz Keane: Bring the draft E&Ving News to the September meeting.
- Ronnie Martin: Put out the latest draft of the Test System assessors appendix for the Tools & Aids on the Net.
- Greg Gicca: Put out the latest draft of the Compilation System assessors appendix for the Tools & Aids on the Net.
- Alan Impicciche/Peter Clark: Put out the latest draft of the Requirements/Design Support Evaluators appendix for the Tools & Aids on the Net.
- Pat Lawlis: Put the appendices together with the updated Tools & Aids Document and put it on the Net by 17 August 1990.
- Becky Abraham: Put out the Reuse Workshop paper on the Net.
- Betty Wills: Send a disk with the Requirements Document to Ray Szymanski, with a 1 June 1990 date.

APPENDIX A

10.0 LIST OF ATTENDEES

Abraham, Captain Rebecca
WRDC/FOIC
WPAFB, OH 45433-6553

Adams, Karyl A.
c.j. kemp systems, inc.
P.O. Box 24363
Huber Heights, OH 45424

Ashby, Sam
Boeing Military Airplane Company
P.O. Box 7730, MSK80-13
Wichita, KS 67277-7730

Baker, David V.
Intermetrics, Inc.
733 concord Avenue
Cambridge, MA 02138

Crawford, Bard
TASC
55 Walkers Brook Drive
Reading, MA 01867

Decker-Lindsey, Barbara
Boeing Military Airplane Company
P.O. Box 7730, MSK80-13
Wichita, KS 67277-7730

Ferguson, Jay
National Security Agency
ATTN T3214
9800 Savage Road
Ft. Meade, MD 20755-6000

Foidl, Jack
TRW Systems Division
RC2/3625
One Rancho Carmel
San Diego, CA 92128

Gicca, Greg
Lockheed-Sanders
NCA1-2232
95 Canal Street
Nashua, NH 03061

Gutzmann, Kurt
SofTech, Inc.
1300 Hercules Drive
Suite 105
Houston, TX 77058

Impicciche, Alan
Naval Avionics Center
NAC Code 825
6000 E. 21st Street
Indianapolis, IN 46219

Lawlis, Major Patricia
AFIT/ENG
WPAFB, OH 45433

Lindquist, Tim
Computer Science and Engineering Dept.
Arizona State University
Tempe, AZ 85287-5406

Maher, Patrick
P.O. Box 1417, M/D H8175
8201 E. McDowell Road
Scottsdale, AZ 85252

Mulholland, Sandi
Rockwell International
MS124-211
400 Collins Road NE
Cedar Rapids, IA 52498

Munck, Robert G.
UNISYS Corporaiton
12010 Sunrise Valley Drive
Reston, VA 22091

Nohalty, Karla
TASC
55 Walkers Brook Drive
Reading, MA 01867

Remkes, David
SofTech, Inc.
1300 Hercules Drive
Suite 105
Houston, TX 77058

Szymanski, Raymond
WRDC/AAAF-3
WPAFB, OH 45433-6553

Thomas, Jerry
Naval Ocean Systems Center
San Diego, CA 92152-5000

Terrell, Kermit
Boeing Military Airplane Company
P.O. Box 7730, MSK80-13
Wichita, KS 67277-7730

Weiderman, Nelson
Software Engineering Institute
Carnegie-Mellon University
Pittsburg, PA 15213