

AD-A034 807

MITRE CORP MCLEAN VA
DOD WEAPON SYSTEMS SOFTWARE ACQUISITION AND MANAGEMENT STUDY. V--ETC(U)
JUN 75 A ASCH, D W KELLIHER, J P LOCHER
MTR-6908-VOL-2

F/G 9/2

UNCLASSIFIED

NL

1 of 2
ADA034807



MTR-6908

Vol. II

D
p. 5.

ADA034807

DOD Weapon Systems Software Acquisition
and Management Study
Volume II
Supporting Material

A. ASCH
D. W. KELLIHER
J. P. LOCHER III
T. CONNORS

JUNE 1975

DISTRIBUTION STATEMENT A
Approved for public release;
Distribution Unlimited

DDC
RECEIVED
JAN 4 1977
B

MITRE

9
MITRE Technical Report

14 MTR-6908 - Vol-2

Vol. II

6
DOD Weapon Systems Software Acquisition
and Management Study.
Volume II.
Supporting Material.

10 A. ASCH

D. W. KELLIHER

John Paul
J. P. LOCHER III

T. CONNORS

11 JUNE 1975

12 143 p.

CONTRACT SPONSOR
CONTRACT NO.
PROJECT NO.
DEPT.

DoD
None
D40C
D-40

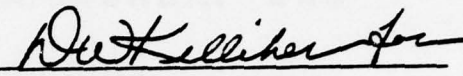
ACQUISITION FOR	
NTIS	Write Section <input checked="" type="checkbox"/>
DOC	Ref Section <input type="checkbox"/>
UNANNOUNCED	<input type="checkbox"/>
JUSTIFICATION	
<i>Letter on file</i>	
BY	
DISTRIBUTION/AVAILABILITY CODES	
Dist. AVAIL. and/or SPECIAL	
A	

THE
MITRE
CORPORATION
McLEAN, VIRGINIA 22101

This document was prepared for authorized distribution.
It has been approved for public release.

402 364
b7g

MITRE Department
and Project Approval:



John Paul Locher III

ABSTRACT

In December, 1974, DoD initiated a two-phase software acquisition study program to identify methods for controlling increasing costs, improving the quality, and minimizing the adverse impact of software in weapon systems. The MITRE Corporation and the Applied Physics Laboratory of Johns Hopkins University were requested to conduct separate, but coordinated, four-month studies in support of the first phase of this study program. Volume I contains the MITRE study findings and recommendations, published in May 1975. This volume provides supporting information for the study findings and recommendations.

ACKNOWLEDGEMENT

The MITRE Corporation would like to acknowledge the assistance received from all participants of this study. Special thanks are extended to the DoD Software Steering Committee for their guidance and coordination, and to the personnel at each of the DoD facilities and within industry visited for their cooperation and assistance in providing information. All organizations of the Office, Secretary of Defense, the Military Departments, and industry visited by the MITRE study team were completely cooperative and dedicated to seeking solutions to existing weapon systems software problems.

TABLE OF CONTENTS

	<u>Page</u>
1. INTRODUCTION	1-1
1.1 Background and Study Goals	1-1
1.2 Study Approach	1-3
1.3 Report Organization	1-4
2. WEAPON SYSTEMS INTERVIEWS	2-1
2.1 Weapon Systems Interviewed	2-1
2.2 Purpose/Goals of Weapon Systems Interviews	2-2
2.3 Weapon System Interviews - Compilation of Meeting Notes	2-5
2.3.1 Army Systems	2-5
2.3.2 Air Force Systems	2-16
2.3.3 Joint Interoperability Testing	2-48
3. SUMMARY OF BASELINE STUDIES AND WORKSHOP PROCEEDINGS	3-1
3.1 CCIP-85, Information Processing/Data Automation Implications of Air Force Command and Control Requirements in the 1980's, Executive Summary (Revised Edition)	3-1
3.2 Electronics X, A Study of Military Electronics with Particular Reference to Cost and Reliability, Volume 2: Complete Report	3-13
3.3 Automatic Data Processing Costs in the Defense Department	3-16
3.4 The High Cost of Software (Proceedings of a Symposium)	3-21
3.5 Government/Industry Software Sizing and Costing Workshop	3-32
3.6 Proceedings of the Aeronautical Systems Software Workshop	3-37
3.7 Project PACER FLASH, Volume 1, Executive Summary and Final Report	3-48
3.8 Tactical Computer Software Acquisition and Maintenance	3-50
3.9 A Report on Air Force Logistics Command Operation Flight Program Support: Introduction and Summary	3-54
3.10 Report of Army Scientific Advisory Panel Ad Hoc Committee for Army Tactical Data System Software Development	3-59

TABLE OF CONTENTS

(Continued)

	<u>Page</u>
3.11 An Analysis of Computer Software Management Requirements for Operationally Deployable Systems, Executive Summary (Volume I)	3-62
4. SOFTWARE ACQUISITION AND MANAGEMENT BIBLIOGRAPHY	4-1
4.1 Baseline Studies and MITRE Developed Bibliography	4-1
4.2 Software Steering Committee Suggested Publications	4-7
4.2.1 OASD (Comptroller, Management Systems)	4-7
4.2.2 IDA, Science and Technology Division	4-8
4.2.3 Army	4-8
4.2.4 Navy	4-9
4.2.5 Air Force	4-9
APPENDIX A: DOD SOFTWARE ACQUISITION STUDY REQUESTED INFORMATION	A-1
APPENDIX B: DISTRIBUTION LIST	B-1

LIST OF ILLUSTRATIONS

	<u>Page</u>
TABLE 2-1: OVERVIEW OF SYSTEMS INTERVIEWED	2-3
TABLE 2-2: B-1 DIGITAL COMPUTERS	2-32
TABLE 3-1: ESTIMATED DOD SOFTWARE COSTS, COMPUTER SYSTEMS UNREPORTED IN GSA INVENTORY, FY 1973	3-20
TABLE 3-2: SOFTWARE EFFORT DISTRIBUTION BY ACTIVITY	3-23
TABLE 3-3: RECOMMENDATIONS OF WORKSHOP 1: UNDERSTANDING THE SOFTWARE PROBLEM	3-25
TABLE 3-4: RECOMMENDATIONS OF WORKSHOP 2: SEMANTICS OF LANGUAGES AND SYSTEMS	3-26
TABLE 3-5: RECOMMENDATIONS OF WORKSHOP 3: PROGRAMMING METHODOLOGY	3-27
TABLE 3-6: RECOMMENDATIONS OF WORKSHOP 4: SOFTWARE-RELATED ADVANCES IN COMPUTER HARDWARE	3-28
TABLE 3-7: RECOMMENDATIONS OF WORKSHOP 5: PROBLEMS OF LARGE SYSTEMS	3-30
TABLE 3-8: FACTORS INFLUENCING SOFTWARE COST	3-35
TABLE 3-9: COST ESTIMATION METHODS	3-36
TABLE 3-10: FIGURES OF MERIT	3-57
TABLE A-1: SOFTWARE COST WORKSHEET	A-7

1. INTRODUCTION

This Volume II of the DoD Weapon System Software Acquisition and Management Study report presents information which supports the study findings and recommended actions included in Volume I, which was published as MTR-6908, Volume I, May 1975, MITRE Findings and Recommendations. The background and study goals, study approach and other introductory information is repeated here from the Introduction to Volume I.

1.1 Background and Study Goals

Modern day weapon systems are making extensive use of computers and software¹ to perform many combat and other functions which were formerly performed manually, by hardware, or were not able to be performed at all prior to the advent of computer technology. The life cycle cost of acquiring and owning the computer software is becoming significant in relation to the other costs of system acquisition. Also, since the software performs many functions which are critical to overall weapon system mission performance, software is steadily becoming more important.

The importance being placed on weapon system software acquisition and management by the Department of Defense (DoD) is reflected by the number of recent management and technical papers, and committees/panels either sponsored by the DoD or participated in directly by DoD personnel. Several trends in the use of software and stored program computers in the design, development, operation and support of weapon systems are becoming increasingly evident. The trends are characterized by:

1. Growing DoD management awareness of the increasing frequency in the use of and increasing mission dependence on software in military weapon systems.
2. Accompanying suspicion that the costs of software are an increasingly significant portion of DoD costs for weapon

¹In this report, the term software is used to refer to computer programs, associated data bases, and related documentation required to define, design, develop, produce, test, operate, and maintain the software-related aspects of the total weapon system, including computer hardware, software, personnel and procedures. A list of definitions for common terms used throughout this report is included in Appendix A of Volume I.

systems and that additional indirect costs can often be attributed to software.

3. Concern by DoD management that present methods and controls for acquiring and maintaining software should be improved upon to reduce risks (e.g., cost, schedule, and performance), to improve the software development and maintenance processes, and to improve the quality and timeliness of software end products.

4. Lack of general understanding of how best to impose software management controls without adding inefficiencies, removing incentive or stifling innovation in the fast changing software management and computer technology areas.

In recognition of the need for a focused and coordinated approach for improving weapon systems software management and technical practices throughout DoD, on 3 December 1974 the Assistant Secretaries of Defense (Comptroller and I&L) and the Director, Defense Research and Engineering (DDR&E) established a joint OSD/Service Weapon System Software Steering Committee. Its charter is to identify critical weapon systems management problems and recommend policies and instruments for their solution. In support of the first phase of the Steering Committee activities, The MITRE Corporation and the Applied Physics Laboratory at Johns Hopkins University were requested to conduct separate, but coordinated, four-month studies. Volume I of this report provides the MITRE study findings and recommendations. Volume II provides supporting materials and analyses collected during the study and used in the development of the study recommendations.

The goals for the first phase of the study were briefly defined in a 3 December 1974 OSD memorandum, which also established the DoD Software Steering Committee. A copy of the memorandum is included as Appendix B to Volume I. These goals are repeated here, with elaboration to indicate the full scope of the MITRE study. In each goal, the effort was to identify and define:

1. The nature of the critical software problems facing the DoD. This required the identification of the critical weapon system software management and software technical problems facing the DoD relative to improving software acquisition and management procedures, to make better use of resources, and to improve software quality and timeliness.
2. The principal factors contributing to the problems. This required the identification of where major software problems

are occurring in the weapon system life cycle acquisition process and their causative factors.

3. The high payoff areas and alternatives available. This required the identification of the software areas where OSD and Service attention will have maximum leverage in controlling costs and improving the utilization of software resources, quality and timeliness, and making recommendations for action programs to achieve these improvements.

4. The management instruments and policies that are needed to define and bound the functions, responsibilities and and mission areas of weapon systems software management. This has resulted in MITRE-recommended DoD management instruments/policies which are needed to implement an action program to resolve problems in the high payoff areas.

The scope of the study considered all system life cycle phases and all types of software associated with the definition, design, development, test and evaluation, production, operation and maintenance of weapon systems. The term "weapon system" could not be precisely defined. However, the DoD Software Steering Committee provided a list of Army, Navy and Air Force systems for review which tended to bound the study. These reviews excluded intelligence and the ADP (Automatic Data Processing) categories except where ADP software was used in support of a weapon system. They excluded review of the Command and Control and Communications (C³) systems except for 427M which was included in the review list.

1.2 Study Approach

The study was conducted over a four-month period by a team of MITRE staff from Bedford, Massachusetts and McLean, Virginia. The MITRE Corporation emphasized the software practices of the Department of the Air Force and the Department of the Army, while the Applied Physics Laboratory (APL) emphasized systems of the Department of the Navy and the Department of the Army. Information concerning weapon systems software acquisition and management practices in the DoD was obtained from the following sources during the study:

1. Review of recent DoD software study reports and workshop proceedings and discussions with selected authors of these reports.¹

¹The reports and workshop proceedings are summarized in Section 3 of this volume.

2. Preparation of a weapon systems software questionnaire oriented towards identifying major areas of needed software cost, quality, and schedule improvement, and the use of this questionnaire in discussions with Air Force and Army project personnel on 14 DoD weapon systems.¹ Similar interviews were also conducted between APL and the Navy.
3. Discussions with DoD staff personnel at Service headquarters and command levels who are concerned with establishing DoD weapon system software acquisition, management, and R&D policies.
4. Review of major DoD Regulations and Standards most frequently used in the procurement of software in weapon systems.
5. Interchange of interim findings and ideas with the APL study team.
6. Guidance received from the DoD Software Steering Committee during periodic progress reviews, and from interaction with members of the committee.
7. Soliciting opinions of MITRE technical and management personnel with experience in weapon systems, software acquisition, and DoD practices.

While time and resources did not permit all sources to be examined fully, sufficient correlation existed between data sources to confidently draw conclusions as to the weapon systems software problem areas facing the DoD, their causative factors, and the high payoff areas of needed DoD action. These conclusions and identified high payoff areas were used as the basis for developing recommended DoD actions.

1.3 Report Organization

This study report is presented in two volumes. Volume I contains a summary of findings and recommended corrective actions of direct management interest. Volume II contains further supporting material.

¹A list of the weapon systems interviewed by MITRE is included in Section 2, Volume II, of this report. A list of study participants is included in Appendix E, Volume I, of this report.

Volume I is organized into four major sections: Section 1 contains introductory information including the purpose and goals of the study; Section 2 contains a summary of major findings and concludes with a list of high payoff areas for DoD action; Section 3 presents MITRE's recommended DoD actions in the areas of software performance specification, software acquisition planning, software technology, and personnel; and Section 4 includes a brief outline for implementing the recommended actions during Phase II of the study. Appendices to Volume I are limited only to that information required to understand the content of the four major sections and to initiate the recommended actions.

Volume II is organized into four major sections: Section 1 contains introductory information; Section 2 provides further detail on the 14 systems interviewed; Section 3 contains a brief summation of the major findings and recommendations of the reports and workshops reviewed during the study; and Section 4 includes a software acquisition and management bibliography. Appendix A is a copy of the questionnaire used by the MITRE team during the weapon systems interviews.

2. WEAPON SYSTEMS INTERVIEWS

This section presents summary information for each of the weapon systems interviewed. The information is presented under three subjects, including:

- The weapon systems interviewed (Section 2.1).
- The purpose and goals of the interviews and the approach used in conducting the interviews (Section 2.2).
- An overview of each weapon system, including organization interviewed, brief description of the weapon systems functions, system status, scope of software, software cost data, and general observations and lessons/learned (Section 2.3).

2.1 Weapon Systems Interviewed

The list of weapon systems project offices to be interviewed for the study was provided by the joint OSD/Service Software Steering Committee. This list was selected for the following reasons:

- To represent a cross section of different types of weapon systems with significant software (operational and support).
- To represent a mix of Service Commands (to provide information on the software acquisition and management practices used by different Service Commands).
- To provide an opportunity to study weapon systems which have become operational (have accumulated operational experience) as well as several newer systems which reflect the most recent procurement practices and industry developed design and management methods.
- To provide a cross section of systems which have experienced both successful software acquisition as well as those which have encountered software acquisition problems.

There were five systems of the Army and nine of the Air Force interviewed. In addition, the Joint Integration Test Facility (JITF) at San Diego was visited. The JITF is responsible for joint interoperability of Service-developed systems performing mission roles of tactical air control and tactical air defense (TACS/TADS). The systems interviewed are as follows:

<u>ARMY</u>	<u>AIR FORCE</u>		<u>JITF</u>
TACFIRE	DSP	485L	TACS/TADS
TSQ-73	MINUTEMAN	427M	
PERSHING	F-111	COMBAT GRANDE	
SAM-D	WILD WEASEL	AWACS	
SAFEGUARD	B-1		

Table 2-1 is a further listing of the weapon systems indicating mission/role, status and organizations interviewed. This is identical to the information included in Table D-1 of Volume I, but it is also included here for the convenience of the reader.

2.2 Purpose/Goals of Weapon Systems Interviews

The weapon system interviews provided substance and verification to the weapon systems problems identified by the 11 baseline studies and workshop proceedings (Reference Section 3 of Volume II). They were also an excellent vehicle to obtain OSD's and the Service's perspective of problems being encountered in software acquisition. Good acquisition practices were also observed and noted on some weapon systems.

The reviews were conducted with software management and engineering personnel of the Army and Air Force responsible for managing the development, testing and transitioning of the software. Prior to each visit, a questionnaire was sent to the system management offices to provide an indication of the information required for the study. The questions were developed/oriented toward meeting the study goals. A copy of the questionnaire is included as Appendix A to Volume II.

The goals which MITRE hoped to achieve through the interviews were as follows:

- To identify major weapon systems software costs (i.e., where are software dollars being spent?)

TABLE 2-1
OVERVIEW OF SYSTEMS INTERVIEWED

<u>NAME</u>	<u>MISSION/ROLE</u>	<u>STATUS</u>	<u>INTERVIEWS</u>
<u>AVIONICS</u>			
B1	STRATEGIC BOMBER	DEVELOPMENT	ASD SPO, MAR 7.
WILD WEASEL	AIRBORNE SAM DEFENSE	LIMITED PRODUCTION/UPDATE	ASD SPO, MAR 7.
F-111 (B,D,F)	STRATEGIC/TACTICAL BOMBER	DEPLOYED/UPDATE	SACRAMENTO ALC, FEB 27.
<u>MISSILES & GROUND SUPPORT</u>			
SAFEGUARD	MISSILE DEFENSE	TRANSITIONING	BMDPO, JAN 9; BMDSC, FEB 7.
PERSHING	TACTICAL MISSILE	DEPLOYED/UPDATE	MISSILE COMMAND PMO, FEB 6.
SAM-D	AIR DEFENSE	DEVELOPMENT	MISSILE COMMAND PMO, FEB 6.
MINUTEMAN	STRATEGIC MISSILE	DEPLOYED/UPDATE	SAMSO SPO, FEB 25.
<u>CONTROL</u>			
TACFIRE	ARTILLERY FIRE SUPPORT	LIMITED PRODUCTION	ARTADS PMO, FEB 4.
AN/TSQ-73	MISSILE MINDER-AIR DEFENSE	LIMITED PRODUCTION	ARTADS PMO, FEB 7.
DSP	INTELLIGENCE	DEPLOYED/UPDATE	SAMSO SPO, FEB 24.
485L	TACTICAL AIR CONTROL	DEVELOPMENT	ESD SPO, MAR 12.
427M	CONTINENTAL AIR DEFENSE, SPACE SURVEILLANCE	DEVELOPMENT	ESD SPO, MAR 12.
COMBAT GRANDE	AIR DEFENSE FOR SPAIN	DEVELOPMENT	ESD SPO, MAR 13.
AWACS	AIRBORNE TACTICAL C&C	PRODUCTION PENDING	ESD SPO, MAR 13.
<u>INTEROPERABILITY</u>			
TACS/TADS	INTEROPERABILITY TESTING	TEST; PREPARING FOR OED	JITF, FEB 26.

- To identify major indirect software costs (i.e., is software contributing to loss of mission effectiveness, schedule delays, shorter mission life?).
- To identify major software acquisition problems and their causative factors (e.g., lack of emphasis in the DoD review process, poor DoD acquisition and procurement practices, or poor industry implementation practices).
- To solicit opinions from Service personnel directly involved with software acquisition regarding their ideas on how DoD methods could be improved upon (i.e., to solicit opinions on alternative solutions).

The following comments are made concerning the MITRE team meeting each objective/goal.

Identify Major Software Costs -- Detailed software cost information was requested at each interview but was generally not available. Software cost data was generally not used by the technical managers as a management tool. Further, current regulations are not clear concerning the collection and maintenance of software costs for weapon systems.

Identify Major Indirect Software Costs -- This could not be accomplished during the relatively short period of time for the study. It was observed, however, that software was generally not considered on the critical path for avionics systems. Although there were slippages and cost overruns in avionics software, total system delays and cost overruns were more attributable to hardware subsystems. This was not generally the case for ground based systems where software often represented a major effort and cost in relation to the total system requirements.

Identify Major Problems and Causative Factors -- The persons interviewed were completely cooperative in presenting problems encountered and lessons learned in weapon system software acquisition. These include poor requirements formulation, lack of understanding of requirements, parallel development of computer hardware and software, inadequate visibility for measuring progress and early identification of problems, poor contracting practices, lack of software development planning, and lack

of adequate testing procedures and resources. The problems and lessons learned are discussed in more detail in Section 2.3

Identify Alternatives and Recommended Actions -- The MITRE team received numerous recommendations for improving DoD weapon systems software acquisition practices and procedures. These include action areas requiring the revision of current DoD publications, the development of better standards and guidance for software acquisition and management, expanded R & D programs, and improving personnel policies and practices. Specific lessons learned resulting from the interviews are discussed in more detail in Section 2.3.

2.3 Weapon System Interviews - Compilation of Meeting Notes

The information received from the weapon system interviews is summarized for each weapon system. They are discussed in the order in which the interviews were conducted; i.e., the Army systems interviews were conducted during February 1975, the Air Force systems during March 1975. The principal participants for each interview are listed in Appendix E of Volume I.

Each interview lasted from 3 to 6 hours. It must, therefore, be recognized that the information about each weapon system does not represent an in-depth study. The information presented here does not include detailed descriptions nor encompass the full scope of the various subjects presented about each weapon system. Also, the information was compiled from rough notes taken during the interviews and was not later validated due to the lack of time and resources. It is, therefore, subject to errors and misinterpretations. However, it is useful for future Phase II activities that these rough notes be documented.

2.3.1 Army Systems

2.3.1.1 TACFIRE

Date of Interview: February 4, 1975

Organization: ARTADS PMO
Fort Monmouth, New Jersey
(Note: Discussion on TACFIRE was also conducted with the Army's Computer Systems Command during December 1974).

System Functions

TACFIRE is a tactical command and control system for field artillery warfare. It includes a computer system which stores information concerning targets, terrain characteristics, weapons, weather and other battlefield information. It has an X-Y plotter capability and CRT displays. Results of system processing suggests the use of available weapons against targets and computes the ballistics information. The fire control centers review the computer outputs and makes the final decisions on employments of weapons.

System Status

The functional requirements (QMR) were developed in 1966. The first article (engineering test system) was to be delivered in October 1969; actual delivery to Ft. Sill was April 1971. After delivery, the user could not accept the system because of checkout problems. The computer hardware memory had to be expanded, and additional software programming and testing had to be accomplished to bring the system to an acceptable level. The system reached a stage of development and confidence where a low rate of initial production ARSAC decision was made in January 1975. A DSARC decision will be needed for entering full scale production, which is planned for 1977. The prime contractor on this system has been Litton Industries.

Scope of Software

The computing equipment was developed by Litton Industries (L-3050). (This computer is also used in TSQ-73, with an expanded memory). Most of the computing power is used for assessing the battle situation, and computing the fire plan. The operating system must be "super smart" to keep account of the processing.

The size of the software programs was estimated at approximately 260K of instructions. All software had to be developed for the system since the computer hardware (militarized) was also a new development. This included an operating system, compiler, diagnostics, test software, training software, utilities, and the operational programs. Litton continues to be under contract to complete the software.

Software Cost Data

Cost information on software was not readily available. It was stated that the original cost estimates were grossly underestimated;

the estimate may have been \$3 million. The costs are probably running close to \$30 million by now. The original package for total system procurement (total package procurement) was for \$120 million.

Observations/Lessons Learned

The observations/lessons learned from the TACFIRE experiences are summarized as follows:

- Requirements were not well developed, and not well understood by the user and the contractor.
- Requirements change during development; program managers should try to get user agreements very early in the process.
- Developing software and hardware together caused an imbalance in the use of resources, and design and debugging problems.
- Current policies, regulations, etc. were not much help at line level in getting contracts started right; there is a need for comprehensive software management standards and procedures.
- Some of the major software problems and delays were caused from the software not performing the functions that management thought were intended. The moral of this is that one cannot expect the contractor's systems analysts to be experienced artillerymen.
- There did not exist good tools to validate software with hardware.
- Software management was accomplished largely by hardware project management personnel. Software visibility was not adequate for the Army until it acquired and operated its own testing facility.
- There was Government emphasis to ship the system when the hardware was ready, although the software was not completed; software should be "bought" at the plant.
- There is no known software management forum for exchanging common experiences/problems/good practices across DoD.

