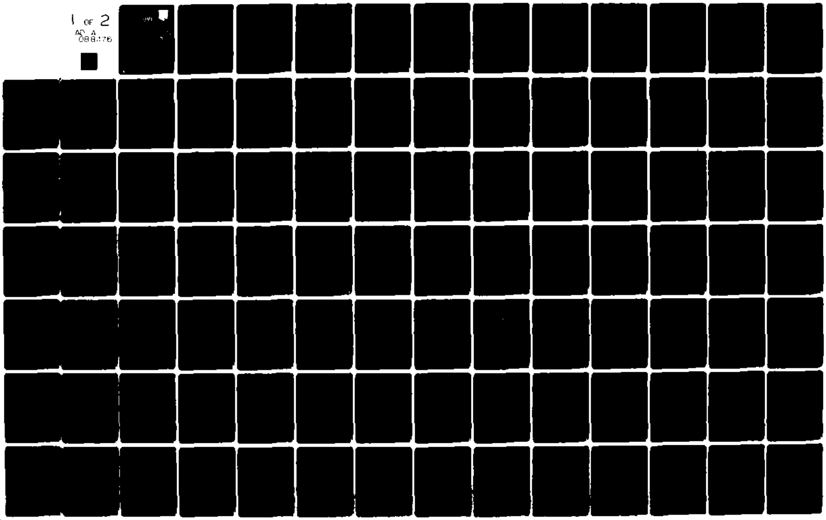
AD-A088 476 HUGHES AIRCRAFT CO CANOGA PARK CALIF F/G 14/1
PREDICTIVE SOFTWARE COST MODEL STUDY. VOLUME I. FINAL TECHNICAL--ETC(U)
JUN 80 R B WAINA, G L FOREMAN, A P BANGS F33615-79-C-1734
UNCLASSIFIED AFWAL-TR-80-1056-VOL-1 NL

1 OF 2
AD A
088476



AD A088476

AFWAL-TR-80-1056
Volume I

LEVEL II

AF WRIGHT AERONAUTICAL LABORATORIES

PREDICTIVE SOFTWARE COST MODEL STUDY

DTIC
ELECTE
AUG 29 1980
S C D

Hughes Aircraft Company
Support Systems
Canoga Park, California 91304

JUNE 1980

VOLUME I: FINAL TECHNICAL REPORT
Final Report for Period 2 April 1979 - 2 June 1980

Approved for Public Release; Distribution Unlimited

DDC FILE COPY

Avionics Laboratory
Air Force Wright Aeronautical Laboratories
Air Force Systems Command
Wright-Patterson Air Force Base, Ohio 45433

80 8 28 022

NOTICE

When Government drawings, specifications, or other data are used for any purpose other than in connection with a definitely related Government procurement operation, the United States Government thereby incurs no responsibility nor any obligation whatsoever; and the fact that the government may have formulated, furnished, or in any way supplied the said drawings, specifications, or other data, is not to be regarded by implication or otherwise as in any manner licensing the holder or any other person or corporation, or conveying any rights or permission to manufacture use, or sell any patented invention that may in any way be related thereto.

This report has been reviewed by the Office of Public Affairs (ASD/PA) and is releasable to the National Technical Information Service (NTIS). At NTIS, it will be available to the general public, including foreign nations.

This technical report has been reviewed and is approved for publication.

Daniel V. Ferens

DANIEL V. FERENS
Work Unit Engineer
AFWAL/AAAS-2

Diane E. Summers

DIANE E. SUMMERS, Tech Mgr
Concepts and Evaluation Group
System Integration Branch

FOR THE COMMANDER

Raymond E. Siferd

RAYMOND E. SIFERD, Colonel, USAF
Chief, System Avionics Division
Avionics Laboratory

"If your address has changed, if you wish to be removed from our mailing list, or if the addressee is no longer employed by your organization please notify AFWAL/AAA *, W-PAFB, OH 45433 to help us maintain a current mailing list".*

Copies of this report should not be returned unless return is required by security considerations, contractual obligations, or notice on a specific document.

(11) TR-80-1056-Vol-1 (9) Final rept. 2 Apr 79-2 Jun 80

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER: (18) AFWAL-TR-80-1056, Vol. I	2. GOVT ACCESSION NO.: AD-A088 476	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle): (6) PREDICTIVE SOFTWARE COST MODEL STUDY. Volume I. Final Technical Report.		5. TYPE OF REPORT & PERIOD COVERED: Final Report 2 April 1979 to 2 June 1980
		6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s): R.B. Waina, G.L. Foreman, A.P. Bangs, J.L. Green, E.E. Rodriguez		8. CONTRACT OR GRANT NUMBER(s): (15) F33615-79-C-1734
9. PERFORMING ORGANIZATION NAME AND ADDRESS: Hughes Aircraft Company, ~~Support Systems~~, Canoga Park, California 91304 (12) 191		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS: P. E. 62204F, Project 2003, Task (16) 200309, Work Unit 20030913 (17)
11. CONTROLLING OFFICE NAME AND ADDRESS: Avionics Laboratory of Air Force Wright Aeronautical Laboratories (AFWAL/AAA), Air Force Systems Command, Wright-Patterson AFB, Ohio 45433		12. REPORT DATE: (11) June 1980
		13. NUMBER OF PAGES: 182
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office): (10) Richard E. / Waina, Gary L. / Foreman, Alan P. / Bangs, James L. / Green, Esperanza E. / Rodriguez		15. SECURITY CLASS. (of this report): Unclassified
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

This technical report is also composed of Volume II, Software Package Detailed Data.

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Software Support
Embedded Computer Systems
Avionics Support
Life Cycle Cost
Cost Effectiveness

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

Current cost estimating techniques do not adequately address the problem of predicting avionics embedded software support costs. This limitation results primarily from an inadequate understanding of how those costs are generated and from a scarcity of historical data on avionics software support and the resultant costs. The objectives of this Phase I study of the Predictive Software Cost Model (PSCM) program were to determine the feasibility of a model to predict those costs and to generate a roadmap →

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

411 118

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered) for development of such a model. The feasibility of such a model was established. It will include the six key resource types required to support avionics software: personnel, support hardware, support software, facilities (buildings), program documentation and flight test aircraft/ranges. Some preliminary estimating relationships were identified. A detailed roadmap for developing the model was generated. Phase II of the PSCM program will provide an operating model for predicting avionics embedded software support costs.

PREFACE

The Predictive Software Cost Model Study Phase I Technical Report is prepared in two separately bound volumes.

Volume I - Final Technical Report

Volume II - Software Package Detailed Data

The Air Force Program Monitor was Mr. Daniel V. Ferens, Systems Evaluation Group, Avionics Systems Engineering Branch (AFWAL/AAAA-3).

SUMMARY

The Avionics Laboratory of the Air Force Wright Aeronautical Laboratories (AFWAL/AA) has identified a requirement to predict support costs of avionics embedded computer system software. The requirement, further, is to be able to predict the software costs early in an advanced avionics system program during the development planning processes. The desire is to be able to evaluate alternate software and software support concepts early enough in their life cycles in order to influence the software, and perhaps hardware, designs to achieve required performance(s) at lowest life cycle cost.

The requirement can be satisfied by an appropriate predictive model. The question was whether such a model existed. If so, is it adequate? If not, can one be developed? AFWAL/AA initiated a two-phase Predictive Software Cost Model (PSCM) study program by means of RFP No. F33615-79-R-1734. Efforts and results of Phase I of the study program are the subject of this report.

The objective of Phase I of the study was to define the methodology for developing a model which will enable AFWAL/AA personnel to predict the support costs of computer software associated with avionics systems. The approach taken addressed two major elements inherent in the objective: establishing the feasibility of developing the desired model and, having done so, defining the methodology for model development in the form of a roadmap for conduct of Phase II of the PSCM program. The essence of the approach encompassed: 1) identifying and evaluating current and proposed practices regarding software support performance and cost estimating, 2) assessing availability of historical software support data, 3) determining feasibility of designing a software support cost prediction model based on available data, and 4) generating a methodology, i.e., a detailed roadmap, for Phase II model development.

The first step of the study was to establish the feasibility of designing a credible model that will predict software support costs with acceptable accuracy. To accomplish this, information was gathered and evaluated to determine whether available historical and other supplemental data is adequate as the basis for a predictive model design. The evaluation proved that a predictive software support cost model design is, indeed, feasible.

Given feasibility, the second step was to define the methodology for developing the model in the form of a detailed roadmap to guide the model design during Phase II of the program.

Establishing model feasibility included several phases: 1) a technology review, 2) field surveys and 3) an analysis of findings.

Technology review assessed the current state of software cost estimating from the standpoint of relevant literature as well as existing software cost prediction models. Numerous studies, reports, journals and periodicals were researched to identify and assess various cost estimating techniques, significant cost drivers, typical cost distributions, possible parameter/cost relationships, and possible data sources. The assessment of relevance to avionics embedded software support cost estimation gave direction to subsequent efforts. Also, twenty-one software cost prediction models were examined to identify commonalities and differences in cost estimating approaches. Results showed that adequate prediction models exist for software development costs but not for software support costs. A main problem was the lack of historical software support data and good metrics for software quality, reliability and maintainability.

The total complement of Air Force avionics software is estimated to number more than 50,000 separate pieces. They are generally categorized as: 1) operational flight programs (OFPs), 2) electronic warfare (EW) programs, 3) communication electronics (CE), including airborne command, control, and communications (C^3) programs, 4) aircrew training device (ATD) programs, and 5) automatic test equipment (ATE) programs.

Field surveys were conducted to determine current processes of avionics embedded software support within the Air Force, and how these processes interact with and affect resultant software support costs. A team of contractor personnel visited each Air Logistics Center, interviewed key software support personnel, and collected data on six major relevant sample software packages: 1) A-7D OFP, 2) F-111F OFPs, 3) FB-111A OFPs, 4) F/FB-111 Support Software, 5) F-15 OFPs, and 6) F-16 OFPs. Limited data was also collected in the EW and ATE areas, and additional data sources were identified.

An analysis of findings from the technology review and field surveys was performed to assess the key factors affecting avionics embedded software support costs. Results revealed six primary resource categories to be considered in the model design: 1) personnel, 2) support equipment, 3) support software, 4) facilities, 5) data and documentation, and 6) flight test. The primary support cost drivers of these resource categories were determined to be: 1) change requirements, both frequency and size, 2) prime system software, including program size, architecture, and hardware constraints, 3) prime system hardware, 4) software support personnel, including experience, training and productivity, and 5) test requirements for software changes.

Findings also indicated, from the data sample and the identification of additional available software support performance data, that adequate data of sufficient quality are available upon which to design a predictive software cost model that emphasizes software support costs.

To define the model roadmap, several alternate modelling approaches for developing the predictive software cost model using the available data were evaluated and the best alternative was selected. The recommmended approach defines a methodology which embodies five fundamental tasks: 1) collect (additional) data, 2) analyze all data, both quantitatively and qualitatively, 3) develop a data base that will be interfaced automatically by the model, 4) design the predictive software cost model using both statistical and qualitative analysis techniques, and 5) perform verification and validation testing of the model design. The methodology is presented in the form of a detailed roadmap for conducting the Phase II model development. Upon completion of model design, provision is made for installing the model in the host computer at Wright Patterson Air Force Base, and for training programmer and user personnel.

A preliminary estimate of resource requirements for performing Phase II efforts showed slightly more than 4.5 personyears of engineering labor plus associated clerical, travel and computing costs.

Primary risks in conducting Phase II in accordance with the recommended methodology center about three areas: 1) quality and quantity of available data being less than expected, 2) constraints on model usage imposed by the design approach which provides ease of use by means of minimum input requirements, and 3) model misuse. These risks, however, are not peculiar to the recommended approach; they are risks for any model development approach.

PREFACE TO VOLUME I

This final report was prepared by Support Systems of Hughes Aircraft Company at Canoga Park, California 91304. This study examined the feasibility of predicting avionics embedded software support costs and generated a roadmap for developing a model to predict such costs. The program began with contract go-ahead on 2 April 1979 and was completed on 2 June 1980.

Mr. Daniel V. Ferens was the Project Engineer from the Avionics Laboratory of the Air Force Wright Aeronautical Laboratories (AFWAL/AA). Under his direction and with the cooperation of Ms. Diane Summers and the personnel of the Air Force Logistics Command located at Tinker AFB, Oklahoma; China Lake Naval Weapons Center, California; McClellan AFB, California; Robins AFB, Georgia; Hill AFB, Utah; and Kelly AFB, Texas, the necessary data on Air Force Support of avionics embedded computer system software were gathered for this study.

The Support Systems and Maintainability Engineering Laboratory under the management of Mr. Robert W. Rowe was responsible for program execution. The program manager was Mr. Ercell C. Hamilton. Dr. Richard B. Waina was the principal investigator. He was ably assisted by Mr. Alan P. Bangs, Mr. Gary L. Foreman and Ms. Esperanza E. Rodriguez. Messrs. James L. Green and Russell A. Vande Steeg provided additional support which contributed to the overall success and content of this report.

* COST EFFECTIVENESS
* AVIONICS
* ~~COMPUTER PROGRAM DOCUMENTATION~~

COMPUTER [illegible] / [illegible]

LIFE CYCLE COSTS

FLOW CHARTING

COST ESTIMATING

* COST MODELS

COST ANALYSIS

ECONOMIC [illegible]

DATA BASES

ALGORITHMS

[illegible]

10/

PSCM (PREDICTIVE SOFTWARE [illegible] MODEL)

[illegible] COMPUTERS

[illegible]

TABLE OF CONTENTS

TABLE OF CONTENTS (Continued)

TABLE OF CONTENTS (Continued)

LIST OF ILLUSTRATIONS

LIST OF TABLES

LIST OF TABLES (Continued)

I. INTRODUCTION

There have been spectacular advances in computer technologies applied in Air Force avionics systems and subsystems in recent years. Since software (i.e., computer programs) is an increasingly significant component of these avionics systems, there have been dramatic rises in the costs of developing and supporting the associated computer software. Accordingly, the Avionics Laboratory of the Air Force Wright Aeronautical Laboratories (AFWAL/AA) and the Air Force in general need to be able to predict the software costs in planning the development of advanced avionics systems.

Software life cycle cost estimating is hampered, however, by a lack of definitive historical data on software operation and support and the associated costs. Also, there is insufficient understanding of the factors that drive and otherwise affect those costs. Although there is some knowledge of cost distributions, and some feel for what the significant cost drivers are, there is no widely-accepted well-validated model for early estimation of avionics embedded software support costs.

There are several models and cost-estimating approaches used by AFWAL/AA to estimate costs of computer hardware and software (see Table 1); however, none of the models or approaches predicts support costs of avionics software to a sufficient degree of accuracy, nor adequately takes into account the effects of various developmental phase concepts on cost-of-ownership of avionics software. The models also do not satisfactorily address support policies and operational and support software requirements for avionics systems.

TABLE 1. AFWAL/AA PREDICTIVE LCC MODELS

	Hardware	Software
Development/ Production	PRICE-H	PRICE-S WOLVERTON
Operation & Support	ALPOS PRICE-L SAVE STEP	? ? ?

AFWAL/AA needs software support cost predictions to:

* Evaluate software design alternatives (e.g., higher-order versus machine language)

* Evaluate software support concept alternatives (e.g., in-house versus contractor support)

* Make total software support cost projections for DSARC and preliminary budget planning purposes.

These cost predictions must have an acceptable degree of accuracy and must be available in a timely manner during the conceptual and early design phases for effective conduct of cost-performance tradeoffs. In order to acquire this capability AFWAL/AA initiated the Predictive Software Cost Model Program in two phases. Phase I was to study the feasibility of such a model, and generate a roadmap for its development. Phase II will result in an operational model.

II. BACKGROUND

OBJECTIVES OF STUDY

Phase I of the Predictive Software Cost Model (PSCM) program has two major objectives:

(1) to establish the feasibility of developing a model which will enable AFWAL/AA personnel to predict the support costs of embedded computer software associated with avionics systems and subsystems,

(2) to define the methodology for model development in the form of a roadmap for conduct of Phase II of the PSCM program.

SCOPE OF STUDY

The scope of this study was limited to the area of support costs for avionics embedded software. Although the contractual statement of work referred to "operations and support costs," it was determined that operations costs for avionics embedded software are basically non-existent, being subsumed under hardware costs. The Phase I study addressed the following tasks:

* Review of existing software cost estimating capabilities, including models.

* Field surveys of avionics embedded software support facilities to identify software packages, to develop an understanding of software support policies and procedures, and to collect sample historical data.

* Analysis of the collected data to determine model feasibility, identify support cost drivers, evaluate alternate cost estimating approaches, and select the best approach.

* Definition of a methodology in the form of a roadmap for developing an avionics software support cost estimating model based on the selected approach.

Although prediction of software development costs is not directly included in the scope of the PSCM program, the effect of certain developmental phase concepts such as structured programming, top-down design and higher-order language utilization in software support cost estimation were considered.

ORGANIZATION OF THE REPORT

The above tasks are discussed in more detail in subsequent sections of this report, which is organized as follows:

Section III, Technical Approach, describes the methodology used in this phase to determine the feasibility of predicting avionics embedded software support costs. It also describes the approach to defining a roadmap for development of a model to predict those costs.

Section IV, Survey of Current Approaches in Software Cost Estimating, relates the findings resulting from the review of software cost estimating technology and knowledge.

Section V, Field Survey Findings, describes the data collected from the various Air Logistics Centers (ALCs). Both software packages and software support policies and procedures are discussed.

Section VI, Analysis of Data and Cost Drivers, discusses significant factors which must be considered in cost estimating, based on the results of the literature search and field surveys.

Section VII, Feasibility of Estimating Software Support Costs, establishes feasibility, defines and evaluates alternate approaches to cost estimating, and identifies the selected approach.

Section VIII, Definition of Phase II Roadmap, details a methodology for developing a model to predict avionics embedded software support costs.

Section IX, Conclusions, presents the major conclusions resulting from the study.

Section X, Observations and Recommendations, contains observations made during the course of the field surveys which relate to the overall efficiency of the avionics software support process. Recommendations are also provided.

Volume II of this report contains the detailed data on software support agencies and sample software packages which were gathered during the field surveys.

III. TECHNICAL APPROACH

This section describes the procedures used in conducting the study. The results of the study are described in later sections.

GENERAL APPROACH

This study has two major objectives: 1) establish the feasibility of predicting avionics embedded software support costs and 2) define a methodology to develop the Predictive Software Cost Model. The study objectives were met by applying a systems approach similar to other system engineering efforts. The approach included:

* Examination of the requirement
* Study of current technology/processes
* Definition of candidate solutions/systems
* Evaluation of alternate approaches
* Selection of the alternative which best satisfied the requirement
* Development of a roadmap for the selected approach

The application of this approach to the problem is illustrated in Figure 1, which identifies the basic tasks performed.

The requirement for a model to predict avionics embedded software support costs was delineated in RFP #F33615-79-R-1734. This requirement was discussed in greater detail in the initial meeting with AFWAL/AA in order to assure mutual understanding of the objectives of the study.

The initial task in the study was to investigate the current technology of software cost estimating. This first task involved a review of existing cost estimating approaches and an assessment of their relevance and usefulness to the job of predicting avionics embedded software support costs.

Field surveys were then performed to determine current processes of avionics embedded software support within the Air Force, and how those processes interact with and affect the resultant software support costs. Data on particular sample software packages were acquired as part of the surveys. Sources of historical data on software support were also identified.

The data resulting from the first two tasks were then analyzed to determine key factors affecting avionics embedded

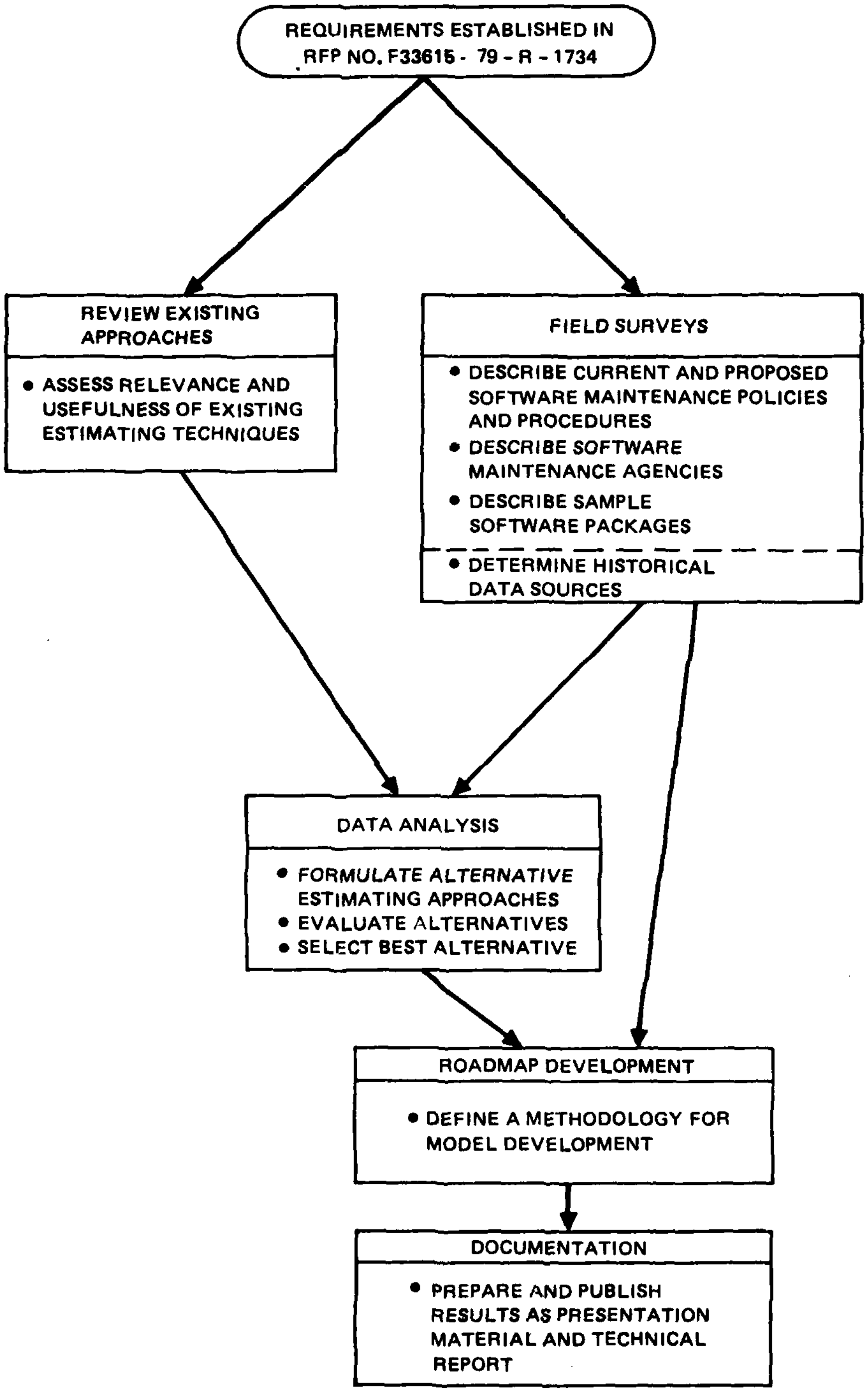


Figure 1. Predictive Software Cost Model Study Approach

software support costs. The feasibility of predicting those costs was determined. Various approaches to estimating software support costs were formulated and evaluated according to criteria established by AFWAL/AA and Hughes. The best approach was identified and a roadmap was then prepared for development of a PSCM based on that approach.

The actual execution of this approach was aided through inputs from three recent Hughes studies: 1) Software Logistics, 2) Software Cost Factors, and 3) Design-For-Repair Concept Definition. An on-going Software Logistics study provided background data on software support processes and cost estimating technology. An IR&D study on Software Cost Factors provided insight into software cost distribution and cost drivers. Experience gained conducting field surveys during the Design-For-Repair Concept Definition study helped to make the field surveys for this study more efficient and productive.

A detailed flow of the methodology used to conduct Phase I of the PSCM study is described in Figure 2. This flow diagram is the focal point for subsequent discussion of the technical approach; it delineates the complexity and scope of the effort.

TASK I: SOFTWARE COST ESTIMATING APPROACH REVIEW

The review of software cost estimating approaches started with a literature search, as indicated in Figure 2. This search utilized sources such as the Defense Logistics Studies Information Exchange and the Defense Documentation Center. Applicable studies and reports were researched, as were various journals and periodicals such as Datamation, Computer, and IEE Transactions on Software Engineering. Proceedings of various conferences and symposia were also reviewed. The bibliography contained in this report identifies the articles and studies reviewed. The objective of the literature search was to identify:

* Various cost estimating techniques
* Significant cost drivers
* Typical cost distributions
* Possible relationships between various parameters and cost
* Possible data sources

Data extracted from the articles and studies were compiled and cost estimating approaches identified. The relevance of the data and cost estimating approaches to avionics embedded software support cost estimation was also assessed. This assessment gave some indication of the direction to be taken in developing an avionics software support cost estimating model. The results of this review are documented in Section IV.

TASK II: FIELD SURVEYS

A major portion of this study was devoted to field surveys for collection of data on avionics embedded software support. The survey phase involved three subtasks:

* Software package identification
* Software package selection
* Software maintenance site visitation

Software Package Identification

This task was to identify avionics software packages within the Air Force inventory of weapons systems. The Air Force categorizes embedded computer system (ECS) software as follows:

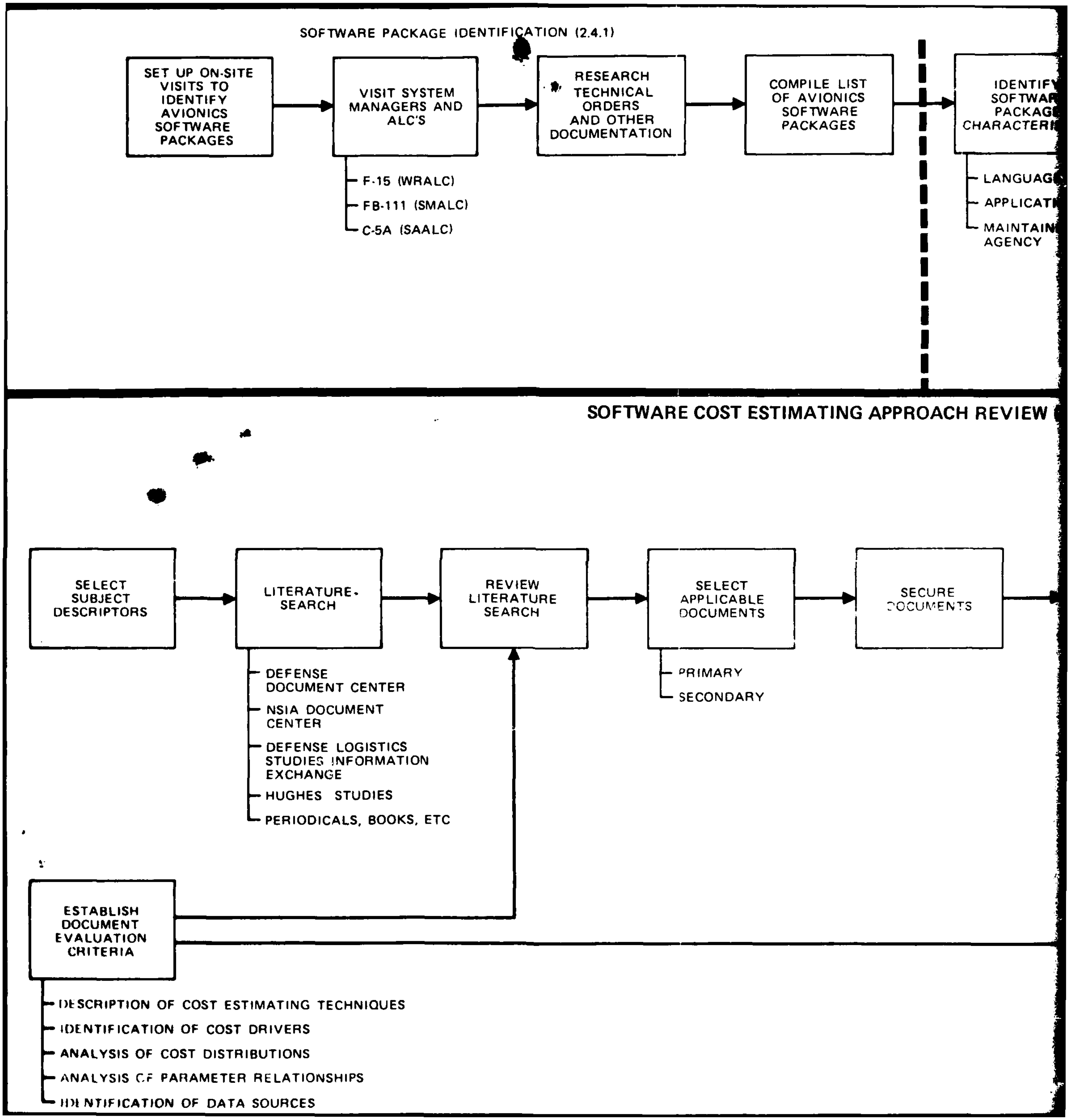
* Operational flight programs (OFPs)
* Electronic warfare (EW) programs
* Communication electronics (CE), including airborne command, control, & communication (C^3) programs
* Aircrew training device (ATD) programs
* Automatic test equipment (ATE) programs

AFWAL/AA and Hughes decided to concentrate the major effort in this study on OFPs, while collecting some data on EW and ATE software support processes. This plan offered the most cost-effective approach to developing a software support cost model roadmap.