2.3.1.2 PERSHING

Date of Interview: February 6, 1975

Organization: Pershing PMO
Army Missile Command
Redstone Arsenal
Huntsville, Alabama

System Functions

Pershing is an Army deployed tactical surface-to-surface medium range missile.

System Status

There have been three versions deployed:

Pershing I - analog and relay controlled, about 1960.

Pershing I (update) - with digital computer ground control, about 1965.

Pershing IA - all digital ground and airborne.

The analog equipment was replaced with digital computers to improve the ground diagnostics for countdown and maintenance, and to provide more flexibility. A new version, Pershing II, is currently in the early validation phase (has passed DSARC I).

The prime contractor has been Martin Marietta, Orlando, Florida from the beginning of the program (pre-1960). The Army relies heavily on Martin for many aspects for Pershing, including software maintenance support.

Scope of Software

The current computer hardware for the ground based portion is a Burroughs special purpose computer with 20K memory; there are approximately 18K stored instructions for the ground control functions. The airborne computer was built by Bendix; it has 4K, 16 bit words. There are approximately 3,500 computer instructions for the airborne functions. Assembly language is used for programming.

Software Cost Data

Software cost information was not available. Indications were that costs for software programming and computer time could

probably be provided by working with the contractor, but software costs for the design and integration testing functions would be very difficult, if not impossible, to obtain.

Observations/Lessons Learned

Generally, the software aspects of this system did not present significant problems in relation to the total system development. The MITRE team's impression was that software was not on the critical path during development; the personnel interviewed could not recall where software problems delayed the program. Nevertheless, there were some observations and lessons learned through this development.

- The Army had some visibility over software development, but the contractor maintained primary control. Impressions were that perhaps too many "frills" were added to the software programs. After the Army became more involved in testing, it gained more visibility into software development and it became easier to work with the contractor.
- The Army exercises close configuration control over operational software, but not over the support and diagnostic software, which is controlled by the contractor.
- Military Standard 490 was used as guidance by the Army in writing contracts. It provided some assistance in documentation standards, but it is too general in most areas. It was stated that there is no good "cookbook" for configuration management and other standards for software.
- An incentive type contract is best for software development, but the Army should have its own capability to verify contractor's work.
- It is difficult to establish a firm schedule for a development program for software.
- Software aspects of a system should not be segregated during the early design.
- The Army did not find computer simulation too useful in identifying hardware/software interface problems on Pershing I. Problems were not found until test firings were accomplished.

2.3.1.3 SAM-D

Date of Interview: February 6, 1975

Organization: SAM-D PMO
Army Missile Command
Redstone Arsenal
Huntsville, Alabama

System Functions

SAM-D is a surface-to-air defensive missile. The system has both airborne and ground based computers. The ground based computer (dual CPU for multiprocessing) receives radar information, makes decisions for weapon assignments and can cause the weapons to automatically fire, or present battle information to battery control. The guidance equations are in the ground based computers, and the guidance control programs are in the airborne computer. The system is mobile, but it must be initialized each time it is moved. It is designed to handle up to 100 incoming aircraft.

System Status

The project was started in 1967, with Raytheon as the prime contractor. The system is presently undergoing V&V, integration testing, and demonstrations; it is between the DSARC II and III decision point. A special DSARC is planned by OSD before a production release will be made.

Scope of Software

The ground based and airborne computers are special purpose computers developed by Raytheon. They are militarized and designed to be carried on a 5-ton truck. The project experienced the usual problems associated with parallel development of computer hardware and software, i.e., hardware was not available for checking software, hardware/software interface problems, and problems with debugging the total system. The operational software (not including initialization) was estimated at 160,000 instructions. The total size (operational, initialization, diagnostics, etc.) was estimated at approximately 250,000 instructions. However, it was noted that an accurate word count including instructions and data was not readily available. The UNIVAC-1108 was used for simulation work, with JOVIAL used as the compiler. Indications were that simulation is very important for V&V on this system to avoid spending large resources (fly aircraft and missiles) for certain testing.

Software Cost Data

Software costs were kept as a single line item. However, they do not represent all software costs, i.e., software related costs that contribute to system integration testing. (The software costs that may be available were not provided to the MITRE team.) It was indicated that meaningful cost information cannot be collected until common definitions for software costs across Systems and Services are developed. Indications were that Raytheon had about 200 people working on software and requirements definition, and there are two subcontracts with IBM to conduct requirements and software analysis, and to provide system engineering support (since 1972).

Lessons Learned

- Documentation arrives from the contractor "100 pounds at a time". The program management people indicated an impossibility to read and digest all of it.
- The government should ensure that computers and software tools are available at the time of contracting.
- There is a need for a family of software related specifications/standards for use in writing contracts. There can be no real cure for software problems if contracts do not adequately define the software efforts.
- There is a need for an early development plan for software. The plan for SAM-D was very late; a plan from the contractor was not received until the project ran several years. Also, the start-up time and resources required were underestimated.
- There is a need for a separate, Service controlled, V&V capability, and it should be available early during the development.
- It takes too long to develop, produce, and deploy weapon systems (most takes 10 years or longer). Requirements and users change over the development cycle which causes expensive and timely redesign efforts.
- Top level management did not always have a sufficient understanding of software.

2.3.1.4 AN/TSQ-73

Date of Interview: February 7, 1975

Organization: ARTADS PMO
Redstone Arsenal
Huntsville, Alabama

System Functions

The TSQ-73 Missile Minder is an Army air defense control and coordination system designed to coordinate actions of surface-to-air weapons against hostile air targets. It serves as an Army air defense command post.

System Status

The TSQ-73 is not considered a major system so it does not come under DSARC review procedures. The program was started in 1968; Litton, the prime contractor, was awarded a development/production contract in 1970. In June 1974, a limited production release decision was reached. A number of prototype systems have been delivered, with one representing the Army's entry into the TACS/TADS interoperability program.

Scope of Software

Each system uses two L-3050 computers in multiprocessing configuration. These computers were Litton developed for the Army's TACFIRE system. The software is broken down as follows:

Operational	20,000	Instructions
Simulation	5,000	"
Diagnostics (Fault detection and isolation)	30,000	"
Support	<u>40,000</u>	"
Total	95,000	Instructions

Software Cost Data

The cost of software was not provided as a separate item. However, system costs were discussed. While the program has been successful, it has not been without prime contractor cost problems. The program started with an estimate of \$8.5 million for R&D, and \$.7 million for each production system. The R&D costs grew to approximately \$20 million; and each production

system is expected to cost \$1 million. Perhaps the main reason for the cost overrun was from underestimating in bidding. (There were five bidders; Litton was low bidder.) Testing costs are often underestimated, e.g., PMO personnel indicated that operational testing would cost on the order of \$4 million.

Observations/Lessons Learned

- When contracts get into dollar problems, some software areas (those not tied down in specifications and documentation) suffer most. Delivery of support software and associated documentation fall within this category.
- More freedom is needed in contractor selection to avoid the "buy in" syndrome. Also, maintain industry competition as long as possible.
- Guidance for software acquisition management is needed; both in the areas of design approach and software management.
- The DSARC process encourages getting to the validation phase as early as possible without doing sufficient front-end software work. The software problems then tend to slide to the end of the project.
- Consider maintenance support requirements early in the acquisition process.
- Develop good software configuration management early in the acquisition process.
- Tie down the user as early as possible; there are major costs associated with changes to requirements.
- Identify documentation standards, software deliverables (including support and maintenance) etc. in the contract. Although there are military standards for documentation, better guidance is needed for software documentation.
- Provide an early in-house software monitoring and testing capability. Perhaps a separate QA contractor is needed. Accomplish as much QA in the plant as possible before field testing and delivery. Many problems were uncovered during field testing after testing had been accomplished at the Litton facility.

2.3.1.5 SAFEGUARD

Date of Interview: January 9, 1975 (BMDPO)
February 7, 1975 (BMDSC)

Organization: BMDPO, Rosslyn, Virginia
BMDSC, Huntsville, Alabama

System Functions

The mission of the SAFEGUARD ballistic missile defense system is to preserve a second strike capability in the event of a missile attack, whether willful or accidental, on the Continental United States.

System Status

Ballistic missile defense started in the mid-fifties with NIKE-ZEUS, followed by NIKE-X, and SENTINEL. With a decision to deploy only one system (stemming from the SALT agreements), SAFEGUARD transitions in March 1975. The Bell Telephone Laboratories (BTL) has been the prime contractor from the beginning. Some subcontracts for software had been given to IBM. The development efforts have basically been on schedule since 1971. Achieving this, however, can be primarily attributed to gaining experience and resolving very complex technical problems through development work on the systems prior to SAFEGUARD.

Scope of Software

The computer hardware was developed by BTL. Early studies indicated that dual processing capabilities would be needed to meet the very high speeds demanded to meet system requirements. Multiple CPUs (as many as 10) are used and each CPU can address any core memory bank. Several design support computers were used, e.g., the IBM 360-65 and IBM 370 systems.

The software for this system is very complex, and must meet very high system requirements in terms of speed and quality assurance. It is probably the largest software development effort undertaken by anyone. The total software is estimated at 3,031,000 64-bit words or equivalent. These are broken down as follows:

Tactical (operational)	653,000	Words
Development support	913,000	"
On-site support (main- tenance and diagnostics)	630,000	"
Off-site support (data reduction, simulation, test)	<u>835,000</u>	"
Total	3,031,000	Words

(Assumed to include instructions and data.)

Software Cost Data

The program management personnel were unable to provide detailed software cost breakouts, but expressed a willingness to attempt to reconstruct costs. There were also indications that costs may be available from the contractor and the project office offered to arrange a MITRE team visit with BTL. It was indicated, however, that one needs to define what constitutes the software effort in order to collect meaningful cost information. An order of magnitude for software cost was given at \$467 million for the period from 1968 through FY 1976. The total data processing system costs were given as \$596 million, including software

Lessons Learned

- Don't compromise the front-end work which is required for good planning and design of a software system.
- Tie down user requirements early in the process.
- Select a good contractor, one with a good "track" record.
- Err on the high side in determining computer equipment requirements.
- Start a project with a good software acquisition management procedure, one where the government can measure progress. Even with a good procedure, it is difficult to measure progress at the early stages, i.e., real progress measuring is difficult until coding has been completed and tested. The Principal Events Reporting System developed by BTL and used on SAFEGUARD was found to be very successful for monitoring software progress.

- Establish a hardware/software V&V facility as early as possible. A separate V&V contractor is most useful. Use of in-house laboratories/resources as the independent facility is feasible, but resources are not always available when needed, and not always under the direct control of the program manager.
- Do a good job of defining the environment in which the system must operate, testing, etc.; must define what to expect from testing; make certain requirements can be interpreted into a test environment and test scenario.
- Project management must make a judgment on which software problems to bypass and to correct later in order to keep project moving.
- It is very important to validate interfaces early. If necessary, GFE the necessary resources to accomplish this.
- The Services must always be in a position to tell the contractors whether his software is good or bad.
- There is no good way to budget for software costs (development and maintenance). There is a lack of historical cost data one can use for cost planning factors. Give the program manager flexibility in use of funds and hold him accountable for progress.
- The DoD should attempt to standardize interfaces, but should be careful about imposing impractical standards in other areas, e.g., languages and computer hardware.

2.3.2 Air Force Systems

2.3.2.1 DEFENSE SUPPORT PROGRAM (DSP)

Date of Interview: February 24, 1975

Organization: DSP System Program Office
SAMSO
Los Angeles, California

System Functions

The DSP is an intelligence gathering system which uses communications and ground based computers for data transmissions, reduction and analysis. (Features of this system are classified.)

System Status

The R&D work started on DSP in 1961, with major software work starting in the late 60's. A limited operational capability was achieved in 1971, with an expanded capability in December 1973. Subsequent improvements were made by July 1974 which provides the current capability. The system has transitioned where ADC maintains the software and AFLC provides hardware maintenance support. The maintenance agreements with ADC and AFLC were made as early as 1968. ADC was given responsibilities for configuration control. However, SAMSO continues to retain responsibility for adding new functions and/or major updating.

There are two developments underway by SAMSO. One concerns having IBM revise the current software so that the Overseas Ground Site (OGS) and the CONUS Ground Site (CGS) software is identical. The other effort is a new development of a Simplified Processing System (SPS), which will be a transportable version of the CGS. A contract was signed with IBM in December 1974 for development of an SPS prototype, which is due in 1976. After operational use of the SPS, in 1978 a production decision will be made on the basis of the prototype. The user of the SPS will be ADC as it is for the current system.

Scope of Software

The system is made up of IBM 360/75 computers, communications, and related facilities. There are 2 IBM 360/75 computers and one backup located overseas. A Sigma 5 computer for communications is also located overseas. There are 2 IBM 360/75 computers located within the CONUS. All IBM 360/75 computers were off-the-shelf.

There were separate software programs developed for the overseas operation and for the CONUS. The major software contractors included IBM, TRW, Aerojet, Philco Ford, and Sandia. In addition, there were other special study contractors. Off-the-shelf software was used wherever possible, e.g., 360 operating system and programming languages. SAMSO accomplished the system integration because of the complexity of hardware, software, communications and facilities. They had assistance

for subintegration, e.g., Philco Ford was responsible for communications integration, TRW for hardware.

Software was usually on the critical path for this system (always on a "short fuse" schedule). Some slippages occurred (9 months on a major update), but generally, performance in meeting schedules was quite good. There is a large Aerospace Corporation involvement (30 to 40 people) in DSP software to assist the SPO in system management and validation.

The IBM Corporation is the major contractor for the software on the SPS, the mobile system under development. The contract requires IBM to use the latest techniques for the software work, e.g., top-down design, structured programming, and management control techniques.

Software Cost Data

Software cost information was not readily available. However, some gross cost information was provided during the interview. These costs are presented only to provide an order of magnitude, and should not otherwise be used unless they are validated.

	<u>Millions</u>
OGS (4 versions)	\$40.6
CGS (7 versions)	44.6
Documentation, Testing, Compilers, etc.	5.77
Data Reduction Center	8.6
SPS	12.1
ATE	<u>1.9</u>
Total	\$113.57

These costs do not include maintenance costs incurred by ADC or AFLC. It was understood that total system cost was just over \$2 billion. The SPS is expected to cost \$25.6 million for the prototype, and \$9 million each for the production systems, which may be 6 systems (\$54 million).

Observations/Lessons Learned

- The SPO indicated that testing requirements were well defined; even so, the SPO indicated he lacked some software visibility during its production.

- The ADC has three groups of software maintenance personnel located at three different locations. Although ADC exercises software configuration control, the groups tend to take "liberties" with the system which makes it very difficult to control.
- There is a need for more contractor engineering rigor in developing software; e.g., use of more structured techniques.
- Good use can be made of a software integration/validation contractor. The use of industry is better than in-house laboratories. Configuration control is very important.
- There is too much documentation required by the current military standards; MIL-STD-483 and -490 are out of date in certain instances. There should be more automated, self-documentating tools.
- The ASPRs provide too much protection to industry concerning software data rights.
- System developments should make more use of existing operating systems and other off-the-shelf software.
- There is a need to stabilize requirements early in the system acquisition. Changes to the requirements for DSP caused major software re-design efforts which delayed the operational date of the system. With extended development of a system, there is the danger that it will be overtaken by technology changes.
- The software military personnel turnover rate is too high (about every 3 years). The SPO needs engineers, not software programmers, in the project offices and in integration activities. Air Force software training is oriented toward business systems. The classification systems need improvement to reflect engineering (education and skills) aspects for software.
- Major updating of this system does not go through the DSARC review process.
- There is a need to further define the components of software costs if OSD requires cost breakouts.

2.3.2.2 MINUTEMAN II AND III

Date of Interview: February 25, 1975

Organization: Minuteman System Program Office
SAMSO
Norton AFB, California

System Functions

The Minuteman II and III is a long range intercontinental strategic ballistic missile.

System Status

The Minuteman II and III are very mature programs. The software programs for these missiles seem to have been very successful during the last several years. The SPO continues to do the software maintenance and improvements on a 18 month to 2-year cycle although the system has transitioned. Generally, the system has always met IOC dates since it has transitioned which are agreed to between the SPO and SAC. Some of the major recent milestones are:

- Development of Minuteman III, 1966-1970.
- IOC, Minuteman III, June 1970.
- Development of CDB Improvement Package (Control Data Buffer), 1970-1974.
- IOC of CDB, June 1974.
- Development of GIP (Guidance Improvement Package), 1974-1978.
- IOC expected for GIP, June 1978.

Scope of Software

There are a number of different types and sizes of computers both for ground support and those integral to the missile. For example, the operational computers include the Autonetics D37C/D, Sylvania MPCU, Boeing DSAP and ALCC DPU, and the Univac WSC. Maintenance trainers require computers, e.g., Control Data (Honeywell) DDP-24, Univac 1624 and 1616, Weapon System Controller Fig A 400, and Interdata Model 70.

Ground maintenance, development, testing simulation and other support include a variety of computers, e.g., Univac 1218, IBM 360/85, IBM 7094, and Univac 1108. The computers are generally programmed in either FORTRAN or assembly language. There was no attempt to get a complete listing of computers, but rather a sufficient listing to gain an understanding for the complexity of software.

The Minuteman software can be broken down into the following categories. The software development and validation contractors are also shown.

<u>Software Package</u>	<u>Development Contractor</u>	<u>Validation Contractor</u>
1. Flight Program	Autonetics	TRW
2. Targeting Program	TRW	Logicon
3. C&C Program	TRW	Logicon
4. SIOP Software	TRW/Logicon/SAI	Logicon
5. Ground Program	Autonetics	TRW
6. Code Generation Program	Autonetics	TRW
7. Trainer Programs	TRW/Boeing/ Sylvania/Autonetics	None

Apparently, the overall software integration task lies with SAMSO. It should also be noted that the above list does not include unique SAC (user) or AFLC software activities. A major point was made of having a separate validation contractor for every piece of software developed. The SPO felt that they owe a major part of their success to this approach.

The number of all operational minuteman programs approaches 320,000 words/source statements.¹ In addition, two simulator programs on the IBM 360/85 require 350,000 bytes each. The size of support and other software programs was not obtained.

Software Cost Data

The following cost information was provided. It excludes costs incurred by AFLC and SAC.

¹Material provided included both word and source breakdown.

	FY73	FY74 (Millions)	FY75	FY76
Total Minuteman Expenditures	\$833.4	\$735.7	\$718.4	\$653.8
Performance & Software Div.	\$ 22	\$ 20.7	\$ 18.9	\$ 23
Guidance & Control Div.	\$ 5.1	\$ 6.2	\$ 13	\$ 13
Personnel Subsystems Division	0	\$ 1.5	0	\$.6
Total SAMSO SPO Software Costs	\$ 27.1	\$ 28.4	\$ 31.9	\$ 36.6

Observations/Lessons Learned

- There is a need to do a good system analysis of requirements, to define all interfaces, and to insure that all components to a system can be built before starting the detailed software design and coding.
- Define early test packages and early deliveries at the start of a contract; don't wait until the end of a project to learn that something is wrong.
- There is a need for a well structured development plan for software. The plan should include all life-cycle considerations.
- Sometimes the software work is forced to follow unrealistic schedules and milestones which may be imposed to meet total system schedules or DSARC decision points.
- There are at times software compromises made at the beginning of a system development because of the lack of funds. However, there always seems to be funds to correct for software deficiencies later.

- Use a separate validation contractor and start him early. One should expect to pay 25% to 30% of total software costs for separate validation support. A validation clause should be agreed upon and included in the software development contract.
- Software cost breakouts would provide more visibility to management and should be identified in the contract.
- There is a need for program reviews of software at all management levels to improve upwards visibility.
- There is no forum for transferring good software management practices between programs.
- There is a need for an R&D program to find ways to accomplish automatic verification through simulation or other means to avoid expensive testing.
- There is a need to have top quality/qualified people for software management. There is also a need for improving high level management's understanding of software acquisition dependencies and constraints.

2.3.2.3 F-111

Date of Interview: February 27, 1975

Organization: Sacramento ALC
Sacramento, California

System Functions

The F-111 is a variable wing-sweep tactical fighter which operates well above Mach 2 at high altitudes, has a capability for low-level supersonic dashes, and has the ability to operate at forward areas where airfields are not well developed. There is also a strategic bomber version of the F-111.

System Status

This system has transitioned. Deliveries of the first aircraft to the first operational wing started in October 1967. The Air Force Logistics Command assumed software maintenance responsibility in July 1973. Since that time a software maintenance support laboratory has been implemented at the Sacramento ALC (currently being updated), procedures have been developed with user groups for updating and maintaining

the software, documentation of the operational software has been brought up to date, and a test aircraft is available for inflight testing of software at Sacramento. This is one of the few avionics systems which has transitioned where AFLC and the user commands are maintaining the software.

Scope of Software

There are three distinct software areas which require maintenance for the F-111: (1) Operational, (2) Automatic Test Equipment (ATE), and (3) Simulator Trainer. The operational and ATE software is maintained at the Sacramento ALC, the simulator trainer software at Ogden ALC.

Operational Software. There are two IBM 4 Pi computers in the F/B-111, with core storage of 16,896 words each, for the operational programs. (There are other small airborne computers with up to 4K memory.) The programs are highly packed in the 4 Pi computers, and if new functions are added, existing functions must be eliminated. There are seven different versions of the operational software which must be maintained.

	<u>Guidance and Navigation Computer</u>	<u>Weapons Delivery Computer</u>
F-111D	X	X
F-111F	X	X
FB-111	X	X
Common Inertial Navigation Program	X	

The size of the programs are approximately 16K words each, except for the common inertial navigation program (Autonetics Computer) which is 4K words. In addition, a mission tape preparation program (8K words) is prepared for use by the user commands on local equipment (8K word Micro-D) for loading the programs. Support software for operational programs includes assemblers, utilities, etc. which operates on IBM 360 equipment. As changes are made to the operational software, the Ogden ALC must be notified so that they can update the simulator trainer programs. Updating/improvements are accomplished on a 15-month schedule. However, critical program errors can be corrected within two weeks. The Sacramento ALC relies heavily on contractor support for maintaining the operational programs.

ATE Software. The Sacramento ALC is responsible for 11 types of test equipment and 996 software programs. The ATE software represents a very large effort. (It is understood that all programs are not for the F-111.) There are various compilers, utilities and specialized languages associated with ATE software maintenance. There are also various support computers involved, including the RCA 301, IBM 360/44, 360/77, and IBM 1401. The development of the ATE software was referred to as "reverse engineering", that is, the ATE software supplied with the F-111 was largely developed for acceptance testing and demonstration of a good unit rather than for determining/diagnosing faulty equipment. Software documentation for ATE was unsatisfactory. The ALC has ongoing efforts to correct those deficiencies. The Sacramento ALC relies heavily on contractor support to improve and maintain the ATE software.

Software Cost Data

The cost data received was for maintenance only at the Sacramento ALC for the operational and ATE software. Cost information for development, user costs, and costs associated with maintaining simulator trainer software at the Ogden ALC were not available. The following is a suggested listing where software maintenance costs are occurring:

1. Operational flight programs (OFF)
2. Automatic test equipment (ATE)
3. Comptroller ADP support
4. Air crew training simulator software
5. Engineering flight testing
6. User flight testing
7. User support costs

The cost information provided is as follows:

OFF

OFF software avionics laboratory	\$17.3 million
Annual OFF software maintenance	3.0 million

Note: Approximately 60-70 people (in-house plus General Dynamics) are required for F-111 OFF maintenance.

ATE

Documentation (acceptance test specifications, test requirements specifications, computer source information, test program source information, adapter box documents, revision to tester documents): \$4 million approved;
final costs may greatly exceed this amount.

Annual maintenance cost: Estimated \$2 to \$4 million, not including cost to establish support facility.

Observations/Lessons Learned

- There are benefits from an engineering standpoint to maintain the software at the same location where maintenance engineering is accomplished on the avionics equipment.
- There is a significant amount of support software in avionics.
- Contracts should be more specific about who owns the software support tools.
- Contracts should specify what ADP equipment the support software will operate on. For this system, there is very limited access to ADP resources (IBM 360/44) to perform software maintenance.
- The ATE software was initially oriented towards acceptance demonstration and not for field checkout and diagnostics.
- There was very limited software documentation, configuration control methods, and test/support software provided at transition. It became expensive to develop documentation, and there is heavy reliance on the prime contractor for maintenance support.
- The current Air Force Technical Order system is being used for software control. This seems expensive, and perhaps not the best suited for software control.
- There were not found to be Air Force guidelines for establishing OFP software maintenance procedures and practices.

- The flight testing of all software changes is expensive. It may be a high payoff area for R&D to determine to what extent simulation or other techniques can be used.
- There was no spare memory remaining in the OFP computers. Old functions must be deleted to add new ones. It appears that the memory size will have to be increased. It was understood that alternatives are under study.
- It appears that a study would be useful to determine the most cost effective method for maintaining weapon systems software in AFLC, e.g., a central facility, a central computer with remote terminals, or having each ALC self-sufficient.
- Personnel and other resources do not seem to be provided to the ALC to implement the AFLC policy of performing software maintenance with in-house resources. It appears that the ALCs will require software maintenance support from the prime contractors for the foreseeable future.
- Most of the above problems can be attributed to the lack of O&M life cycle planning in the early acquisition of this system.

2.3.2.4 Wild Weasel II (AN/APR-38)

Date of Interview: March 7, 1975

Organization: SPO, Wild Weasel
Aeronautical Systems Division
Wright Patterson AFB

System Functions

The F4D/E Wild Weasel AN/APR-38 mission is to seek out and destroy surface-to-air missiles and radar-guided anti-aircraft guns.

System Status

The program began with the TAC ROC 35-68 in 1968. Early versions were hardwired analog AN/APR-25 and AN/APR-26 systems and were used in Southeast Asia during the Vietnamese actions. The current system under development is for the F4D/E aircraft and will have digital computers. The software will perform the offensive and defensive information processing. This decision was made because defensive/offensive parameters are subject to

changes which can be handled more easily and cheaper by software than through hardware changes. At the same time, the system update provides increased accuracy, clearer displays, and improved range determination.

The testing phase with the F4D aircraft was completed in 1974, and was sufficiently successful so that a production decision could have been made in February 1975. However, the requirements were changed in that the system is to be further developed for the F4E rather than the F4D. This slips the production decision until June 1976. The DoD believes that the hardware risks have been reduced so that four sets can now be built.

Scope of Software

The software generally consists of the following:

<u>Software Categories</u>	<u>Computer</u>
Homing and Warning	Airborne -- TI-2540, 64K
Electromagnetic Environment Simulator (EES)	Ground -- Datacraft 6024, 64K
Special Warning Receiver (SWR)	Airborne -- ATI ATAC-8, 4K
Various data reduction software for development flight test	Ground -- IBM 370/165 (to be translated to CDC 6500 for Edwards AFB flight test)
Test Bench Software	Ground -- Varian
Trainer Simulator	Ground -- Computer unknown

A problem was encountered in that the memory capacity for the homing and warning computer (TI 2540) was exceeded. Going from 16K to 32K caused redesign problems. For the production model, the computer size is being increased to 64K.

On this system, there were more hardware problems than software. The hardware tended to hold up software development. A considerable amount of time was needed to integrate hardware and software, e.g., there were timing and interface problems.

The AFLC has officer personnel assigned to the SPO for maintenance planning after the system transitions. The Warner Robbins ALC will be the prime software support facility. Some time by Warner Robbins personnel is spent at McDonnell; however, shortage of TDY funds is impacting travel by these personnel.

Software contractors include: (1) Texas Instruments, Dallas, Texas; (2) ATI, Sunnyvale, California; and (3) McDonnell, St. Louis, Missouri. Software work for testing is being done by the Air Force at Wright-Patterson AFB and at Edwards AFB.

Software Cost Data

The personnel interviewed indicated that they do not breakout software costs from the total R&D costs. Indications were that it would be very difficult to do this because there is a "mountain" of software costs behind OFP, ATE, and SIM programs. For example, there is software needed for equipment design, prototyping, testing, etc., that the contractor must use to complete the overall system. It was stated, however, that an estimate of \$14 direct cost per computer instruction has been made. If overhead is applied, the cost is \$45 per instruction.

Observations/Lessons Learned

- The software has been acquired using a hardware/software prime-item development specification, i.e., no software specifications per se were imposed by USAF on the contractor. On this system, firm requirements could not be established early because of the R&D nature (what ifs) of the program.
- A phased approach to system development was used. Although this may have cost more at the beginning of the project, it was probably less costly for the total development cycle.
- More use of breadboards, prototypes, etc., should be made to prove design approaches. A problem arose with this system as they underestimated computer memory requirements.
- This type of system is too complex for independent validation and verification; actual flight tests are needed to determine whether the system works.

- Every contractor poses different computers and software tools. It is a problem to maintain many kinds of computers. In the long term, there should be a standard family of computer hardware and languages, and there should be a higher order language for avionics.
- There is no organized way to transfer technology between projects. Project people are generally unaware of what others are doing.
- Current military standards require too much documentation. Efforts should be made to develop self-documentation techniques for software.
- There should be a system manager for software as is done for hardware.
- There is a need for weapon systems software management personnel with specific software management experience. Personnel continuity is a problem, both in-house and with the contractors. The software people must understand hardware and vice versa.
- The contractor should be project oriented, i.e., organize a team and keep it together until the project is completed.
- Good relationships must be established with the contractor. This seemed to be the case for this system which has a cost-plus contract. The belief was that fixed-price contracts should be discouraged for software development.

2.3.2.5 B-1

Date of Interview: March 7, 1975

Organization: B-1 System Program Office
Aeronautical Systems Division
Wright-Patterson AFB

System Functions

The B-1 is a strategic bomber under development as a possible replacement for the B-52.

System Status

The avionics contract was let 2 years after the airframe/power plant contract. The avionics package including computers and

software was installed in a C-131 in mid 1974 for testing. Although Rockwell International is the system contractor, Boeing is the software/integration contractor for avionics. The system has entered the later stages of development and testing. However, the program has slipped approximately 15 months. It is expected the complete system (the avionics airplane) will fly in the spring of 1976. A production decision is planned for November 1976. A working group is now developing material for a production contract.

Scope of Software

The B-1 will have some 23 computers (seven are 2 or more of one kind) on board provided by 11 different manufacturers; see Table 2-2. The computers range in size (depending on their functions) from 32 words of memory to one which has 32,768 words of core storage, and 400,000 words of mass storage. The computers vary in capability; e.g., programmable read only memory, read only memory and general purpose. The IBM 370 computers are the principal ones used for software development work.

The B-1 avionics software is categorized as follows:

- Offensive Flight Software
- Support Software
- Simulation Software

The offensive flight software performs the following functions: (1) navigation, (2) control and displays, (3) weapon delivery, (4) avionics integrated testing, and (5) executive. The programming for these functions was generally done in higher order language, i.e., the Air Force specified that a JCVIAL language would be used.

The support software is that required for development and module level verification of flight software. It operates on the host computer at Boeing, IBM 370, which is used to generate the software for the avionics computers. It provides language processing, assembly, editing, utilities, simulation support, and software management aids.

The simulation software provides dynamic testing support capabilities. A software test bed has been developed with the capability to combine simulated elements and real hardware. Some of the functions that are simulated are navigation, radar, EVS, and weapons; e.g., SRAM, nuclear gravity, and conventional gravity.

TABLE 2-2
B-1 DIGITAL COMPUTERS

Subsystem	Subsystem Vendor	Computer Manufacturer	Number of Computers	Memory Types	Memory Size	
					Word Length (Bits)	No. of Words
Central Air Data Computer	Air Research	Air Research	2	PROM	20	2560
RGA/AOA/AVL	Sundstrand	Sundstrand (Serial)	2	PROM PROM RAM	8 16 16	2048 512 64
Fuel and C.G. Management	Simmonds	Autonetics D216	2	Core	16	8192
Gyro Stabilization (ECA)	Northrop	Bendix BDX900	1	PROM	16	2048
Engine Instruments SCUDU	General Electric	Intel	1	PROM RAM	8 16	1536 257

TABLE 2-2
(contd)

Subsystem	Subsystem Vendor	Computer Manufacturer	Number of Computers	Memory Types	Memory Size	
					Word Length (Bits)	No. of Words
Flight Inst. Signal Converter	Singer	Singer	2	ROM	4	256
Vertical Situation Display	Sperry Flight Systems	Sperry	2	ROM RAM	16 16	1024 32
EMUX (Control Boxes) (CI Boxes)	Radiation	Radiation	4 2	PROM	20	11000
Central Integrated Test	RI	IBM AP2	1	Core	16	49152
Offensive Avionics	Boeing	Singer SKC2070	2	Core ROM MSU	32	32768 2048 400000

TABLE 2-2
(contd)

Subsystem	Subsystem Vendor	Computer Manufacturer	Number of Computers	Memory Types	Memory Size	
					Word Length (Bits)	No. of Words
Defensive Avionics	AIL	Litton 4516	1	Core	32	32768
Defensive Avionics	Boeing	SKC2070	1	Core ROM	32	32768

The SPO has approximately 30 people from Logicon to assist in software management. The design control is done through the Boeing configuration management system. The opinion was that if a contractor has a good configuration control procedure, use his as he is familiar with it.

The software has not been without problems. However, it has avoided being on the critical path for the total system because of hardware problems with other system components.

Software Cost Data

The B-1 RDT&E contracts do not provide for cost reporting at a level such that software development costs can be separated from hardware costs. This information could be developed from contractor cost data; however, it would require an extensive amount of work for both contractor and SPO personnel. Contractual action might be necessary to obtain cost data since the basic contracts do not require this level of reporting at this time.

In May 1974, an informal estimate of the costs of software development at The Boeing Company was made by Air Force personnel. This estimate was made on the offensive avionics software development, which amounts to better than 25% of the total software for the on-aircraft computer software. The estimate was approximately \$11.6M for FY's 72 through 76. This estimate includes costs for the entire support software system (compilers, assemblers, simulation tools, etc.), documentation (milestone documents and other reports), test facilities (System Development Laboratory), coding laboratory testing (software-hardware integration testing), and support of tests through the flight test program. It should be emphasized that these are only estimates, and are very difficult to absolutely separate out since many of the facilities involved are used for both hardware integration testing and for software testing. In addition, some system design engineering costs cannot be separated between software and hardware. However, it was felt by the SPO that this figure, which represents approximately 10% of the costs for the offensive systems development as of May 1974, is a fair estimate.

In addition to the above, the B-1 SPO has contracted with Logicon for support services in software development. They provide independent assessment of the offensive and defensive software development, are developing verification tools, and have provided software planning inputs. The cumulative cost of this contract through FY75 is \$1.9 million.

Observations/Lessons Learned

- Software must be properly addressed in the contract. The SPO personnel indicated at least these software areas should be covered by the contract: (1) requirements, (2) specifications, (3) management plan, (4) testing (V&V), (5) integration, and (6) documentation.
- Support software is very expensive to develop. Some commonality is needed for transfer across systems.
- Apply good engineering to software as one does with hardware. People do not consider software as an engineering science, even the universities. The Services must stress good software management and engineering practices.
- Don't always correct all software problems and delay progress, but maintain a good accounting of those problems which can be corrected at a later time.
- Management does not always understand software.
- The Services must develop more engineers who know more about software.
- Software has an advantage over hardware in that it is more flexible for changing to respond to mission requirements, and it is less expensive to maintain.
- Software reliability doesn't seem to be a major problem. One must have different levels of effort for V&V, depending upon the criticality of the weapon. (Minuteman was cited as a weapon requiring extraordinary V&V.)
- It is most difficult to identify all software costs. There is no reason to believe from the available information that weapon systems software costs are too high. There is no historical data for comparison. In addition, software costs are becoming greater because the Services are using it to perform more functions. This trend may continue.

2.3.2.6 485L Tactical Air Control System

Date of Interview: March 12, 1975

Organization: 485L System Program Office
Electronic Systems Division
L. G. Hanscom Field, Bedford, Massachusetts

System Functions

The 485L program is to develop the tactical air control system (TACS) for theatre control of air forces. It represents a set of improvements to the existing system (407L) which was developed for use by TAC, USAFE and PACAF.

System Status

The 407L system was developed during the 1964-1972 time period. The 485L programs, started in 1972, is expected to continue through 1979. The new features include such improvements as automating the mission planning, targeting, and status of forces at the Tactical Air Control Centers (TACC). Data links will be provided for automatic transmission of operations and intelligence data between various components of TACS (e.g., Control Reporting Center (CRC), and Direct Air Support Centers (DASC)). Interfacing to provide for interoperability with Army, Navy, and Marine Corps systems is underway. The upgrading of TACS involves rewriting some of the current computer programs, new computers, and acquiring expanded communications capabilities. The software contract with CSC was completed September 1973. A production decision for the TACC is scheduled for October 1976; transition (going operational) is scheduled for October 1978. None of the system upgrading is subject to DSARC reviews.

Scope of Software

The computers used in the system include the AN/UYK-7 and -25 (used extensively by the Navy in NTDS), and the Hughes 4118. The UNIVAC 9300 is used for development support. General Dynamics is providing the hardware for TACC, and Hughes is providing hardware for automatic radar tracking and TACS/TADS functions and interfacing.

The software consists of operational and support functions. The operational software accomplishes the broad functions of data processing and display, communications processing, and data source terminal processing. The support software includes the operating systems, JOVIAL compiler, assembly systems, utility programs, and diagnostics. Some program sizes were given in order to gain an understanding of the magnitude of the software effort.

<u>Operational (Applications) Software</u>	<u>Sizes</u>
Data Processing and Display	100K instructions 150K data
Communications Processor	36K instructions 9K data
Data Source Terminals	50K instructions 3K data
Exercise Data Generator	20K instructions
 <u>Support Software</u>	
DPD OS	60K instructions
CP OS	65K instructions
DST OS	60K instructions
JOVIAL	
DPD	99K instructions 151K data
MPP	98K instructions 137K data
Assemblers	
DPD ASM	10K instructions and data
CP/DSP/ASM	15K instructions and data
Utilities	
DPD	30K instructions 13K data
CP/DST	40K instructions and data

An additional software effort is the modification of programs for interfacing with TACS/TADS.

The CSC prime contractor organized a staff at Langley AFB to develop the software. It is believed the location at Langley made it difficult for the contractor to acquire the best people for the project. The Air Force also assigned approximately 25 personnel to the project at Langley.

The functional requirements were specified in the RFP, but computers were not identified per se. However, the RFP was written in such a way that an existing computer(s) with support software would be required. The AN/UYK-7 and -25 and H-4118 met the requirements.

Software Cost Data

The software costs were estimated at \$6 million to CSC for TACC until the production decision date of October 1976; and perhaps another \$2 million for software support associated with equipment deliveries from General Dynamics. Any additional costs by the Air Force, e.g., the TAC support, ESD, and other contractor support would be added costs.

Observations/Lessons Learned

- The government ought to consider producing Part I specification before contracting. On this system, it was difficult for the contractor to arrive at acceptable Part I specifications and thus startup was slow and costly.
- Schedules should be more realistic. It was felt that the schedules developed for this program were unrealistic on the part of both the government and contractor. The tendency is for contractors to agree to schedules in the RFP.
- The system of contractor selection does not permit realistic bidding by contractors on either costs or schedules. Contractor "track records" are also not considered in source selection. The "buy-in" syndrome ought to be corrected.
- Splitting software and computer hardware contracting may not be the best practices for some systems. Having only one contractor provides better overall scheduling of resources and activities, and avoids "finger pointing" as problems arise.
- For some systems, the government ought to consider advantages for a parallel development effort at the beginning of a project before awarding the final contract. This will help preserve competition factors to get better contractor personnel.
- Current DoD regulations do not cover firmware. Problems arose with this system in the areas of firmware, micro-programming and software/hardware interfaces. The contractor is at liberty to make changes as he sees fit as these areas are not configuration controlled.
- The government needs to provide better guidelines on testing requirements to the contractor in the RFP.

- Interoperability requirements must be considered early in the project to avoid costly redesign of the system. (It was estimated that some 2000 software patches may be required in 485L to interface with TACS/TADS.)
- There is too much documentation required by current directives. The users tend to impose all requirements specified by current military standards.
- Conduct R&D to determine what is needed to make software more transportable.
- Present procedures do not provide meaningful cost information.

2.3.2.7 427M

Date of Interview: March 12, 1975

Organization: 427M System Program Office
Electronic Systems Division
L. G. Hanscom Field, Bedford, Massachusetts

System Functions

The 427M is a command and control system in support of NORAD for performing its mission of continental air defense and space surveillance.

System Status

The 427M system is an update/replacement for 425L, which is currently operational at NORAD. The system requirements were developed during the 1970-71 time period. However, they were not completely acceptable to the user and further work was necessary, using the ADC concepts of operations plans as a basis. The approved (Air Staff) requirements became the baseline for the system work. A contract was awarded to Philco in March 1973. The planned transition date is October 1977. This has slipped from the original date of July 1976.

Scope of Software

The WWMCCS Honeywell 6050 and 6080 systems will be used for the main processors, Data Net 355 systems for communications, and NOVA 800,840 and 1220 systems for display processors. A UNIVAC 1106 and additional NOVA systems may be added.

The system is composed of 3 subsystems: (1) Communication System Segment (CSS), (2) Space Communication Center, and (3) NORAD Computer System (for warning, air defense, staff support, etc.). The operational software of these systems performs the functions of communications processing, space tracking, and weapons order of battle and other air defensive operational information processing and display. The support software consists of WWMCCS GCOS, FORTRAN compilers, GERTS, test tools, debug tools, simulation software, and NOVA/Honeywell interface software. Some indication as to the size of the computer programs is as follows:

NOVA display	66,000 instructions
Space tracking	553,000 FORTRAN instructions, 1.6 million words for data base
Assembly system	31,000 words
Weapons order of battle, etc.	225,000 FORTRAN instructions
Assembly System	50,000 words

The contractors for space tracking are Philco and SDC. The NORAD in-house personnel are developing the weapons order of battle and other operational and staff support functions for the NORAD Computer System.

Software Cost Data

The software cost information was not available from the interview. However, it was stated that the total cost of the system is \$124 million, and that the monthly expenditure rate with Philco is \$800,000.

Observations/Lessons Learned

- Changing requirements presented major problems; trying to cope with changes to the WWMCCS of GCOS (new release each 6 months) was expensive and time consuming. Management should recognize the changing nature of software and consider this in their planning.
- Functional requirements should be clearly stated before entering into a development contract. A good check on feasibility of requirements is to determine whether they are testable; if not, they should be eliminated.

- More realistic schedules and budgets should be established. Management reviews of originally developed budgets and schedules resulted in a reduction of both. The system now faces the problem of cost and schedule overrun. All contractor bidders were too low.
- Literal interpretations of current Military Standards and ASPRs make them inconsistent with the use of commercial software. Reasonable interpretations can be made, however, users do not always agree. They tend to insist on provisions for maintainability, safety, etc., as required under ASPRs and Military Standards.
- The philosophy of software has changed over the years. Engineers are starting to use a modular approach to design, and structured programming. In doing this, however, one must be able to perform validation testing on a functional basis.
- Higher orders of language may be a good thing, but it is important that the contractor is familiar with languages and development tools he is to use to avoid expensive startup costs.
- There is a need for more exchange across programs of experiences and lessons learned.

2.3.2.8 451D - Combat Grande

Date of Interview: March 13, 1975

Organization: Combat Grande System Program Office
Electronic Systems Division
L. G. Hanscom Field, Bedford, Massachusetts

System Functions

Combat Grande is an air defense system being acquired for the Spanish government. It is similar to the BUIIC system of the United States. It will be tied into 7 radar sites when the system becomes operational in Spain.

System Status

A production decision has been made and a contract has been awarded to a joint Spanish/Hughes Aircraft Company. The RFP was released February 1973, and the contract awarded February 1974. The system is expected to become operational March 1978,

i.e., turnover and transition to the Spanish Air Force. The Spanish are expected to maintain and operate the system.

Scope of Software

The computers consist of the H-5118 (Hughes) as the main processor, and TI-980 Minis to drive 28 consoles and at each of the 7 radar sites. Also included is a TI-980 to receive and display the RAPPI, and the PDP-11 for communications control/branching.

The main software consists of:

H-5118 computer	Approaches 375,000 instructions for operational (radar data processing and display) and support programs; and just over 340,000 positions of memory required for the data base.
-----------------	---

In addition, the Hughes Company is using simulation software on an IBM 370 support computer, and data reduction software. There are utilities and diagnostics being developed, and software is required for the TI-980 and PDP-11 computers.

Software Cost Data

A great deal of planning and front-end requirements specifications went into this system, so much so that the Air Force and the contractor entered into a fixed price with incentive contract with a great deal of confidence. The previous ESD experience with the BUIC system was used to a good advantage. The complete system cost is \$58 million. The software costs were given as an estimate of \$5 million.

Observations/Lessons Learned

The items listed below are not necessarily lessons learned from Combat Grande acquisition, but rather they are items which have been applied to this system and apparently to a very good advantage. The system seemed to have gotten started in the proper way, e.g., firm requirements, software development plan required by the RFP (quality of plan was part of proposal evaluation), good use of existing hardware and software, and realistic schedules. The following are some of the management practices and procedures employed:

- Do a good job on requirements definitions.
- Require a good software management plan.

- Identify support software as a deliverable item.
- Develop realistic costs and schedules, and hold contractor to them.
- Require the contractor to report his manning against progress, costs, budgets.
- Develop specifications to minimize risks (to the extent possible).
- Be as specific as possible in defining system performance without undue constraints on the contractor.
- Identify, if possible, things to be used off-the-shelf and what has to be developed.
- Get the user involved.
- Do your homework before conducting reviews.
- Emphasize top down design; do design work before starting the software coding.
- De-emphasize testing off-the-shelf software when it has been in use on other systems.
- Keep down support program documentation.
- Use HOL when one can.
- Have growth requirements defined so that the system won't use up all resources, closely monitor storage budget, throughput timing, etc.

Other observations made during the interview of a broader nature include:

- Better cost data is needed, and tied to what contractor people are doing.
- More standardization is needed - terminology, compilers, and equipment.
- Software management techniques need to be improved; a good area for R&D.
- Training is needed for software acquisition management.

- Personnel career field is needed for software engineers.
- Conduct risk analysis before making contract awards.
- Fixed price contracts are troublesome with software acquisition unless requirements are well thought out.

2.3.2.9 AWACS

Date of Interview: March 13, 1975

Organization: AWACS System Program Office
Electronic Systems Division
L. G. Hanscom Field, Bedford, Massachusetts

System Functions

This is an airborne warning and control system (AWACS) being developed for ADC and TAC. The primary use by ADC will be as a survivable early warning command-and-control center for the identification, surveillance, and tracking of airborne enemy forces, and for command and control of NORAD forces. A similar system operated by TAC will be used as airborne command-and-control centers for quick-reaction deployment and tactical operations.

System Status

This system was initiated in 1965. There were study efforts to determine feasibility, e.g., concern over practicality of airborne radar, and formal contract definition efforts through 1968-1969 time period. In 1971, the Air Force entered into a contract with Boeing to develop Phase I specifications. This was the first time software was considered as part of the system. Also in 1971, the Air Force started two separate development efforts on the radar antenna to reduce risks.

The Boeing Company was selected as the prime contractor for software. A software subcontract was given to IBM, but this effort has been phased out, leaving only Boeing.

Development and testing of the system has progressed to the point where a production decision was to be made in December 1974. It is understood that the production decision is pending while further evaluations are being made.