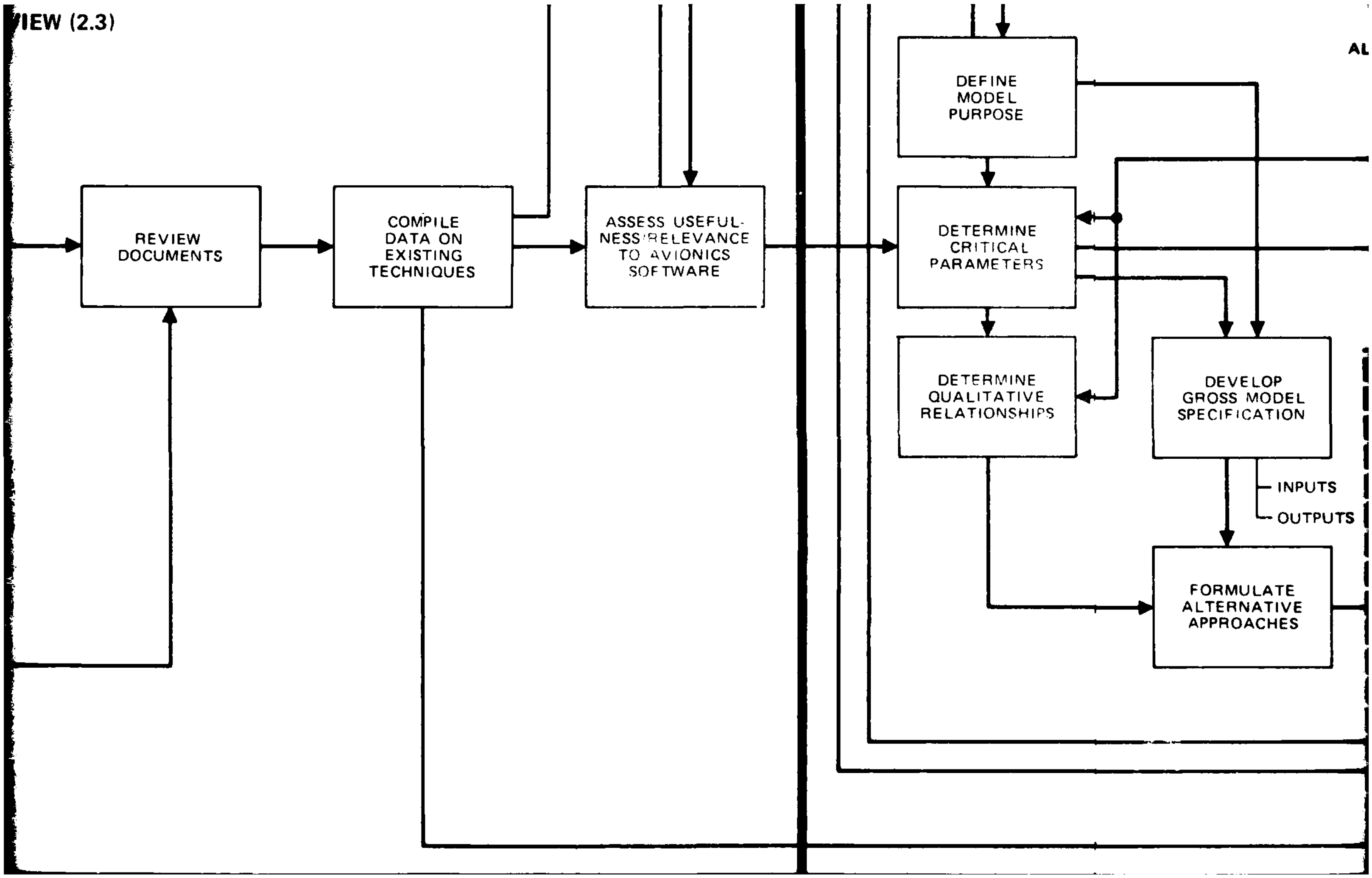
Software Package Selection

Hughes identified a representative sample of software packages for detailed study. Individual package characteristics such as program language, application, maintaining agency and responsible ALC were first considered. Candidates were nominated from those identified. These candidates were then discussed with AFWAL/AA at the Software Selection Conference. Selection of six packages was approved by the AFWAL/AA Program Manager. These were:

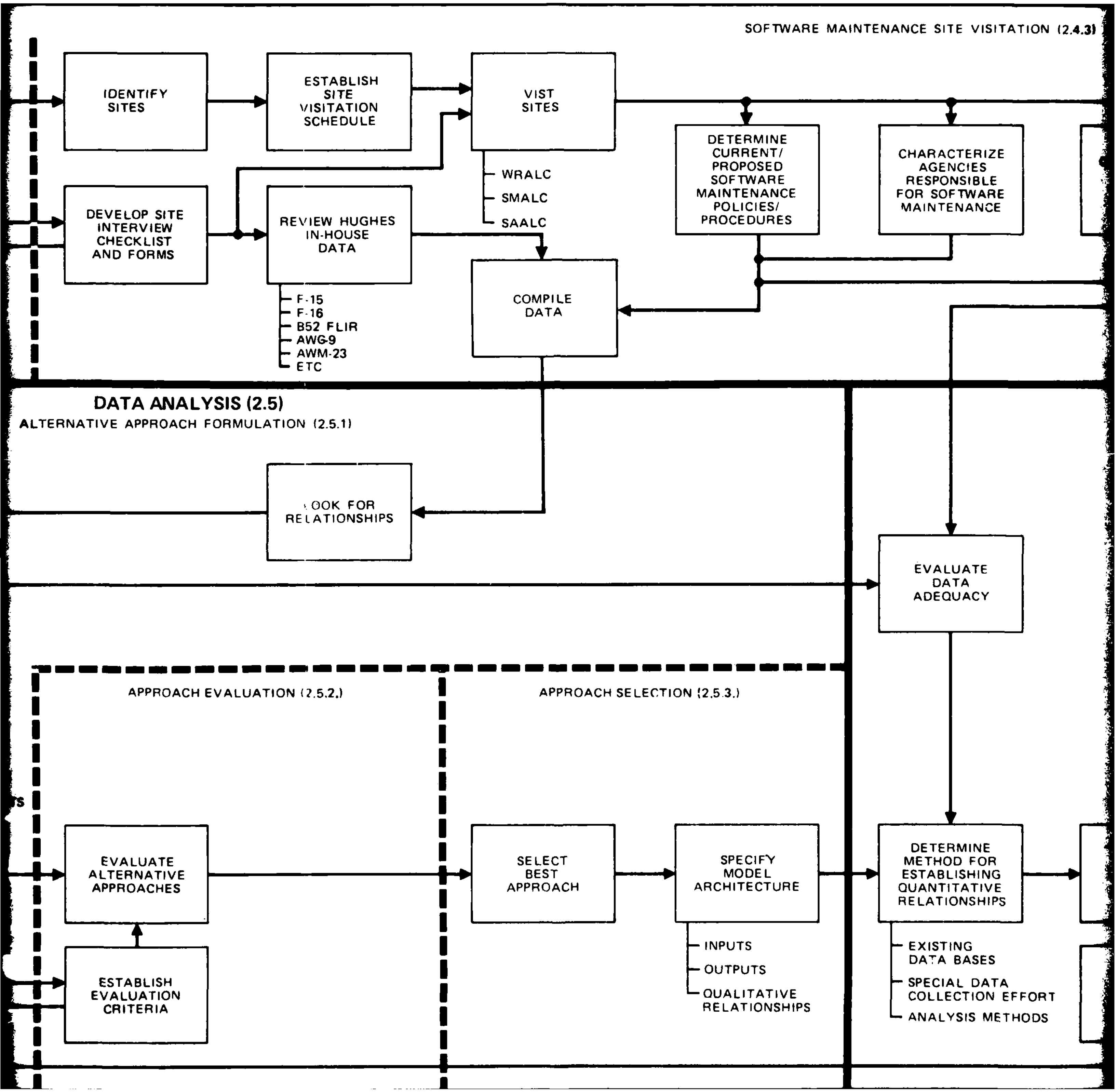
* A-7D OFP
* F-111F OFPs
* FB-111A OFPs
* F/FB-111 Support Software

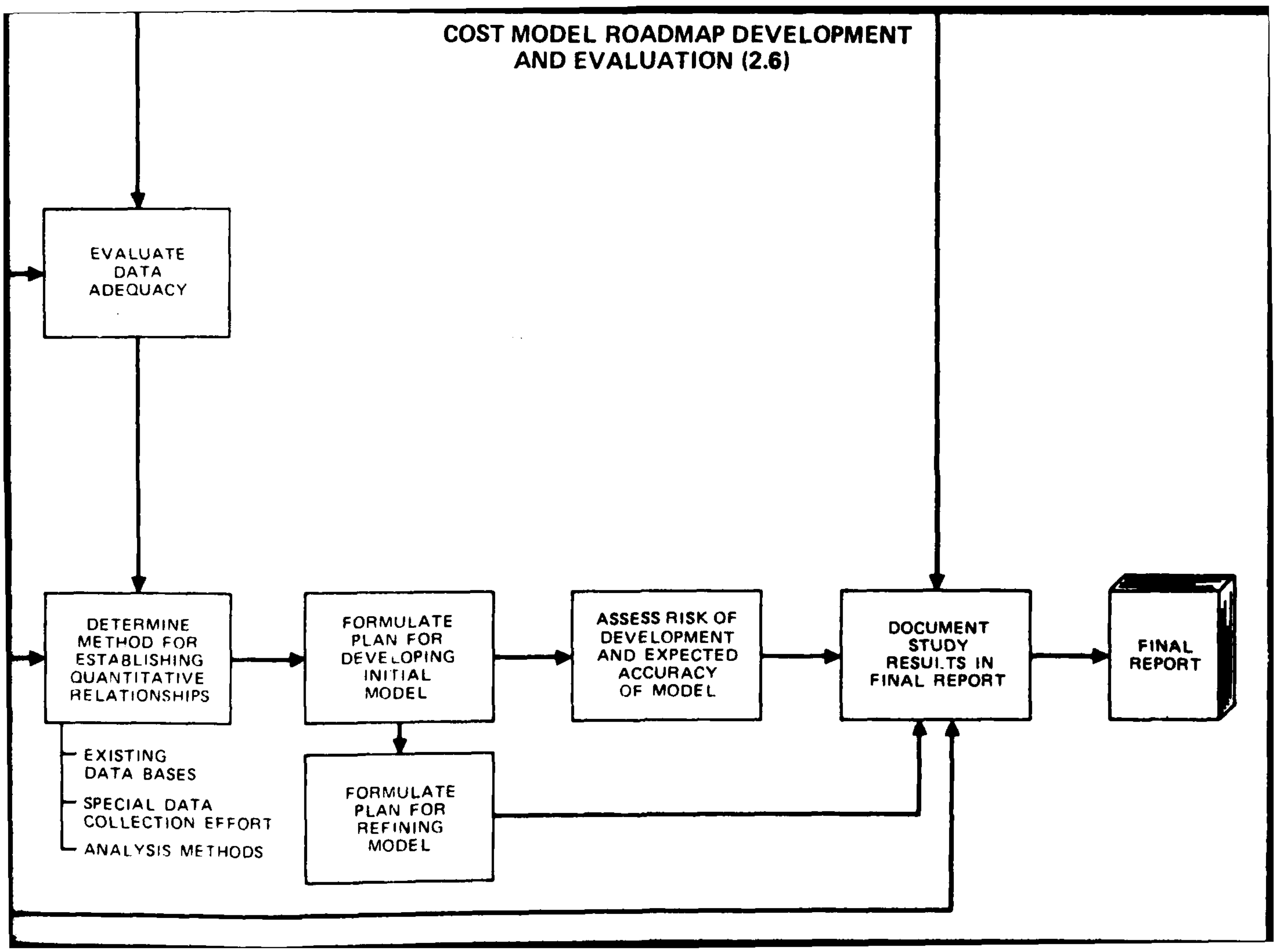

SOFTWARE PACKAGE IDENTIFICATION (2.4.1)
SET UP ON-SITE VISITS TO IDENTIFY AVIONICS SOFTWARE PACKAGES
VISIT SYSTEM MANAGERS AND ALC'S
F-15 (WRALC)
FB-111 (SMALC)
C-5A (SAALC)
RESEARCH TECHNICAL ORDERS AND OTHER DOCUMENTATION
COMPILE LIST OF AVIONICS SOFTWARE PACKAGES
IDENTIFY SOFTWAR PACKAGE CHARACTERI
LANGUAG
APPLICATI
MAINTAIN AGENCY
SOFTWARE COST ESTIMATING APPROACH REVIEW
SELECT SUBJECT DESCRIPTORS
LITERATURE SEARCH
DEFENSE DOCUMENT CENTER
NSIA DOCUMENT CENTER
DEFENSE LOGISTICS STUDIES INFORMATION EXCHANGE
HUGHES STUDIES
PERIODICALS, BOOKS, ETC
REVIEW LITERATURE SEARCH
SELECT APPLICABLE DOCUMENTS
PRIMARY
SECONDARY
SECURE DOCUMENTS
ESTABLISH DOCUMENT EVALUATION CRITERIA
DESCRIPTION OF COST ESTIMATING TECHNIQUES
IDENTIFICATION OF COST DRIVERS
ANALYSIS OF COST DISTRIBUTIONS
ANALYSIS OF PARAMETER RELATIONSHIPS
IDENTIFICATION OF DATA SOURCES

IEW (2.3)



AL
REVIEW DOCUMENTS
COMPILE DATA ON EXISTING TECHNIQUES
ASSESS USEFUL-NESS/RELEVANCE TO AVIONICS SOFTWARE
DEFINE MODEL PURPOSE
DETERMINE CRITICAL PARAMETERS
DETERMINE QUALITATIVE RELATIONSHIPS
DEVELOP GROSS MODEL SPECIFICATION
INPUTS
OUTPUTS
FORMULATE ALTERNATIVE APPROACHES


SOFTWARE MAINTENANCE SITE VISITATION (2.4.3)
IDENTIFY SITES
ESTABLISH SITE VISITATION SCHEDULE
VIST SITES
WRALC
SMALC
SAALC
DETERMINE CURRENT/ PROPOSED SOFTWARE MAINTENANCE POLICIES/ PROCEDURES
CHARACTERIZE AGENCIES RESPONSIBLE FOR SOFTWARE MAINTENANCE
DEVELOP SITE INTERVIEW CHECKLIST AND FORMS
REVIEW HUGHES IN-HOUSE DATA
F-15
F-16
B52 FLIR
AWG-9
AWM-23
ETC
COMPILE DATA
DATA ANALYSIS (2.5)
ALTERNATIVE APPROACH FORMULATION (2.5.1)
LOOK FOR RELATIONSHIPS
EVALUATE DATA ADEQUACY
APPROACH EVALUATION (2.5.2.)
APPROACH SELECTION (2.5.3.)
EVALUATE ALTERNATIVE APPROACHES
ESTABLISH EVALUATION CRITERIA
SELECT BEST APPROACH
SPECIFY MODEL ARCHITECTURE
INPUTS
OUTPUTS
QUALITATIVE RELATIONSHIPS
DETERMINE METHOD FOR ESTABLISHING QUANTITATIVE RELATIONSHIPS
EXISTING DATA BASES
SPECIAL DATA COLLECTION EFFORT
ANALYSIS METHODS



COST MODEL ROADMAP DEVELOPMENT AND EVALUATION (2.6)
EVALUATE DATA ADEQUACY
DETERMINE METHOD FOR ESTABLISHING QUANTITATIVE RELATIONSHIPS
EXISTING DATA BASES
SPECIAL DATA COLLECTION EFFORT
ANALYSIS METHODS
FORMULATE PLAN FOR DEVELOPING INITIAL MODEL
FORMULATE PLAN FOR REFINING MODEL
ASSESS RISK OF DEVELOPMENT AND EXPECTED ACCURACY OF MODEL
DOCUMENT STUDY RESULTS IN FINAL REPORT
FINAL REPORT

* F-16 OFPs

* F-15 OFPs

In addition to these software packages, AFWAL/AA and Hughes decided that preliminary information should be collected regarding EW and ATE software support.

Software Support Site Visitation

The particular sites to be visited were based on support responsibility for the software. Support sites were:

Software Support Site	Packages
Oklahoma City ALC (OC-ALC)	A-7D
China Lake NWC (OC-ALC/MMECZA)	A-7D
Sacramento ALC (SM-ALC)	F/FB-111
Warner-Robins ALC (WR-ALC)	F-15, EW, ATE
Ogden ALC (OO-ALC)	F-16
San Antonio ALC (SA-ALC)	ATE

Once the sites were selected a base visitation schedule was prepared and coordinated with AFWAL/AA. The final visit schedule was also coordinated with AFLC and the selected bases to ensure availability of cognizant personnel.

Four major data collection objectives were established for the site visits:

* Determine current and proposed software maintenance policies and procedures

* Identify software maintenance agencies

* Define characteristics of the sample software packages

* Identify sources of historical data on software maintenance actions and costs.

Data were collected at the maintenance sites using a structered interview process. Field evaluation forms specifically designed for this study were used to provide a structured set of questions covering desired aspects of avionics embedded software support. These questions were presented by the interviewer who then annotated the interviewee's responses, both specific and general, on the forms. Various regulations and procedures were also reviewed. The basic categories of information gathered were:

* General Software Package Description
* Maintenance Agency Personnel
* Maintenance Agency - Work Distribution
* Maintenance Agency - Cost Accounting System
* Maintenance Agency - Policies & Procedures
* Personnel Description
* Software Package Characteristics - Facilities
  Buildings
  Computing Equipment
* Software Package Characteristics - Support Software
* Software Package Characteristics - Flight Test Requirement
* Software Package Characteristics - Training Requirements
* Software Package Maintenance History
* Software Package Maintenance Cost History
* Availability and Sources of Historical Data
* Recommendations for Software Support Cost Predicting

The completed forms for each sample package, plus information on EW and ATE support, are included as Appendixes A through H of Volume II of this report. The basic findings of the field survey, including discussion of software population and the study sample, and description of the software life cycle and software support processes, are presented in Section V.

TASK III: DATA ANALYSIS

The data analysis phase involved five major subtasks:

* Identification of support cost drivers
* Evaluation of software support cost prediction feasibility
* Alternative approach formulation
* Approach evaluation
* Approach selection

Identification of Support Cost Drivers

A key element in the study was the identification of critical parameters which have major influence on software support costs. Factors affecting this task included:

* Relevance of current estimating approaches to avionics embedded software

* Detailed definition of model purpose

* Analysis of field survey data

In Task I, Software Cost Estimating Approach Review, the usefulness and relevance of current software cost estimating approaches to the problem of estimating avionics embedded software support costs were assessed. Critical parameters identified in that review are discussed in Section IV.

The purpose and proposed use of the Predictive Software Cost Model were reviewed at the Software Selection Conference. Consideration of the decision-making needs of the model users helped determine the critical parameters. The model is to be used primarily to assess the support cost impacts of various alternative avionics embedded software designs and support postures.

During the field surveys detailed data were gathered on software support policies, software support agencies, and specific software packages. Those data were analyzed to determine what variables have the greatest impact on support costs, and what the qualitative relationships are.

A preliminary model specification was developed based upon determination of the critical parameters as they relate to the model purpose, required outputs, and inputs.

The results of this analysis task are discussed in Section VI.

Evaluation of Software Support Cost Prediction Feasibility

The feasibility of implementing a cost estimating procedure is based on three elements:

* An understanding of the process by which those costs are generated.

* The availability of historical data describing the process.

* The development of an approach which considers a) the purpose and use of the estimating procedure, b) the input data available at the time when the procedure is to be used, and c) estimating algorithms which reflect the cost generation process.

These elements were evaluated in parallel with the process of identifying cost drivers and analyzing estimating approaches. The feasibility of estimating avionics embedded software support costs is discussed in Section VII.

<u>Alternate Approach Formulation</u>

The next task was to formulate alternative software support cost estimating approaches. Three basic approaches were investigated:

* Analogy

* Element estimate

* Cost estimating relationship

Those approaches are described in Section VII.

<u>Approach Evaluation</u>

The three estimating approaches listed above were evaluated on the basis of their strengths, weaknesses and model development risks. A detailed set of evaluation criteria was established. A panel of experienced modellers then judged each approach on each criterion. The evaluation is documented in Section VII.

<u>Approach Selection</u>

Each estimating approach was given a numerical rating on each criterion. An overall rating was then computed, based on the established weight for each criterion. The approach with the highest rating was then considered as the primary candidate for implementation. Care was taken to ensure that an approach which was completely unacceptable on a particular criterion could not achieve an otherwise acceptable rating. The selection methodology is described in detail in Section VII.

TASK IV: MODEL ROADMAP DEVELOPMENT AND EVALUATION

The final task was to define the methodology for developing the Predictive Software Cost Model. A key element within this task, given the model specification, is determination of the method for establishing the quantitative relationships necessary to implement the estimating approach. This depends on an evaluation of the adequacy of the data from the historical data sources identified during the field surveys. The adequacy of the data is a function of its quality, quantity, availability and format in relation to the critical parameters. It was deemed necessary to supplement the available historical data with a special data collection effort.

The next step was to formulate a roadmap for developing the Predictive Software Cost Model. This roadmap describes the collection of historical data on software support costs, analysis of the data to determine required cost factors and/or estimating equations, model design, model coding, and model verification. Also, this roadmap specifies the estimated manhours associated with the tasks, and the data required to support the model development. The risks associated with developing a predictive software cost model are also discussed. The roadmap is described in Section VIII.

IV. SURVEY OF CURRENT APPROACHES IN SOFTWARE COST ESTIMATING

INTRODUCTION

A review of current software development and support cost estimating methodologies and techniques indicates that a substantial effort will be required to develop the capability to effectively predict software costs. The growing number of papers dealing with this subject indicates that software managers and professionals have become keenly aware of the software cost prediction problem. Indeed, the vast majority of this literature is concerned with the problem, but unfortunately, only a small portion proposes quantitative methods or approaches to software cost prediction.

Although there is some understanding of the distribution of software costs and the factors affecting these costs (cost drivers), there are few, if any, well accepted models for software cost estimating. The lack of good historical data on development costs, software change rates, support costs, and good metrics for software quality, reliability, and maintainability are stumbling blocks to effective software cost estimating.

Literature on software cost estimating reviewed during Phase I is listed as items 48 through 73 of the Bibliography.

COST MODELS

A large number of software cost prediction models or techniques are described or referenced in the technical literature reviewed. Twenty-one models were examined in some detail in an effort to identify commonalities and differences in the cost estimating approaches. Ten models are limited to software development cost, while eleven have software support cost as a primary or secondary output. Table 2 lists the models in alphabetical order.

Software development cost prediction models are generally more sophisticated than the techniques used to estimate software support costs. Support costs are usually computed as a function of either development costs or program size. Regardless of the degree of sophistication, the models do not compute cost estimates to the depth or detail necessary to evaluate the impact of alternative software designs or support concepts on software support costs. Most models concentrate only on manpower.

TABLE 2. SOFTWARE DEVELOPMENT AND SUPPORT COST PREDICTION MODELS

Model Nomenclature & User	Application	Techniques	Data Required	Comments
Aerospace Model, Air Force Avionics Laboratory/Aerospace Corporation	Manmonth estimates for design, development, operation and maintenance	Top-down based on number of instructions	Number of instructions	Cost equations for real-time and support programs; includes gross software maintenance calculations
Software Development Cost Method, Boeing	Manhour estimates for development	Top-down based on number of inputs and outputs	Number of inputs/outputs	Single equation: MH=.33I**2, where I= inputs + outputs
Software Military Support Cost Method, Boeing	Support personnel requirements estimation	Top-down based on number of instructions	Number of instructions	Single equation: NSP=I/10000 where I= number of instructions
Computer Resources Integrated Support Plan Model (CRISP), Boeing	Software life cycle costing	Bottom-up	Number of instructions changed, shifts per instruction, manhours per shift, manhours per month, manhours per line of code	Not suitable for predicting costs but good approach for tracking costs
C-14 Model, Boeing	Support costs estimates	Bottom-up	Number of instructions changed, manmonths, and costs per manmonth plus other support costs	Cost of Support = (cost/manmonth x manmonths required) + additional support costs
ESD Model, Electronic System Division	Listing of factors and hardware development cost	Top-down based on number of instructions	Number of instructions language, application, etc.	Subjective parameters input to algorithms reflect development technique, personnel skills, etc.
Concept Model Electronic Systems Division	Life cycle cost estimation	Top-down based on number of instructions	Number of instructions,	Developed for ESD by Doty Associates; good approach for software LCC estimation
F-16 Model, Air Force Avionics Laboratory	Manmonth estimates for operation and support	Top-down based on number instructions	Number of instructions,	Single equation: MM=.00712 x I where I = number of instructions
GRC Model, General Research Corporation	Manmonth estimates for development programs	Top-down based on number of instructions and language	Number of instructions, language, degre of difficulty, degree of host computer saturation	Single equation: T = 0.04N ** 0.79 x 0.54 **L where N=number of instructions and L=1 for high level language and zero for assembly language

TABLE 2. SOFTWARE DEVELOPMENT AND SUPPORT COST PREDICTION MODELS (continued)

Model Nomenclature & User	Application	Techniques	Data Required	Comments
GRC Model (Version A), General Research Corporation	Resource consumption estimates by program phase	Top-down based primarily on number of instructions	Number of instructions, language, degree of difficulty, degree of saturation of host computer	Error correction is considered only legitimate maintenance effort; modifications are defined as development efforts
GRC Model (Version B), General Research Corporation	Error correction cost estimation subsequent to delivery	Top-down based on number of instructions	Number of instructions	Single equation: cost = 1740 x number of instructions x (-.00111 x life cycle year + 0.01611)
Hughes Model, Hughes Aircraft Company	Research, design, test, and evaluation cost estimation	Top-down using cost estimation relationships	Software, host computer and development program descriptions	Extensive historical data used for development
Cost Estimation Model, IBM	Development cost estimation	Top-down using number of lines	Software description	Single equation: Manmonths = 5.2 L**0.91 where L = number of lines
Cost of Function Model, Naval Air Development Center	Estimation of total acquisition costs: research development, test and evaluation plus production	Bottom-up based on six variables (see data requirements)	Number of instructions, contractor travel, document types, programmer experience, number of independent consoles, percentage of new instructions	Ten modules are employed in this model
Manpower Model, Norden	Prediction of manpower utilization by program phase	Bottom-up based on three variables (see data requirements)	Program manpower requirements, program length, and peak manloading	Single equation: y = 2 Kate **(-at**2) where y = manpower by time period, K = total cumulative manpower by end of project, a = shape parameter, and t = elapsed time from start of cycle
Price Software Cost Model (PRICE-S), RCA	Cost estimation for design, implementation, and integration test	Top-down based on an extensive set of inputs; incorporates some bottom-up features.	Seven catagories: project magnitude, program application, level of design and code, resources, project difficulty, project specifications, reliability requirements, and utilization	New cost estimating relationships developed for each specific application. Proprietary model.
Putnam Model	Estimation of manyears, costs and schedules for software projects	Top-down based on five variables	Elapsed time, manpower on board, cumulative effort expended, difficulty, expected change in difficulty	Uses the Rayleigh/Norden equation to describe the software life cycle manpower curve; predicts cash flow and cumulative costs either for software development or the entire life cycle.

TABLE 2. SOFTWARE DEVELOPMENT AND SUPPORT COST PREDICTION MODELS (continued)

Model Nomenclature & User	Application	Techniques	Data Required	Comments
Software Life Cycle Model (SLIM)	Life cycle cost estimation and resource calanderization	Top-down using development inputs to compute support costs	Development costs	Parametric model based or Putnam's approach.
Software Development Cost Model, System Development Corporation	Predicts development costs	Top-down based on 12 variables	Interface complexity, percent clerical instructions, percent information storage/retrieval, frequency of operation, pages of documentation, business or non-business application, new or familiar computer special displays, random recess device, percent programmers participating in design, personnel continuity, and number of locations for program development	Cost relationships are the result of regression analysis on 94 variables for 105 programs
Tecolote Model, Tecolate Research, Inc.	Development cost estimation for tactical software used in a fire control system	Top-down using five equations	Manmonths, number of instructions, air threats or sea threats, computer system speed, and fast storage capacity	Computer speed and storage capacity are the major drivers
SPREAD Model, Wolverton/TRW	Estimation of manmonths and computer usage on a monthly basis during development	Top-down based on number of instructions and type of software	Cost history data base, number of instructions, degree of difficulty, and software category	Six categories of software are considered; basic assumption is that development cost is proportional to number of instructions; historical data base consists of cost per instruction data

Currently, only two software cost models are available to public users on a widespread basis: the RCA PRICE S2 model and the SLIM model based on L.H. Putnam's approach. SLIM computes total life cycle cost (development and support) while PRICE S2 computes only development cost (and time). Both are parametric models and have had a large impact in the area of software cost estimating. Putnam's approach is significant in that he postulates that total required manpower, development time, and system difficulty are fundamental parameters of any software project. Furthermore, Putnam's approach has the desirable feature of indicating when estimating parameters fall outside of practically achievable regions. PRICE S2 (and recently S3) is one of the RCA family of PRICE models. Numerous government organizations are using or planning to use PRICE S2 to estimate software development costs.

COST DRIVERS AS PERCEIVED BY MODELLERS

Many authors have noted that software development costs are consistently underestimated, while overestimates are rare. This historical tendency toward underestimation may indicate that not all key drivers of software cost have been identified or included in current software cost prediction techniques or models.