AWACS is to be the first user of the TACS/TADS interface standards. The interface capability is to be completed with delivery of the first system to TAC which is planned for November 1976, the IOC is planned for March 1977 (dependent upon a production decision).

Scope of Software

The system components (hardware and software) consist of operational, automatic test, and crew training programs. In addition, there is considerable support software and utility programs to support development and maintenance activities. The computer used for the operational flight programs were generally a development effort, i.e., the IBM 4 Pi CC-I used for the flight operational computer and crew training. The IBM 370-155 (commercially available) is also used for crew training support, and for software development and maintenance support. Other special developed computers are used for radar data analysis and reduction (Westinghouse 64K memory dual processor), navigation (2 Delco 16K computers), and computers for communications, and for system performance analysis.

The software development in terms of program sizes represented major efforts. The following provides an indicator of the efforts required:

	<u>Instruction</u>	<u>Data Base</u>
Operational Programs	280K	223K
System Error and Analysis	136K	555K
Training Flight Simulator	208K	223K
Ground Support	1190K	895K

The ground computer (IBM 370-155) had off-the-shelf software support, i.e., the operating system, FORTRAN, and utilities. The airborne computers required complete development of software except for the navigation system, which was generally available.

Software Cost Data

The software costs were about 2% of total system cost (\$2.7 billion for total system). The following cost information for software was provided.

	<u>Instructions</u>	<u>Budget/Cost</u>	<u>Cost/Instruction</u>
Brassboard	240K	\$ 1,500,000	\$ 6.00
Airborne Tracking	70K	418,000	6.00
System Demonstration			
Boeing	610K	8,000,000	13.00
IBM	240K		
 RDT&E			
SID Carryover	640K		
Boeing New (est.)	1,430K	19,000,000	13.00
SDC New (est.)	<u>60K</u>	<u>450,000</u>	7.50
total:	3,290K	\$29,368,000	

Costs ran close to budget. The effort to utilize the TACS/TADS was estimated to cost from \$800,000 to \$1 million.

Observations/Lessons Learned

- The software development effort was accomplished in phases. Phase I was a software prototype developed on a general purpose computer. There were no core limitations and the specifications were quite loose. The second phase included the use of limited hardened computers to prototype an airborne system. (This phase was completed six months ahead of schedule.) Integration activities started in 1972 as the computing and other system hardware was becoming available.
- Although the contract did not require a software management plan, Boeing agreed to develop one. The plan has not been maintained because of its high cost, i.e., \$300,000 over a 5-year period. Management is carried out by informal contact with Boeing, and the contractor's internal milestones, procedures, etc., are used by the AF for project control. A software development plan should be required at the proposal phase. The quality of the plan should be part of the criteria for contractor selection.
- The progress review process was left up to the contractor (Boeing). However, test scripts, mission plans, and documentation were very carefully reviewed by Air Force program managers and users before reviews were conducted.

- Apparently, the software development was well managed. Cost estimates and schedules were realistically developed, and the system was developed within budget and schedule. There was emphasis on development tools, costs and schedules. There were several DSARC reviews as the system was being developed. Software was always addressed, i.e., risk, cost, performance, test, and schedules.
- The program manager should assess the contractor's software production management capability; do this early.
- The DoD acquisition process for software needs improving. Current regulations are generally oriented toward hardware, something is needed for software DoD-wide like the Air Force's 800.XX.
- Part I and II specifications are used for configuration management. Time to develop Part I specifications is generally too short.
- Part I specifications are required to baseline the system. One always tries to put more things into specifications than is possible to accomplish; things change anyway.
- A separate V&V activity is needed; V&V is 20% to 30% of software costs. Boeing provided V&V separate from the software group.
- There is a need for an AFSC for weapon systems software personnel. There is also a need for training system managers in software management.
- There is a need for R&D to improve software management techniques.

2.3.3 Joint Interoperability Testing

2.3.3.1 TACS/TADS

Date of Interview: February 26, 1975

Organization: Joint Integration Test Facility (JITF)
San Diego, California

System Functions

The JITF was established by the Joint Chiefs of Staff (JCS) and placed under the executive management of the Navy. Its purpose

is aimed at achieving true interoperability among tactical air control systems/tactical air defense system (TACS/TADS) of the Army, Navy, Air Force and Marine Corps. The specific systems currently involved are NTDS, Q-73 Missile Minder, portions of 407,485L, and MACCS.

System Status

Testing facilities and resources have been organized in the Southern California area by the Army, Navy, Air Force and Marine Corps, and some inoperability testing has been accomplished. Both interface hardware and software problems have been encountered. Documentation of a standard inter-system tactical language has been published. This will become JCS Publication #10 in its final form. The systems are to be demonstrated next year (1976) in a joint exercise along the Atlantic seaboard under the control of CINCLANT.

Scope of Software

Although some new software is involved for TACS/TADS interfacing, MITRE interest in reviewing TACS/TADS was primarily because it represents an attempt by the Services to tie independently developed systems together into a common mission. The JITF interview identified a new dimension to the software problem, that is, that many weapon systems developed by the Services are in fact sub-systems and are part of an overall DoD system.

Software Cost Data

A cost estimate of \$54 million was given for the ongoing effort at JITF. A breakdown of this cost was not provided. However, it probably does not include some of the costs by the Services to revise their software to interface with other systems, e.g., AWACS estimates software costs of \$800,000 to \$1 million to utilize the TACS/TADS.

Observations/Lessons Learned

- There is a need to establish a formal and continuous inter-system configuration control mechanism. The TACS/TADS charter ends in 1976-1977 after the joint exercise is completed on the Atlantic coast during 1976. It is understood there are plans under GAMO (the next TACS/TADS generation) to configuration control JCS Publication #10. However, it was understood this activity will not start until the 1980's.

- The trend toward use of digital data links in weapon systems has created tactical interoperability problems. The use of digital data instead of other means (e.g., voice) has become an accepted practice in weapon systems.
- Revising weapon systems software and interfacing hardware to operate with other systems is expensive and time consuming. Interoperability requirements should be considered and provided for during the early stages of a system acquisition.

3. SUMMARY OF BASELINE STUDIES AND WORKSHOP PROCEEDINGS

This section presents a brief summation of the major findings and recommendations of the baseline studies and workshop proceedings which were used during the MITRE study. For many of the studies and proceedings, the information presented here is summarized and paraphrased from the publications. There are also cases where direct quotes are made. A notation to this effect has been added where direct quotes appear. The complete citations to these studies and proceedings appear in Section 4 of this Volume to the MITRE study.

3.1 CCIP-85, Information Processing/Data Automation Implications of Air Force Command and Control Requirements in the 1980's, Executive Summary (Revised Edition)

The executive summary of CCIP-85 is almost completely quoted here as it provides an excellent overview of the 11 volumes of the study. The titles to all volumes of CCIP-85 are included in Section 4, Software Acquisition and Management Bibliography. (This revised edition was authored by Barry W. Boehm, et. al., Space and Missile Systems Organization, Los Angeles, California, February 1972.)

A. INTRODUCTION (STUDY PURPOSE)

The study purpose was to construct an integrated Air Force R&D program for the 1970s which will develop the information processing technology needed to meet the likely Air Force command and control (C&C) information-processing requirements of the 1980s. The central concern was with C&C for Air Force combatant units.

B. STUDY CONTEXT

Information processing technology is barely adequate to support Air Force C&C functions today. The major technological strains are not in the computer hardware area, but in software technology; the technology of transforming broad functional C&C requirements into specific, detailed, and unexceptionable sequences of commands for the computer hardware to execute.

A number of trends are coalescing which, by the 1980s, will

- Make C&C considerably more important to Air Force roles and operations;
- Make C&C much more dependent on information processing

technology; and

- Sharply increase the strains on software technology imposed by C&C requirements.

Some of these trends are highly visible today; some are not. But, together, they are gathering momentum from domestic and international pressures and from mutual reinforcement.

- Increasing Dependence of Air Force C&C on Information Processing Technology

Relative to overall Air Force command and control operations, automated information-processing capabilities will assume much more significance by the 1980s.

Continuing rapid advancement in information gathering sensors will result in the timely availability of an ever-increasing quantity of high-quality information that pertains to the battle in progress. Manual processing of this information will be incompatible with its high degree of perishability. Also, continuing rapid advances in computer hardware technology will offer more and more opportunity for benefits in economy, reliability, and performance by automation of command and control functions.

Continuing trends toward less labor-intensive operation of the military services will accelerate trends toward automation of command and control functions.

- Increasing Strains on Software Technology

Except for airborne functions, the Air Force does not need now, and will not need in the 1980s, the largest, fastest computer hardware available to support C&C operations. However, both now and even more in the 1980s, Air Force C&C will place greater demands on software technology than will other applications. In addition to the growing demands caused by increasingly large data bases, sensor input volume, and user traffic, and the added range and sophistication of C&C decision aids, three unique factors of C&C software will continue to stand out.

- It must operate in a highly changeable and unpredictable environment.
- It must operate in a hostile environment.

- Critical outages or mistakes would affect national survival rather than just costs or the safety of individuals.

Thus, trends toward a multipolar world, limited strategic options, and more reliance on automated decision aids, for example, will drive the demands on complexity, flexibility, certification, and security of Air Force C&C software even further beyond civilian demands on software technology.

- Software Technology: Current Strains

First, software strains the Air Force budget. Estimates of current Air Force annual expenditures on software are between \$1 billion and \$1.5 billion, compared to \$300 to \$400 million per year on computer hardware. The recent WWMCCS computer procurement was estimated to involve expenditures of \$50 to \$100 million for hardware and \$722 million for software.

Second, the indirect costs of software slippages far exceed direct costs, because software is on the critical path in overall C&C system development. On one current project, providing an expected seven years of C&C capability for a total cost of about \$1.4 billion (or about \$200 million per year), software delays have caused a six-month delay in making the system available to the user command, resulting in a loss of about \$100 million worth of C&C capability. Moreover, in order to keep the software from causing further delays, several important functions will not be provided in the initial software delivery.

Third, software is unreliable. Recently, a software error aboard a French meteorological satellite caused it to "emergency destruct" half of its force of weather balloons instead of interrogating them. Current Air Force software reliability problems indicate that similar software errors could cause the Air Force to lose critical command post or satellite capabilities in a strategic crisis situation.

Fourth, software is frequently unresponsive, due mainly to the dearth of techniques for requirements analysis and design of C&C information-processing systems. For example, 95 percent of the 465L software delivered to SAC had to be rewritten to meet SAC's operational needs; 67 percent of the Seek Data II software delivered in Vietnam had to be rewritten.

C. MAJOR FINDINGS: CRITICAL PROBLEMS

During the 1980s, Air Force C&C information processing will be a key to the essential contributions of speed, precision, controllability, and flexibility required for dynamic force management and the expressed long-term goal of "surgical use of air power". In order to realize this potential for Air Force C&C, some serious problems must be alleviated. The Study Group found the following five problems to be the most critical in their need for further research and development (the CCIP-85 Study Group).

- Requirements Analysis/Design/Exercise Technology

The nation's survival and prestige rest continuously on the assumption that incidents similar to the Pueblo and Liberty incidents would not occur during grave strategic confrontations. Information processing techniques could and should be doing more in the areas of C&C system requirements analysis, system design, and system exercising to assure that this will not happen.

- Software/System Certification

The Air Force implicitly provides a guarantee to the nation that there are no errors in its command and control software that might escalate a crisis situation or seriously degrade performance during a crisis. Current software technology does not provide the highest possible confidence to back up that guarantee.

- Software Timeliness and Flexibility

Software development is on the critical path in the development of overall Air Force command and control systems. Resulting slippages of six to 12 months in system delivery are typical; often, serious compromises in software flexibility are made to prevent further slippages.

- Computer Hardware Survivability

While the hardware technology forecast reveals no serious mismatch in hardware speed and capacity, there is a serious problem with nuclear hardness. The most serious symptom of this problem is the threat to strategic missiles, which could create an unfavorable strategic asymmetry.

- Data Security

Plans for future Air Force command and control systems assume that this problem will be solved. Current technology provides no assurance that it will.

D. MAJOR FINDINGS: SIGNIFICANT PROBLEMS

These five problems also pose serious obstacles to effective support of command and control operations in the 1980s if no further R&D effort is applied; but they are not so critical as the first five.

- High-Capacity Airborne Computers

A comparison of the CCIP-85 analysis of strategic C&C information processing requirements with the CCIP-85 hardware technology forecast indicates that airborne computers of sufficient speed, size, and hardness for a 1985-era airborne command post will not exist without a dedicated R&D effort.

- Multisource Data Fusion

The TIPI (Tactical Intelligence Processing and Interpretation) system currently under development will provide an initial step toward a capability for fusion of data from many sources into useful information. To exploit this framework properly in the long run, more fundamental studies are necessary to develop and evaluate advanced automated aids to the fusion process.

- Communications Processing

This study supports and reinforces the findings of the MCT Mission Analysis that command and control operational requirements will be significantly degraded if R&D for communications processing is not increased.

- Source Data Automation

Dynamic force management is as susceptible to the "garbage in, garbage out" phenomenon as any other information processing activity. Advanced computer technology has the potential of providing considerably improved source data reliability and accuracy, as well as improved data acquisition speed, cost, weight, and volume factors.

- Image Processing

Research and development on mission-oriented image-processing functions such as change detection, outline recognition, and semi-automated aids to photointerpreters can yield near-term incremental C&C capability improvements and a base of data and insights for more fundamental future studies.

E. MAJOR FINDINGS: APPRECIABLE PROBLEMS

These five problems will pose appreciable obstacles to effective support of Air Force command and control operations in the 1980s if no new R&D effort is applied.

- Computer System Performance Analysis

Additional R&D efforts would not only pay for themselves in savings; they would also provide significant contributions to software certification and data security assurance.

- Associative/Parallel Processor Exploitation

Particularly for sensor data processing, parallel computer architectures give indications of major potential performance benefits.

- Software Transferability

Two inevitable trends are the continuing increase in C&C software inventory and the eventual upgrading of C&C computer hardware, with its attendant conversion problems. For current perspective, it will take 200 programmers working for three years (or 600 man-years) to convert SACs software to the new WWMCCS machine.

- Computer-Aided Instruction in Computing

Many problems can be alleviated by increasing the awareness among Air Force personnel of the capabilities and limitations of information processing technology. A modest problem in this area could achieve appreciable benefits.

- Hardware Destructibility

Increasingly, aircraft carrying computer hardware containing highly sensitive data will be used in operations outside the United States. Hardware destructibility capabilities must be developed to assure that such data do not fall into enemy hands.

F. REQUIREMENTS/R&D MISMATCHES

The current Air Force R&D program in information processing needs considerable enhancement and reorientation to meet the C&C requirements challenges as specified above. To make significant headway, the current \$10 million per year in information processing must increase to a level of \$25 million per year by FY 1976. This study indicates that, by the 1980s, the Air Force will be leaning at least as hard on its information structures as it will on its physical structures. However, very little R&D support is being devoted to combatting the critical upcoming software problems, compared, for example, with the level of R&D support in areas such as structures and materials.

Further, major requirements/R&D mismatches exist within the information processing field. Of the five most critical problem areas identified above, four primarily involve software technology needs. Eight of the 15 top problem areas primarily involve software (four primarily hardware, three about equal parts of each). Yet, only about 30 percent of the Air Force information-processing R&D budget is devoted to software technology.

Even within software technology, there are major mismatches between Air Force C&C requirements and R&D support. The major problems, and the major portions of the development efforts, are in the design and certification of software systems; only about 15 percent of the effort involves the writing of the computer programs. But about 50 percent of the R&D support is devoted to improving tools for the writing of programs.

G. RECOMMENDATIONS: R&D ROADMAPS

To correct the mismatches between C&C requirements and R&D support, Volume XI of the study provides a series of 18 integrated Air Force R&D Roadmaps for information processing. Fifteen of them specifically address the 15 findings stated here. In addition, Roadmaps are included for preparation for the next-generation WWMCCS computer procurement, for interservice coordination activities, and for a USAF computer hardware laboratory.

These Roadmaps provide R&D project guidelines by which the Air Force can lead information processing technology in directions that could:

- Provide more versatile, yet more economical and less manpower-intensive C&C operations for the 1980s;

- Reduce the typical C&C information-processing system development time from six to four years, and the resulting computer hardware age at IOC from three or four years to one or two years;
- Reduce significantly the danger that software errors could escalate crisis situations or degrade defenses at critical times;
- Provide survivable, high-capacity airborne computer capabilities allowing the functional equivalence of ground-based and airborne C&C operations; and
- Provide combat-ready C&C information-processing systems which are far more reliable and responsive in their support of dynamic force management requirements.

Given the Air Force's estimated \$5 billion investment in computer hardware and annual \$1 billion expenditure on software, the added R&D would be likely to pay for itself quite soon. Some representative potential savings are shown below:

	<u>Savings Per Year</u>
- Improved requirements analysis and design techniques sufficient to save one man-day of effort per man-month	\$17M
- Improved software certification techniques sufficient to save one man-day of effort per man-month	20M
- Increased software transferability by 1%	10M
- Increased software productivity from 10 to 11 instructions per man-day	100M
- Improved computer system performance analysis sufficient to realize a 25-percent improvement in hardware system efficiency on only 25 percent of the Air Force's computers	20M

Furthermore, within the Roadmaps are programs to explore and develop several very promising concepts for information processing research and development. Three such examples are:

- Semiautomated exercising of C&C systems -- An effort to automate as much as possible of the current time-consuming

manual processes involved in C&C system exercising -- for improving both current combat readiness and requirements analysis for future systems;

- Software-first machine -- A computer furnished with upcoming microprogramming capabilities which would allow it to simulate the behavior of a wide range of hardware configurations. This would allow the Air Force to begin developing C&C software before having to make an irrevocable commitment to a particular hardware configuration; and
- Structured programming -- An approach (with several current versions) to simplifying software problems by use of standardized software components and a "top-down" structuring of software functions.

(In Figure 9 of CCIP-85 Summary, it shows the total R&D investments necessary to make at least a minimum significant effort to remove the information processing roadblocks to effective support of Air Force C&C operations in the 1980s. Appropriately, software R&D, rather than hardware R&D, soon becomes the major focus. Also, the potential dollar benefits cited above indicate that these R&D investments are highly likely to pay for themselves in the long run, besides providing better C&C support. In Figure 9, CCIP-85, it suggests pure software R&D dollars to be increased from approximately \$5 million in 1973 to approximately \$20 million by 1978, and leveling at that amount through 1980.)

H. RECOMMENDATIONS: R&D LEVERAGE

Air Force R&D in information processing is only the tip of a large national R&D iceberg which should be more imaginatively exploited to meet future Air Force needs. The Air Force can realize significant R&D advantages by appropriate development of hardware and software standards, software library and tool inventory, data collection and dissemination, and related activities to orient the huge bulk of other information processing R&D activities toward critical Air Force problems. However, it is particularly important to note that, in many C&C areas, the Air Force does not have an adequate data base on how it uses information processing capabilities to support C&C operations. Without this, it would be risky to embark on either definitive standards efforts or large-scale R&D efforts in many areas. This leads to the following key recommendations:

The Air Force will most strongly improve its information processing capability to support C&C operations in the 1980s by entering now into a serious, coordinated effort to measure how its current and evolving command and control software

inventory, development efforts, and maintenance activities are distributed. This effort will build a solid foundation for powerful and pioneering Air Force R&D and standards programs during the later 1970s.

I. RECOMMENDATIONS: NEXT STEPS

- Air Force Information Processing Technology Staff Functions

An organization should be formed to provide the Air Force with the above-mentioned necessary staff functions in the field of information processing technology. At this stage, it should concentrate on such staff functions, to the exclusion of competing with existing organizations in the areas of information system research, development, operations, and management. These staff functions include user services, library and tool-inventory activities, information gathering and analysis, standards activities, interservice coordination, experimental development evaluations, and long range planning. (A recommended time-phased manpower plan for performing these functions is shown in Figure 11 of the CCIP-85 Executive Summary.)

Some portion of the initial activity of such an organization should be directed toward projects with potential early payoffs to demonstrate its feasibility. Although such an organization should support the full range of Air Force activities, several C&C problem areas provide attractive potential candidates: software/system certification, data security, information system design/analysis methodology, and computer system performance analysis.

- R&D Roadmap Preparation

Useful efforts can and should begin now to preapre the way for the recommended R&D Roadmap efforts in particularly significant areas:

- Initiate a development planning program for the software-first machine;
- Initiate a development planning program for automated system exercise aids;
- Initiate a development planning program for structured programming;
- Initiate a C&C usage study; and
- Initiate efforts to define terminology and procedures for information gathering and analysis on C&C software development and usage.

J. POTENTIAL PITFALLS

- Overreliance on Technological Initiatives

Technology alone will not solve the Air Force's C&C information processing problems. Serious management problems and institutional roadblocks -- largely identified in the Select Committee report -- must be addressed with equal vigor and perseverance. For example, current procurement and configuration management practices will need major reorientations to reflect the increasingly dominant role of software, technical advances in hardware architecture, and innovations such as structured programming and the software-first machine.

- Passiveness

The Air Force is one of the few organizations in the world which can exercise, through its R&D program, a significant amount of leverage over the pace and direction of information processing technology.

However, such Air Force opportunities are running out. Defense concerns no longer dominate even Federal government expenditures, and their share is likely to decrease in the future. The interests of the computer industry will continue to diverge toward commercial requirements and away from the interests of the military and its needs for relatively small quantities of specialized technology.

Unless the Air Force actively and imaginatively uses the leverage now available from its concentrated share of the R&D and procurement market place, it may soon find itself adapting C&C information processing systems from computer hardware and software originally designed to meet the needs of banks, insurance companies, weather bureaus, and welfare agencies. If the Air Force adopts such a passive role now, the Air Force Chief of Staff of 1985 will probably find the following on his list of recent C&C information processing pitfalls:

- Reduced force effectiveness -- The Highlights volume provides data and examples which indicate how seriously an underexercised C&C system with uncertified software could degrade Air Force capabilities.
- Rejected Air Force initiatives -- Unless better assurance can be provided that dynamic force management can be performed effectively, decision-makers may reject Air Force initiatives for weapons systems to provide such capabilities.

- Reduced deterrence credibility and national prestige --
Future command and control performance which leaves a less-than-surgical impression in the minds of leaders of other nations will reduce their image of our ability to back up our defense commitments and policies.
- Reduced human performance in C&C systems -- Lack of confidence in information inputs and processing and insufficient practice in exercising a command and control system generally lead to a highly redundant and exasperating mode of operation for the commander and his staff.
- Late delivery of C&C systems and expensive software --
Unless software R&D provide improved capabilities to match increasing C&C software requirements, slippages in schedules or critical-path software and overexpenditure of scarce dollar and manpower resources will become even more of a problem than today.

3.2 Electronics X, A Study of Military Electronics with Particular Reference to Cost and Reliability, Volume 2: Complete Report
(This report was authored by the Institute for Defense Analysis, Science and Technology Division, January 1974.)

Part C of Section IV of this report deals with the special topics of Software and Digital System Architecture. The following are the findings and recommendations of this report for those topics.¹

Findings in the Area of Software

- Software costs have exceeded hardware costs by large factors in some military systems using general-purpose computers.
- Software developments are frequently behind schedule, causing other costs to spiral.
- The complexity and extent of the software may well be a measure of the mismatch between the hardware and the problem; conversely, by properly designing and structuring the processor, the software problem can be mitigated.
- The major sources of excessive software costs in conventional systems employing central uniprocessors are the following:
 1. Selecting hardware and starting programming before the system is designed in detail -- that is, before the system functions, organization, inputs, outputs, and transfer functions are thoroughly defined.
 2. Overburdening the central processor with tasks that can be accomplished by specialized peripherals.
 3. Selecting too small a central processor, with consequent over-utilization of the computer and resort to bad programming practices.
 4. Program overintegration, which makes changes difficult.
 5. Lack of adequate discipline in software development.
 6. Developing a new high-level programming language for every job.
 7. Starting programming before the computer design is complete.