Examination of software cost models served to identify some of the factors contributing to software cost. Program size and complexity are probably the two most significant factors contributing to software development cost, since these parameters are considered in all of the cost models. Other major factors include the stability of software design requirements and whether the hardware design is pretty well frozen, availability of the development computer to the programmers, computing speed requirements, degree of fill of the operational computer main memory, applicability of previously developed software, competence/productivity of the programmers, and the development schedule requirements. Table 3 lists the major factors considered by the models reviewed. Those factors are grouped into six categories:

* Requirements variables address the system and software requirements.

* Design and Coding variables describe the size and functions of the programs developed to meet the requirements.

* Programming Environment variables describe the kinds and productivity of programmers and the hardware and software support they have.

* Management Environment variables consider management influences on the programming process.

TABLE 3. COST MODEL FACTORS

REQUIREMENTS

Program type/application
Timing requirements
On-line program
Requirements change/design stability
Response time
Security classification
Vagueness/lack of knowledge of requirements
Innovation required
Design carryover
System interface complexity

DESIGN AND CODING

Number of object instructions delivered
Program complexity
Language
Source instructions written
Number of functions
Types of functions (mix)
Number of subprograms (modules)
Number instructions/module
Number object instructions not delivered
Percent object instructions reused
Percent source instructions in POD
Types of instructions (mix)
Number of words in data base
Number classes in data base
Number of input variables
Number of output variables
First program on computer

INSTALLATION, OPERATION AND MAINTENANCE

Number of user centers
Frequency of operation

PROGRAMMING ENVIRONMENT

Programmer experience
  Language
  Application
Programmer participation in design
Personnel continuity
Maximum number of programmers
Percent senior analysts
Percent senior programmers
Average programmer utilization
$/Manyear
Travel required
Programming philosophy
Closed/open shop availability
Development not at operational site
Program turnaround time
Use of automated validation/verification tools

MANAGEMENT ENVIRONMENT

Amount of external documentation
Schedule realism
Coupling - system/SW engineering
Orgn. resp. for SW management
Number of agencies concur/review
Customer inexperience
Total nr. document types
Validation/verification responsibility

HARDWARE CONSTRAINTS

Core capacity
Concurrent development
Number of bits/word
Machine speed
Computing cost
Special display equipment
Random access device
Input/output capacity

* Hardware Constraints take into account the effect of target computer limitations on programming difficulty (e.g., memory size).

* Installation, Operation and Maintenance variables describe the impact of operations and support on development programming. (In the models reviewed, this area related to large ground-based installations.)

Within each category the top two to four factors (i.e., those included by the most models) appear at the top of the list. Table 4 lists the major factors accounted for by PRICE-S2, probably the most sophisticated software development cost model in wide use today.

VALIDITY OF COST MODELS

Recent studies of development cost estimating methodology have concluded that "cost estimation methods must rely on the estimators' prior experience or rules of thumb derived from historical data which are inappropriate or inaccurate."[1] An Air Force study,[2] for example, took eight cost estimating models and applied them to six different computer programs. The results of that study are presented in Table 5. Dollar values are normalized to 1.0. Analysis of the results in Table 5 reveals not only a wide variation among cost model estimates on the same program (as much as thirty to one), but the fact that the biases are not consistent among the models. Even though model #8 is always the high estimator, its degree of highness is variable Model #3, which tracked the mean for cases A-D, estimated very low on cases E and F. Note also the actual costs on cases E and F, and the estimated actual costs on cases C and D. Neither of those sets of numbers tends to give any great degree of confidence in the outputs of the models. Of course, a major cause of variation is probably the difference in data base size and attributes used to develop the cost estimating equations of the various models.

---

1. J. A. Clapp, "A Review of Software Cost Estimation Methods," Report Nr. ESD-TR-76-271, Electronic Systems Division, AFSC Hanscom AFB, Bedford, Mass. 01731

2. T. G. James, "Software Cost Estimating Methodology," Report Nr. AFAL-TR-77-66, Air Force Avionics Laboratory, Wright-Patterson AFB, Ohio 45433

TABLE 4. COST FACTORS ACCOUNTED FOR BY PRICE-S2

Project size	Project familiarity
Project type (e.g., MIS, radar, telemetry, etc.)	Intensity of effort
Operational/customer environment	Changing requirements
Hardware constraints (system loading)	Programming language
Existing design	Compiler power and efficiency
Existing code	Development in-house or on-site
External interfaces (type and quantity)	Project complexity
Hierarchical design/functional flow structure	Engineering requirements
Number of functions performed	Programming requirements
Amount of code per function	Configuration control
Schedule constraints, lead times and overlaps	Documentation
ECN effects	Program management
Economic trends	Design phase requirements
Technology growth	Implementation
Fee, profit, and G&A	Test and integration
Computer operation costs	Integration of independent projects
Overhead	Verification and validation
Organizational efficiency	Multiple test beds/installations
Skills	Government furnished software
	Purchased software (e.g., subcontracts)
	Design-to-cost
	Resource allocation with respect to time

TABLE 5. RESULTS OF COST MODEL STUDY

Model	Program					
	A	B	C	D	E	F
1	.35	1.05	1.12	1.09	**	**
2	.75	1.05	1.15	1.12	**	**
3	1.06	.96	1.00	.99	.15	.58
4	.70	.66	.50	.51	.60	.57
5	**	**	**	**	.34	.54
6	1.49	1.25	1.38	1.35	.17	.24
7	.27	.24	.25	.26	.35	.34
8	1.89	1.79	1.59	1.68	4.39	3.74
Average cost estimate, all eight models	1.0	1.0	1.0	1.0	1.0	1.0
Actual Cost	-	-	6.09*	5.95*	.12	.34

* Contractor Estimate

** Insufficient Data

A recent Hughes in-house study investigated the feasibility of forecasting software support costs.[1] Part of the study effort involved comparing the manpower expenditure actuals for development of three Hughes embedded software systems with manpower expenditure forecasts made using one of the equations developed by Lawrence Putnam.[2] Putnam's forecast, if extended into the period of software operational use, would be a basis for prediction of support costs. However, as Figure 3 portrays, the actuals don't match the Putnam forecasts very well. Indeed, actuals didn't match Putnam forecasts for any of three systems studied.

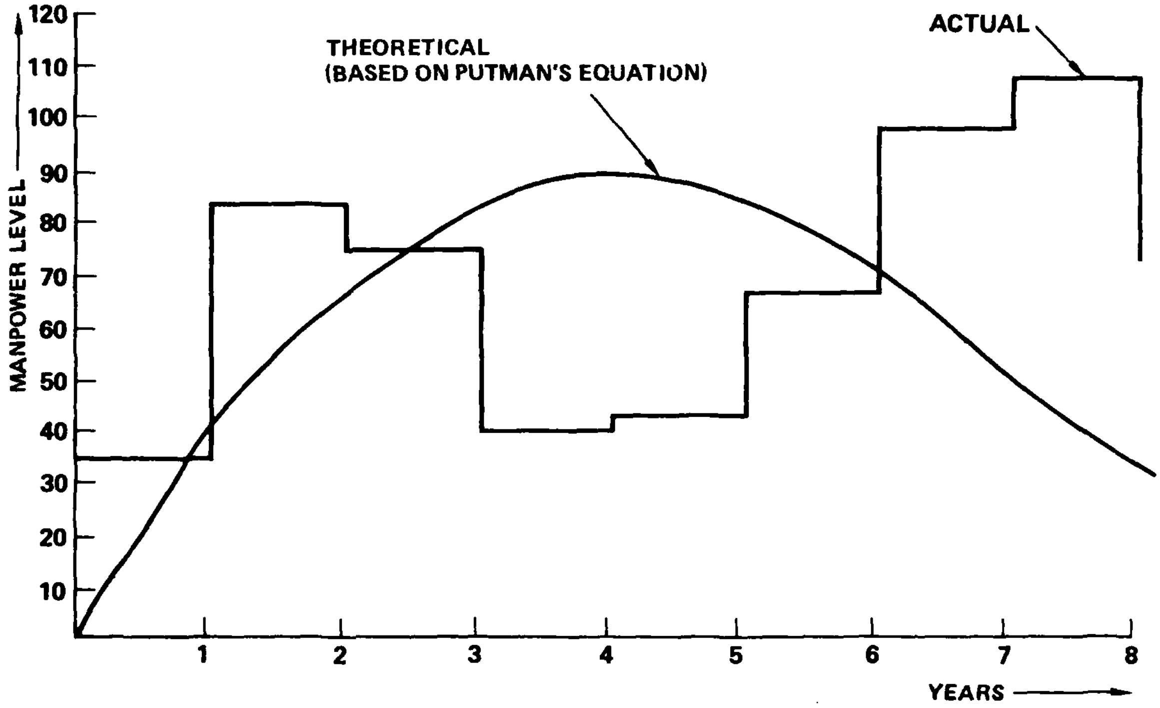


Figure 3. Actual Versus Predicted Manning Levels

1. R. W. Highland, "Exploratory Study of Software Support Costs," IDC Ref 281742/111, Hughes Aircraft Company, 24 July 1979

2. L. H. Putnam, "A General Empirical Solution to the Macro Software Sizing and Estimating Problem," IEEE Transactions on Software Engineering , Vol. SE-4, July 1978

A possible reason offered for the mismatch of actuals and Putnam forecasts is that development of embedded software for military avionic systems is significantly different from development of non-embedded systems software of the type included in Putnam's sample. In particular, the technology constant Ck, which Putnam estimates at between 5000 and 10000, has to be approximately 1000 in order to satisfy his equation relating program size, effort in manyears, and development time. Ck is a measure of productivity, and hence difficulty of developing software.

CONCLUSIONS

The review of current approaches in software cost estimating revealed that none of the models examined adequately fulfills the AFWAL/AA requirement for estimating avionics embedded software support costs. Other specific conclusions are provided below.

Software life cycle cost estimating is hampered by a lack of good historical data on development costs, change rates and support costs, and good metrics for software quality, reliability and maintainability.

There are few (if any) well-validated, well-accepted models for software development and/or support cost estimating.

Software development cost prediction models are more sophisticated than software operating and support cost models.

Software program size (or the number of instructions) forms the basis for most software development and software support cost models. Identification of other software program characteristics and development of additional estimating relationships are needed in the software support cost prediction model development effort.

Computing techniques and analysis approaches used in software development cost prediction models may be useful in software support cost prediction models.

Most modellers perceive manpower as the key driver in software costs; hence, almost all models include only manpower predictions. They universally neglect the cost of acquiring the other resources required in the support process (e.g., hardware, support software, documentation, etc).

V. FIELD SURVEY FINDINGS

The purpose of the field surveys was to develop an overview of the Air Force avionics embedded software population and an understanding of how that population is supported by AFLC.

General and specific information on the AFLC support process was obtained by reviewing pertinent Air Force and Air Logistics Center (ALC) regulations, procedures and operating instructions, and by interviewing key software managers and engineers at the ALCs. Sample data were collected on six packages in order to determine the general availability of the kinds of data needed to support a model development effort.

This section describes the basic findings of the field surveys and provides a compilation of the essential information necessary to understand avionics software support in the Air Force. Major subsections are:

* Software population
* Software sample
* The software life cycle
* The AFLC resource-level planning process
* The AFLC software support process
* General observations

SOFTWARE POPULATION

OC-ALC estimates that there are now over 50,000 separate embedded computer system programs within the Air Force, plus a similar amount of program documentation. This is increasing at an estimated rate of 6400 packages per year. A central inventory system for this software has been established at OC-ALC under the auspices of the Technical Order Section of the Operations and Support Branch (OC-ALC/MMEDU). That system includes the Computer Program Identification Number (CPIN) system and the Air Force Computer Resources Inventory (AFCRI).

Air Force Inventory Systems

The CPIN system is currently a semi-automated system (local word processing) planned to be on-line at Tinker AFB in 1981. Compendia of the data are presently issued monthly. Appendix A of this volume provides a brief description of the CPIN system.

The AFCRI, which utilizes the same basic data as the CPIN system, is on-line on the ASD Information Center computer at WPAFB. It is accessed via a remote terminal at OC-ALC. That data base contains approximately 8,900 items as of January 1980. The majority of those are ATE programs, of which most are unit under test (UUT) programs. Approximately 600-700 items are currently being added to the system each month. That rate is expected to increase in the future.

Data for both the CPIN and AFCRI systems are supplied to OC-ALC/MMEDU by the various system/item managers or responsible activities utilizing AFLC forms 505 and 506 (see Appendix A). This is a major on-going, low-to-medium priority task. OC-ALC estimates it will take about 5-10 years to completely inventory all embedded computer system software for which AFLC is responsible.

PSCM Study Software Inventory

The PSCM study team compiled an inventory of software systems in the categories of Operational Flight Programs (OFPs) and Electronic Warfare (EW) software by interviewing cognizant personnel at the various ALCs. The results of that inventory are displayed in Tables 6 and 7. Table 8 is a list of ATE system software controlled by SA-ALC, the system managers for ATE. SA-ALC estimates that of the 400-500 identifiable ATE systems, about 20-30 are particularly active with regard to software.

SOFTWARE SAMPLE

In the early part of the study a sample of software packages was selected on which to gather data. The sample consisted of five systems having OFPs (A-7D, F-15, F-16, F-111F and FB-111A) plus the F/FB-111 support software. Some data on change history and manhours were also obtained for the F-111D at SM-ALC.

The F/FB-111 support software includes simulation programs used for analysis, program development and verification/ validation of F/FB-111 OFPs. Support software is generally considered as one of the resources supporting OFPs; in this instance it was treated separately to enhance the identification and quantification of the key factors impacting avionics embedded software support costs.

Brief descriptions of the five systems and the F/FB-111 support software package are provided in the following paragraphs. Table 9 presents the pertinent data in tabular form to facilitate comparison.

TABLE 6. OFP SOFTWARE SYSTEMS

ALC	SYSTEM
Warner-Robins	F-15 EAR GPS Pavetack/VATS C-130 JTIDS
Oklahoma City	A-7D/K AGM-69 (SRAM) ALCM GLCM B-52D B-52G/H E-3A E-4B EC-135 Inertial Navigation Systems and Star Trackers
Ogden	F-4 F-16 GBU-15
San Antonio	F-106 MA-1 C-5 F-5E
Sacramento	F-111D F-111F FB-111A

TABLE 7. EW SOFTWARE SYSTEMS, WR-ALC

ALR-46	ALQ-125	APR-38
ALR-56	ALQ-131	ESAS
ALR-62	ALQ-135	
ALR-69	ALQ-155	

TABLE 8. ATE SOFTWARE SYSTEMS, SA-ALC

Category	Systems
Electronics ATE	A-7 SE (Support Equipment) A-10 SE B-52 SE C-5A SE C-130 SE C-135 SE C-141 SE E-3A SE F-4 SE F-5 SE F-15 AIS MTTU F-15 SE F-16 SE F-101 SE F-105 SE F-106 SE F-111 SE HH-53 SE T-38 SE T-43 SE Engine SE General Purpose SE Minuteman SE Special Weapons SE
Funded Studies	F-16 Minuteman ATE E-3A
Process Control	Pacer Comet ATS JEA Stacker MIPVS F-100 Fuel Controls Vibration Diganostic System M37 Engine TE Automation
Test Software Development	F-15 SE Add-on F-16 Support Equipment MM4920 Modules 4920 DQ Modules 6625 Modules E-3 SE Modules F-100 EECC, EEC ATE and EHR Module

TABLE 9. SAMPLE SOFTWARE PACKAGE DESCRIPTIONS

SOFTWARE PACKAGE	APPLICATION	PROGRAM SIZE	LANGUAGE	COMPUTER	WORD SIZE	MEMORY SIZE	MEMORY FULL	YEAR INITIALLY DEVELOPED	DEVELOPER
A-7D OFP	Navigation Weapons Delivery	15K	Assembly	IBM TC-2	16-Bit	16K	89%	1968	Vought
F-15 OFPs:									
F-15 Central Computer OFP	Fire Control/ Weapons Delivery	11K	Assembly	IBM AP-1	32-Bit	16K	70%	1970	McDonnell
F-15 Radar Data Processor OFP	Weapons Delivery	24K	Assembly	HCM-231	24-Bit	24K	100%	1972	Hughes
F-16 OFPs:									
F-16 Fire Control Computer OFP	Navigation/Weapon Delivery	26K	Jovial (85%) Assembly (15%)	MAGIC 362 F-2	32-Bit Instructions 48-Bit Data	32K (16-Bit) + ROM	80%	1975	General Dynamics
F-16 Inertial Navigation Set OFP	Navigation	9K	Assembly	SKC-3000	15-Bit Instructions 19-Bit Data	32K	28%	1976	Singer Kearfott
F-16 Head-Up Display OFP	Display Functions	7K (Blk 1) 11K (Blk 2)	Assembly	Marconi	16-Bit	8K (blk 1) 16K (blk 2)	88% (blk 1) 70% (blk 2)	1976	Marconi Elliot
F-16 Fire Control Radar	Navigation/ Weapon Delivery	33K	Assembly	Westing-house	16-Bit	32K* EPROM +4K RAM	100%	1976	Westing-house
F-16 Stores Management OFP	Weapons Control	34K	Assembly	GD 8080	8-Bit	76K	94%	1978	General Dynamics
F-111F OFPs	Navigation/ Weapon Delivery	2X 16 K	Assembly	IBM CP-2	16-Bit	2X 16K	99%	1968	Autonetics
FB-111A OFPs	Navigation/ Weapon Delivery	2X 16K	Assembly	IBM CP-2	16-Bit	2X 16K	99%	1968	Autonetics
F/FB-111 Support Software	System Simulation	300K+	Fortran/ Assembly	Harris/4	24-Bit	480 K	Virtual	1968	General Dynamics

*Planned expansion to 40K EPROM + 16K RAM

A-7D OFP

The A-7D OFP was developed by IBM/Vought in 1968 for use in the IBM TC-2 computer which controls navigation and weapons delivery functions in the A-7D. The TC-2 computer is a 16-bit machine with 16K of memory of which 15K or 89 percent is presently required for the OFP. The OFP is programmed in assembly language and is considered to be of average complexity relative to other avionics software.

F-15 OFPs

The F-15 weapons system includes four programmable devices, of which the central computer and the radar data processor utilize OFPs. The other two programmable devices, the radar warning receiver and the internal countermeasures set, are part of the electronic warfare system and utilize EW software.

The central computer OFP was developed by McDonnell Aircraft in 1970 for use with the IBM AP-1 computer used for mission oriented calculations. The AP-1 is a 32-bit word size machine with a 16K memory, of which approximately 70 percent is currently required for the OFP. The OFP is written in assembly language and consists of eight modules. The modular structure of the program allows considerable flexibility in accomplishing program changes or adding additional functions.

The radar data processor OFP was developed by Hughes in 1972 for use with the HCM-231 computer. The HCM-231 accomplishes radar signal processing and control and provides the interface with other avionics equipment. The computer is a 24-bit word size machine with 24K of memory which is currently 100 percent filled by the OFP. The existing 24K memory is scheduled for replacement by a 96K memory device beginning in 1980. The OFP is written in assembly language and is considered to be very complex relative to other avionics software.

F-16 OFPs

The F-16 weapons system includes seven computer controlled subsystems, five of which currently utilize software OFPs: fire control computer, stores management system, fire control radar, inertial navigation system and head-up display. Two subsystems (central air data computer and radar/electro-optical display) use programs loaded in Read Only Memories (ROMs). All seven programs are controlled as Computer Program Configuration Items (CPCIs). Taken as a whole, the F-16 OFP software package includes 1,200,000+ lines of code programmed in higher order, assembly, and machine languages.

The fire control computer OFP was developed in 1975-1976 by General Dynamics. The OFP is used with the MAGIC 362 F-2 computer manufactured by Delco Electronics. The fire control computer OFP functions include air-air/air-ground fire control, data display, stores selection, navigation, mission planning and fixtaking. The machine utilizes 16 and 32-bit instructions, 16 and 32-bit fixed decimal point data with 24 and 48-bit floating decimal point data. Memory includes 32K of 16-bit storage plus 1K of 40-bit ROM. The OFP currently requires 26K (80 percent) of the available memory. The OFP is programmed using both MAGIC F-2 assembly language (15 percent) and JOVIAL J3B-2 HOL (85 percent). Complexity of the program is considered to be average relative to other avionics software.

The inertial navigation set OFP was developed by Singer Kearfott Division with an initial release date in 1976. The OFP provides navigation calculations for the navigation panel display and provides back-up multiplex bus control for the fire control computer. The OFP is used with the Singer Kearcoft SKC-3000 computer which provides 32K of memory that is presently 81 percent filled by the OFP. The SKC-3000 utilizes 15-bit instructions and 19-bit data. The OFP is programmed in SKC-3000 assembly language and is considered to be of low to medium complexity.

The head-up display OFP was developed by Marconi Elliot Avionic Systems in 1976. This software was originally purchased as a hardware configuration item and loaded in a ROM. The ROM was replaced by an Erasable Programmable Read Only Memory (EPROM) however, and the head-up display software is now reprogrammable and thus classified as an OFP. Functions include control/generation of displays for snap-shoot missile launch, and flight direction. The OFP is used with a Marconi/General Dynamics 16VE017003 computer providing 16K of 16-bit memory which is 70 percent filled. The OFP is programmed in assembly language and has not been rated in complexity due to the lack of experience resulting from its low change rate.

The fire control radar OFP was developed by Westinghouse Electric Corporation in 1976 for use with the Westinghouse radar processor unit. Functions include air-air and air-ground target tracking, ground mapping, inertial navigation coordinate updating, and video processing. The OFP is presently loaded in a 32K EPROM Random Access Memory (RAM) using 16-bit word size. The OFP, however, presently fills 100 percent of the available memory, so expansion to a 40K EPROM is planned. The OFP is programmed in assembly language and is considered to be very complex relative to other avionics software due to its limited modularity.

The stores management system OFP was developed by General Dynamics in 1978 for use with a 8080 computer. The stores management OFP monitors weapons status and controls/releases weapons. The OFP currently consists of 34,816 words which occupy 94 percent of the computer's 36K 8-bit word memory. The OFP is programmed in assembly language and is considered to be of high complexity.

FB-111A and F-111F OFPs

The FB-111A and F-111F OFPs were developed by Autonetics in 1968 for use in the IBM CP-2 computers which handle navigation and weapons delivery in the FB-111A and F-111F aircraft. The CP-2 is a 16-bit word size machine with 16K of memory. Memory fill in each case is 99 percent. The 32K word OFPs are programmed in assembly language and are considered to be of high complexity relative to other avionics software.

F/FB-111 Support Software

The F/FB-111 support software package was developed by General Dynamics in 1974 for use with the Harris/4 computer used for simulation of F/FB-111 operational environments. The Harris/4 computer is a 24-bit word machine with 480K of memory, of which 300K is required for the source lines and data files included in the support software package. The support software consists of 75 percent Fortran code and 25 percent machine language code and is considered to be of high complexity.

THE SOFTWARE LIFE CYCLE

A basic understanding of the software life cycle will enhance understanding of software support. This is so because software support essentially consists of a series of mini-development cycles. Each change undergoes virtually the same process as the original software underwent when it was developed.

This section begins with a comparison of hardware and software characteristics. The computer program life cycle as defined in AFR 800-14 is then described. The Computer Resources Integrated Support Plan (CRISP) is briefly outlined. Finally, causes of software changes are discussed.

Comparison of Hardware and Software

The software life cycle is similar to the hardware life cycle, except that the manufacturing process is greatly simplified, and "maintenance"* is really a modification process. Hardware goes through an engineering development and design

* Throughout this report "maintenance" will be used to refer only to those changes that do not alter the functional specifications (input/output) of the software (i.e., error corrections or minor efficiency changes). "Support" is the more general term applied to the total change process.

phase; software has a similar development cycle, beginning with an analysis of software requirements. Hardware is fabricated; this can be either the final product, in the case of a one-of-a-kind system, or it can be a preproduction prototype. Software coding is similar to hardware fabrication. The software listing is analogous to the hardware engineering drawing, except that it is "as-built" instead of "build-to." That listing goes through numerous iterations as the code is debugged.

In the case of hardware, a major portion of the acquisition effort normally goes into the manufacturing cycle. In the case of software, manufacturing (i.e., making more than one copy) is a completely automated process of taking the master and copying it. Hardware quality assurance/quality control is concerned with ensuring that many units conform to the design specifications. Software quality assurance/quality control focuses on the quality implications of software engineering practices, since a single master program is the main product.

Hardware reliability is a function of the fact that components physically degrade or fail. Software, however, never fails or physically degrades (although the physical medium in which it resides may do so). Software unreliability is caused by inherent logic errors which were not detected and eliminated during development or verification. It is difficult to detect such errors because of the complex logical relationships and the vast number of distinct internal states which exist in computer programs. No reasonable (i.e., affordable) amount of testing can completely check out any but the simplest programs (although certainly a large number of critical paths can be tested - you can have as much confidence as you are willing to pay for).

Hardware maintenance (either repair or preventive) consists of returning the hardware to its original state by either replacing failed components or adjusting the mechanism (or both). Software maintenance involves modifying the program, changing its original state by removing the logic errors (hopefully without introducing additional errors as a result of the modification).

Hardware undergoes engineering modifications to fix design faults (as does software) or to attain new capabilities. A significant part of software modifications is upgrading in response to new operational requirements. A major attraction of software is the relative ease with which new capabilities can be implemented, as compared to hardware retrofits. It is therefore important that software be designed with future modifications in mind.

In summary, software is similar to hardware with the exception that it never physically degrades because it is not physical. The abstract nature of software, coupled with the logical complexity of the structures embodied, causes it to have a different failure mechanism. This in turn causes a different kind of maintenance process. Modifications of both hardware and software follow a similar path. A number of software changes are often grouped into a single "block change" in order to simplify configuration management.

The Computer Program Life Cycle

The computer program life cycle, as defined in AFR 800-14, Volume II, "Acquisition and Support Procedures for Computer Resources in Systems," is diagrammed in Figure 4. The phases do not necessarily coincide with any particular hardware phase, but occur in relation to the requirement to develop particular Computer Program Configuration Items (CPCIs). The phases are defined as follows:

Analysis Phase . The purpose of the analysis phase is to define the functional performance requirements for a computer program. These requirements describe the functions the CPCI is required to accomplish as part of the system. Additionally, the functional interfaces and the necessary design constraints are defined. This phase normally begins with the release of the system specifications, and terminates with the successful accomplishment of the Preliminary Design Review (PDR). During this phase, various design approaches are considered, analyses and trade-off studies are performed and design approaches selected. The authenticated development specification forms the baseline from which the design phase initiates.

Design Phase . The purpose of the design phase is to develop a design approach including mathematical models, functional flow charts, and detail flow charts. The design approach should also define the relationship between the computer program components. The detail flow charts define information processing in terms of the logical flow and operations to be performed by the set of computer instructions. This information is contained in the preliminary product specification and is normally presented and reviewed during the Critical Design Review (CDR). The design approach is documented in a preliminary Computer Program Product Specification and reviewed against the requirements of the development specification prior to initiating the coding phase.

Coding and Checkout Phase . The coding and checkout phase normally follows the CDR. The purpose of coding is to translate the flow charts into computer programs and data. The purpose of checkout is to convert the initial computer program code and data into an operational computer program. The determination that a computer program is operational is based upon checking that it produces correct outputs when operating upon predefined inputs. This first check is usually limited with each computer program and, upon successful completion, leads into the test and integration phase.

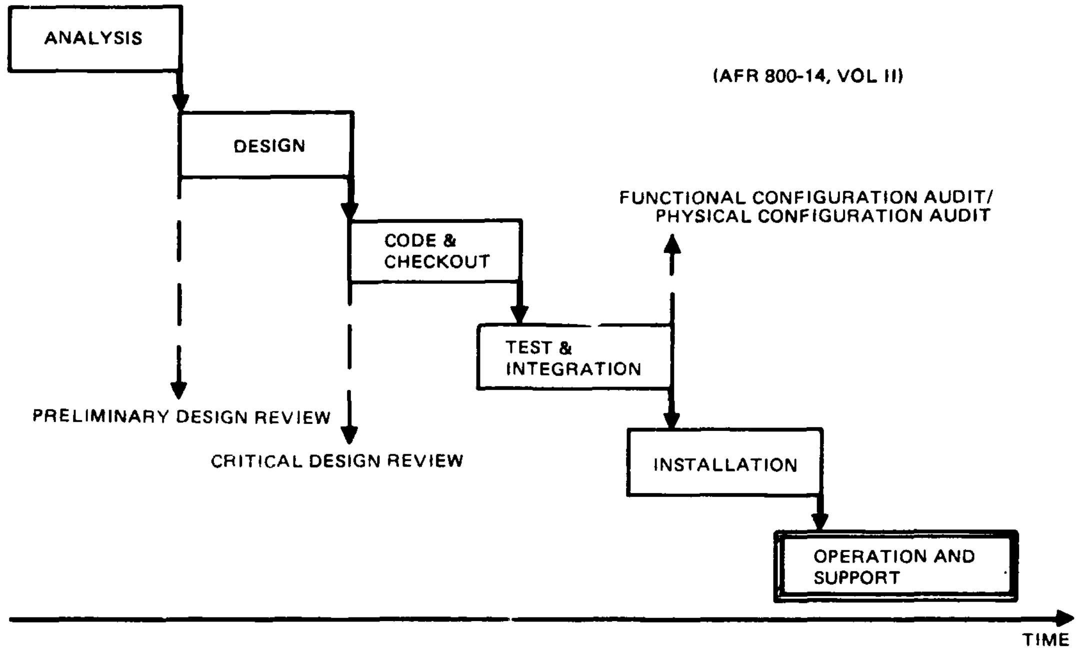


Figure 4. Computer Program Life Cycle

Test and Integration Phase. The purpose of the test and integration phase is to test the computer program against the requirements specified in the computer program development specification. This test and integration process includes the individual computer program function or module test and extends through total computer program formal qualification tests. Integration of the computer program with the total system is also accomplished and tested during this phase.