¹Much of this material is direct quotes from the Electronics X Study.

Recommendations

To reduce costs of software in processors employing conventional general-purpose machines, the recommendations are:

- Complete the design of the system and the basic program structure in substantial detail before making major commitments to hardware or coding.
- Limit the aggregation of problems to be solved on a central machine; as an alternative, decentralize processing by providing peripheral special-purpose devices (either analog or digital) or separate peripheral general-purpose machines to perform specific separable functions.
- Select a processor of adequate size to permit underutilizing the computer; write highly modular programs; emphasize structure and overall efficiency rather than hardware efficiency alone.
- Use rigorous discipline in software development, such as the top-down Structured-Programming approach.
- Use a standard well-established programming language with which programmers are thoroughly familiar. Use the highest level language appropriate to the task at hand, but avoid the unnecessary development of a unique language.
- Defer coding until the computer design is substantially complete and firm, except for that necessary to verify hardware-software design compatibility.

Findings in the Area of Digital System Architecture

- No current basis exists for the common assumption that conventional centralized programmable uniprocessors are the most effective or most economical basis on which to structure military tactical data systems.
- The cost of programming is escalating, while the cost of standard computing hardware is plummeting; a new look is needed at the balance between hardware and software in system architecture.
- The advent of large-scale integration has led to the cheap and plentiful implementation in hardware, on single chips, of standardized complex algorithms together with memory.

- There is a growing library of these hardware-implemented, standard, complex computing functions that makes possible the synthesis of specialized processing units and the elimination of much of the software. The low cost and small size of these units mitigate the need for time-sharing their use, and permit distributed processing.

Recommendations

- System-function-oriented processing-hardware structures should be considered as alternatives to the conventional centralized programmable uniprocessor for use in military tactical systems.
- The military processing problem should be clearly stated; the system design should be spelled out in detail; and alternate processor architectures and designs should be compared before a hardware approach is selected.
- A processor design for each system should be selected and developed that will minimize the combined costs of hardware and software; the allocation of functions between hardware, software, and human operators should be consciously worked out prior to decision.
- Standard large scale integration (LSI) processing elements available from more than one source should be used to the maximum extent possible; development of uniquely military LSI elements should be minimized.
- Military laboratories should be encouraged to investigate and develop processor architectures, including federated architectures, that fit military problems and are cost-effective.
- Commercially successful processors for which software already exists should be considered for DOD applications wherever appropriate.
- Formats and speeds for data interchange among sensors, actuators, processors, controls, and displays should be standardized across Service lines and for as wide a variety of applications as practicable.

3.3 Automatic Data Processing Costs in the Defense Department

(This paper was authored by David A. Fisher, Institute for Defense Analysis, October 1974.)

"This paper limits itself to the development of estimates of the costs and cost trends of DoD computer software and other ADP activities and the major components of those costs." Its approach is to develop lower bounds on costs. It is based primarily on the "Inventory of Automatic Data Processing Equipment in the United States Government," published by GSA. Its treatment of weapons-system costs is by analogy and estimation based on line items in the DoD budget. Weapons Systems computers are included as part of ADP.

A summary of its findings is:

TOTAL DOD ADP COST ESTIMATE, FY 1973 (dollars in billions)

	<u>Air Force</u>	<u>Army</u>	<u>Navy</u>	<u>Other DoD</u>	<u>DoD Total</u>
Software	\$1.0-\$1.3	\$0.7-\$0.8	\$1.0-\$1.3	\$0.2	\$2.9-\$3.6
Hardware	\$0.4-\$0.5	\$0.3	\$0.3-\$0.5	\$0.1	\$1.0-\$1.4
Other ADP	<u>\$0.8-\$1.2</u>	<u>\$0.6-\$0.8</u>	<u>\$0.8-\$1.2</u>	<u>\$0.1-\$0.2</u>	<u>\$2.3-\$3.3</u>
Total	\$2.2-\$3.0	\$1.5-\$1.9	\$2.2-\$3.0	\$0.4	\$6.2-\$8.3

Of these totals, those unreported in the GSA Inventory (largely weapons systems) account for:

Software	\$0.5-\$0.7	\$0.2-\$0.3	\$0.5-\$0.8	\$.04-\$0.06	\$1.3-\$1.9
----------	-------------	-------------	-------------	--------------	-------------

From the paper: "The principal findings of this paper may be summarized as follows:"

- "- Reliable information on most software and ADP costs in DoD is unavailable in a clearly identifiable form. The following data for FY 1973 were developed in this paper."
- "- Documented and identified annual ADP costs in DoD were \$1.5 billion in FY 1973; this rises to \$2.3 billion when an estimated cost of personnel burden is included."
- "- The estimated annual ADP costs in DoD are \$2.9 billion to \$3.6 billion for software and a total \$6.2 to \$8.3 billion when hardware and other ADP are included (approximately 30% to 50% of all electronics costs in DoD)."

"- ADP costs in DoD are apportioned approximately as follows:

Software	45%
Computer Hardware	16%
Other ADP (includes computer operation, key punching, support and supplies)"	38%

"- An estimated 70% of ADP costs in DoD are for personnel."

"- The number of ADP man-years in DoD is divided almost equally between:

- Software systems analysis, design, and programming
- Operation of ADP equipment (except key punching)
- Services, support, and key punching."

"- The annual costs of Air Force software and computer hardware estimated in this paper and in CCIP-85 are similar. This is surprising because the CCPI-85 cost estimates are based entirely on analogy with industry and are not well documented."

"- A comparison of DoD ADP costs reported for FY 1968 and FY 1973 shows that:

1. Total ADP costs and total ADP personnel salary costs remained unchanged, while costs per system rose 4% to 5%.
2. Total ADP contract service costs rose 54%, or 61% per system.
3. Rental and capital costs for ADP equipment dropped 8%, which is 5% per system.
4. The total of in-house ADP man-years dropped 10%.
5. There was a shift from use of in-house personnel to contract services for system analysis/design, programming, and maintenance.

6. There was a shift from rental to purchase of ADP equipment.

7. The number of computer systems increased 28%, while the number of systems reporting ADP operational and capital costs and personnel activities declined."

Availability of Data

The paper reports that "there is no definitive way of describing ADP," that the terminology of existing data sources is "often vague or ambiguous," and that sources "report ADP costs and activities as ADP only if they fit into no other reasonable classification."

The paper discusses inconsistencies in the GSA Inventory for 1973. For example, WWMCCS computers are in the General Management category, SAGE computers in Special Management, and no store-and-forward message switching systems are included. Figures for programming services reported on DD Form 350 (DoD procurements and contracts in excess of \$10,000) do not match inventory figures for contracted systems analysis, design and programming.

Documented Costs

The documented costs referred to are primarily conventional ADP (i.e., non-weapons systems).

Estimated Costs and Cost Apportionment

Estimated costs are based primarily on costs for General Management Classification computers. An average annual cost-per-computer system for software is derived (\$.467M) and multiplied by the number of Special Management Classification computers (1195) to yield total software costs for SMC computers (\$551M).

For weapons systems the report states: "There is no accurate way to estimate software costs for computer systems not reported in the Inventory . . . Costs for individual systems can be obtained, but there is no practical way . . . to estimate total costs." It develops costs by analogy with industrial firms doing similar work. "The proportion of total costs attributable to software for a given application area are those for industry used by the CCIP-85 authors They are not necessarily correct, but we have no better figures." The figures discussed state that software costs:

3.4% to 5.5% of aircraft and missile procurements

8% to 12% of intelligence and C³ costs

7% to 10% of RDT&E costs.

A full set of information is included in Table 3-1.

In the personnel area, estimates of in-house costs apportionment are made based on civil service pay scales and estimated average salaries of military and civilian personnel to arrive at a distribution to total costs among computer operators, computer aides (e.g., keypunch) and systems analysis/design and programming. Burdened costs are estimated after calculation at \$23,310 per man year, which is called "conservative" compared to CCIP-85 estimates of \$30,000 to \$40,000 per man year.

Software costs (45% of total) are taken to include in-house personnel, contract services and a portion of hardware costs allocated to software support.

Comment

The report is careful to caution that care must be taken in interpreting cost and man-year figures, since they are indirect measures. It states:

"That largest costs and numbers of man-years in ADP are attributable to software systems design, analysis and programming and that software is expensive and labor intensive. The data does not explain why or whether it is unduly so or how one might reduce the cost and effort associated with software."

The report documents its assumptions well, breaks down all figures to Air Force, Army, Navy and other DoD contributions, and gives a large number of intermediate calculations.

TABLE 3-1
ESTIMATED DOD SOFTWARE COSTS, COMPUTER SYSTEMS
UNREPORTED IN GSA INVENTORY, FY 1973
(dollars in millions)

	Air Force	Amy	Navy	Other DOD	DOD Total
<u>Aircraft, Missile, & Ship Procurement</u>	\$4,325	\$814	\$6,635	--	\$11,775
Software @ 3.4%-5.5% of Above	\$147-\$238	\$28-\$45	\$226-\$365	--	\$400-\$648
<u>Intelligence & C³</u>	-- ^a	-- ^a	-- ^a	-- ^a	-- ^a
Software @ 8%-12% of Above	--	--	--	--	--
<u>ROUTE</u>					
Budget	\$3,120	\$1,885	\$2,542	\$446	\$7,995
Estimated In-House Personnel	1,000	605	816	143	2,566
Total	\$4,120	\$2,490	\$3,358	\$589	\$10,559
Software @ 7%-10% of Above	\$288-\$412	\$174-\$249	\$235-\$336	\$41-\$59	\$739-\$1,056
<u>SOFTWARE SUBTOTAL, ABOVE THREE ITEMS</u>	\$435-\$650	\$202-\$294	\$461-\$701	\$41-\$59	\$1,139-\$1,704
Software Cost of Hardware @ 15% of Above	\$65-\$98	\$30-\$44	\$69-\$105	\$6-\$9	\$171-\$256
<u>SOFTWARE TOTAL, SYSTEMS UNREPORTED IN GSA INVENTORY</u>	\$500-\$748	\$232-\$338	\$530-\$806	\$47-\$68	\$1,310-\$1,960

^aValue unknown.

Note: This table is a copy from IDA Paper P-1046, ADP Costs in the Defense Department, dated October 1974.

3.4 The High Cost of Software (Proceedings of a Symposium)

(The symposium was sponsored by the Air Force Office of Scientific Research, Army Research Office, and Office of Naval Research, Monterey, California; 17-19 September, 1973. The proceedings were published by the Stanford Research Institute, September 1973.)

The objective of the symposium was to consider what research is needed to achieve a major reduction in software costs. Attendance was by invitation. The 97 attendees were organized into five workshops.

The attendees were in strong agreement that direct and indirect software costs are unnecessarily high and are growing rapidly, that they constitute a serious limitation on the effectiveness of information-processing systems, and that the high cost is a consequence of the poor state-of-the-art of software design, production, and maintenance. There was a strong feeling of urgency that an energetic program of research be undertaken to advance the software art.

The proceedings of the symposium contain reports by the workshop chairmen in five areas:

- Understanding the software problem
- Contributions from the semantics of language and systems
- Advances in software methodology
- Software-related advances in computer hardware
- Approaches to the problems of large systems

These proceedings also include a summary of a keynote speech on the high cost of software and a report of a panel on software technology transfer. Five tables summarizing the workshop recommendations appear at the end of this summary.

Co-chairpersons' Summary

The summary produced by the chairpersons of the symposium made the following conclusions (among others):

- Maintenance costs for old software may be an order of magnitude larger than the production cost, due to poor original design and production.
- Incompatibilities between computers results in costly reprogramming and the inability to take advantage of hardware advances.

- Experience has shown that determined efforts to remedy poor management practices (e.g., inadequate production control, programmer versatility, failure to acquire available tools and modern hardware) have been successful in reducing costs and improving quality.
- Program complexity depends on problem complexity; methods of characterizing large problem classes in general ways could reduce costs "enormously."

Keynote Address

Some of the points made in B. W. Boehm's keynote address are:

- Historically, avionics software has cost \$75 per instruction for development; in some cases maintenance may cost as much as \$4000 per instruction.
- The percentage of development effort spent in each of the three phases of software production (i.e., analysis and design, coding and auditing, and test and integration) depends upon the type of software. Command-and-control and avionics software differ markedly in this respect from business and scientific programming (see Table 3-2).
- Very little hard data is available on software costs, and it is of varying applicability to DoD problems.
- Individual programmers vary in productivity by a ratio of 10 or 20 to 1.
- The one factor that correlates well with success of a software project is the degree to which the leaders are pragmatic and knowledgeable about software.
- Many DoD standards on documentation and test procedures are incompatible with top-down design and structured programming.
- TRW found that on large projects their software tools (i.e., PACE and ASIST) would help find errors on the average 4 to 5 months earlier.
- Severe memory constraints drive up software costs dramatically. Procurement decisions should recognize software and not hardware as the dominant cost, and hardware should be sized accordingly.

TABLE 3-2
SOFTWARE EFFORT DISTRIBUTION BY ACTIVITY

	Analysis and Design (percent)	Coding and Auditing (percent)	Test and Integration (percent)
Command-Control (SAGE, NTDS)	35%	17%	48%
Command-Control (TRW)	46	20	34
Spaceborne (GEMINI, SATURN)	34	20	46
General Purpose Executive (OS/360)	33	17	50
Scientific (TRW)	44	26	30
Business (RAYTHEON)	44	28	28

Note: Additional effort for documentation: 9-100 percent.

NOTE: This table is a copy from The High Cost of Software Symposium.

Workshop 1: Understanding the Software Problem

The recommendations of this workshop appear after this summary as Table 3-3.

Workshop 2: Semantics of Languages and Systems

In addition to its recommendations (see Table 3-4), this workshop report made the following points:

- No fully automatic verification system will be feasible for practical purposes for many years. Some nearer-term payoff may come, however, in the form of guidelines for program construction and semi-automated aids for testing.
- Programs that depend on operating system features and peripherals are very hard to transport, even if they are written in a standard language.

Workshop 3: Programming Methodology

In addition to its recommendations (see Table 3-5), the workshop report made the following points:

- "The key to achieving understandability appears to be structure, i.e., the partitioning of a program into parts and the organization of the parts into levels of abstraction . . ."
- Application of any of the current methodologies (i.e., top-down design, modular decomposition, formal specification, etc.) would be unwise unless appropriately supported - both organizationally and technically.

Workshop 4: Software-Related Advances in Computer Hardware

In addition to its recommendations (see Table 3-6) the report on Workshop 4 made the following points:

- Many computing problems result from poor communications between the hardware and software communities.
- It is no longer possible to design the hardware separately from the software.
- Appropriate hardware architectures can help alleviate software complexity problems.

TABLE 3-3

RECOMMENDATIONS OF WORKSHOP 1:
UNDERSTANDING THE SOFTWARE PROBLEM

Research Activity	Benefits
<p>Improve understanding of Department of Defense software costs.</p> <ol style="list-style-type: none"> 1. Determine distribution of present and anticipated software costs by activity, application, and life cycle phase. 2. Establish pragmatic measures of and measuring techniques for programmer productivity. 3. Establish an ad hoc group for survey of software producers and practices. 4. Establish a medium for software information exchange. 	<p>Improved priorities for funding of R&D and technology transfer.</p> <p>Better characterizations of costs, e.g., fundamental/derived, present/future, management/technical, general/application-dependent.</p> <p>Development of standards for planning and control.</p> <p>Rapid collection of useful data and stimulation of awareness of the software problem.</p> <p>Propagation of good present and future ideas and practices. Encouragement of cooperative ventures.</p>

NOTE: This table is a copy from The High Cost of Software Symposium.

TABLE 3-4

RECOMMENDATIONS OF WORKSHOP 2:
SEMANTICS OF LANGUAGES AND SYSTEMS

Research Activity	Benefits
<p>Advance fundamental understanding of program and system semantics.</p>	<p>Creation of opportunities for breakthroughs and guidance of future short-range tool building.</p>
<p>1. Develop the theory and tools for formal verification and proof of program properties.</p>	<p>Reduction of program testing costs and assurance of correctness of the computing function in large systems.</p>
<p>2. Study the semantic properties of programming languages and systems.</p>	<p>Vital guidance to language designers and computer architects. Drastic reduction of programming costs through more unified and coherent realization of essential functions.</p>
<p>3. Develop the concept and practice of knowledge-based systems for important application domains.</p>	<p>Drastic reduction of programming costs in specific application domains.</p>

NOTE: This table is a copy from The High Cost of Software Symposium.

TABLE 3-5

RECOMMENDATIONS OF WORKSHOP 3:
PROGRAMMING METHODOLOGY

Research Activity	Benefits
<p>Develop the theory and practice of structure-oriented programming methodology, and develop understanding of human factors in programming.</p>	<p>Practical tools and procedures for coping with the problems of large software systems.</p>
<p>1. Study, develop, and integrate the several attractive methodologies for structuring software systems, including top-down design, modular decomposition, and black-box.</p>	<p>Enhanced human understanding of complex programs, resulting in more correct, robust, modifiable, and efficient programs.</p>
<p>2. Develop tools to support and implement present and future software methodologies. The tools should be technically adequate and well engineered by individual and social human considerations.</p>	<p>Improved interaction of theoretical and practical work, leading to the development of more powerful tools for field use.</p>
<p>3. Investigate new concepts for objective evaluation and testing of programs and the programming process.</p>	<p>Improved planning, procurement and control of programming services.</p>

NOTE: This table is a copy from The High Cost of Software Symposium.

TABLE 3-6

RECOMMENDATIONS OF WORKSHOP 4:
SOFTWARE-RELATED ADVANCES IN COMPUTER HARDWARE

Research Activity	Benefits
<p>Develop new computer architectures that enhance software practice in all phases, including creation, debugging, operation, and transfer.</p>	<p>Exploitation of the rapid advances in hardware technology to reduce software costs.</p>
<p>1. Develop techniques for improving the hardware/software interface, including:</p> <ul style="list-style-type: none"> (a) High-level, application-specific machine primitives. (b) Descriptor architectures. (c) Virtualizable architectures. 	<p>Simplified programming, including:</p> <ul style="list-style-type: none"> (a) Better debugging and reduction of machine-oriented artifacts in programming languages. (b) Greatly simplified programming for data-intensive computations. (c) Easier transferability of programs among machines, easier debugging, and better security.
<p>2. Investigate the potentials and problems of distributed processing architectures.</p>	<p>Anticipation of software problems of future high-powered computers.</p>
<p>3. Develop architectural techniques that enhance the observability of the state of generalized data structures.</p>	<p>Better debugging.</p>
<p>4. Develop the use of system definition languages.</p>	<p>Better integration of hardware and software.</p>
<p>5. Foster the study of architecture at universities and research laboratories.</p>	<p>Stimulation of innovation; increased understanding of hardware/software relationships.</p>

NOTE: This table is a copy from The High Cost of Software Symposium.

- More data and study are needed to guide hardware/software trade-offs, and to guide the decision of when to use firm-ware.
- "It is a widely held opinion that the definition phase of a complex software development is the one for which we have the most inadequate understanding and tools. It is also generally agreed that many of the problems that plague later phases of the product have their root cause in the definition phase."

Workshop 5: Problems of Large Systems

This workshop dwelt more directly with the considerations of military software than did any of the other workshops. In addition to its recommendations (see Table 3-7), the following points were made:

- Large military software generally differs from other kinds of software in the following ways:
 1. The designers can choose their hardware and operating system.
 2. The software has more unknowns at the start of the design process, since the external system is not yet fully specified.
 3. The software production schedule is locked into other system procurements.
 4. The production team is larger and the production time is longer.
- In practice, definition and design are deeply entwined and badly confused, partly because of complexity and indeterminacy and partly because of poor management.
- There is a general lack of tools to analyze trade-offs, such as speed vs. flexibility. Prototyping (which is the rule in hardware systems) would help and should be allowed for.
- A serious source of problems is that often the software part of a system is assigned problems for which solutions have never before been devised.
- Premature "freezing" of system structure into management structure causes problems.

TABLE 3-7

RECOMMENDATIONS OF WORKSHOP 5:
PROBLEMS OF LARGE SYSTEMS

Research Activity	Benefits
<p>Develop methods that enhance the practice of managing large software systems.</p> <ol style="list-style-type: none"> 1. Develop management tools based on realistic models of the design and implementation processes. 2. Develop requirement languages for specifying systems independently of implementations. 3. Develop improved architectures that absorb resource management functions for operating and data base systems. 4. Develop automatic software testing tools in all areas. 5. Develop methods for integrating sets of software tools and for preserving successful techniques in changing environments. 	<p>Management practices attuned to real technical factors of software design and production.</p> <p>Assurance of the effectiveness of management controls.</p> <p>Reduction of the cost of changes.</p> <p>Reduction of programming cost by removing functions that are not application-oriented.</p> <p>Reduction of the cost of ensuring software quality.</p> <p>Minimization of the overhead cost of using individual tools and building on previous good techniques.</p>

NOTE: This table is a copy from The High Cost of Software Symposium.

- Techniques are needed to improve transferability of implementation tools.
- Overcontrol and excessive demand for documentation are common management errors.
- Configuration control can be applied too early and at too detailed a level to allow for the continuing design that must go on.
- Field requirements of military systems force new hardware to be developed, with a resulting lack of software tools, peripherals, and a capability for reconfiguration. "Application design is chancy in such an environment."
- The procedures usually followed in government contracted software are not flexible enough to allow significant trade-offs to be made during design in a timely fashion.
- Tight configuration control should be held off until major design questions have been found.
- Excessive detail, revealed too soon, swamps the review teams.

3.5 Government/Industry Software Sizing and Costing Workshop
(Sponsored by the Air Force Systems Command, Electronic System
Division, L. G. Hanscom Field, Bedford, Ma., 1-2 October 1974.)

Formal proceedings of this workshop have not yet been published; the data reported here is based on draft notes by the workshop chairman and one of the workshop panel chairmen. The notes present viewpoints and opinions by industry, academia, government (primarily Air Force) and FCRC participants. No source data is quoted.

Major Conclusions

The notes start with two questions: What are the attributes of a good software requirements specification? and What are the dominant factors that affect software costs? They proceed to conclusions over a wider area:

Cost estimating is only part of the problem; cost estimating methods can be identified (with reservations), and difficult changes to the acquisition procedures are required. The changes include:

- A multi-step or multi-phased procurement approach with separate software contracts considered.
- Improved, more formal requirements specification practices, possibly leading to formal requirements languages.
- More standard terminology, improved work breakdown structures, good historical cost data.

Requirements Specification

The requirements specification is defined here as the specification which accompanies an RFP. A number of deficiencies are noted:

- Critical performance goals (e.g., maximum load, minimum response time) and allowable trade-offs are not communicated. Evidence given is the common experience of a 5:1 ratio of high-to-low bid in response to an RFP.
- Some data required in an RFP is not present:
 - Distinctions between minimal, nominal and maximal requirements.
 - Distinctions between required and recommended design.

- Distinctions between performance and design.

The expense of accurate cost estimating is discussed in general terms, concluding that "in order to accurately predict software costs for a project, one must do a considerable amount of design work in addition to project planning".

Acquisition and Procurement Strategy

The distinction between software and hardware phases is described as:

<u>Software</u>	<u>Hardware</u>
Design	Gross Design
Implement	Detailed Design
Test	Design Validation
--	Production
--	Operation/Use

with the idea that the software analogy to hardware is or can be defective.

The phased procurement strategy recommended is given in more detail as: "Dual development by two contractors for the preliminary design and detailed cost estimate to produce, test, operate and maintain the software, with recompetition for actual software development". A separate software contractor is also suggested, apparently to allow software houses to bid separately on parts of hardware and software developments.

In another area, unrealistic milestones are claimed to be a problem; in particular Preliminary Design Review 90 days after a contract award is claimed to be unrealistic. "The milestone schedule should be determined by the bidder based on his experience and methods of software development."

Costs

Some software cost estimating factors are included in a chart of "factors influencing software costs" (Table 3-8); most are relative, e.g., an operating system costs 2.5 times an application or utility program; testing costs 15%-40% of system¹ cost; documentation 10%-25% of system cost.

¹ Note: System referred to is apparently the software and not the entire weapon system.

AD-A034 807

MITRE CORP MCLEAN VA

F/G 9/2

DOD WEAPON SYSTEMS SOFTWARE ACQUISITION AND MANAGEMENT STUDY. V--ETC(U)

JUN 75 A ASCH, D W KELLIHER, J P LOCHER

MTR-6908-VOL-2

NL

UNCLASSIFIED

2 of 2
ADA034807



END

DATE
FILMED
2 - 77

Specific dollar amounts include:

Cost per <u>source</u> instruction:	Higher Order Languages \$6-\$12
	Machine Oriented Language \$12-\$24
	"Real-Time Software" \$20-\$60
Documentation:	Per page (non-automated) \$35-\$150

Prototyping

Complex software systems should be procured like complex hardware systems. Systems requirements cannot be fully and correctly specified before some operational experience has been gained with a system similar to the one to be procured. Software procurements should be preceded with the building of a software breadboard to explore high risk parts of the potential system, or a software prototype to explore tradeoffs, to refine requirements, to evaluate design tradeoffs, and to permit realistic feasibility studies.

Cost Estimating

A number of methods of cost estimating are named with their advantages and disadvantages. The attached Table 3-9 gives the list.

TABLE 3-8
FACTORS INFLUENCING SOFTWARE COST

FACTOR	RELATION TO COST
Number of delivered source instructions Language	Linear, modified by other factors HOL: \$6-12/source instruction MOL: \$12-24/source instructions
Real-time application Type (OS, application, utility) Point on learning curve Application area (MIS, avionics,...)	RT: \$30-60/source instruction If OS, multiply by 2.5 If unfamiliar, multiply by 1.5-2.0 Sometimes, as percentage of total system cost
Desired Quality	"Man-rated": test cost 40% of total Non-"man-rated": test cost 15% of total
Turnaround Time	Approximately linear relation to testing cost
Amount of documentation	Approximately 10% of total; \$35-150/non-automated page
Hardware constraints	Asymptotic as approach full capacity
Schedule realism	Percent added cost = percent of schedule acceleration
Amount of previous software used	Breakout and subjective
Size, structure of data base	Subjective
Complexity	Subjective
Stability of requirements	Subjective
Stability of development environment	Subjective
Representativeness of development environment	Subjective
Personnel	Subjective; approximately 5:1 variability
Development methods (e.g., structured programming)	Subjective; systematic approaches cheaper
Management	Subjective; high variability

NOTE: This table is a copy from the Government/Industry Software Sizing and Costing Workshop.

TABLE 3-9

COST ESTIMATION METHODS

<u>METHOD</u>	<u>GOOD POINTS</u>	<u>WEAK POINTS</u>	<u>WAYS TO IMPROVE</u>
FACTORS EXPERTS (GROUP, DELPHI)	QUANTITATIVE RELATIONSHIPS SUBJECTIVE PARTS	SUBJECTIVE PARTS NO BETTER THAN PARTICIPANTS	MORE DATA COLLECTION AND ANALYSIS
RATIO TO PREVIOUS	BASED ON REAL EXPERIENCE	DEGREE OF REPRESENTATIVENESS	BETTER DATA COLLECTION AND ANALYSIS
RATIO TO TOTAL SYS \$	BASED ON REAL EXPERIENCE	DEGREE OF REPRESENTATIVENESS	BETTER DATA COLLECTION AND ANALYSIS
PARKINSON (H/W, LOE)	CORRELATES W/SOME EXPERIENCE		
COST-TO-WIN	OFTEN WINS	ALWAYS OVERRUNS	BETTER SOURCE SELECTION, FEEDBACK
PROBABILISTIC	REFLECTS REALITY	HARDER TO SELL	MORE USAGE

NOTE: This table is a copy from the Government/Industry Software Sizing and Costing Workshop.

3.6 Proceedings of the Aeronautical Systems Software Workshop

(This workshop was sponsored by the Air Force Systems Command, Aeronautical Systems Division, Wright Patterson Air Force Base, Dayton, Ohio, 2-4 April 1974.)

The program for the Aeronautical Systems Software Workshop was divided into two sections. The first section was a DoD section only with 7 invited papers. The second section was composed of 60 contributed papers from industry and government alike. The following is a summary of the 7 DoD papers followed by a summary of most of the 60 contributed papers.

Colonel Leo Danielian gives the background leading to the establishment of program PACER FLASH. He identifies nine software problems associated with operational flight programs (OFP), automatic test equipment (ATE), and simulation (SIM). These were: lack of standards, no validation (with exception of the B-1), bad contractor reliability, lack of software management plans, difficult maintainability, high support and contractor costs, poor responsiveness (a change to four to six tapes for the F-111 OFP took 8 months), inadequate planning, and poorly defined transitions. He then discussed actions being taken including the AFM 800-XX, the F-111 Dynamic Flight Simulation Implementation Plan and the software development plan to be produced for all AWACS software.

Lt. Colonel Hinton discusses OFP, their functions, their software characteristics, the problems, and the current approaches to the problems. Systems using OFP include the B-1, and the F-111. He cites as the major problems inadequate requirements specifications, increasing program size and cycle time, documentation inadequacy, and finally the lack of qualified software engineering personnel. OFP support problems include supportive software, test bench hardware, and documentation. In terms of the magnitude of the problems, there are approximately 50 airborne computers today with OFP maintenance costs of \$10 million. FY-74 OFP development costs are \$50 million which implies that in 5 years maintenance costs will be \$50 million. Current approaches to solving the problems include B-1 "milestone" event tracking as well as in-house personnel to monitor software development in detail.

Mr. Behymer briefly describes Automatic Test Equipment (ATE) and its software including a brief synopsis of historical trends leading to today's extensive use of computers in performing field shop and depot maintenance. The paper discusses the development cycle for ATE software and its support software, e.g., compilers. ATE programming languages and their standardization are also discussed. He recommends ATLAS be standardized.

He discusses the digital-module-testing current common practice of developing special computer programs to produce test patterns and resulting responses for the modules. He identifies as "management" challenges in ATE software early identification of the software, thorough transition planning. The paper mentions "people" problems, namely, the need to re-educate traditional hardware managers in software management and he cited there are technical personnel shortages.

Mr. Babel's paper on crew training simulator software points out the criticality of the computational system in the training system. He briefly describes the A-7D simulator and relates it to preceding and succeeding simulator procurements and the maturing in these procurements.

He describes in some detail the software development process including his set of milestones used to control the process and points out the simulator software support plan is initiated concurrently with the definition of the software requirements in accordance with MIL-STD-876A. He describes the current post delivery change control process by means of a "prototype" center. The most critical challenge in simulator software acquisition management is pointed out to be again a shortage of software engineering skills. He also points out the need for management awareness and the requirement for AFSC and AFLC to work closely together.

Mr. Schmidt's paper discusses the responsibilities of AFLC in software management and some of the activities currently underway within AFLC including development of facilities to support some weapon system avionics software, and the "organic" (in house) support of ATE software.

He points out that while software has been rapidly growing within the Air Force in the past 5 years, this has been coincident with major reductions in personnel force which has made hiring of these special skills quite difficult. Also he points out the consequences of acquisition economies on operational phase support costs. He suggests the possibility of requiring a configuration management scheme for software differing from hardware because of software intangibility.

Other problems mentioned in the paper include inadequate coding standards, the use of machine languages and failure to identify support requirements early in the program.

Lt. Col. Daye is currently the AFSC full time manager for the development of a manual which is being developed to support the new Air Force Regulation 800-14.

Lt. Col. Daye's paper describes the techniques being used to develop the manual, including the participation of all involved Air Force agencies. The subject matter expected to be contained in the manual is described in some detail.

The primary purpose of the manual is to provide guidance in implementation of the Air Force policy contained in AFR 800-14 and, therefore, as Lt. Col. Daye points out, there are chapters on the life cycle of computer programs, planning for this life cycle, the contractual aspects of management and technical tasks, the software documentation techniques, the related functions of systems engineering and configuration management, the turn-over and transition phase as well as the maintenance and support phase.

Colonel Fernandez's paper emphasizes current actions under way at Hq AFSC to cope with the aeronautical systems software problems. He points out that they are aimed at obtaining greater visibility into the problem and then generation of effective action. Among actions already taken, he discusses the formation last fall of the office he holds in DCS/Development Plans as well as an AFSC Software Steering Group to focus attention on this area; the integration of AFSC critique of the PACER FLASH reports; the requirements for inclusion of software development status reports in program reviews; the participation in the development of AFR 800-14; the formation of the Information Systems Technology Applications Office at ESD; and information systems technology newsletter; as well as the Workshop represented by these proceedings. Actions under way, but not yet complete, which the paper discusses include: a functional overview of computer technology; improvement of personnel tools for identification of critically short personnel for the software area; development of a methodology for capturing software cost data; and the previously mentioned AFM 800-XX preparation.

At the conference there were 60 contributed papers. They can be categorized into logical groups according to the message in each.

Life Cycle Management Considerations

The first group of contributed papers deals with the complete process of planning, developing and maintaining aeronautical systems software over its entire life cycle. The principal items discussed were developing cost-effective, reliable software together with some suggested solutions.

The first paper by Davis presents some "controversial points of view on the requirements and nature of software visibility, areas

of responsibility, and the problems of acquiring the needed people." Schulz discusses the need for a union of hardware, software and systems engineering and the deficiency in our educational system with respect to formal training of software engineers. In the next paper, a number of topics such as current problems, standards, support software, management and suggested solutions are offered by Trainor. Included in his list of problems for avionics software are: poor software development standards (he cites the \$290,000, 70 word modification to an OFP as a result of such a problem), "Hardware Priority" philosophy, costly and unique support software, and inadequate management. His solutions included the standardization of OFP structural design, software production, HOL programming, and hardware design for software utilization (consideration of software while designing hardware). Schiff follows this by pointing out that "past experience indicates that the current approach to software procurement and software program budgeting neglects actual life cycle costs and in doing so may lead to significant underestimates of the actual cost of operational software." Suggested new approaches include structured programming, top-down development, development support libraries, and HIPO techniques. Finally, Babel addresses the controversial subject of whether or not to utilize inflight computers in crew training simulators.

Software Development: Engineering and Management

Although not quite as general as the papers in the preceding group, the nine papers taken together comprehensively discuss the major problems concerned with software development from both engineering and management viewpoints.

Boehm begins by advocating "careful thought, thorough planning, good engineering judgment, and sound management" as some of the keys to successful software development. He also asserts that the use of concepts such as structured programming should not be used simply in a "cookbook fashion". Boehm provides a table that compared alternative software design techniques. He gives an analysis of errors on a large, good software project which he feels supports his emphasizing the necessity of more work at the software design and structuring stage. Schlight stresses the need for objective project control and user involvement with the developer. He proposes that the CSC automated "threads" system will satisfy these objectives. Charles agrees in principle that a systems approach toward software development is necessary but also points out differences in software design and implementation problems in the areas of training and system simulation software.

The last six papers in this group concentrate more on the management of software development than on the engineering aspects. Mangold points out that "good management is embodied in six functions which tend to stabilize the process." They are: performance of a complete preliminary design, involvement of the customer and user, insurance that the documentation is current and complete, the provision of disciplined test planning and implementation, a guarantee of effective product control, and preparation of operational errors. McNeely feels that "control of software projects by conventional software management techniques is no longer sufficient." He stresses the need for "software engineering management" based on "sound and realistic engineering principles."

Allen describes the current aeronautical systems software program as it exists at the Aeronautical Systems Division. He makes the observation that of the 25 members of the newly formed Software Engineering Branch, 50% of them have less than one year's experience in engineering. He discusses some programs that ASD is involved with, including the approach to software tasks as well as development problems encountered and lessons learned. Included in this list is the B-1 and F-111. For the F-111, hardware development problems played havoc with the software development. For the B-1, a tight schedule including a late start, as well as lack of communications has created problems for software development. It has been aggravated by the highly integrated design of the B-1.

Hartwick follows with an industry viewpoint that the Air Force acquisition management process might be improved in the areas of documentation, technical reviews and change processing, among others. Wenk provides additional industry suggestions for proper management of resources and personnel that can increase productivity and hold software costs to a minimum. He attributes 40% of software costs to design, 20% to coding and 40% to checkout. A third industry viewpoint on software development, this one involving techniques to increase operational reliability of real-time programs within a shorter schedule, is offered by Kulp. Included in his techniques were first the RCA CPDAMS (Computer Program Development and Management System), which is the system they use to bring programs up and get them running, and Loop Testing which is their testing strategy.

Software Configuration Management

This set of three papers concentrates on software configuration management, or the control of changes to software. Charnell leads off by describing the need for configuration management

and how his firm is attacking the problem through the use of the manual "Unit Development Folder" and the automated "Master Development File" as parts of a comprehensive "Software Accounting System." Teutsch and Klammer describe in detail how their company has processed software changes for the F-14A/AWG-9 Weapon Control System. Wolverton's paper addresses standards. His paper examines "government rules and regulations which influence the contractor's technical approach, his software development procedures, and his product control in view of the dilemma presented by proliferating -- and often contradictory or silent -- government procurement procedures." He maintains that proliferating procurement procedures can and often are impediments to good software. He includes charts on the development of standards and a case study of the adherence to selected practices. Developments that could solve some of the problems are TRW's Site Defense Program "software standards and procedures" as well as other advanced concepts such as structured programming, top-down programming, problem statement languages, software first design, and model driven software design and automated requirements analysis.

Support Software

According to Wilson and Scallon, "support software is now changing from its traditional image as loosely related non-deliverables and is emerging as a recognized discipline." For example, expanded support capabilities are being used on the B-1 Avionics Program and have proven essential to a systematic and disciplined approach. They contend that larger percentage cost reductions would result from increased work in the software support field than from continuing efforts in the implementation and development fields.

Irvine asserts that "any careful analysis of the full life-cycle costs of software will reveal that the bulk of these costs arise during the "maintenance phase" and that "the problems associated with avionics software development lie in the cost and quality of support software . . ." More emphasis on support software should reduce the overall total cost of a system. Some examples of support tools given are Softech's Structured Analysis, Directed Flow Graphs, and AED. A main goal of these systems is the reduction of the cost of changing software. Brown presents arguments to support his conclusion that "the current trend toward more tools, tools that are better than ever and more flexible to support general applications, is going to continue." He claims that use of a combination of tools such as METAFORE, PACE-I, COMGEN, DOCCEN, and AUTOFLOW can reduce costs of producing software by 50%.

Stucki gives a report on the APSS (Automated Program Support System) used at McDonnell Douglas. It contains five parts: program development tool, test statistics tool, documentation tool, simulation tool, and a test data generator tools. Willis emphasized the use of models during the conceptual phase of software development pointing out the advantages of predicting problems as well as the difficulties of actually building the models. He gives three examples of models. Maher's paper in particular is extremely comprehensive by summarizing a recent survey of digital simulation techniques, facilities and their possible application to the F-111.

Software Maintenance

Maher describes a concept of how to best accomplish software support tasks using digital simulation facilities that are operated locally or remotely; regionalized or centralized; or in a network configuration. Karpinski discusses a number of near-term issues in the design of a Digital Avionics Support Facility for the AN/ARN-101, Digital Avionics System for the F-4. He estimates that initial cost of a simulator would be between \$1-2 million. He concludes that simulation will be suitable for certification, that high order languages and existing large computer facilities should be used and that automated machine code patching should be developed.

Support of A-7E software at the Naval Weapons Center is next described by Freedman. It describes the NWC's organization and orientation and how various tasks are tackled. Finally, Patterson outlines an Air Force approach toward organic support for the F-111 avionics system at the Sacramento Air Logistic Center. The center is still being developed and should be complete in about two years. Once the dynamic simulation is complete, approximately 87% of the OFP performance will be able to be checked out in the lab as opposed to 60% at the present time. This will reduce flight test time by approximately 50%. The dynamic simulation will allow for OFP problem identification, solution development/checkout and final verification.

Testing, Verification and Validation

The first paper in this group by Wattenberg describes the concepts involved in independent test and evaluation (IT&E) in addition to some of the techniques and tools used, and the cost and future of IT&E. He confines verification and validation to the systematic evaluation of a computer program by an agency independent of the developing group. He claims that such a technique produces a more orderly development process with few latent errors. Such

IT&E or V&V should be utilized whenever dealing with a critical program. To perform IT&E, appropriate tools must be chosen. Perhaps most useful is simulation. How much IT&E is used depends upon criticality of a program. It was estimated that 35-70% of program development costs were IT&E with 45% being typical. Program development costs were defined as costs between system design review and formal qualification review. Evans in his paper promotes independent V&V as a method to increase operational effectiveness. Evaluation takes place in four distinct phases: requirements audit, system verification (including specification and code analysis and model development), operational validation and performance analysis. Roderick adds little to Evans paper except to include that V&V should be performed in parallel with the development cycle and that automatic test tools and simulation are needed to perform it properly. He notes that historically 15-30% of the development effort resources have been devoted to V&V efforts.

The next paper by Ziemer continues on the V&V theme by describing both current methods and problems that must be overcome in the future if aeronautical systems software is written in higher order languages. He concludes that the use of HOL's is feasible only if the verification process can insure verification and validation of compilers. The last paper in this group by Baum and DiStefano discusses a problem frequently encountered in flight testing, that is, the occurrence of anomalies; or one-time, non-repeatable, random occurrences that cannot "be duplicated in laboratory environmental factors." The author describes one approach to solving this difficult problem involving "a form of dynamic, real-time, inflight instrumentation." Such an instrumentation module in the computer occupied only 500 words and took only 2% of the available cycle time. 150 systems parameters were recorded. The empirical data showed that over 75% of the anomalies were not of software origin.

Automatic Test Equipment (ATE)

The recent explosion in the use of automatic test equipment (ATE) in aeronautical systems has prompted a flurry of management activity in this area due to a seeming proliferation of types of ATE and ATE programming languages. The five papers in this section provide the reader with an excellent insight into the types of problems that have been encountered and many of the corrective actions being taken.

The first paper by Fischer describes various types of ATE software, reviews the past 15 year history of ATE application program

cost performance, and makes recommendation for the future. Fischer states that ATE programming costs have decreased as much as 100:1.

Ellis describes one company's approach toward producing ATE test programs automatically which produces documentation automatically and which eliminates the coding phase and its associated errors and costs.

Fischer makes several recommendations to improve the process of transferring engineering responsibility from the developer to the maintainer with respect to ATE test tapes and other types of software, based on the F-111 experience.

The special problem of automating analog ATE test program development is treated by Houston. He describes a Computer Aided Programming (CAP) package which will help automate ATE program generation.

The final paper by Kurkjian also deals with a specific problem, this one being the advocacy of a special HOL for ATE called a TOL or "test-oriented language," which will "increase the population of qualified test programmers, enhance test program quality and maintainability and improve test program documentation.

Programming Languages: Pros and Cons

As a counter to HOL advocacy, the paper by Griffith trades off on-board memory costs against the savings in programming costs using HOLs and presents an analysis that "argues that HOL use is not cost-effective for avionics at the present time, and may not be in the foreseeable future."

While not addressing whether or not to use an HOL in the first place, the next paper by Liples, Allen and Trainor presents the rationale for selecting a higher order language for the Digital Avionics Information System (DAIS) program.

Corn states that the rapid decrease in cost and increase in capabilities of hardware, coupled with the rapid increase in the complexity of tasks demanded by avionics software, requires that HOLs be used for current systems. A history of various HOL uses in systems is given. Some rather thought-provoking recommendations are included such as "we also feel that another look should be taken at FORTRAN for flight software." Finally, Corn noted that a lack of communication of work on software problems has led to a serious duplication of effort.

Radkowski, Minnick and Blondin explore the pros and cons of assembly language versus an HOL for electronic warfare systems, and describe such an HOL.

Finally, DiNitto describes the approach being taken at the Rome Air Development Center (RADC) to systematize the development and control of HOLs for all Air Force applications.

Research and Development

DeRoze, B. C., "Software Specification Methodology, An Analytical Approach."

The first paper by DeRoze describes a new "global approach" toward software development which permits "anticipatory rather than reactionary decision" and will therefore provide a "significant reduction in overall system risk and investment cost, primarily because more emphasis is placed on planning the design."

Hardware productivity has increased fifty-fold in the last 15 years, while software productivity has at best doubled. This is due to the usage of higher order languages, so that 85% of software costs is due to non-coding activities such as specification, design, validation and documentation.

A software development facility which will support these non-coding activities is being developed, which includes: a set of tools providing management visibility and control, a System Data Definition Tool to facilitate the use of different higher order languages within one system, a Program Flow Analyzer for test runs, an automatic documentation tool, a modeling tool for design description and validation and a programming language both for design and coding.

Small, Maj. Albert W. and Engellard, James D., "Operational Software Concept."

The Operational Software Concept (OSC) is the "first major R&D effort to address the total avionics software problem in an integrated and farsighted manner." Its principal objective is to create an integrated avionics software development methodology, including a modular software framework, procedures for design and documentation and development and test, coupled with a support software approach. Portions of the OSC results are already being used by the F-111 and the Digital Avionics Information System.

Allen, Byran M., "Simulation Software to Support OFP Development Maintenance."

Past experience shows that more than 90% of the software errors in a new flight program are found by the use of a real-time simulator at a great cost savings over what a comparable flight test program would cost. The software groups of the Digital Avionics Information System (DAIS) is developing a simulation system to provide the DAIS system with all signals normally present in the aircraft and environment.

3.7 Project PACER FLASH, Volume 1, Executive Summary and Final Report

(This report was the responsibility of the Air Force Logistics Command, Wright Patterson Air Force Base, Dayton, Ohio, 28 September 1973.)

This study was established "for the purpose of assessing alternative methods of providing support for weapon systems software and recommending the most cost/mission effective method for implementation." The task group was to review existing policy and procedures for software support, develop changes to these procedures or recommend new ones, recommend an Air Force position on software support management and publish new or changed policy and procedures in appropriate directives. The task group also considered alternative organizational locations for software support. Software support discussed includes Automatic Test Equipment (ATE), Operational Flight Program (OFF) and Air Crew Training Simulators (SIM).

The major conclusions and recommendations were:

1. Hard cost data on software support is not available. The task force was unable to obtain such data and was forced to evolve a method of providing relative costs for assessing alternatives.
2. The annual Air Force expenditure for software is estimated to be between \$1 billion and \$1.5 billion or about three times the amount spent on hardware. These are primarily people costs.
3. The software-hardware cost ratio was approximately 1:4 in 1955, while the 1985 estimate is 9:1. However, testimony before Congress by the National Bureau of Standards revealed that only 9% of federally funded computing R&D projects were devoted to software problems, although this trend is changing.
4. The Air Force should increase its organic capability for software support, as this will provide substantial cost savings.
5. Checkout and testing account for 45-50% of software costs. "An organic capability for dynamic simulation and verification/validation for airborne weapon systems software is required."
6. Post-acquisition costs of software are extensive. Adequate documentation for software support is not provided by the contractor. All documentation required to support the software must be acquired during the acquisition phase. New MIL-STDs and DIDs are required.

7. The Air Force should consider computer hardware and software as an integral problem. Software must be accorded the same degree of management control presently accorded hardware. Management systems and configuration management must be evolved to support this concept.

8. Air Force directives (MIL-STDs and DIDs) require revision, expansion or new issue to adequately cover software support problems. The Air Force Specialty Code (AFSC) requires adjustment to provide for needed software skills.

9. Support of weapon system software should be managed by the AFLC with participation by AFSC. Relative costs considerations indicate AFLC as the most economical organizational location for software support.

10. "The weapon system manager (SM) should have the ultimate authority and responsibility for software support for those aircraft which are his specific responsibility."

A proposed USAF regulation for defining an Air Force Software Management System was included applicable to all airborne weapon systems and related equipment.