Installation Phase. The installation phase includes the loading and running of computer programs which have been successfully qualified and integrated. It may include peculiar adaptation to various sites for multi-site systems. It includes checkout to establish that the system operates with a required or specified level of confidence in support of the total system within the operational environment.

Operation and Support Phase. During the operation and support phase, the operational suitability of the system is assessed. Also, the capability of the computer program to operate on the total set of input data presented in an operational environment is evaluated. The support of a computer program includes all resources and activities required to insure that the computer program continues to meet the required operational capability. These activities may include responding to changes by modification of existing computer programs and the creation of new computer programs. Changes, not only to the computer programs themselves, but also to the associated documentation, are addressed. Incorporation of new programs or program modifications to an existing system normally requires reaccomplishment of all the phases in the computer program life cycle. Hence, the computer program life cycle is a continuing process throughout the system life cycle.

Computer Resources Integrated Support Plan

The plan for logistics support of hardware during the operation and support phase is documented in an Integrated Logistic Support Plan (ILSP). There is a similar document for embedded computer systems and software called a Computer Resources Integrated Support Plan (CRISP).

The CRISP identifies organizational relationships and responsibilities for the management and technical support of computer resources. It functions during the full-scale development phase to identify computer resources necessary to support computer programs after transfer of program manangement responsibility and system turnover. The CRISP continues to function after the transfer of program management responsibility and system turnover as the basic agreement between the supporting and using commands for management and support of computer resources. The following items are included, as applicable:

a. Offices of primary responsibility and management focal points for support of computer resources and the channels of communication among organizations.

b. Planning for configuration management of computer programs, including the assignment of configuration control responsibilites during the deployment phase. This planning will reflect the operational and support concepts for the system.

c. Responsibilities for composite system integrity, which include:

    (1) Computer storage utilization.
    (2) Computer program operating time and prioritites.
    (3) Computer program interface techniques.
    (4) Computer program baseline integrity.
    (5) Utilization of computer modules and peripherals.

d. Documentation required to support each type of computer program.

e. Responsibility for funding, scheduling, and system integration.

f. Personnel required for supporting computer equipment and computer programs together with training requirements.

g. Computer equipment and devices required to facilitate computer program changes, including acquisition responsibilities.

h. Computer programs required to support computer equipment and other computer programs, including acquisition responsibilities.

i. Verification and validation (V & V) of computer programs.

j. Plans to establish and operate necessary support facilities. Common and existing facilities will be used whenever practicable. The size and scope of the support facility will be based on workload predictions.

k. Provisions for the transfer of program management responsibility.

l. Provisions for system/equipment turnover.

Causes of Software Modifications

Changes to avionics software systems can be categorized into five different types:

* Corrections of coding errors and design deficiencies
* Optimizations of the computer code to save memory space or execution time
* Enhancements to existing capabilities
* Additions to existing capabilities
* Deletions of existing capabilities

The presence of errors in a highly complex real-time control system (which is what most avionics systems are) is virtually a foregone conclusion. It is almost impossible to completely prevent their occurrence and almost as difficult to detect them all during development and test. However, under the appropriate circumstances and operational environment, an error will manifest itself as a failure of the system to perform properly. One instance was cited during the field surveys of an F-111 whose indicated position would sometimes jump 8000 miles during flight. The problem was finally traced to two specific instructions in a 1200-instruction block of code; if they were executed while the computer was shifting from foreground to background mode certain counters would be cleared, which then resulted in the erroneous position indication. Once the cause of the problem was isolated the solution was simple.

In addition to error-corrections, simple changes are sometimes made to enhance the efficiency of program execution. Both the above classes of changes are considered program maintenance.

Another factor leading to software change is changes in the hardware. Modifications of the hardware can be either in response to design deficiencies discovered during development and early production, or for improved performance later on. Very often such changes require corresponding changes in the software.

The systems discussed in this report are built to respond to a military threat. A change in that threat environment can lead to a need to change the system; or new methods of dealing with the threat may be developed. Such changes can possibly be implemented by changing the software. The change might be an enhancement of an existing capability, or the addition of an entire new capability such as a new missile. In either case, software changes will almost certainly be required.

THE AFLC RESOURCE-LEVEL PLANNING PROCESS

In the short run (i.e., less than a year or two) the quantity of resources (especially manpower and hardware) available to support software changes at any given ALC may be regarded as fixed. This then leads to the situation where software changes will be made up to the maximum level permitted by the available resources, rather than resources being made available to support the required level of changes. (This can be less of a problem when an ALC is supporting multiple systems which utilize similar support resources.) This being the case, it is desirable to have some insight into that original decision-making process.

The F-16 was chosen by AFLC as the first candidate for processing under their Generic Logistics Decision Tree approach. As such it serves as an example of that decision process. The following kinds of questions were asked by the Air Force as part of that approach in making support resource decisions for the F-16 avionics embedded computer systems (ECS).

Are Governmental Functions Included? In order for AFLC to perform the inherent management functions associated with weapon system control a core of in-house engineering expertise, along with certain essential support resources, is required. As a minimum, a sufficient resource base must exist to permit the system manager to make value judgements in evaluating change requests, reviewing contractor proposals and accepting non-organically developed changes or modifications. Since non-avionic changes may have an impact on avionics software, these capabilities must be viewed at the system engineering or integrated level. These resources are needed to provide technical direction, perform technical analyses, evaluate system effectiveness, and manage the technology base.

Can the Functions be Segregated? In the case of the F-16, the resources required to perform the inherent management (governmental) function can be quantified and segregated from those required to provide direct operational support.

Is an Organic Nucleus Required? The organic nucleus required to fulfill the inherent governmental functions involves an avionics equipment bay staffed by digital systems engineers, mathematicians and other professionals. Also a flight test area staffed by instrumentation engineers having access to an instrumented flight test aircraft will be required.

Are AFLC Readiness Functions Included? Certain readiness functions are inherent to combat weapon system software support. The ability to modify a program in response to a changing threat environment may spell the difference between mission failure and success. For the F-16, three digital avionics subsystems contain logic which is critical to mission accomplishment. Digital avionics experience dictates that this logic will change in response to changing mission roles, changes in the operational environment, enhancements to system capabilities and operationally discovered deficiencies. Statistically, the peace-time change rate approximates 5% of the current program size. It is known that there will be a surge in this change rate during engagement in a wartime scenario; this surge, however, cannot be quantified before the fact. Of major concern for the F-16 is the ECCM logic to be placed in the fire control radar, the weapons delivery algorithms in the fire control computer and the ballistics equations in the stores management system.

Is Total Organic Accomplishment Required? Direct engineering support for the fire control radar, fire control computer and stores management system can be provided as a separate capability by procuring the necessary documentation for the three subsystems, providing individual dynamic test stations, and augmenting the established personnel baseline.

Is It Necessary to Increase the Organic Nucleus? Organic support of the fire control radar fire control computer and stores management system is estimated to necessitate more than 16 new support personnel and additional equipment costing more than $7M.

Are There Organic Obstacles? The fire control radar is under a Reliability Improvement Warranty (RIW) which would be voided by organic software support. Additionally, there are studies underway to define how to improve its ECCM capabilities. Because of the RIW and the unstable baseline, establishment of organic support is not recommended initially.

Is Interim Contractor Support Possible? The least risk alternative is to program for interim contractor support for the fire control radar while continuing to define organic requirements.

Are Contract Sources Available? Because the F-16 is in production, contract sources are available for the four subsystems not yet set aside: the inertial navigation system, the central air data computer, the head-up-display and the radar/electro-optical display. There is no indication that these sources will become unavailable in the foreseeable future. Contract sources could be developed to provide scientific computer support; however, this appears not to be cost-effective.

Is a Cost Study Required? For the inertial navigation system, the central air data computer, the head-up-display and the radar electro-optical display, AFLC does not presently possess the skills necessary to accomplish in-house engineering support. Additionally, a lack of source data for the central air data computer, head-up-display and radar/electro-optical display renders in-house engineering support for these systems impractical. In light of the above considerations, a formal cost analysis is not deemed necessary. For these four avionics subsystems, OFP support will be provided contractually.

THE AFLC SOFTWARE SUPPORT PROCESS

Within the Air Force, software is basically supported either in-house or by contractors. In some cases the level of in-house support is augmented by on-site contractors. This study focussed only on support which occurs at the five Air Logistics Centers (ALCs) and is performed by either Government personnel (Civil Service and Air Force) or on-site contractors, or both.

This section describes the support process both generally and at the specific ALCs. The analysis of those factors relating to the cost of that support is discussed in Section VI.

At each ALC the basic responsibility for support of embedded computer resources falls within the Engineering Division of the Material Management Directorate. The Computer Resources Branch is designated MMEC. Specific details of organizational implementation at each ALC are treated later in this section.

General Description of the Support Process

Avionics embedded computer system software support is essentially an ECP (Engineering Change Proposal) process. Each proposed software change is evaluated and processed as an individual entity. There are many similarities to software development cycle activities in the aspects of design, coding and verification/validation. However, a number of changes are normally processed together and released for operational use as a block update to the software. This is done in order to minimize configuration management problems. The primary tool for processing changes to OFPs is the Avionics Integration Support Facility (AISF). The same basic processes are applied to other categories of embedded software, such as EW and ATE. This subsection describes the AISF, the basic OFP change process, and the configuration management requirements.

Avionics Integration Support Facility: The AISF is a broad gauge engineering tool. It is used not only for software support, but also for hardware support, system support, engineering studies, training support and augmentation of system maintenance capabilities. Table 10 lists some of the functions in each of those AISF activity areas.

An AISF can be considered a collection of hardware and software assembled in a ground facility for the purpose of performing the activities required to implement changes in avionics computer programs, including integration testing of software and hardware changes. It consists basically of: 1) various functional simulations used for problem analysis and solution development; 2) a broad range of support software tools such as compilers, assemblers, verification/ validation tools, analysis programs and management systems; 3) a dynamic exerciser to more completely check out the characteristics of a proposed solution; and 4) an integrated test bed which serves to verify the adequacy of a total set of software and hardware changes. Aircraft flight tests are then used as the final "proof of the pudding."

A specific implementation of an AISF is shown in Figure 5 as an example. It includes the basic elements needed to support OFP software. The F-111 AISF consists of an avionics integration area, subsystem test area, OFP dynamic simulation area and computer support area. Instrumented flight test aircraft are also used in the support process. The integration, simulation and computer support areas are used extensively throughout the change process while the flight test capability is extensively used during the test and evaluation phase.

The integration area, which contains avionics integration test equipment (ITE), is used to integrate the OFPs with the avionics system. It further is used to recreate flight problems; check hardware/software interfaces; evaluate timing, stabilization and synchronization; and to conduct OFP/avionics system compatibility tests. On-line OFP change capability is available in this area, which enables efficient and expedient implementation of trial solutions.

TABLE 10. AISF SUPPORT OBJECTIVES

SOFTWARE SUPPORT

- Problem/Modification Analysis
- Modification Development
- OFP V&V
  - Development Test & Evaluation
  - Independant Validation and Verification

HARDWARE SUPPORT

- Analysis
- Modification
- Integration

SYSTEM SUPPORT

- Design
- Analysis
- Integration
- Documentation Validation & Modification

ENGINEERING STUDIES

- Effectiveness
- Reliability
- Maintainability
- Trade Offs

TRAINING SUPPORT

- Operational Procedures
- Avionics Subsystems Familiarization
- Maintenance Procedures

MAINTENANCE AUGMENTATION

- Failure Verification
- Test Procedure Development
- Operational Test Program Development

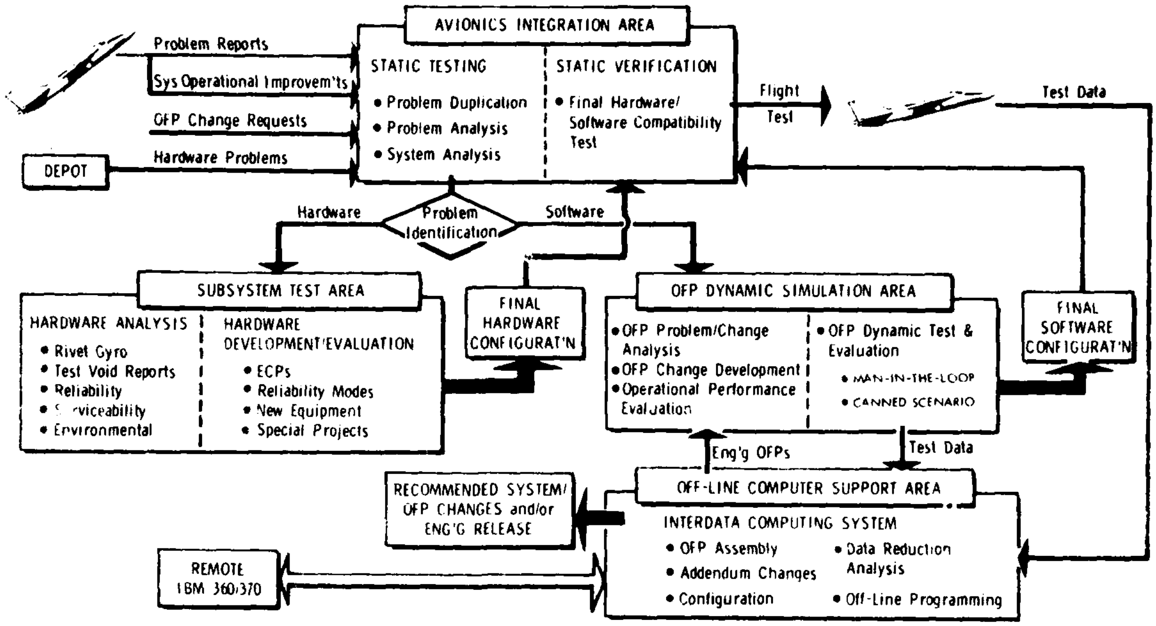


Figure 5. Avionics Integration Support Facility

The dynamic simulation area provides a capability to quantitatively analyze, develop, test and evaluate OFPs and OFP changes under realistic and repeatable conditions. The systems are hybrid simulators which retain the avionics computers with their resident OFPs and simulate the world as seen by these computers in actual flight. Visibility is gained into the inner-most parts of the OFPs through data monitoring and acquisition systems which provide for full real-time traces of OFP execution. Each simulation system is made up of three Harris Corporation 6024/VM mini-computer systems, an aircraft cockpit mock-up, special interface devices and a simulation software package.

The computer support area satisfies all computer support requirements associated with maintaining and updating OFPs. These requirements include reassembly; data reduction and analysis; documentation generation, maintenance and storage; maintenance of support software; specialized programs and programming; and automated configuration control. The computer support system includes two Interdata 8/32 mini-computer systems, a PDP 11/40 mini-computer system and a remote terminal to an IBM 360/65 complex.

The flight test capability includes aircraft equipped with special instrumentation packages designed specifically for monitoring and recording OFP flight performance. Flights are conducted to test overall OFP performance and mission suitability; analyze change and problem areas; test specific modes and functions; and to obtain engineering data to define and verify system performance.

The AISF technical staff consists of engineers, programmers and technicians. They encompass a spectrum of expertise on the aircraft system, avionics, computers, operational software, support software, bomb navigation, scientific programming, instrumentation, data reduction, systems analysis, configuration management, and equipment and software maintenance.

The mainstay in developing and integrating software changes is a GS-855 Electronics Engineer (Embedded Computer Systems), typically a grade 12. An electronics engineer is necessary because of the requirement to understand the hardware and the total avionics system as well as the software. A typical position description is given in Appendix B of this volume.

The OFP Change Process: The basic philosophy of avionics software changes is that of the block change cycle, as opposed to continuous changes. This means that as change requirements and problems are brought to the attention of the supporting facility they are collected for inclusion into the next block change cycle. The primary advantage of this approach is simplified configuration control of the complex software and systems involved. Specific changes of an urgent or emergency nature can be handled outside the normal block change schedule, but they are the rare exception.

Figure 6 details the flow of an OFP through the change process, while Figure 7 portrays a typical sequence and schedule. The phases are as follows.

The Feasibility Study Phase is conducted by engineering in accordance with user priority requests. It consists primarily of determining the update task for each change; scoping the resource requirements; investigating change impacts on other parts of the weapon system and support equipment; looking at computer memory and timing impacts; investigating integration problems; and determining if each change requirement is technically feasible and will actually provide the user with what is expected.

The results of the feasibility study are then presented at an OFP Block Change Definition meeting attended by the user, the system manager and software engineering. Based on the results of the feasibility study, an OFP Block Change Definition is established and agreed to. Constraints adhered to are: the block change contains only change candidates which do not impact hardware; the changes can be worked within existing resources; and the cycle time is maintained. Changes which do not meet these constraints are referred to the system manager for processing in accordance with hardware procedures. The main output of the feasibility study is the OFP Block Change Requirements Document.

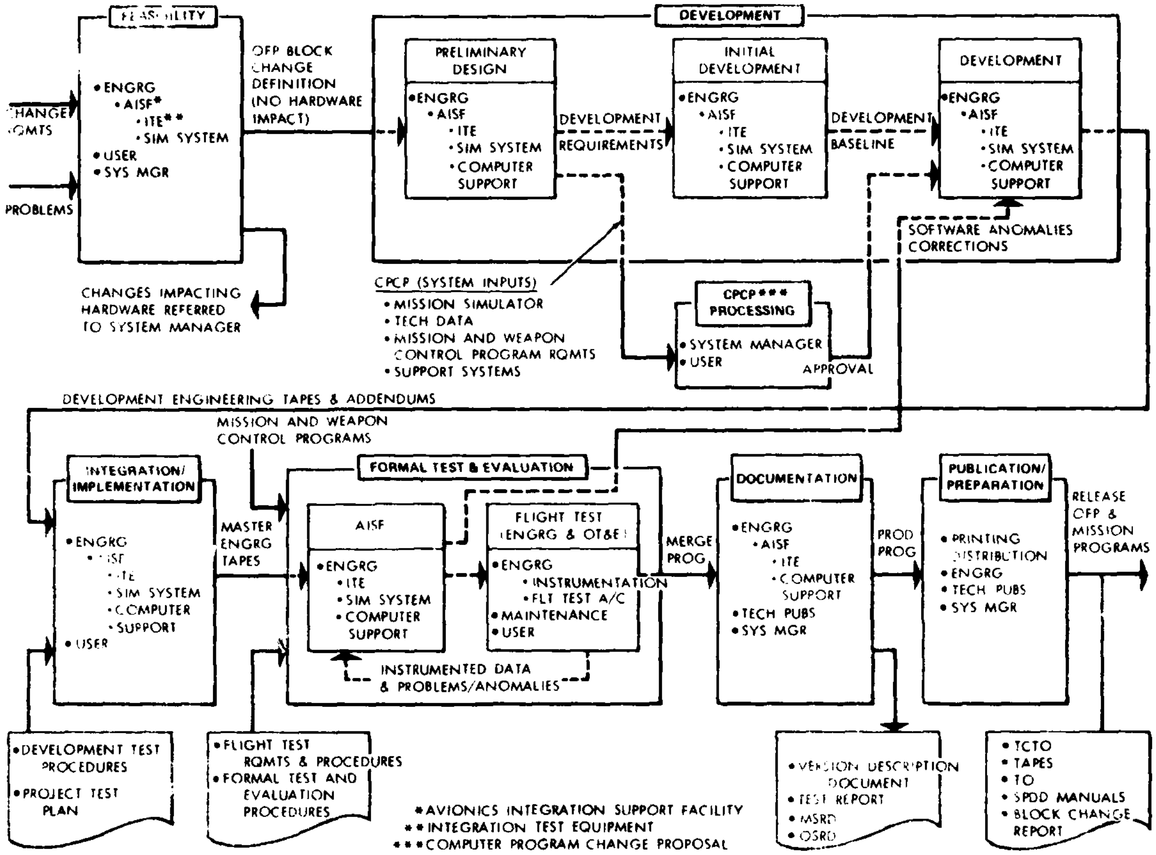


Figure 6. OFP Change Process

The Preliminary Design Phase consists of: translating requirements into engineering terms; researching flow charts and logic layouts; defining mechanization, interface, scaling, and timing requirements; developing change narratives; determining the scope of impact to documentation, technical orders, mission simulator and other weapon system software; and preparing and submitting the Computer Program Change Proposal (CPCP).

The Initial Development Phase consists of: establishing the development baseline block change programs; firming up mechanization; programming and testing preliminary code; and establishing documentation files.

The Development Phase begins with the approval of the CPCP by both the user and system manager. The development phase consists of: finalizing and testing program code for each OFP change; developing engineering tapes, addendums, and documentation; developing the project test plan; developing

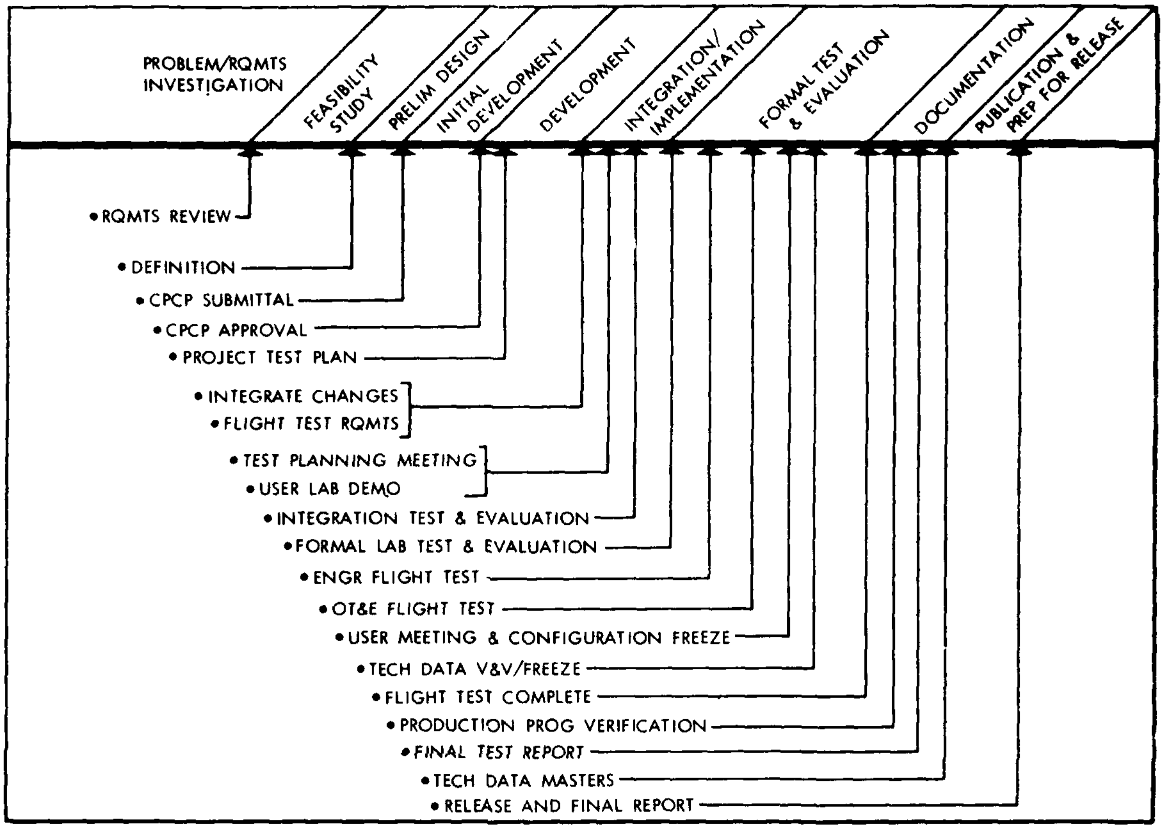


Figure 7. OFP Change Cycle

flight test, data reduction and instrumentation test requirements; preparing test procedures; and providing preliminary data for mission simulator updates.

The Integration and Implementation Phase begins with the laboratory integration of all OFP Block Change requirements. A user/engineering meeting is convened to discuss engineering and user flight test policy and to conduct a laboratory demonstration of each OFP change. Final reassembly of all approved OFP changes with the development baseline program is accomplished and the master engineering OFP tape produced. Verification testing and evaluation by the development engineering group is completed. Engineering source data for technical orders and engineering documentation is developed. Formal test and evaluation procedures are finalized. The mission and weapon control programs are produced. Laboratory test and flight test aircraft configurations are established to include aircraft computer data dumps and data reduction software. These steps are in preparation for formal test and evaluation.

The Formal Test and Evaluation Phase starts with the turnover of the master engineering OFP tape to a separate group for test and evaluation. Formal testing consists of a three phase laboratory test, instrumented engineering flight test, and user Operational Test and Evaluation (OT&E). Phase I of laboratory testing is a dynamic functional test of all OFP modes. When completed, the master engineering OFP tape is cleared for engineering flight test. Initial engineering flight test looks at overall mission suitability and clears the master engineering OFP tape for user OT&E. Once cleared, OT&E and final engineering flight test are conducted concurrently. Phase II and III of the formal laboratory test are also run concurrently. Phase II is a quantitative test of performance, a look at performance envelopes and an inspection of code and baseline documents. Phase III is the retesting of modifications resulting from problems discovered during test. Part way through formal testing a meeting between the user and software engineering is convened to review test results and to establish an OFP Block Change configuration freeze. Mandatory corrections to program discrepancies are defined, implemented and retested; trivial anomalies are accepted; and in the event a change cannot be accomplished, its coding is removed. Also, during this phase technical order source data is verified and validated by the user, engineering and the system manager. Source inputs for the mission simulator updates are finalized and delivered. At the completion of the formal test phase, the master OFP addendum tape, incorporating all corrections found during test, is merged with the master engineering OFP tape to produce the OFP release tape and the final OFP Block Change documentation.

During the Documentation Phase the OFP release tape is converted into a production version and tested. All engineering documentation is finalized; the technical order masters are prepared and made ready for reproduction. The evaluation of test results is completed and the final test report is issued.

During the Publication and Preparation for Release Phase the production OFP tapes are duplicated; engineering documentation and technical orders are published; the final OFP Block Change Report is issued; and the new OFPs and associated technical orders are concurrently released to the user under a TCTO.

The OFP Block Change process from start to finish is highly technical and primarily involves engineering resources. However, system management, technical publications and user participation are essential. The system manager has complete responsibility for the control, coordination and integration of OFP changes into the overall integrated logistics management support system, and participates to that extent. When more than one system manager is involved, the subsystem manager must work with all system managers to insure that all are satisfied.

The user is intimately involved during feasibility and change definition to establish requirements and priorities, and to assure that requirements are properly interpreted. Further, the user actively participates during the integration and test phases so that performance can be verified and acceptance granted prior to configuration freeze and OFP release. The user's primary participation during these phases is in the laboratory verification. During the documentation, publication and preparation for release phases, the system manager and technical publications personnel are extensively involved in the preparation and publication of technical orders, the duplication of OFP tapes and preparation of the TCTO for release. Engineering is responsible for the technical management, planning and direction of the complete OFP change program and is also responsible for the development and implementation of all OFP changes. Therefore, engineering is actively involved in all phases, both from the program management and technical detail aspects.

Configuration Management: Configuration Management is a discipline applying technical and administrative direction and surveillance to (1) identify and document the functional and physical characteristics of a configuration item (CI), (2) control changes to those characteristics, and (3) record and report change processing and implementation status. It is also the means by which design engineering, and cost trade-off decisions are recorded, communicated and controlled.

OFPs are managed as Computer Program Configuration Items (CPCIs). Configuration management procedures, although embodying common principles, are established by each individual ALC for the items they support. The procedures below are typical and not necessarily definitive for any specific ALC or OFP.

Each computer program planned to be designated as a CPCI is identified. The specifications and associated documentation define the CPCI baselines which the Air Force will maintain during the operational phase. Changes to the computer programs will require a corresponding change to the specifications. The Computer Program Configuration Sub-Board (CPCSB) is the central point for processing computer program changes.

When suspected system OFP problems are discovered in the field they are documented and submitted in accordance with T.O. 00-35D-54, USAF Material Deficiency Reporting System. They are reviewed and a recommendation submitted to the System Manager as to the action required. All such problems requiring OFP changes are separated into "emergency change," "urgent," or "collect for next scheduled update" categories. Problems which have a significant impact on avionics system capability or safety are placed in the emergency or urgent change category, in accordance with MIL-STD-480 priority definitions. Emergency and urgent changes will proceed quickly through the problem analysis, coding and check-out phase. The design goal is to implement the necessary requirements as quickly as practicable with a minimum change to the source OFP. Design interface problems are resolved whenever possible by person-to-person contact, followed by formal documentation, At completion of check-out, the change to the updated OFP undergoes independent verification, the goal of which is to determine that the change solves the problem and does not interfere with other normal operating modes. Verification is performed in the AISF and flight test/range facility. Technical data is compiled during the development and verification/ validation phases and made available to the appropriate technical publications organization for T.O. update. At completion of verification, the updated OFP and T.O.s are fielded, and trainers and ATE are updated as soon as practical. Documentation, such as criteria, requirements, program description, and interface documents, is made compatible with the new program.

Support software such as compilers, assemblers, simulators, loaders, link editors, and V&V programs are updated to reflect changes made to the operational software. During the operational software change cycle, required changes to support software and hardware are accomplished to accomodate the operational software change. Both support hardware and software baseline documentation are maintained to show details of all changes required for a particular operational software change. Then, upon approval of new/revised operational software, these data are updated to indicate permanent change approval. Changes to support software/hardware to enhance their capabilit are similarly documented and controlled.

Since software is intangible (can't see or touch it), the documentation must be very thorough in describing its functional and performance characteristics. Equally important is the requirement to have total visibility as to how these characteristics were derived. Without documentation that does these things, the on-going change process would eventually collapse. Figure 8 illustrates what is considered a complete set of OFP configuration control documents, and where in the OFP change cycle these documents are completed and available. The list is confined to the end item OFP and is not intended to include documentation on supporting resources, support software or other portions of the weapon system impacted by the OFP changes. A similar set of documents is obviously required for these areas. An exception to this is in the formal test and evaluation process. As noted in Figure 8, documents defining the test configuration of the laboratory, test aircraft, and mission and weapon control program are required. If and when other test resources are used in formal testing, their configuration would also be documented and become a part of the OFP configuration control documents. The physical documentation includes both automated and manually prepared documents as well as computer-stored programs.

A historical list of all requirements and problems is maintained in the Master Software Requirements Document (MSRD). All OFP source programs and programs generated after the final OFP Block Change assembly are stored on magnetic tape and hard copy listings are maintained on mircrofilm or microfiche. The

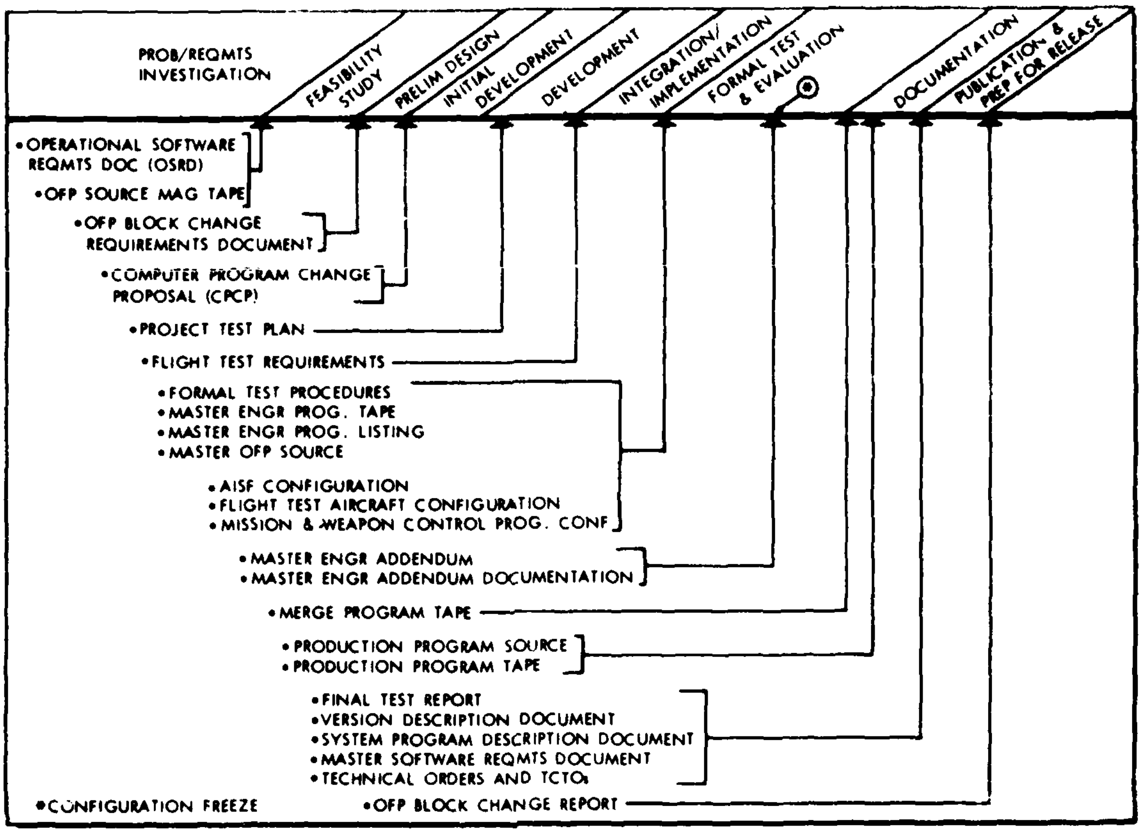


Figure 8. OFP Configuration Control Documents

OFP Block Change Requirements Document defines the initial block change definition while the final release configuration is documented using a Documentation Program. These documents become a part of the OFP Block Change Version Description Document (VDD). The Computer Program Change Proposal becomes the system manager's official configuration control document and is updated as required to reflect the final released OFP configuration. All formal test requirements, plans, procedures and reports become a part of the VDD and are a record of actual OFP performance. The OFP Block Change Report is a summary of total block change activity and results. The System Program Description Document (SPDD) is the OFP specification and is updated with each block change. It describes each of the OFP subroutines in detail and includes: narrative descriptions. inputs/outputs, interfaces, logic, timing, equations and flow charts. The VDD is the historical record of the OFP Block Change and includes all other block change documents. In summary, the

OFP source data, SPDD and program listings define the newly released OFP and the VDD defines the OFP Block Change to it. Technical orders generally aren't considered configuration control documents, but are shown because of their importance to the user and because of the detail they offer in describing the OFPs and their relationship to the aircraft system operation.

Data Collection Systems

Data collection on software support does not, in general, go to the breadth and depth desirable to support a good, solid model development effort. The most extensive automated system now in operation exists at SM-ALC, where manhours are collected and reported at the level of specific OFP block changes or software support functions.

The ALCs usually have some kind of project control system which tracks the status of individual program changes. WR-ALC is building an automated Engineering Data Management System to provide an automatic means of tracking manpower estimates versus actuals over the lifetime of the set of engineering projects within the Computer Resources Branch, as well as all other personnel charges (leave, training, etc.). That system is expected to be operational in 1981. Typical data provided by the system includes the following costs by specific change task:

- Contractor costs
- Organic Costs
  - Personnel
  - TDY
  - Test Range
  - Equipment
  - Maintenance
    - Aircraft
    - AISF

The costs are available both weekly and cumulatively. Estimated and actual manhours are also provided by task.

WR-ALC/MMRR, the Electronic Warfare Management Branch, has an on-line project control system for EW systems which tracks estimated and actual hours by specific task. Once a project is closed out it is deleted from the system. There is thus no simple way of obtaining an annual report of all manhours expended on all tasks.

00-ALC has an automated Project Accounting and Control system currently operating. It tracks estimated and actual manhours by task identifier, but a manual analysis would be required to summarize manhours by task type. That system does not track all manhours, but only those expended in support of specific program changes. Figure 9 portrays a report produced by that system.

PROJECT DETAIL ANALYSIS

F16 BLOCK IIMMA-C76-003 3337 PERIOD AUG 25,79

PROJECT LEADER W WELCH

DESCRIPTION TAG DEF IDENT. PRI	REVISED EST	TIME REMAINING	ACTUAL TIME LST PRD/ TO DATE	REVISED START	REVISED FINISH	PROJECTED FINISH	VAR +/-	PERCENT CPL/COM	ASSIGNED BUDGET VS	RESOURCE ACTUAL	FLAG
SMS FLT LC MODE 3LAC	150.0	134.0	16.0	MAR 2,79	MAR 2,80	JUN 5,80	95-	10/ 10	L CLOHAN 150/	16	D
STORES CANDIDATE CHANG 3MAC	354.0	176.0	176.0	MAR 2,79	MAR 2,80	MAR 5,80	3-	49/ 20	W WELCH 354/	176	C
H V MECH & SLEW RATE 3MAD	200.0	62.0	138.0	APR 21,79	MAR 2,80	OCT 30,79	125	69/ 20	G STRONG 200/	138	
CCFF/VFF 3MBC	177.0	117.0	60.0	MAR 2,79	MAR 2,80	MAY 2,80	61-	33/ 10	W WELCH 177/	60	D
TABLE DRIVEN FX 3MCC	177.0	53.0	124.0	MAR 2,79	MAR 2,80	DEC 20,79	74	70/ 10	W WELCH 177/	124	
PREV INDV FCS UPDATE 3CAC	80.0	1.0	80.0	MAR 2,79	MAR 2,80	SEP 6,79	179	98/ 5	J OLDHAM 80/	80 B	T
CCFF/VFF 3CBC	160.0	86.0	74.0	MAR 2,79	MAR 2,80	MAR 6,80	4-	46/ 10	J OLDHAM 160/	74	C
MARK INITIALIZATION 3CCC	40.0	1.0	40.0	MAR 2,79	MAR 2,80	SEP 7,79	178	97/ 3	J OLDHAM 40/	40 B	T
VIF ON SMS 3CDC	160.0	160.0		MAR 2,79	MAR 2,80	AUG 8,80	155-	/ 10	J OLDHAM 160/	0	C
STANDARD ELEV ENTRY 3SAC	40.0		10.0	MAR 2,79	MAR 2,80	JUN 16,79*	261	100/ 3	R SUZUKI 40/	10	
PREV INADV FCS UPDATE 3SBC	90.0	60.0		MAR 2,79	MAR 2,80	AUG 8,80	155-	/ 5	R SUZUKI 80/	0	D

Figure 9. OO-ALC Project Accounting and Control Report

Tracking manhours in support of ATE at SA-ALC is difficult because of the large number of organizations involved. The Software Support Center (OC-ALC/MATT) can relate manhours to projects, but cannot provide a good description of specific projects without a lot of detailed digging.

Manhours expended in support of the A-7D software by the OC-ALC organization on-site at China Lake Naval Weapons Center are tracked only at a general level or on annual basis. Because of the small size of the organization, a formal control system is not required to keep track of what's going on.

OFP Support at OC-ALC/China Lake (ref Vol. II, Appendix A)

OC-ALC (Tinker AFB, Oklahoma) is responsible for a variety of systems, previously identified in Table 6 on page 31. Table 11 lists the specific navigation systems and their current support status. Most of the systems are currently being supported by contractor personnel. OC-ALC is in the process of developing a consolidated support concept to service a number of the upcoming systems. Total forecasted manpower requirements for embedded computer system support within OC-ALC (D/MM) exceed 100, growing to 185 by FY85.

TABLE 11. NAVIGATION SYSTEMS SUPPORT, OC-ALC

System	Navigation Unit	Support Posture
A-10	Form, Fit, Function (F^3)	Contract
E-3A	Dual Carousel IV/Omega	Integrated with E-3A AISF
AGM-69	Carrier (LN-15) Air Vehicle (KT-76)	Integrated with SRAM AISF
A-7	KT-73	Integrated with A-7 AISF
F-16	SKN-2400 IMU (F^3)	Integrated with F-16 AISF
B52D	GEANS	Integrated with B-52 AISF
B-52G/H	Dual GEANS	Integrated with B-52 AISF
F-15	LN-31	Contract
F-4	F4-E and RF -4C (LN-12) (to be replaced; F^3 or LN-33 or?) and ARN 101 (SKN-2400)	Integrated with F-4 AISF
E-4	Carousel IV	Contract
C-135	Carousel IV (KC-135 and EC-135J) LN-20 (RC-135) and DNC	Contract or under development
C-5A	Triple Carousel IV	RIW
C-141	Dual Carousel IV	RIW
C-130	LN-16, AC-130 Gunship (KT-73)	Contract
F-5	Foreign Countries (LN-33)	Contract

The A-7D aircraft is supported through a joint venture with the Navy at China Lake Naval Weapons Center, California (office symbol OC-ALC/MMECZA). MMECZA has a work force of six civil service personnel. Additionally, they obtain eight manyears per year of assistance from the Navy through a Military Interdepartmental Purchase Request (MIPR). The civil service staffing consists of one supervisory electronic engineer, a mathematician, a computer scientist, an equipment specialist (avionics), a computer operator, and a secretary.

MMECZA utilizes approximately 4500 ft^2 of laboratory and office space. Computer support consists of eight DEC PDP 11/xx series computers, a Honeywell Sigma V, and a Hewlett Packard 9830. The resident support software to support the 16K A-7D OFP occupies about 275K 16-bit words of core memory. Approximately 50 flight tests of an instrumented A-7D aircraft are required each year to check out software changes.

OFP Support at SM-ALC (ref Vol. II, Appendixes B, C, and D)

SM-ALC (McClellan AFB, California) is responsible for supporting the F/FB-111 aircraft. In addition, SM-ALC/MMECF is responsible for a number of ground communications, electronics and meteorological systems. The organizational breakout of SM-ALC/MMEC is as follows:

- MMEC - Computer Resources Branch
  - MMECP - F/FB-111 Support
  - MMECM - Software Management
  - MMECS - Administration
  - MMECF - Ground Communications, Electronics and Meteorological Systems Support

MMECP has approximately 81 personnel (as of December 1979) organized as shown in Table 12.

MMECP supports seven OFPs: one for each of the two computers (general navigation computer and weapons delivery computer) on each of the three aircraft types (F-111D, F-111F, FB-111A) plus one OFP for the navigation computer unit common to all three aircraft.

MMECP occupies 10,800 ft^2 of standard computer-type facilities. The AISF was described earlier, and is portrayed in Figure 5 on page 47. Equipment cost is estimated at $40 million. The support software in the AISF consists of over 700,000 source lines of computer programming. Flight testing requires approximately forty sorties (120 flight hours) per year; this is on the basis of one block change every eighteen months for each of the three aircraft, or one block change every six months.

TABLE 12. SM-ALC/MMECP STAFFING

Function	Number of Personnel		
	Air Force	Civil Service	Contractor
Management/Secretary		2	3
FB-111A S/W Engineering		1	5
F-111D S/W Engineering	1		5
F-111F/Pavetack S/W Engineering	1		5
Mission Programs	1		
F-111 A/E Acquistion Support		1	1
F-111 AISF Enhancements and S/W Support			15
F-111 OFP Mk II V & V		2	3
Flight Test Support			5
S/W Configuration Management			4
TSU			5
Special Projects	3	8	10
Major AISF Upgrades			(5-10 off-premise)
Totals	6	14	61(+ 5-10)

<u>OFP Support at 00-ALC</u> (ref Vol. II, Appendix E)

00-ALC (Hill AFB, Utah) is currently implementing a capability to support F-4 and F-16 OFPs. The F-16 has seven computers, of which five currently have software OFPs. The F-4 has three OFPs. The F-16 is still under contractor support; 00-ALC operations are basically restricted to independent validation and verification activities and preparation for F-16 support.

00-ALC has an organization which is somewhat more segmented than the other ALCs. While 00-ALC/MMECA is responsible for design and development of OFP changes, MMETA provides independent validation and verification (both ground simulation and flight testing) of those changes, and also provides AISF services to MMECA. ACDCS (comptroller) provides programming support for the support software (both AISF and general purpose computer complex). Table 13 provides an organizational breakout of the 87 personnel involved in OFP support. 00-ALC is currently manned to organically support only three of the seven F-16 OFPs. The other four are under contractual support and require about 1/2 person each. Radar support may go organic in the future.

MMECA and ACDCS occupy 5000 square ft of office space. MMETA occupies over 20,000 square ft of laboratory/office space. Computing support equipment for the F-16 is listed in Table 14.

TABLE 13. OO-ALC OFP SUPPORT

Organization	Total	F-16	F-4	Flight Test
MMECA(1)	33(2)	15	17	
MMETA	15(3)	7	3	4
ACDCS:	39(4)			
AISF		9	14	
GPCC		8	8	
	87(2)	39	42	4

1. Personnel shift between F-4 and F-16 in response to workload requirements
2. Includes section chief.
3. Includes section chief.
4. Five persons shared between F-4/F-16.

TABLE 14. OO-ALC F-16 COMPUTING SUPPORT EQUIPMENT

IBM 360-65 - General Purpose Computer
DEC System 10 - F-16 AISF
Dynamic System Simulator
Avionics Equipment Bench (Tower)
Avionics Intermediate Shop

The support software involves programs requiring over 1.8 million words of core, plus another set of data reduction, post flight and general purpose programs of over 65,000 lines of code and comment. Flight test requirements are anticipated to be 90 flight hours per block change for the F-16.

OFP Support at WR-ALC (ref Vol. II, Appendix F)

WR-ALC (Robins AFB, Georgia) supports a number of systems, identified previously in Table 6 on page 31. Their major effort is devoted to preparing for support of the F-15 aircraft. The F-15 has two OFPs, one in the central computer and one in the radar data processor.

WR-ALC/MMEC currently has 95 personnel assigned, including 3 military, against an authorization of 130, excluding those devoted to ATE support. Of those 95, 50 are devoted to F-15 support.

WR-ALC/MMEC is being reorganized from three into five sections. One section will have three units. The new organization will be as follows:

- MMEC - Computer Resources Branch
  - MMECA - ATE Acquisition
  - MMECT - ATE Support
  - MMECE - AISF Equipment and Support
  - MMECV - Validation & Verification
  - MMECD - Weapon System Integration
    - MMECDF - F-15 OFP Design
    - MMECDA - Acquisition Support
    - MMECDM - Management

MMEC will ultimately occupy over 50,000 sq. ft. of office and laboratory space. They currently have about 16,000 sq. ft. Computing support equipment will include a Dynamic Simulation System, a Data Reduction and Analysis System and a Flight Test Preprocessing System. There will be one fully instrumented F-15 dedicated to flight test. It is expected to fly about 100 hrs/year in support of all changes (hardware, software and EW). Approximately 15 flight hours are expected to be required to check out each software block change.

<u>EW Support</u> (ref Vol.II, Appendix G)

EW systems, including software, are managed at WR-ALC. WR-ALC/MMR (EW Management) is responsible for the EW systems shown in Table 15. Current staffing (Dec'79) of MMRR, the Engineering and Reliability Branch, is 223 against 283 authorized positions. Their estimated manpower requirement for FY'80 is 318. The breakout of that requirement by system is also given in Table 15.

MMRR is organized into six sections:

- MMRRC - Jammers
- MMRRV - Receivers
- MMRRI - Integrated systems
- MMRRA - Threat simulation to test systems
- MMRRS - Technical data, spares definition, user interface. deficiency reports
- MMRRW - Administration, budget, configuration control

MMRR will have an integrated support station (illustrated in Figure 10) for each software controlled system they support. Any flight test requirements will be supported by the host aircraft of the specific EW system. The F-15 Tactical Electronic Warfare System (TEWS), for example, is tested on the same aircraft which supports F-15 OFP software changes.

TABLE 15. EW SYSTEMS

System	Personnel Requirement, FY80
ALQ-131*	30.7
ALQ-165 (ASPJ)	4.1
ALQ-155*	16.2
ESAS*	6.3
ALQ-117	5.3
ALQ-119	27.6
APR-38*	34.6
ALQ-125*	7.2
ALR-56*	18.8
ALQ-135*	11.5
ALE-45	4.0
IRS	7.3
USM-464 (FLTS)	16.4
ALQ-99	16.4
ARC	8.6
ALR-46*	39.6
ALR-62*	20.8
ALQ-153	5.7
ALR-69*	36.7
	317.8

*Software-controlled systems

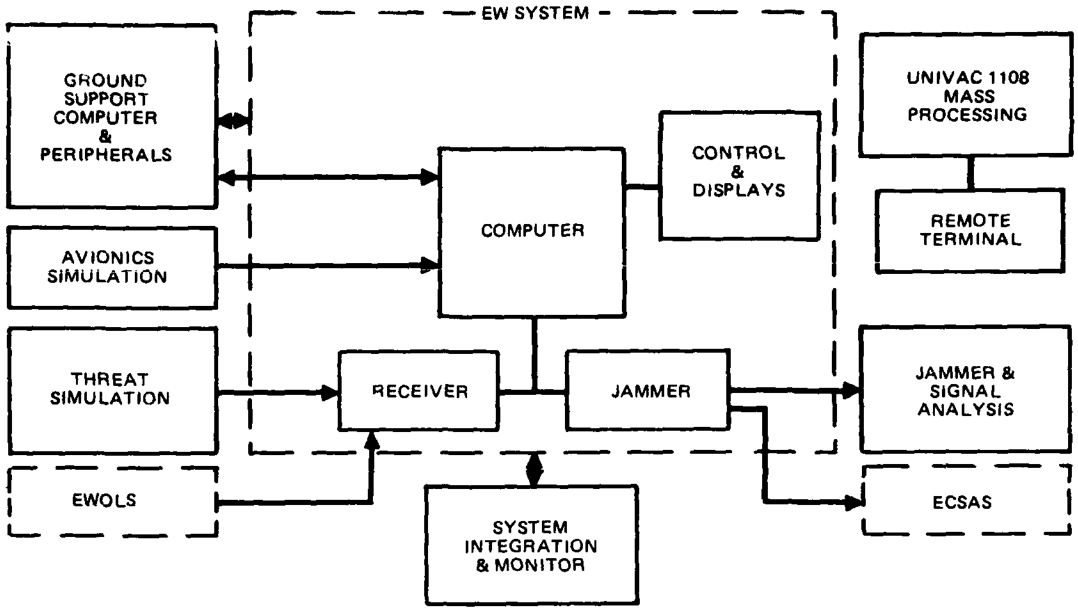


Figure 10. EW Integration Support Station

ATE Software Support (ref Vol.II, Appendix H)

ATE has three categories of software: system software, support software, and unit under test (UUT) software. System software controls the basic function of the ATE system. Support software is the assemblers, compilers, verification tools, etc. used in developing system and UUT software. UUT software consists of those programs necessary to perform the testing of a specific piece of hardware on an ATE system. Each unit which is being tested on ATE has one or more UUT programs which exercise the unit in order to test various functions.

SA-ALC (Kelly AFB, Texas) has been designated as the system manager for all ATE. As such, they control the system software for those systems that are programmable. SA-ALC estimates that of the 400-500 identifiable ATE systems, about 20-30 are particularly active with regard to software. ATE system software is generally less formally controlled than OFP and EW software, due primarily to the large size of the programs. For example, the F-16 Avionics Intermediate ATE System has 500,000 lines of code. Table 8 on page 32 lists the present and projected systems SA-ALC is supporting.

The major organizations at SA-ALC which are involved in ATE software support are:

MMIM - Logistics Management Branch
MMIR - Engineering and Reliability Branch
MMEC - Computer Resources Branch
MATT - Software Support Center

MMIM analyzes planning and programming documents and data to assure adequate logistics coverage. It also provides managers to administer, coordinate, and control the management of ATE software.

MMIR has responsibility for full range engineering and technical integration of ATE equipment and software to assure design performance and compatibility, and to insure that all ATE computer program deficiency reports are processed and controlled.

MMEC performs the following functions: identify minimum essential weapon system computer resources documentation requirements for operational support; conduct or participate in verification and validation of assigned ECS programs; evaluate and define the cause of software deficiencies related to ATE, determine and recommend changes required to correct those deficiencies; maintain files and issue computer programs and documentation; evaluate contractor-prepared ECPs for computer programs and documentation and apply cost effectiveness criteria. It is also the final engineering approval authority for embedded computer systems integral to ATE systems.

MATT provides programming support resources as required. In particular, since SA-ALC is system manager for ATE, they are also item managers for ATE components, some of which are themselves tested on ATE, and require UUT software. MATT, as a major responsibility, develops UUT software for any ATE for which MMI has management responsibility.

UUT software is that software written so that the ATE can perform specific tests on specific hardware items. It is controlled under the CPIN system as part of the specific hardware item, rather than as part of the ATE. As such, each UUT program is controlled by the cognizant item manager. UUT makes up the largest proportion of embedded computer system software in terms of number of programs.

GENERAL OBSERVATIONS

Software support within the ALCs is a task-oriented process. A set of individual computer program changes is grouped into a single block change which is controlled as a configuration-managed item. (This is somewhat less true for EW and ATE software).

Although the same basic process occurs at each ALC, there are differences due to particular organizational arrangements, workload requirements, and resource availabilities. These differences (especially in data collection systems) make it somewhat difficult to discern the underlying parameters relating software change requirements to the resources necessary to implement those changes. Those relationships are the subject of the next section.

VI. ANALYSIS OF DATA AND COST DRIVERS

As part of the field survey process, various data were acquired which relate to software support costs and utilization of support resources. Those data include manhours related to software changes, flying hour cost factors for test aircraft, quality ratings of software packages, and the opinions of experienced software engineers as to the key factors which should be considered in predicting support costs.

GENERAL SOFTWARE SUPPORT DATA

Data were collected from four different ALCs on six different sample systems. The ALCs and systems are shown in Table 16.

TABLE 16. ALCS AND SAMPLE SYSTEMS

ALC	System
OC/NWC	A-7D
SM	FB-111A
SM	F-111F
SM	F/FB-111 Support S/W
OO	F-16
WR	F-15

Table 17 summarizes the quantitative data related to general ALC support. Note that these numbers do not necessarily reflect the total software support at the ALCs, but only that related to specific packages. Table 18 summarizes the data related to the sample packages.

Examination of those tables reveals several interesting items. The number of personnel per package varies from 8.5 on the F/FB-111 to 25 on the F-15. It runs about 13 on the F-16. The reason for the variation on the two most recent systems needs to be explored in greater depth. It should also be noted that of the 121K-words of F-16 software, about 72K are currently being supported organically; the F-15 has approximately 40K-words of operational software.

The cost of support equipment at both SM-ALC and OO-ALC is on the order of $30-40 million. As that figure totally swamps the annual operating costs for a number of years, in-depth investigation is needed to determine how sensitive that figure is to various OFP design alternatives and support concepts. A related fact is that vendor support on the Harris, Interdata and PDP computers at SM-ALC runs over $400k/year.

TABLE 17. ALC-RELATED DATA

ALC:	OC/NWC	SM	OO	WR
Aircraft Supported	A-7D	FB-111A F-111D F-111F	F-16 F-4	F-15
Nr of OFPs	1	7	7 (F-16) 3 (F-4)	2
Nr of OFPs organically supported	1	7	6	2
Nr of Personnel organically supporting OFPs	14	60	85	50
Nr Pers/Package	14	8.5	14.2	25
Facilities/ (ft^2)	4500	10,800	14,000	10,300
Support Hardware Cost	$265K	$40,000K	$31,400K	N/A

The frequency of changes is another interesting datum. The number of changes (where a change can vary in size from "petite" to "extra large") on the A-7D and F/FB-111 is on the order of 20/yr. (The F-111D is about 16/yr.) Table 19 portrays the composite change history on those aircraft. Change history on the A-7D and the F-111D/F for the period 1970-1975 was detailed in an AFIT thesis[1] by Lt. Bruce Vendt. F-111D change history was acquired from SM-ALC along with data on the FB-111A and F-111F.

Table 19 shows that approximately 1/4 to 1/2 of the changes are corrections. (Note that a correction is not necessarily a programming error, but could be a design deficiency.) Approximately 1/3 of the changes are refinements of existing capability. Most of the remainder are additions to existing capability. For the F/FB-111 this amounts to about 10-20%, while for the A-7D it is 40% (most of which came later in the life cycle).

1. B. A. Vendt, "Software Support for F-16 Avionic Computers," Air Force Institute of Technology, Wright-Patterson AFB, Ohio 45433, December 1975. (AD A020 361)

TABLE 18. SAMPLE PACKAGE DATA

	A-7D	FB-111A	F-111F	F/FB-111 Support	F-16	F-15
[illegible]	1	2	2	-	7*	2
[illegible]	Nav/Wpns Del.	Nav/Wpns Del.	Nav/Wpns Del.	Ground Simulation	Nav/Wpns Del.	Fire Control/ Wpns Del.
[illegible] (OFPs)	15K	2x16K	2x16K	300K+	113K	35K
[illegible]	Assembly	Asembly	Assembly	Fort/Assy	Jovial/Assy	Assembly
[illegible]	IBM TC-2	IBM CP-2	IBM CP-2	Harris/4	Multiple	IBM AP-1 HCM-213
[illegible]	16 Bit	16 Bit	16 Bit	24 Bit	Varies: 8/32 Bit	32/24-Bit
Memory Size	16K	2x16K	2x16K	6x80K	152K	16K, 24K
Memory Fill	39%	99%	99%	Virtual Memory	Varies: 69 - 100%	69%, 100%
Year Developed	1968	1968	1968	1968	1976	1970/72
Developer	IBM/Vought	Autonetics	Autonetics	General Dynamics	Various	McDonnell/ Hughes
Support S/W (Words)	263K	702K	702K	-	From 2-3 million Words	n/a
Changes/Yr	22	21	20	26	n/a	n/a
Nr Personnel Supporting	14	17	17	5	39	50
Nr Pers/Package	14	8-1/2	8-1/2	-	13	25
Flight Test Sorties	52/yr	21/block	13/block	-	n/a	n/a
Flight Test Hours	104/yr	64/block	34/block	-	132/yr	15/block

* 5 software OFPs, 3 supported organically

TABLE 19. COMPOSITE CHANGE HISTORY

System:	A-7	FB-111A	F-111F		F-111D	
Time Period:	1969-1978[1,2]	1975-1979[3]	1970-1973[2]	1975-1979[3]	1970-1973[2]	1975-1979[3]
Total Nr. Changes	225	103	95	106	88	73
Corrections	40	38	49	38	46	40
Deletions	3	16	0	11	0	1
Optimizations	1	2	0	5	0	6
Refinements	80	32	31	29	32	19
Added Capabilities	91	15	4	23	6	7
Unknown	0	0	11	0	4	0

1. Source is Mark Jacobson, OC-ALC.
2. Source is AFIT thesis, "Software Support for the F-16 Avionics Computer", Lt. Bruce A. Vendt, USAF, December 1975.
3. Source is A. E. Patterson, SM-ALC.