3.8 Tactical Computer Software Acquisition and Maintenance

(This study was sponsored by the Deputy Assistant Secretary of Defense, Production Engineering and Material Acquisition, Washington, D. C., 31 October 1973.)

Condensed Findings of the Study

- Senior DoD personnel have not been aware of the extent of the computer/software role (and costs) in tactical systems.
- Congress advocates efficient management of ADP resources while OMB and OSD management policies do not cover those resources associated with weapons systems.
- In the absence of OSD policy:
 1. Certain air control and defense systems in use in S. E. Asia were found to be unable to exchange commonly useful data automatically.
 2. System program/project offices gave little attention to the software development process, resulting in a lack of separate progress and cost visibility for software.
 3. Acquisition of a wide variety of language processors and support software not only cost millions of dollars, but also made tactical systems engineers dependent on programmers "because of language orientations."
 4. Failure to recognize the software management (maintenance) function early enough yielded late and inadequate contractor documentation, ineffective configuration management, lack of standards, multi-million dollar integration facilities, and lack of adequate government participation in development programs.
- Software costs for major tactical systems range from \$25 million to over \$50 million.
- The comptroller manages ADP resources while DDR&E, ASD (I&L), and ASD(T) are concerned with acquisition, use, and maintenance of tactical digital computers and software.
- Four different documents defined software documentation standards for the services within DoD, some calling for "excessive detail."
- Configuration management directives are hardware-oriented, failing to address the software development process.

- The DoD quality assurance standard (MIL-Q-9858) should be updated to include software development.
- Separation of certain software from hardware acquisition would permit wider use of executive and compiler software from government libraries.
- Military departments are moving toward in-house (non-contractor) management of tactical software for life cycle system support for improved responsiveness and assumed lower cost.

Costs

- Software costs for a major tactical system run from \$25 million to over \$50 million.
- System program/project officers believe they're acquiring "integral systems" rather than hardware plus software, resulting in low software cost visibility "with no portion of integration test costs included."
- Executive software for aircraft computers are estimated to cost from \$.2 million to \$.5 million, and \$.7 million when multi-processing is involved. The AEGIS ATEP executive cost \$1.25 to \$1.50 million.
- Compilers cost from \$1 million to \$5 million each.
- For four major systems/projects late availability of compilers generated extra costs of over \$10 million.

Characteristics

- Tactical system software development requires from three to seven years. There is little evidence of familiarity with the process in program/project offices.

Condensed Recommendations

Secretary of Defense should:

1. Educate DoD top management as to effect of digital computers and software on tactical system acquisitions and life cycle support including relative costs involved.
2. Review DoD organizational responsibilities for software acquisition, use, and maintenance.

3. Issue policies covering use of standard computers, standard languages, standard program libraries, and the "separation of software from hardware acquisition when appropriate".

Summary

- Computer equipment "integral to weapons systems" is excluded from the scrutiny of OMB, GSA, OSD, and the military departments have in turn delegated tactical computer and software policy decision making to system program/project managers.
- As a result of inadequate digital data interfaces as demonstrated in S. E. Asian operations, OSD (DDR&E) has defined interoperability requirements for tactical systems (DODD 4630.5 "Compatibility and Commonality of Equipment for Tactical Command and Control and Communications", and ACP-12 which defines interoperability of existing and planned command and control systems).
- Some organizational steps taken to improve management of a tactical system resources: in August 1971 the Navy established a Tactical Digital Systems Office (Material Command) to promote standardization; the Army established a Tactical Data System Office in April 1971 (excluding weapons systems); the Air Force designated a single ADP manager in July 1971.
- Major Tactical Systems employ a variety of computers and languages: five Air Force/Navy aircraft systems use eight types of computers and nine languages; eight Army/Air Force ground systems use five types of computers and three languages.
- Development of new support software (executive, compiler, simulator) concurrently with other elements of a tactical system impacts the system schedule adversely and is "unnecessarily expensive".
- Executive software should be able to function in degraded mode when hardware components are damaged in combat "but no policy is evident."
- Development of a tactical system engineering language would free the engineer from his dependence upon programmers. The tactical procedures embodied in current software are not obvious to engineers.

Program Specifics

SAM-D

Budget:

	<u>Millions</u>
Advanced Development	\$250
Software	12
Engineering Development	\$507
Programming (not including integration)	27
Integration (fire ctl group)	\$68
Programming (to modify advanced development software)	5

TACFIRE

"The TACFIRE project called for simultaneous development of: (1) a tactical language (TACPOL), (2) support software, and (3) operational programs. Its approach involved highly sophisticated support software manipulations. The original plan called for writing in IBM's PL/1 language a compiler and assembler to execute on an IBM 360 computer. These were called the PSSA Compiler and PSSA Assembler, respectively. Next, the plan called for translating the PSSA Compiler and PL/1 into the Army's TACPOL tactical computer language and compiling it using the PSSA Compiler and Assembler to produce a new compiler to execute on the TACFIRE Litton 3050 computer. This was to be the PSSB Compiler. Additionally, a PSSB Assembler was to be written in TACPOL and compiled and assembled with PSSA software for execution on the 3050."

"In the fall of 1970, after two plus years of effort, a review of the PSSB Compiler showed it was too slow, and resulted in a new development effort. The resultant new PSSB Compiler delivered this year, is faster and more efficient than the earlier PSSA. However, most TACFIRE programs have been written in PSSA, and since the A and B languages have diverged, a rewrite and retesting is planned. Additionally, the ARTADS office is considering taking the PSSB off the tactical computer and executing it on a large centrally located host computer to facilitate maintaining and distributing programs from one location. This will require another compiler."

3.9 A Report on Air Force Logistics Command Operation Flight Program Support: Introduction and Summary

(This report was developed by the System Development Corporation, Santa Monica, Ca., 9 December 1974.)

This study is concerned with the maintenance and support of Operational Flight Programs (OFPs) for programmable digital avionics computers for the Air Force. Problems existing within these types of systems are addressed, the range of management options are given, and recommendations for future support are stated. In addition to observations, conclusions, and recommendations of an overall nature, the study discusses and makes recommendations for specific aircraft types and their systems:

- AC-130A and AC-130H
- B-52/SRAM
- C-5A
- C141A and DC-130 (AWADS)
- EC-135J and HH-53C
- F-4E and RF-4C
- RC-135 and DC-130E
- Undergraduate Navigator Training System (UNTS)

Four alternative OFP support concepts (implicit assumption is made that software support will be provided organically to USAF for the majority of OFPS) are given in the Report. It became apparent that there were common factors to be included regardless of which methodology (alternative) is used for supporting OFP for the weapon systems studied.

Common Requirements

"In order to implement positive management control over OFP support, and to perform this support organically, the following capabilities and services are minimally required:

- Complete support software for each OFP is mandatory for OFP maintenance.
- Baseline documentation on the support software and the OFP itself must be procured and procedures developed for documentation maintenance.
- Training, by the prime contractor of technical support personnel, will be required for those OFPs for which organic technical expertise does not currently exist.

- For the complex avionics systems, integration mock-ups and simulators and emulators will be required for analysis and testing."

Mandatory Controls of OFP Content and Documentation

- AFM 800-XX implementation.
- Using T.O. system to control software specifications is outdated and should be phased out.
- OFPs should be procured with MIL-STD 483 specifications to the greatest extent possible.
- AFLC should refuse any operational programs if the support software and documentation are not supplied before the OFP has been transitioned.

The four proposed alternatives are as follows:

1. Total Software Capabilities (recommended in Project PACER FLASH) - consolidates all resources required for OFP support at a single Software Support Center (SSC), located at an ALC.
2. Functional subsystem centralization - support responsibility for all subsystems performing similar functions, e.g., navigation, is centralized in one Functional Area Support Center (FASC).
3. Weapon System Centralization - OFP support for any given weapons system is the responsibility of, and directly controlled by the System Manager (SM), for that weapon system.
4. Combined Functional and Weapon System Control - combines the best features of 2. and 3. This is quite close, in terms of organizational structure, to the present mode of operation.

Facilities, personnel, and costs to support each OFP are described. Square foot areas to house computing, simulation and mock-up equipment are used as "facilities" and no exact size is available. Personnel requirements are difficult to highlight because of the problem of differentiating between technical managers and technical people doing the work - again no exact number available. Cost is not exact due to three estimates used, viz., government, SDC analysis, and prime contractors. A significant amount of data on the characteristics of the computers, the software, the OFP software maintenance methods/tools/capabilities

used today, and the estimated yearly costs for software maintenance in the future were collected in this study.

Each organizational alternative was analyzed in depth to identify the various advantages and disadvantages. A method of "Figures of Merit" was used to weight each of the alternatives. The factors used are listed in Table 3-10, (Table 3.3-1 from the Report). The report presents the following analysis for choosing alternatives.

"Functional organization should be chosen for support in those cases where a given avionic system and its OFP is utilized in several different aircraft systems. On the other hand, where the OFP is unique to a given weapon system, and where the skills for this support require detailed knowledge of that system, then the support organizations should report to the SM by the shortest management path feasible."

Several comments made in the study that apply to all OFPs considered are as follows:

- OFPs studied were generally written in assembly language.
- No "standard" programming language is feasible for the OFPs studied.
- Advantages of standardized languages is acknowledged.
- Feasibility of rewriting a program from one system to another is expensive (20-40% of development cost) course.
- Transfer of developed algorithms has obvious merit.

A detailed analysis is given for each system considered in this study. These explain the reasoning for changes in the current transition policies, if any, and the associated costs.

The report conclusions are:

- Configuration Management practices and procedures are not performed in a standard manner.
- In order to adequately maintain documentation, it must be initially delivered by the contractor in charge of OFP software and procedures for update followed.

TABLE 3-10
FIGURES OF MERIT

1. Initial Capital Expenditure Required
2. On-Going Costs - General OFP Maint.
3. On-Going Costs - Special Action
4. Response Time - Standard Ops.
5. Response Time - Special Action
6. Responsiveness to SM
7. Responsiveness to User
8. Management Communications Effectivity
9. Technical Communications Effectivity
10. Technological Growth Potential
11. Product Visibility
12. Skill Availability/Utilization
13. Facility Resource Utilization
14. Support Continuity
15. Staffing - Personnel Continuity
16. Staffing - Buildup

NOTE: This table is a copy from A Report on Air Force Logistics Command Operation Flight Program Support by SDC.

- In most cases, support software was not provided to AFLC for the OFPs at transition. The impact on organic support of OFP software is obviously detrimental.
- In no case was a condition found that standardization of programming languages, or the rewrite of a program from one system to another, was warranted.
- Validation, verification, and certification was found to present significant difficulties in discussions of OFPs. This is due mainly to a wide diversity in defining V, V&C.

The personnel with AFLC responsible for support of avionic computers have a difficult task to perform because:

- Organizationally, ALCs do not lend themselves to resolution of software problem solving.
- Few established software configuration management procedures and those adopted by the ALCs work because the people who administer them make them work.
- Insufficient support hardware and software create one of the most difficult road blocks in software maintenance.
- Historically, procurement practices have failed to recognize the criticality of providing AFLC with proper OFP support tools required for proper maintenance of the computer software.

The report's recommendations are:

- "Baseline software configurations should be established and proven techniques for the management of this configuration must be adhered to."
- All future acquisitions should include provision for delivery of support software (e.g., assemblers, compilers, etc.).
- "The Technical Order system with respect to software control should be phased out and replaced with the proven specification system, defined in MIL-STD 800-14 and amplified in AFM 800-XX."
- The combined Weapon System Manager and Functional Area organizational structure should be further defined and implemented to manage all OFP support activities.

3.10 Report of Army Scientific Advisory Panel Ad Hoc Committee
for Army Tactical Data System Software Development
(This report was developed by an Ad Hoc Group of the Army
Scientific Advisory Panel.)

The charge to the Group was originally expressed as follows:

Statement of Work

"To determine exploratory development efforts which offer the best promise of satisfying the Army's requirement to reduce cost and schedule, and increase performance and reliability of software for major Army tactical data systems."

"It became clear early in the discussion that the solution of tactical data system software problems should not be approached solely through software or programming research and development, although certain efforts in that area will prove fruitful. Rather, the software problem often stems from an inadequate initial effort on design of the tactical data system: selection of hardware before the problem is understood; provision of a system structure that fails to match the problem to be solved; instinctive and arbitrary choice of conventional central uniprocessors when multiprocessors, a federated system, associative processors, or special purpose processors might be a superior choice. When an inferior hardware system approach is taken, unduly extensive and complex software may be required. Often development of the software itself is inadequately managed: the kind of discipline normally found in hardware development is missing in software development; the problem is not clearly and fully defined before the programming is cranked up; the work is not properly broken down, assigned, and monitored; interfaces between software packages and program segments are not formally established; programming is accomplished haphazardly, rather than being structured."

These and other observations led to an attack on the problem as follows:

"Determine the factors that lead to extensive and complex software and to problems in developing software for tactical data systems, and recommend practices and useful exploratory efforts to mitigate these difficulties."

It should be noted that the above are direct quotes from the Ad Hoc Committee Report. The findings and recommendations are also quoted verbatim from the report as follows.

FINDINGS AND RECOMMENDATIONS

"The ad hoc study group's investigations led to findings and recommendations in four major areas:

1. System Design and System Hardware.
2. Software Design and Development.
3. R&D Related to Software.
4. Army Management of Software Development

Findings and Recommendations in these areas are summarized in this Section of the report."

A. System Design and System Hardware

"The study group feels that many software problems are rooted in an inadequate initial system design effort. It recommends an orderly system design, considering all reasonable alternative system architectures before a development is initiated. Specifically, it is recommended that:

- a. Alternative system architectures to that based on a large general purpose computer be evaluated in detail.
- b. The system design emphasize the processes to be performed in the tactical environment before defining the processors to be employed.
- c. Existing computer systems with standard system software be evaluated before considering the development of new hardware which will require new system software.
- d. Hardware and software be defined and developed interactively starting with a definition of the language to be used and the operating system parameters required.
- e. Hardware capacity be specified with adequate allowance for a safety factor to reduce the difficulties of programming."

B. Software Design and Development

"Problems in software design usually result from attempts to obtain very efficient programs to be run on minimum size hardware. Often the program writing must be accomplished without adequate system software tools. In Section V the study group deals with this problem and recommends:

a. Early design (or selection) of system software programs and software testing tools.

b. Standardization, specification, documentation and use of a higher-level programming language for Army Tactical Data Systems.

c. Selection and documentation of program libraries of standard tactical operating systems, and of operational tactical program segments with proper consideration of applicable commercial software.

d. Application of the principles of structured programming in tactical data system design.

e. Use of an outside system advisor to assist in program development."

C. R & D Related to Software

"A number of subjects recommended for research and exploratory development have been identified (Section VI). The research and exploratory development for Army Tactical Data Systems must be conducted and coordinated in a manner to offer maximum responsiveness to PM ARTADS. Specific studies recommended are:

a. Continued development and evaluation of the standard programming language for Army Tactical Data Systems taking advantage of commercial developments.

b. Development of standard operating systems for tactical applications.

c. Development of computer architectures optimized to the tactical problem, to the standard language, and to means for optimum program development.

d. Improved methods for specifying, selecting, developing, testing and evaluating tactical hardware and software."

D. Army Management of Software Development

"The study group supports the efforts of PM ARTADS to develop and apply improved management techniques in tactical software development. In Section VII the following recommendations on management of software development are discussed:

- a. Meaningful and realistic tools for management and documentation of software developments.
- b. Early agreement on programming language and operating system requirements.
- c. Agreement by all parties on software tasks and software specifications.
- d. Evaluation of software development progress related to satisfying operational requirements.
- e. Specifications and tools for software testing.
- f. The requirement for accepting evolutionary development of software in tactical systems."

3.11 An Analysis of Computer Software Management Requirements for Operationally Deployable Systems, Executive Summary (Volume I), (This analysis was developed by Bernard A. Zempolich, Department of the Navy.)

The analysis was conducted within the Research Fellows Program at the Industrial College of the Armed Forces. This program allows individuals to perform independent research into topics related to defense management and national security.

The analysis noted that over the last decade, the DoD acquired and deployed a large number of operationally deployable systems, each of which has at least one general-purpose programmable digital computer to perform computations necessary to carry out the operational mission. This explosion of applications of digital computers has resulted in a large inventory of myriad types of computer systems which need to be supported. The DoD not only has the problem of monitoring computer hardware, but must provide the means to maintain both the operational and support software. The author divided the software problem into three areas, quoted as follows:

"a. What actions are required to provide the "men, material, and money" needed to overcome the present resource shortcomings?

b. What management policy, planning, and guidelines are necessary to provide interim solutions for limiting further proliferation of computer hardware and ensure that system life-cycle software management requirements are taken into consideration?

c. What efforts should be undertaken to ensure that the proper permanent (long-term) directives, instructions, and guidelines are established and all necessary resources are identified?"

The purpose of the research is quoted from the study as follows: "(1) identify those factors which contributed to the current problems facing DoD relative to operational software development, maintenance, and costs; (2) examine present policy planning, and organizational structures which impact on the management of operational software and the associated support facilities; (3) identify and analyze present techniques used to develop, maintain, project costs, and document operational software; and (4) present the findings, conclusions, and recommendations of the research."

The author examined the management of software which is part of an operationally deployable system or related software support activities. The following applications were thus included:

- Weapon Systems
- Tactical Data Systems
- Trainers and Simulators, and certain types of Test Range and Test Vehicles
- Automatic Test Equipment
- Software Support Activities (Centers) which are required to provide sustaining capability relative to the generation of software programs for the aforementioned systems.

The study provides a digest of Finding and Conclusions and Recommendations which are quoted as follows:

Findings

• "The uncontrolled introduction of myriad types of general-purpose, programmable digital computers into the DoD inventory has engendered a need for substantial increases in manpower, funding, and material resources to support these systems. In many instances, the necessary resources to support these systems have not been provided for by project offices and/or the cognizant agency.

• There is a need for DoD-wide standards, policy, guidelines, and direction relative to computer software for operationally deployable systems.

- DoD does not have effective means for determining the accuracy and reasonableness of software pricing data submitted by contractors. (Note: In most cases neither do the contractors.)

- Additional training is needed for technical, management, procurement, and logistics personnel if effective management of operational software is to be established within DoD."

Conclusions and Recommendations¹

- "OSD and/or the individual Services should review the adequacy of present approaches for such software management considerations as: visibility during development, configuration control, funding, test and evaluation, support requirements after deployment, and future policy so as to prevent a reoccurrence of the situation which exists in regard to software for operationally deployable systems.

- A study should be undertaken by OSD to determine the manner by which the present shortcomings in the procurement of software can be overcome.

- Formal training for DoD personnel in software management, procurement, development, and subsequent support is considered to be urgently needed. It is recommended that the DoD Computer Institute be given the responsibility and resources to establish appropriate, formal courses.

- OSD should re-examine whether the division of ADP resources into "weapon system" and "non-weapon system" categories is still a viable and logical demarcation of organization responsibilities.

- OSD should consider the establishment of a technical office/agency which would concern itself solely with Service-wide factors such as "commonality" and/or standardization, development of software "tools", maintainability of digital equipment, and computer program documentation techniques.

- A DoD compilation of "militarized" hardware/software should take place following which all Services should be required to examine these items for use before initiating new developments.

- A separate research and development program for computer software for operationally deployable systems should be undertaken by the Department of Defense."

¹These are quoted from the author's digest. Much more detail is provided in the body of the report.

4. SOFTWARE ACQUISITION AND MANAGEMENT BIBLIOGRAPHY

This section contains a bibliography of publications which is presented for use as reference material. The bibliography is presented in 2 sections. Section 4.1 includes the 11 baseline studies and software workshop proceedings used during the MITRE study, with an additional listing of other publications where MITRE personnel has an awareness. Section 4.2 includes listings of additional publications which were recommended through the OSD Software Steering Committee. In some cases, the publication citations in Section 4.2 are incomplete. They are in the format provided to MITRE, and the study team was unable within the scope of the effort to conduct research to obtain complete citation information. In a few cases, Section 4.2 contains citations which also appear in Section 4.1.

4.1 Baseline Studies and MITRE Developed Bibliography

1. Agnew, Carson, E., et. al. ARPANET Management Study: New Application Areas. Palo Alto, CA: Cable Data Associates, Inc., 5 August 1974. (AD 787 039)
2. Aircraft Avionics (Digital Avionics Study). Volume I: Final Report. Wright Patterson Air Force Base, Dayton, Ohio: Air Force Systems Command, Aeronautical Systems Division, Avionics Engineering Division, April 1973. (ASD-TR-73-18)
3. Annual Reliability and Maintainability Symposium. (Proceedings). Los Angeles, CA, 29-31 January 1974. New York: The Institute of Electrical and Electronics Engineers, 1974.
4. Aron, Joel D. The Program Development Process. Part I. The Individual Programmer. The Systems Programming Series. Reading, MA: Addison-Wesley Publishing Company, 1974.
5. August, Elsa F., Ms, Project Principal. A Survey of Avionics Simulation Facilities, FEDSIN Report, MV-409-012-TAC/AFDAA, Federal Computer Performance Evaluation and Simulation Center, August 1974.
6. Bell, D. E., Burke, E. L., A Software Validation Technique for Certification: The Methodology (MTR-2932, Volume I). The MITRE Corporation, Bedford, Massachusetts, Contract Sponsor: ESD, 12 November 1974.
7. Brandon, Dick H., Data Processing Organization and Manpower Planning. New York: Petrocelli Books, 1974.
8. Burr, W., et. al. TACFIRE History and "Lessons Learned" Executive Summary. Meta Systems Corporation, n.d., Project Sponsor: Army Computer Systems Command, Fort Belvoir, Va.

9. Buxton, J. N. and B. Randall, eds. Software Engineering Techniques. (Report of a Conference). Sponsored by: The NATO Science Committee, Rome, Italy, 27-31 October 1969. Published: Burmingham, England: Kynoch Press, April 1970.

10. CCIP-85. Information Processing/Data Automation Implications of Air Force Command and Control Requirements in the 1980's: Executive Summary, Revised Edition. Los Angeles, California; Air Force Systems Command, Space and Missile Systems Organization, February 1972. (AD 742 292)

CCIP-85. Information Processing/Data Automation Implications of Air Force Command and Control Requirements in the 1980's: Eleven Volumes. Los Angeles, California; Air Force Systems Command, Space and Missile Systems Organization.

Volume I	<u>Highlights</u>	(AD 900 031L)
Volume II	<u>Command and Control Requirements: Overview</u>	(AD 521 887L)
Volume III	<u>Command and Control Requirements: Intelligence</u>	(AD 523 881L)
Volume IV	<u>Technology Trends: Software</u>	(AD 919 367L)
Volume V	<u>Technology Trends: Hardware</u>	(AD 907 626)
Volume VI	<u>Technology Trends: Sensors</u>	(AD 525 661)
Volume VII	<u>Technology Trends: Integrated Design</u>	(AD 906 757L)
Volume VIII	<u>Interservice Coordination Trends</u>	(AD 522 216L)
Volume IX	<u>Analysis</u>	(AD 524 549)
Volume X	<u>Current Research and Development</u>	(AD 905 654L)
Volume XI	<u>Integrated Research and Development Roadmaps</u>	(AD 902 515L)

11. Cetron, M., et. al., eds. Quantitative Decision Aiding Techniques for Research and Development Management. New York: Gordon and Breach, 1972.

12. Cheng, L., Sullivan, J. E., Case Studies in Software Design, (MTR-2874, Volume I). The MITRE Corporation, Bedford, Massachusetts, Contract Sponsor: ESD, 30 January 1974.

13. Computer Software Reliability. (Record of a Symposium). New York City, 30 April - 3 May 1973. New York: Institute of Electrical and Electronics Engineers, 1973.

14. Corrigan, Ann E., Results of an Experiment in the Application of Software Quality Principles (MTR-2874, Volume III). The MITRE Corporation, Bedford, Massachusetts, Contract Sponsor: ESD, 30 June 1974.