MANHOUR AND CHANGE DATA

SM-ALC has a detailed history of manhour and change data dating back to FY'77, covering all three aircraft. Table 20 summarizes these data for fiscal years 1977, '78 and '79. Figure 11 summarizes the F/FB-111A manhour data with respect to specific OFP block changes. The most complete data available are on blocks FB-15, F-12, D-19 and FB-16. The data for those blocks are given in Table 21, and graphed in Figure 12. Note that there appears to be a fairly regular relationship between number of changes and amount of manhours. Flight test requirements do not correlate nearly as well.

Similar data for the A-7D are shown in Figure 13, Table 22 and Figure 14. The A-7D data embody various assumptions regarding manhours per year and the application of those manhours to specific block changes, and therefore are not as reliable as the F-111 data. Nevertheless, they do indicate that the manhours per change are roughly the same magnitude as for the F-111, that is, roughly 1000 hours per change.

TABLE 20. ANALYSIS OF F/FB-111 MANHOURS

Aircraft/ Function	OFP Block	FY77	FY78	FY79	Total
FB-111A		18041	15069	9809	42919
	FB-14	329	-	-	329
	FB-15	17704	366	10	18080
	FB-16	8	14703	6932	21643
	FB-17	-	-	2867	2867
F-111F		16926	8877	20243	46046
	F-11	393	-	-	393
	F-12	16533	7928	10168	34629
	F-13	-	949	10075	11024
F-111D		13880	19376	14373	47629
	D-17	130	-	-	130
	D-18	12072	963	170	13205
	D-19	1678	16732	3353	21763
	D-20	-	24	8366	8390
	D-157*	-	1657	2484	4141
OFP Mgt/ Other		6391	3288	6467	16146
Software Support		23790	29976	2309[illegible]	76660
Special		28982	35224	3254[illegible]	967[illegible]4
Leave/Training		19904	23580	24597	[illegible]81
Total		27914	1135190	129131	[illegible]

* Special flight test/engineering required outside a normal block change cycle to analyze a specific weapon delivery problem.

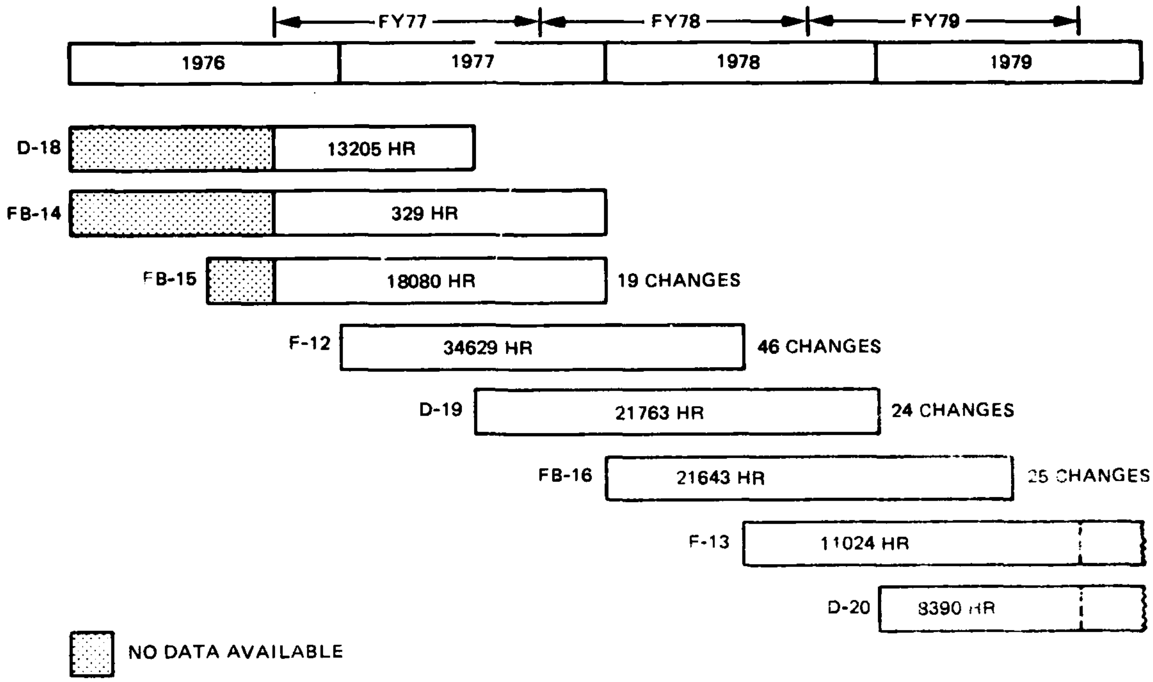


Figure 11. F/FB-111 Change History

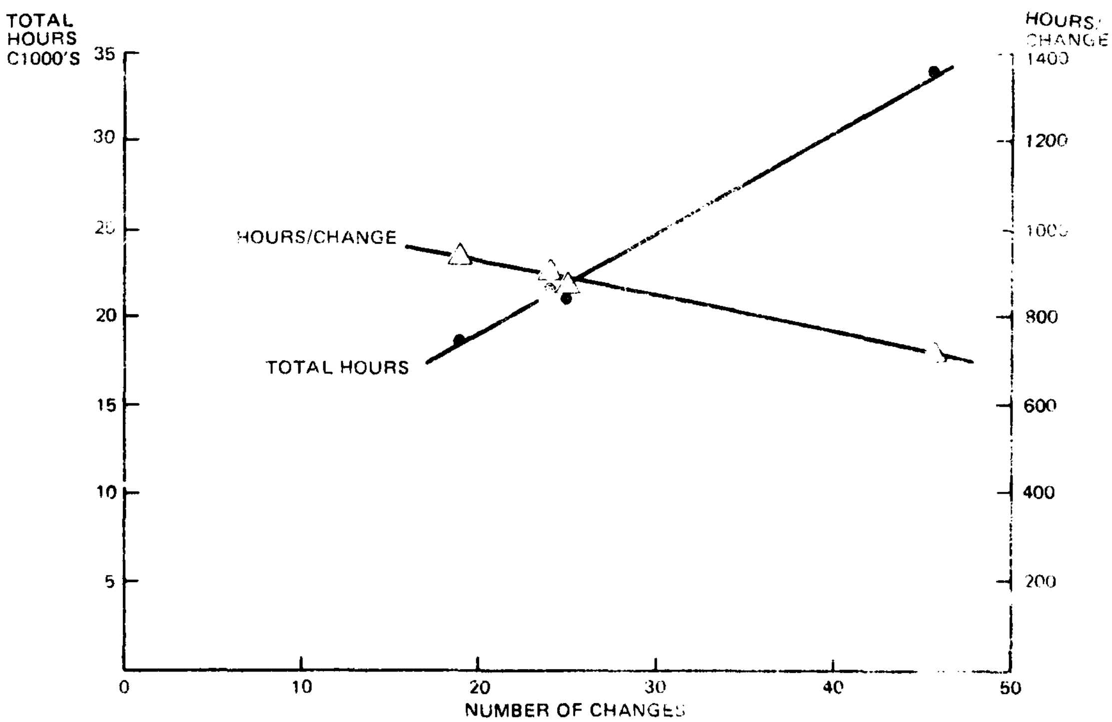


Figure 12. F/FB-111 Manhours Per Change

TABLE 21. F/FB-111 BLOCK CHANGE DATA

Block	Release Date	Number Changes	Manhrs	Hr/ Change	Flight Test Sorties	FH
FB-15	12-77	19	18080	951	23	67
FB-16	6-79	25	21643	866	19	60.5
F-12	6-78	46	34629	753	13	34.2
D-19*	-	24	21763	907	37	88.2

*Not formally released; awaiting further engineering action

TABLE 22. A-7D BLOCK CHANGE DATA

Release Date	Number Changes	Manyr	HR/ Change	Flight Test Sorties
12-72	30	13.5	846[1]	-
6-75	20	14.5	1631[1]	-
[illegible]-76	4	n/a	n/a	-
10-78	58	30	910[2]	103

1. Contractor: Assume 1880 manhr/yr
2. Air Force: Assume 1760 manhr/yr

The OFP block data for the F/FB-111 and A-7D appear to indicate that the amount of effort expended per change is relatively constant across the blocks, in spite of the fact that there can be significant variations in the amount of effort expended on individual changes. Several conclusions can be drawn from this observation, specifically the following two: 1) manhours required for individual changes are lognormally distributed, and 2) technical factors due to resource and time limitations impact the mix of changes accepted in a block.

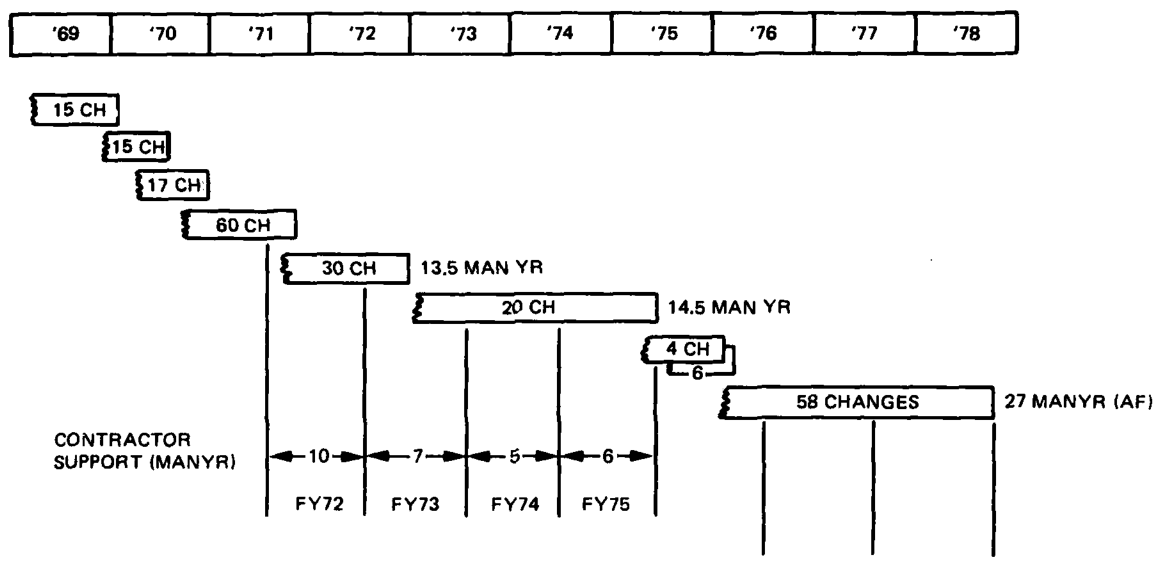


Figure 13. A-7D Change History

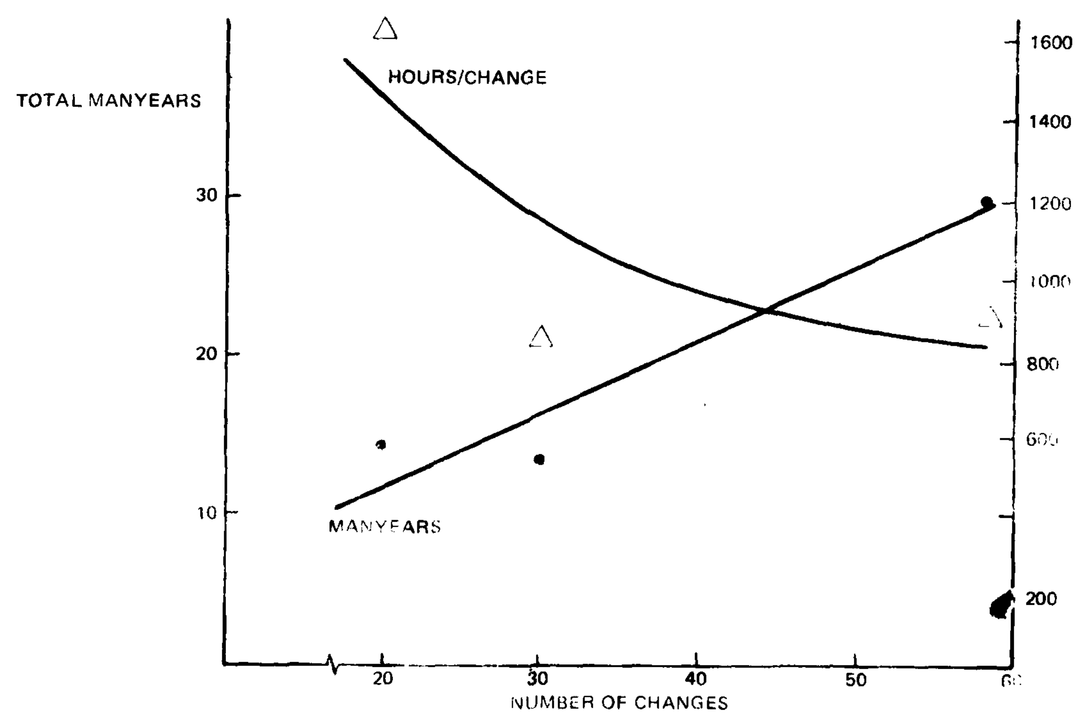


Figure 14. A-7D Manhours Per Change

Unfortunately there is no substantive data available to accept or reject these hypotheses. However, if either of them is true, a prediction of the estimated change rate on future systems could be a significant predictor of support costs. The process of how changes are generated, evaluated and accepted should be investigated in detail during the model development effort.

FLYING HOUR COST FACTORS

After changes to an OFP are verified/validated in the AISF, it is necessary to check them out in an instrumented test aircraft operated by a qualified test pilot. The flying time, excluding cost of range time and pilot salary, can run several hundred thousands of dollars per year. Table 23 gives the flying hour cost factors for the sample aircraft, as computed based on AFR 173-10, "USAF Cost and Planning Factors." No attempt was made to gather data on range time usage and cost during this study. Those data need to be collected during the Phase II model development.

TABLE 23. AIRCRAFT FLYING-HOUR COST FACTORS

Aircraft	Total $ per F/H	Fuel $ per F/H	Data Source[1]
A-7D	1,410 1,439	297 297	Table 1[2] Table 1A[3]
F-111F	2,874 2,992	637 637	Table 1 Table 1A
FB-111A	3,007 2,957	611 611	Table 1 Table 1A
F-16A	1,248 1,236	323 323	Table 1 Table 1A
F-15	2,158 2,017	552 552	Table 1 Table 1A

Notes: 1. AFR 173-10,, Vol. I, 6 Feb 75, Rev 1977
2. Factors for planning, programming and budgeting
3. Factors for cost estimating studies

SOFTWARE PACKAGE QUALITY

It seems intuitively that the "quality" of an OFP will have some impact on its modifiability and hence on OFP support costs. An attempt was made to measure the quality of several of the sample packages using a list of 45 quality attributes. Software engineers at the ALCs familiar with the packages were asked to rate their package on the 45 attributes on a scale from 1 (poor) to 10 (excellent). A simple average quality rating was then computed for each package, summing the attribute scores and dividing by the number of attributes. (Not all packages were rated on all attributes.) The individual package quality ratings are contained in Volume II of this report. It should be noted that no attempt was made to control inter-individual differences in package rating.

Table 24 shows the package averages. Note that the four aircraft OFPs all rated approximately the same, while the F/FB-111 support software is a full two points higher. The major factors on which it seemed to rate higher were accessibility, augmentability, legibility, maintainability, modifiability, robustness and simplicity.

TABLE 24. PACKAGE QUALITY RATINGS

A-7D	5.7
FB-111A	5.2
F-111F	5.2
F/FB-111 Support S/W	7.8
F-16 (Combined)	5.6

Although the results of this initial study of package quality showed no significant differences which might impact OFP support costs, this area should be studied in more depth in Phase II.

WHAT DO THE EXPERTS SAY?

As part of the field survey process, key people at each ALC were asked what they thought the key elements were to consider in making a projection of software support costs on a new avionics system. Their responses are tabulated in Table 25.

The greatest number of factors have to do with the overall system requirements and functions. The amount of spare memory (expandability) and the quality of programming support tools available (hardware, software, documentation) were mentioned as very important variables. Programming language appears only implicitly. as one factor affecting degree of maintainability Several times during the surveys the fact was mentioned that problem analysis is a major time consumer; once the problem is correctly discerned, the decision can be (sometimes) trivial.

TABLE 25. FACTORS CONSIDERED IMPORTANT BY ALC PERSONNEL

REQUIREMENTS VARIABLES
Weapon system scenario/utilization
System application/functions
Similarity to existing systems
Likelihood of substantial system enhancements
Mission requirements (TAC has more precise testing requirements than SAC)
Data flows
Accuracy requirements
System interfaces
DESIGN AND CODING VARIABLES
System structure and complexity
Degree of maintainability
Development methods and rationale
HARDWARE CONSTRAINTS
Amount of spare memory
Amount of firmware (ROM)
PROGRAMMING ENVIRONMENT
Quality/availability of documentation
Support tools - hardware and software
MANAGEMENT ENVIRONMENT
Amount of software assigned to system manager versus item manager.
PMRT date
Whole system transfer, or just certain configurations?
Is system being built by somebody who supports another system I'm using?
Degree of management support for maintenance effort

SUPPORT COST GENERATION

The software in a typical digital weapon system must be changed in response to changes in the operational environment. changing mission requirements, enhanced capabilities or operationally discovered deficiencies. The change process itself involves aspects of management, such as change receipt and control, funds allocation and change distribution, as well as aspects of system engineering, design engineering and product engineering such as technical evaluation performance evaluation, change definition, integration testing and acceptance testing.

Supporting this change process requires six categories of resources: personnel, support equipment, support software, facilities, data and documentation, and flight test aircraft. These six categories represent the support cost elements that need to be considered in predicting software support costs.

Costs related to the six resource categories fall into two classes: initial and recurring. Specifically, resource expenditures are as follows:

Initial - support hardware acquisition
support software acquisition
program documentation
test aircraft
facilities
personnel (training, etc.)

Recurring - personnel salaries
flight test & range time
support hardware maintenance expense

There are a number of intermediate variables and primary drivers listed in Table 26 which affect the initial and recurring costs of the six resource categories. Their interrelationships are diagrammed in Figure 15. These relationships will form the basis for structuring the model to be developed in Phase II. Using just the data available on the intermediate variables plus past history on amount of support hardware, etc., it would be fairly simple to construct an analogy model to predict support costs. The major data points would come from A-7D, F/FB-111, F-15, and F-16. However, in order to do in-depth trade studies of alternative software designs, support concepts. etc., it is necessary to move beyond the intermediate variables to the primary inputs.

<u>Personnel</u>

The category of personnel consumes the largest amount of recurring dollars annually, and is almost universally the only cost element treated by most models. The number of personnel required is basically a function of the number of changes occurring and the manhours required per change. Quantity of

TABLE 26. MAJOR COST CATEGORIES AND DRIVERS

Cost Category	Intermediate Variables	Primary Drivers
Number of personnel	Manhours	
	Manhours/change	
	Productivity	Personnel experience/ training Available tools Incentive environment
	Modifiability	Program architecture Hardware constraints
	Size of change	Requirements
	Number of changes	
	Reliability Requirements	Program architecture Changes in threat, capability, etc.
	Efficiency factor (productive manhours/ year)	
Personnel (training, etc.)	Number of personnel	
Amount of support hardware		Prime system hardware
	Analysis requirements V & V requirements	Operational requirements changes
Facility size	Number of personnel Amount of support hardware	
Support hardware maintenance expense	Amount of support hardware	
Flight test aircraft acquisition	Number of hours	V & V requirements
Flight test/range time expense	Number of hours Cost per hour	V & V requirements
Software documentation acquisition		Desired support center productivity

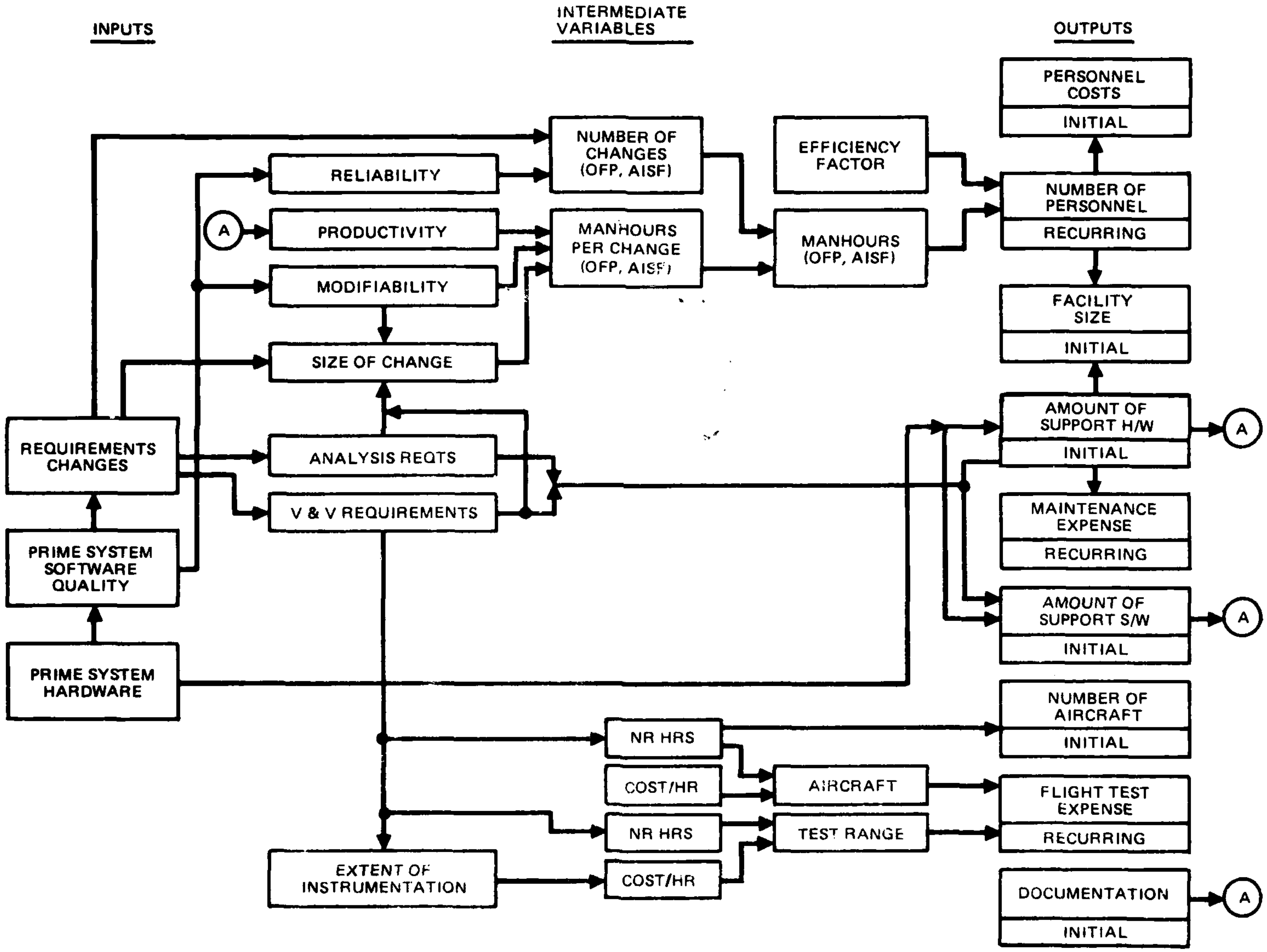


Figure 15 Software Support Cost Generation

personnel can also be affected by administrative requirements, training requirements, etc. Note that number of personnel is driven both by operational software modifications and by support software (AISF) modifications.

The intermediate variables affecting the number of personnel are the efficiency factor (productive manhours per year per person) and total manhours required (for both OFP and AISF support). Those manhours, in turn, are the product of the number of changes and the manhours per change. Data are fairly readily available on all those parameters.

However, rather than just extrapolating past levels of software change rates, it is desirable to understand the relative proportion of reliability-induced and requirements-induced changes, what really drives the requirements-induced changes, and how well they can be predicted.

Similarly, manhours per change is conditioned by support center productivity, software modifiability, and magnitude of specific changes. Productivity is a function of a number of factors such as available tools (software, hardware and documentation), personnel experience and training, and the incentive environment. Modifiability relates to the whole complex of factors regarding program structure (architecture, size, complexity, language, understandability, etc.) and its host hardware (especially spare memory, and timing requirements). The factors affecting size of a change also need to be understood.

Support Hardware/Software and Facilities

A major cost element which is universally left out of software cost models is the cost of acquiring the support hardware and software required to make and check out changes to the operational software. This cost can exceed the cost of supporting the required personnel over ten years or more. Also in this category is the cost of the facilities required to house the personnel and equipment. Once the hardware, software and facilities are acquired, some expense is realized in operation and maintenance. These costs also need to be accounted for. Reasonable data on these costs can probably be fairly readily obtained from the ALCs.

The analysis and verification/validation requirements, which affect individual changes, also need to be understood in the aggregate in order to predict the requirements for support hardware and software, and for flight test. Those requirements establish, in a sense, a minimum level of necessary resources. It may be (and probably is) desirable to increase those levels in order to raise the productivity of the support center. The interesting question is, what is the optimum level of those resources?

Documentation

Program documentation is another key resource which can affect the cost of supporting operational software. Documentation includes program listings, system specifications, flowcharts, coding and algorithm rationale, and any other paperwork which makes the task of implementing changes simpler. This documentation needs to be acquired from the software developers, and then changed to reflect the current configuration of the operational software as that is changed. A related cost is whether Technical Order pages need to be changed as a result of changing software. Data on documentation costs (especially the cost of acquiring program documentation) may be difficult to obtain.

Aircraft Flight Test

The final cost element to be considered is flight test. Typically a dedicated, specially-instrumented operational aircraft is used to check out both software and hardware changes. The cost of acquiring and instrumenting this aircraft may or may not be considered in performing software support cost analysis. However, the cost of flying time to check out software changes should certainly be included. Related to this is the cost of range time - those expenses related to utilizing an instrumented test range.

The key intermediate variables affecting flight test expense (and initial aircraft acquisition) are hours of aircraft time, hours of test range time, and cost per hour for each. Data on these parameters are fairly easily obtained.

As in the case of support hardware, the V & V requirements are the primary driver conditioning how much flight test is required.

CONCLUSIONS

Analysis of the kinds of historical data available revealed the following:

* There is generally reasonably good manhour data available, although some of it may require manual search of project files.

* Reasonable estimates of personnel training requirements are available.

* Acquisition costs of support hardware are generally available. Acquisition costs of support software may be difficult to ascertain; in many cases much of that software is developed by on-site personnel as part of setting up the AISF and developing a cadre of trained personnel.

AD-A088 476 HUGHES AIRCRAFT CO CANOGA PARK CALIF F/G 14/1
PREDICTIVE SOFTWARE COST MODEL STUDY. VOLUME I. FINAL TECHNICAL--ETC(U)
JUN 80 R B WAINA, G L FOREMAN, A P BANGS F33615-79-C-1734
UNCLASSIFIED AFWAL-TR-80-1056-VOL-1 NL

2 of 2
AD A
088476

END

* The maintenance costs of the support hardware are readily available when that maintenance is done on contract. If it is done by organic personnel it is probably buried in the personnel cost.

* The cost of acquiring documentation on the operational software is often buried in the cost of acquiring the software. Documentation cost may therefore be difficult to ascertain, especially since it is also often difficult to segregate the software cost from the cost of acquiring the operational hardware.

* Standard factors on cost of flying aircraft are readily available. The cost per flight hour should be similar for instrumented test aircraft.

* Cost of facilities was not pursued. Standard cost factors are probably readily available.

Preliminary analysis of the data revealed that there appears to be some kind of regular relationship between number of changes implemented and manhours expended. For the A-7D and F/FB-111 aircraft this was on the order of 1000 hours per change, with a range of 700-1600 hours. If the number of changes can be forecast with any degree of confidence, and if the change mix (i.e., proportion of small, medium and large changes) does not alter drastically, there is a good probability of being able to predict manhour requirements.

Aircraft flying time in support testing software changes did not reveal a regular pattern.

Software package "quality" did not vary significantly by aircraft, when computed using the crude index of averaged attribute quality ratings.

Experienced software engineers at the ALCs identified a number of factors they consider important in projecting support costs. These factors need to be considered in the Phase II model development, along with in-depth study of the factors affecting manhour expenditures, factors affecting aircraft flight test requirements, and the impact of software package quality.

VII. FEASIBILITY OF ESTIMATING SOFTWARE SUPPORT COSTS

INTRODUCTION

Three elements are necessary in order to be able to successfully estimate avionics embedded software support costs:

1) An understanding of the process by which those costs are generated,

2) The availability of historical data to establish the numerical parameters and relationships describing the cost generation process,

3) The establishment of an estimating approach which relates the following: a) the purpose and desired use of the model, b) the input data available at the time the model is to be used, and c) estimating algorithms which reflect the cost generation process and the available data.

The avionics embedded software support cost generation process is now reasonably well understood qualitatively. This understanding is documented in Section VI.

The availability of various historical data has been studied. Quantitative data regarding manhours, salaries, software changes, support hardware costs, etc are reasonably accessible either through ongoing data recording systems at the various ALCs or by manual search of software project files and inquiry of cognizant cost controlling organizations. Determining an appropriate estimating approach is the subject of this section.

PURPOSE/USE OF DESIRED MODEL

The ultimate objective of the PSCM program is to develop a software support cost estimating model for use in the conceptual phase to accomplish the following (ranked in order of priority):

1) Evaluate software design alternatives (e.g., higher-order versus machine language)

2) Evaluate software support concept alternatives (e.g., in-house versus contractor)

3) Make total software support cost projections for DSARC and preliminary budget planning purposes.

There is no requirement for interfacing this model with any other models.

COST ESTIMATING APPROACHES

Twenty-one software prediction models were described or referenced in the technical literature reviewed. Table 2 on page 18 presents an overview of the cost estimating approach utilized in each of those models. Ten of the models are limited to software development cost; the other eleven models consider software operating and support costs as the primary output or as part of the software life cycle cost.

Cost estimating approaches can be classified into three basic categories:

* Analogy (sideways)

* Element estimate (bottom up)

* Cost estimating relationship (top down)

These three estimating approaches represent the candidate approaches for use in the PSCM to be developed.

Analogy

The *method of analogy* is the most primitive estimating technique. It involves comparing the project under consideration with other similar projects. Scope and complexity factors are used to adjust the baseline costs to develop the estimates. Estimates can be made either by an individual or a group. A group goes through the same process as an individual estimator. They compare the upcoming project to similar past projects in terms of size, complexity, schedule, etc., and develop individual estimates. These individual estimates are then combined into a composite. The simplest and most straightforward way is to compute the mean of the individual estimates. A somewhat more sophisticated approach is to require each individual to develop pessimistic, most likely, and optimistic estimates for the project. These are then combined in the standard PERT equation of

$$E = \frac{o + 2m + p}{4}$$

where E = final estimate

o = optimistic estimate

m = most likely estimate

p = pessimistic estimate

Element Estimate

The element estimate approach is the next level up of sophistication in estimating. The project is analyzed into its component tasks, and each of those tasks is individually estimated. The estimates are then added together to obtain the top-level estimate of resources. Schedule can be estimated by development of a PERT network which defines the critical path. Considerable technical/management analysis and judgement is required in the selection of input parameters for models using the bottom-up cost estimating method.

Cost Estimating Relationship (CER)

The CER approach is the most sophisticated estimating technique. It relies on statistical analyses of historical cost and resource data to develop estimating relationships based on key independent variables such as program size and type, memory fill, schedule constraints, etc. These independent variables are then estimated for the upcoming project, input to the estimating equations, and the resource requirements computed. Most current software cost models are of this type. Few have been well validated.

Hybrid

Many models are a composite of two or all three of the methods described above. For example, the basic structure of a model might be element estimate down to a certain level of detail. The inputs at that level could then be developed either by estimating relationship or analogy. This approach is used in Optimum Repair Level Analysis, where transportation costs are one of the elements. Those costs are computed by inputting the weight of the item and the distance to be shipped (from intermediate to depot maintenance). Those factors are then multiplied by a shipping cost per pound-mile, which was derived by analysis of historical data.

RANKING COST ESTIMATING APPROACHES

Ranking of the three cost estimating approaches (sideways, bottom-up, top-down) by means of an objective methodology is basic to the development of a recommendation for the cost estimating approach to utilize in the PSCM. The ranking methodology employed has to bring into account various requirements and attributes associated with each approach, such as historical data requirements, expected accuracy, etc.

A methodology that provides the capability to quantify attributes of the three alternatives and lead to their objective ranking is the Simple Multi-Attribute Ranking Technique (SMART) described in Logistics Spectrum (Winter 1978 - Summer 1979 issues). SMART, as utilized for ranking of the cost estimating approaches, consists of seven steps:

1) Identify the alternate entities to be evaluated (the three cost estimation approaches in this case)

2) Identify the relevant dimensions of value (evaluation criteria)

3) Rank the relevant dimensions/criteria in order of importance and assign numerical weighting values/percentages.

4) Develop a natural scale (low-high, difficult-easy, etc.) for each dimension/criterion and assess the performance the alternatives in terms of those scales.

5) Develop a utility curve for each dimension/criterion using the natural scale for the X-axis and a 0-100 utility scale for the Y-axis.

6) Obtain utility scores on each criteria from the Y-axis of the utility curves as a function of the predetermined location of the alternative on the X-axis.

7) Calculate overall scores for each alternative by summing the products of the utility values (step 7) for each dimension and the weighting percentage (step 3)

A four-member panel of experienced Hughes analysts provided the ranking for the three estimating approaches using the SMART approach. Each panel member provided the following:

* Identification, ranking and weighting for evaluation criteria, given AFWAL/AA's recommendations as a starting point

* Utility curve recommendations for each dimension

* Assessment of performance for each alternative in each dimension

Objectivity is provided with the SMART approach by combining the subjective assessments of knowledgeable persons regarding performance in terms of each dimension. These assessments are summed together to provide scores for the alternatives which are objective in that they represent no single panel member's judgements. The results of the panel's analysis are provided in subsequent paragraphs as rationale for the PSCM cost estimation approach recommendation.

<u>Evaluation Criteria</u>

Evaluation criteria considered for use in the ranking of the PSCM cost estimating approaches are identified in Table 27. The criteria and the associated weights in Table 27 are AFWAL/AA's recommendations regarding model evaluation.

TABLE 27. PSCM EVALUATION CRITERIA SUGGESTED BY AFWAL/AA

Criterion	Weight	AFWAL/AA Rationale
Ease of Development		
Data Required Analysis Required Programming Required	Low	AFWAL/AA will pay to do it right but encourages savings wherever possible
Verifiability	High	SOW requirement
Expandability	Low	Model mainly designed for avionics software maintained by AFLC
Understandability		
Algorithm Acceptability	Low-Med	This is really a fallout of verifiability
Ability to Handle Uncertainty	Medium	Model must be designed for parameters usually available during conceptual phase
Inclusion of Relevant Factors	High	See "ability to handle uncertainty"
Range of Applicability	High	Eventually should cover all AF avionics software
Achievement of Purpose		
Design Evaluation	High	Want to find out effects of development activities (examples: HOL, structured design, V&V, etc.) on operation/support costs
Maintenance Policy Evaluation	Med-High	May be used to evaluate possible changes in policy; also, policy affects costs
Budget Planning/Accuracy	Very High	Model must be able to give as accurate an estimate of operation/support costs as possible
Model Interface	Very Low	AFWAL/AA can get development costs (if necessary) from PRICE-S
Ease of Use		
Input Data Required	High	It's very important that input data required is data easily obtainable, and not overwhelming in volume, yet sufficient for accurate estimates
Computation Cost	Low	Model will be on ASD computer and free to AFWAL/AA
Output Understandability	High	Obvious reasons
Usability	High	Model must be interactive and easy for anyone including non-programmers to use (like PRICE-S)

TABLE 28. WEIGHTING OF EVALUATION CRITERIA

CRITERION	RANK	WEIGHTING VALUE*
Data required for development	Very High	80 (.176)
Design evaluation capability	Very High	80 (.176)
Input data required for use	High	40 (.088)
Verifiability	High	40 (.088)
Relevant factors inclusion	High	40 (.088)
Range of applicability	High	40 (.088)
Cost projection accuracy	High	40 (.088)
Support policy evaluation	Medium-High	30 (.066)
Ability to handle uncertainty	Medium	20 (.044)
Algorithm understandability	Medium-Low	15 (.033)
Analysis required	Low	10 (.022)
Programming required	Low	10 (.022)
Expandability	Low	10 (.022)
		455 (1.00)

*Lowest ranked criterion assigned value of 10. Each higher ranked criterion is then assigned a value indicating its importance relative to the least important criterion. The numbers in parentheses are the normalized weights used in calculating weighted average scores for each approach.

Evaluation criteria/dimensions, ranking, and relative weighting values used with SMART for ranking purposes are presented in Table 28. Criteria utilization and weighting shown in Table 28 follow AFWAL/AA recommendations, with three exceptions. "Data required" has been assigned a "very high" weight versus the "low" weight recommended by AFWAL/AA. Secondly, the ranking of budget planning accuracy and design evaluation capability were reversed, in accordance with later communications with AFWAL/AA. Also, four criteria identified in Table 27 are not utilized for ranking purposes and do not appear in Table 28.

"Data required for development" was assessed to be a "very high" ranked criterion because historical data is crucial to the development of either a sideways or top-down PSCM. Non-availability of a large amount of appropriate historical data could severely limit or preclude the development of these approaches. Appropriateness is a mixture of heterogeneity, in that data apply to many different types of software, and homogeneity, in that it summarizes costs across the software range in a consistent manner. The amount of quantitative data available is less critical for development of a bottom-up model.

The "very high" ranking of "design evaluation capability," with "budget planning accuracy" ranked only "high," is consistent with AFAL's stated purposes for the PSCM. Design evaluation capability has been identified as the top priority purpose, with budget planning accuracy a secondary priority.

Four criteria suggested by AFWAL/AA, including model interface, computation cost, output understandability, and useability were not used for approach ranking. These attributes are primarily functions of computer hardware and program sophistication and are essentially independent of the basic cost estimating approach implemented in the PSCM. They will be used as considerations in detailed model development.

Performance Assesment

The performance of each cost estimating approach was assessed in terms of a natural scale or continuum for each of the thirteen criteria/dimensions of interest. The performance judgements provided by the evaluation panel are presented in Table 29. Each performance judgement represents the average of the judgments provided by the panel members.

Utility Value Assessments

Each performance assessment tabulated in Table 29 was converted to a utility value. Four basic utility curves were chosen for scoring purposes (linear and curvilinear with positive and negative slopes). Figure 16 illustrates these curve types. The "linear-positive" curve represents a linear relationship with the particular parameter: a low value yields

TABLE 29. COST ESTIMATING APPROACH PERFORMANCE ASSESSMENTS

Evaluation Criteria/Dimensions	Bottom-Up Approach	Sideways Approach	Top-Down Approach
Data required for development	Low	Medium	High
Design evaluation capability	Basically-Unlimited	Extensive	Some Limitations
Verifiability	Difficult	Very Difficult	Very Difficult
Relevant factors inclusion	Comprehensive	Limited	Limited
Range of applicability	Extensive-Unlimited	Extensive	Some Limitations
Budget planning accuracy	Medium-High	Medium	Medium-High
Input data required for use	High	Medium	Low
Maintenance policy evaluation capability	Basically-Unlimited	Some Limitations	Some Limitations
Ability to handle uncertainty	Very High	High	Low
Algorithm understandability	Easy	Very Easy	Some Difficulty
Analysis required for development	Limited-Extensive	Minimal Limited	Limited Extensive
Programming required for development	Limited-Extensive	Minimal-Limited	Minimal-Limited
Expandability	Extensive	Some Limitations	Some Limitations

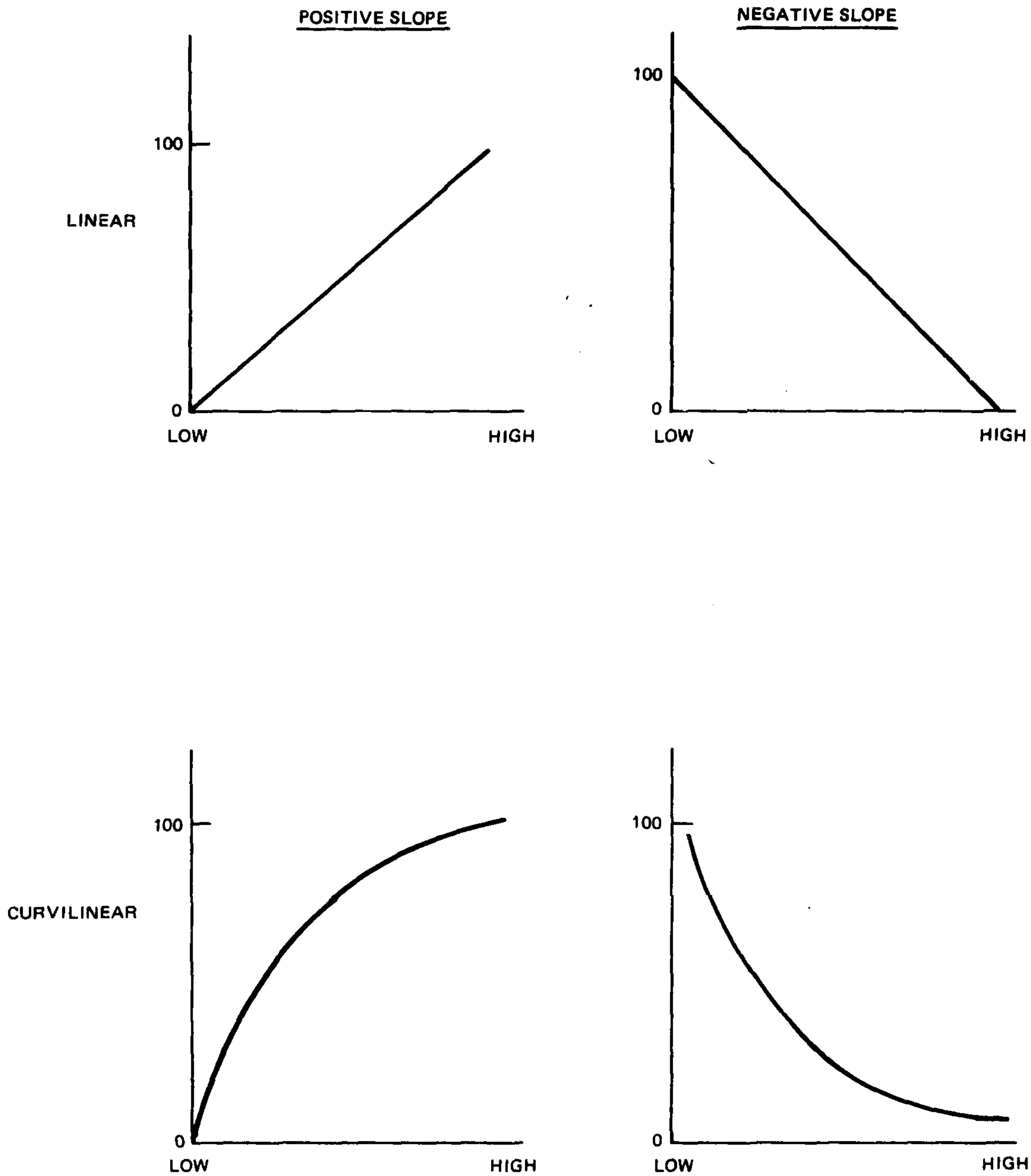


Figure 16. Basic Utility Curves

low utility, medium value yields medium utility, high value yields high utility. The "linear-negative" curve is the reverse.

The curvilinear relationship reflects the law of diminishing returns. Past a certain point, increased magnitude doesn't yield a proportionate increase in utility; below that point, utility rapidly drops off. The "curvilinear-negative" is the reverse.

The actual curves, parameter assessments and utilities of the thirteen criteria are illustrated in Figure 17. "Data required for development," for example, was scored using a "linear-negative" curve. Alternatives requiring more data to develop received lower utility values than those requiring less data. Since the top-down approach (coded as "T") requires the most data (see assessments in Table 28), it received the lowest utility value.

On the other hand, the top-down approach requires less input data to use than either the bottom-up or sideways approach. Since the curve used for that criterion was the "curvilinear-negative," the top-down approach receives the highest utility value on that dimension.

Weighted Average Scores of Approaches

The weighted average utility scores for the three cost estimating approaches are computed as shown below. Table 30 contains the details of the computations.

$$\text{Score}_j = \sum_i W_i U_{ij} \quad \text{where}$$

- i = evaluation criterion
- j = alternate approach
- W_i = weight of ith criterion (see Table 28)
- U_{ij} = utility value of jth approach on ith criterion (see Table 29, Figure 17, and Table 30)

The final results are as follows:

Cost Estimating Approach	Average Score
Bottom-up	72.88
Sideways	55.02
Top-down	50.82

The top-down and sideways approaches appear as the least attractive for use in the PSCM, based on their significantly lower scores in the evaluation. As indicated by the relative closeness of their scores, however, the top-down and sideways approaches could possibly be utilized with equal effectiveness in developing a PSCM.

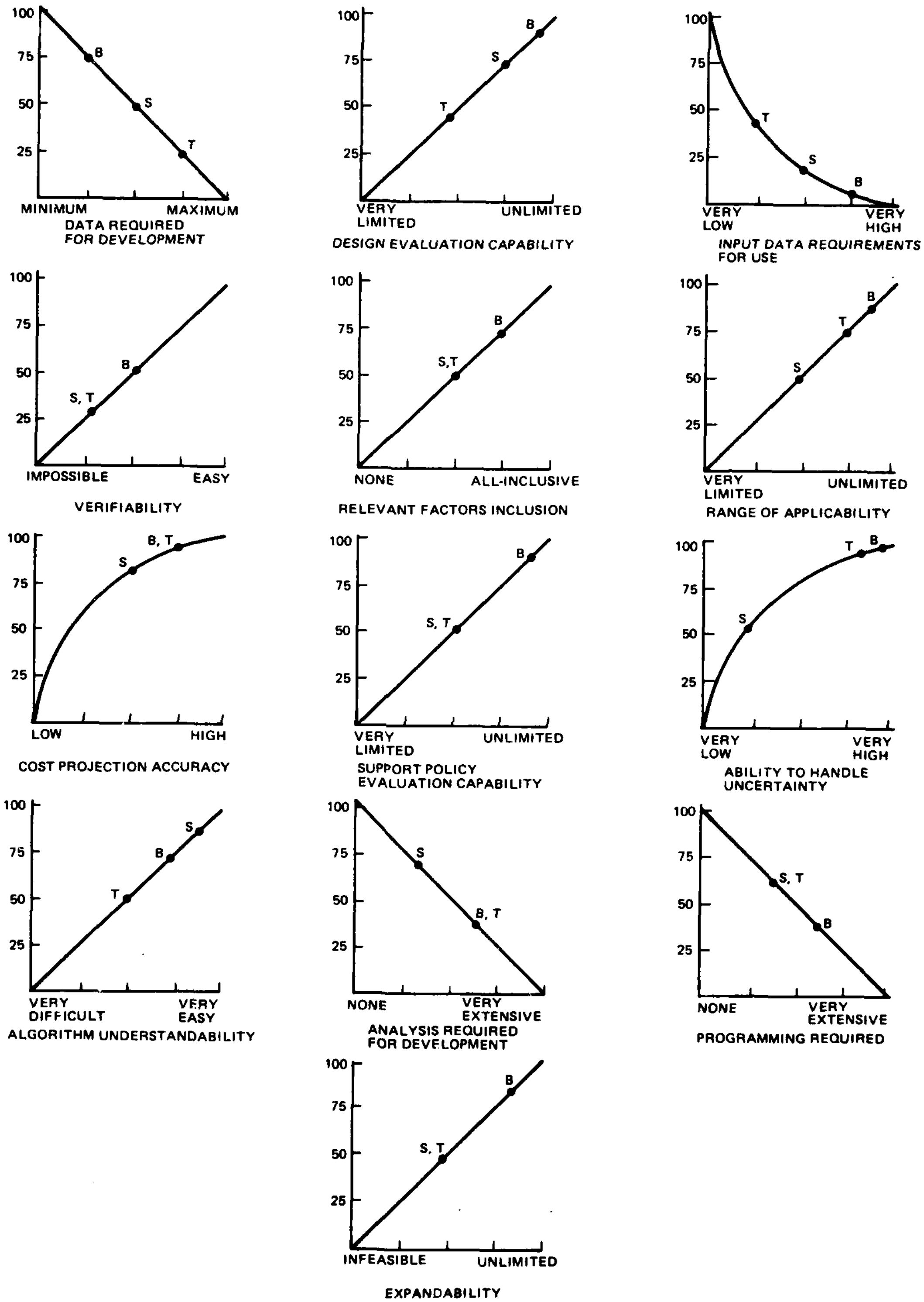


Figure 17. Utility Assessments for Evaluation Criteria

TABLE 30. COST ESTIMATING APPROACH EVALUATION

Evaluation Criteria	Assigned Weight	Bottom-Up Approach		Sideways Approach		Top-Down Approach	
		Value	Score	Value	Score	Value	Score
Data required for development	.176	75	13.20	50	8.88	25	4.40
Design evaluation capability	.176	90	15.84	75	13.20	50	8.80
Verifiability	.088	50	4.40	25	2.20	25	2.20
Relevant factors inclusion	.088	75	6.60	50	4.40	50	4.40
Range of applicability	.088	90	7.92	50	4.40	75	6.60
Budget planning accuracy	.088	90	7.92	80	7.04	90	7.92
Input data requirements for use	.088	10	.88	25	2.20	50	4.40
Support policy evaluation capability	.066	90	5.94	50	3.30	50	3.30
Ability to handle uncertanity	.044	90	3.96	60	2.64	85	3.74
Algorithm understandability	.033	75	2.48	85	2.80	50	1.65
Analysis required for development	.022	40	0.88	65	1.43	40	0.88
Programming required	.022	40	0.88	65	1.43	65	1.43
Expandability	.022	90	1.98	50	1.10	50	1.10
TOTAL WEIGHTED AVERAGE SCORES			72.88		55.02		50.82

The bottom-up approach scored considerably higher than the sideways and top-down approaches. Taken individually, the bottom-up approach scored highest for eight of the thirteen dimensions, with significantly higher scores achieved for five dimensions:

* Data required for model development

* Design evaluation capability

* Verifiability

* Relevant factors inclusion

* Maintenance policy evaluation

The bottom-up approach scored significantly lower than the sideways or top-down approaches on only one dimension; input data required for utilization. A normal bottom-up model requires significantly more input data than the other two approaches. This disadvantage can be overcome through good model design.

The scoring results reflect the general opinion of the evaluation panel that, due to the limited availability of appropriate historical data on which to build a model, selection of the bottom-up approach will result in the fewest design, development, and verification constraints for the PSCM. The sideways and top-down approaches are considered to require much more data for development of the estimating relationships and algorithms than are currently available.

CONCLUSIONS

The bottom-up approach using subtask resource/cost estimates summed to produce the total support cost estimate represents the most practical single cost estimating approach on which to base the PSCM. The bottom-up approach has the advantage of making good use of the limited quantitative and qualitative data currently obtainable, with the potential for extensive refinement as more and better data become available.

A bottom-up model could be structured basically as described in Figure 15 on page 78 with the various initial and recurring support cost estimates computed from a minimal set of input data describing the proposed software design. Other input data values could be supplied as default values from the data base supporting the model. Those default values could be a function of system type, application, etc.

The bottom-up approach can best model the avionics embedded software support cost generation process. Insufficient data are available to adequately support a top-down approach which could be well-validated and accepted by the model user community.

VIII. DEFINITION OF PHASE II ROADMAP

GENERAL

Development of a model that will effectively predict software support costs requires implementation of a logical, well-defined, yet flexible modelling approach. The approach must build upon a combination of what is... and what should be... in the world of software support. In other words, the approach must be responsive to and reflect existing software processes and achieved software performance in the real world, yet be sensitive to the positive and negative lessons learned and the advanced concepts which will influence the future software support scenario.

To this end, as a result of efforts during Phase I of the PSCM study program, a modelling approach has been selected which blends classical model development with refinements specifically applicable to avionics systems. The refinements are the result of the Phase I literature search and data collection and analysis tasks.

The selected approach is embodied in the cost model development methodology overview presented in Figure 18. The portrayed methodology illustrates, in simplistic form, five fundamental tasks required in Phase II to develop and deliver the predictive software cost model. The tasks, i.e.,

* collect data,
* analyze data,
* develop data base,
* design PSC model, and
* test PSC model,

are the subjects of subsequent major paragraphs of this report section, and are individually developed and discussed therein.

The methodology, basically, defines two related parallel efforts, one for design of the predictive software cost model itself, and the other for development of the comprehensive historical data base upon which the model design will be based. Preliminary data organization and analyses will provide an assessment of total data base requirements as well as "working files" for parametric relationship determination and algorithm development. From there, development efforts for the operational data base and the operational PSC model will run parallel paths until the loop is again closed during the model test task. There the model will be tested to demonstrate that it properly reflects its derivation from intelligence contained in the data base as well as its software cost prediction capability.

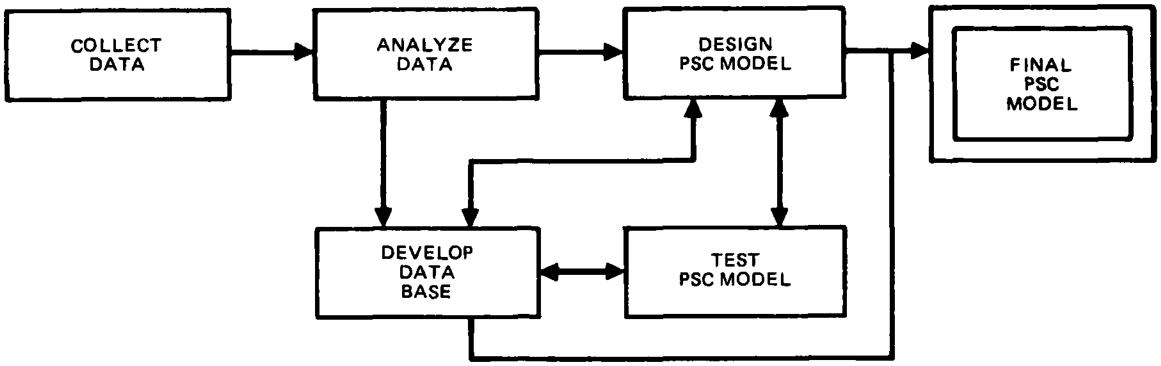


Figure 18. Model Development Methodology.

An expansion of the methodology overview is presented in the model development roadmap of Figure 19. This roadmap illustrates a more detailed task flow, showing relative sequences, interrelationships, and the iterative nature of the subtasks.