15. Couger, J. Daniel and Robert W. Knapp, eds. System Analysis Techniques. New York: John Wiley and Sons, 1974.
16. Definition of an Approach to Establishment of a F-15 Avionics Software Support Capability, "F-15 Avionics Software Support". Avionics Engineering Directorate, ASD, AFSC, Wright-Patterson AFB, Ohio, July 1974. (Distribution is limited to U.S. Government Agencies only.)
17. Development in Information Processing: A Look at the Future. Systems and Procedures Report #24. New York: Life Office Management Association, August 1974.
18. Electronics-X: A Study of Military Electronics with Particular Reference to Cost and Reliability. Volume 2: Complete Report Arlington, Virginia; Institute for Defense Analyses, Science and Technology Division; January 1974. (R-195) (Volume 1 was not available for referencing in this bibliography.)
19. Evans, D. J., ed. Software 70. Proceedings of Software World Conference, University of Sheffield, England, April 1970. New York: Auerbach Publishers, 1970.
20. Evans, D. J., ed. Software 71. Proceedings of Software World Conference, University of Kent at Canterbury, England, July 1971. New York: Maxwell Scientific International, Inc., 1971.
21. Fisher, David A., Automatic Data Processing in the Defense Department. Arlington, Virginia; Institute for Defense Analyses, Science and Technology Division; October 1974. (P-1046) (Contract DAHC15-C-0200, Task T-36)
22. Forecast 1968-2000 of Computer Developments and Applications. 1602 Copenhagen V, Denmark: Parsons and Williams, 1968.
23. Fox, J. Ronald, Arming America, How the U. S. Buys Weapons. Cambridge, MA: Harvard University Press, 1974.
24. Gilb, Tom, Reliable EDP Application Design. New York City: Petrocelli Books, 1974.
25. Goldberg, Jack, ed. The High Cost of Software, (Proceedings of a Symposium). Sponsored by: Air Force Office of Scientific Research, Army Research Office, and Office of Naval Research, Monterey, California; 17-19 September 1973. Published: Menlo Park, California; Stanford Research Institute, n.d. (Contract N00014-74-C-0028)
26. Government/Industry Software Sizing and Costing Workshop, (Summary Notes). L. G. Hanscom Field, Bedford, Massachusetts; Air Force Systems Command, Electronic Systems Division, 1-2 October 1974. (Draft)

27. Gossick, Lee V., Major General, USAF. Management of System Acquisition Programs in the Air Force. Article published in the Defense Industry Bulletin, Summer 1971.
28. Hetzel, William C., ed. Program Test Methods. Prentice-Hall Series in Automatic Computation. Englewood Cliffs, NJ: Prentice-Hall, 1973.
29. Hice, G. F., et. al. System Development Methodology. New York: American Elsevier Publishing Company, 1974.
30. Kelliher, D. W., Software Quality Assurance and Production Control Practices in the Acquisition of Large Systems. The MITRE Corporation, McLean, Va., June 1975. (MTR-6906).
31. Kirk, Frank G. Total System Development for Information Systems. New York: John Wiley and Sons, 1973.
32. Kosy, D. W. Air Force Command and Control Information Processing in the 1980's: Trends in Software Technology. Santa Monica, CA: The RAND Corporation, June 1974. (AD 525 661L) (Version of CCIP-85)
33. Lieblein, Edward, AMC Center for Tectical Computer Sciences (CENTACS). (Briefing). Presented at ARMCOM, 6 November 1974.
34. Lieblein, Edward, Problems in Software Development. Keynote Address Presented at the Joint Services Electronics Program Topical Review in Information Sciences, University of Illinois, 16 October 1973. Fort Monmouth, NJ: U. S. Army Electronics Command, Communications/Automatic Data Processing Laboratory, n.d.
35. Liebowitz, B. H., et. al. Procedures for Management Control of Computer Programming in Apollo. Washington, DC: Bellcomm, Inc., 28 September 1966. (TR-66-320-2) (Contract NASw-417)
36. Liebowitz, B. H., et. al., Procedures for Management Control of Computer Programming in Apollo, Bellcomm, Inc., September 28, 1966, reissued June 15, 1967.
37. MacGowan, Roger A., and Reid Henderson, eds. CDP Review Manual: A Data Processing Handbook. Third Edition. New York: Petrocelli Books, 1973.
38. Management Plan for the AFSC Information Systems Technology Applications Office. L. G. Hanscom Field, Bedford MA: Air Force Systems Command, Electronic Systems Division, Deputy for Command and Management Systems, 1 December 1973.

39. Manley, John H., Colonel, and Archibald, W. Robert, eds. Aeronautical Systems Software Workshop, (Proceedings). Sponsored by: Headquarters, Air Force Systems Command, Aeronautical Systems Division, Wright-Patterson Air Force Base, Dayton, Ohio, 2-4 April 1974. (Draft)
40. Mathis, N. S., et. al., Software Milestone Measurement Study. San Diego, CA: Naval Electronics Laboratory, November 1973. (AD 775 305)
41. Metzger, P. W., Managing a Programming Project. New Jersey: Prentice-Hall, 1973.
42. Morin, Lois H., Estimation of Resources for Computer Programming Projects. Masters in Computer Science Thesis, University of North Carolina at Chapel Hill, 1974.
43. Nauer, Peter and Brian Randall, eds., Software Engineering. (Report of a Conference). Sponsored by: The NATO Science Committee, Garmisch, Germany, 7-11 October 1968. Published: January 1969.
44. NEREM 74 Record. Part 1: Technical Papers. Boston, MA, 28-31 October 1974. Boston Section of the Institute of Electrical and Electronics Engineers, 1974.
45. Project PACER FLASH. Four Volumes. Wright-Patterson Air Force Base, Dayton, Ohio; Air Force Logistics Command, 28 September 1973.
- | | |
|------------|---|
| Volume I | <u>Executive Summary and Final Report</u> |
| Volume II | <u>Appendix A: Automatic Test Equipment (ATE)</u> |
| Volume III | <u>Appendix B: Operational Flight Program</u> |
| Volume IV | <u>Appendix C: Air Crew Trainers (Simulators)</u> |
46. Reich, Eli T. Tactical Computer Software Acquisition and Maintenance Staff Study. Washington, D. C.; Deputy Assistant Secretary of Defense, Production Engineering and Materiel Acquisition, 31 October 1973.
47. A Report on Air Force Logistics Command Operation Flight Program Support. Two Volumes. Santa Monica, California, System Development Corporation, 9 December 1974. (TM-5439/000/00)
- | | |
|-----------|---|
| Volume I | <u>Introduction and Summary (TM-5439/000/00)</u> |
| Volume II | <u>Data Analysis and Conclusions (TM-5439/001/00)</u> |
48. Report of Army Scientific Advisory Panel Ad Hoc Group on Army Tactical Data System Software Development. October 1974.

49. Ridge, W. J., and L. E. Johnson, Effective Management of Computer Software. Illinois: Dow Jones-Irwin, Inc., 1973.
50. Rubin, Martin L., Introduction to the System Life Cycle. Handbook of Data Processing Management. Volume 1. Princeton, NJ: Brandon/Systems Press, 1970.
51. Ruckert, William C. Fiscal and Life Cycles of the Defense Systems, 1st ed., Pomona, CA: General Dynamics, September 1973.
52. Ruckert, William C. Fiscal and Life Cycles of the Defense Systems, supp. to 1st ed., Pomona, CA: General Dynamics, August 1974.
53. Sackman, H., Computers, System Science and Evolving Society: The Challenge of Man-Machine Digital Systems. New York: John Wiley and Sons, 1967.
54. Searle, Lloyd V., and Robert L. Henderson, System Engineering Guide for Computer Programs. Santa Monica, CA: System Development Corporation, March 1968. (AD 666 430)
55. Software Acquisition Handbook. Forth Monmouth, NJ: U. S. Army Tactical Data Systems (ARTADS), 15 June 1973. (Draft)
56. Summary of the Report of the Commission on Government Procurement. Washington, DC: U. S. Government Printing Office, December 1972.
57. SADPR-85 (Support of Air Force Automatic Data Processing Requirements Through the 1980's). Appendix to Technology and Cost Forecasts for Electronic Data Processing Hardware, Software and Data Communication Networks. Cambridge, MA: Arthur D. Little, Inc., 15 February 1974. (Contract F19-628-74-C-0093)
58. SADPR-85 (Support of Air Force Automatic Data Processing Requirements Through the 1980's). Volume 5: Economic Analysis. L. G. Hanscom Field, Bedford, MA: Air Force Systems Command, Electronic Systems Division, June 1974.
59. SADPR-85 (Support of Air Force Automatic Data Processing Requirements Through the 1980's). Volume 6: Programming Planning. L. G. Hanscom Field, Bedford, MA: Air Force Systems Command, Electronic Systems Division, June 1974.
60. Tactical Digital Systems Documentation Standards. (Technical Note). Washington, DC: Office of the Secretary of the Navy, Department of the Navy, Research and Development, 8 August 1974. (SECNAV Instruction 3560.1, TN-01)

61. TRW Systems Engineering and Integration Division. Proceedings of the TRW Symposium on Reliable, Cost-effective, Secure Software. TRW-SS-74-14, 20 and 21 March 1974.
62. Turn, Rein, Computers in the 1980s -- Trends in Hardware Technology. (The Rand Paper Series), the Rand Corporation, Santa Monica, CA, March 1974.
63. USAF Aeronautical Systems Operational Software Maintenance Study. Avionics Engineering Directorate. Wright-Patterson Air Force Base, Dayton, Ohio: Air Force Systems Command, Aeronautical Systems Division, October 1973.
64. U. S. Military R&D Management. Washington, DC: Georgetown University, The Center for Strategic and International Studies, 1973.
65. Vance, P. R., Command Systems Study - Phase I Report. Bedford, MA: The MITRE Corporation, 30 June 1966. (MTR-249) (Contract F19 (628)-5165)
66. Weinwurm, George F., ed. On the Management of Computer Programming. Princeton, NJ: Auerbach Publishers, Inc., 1970.
67. Whitehouse, F. Systems Documentation: Techniques of Persuasion in Large Organizations. London: Business Books, LTD., 1973.
68. Zempolich, Bernard A., An Analysis of Computer Software Management Requirements for Operationally Deployable Systems: Executive Summary (Interim). Department of the Navy: Industrial College of the Armed Forces Research Fellows Program, n.d. (Draft)
69. Zempolich, Bernard A., An Analysis of Computer Software Management Requirements for Operationally Deployable Systems. Volume I: Executive Summary. A Student Committee Report. Department of the Navy: Industrial College of the Armed Forces, Research Fellows Program, November 1972.

4.2 Software Steering Committee Suggested Publications

4.2.1 OASD (Comptroller, Management Systems)

1. Final Report of Ad Hoc Committee on Interoperability of Tactical Data Systems with DoD Management Information Systems, 1 October 1974.
2. OASD (I&L) Study "Tactical Computer Software Acquisition and Maintenance Staff Study" (FOUO), 31 October 1973.

3. GAO Report "Tools and Techniques for Improving the Efficiency of Federal Automatic Data Processing Operations, B-115369", June 3, 1974, OSD Case #3851.
4. GAO Report, "Advantages and Limitations of Computer Simulation in Decision Making, B-163074" May 3, 1973, OSD Case #3568.
5. GAO Report, "Opportunity for Greater Efficiency and Savings Through the Use of Evaluation Techniques in the Federal Government's Computer Operations, B-115367", August 22, 1972.
6. GAO Report, "Acquisition and Use of Software Products for Automatic Data Processing Systems in the Federal Government, B-115369," June 30, 1971, OSD Case #3301.
7. House of Representatives Report No. 92-1389, "Department of Defense Appropriation Bill, 1973", September 11, 1972, pp. 100-102.

4.2.2 IDA, Science and Technology Division

1. The entire issue of ACM Computing Surveys Special Issue on Programming, Vol. 6, No. 4, December 1974.
2. "The Humble Programmer," by Edsger W. Dijkstra in Communications of the ACM, Vol. 15, No. 10, October 1972, pp. 859-866.
3. "The Mytical Man-Month," by Frederick P. Brooks, Jr. in Datamation, December 1974, pp. 44-52.
4. The Psychology of Computer Programming by Gerald M. Weinberg (Von Nostrand Reinhold Company, New York, 1971).

4.2.3 Army

1. AMC PAMPHLET 70-4 Software Acquisition, A Guide for the Materiel Developer.
2. ARTADS Interface Management Plan (AIMP)
3. AR 18-1 (Rev. 4) - Management Information Systems Policy, Objectives, Procedures, and Responsibilities
4. Report of the Army Scientific Advisory Panel Ad Hoc Committee for Army Tactical Data System Software Development

5. Guidelines for the Center for Tactical Computer Sciences
6. AR 70-1 - Army Research and Development
7. Charter for the Integrated Software R&D Working Group
8. Software Management Through Product Control, by Dr. R. Merwin
9. MIL-S-52779 - Software Quality Assurance Program Requirements

4.2.4 Navy

1. SECNAVINSTs: 3560.1; 5420.176
2. OPNAVINST 3500.27B
3. NAVMATINSTs: 5230.5A; 5200.27A; 4130.3A; 4130.4A; 4120.101A
4. TADSTANDS 1 thru 7
5. TADTASKS: 1-73; 2-73; 1-74; 2-74; 4-74
6. Master Combat Direction Systems Plan Draft
7. IDA Report R-195. Electronics-X: A Study of Military Electronics with particular reference to cost and reliability
8. Joint Fleet Combat Direction Systems Support Activity Standardization Manual
9. NAVMAT Document: Requirements for Intercomputer Interface Documentation
10. Weapons Specification 8506 (Rev 1)
11. Navy and Marine Corps Tactical Digital Equipment Catalog
12. Fleet Combat Direction System Support Activity, Dam Neck Document 74-95: Handbook for Program Development and Production Procedures of Digital Processor Program.

4.2.5 Air Force

1. Project PACER FLASH, dated 28 September 1973

2. Adams Committee Report on Software Acquisition (1971 RDPQ report)
3. Fubini Avionics Study
4. Study of Command Control Information Processing in the 1980s (CCIP-85)
5. Aeronautical Systems Software Workshop Proceedings
6. Monterey High Cost of Software Symposium
7. IEEE Software Reliability Workshop, 1973
8. Aircraft Avionics (Digital Avionics Study), Final Report ASD TR-73-18, dated April 1973
9. Support of Air Force Automatic Data Processing Requirements through the 1980s (SADPR-85), dated April 1972
10. System to Automate Logistics at the Base Level (STALOG)
11. Support of Mobility Automatic Data Processing Requirements Study (SOMARS)
12. Project ACE
13. ESD Software Cost Workshop
14. GAO Report No. B-115369, dated 30 June 1971, "Acquisition and Use of Software Products for ADPS in the Federal Government"
15. GAO Report No. B-163074, dated 22 May 1972, "Practice of the Naval Ship Systems Command Related to the Procurement of the AN/UYS-7(V) Computer System"
16. Deputy ASD (Prod. Eng. and Mat. Acq.) Staff Study, dated 31 October 1973, "Tactical Computer Software Acquisition and Maintenance"
17. Department of the Army Staff Report, dated 15 December 1974, "Communications Software Management"
18. U.S. Army Scientific Advisory Panel -- Ad Hoc Group on Army Tactical Data Systems Software Development (Final report currently in preparation)

19. U. S. Navy (NAVAIRSYSCOM) Staff Study, dated 3 March 1972, "Software Management"

20. U. S. Navy (Tactical Software Division, Naval Missile Center) Staff Study, dated 14 December 1974, "Navy Airborne Software Support Activity"

21. NASA Staff Report of the Information Management System Steering Group - Ad Hoc Committee on Software, dated January 1973, "IMS Software Study for Manned Shuttle Payloads"

22. Defense Sciences Board Task Force on Electronics Management report, dated 31 August 1973, "Standardization of Avionics Information Systems"

APPENDIX A

DOD SOFTWARE ACQUISITION STUDY
REQUESTED INFORMATION

(Questions apply to all software life-cycle phases and include both operational and support software categories; i.e., all software required to develop, produce, operate, and maintain the system including training of personnel.)

1. BACKGROUND INFORMATION

- 1.1 Identify the major system and software acquisition phases and decision milestones applicable for this system.
- 1.2 Describe the current status of the system (i.e., where the system is in the acquisition cycle).
- 1.3 Identify major project redirections or delays and reasons if applicable.
- 1.4 List major industry contractors (i.e., prime contractor, software (sub) contractors for both operational and support software). Identify portions done in-house if applicable.
- 1.5 Provide a list of the major DoD and Service regulations and standards documents, and management systems used during the system and software acquisition; provide a list of similar documents developed specifically for this program.

2. TECHNICAL INFORMATION

- 2.1 List types and major characteristics of computers used in the operational system.
- 2.2 List types and major characteristics of computers used in the support subsystems. Indicate nature of support (e.g., Automatic Test Equipment, crew training/simulation, software development support, software test support, software maintenance support, other).
- 2.3 List major characteristics of each software component used in operational and support subsystems (e.g., program size, data base size ...). A component here means a major segment; or software subsystem; or separately developed, purchased, or supplied software package; or other breakout appropriate to your program. For each component its category should be identified. For example:

. Operational Program - system software

- Operational Program - application, function or mission specific software
- Automatic Test Equipment
- Crew Training/Simulator
- Software Development Support
- Software Test Support
- Software Maintenance Support)

- 2.4 List major software production tools used in development of operational and support systems (e.g., higher order languages, utility software packages, operating systems).
- 2.5 List major software maintenance tools used (or planned for use) during maintenance of system. Indicate which are different from or the same as those of 2.4.
- 2.6 Describe special support facilities required to support the system software V&V, training, and/or ongoing maintenance requirements (e.g., avionics simulation facility, crew training simulators).

3. SOFTWARE ACQUISITION PROCESS

- 3.1 Discuss how/when/to-what-degree software was addressed during each major step of the Service and OSD system acquisition review process. Specifically:
- When were operational and support software requirements first identified?
 - Were hardware/software tradeoff studies performed during the early system concept definition phase in order to arrive at most cost-effective design approach? By whom?
 - How were life cycle software costs considered in reaching the system design approach?
 - Describe level of computer hardware and software material (backup data) that was presented (available) to management at each (DSARC) decision point.

- Were use of existing military and/or commercially available computer hardware and software considered?
 - Were existing standard hardware and software packages developed for or used in other systems considered for this system? Which ones? Were they used?
 - How was software performance validated before proceeding with each new system phase? Were the results satisfactory?
- 3.2 Describe how software requirements were developed and who was responsible for controlling them during each acquisition phase.
- 3.3 Were quantitative software performance standards (e.g., program size, response times, throughput criteria) established for the operational and support software before the contract? Were they met? Reasons not met?
- 3.4 Were software reliability and maintainability requirements established before or during contract? Were they verified prior to operational transition?
- 3.5 Were software development schedules established? Where they met? Major reasons not met?
- 3.6 How were operational and support software packages contracted for (e.g., CPFF under prime contractor, separate software contractor, in-house)?
- 3.7 Was software largely done on contractor 'best effort' rather than by formal end product acceptance?
- 3.8 Describe major steps (milestones) in software contractor's development process.
- 3.9 Did software proceed through prototype development phase before production?
- 3.10 Did software requirements significantly change over the development, production phase? Were these changes responsible for extra costs or schedule delays? How could they have been prevented?

- 3.11 Describe the Service, contractor software design and production review process. Was it successful in identifying problems/issues early?
- 3.12 How was software progress reported during the software design, coding, and checkout phases? Was the system manager able to successfully gauge software progress?
- 3.13 Was software in the critical development path (e.g., in series with hardware) or was it successfully developed in parallel?
- 3.14 Was software cost, schedule, and performance data kept during the development, production and operational phases (e.g., data collected on programmer productivity, errors and changes to software during each phase, software costs by package or phase)? Is this data available?
- 3.15 Provide subjective opinion on the 'quality' of the operational and support software packages developed for this system.

4. OPERATIONS AND MAINTENANCE (O&M) EXPERIENCE

- 4.1 At what point were software maintenance support plans first developed? Are they adequate?
- 4.2 Is software maintenance performed centrally (depot) or at each field facility?
- 4.3 Discuss the policy followed on use of contractor vs. in-house support for software maintenance. Were tradeoff studies for this system performed in reaching this policy?
- 4.4 If performed in-house, list the separable software maintenance facilities and skills required for the total system. Comment on problems in acquiring and training the necessary software personnel.
- 4.5 Is software cost/performance data collected during the O&M phase in order to identify needed changes, improvements? What method(s) or system(s) of collection were used?
- 4.6 In question 2.5, the major maintenance tools were identified. Identify here which are adequate and which need improvement. Identify whether each is owned by and maintained by:

- a. a contractor
- b. a government program, agency or service other than this program
- c. the government under the supervision of this program.

5. SOFTWARE COST DATA

- 5.1 Provide estimates of total system cost for each phase (as defined in question 1.1).
- 5.2 Provide estimates of direct software costs for following cost categories:
 - Over each phase (as defined in question 1.1)
 - For each major software component (see question 2.3)
 - Budgeted vs. actual and reasons for major variance.
- 5.3 Estimate indirect software costs due to system delays, loss of mission life or effectiveness.
- 5.4 If above cost information is not available, provide subjective opinion of where major software costs occur in above categories.

6. OPINIONS ON AREAS OF FUTURE IMPROVEMENTS

- 6.1 Based upon your experience with this system, describe how the DoD software acquisition process can be improved in the future.
- 6.2 Are there areas where R&D projects should be initiated in order to improve the software acquisition process?
- 6.3 Which software management tools/methods have proved most effective?
- 6.4 Do you have specific problems you feel study should address?
- 6.5 Opinions on:
 - standardization?

- transfer/sharing of successful practices?
- improving/streamlining acquisition process?
- procurement practices/incentives?
- organizational improvements?

Note: Table A-1 is a copy of a "Software Cost Worksheet" provided at the weapon systems interviews for collecting cost information.

TABLE A 1
SOFTWARE COST WORKSHEET

BREAKDOWN		SOFTWARE COST WORKSHEET												2/14/75
		Software Related Costs Over Major Life Cycle Phases ^{1,2}												
LIFE CYCLE PHASES														
FISCAL YEARS														
1.	Government Software Management Costs (Including separate consultants, independent evaluation, etc.)													
2.	Operational Software Development Costs (Include government and contractor costs to specify, design, develop, and test the operational software)													
	<ul style="list-style-type: none"> . Development Tools (e.g., compilers, utility systems) . Operating System(s) . Application Software . Testware 													
3.	Separate Government (and Contractor) V,VAC Software Related Costs (Include software and facilities not included in 1 or 2 above)													
4.	Software Related Operational Support Costs													
	<ul style="list-style-type: none"> . Costs to develop and update software maintenance tools not included in 2 above . Costs to develop and update ATE and Diagnostic software . Costs to develop and update Air Crew/Simulator software . Costs for operational software maintenance and related facilities 													
5.	Other (Explain)													
Total Software Related Costs														
Total System Costs														

1. Explain Future \$s (e.g., constant FY75 \$s, 4% inflation, etc.)

2. U = \$ Unknown (or not reconstructable); M = Not Applicable; E = Estimated; A = Actual

APPENDIX B

DISTRIBUTION LIST

MITRE/Washington

H. D. Benington
C. C. Grandy
C. A. Zraket
R. R. Everett
Washington Library

D-40

H. J. Kirshner
D. L. Bailey
J. P. Locher III (25)
D. W. Kelliher (125)
A. Asch (5)
R. F. Robinson
W. Potter
L. G. Culhane
F. C. Holland
J. Greenwood

MITRE/Bedford

W. Attridge
J. Clapp
T. Connors (5)
O. Kinney
E. Lafferty
N. Waks
R. R. Everett

MITRE/Washington

G. Raichelson
J. Summers
K. Rebibo
E. Sharp
T. Hilinski
J. K. Summers
R. P. Foreman
E. L. Rabben

DoD

Col. R. D. Hensley (30)