The intent of the model development approach, as portrayed in the roadmap, is to:

* conduct an organized data collection effort which will bring together all currently and feasibly available pertinent information concerning the software support world, including conceptual as well as process and performance data,

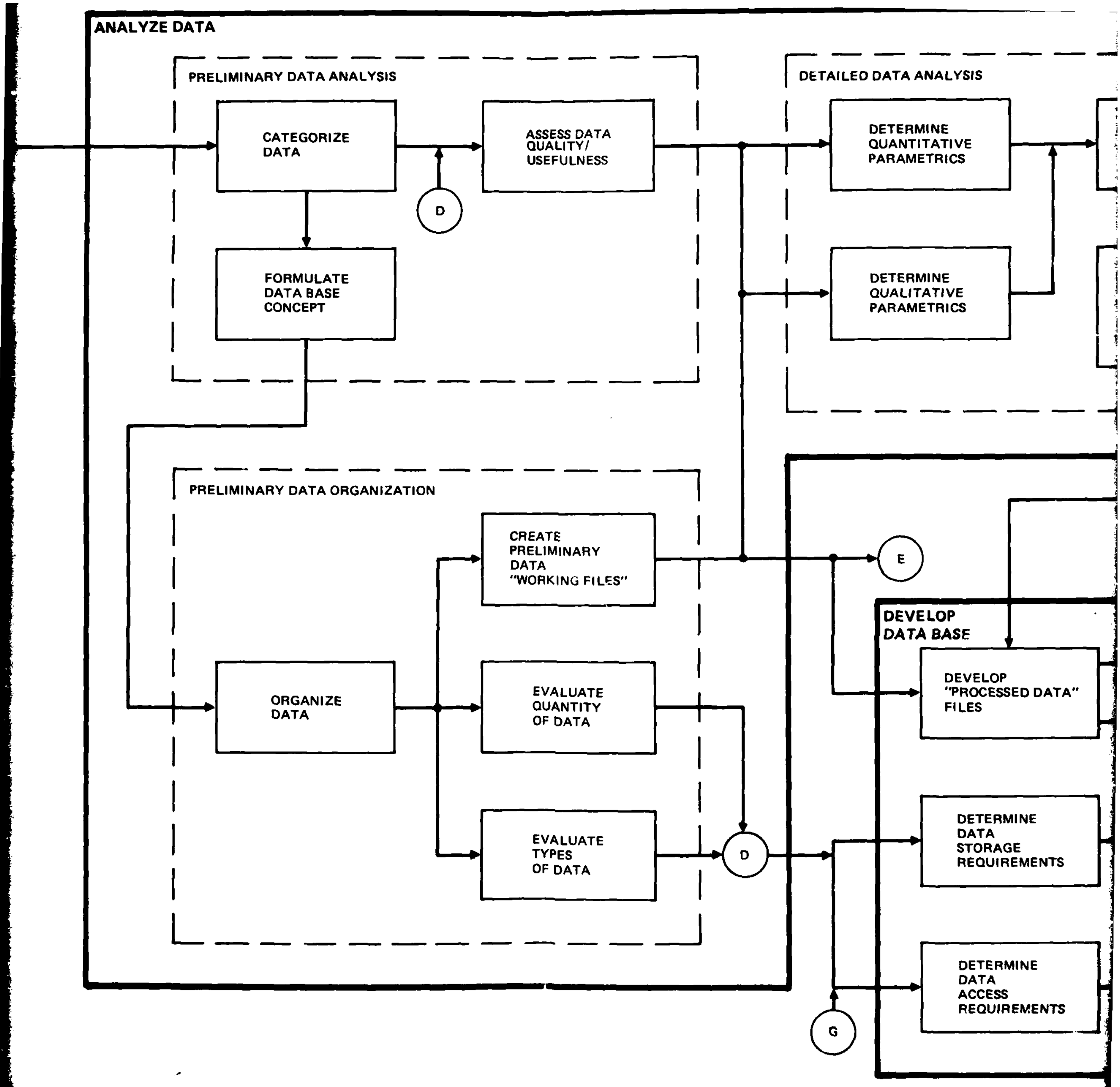
* glean from conceptual studies how software should be supported in the ideal or typical situation,

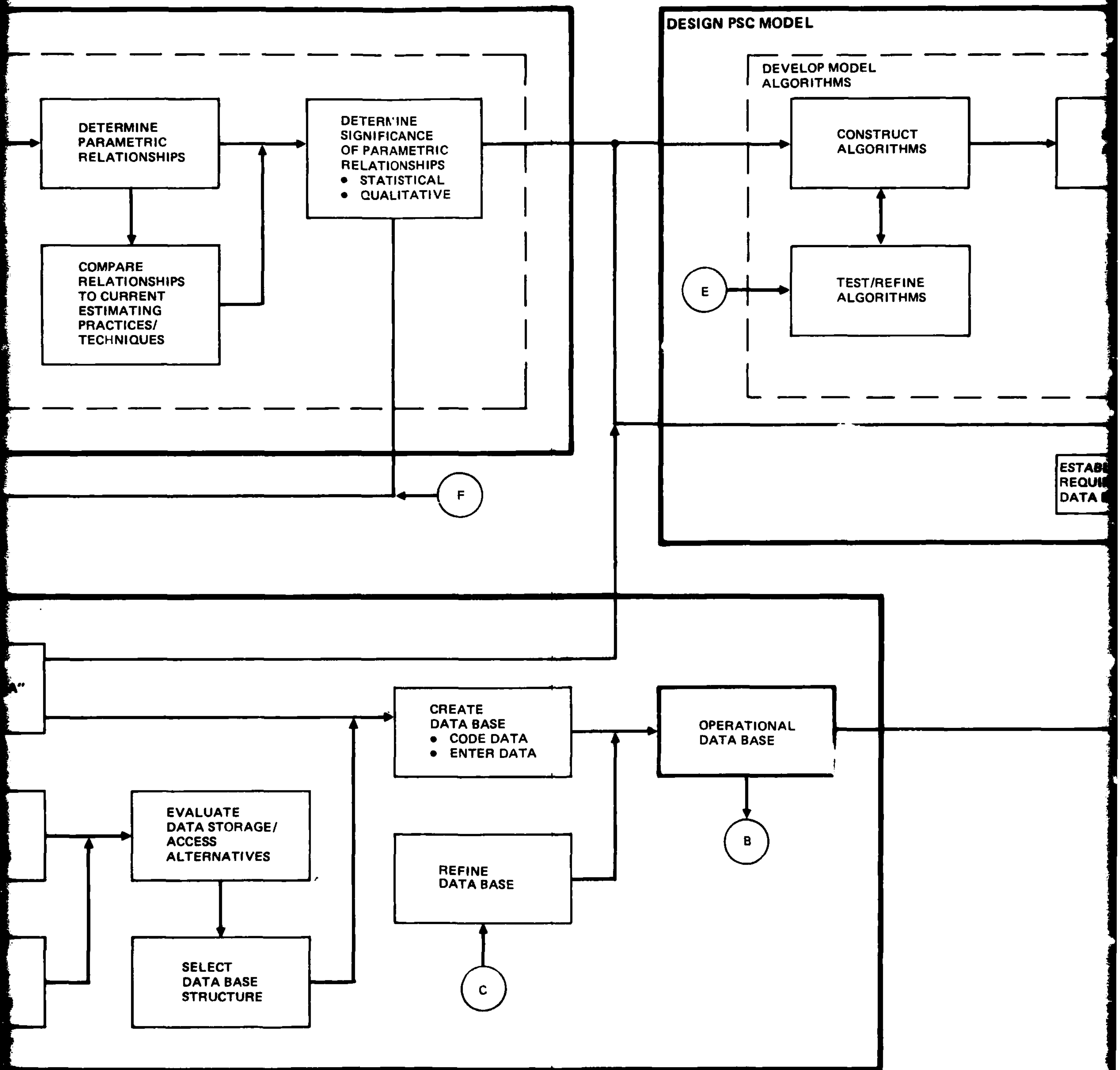
* understand the software support and cost estimating processes that are applied in the real world in accordance with, or in spite of, regulations and procedures for their application,

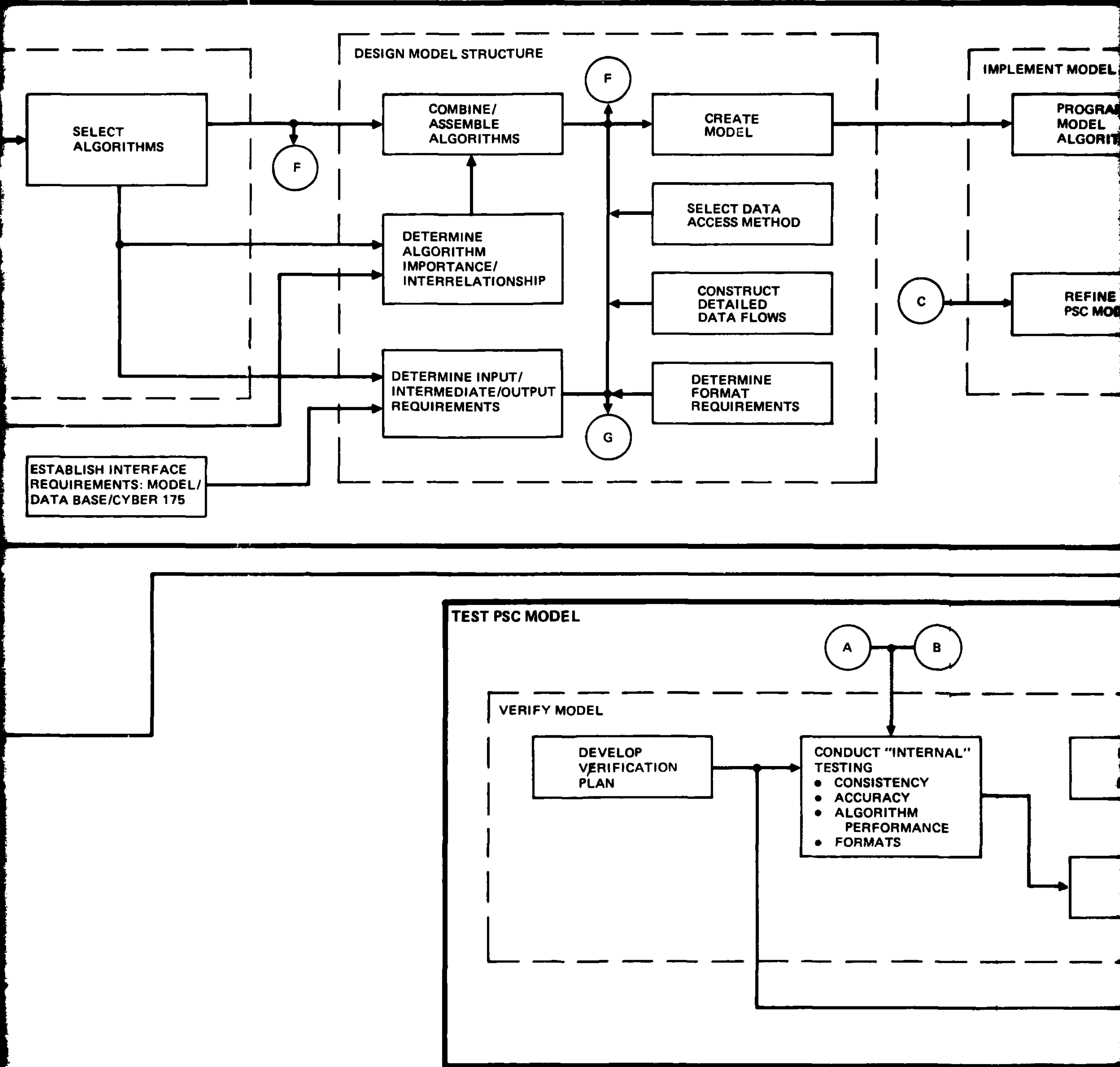
* perform analyses of end results of the software support processes, i.e., the historical cost element performance data,

* benefit from hard lessons learned, both good and bad, from past software support experience,

* consolidate this knowledge and information in a comprehensive qualitative and quantitative data base which can readily be updated and accessed for model development and refinement,


ANALYZE DATA
PRELIMINARY DATA ANALYSIS
CATEGORIZE DATA
ASSESS DATA QUALITY/ USEFULNESS
D
FORMULATE DATA BASE CONCEPT
DETAILED DATA ANALYSIS
DETERMINE QUANTITATIVE PARAMETRICS
DETERMINE QUALITATIVE PARAMETRICS
PRELIMINARY DATA ORGANIZATION
CREATE PRELIMINARY DATA "WORKING FILES"
E
ORGANIZE DATA
EVALUATE QUANTITY OF DATA
EVALUATE TYPES OF DATA
D
DEVELOP DATA BASE
DEVELOP "PROCESSED DATA" FILES
DETERMINE DATA STORAGE REQUIREMENTS
DETERMINE DATA ACCESS REQUIREMENTS
G


DETERMINE PARAMETRIC RELATIONSHIPS
COMPARE RELATIONSHIPS TO CURRENT ESTIMATING PRACTICES/ TECHNIQUES
DETERMINE SIGNIFICANCE OF PARAMETRIC RELATIONSHIPS
• STATISTICAL
• QUALITATIVE
F
DESIGN PSC MODEL
DEVELOP MODEL ALGORITHMS
CONSTRUCT ALGORITHMS
TEST/REFINE ALGORITHMS
E
CREATE DATA BASE
• CODE DATA
• ENTER DATA
OPERATIONAL DATA BASE
B
EVALUATE DATA STORAGE/ ACCESS ALTERNATIVES
SELECT DATA BASE STRUCTURE
REFINE DATA BASE
C


DESIGN MODEL STRUCTURE
SELECT ALGORITHMS
F
COMBINE/ ASSEMBLE ALGORITHMS
F
CREATE MODEL
IMPLEMENT MODEL
PROGRA
MODEL
ALGORI
DETERMINE ALGORITHM IMPORTANCE/ INTERRELATIONSHIP
SELECT DATA ACCESS METHOD
CONSTRUCT DETAILED DATA FLOWS
C
REFINE
PSC MO
DETERMINE INPUT/ INTERMEDIATE/OUTPUT REQUIREMENTS
DETERMINE FORMAT REQUIREMENTS
G
ESTABLISH INTERFACE REQUIREMENTS: MODEL/ DATA BASE/CYBER 175
TEST PSC MODEL
A
B
VERIFY MODEL
DEVELOP VERIFICATION PLAN
CONDUCT "INTERNAL" TESTING
• CONSISTENCY
• ACCURACY
• ALGORITHM PERFORMANCE
• FORMATS

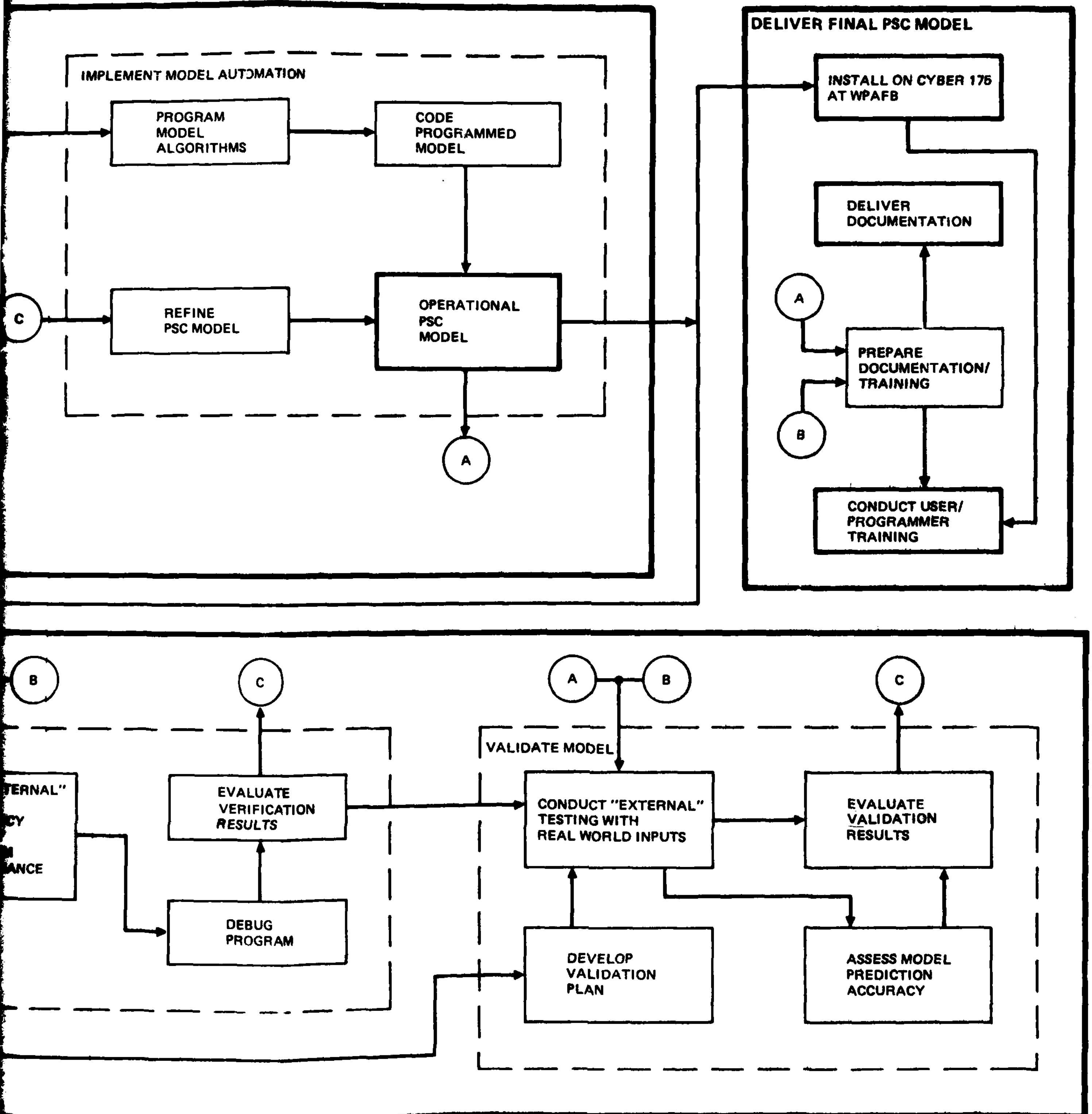


Figure 19. Model Development Roadmap

* build a support cost prediction model, upon intelligence from the composite of information, which can effectively trade off and assess expected software support costs early enough during program software development phases to influence the software design, and

* verify that the model performs as intended with acceptable levels of consistency and accuracy.

COLLECT DATA

The data collection task is the solid foundation which establishes the degree of effectiveness achievable in developing the predictive software cost model. Sophisticated and technically correct analysis and design efforts using inadequate data will provide a rigorous and technically consistent model. A model so constructed, however, will produce only misleading results at best, because the basis for the model construct is insufficient.

Collecting all data that exists in the software support world is not possible... or even necessary. But that which is collected must be (1) comprehensive and complete, (2) representative of the software support universe, (3) possess sufficient quantity and quality to be meaningful, and (4) be pertinent to the task at hand. Satisfaction of these attributes will establish the basis necessary not only for completing a predictive software cost model design, but for achieving a model design that will produce effective and meaningful results. Further, the data collection task in Phase II should be oriented primarily toward collecting the data required to develop a deeper understanding of the key factors identified in Section VI (especially Figure 15).

The objective of this task in Phase II must be to accumulate the best data feasibly available so that the analysis and model design efforts are optimized.

Data Types

The data to be collected should include qualitative as well as quantitative information, categorized under the following data types:

* Conceptual
* Regulatory
* Procedural
* Process
* Performance
* Special

The quantitative data will include primarily historical performance data supplemented by special studies. The qualitative data will focus both on 1) understanding in greater detail the processes of software support and cost estimating in the Air Force, and 2) supplementing the available historical performance data with expert opinion in order to fill quantitative data voids.

Conceptual data is that information which, for this case, reflects what ideal support of software should be like. The advanced concepts set the targets to shoot for in the real world applications. Consideration of this type of information is necessary due to its influence and impact in shaping the future software support situation. How, and to what degree, should conceptual data influence the PSC model design?

Regulatory data consist of official rules of direction, i.e., the various AF regulations that govern software support. What are they? Are they adequate? Are they effective? What influence should they have on the PSC model design?

Procedural data relate to the established step by step order of acting to accomplish the regulations. What are the official procedures? Are they AF-wide or local? How good are they? Do they accomplish the intent of the regulations? Should they influence the PSC model design, and to what degree?

Process data relate to the actions or operations leading to desired results. For this case, these data reflect how the processes of software support really work in the real world because of, or in spite of, the regulations and procedures. How, and to what degree, should this type of information influence the PSC model design?

Performance data describe the end results of the actions and processes. What software support tasks were performed and what were their sizes? What were the task types? When were they performed? What manhours were expended? What other resources were used? What were the costs? etc. These data represent the recorded end results which should provide quantitative parametrics for determining cause/effect relationships and for relating to expenditures as the basis for cost estimating relationships.

Special data relate to additional information not included in the above data types. This should include such things as opinions of experienced practitioners to provide qualitative assessments of non-measureable software performance and other attributes.

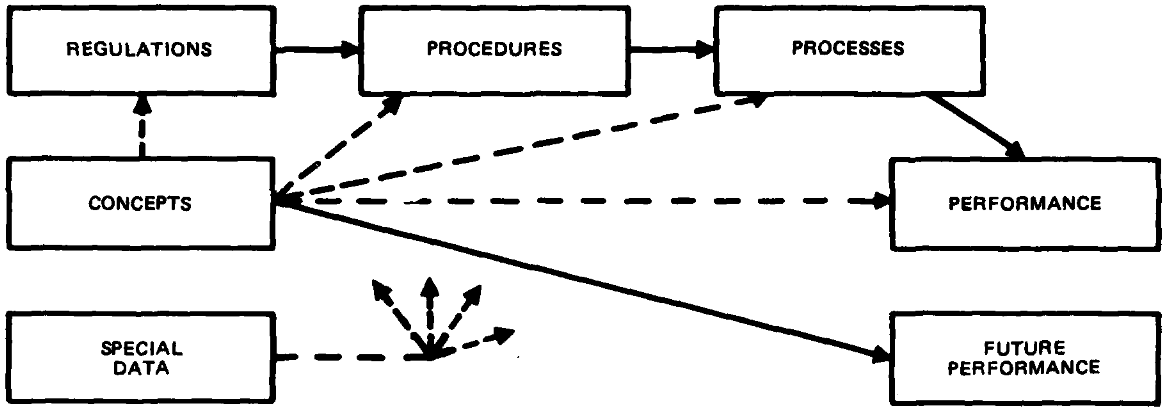


Figure 20. Data Type Relationships

The data from all the above categories, when collected, must provide a basis to resolve, or at least systematically address, the factors necessary to design the PSC model. Figure 20 illustrates the relationship of the data types.

Data Required

A primary purpose of Phase I efforts was to identify and size the data collection requirements for Phase II. The resulting Phase II data collection requirements are summarized in the following paragraphs.

The Phase II data collection task expands upon efforts begun in Phase I. There, the extent and nature of avionics software within the Air Force was determined, and specific relevant sample software packages were identified for the PSCM study program (see Section III).

During Phase I, visits were made to various ALCs to identify the specific program software data sources, and to gain insight into general software maintenance practices and data recording activities within AFLC (see Section V). Preliminary sample data were collected on selected software packages, and specific sources were identified for additional pertinent OFP, EW and ATE data collection in Phase II. Detailed information from the Phase I visits is contained in Volume II of this report.

Several conceptual studies/projects were identified during Phase I (Table 31) which relate to the software world. The Phase II task is to investigate them (and any others identified during Phase II) to determine if and how they may influence the PSCM study and the resultant PSC model design. Their scope may be much broader than that of the PSCM program, but their potential impact must be considered.

TABLE 31. DATA REQUIRED - CONCEPTUAL

Data Type	Source	Comments
Digital Avionics Information System	AFWAL/AA	On-going software development/analysis projects and studies.
Rand Study	Rand	Broader in scope, but may influence/impact PSC model effort
RADC Data Bank	RADC	
Associated studies	Various	

Table 32 identifies primary regulatory information and standards representative of the policies and requirements imposed upon the management, use and support of avionics software. The regulatory data, collected during Phase I, is not necessarily limited to software and computer related resources, but provides direction that encompasses software aspects. The list is not exhaustive, but is meant to include primary top-level regulatory data and to provide a representative cross-section.

Procedural data that implement the regulations and standards are identified in Table 33. Again, the list is representative and not exhaustive. The plans, procedures and operating instructions are those generated and used by the organizations that operate and support the avionics software and associated computer resources. Copies of these documents were collected during Phase I.

Regarding process data, it will be important to increase understanding of the real world processes that lead to establishing software support requirements and providing resultant resources. In this regard, Phase II emphasis should be placed on gaining insight into several key AF processes, i.e.,

* the user and ALC software change requirement establishment and change selection process,
* the AFLC/ASD support resource planning process, and
* the AFWAL/AA cost analysis process.

How does the software user evaluate and establish the needs for changes to the software? How are the requirements transmitted to the supporting ALC? From all stated requirements, how does the user/ALC select those changes to be implemented? Are the selections constrained by budgets or other ceilings imposed on the resources and services allocated for software support? If

TABLE 32. DATA REQUIRED - REGULATORY

DoD Directive 5000.29	Management of Computer Resources in Major Defense Systems
- establishes DoD policy for the management and control of computer resources during the development, acquisition, deployment and support of major defense systems.	
DoD Directive 5000.31	Interim List of DoD Approved High Order Programming Languages (HOL)
- specifies the High Order Programming Languages that are approved for the development of software for new major programs.	
MIL-STD-480	Configuration Control - Engineering Changes, Deviations and Waivers
- sets forth requirements for maintaining configuration control of configuration items; requirements apply to computer software that is designated as a configuration item.	
MIL-STD-483 (USAF)	Configuration Management Practices for Systems, Munitions, and Computer Programs
- establishes uniform configuration management practices that can be tailored to all USAF systems and configuration items, including those procurred by USAF for other agencies.	
MIL-STD-490	Specification Practices
- establishes format and content requirements for program peculiar items, processes and materials; includes requirements for computer program development specifications and computer program product specifications.	
AFR 800-14 Vol I	Management of Computer Resources in Systems
- establishes overall policy for the acquisition and support of computer resources; assigns management responsibilities to HQ USAF, AFSC, AFLC, ATC, Air University and using activities.	
AFR 800-14 Vol II	Acquisition and Support Procedures for Computer Resources in Systems
- contains procedures for implementing the policies included in Volume I.	

TABLE 32. DATA REQUIRED - REGULATORY (Continued)

MIL-STD-1521A (USAF)	Technical Reviews and Audits for Systems, Equipments and Computer Programs
DoD Standard 7935.1-S	Automated Data Systems Documentation Standards
- provides guidelines for the development and revision of the documentation for computer...programs.	
AFLCR 66-27	Automated Support of Automatic Test Equipment Software
- establishes policy, assigns responsibility, and provides procedures pertaining to...requirements for automatic data processing resources when required for the organic preparation, maintenance and management of ATE software.	
AFLCR 66-37	Management of Automated Test Systems
- establishes policies for automatic test system management and defines responsibilities; applicable to AFLC activities associated with management, use, and support of ATE hardware and software; used in conjunction with AFLCR 66-27 to provide policies and procedures for support of automatic test system software.	

so, how are the constraints set? Inherent in understanding the change selection process is understanding how individual software change criticalities, complexities and costs are established.

How does the AFLC/ASD support resource planning process operate? Traditionally, software support resource allocations are determined early in the software development phase. How is this currently accomplished? These allocations are then used to select (constrain?) the individual software changes to be implemented as part of software support. How do the ALCs influence the allocations of software support resources?

What does AFWAL/AA currently use as the basis for their program cost analyses, especially as they relate to software support? How does the cost analysis process work?

Do these planning and resource establishing processes operate in a coordinated manner, or are they basically independent of one another? How formal are they? Do they differ among programs? Do they follow regulations and procedures?

TABLE 33. DATA REQUIRED - PROCEDURAL

Document	Source	Title
F-15 CRISP	SPO	F-15 Avionics Software Computer Resources Integrated Support Plan
F-16 CRISP (F-16-1001/I/2)	SPO	F-16 Multinational Computer Resources Integrated Support Plan
Planning Guide	SA-ALC	Automatic Test Equipment Acquisition Planning Guide
CM Plan	OO-ALC	OFP Configuration Management Plan
MMOI 800-2	SA-ALC	Preparation and Use of AFLC Form 75, Computer Program Configuration Sub Board Item Record
MMOI 800-14	WR-ALC	Acquisition, Management, and Support of Computer Resources in Systems (Software and Associated Equipment)
MMROI 800-01	WR-ALC	Software Change Processing/ Configuration Management for EW Systems
MMROI 800-03	WR-ALC	System Verification Test Procedures using Simulation and Analysis Test Systems
MMECOI 65-2	WR-ALC	Configuration Management for Avionics Integration Support Facilities
MMECOI 800-14	WR-ALC	Software Change Processing Procedures for Operational Flight Programs
O/S CMP (F/FB-111)	SM-ALC	Operational/Support Configuration Management Procedures for F/FB-111 Operational Software
O/S CMP (F-16)	OO-ALC	F-16 Multinational Operational/Support Configuration Management Procedures

TABLE 33. DATA REQUIRED - PROCEDURAL (Continued)

Document	Source	Title
O/S CMP (ATE)	SA-ALC	Operational/Support Configuration Management Procedures for ATE Software
Work Units	WR-ALC	Computer Resources (MMEC) Work Center-Time Standard Descriptions
Testing Guide	WR-ALC	Software Testing Guideline (Preliminary)

Regarding the AF processes, these are the types of questions that need to be answered. The properties related to software support must be identified and, if possible, quantified for consideration in the PSC model development. Understanding these processes will require extensive interfacing with the AF managers and analysts directly involved with their conduct.

Available performance data are identified in Table 34, and required performance relationship data are identified in Table 35. Since avionics software support is a fairly new AFLC responsibility, software support process and performance data are not currently well documented, but are developing out of necessity. As discussed in Section V, the A-7 and F/FB-111 family of software packages are the only OFP software systems which can currently provide actual support resource requirement and cost data within AFLC. The F-15 and F-16 software packages will be excellent representative data sources in the future. WR-ALC has begun to work on F-15 OFP changes, and should have some data available in the 1981 time frame. 00-ALC is currently manned to organically support three F-16 OFPs, but is only performing V&V of contractor changes at this time. Again, they should have some data available in 1981. However, the quantity of hard performance data available for estimating relationship analysis and development during Phase II may be somewhat limited.

In any event, that performance data which was collected during Phase I must be updated and supplemented to provide as complete a package as possible on all targeted systems. These data will comprise the primary bases for establishing estimating relationships for model development.

TABLE 34. REQUIRED DATA - PERFORMANCE

Data Type	System					
	A-7D	F/FB-111	F-15	F-16	EW	ATE
Number of Changes	X	X	X	X	X	X
Manhours	Aggregate	X	?	V&V only	X	X
Personnel Salary and Overhead	X	X	X	X	X	X
Facility Size	X	X	X	X	?	?
Facility Cost	?	?	?	?	?	?
Support Hardware Cost	X	X	X	X	X	?
Support Software Cost (Acquisition)	?	?	?	?	?	?
Documentation Cost (Acquisition)	?	?	?	?	?	?
<u>Flight Test:</u>						
Aircraft Cost (Acquisition)	X	X	X	X	X	n/a
Flight Hours (FH)	X	X	?	X	X	n/a
Cost/FH	X	X	X	X	X	n/a
Range Hours (RH)	?	?	?	?	?	n/a
Cost/RH	?	?	?	?	?	n/a
Data Source(s)	OC-ALC MMECZA	SM-ALC MMECP	WR-ALC MMEC	00-ALC MMEC MMET ACDCS	WR-ALC MMRR	SA-ALC Various WR-ALC MMECT

Legend: X = available
? = possibly available
n/a = not applicable

TABLE 35. DATA REQUIRED - PERFORMANCE RELATIONSHIPS

Change volume/rate
Software reliability
Factors affecting avionics software reliability
Requirements changes
Factors affecting changes
Manhours per change
Productivity
Factors affecting productivity, e.g.,
Support hardware
Support software
Documentation
Personnel experience/training
Incentives
Modifiability
Program structure
Size
Complexity
Architecture
Language
etc.
Hardware characteristics
Memory fill
Timing requirements
Size of change
Application/function affected
Performance requirements
Analysis requirements
V & V requirements
Amount of support hardware
Prime system hardware
Analysis requirements
V & V requirements
Amount of support software
Support hardware
Analysis requirements
V & V requirements
Automation of support process
Hours of Flight test time/range time
V & V requirements
Cost/hr of range test time
Extent of instrumentation
V & V requirements
Efficiency factor (productive manhour/year)
Holiday/sick leave
Administration requirements
Training requirements
etc.

Table 36 identifies types of special data that should be collected in Phase II, if possible. Phase I visits indicated that performing organizations sometimes conduct "pocket" studies and analyses concerning specific software related problems that trouble their operation. They are often accomplished in addition to their normal, routine data recording activities that reflect performance data. Data sought from these sources should be solicited and collected. Also, qualitative information from practitioners should be solicited in areas where no hard data are available.

TABLE 36. SPECIAL DATA

Manhours per type of software change
Relationship of changes to underlying requirements
Programmer productivity
Capacity vs. workload in the software support center
Manhours related to software complexity
Relationship of changes to test requirements
Relationship of changes to program maturity
Manhours related to skill/training levels
Qualitative information

Having completed collection of the required data from all data categories identified above and in the Model Development Roadmap (Figure 19), the Phase II analysis and PSC model design efforts can begin.

ANALYZE DATA

As stressed earlier in this section, the collection of data is considered foundational to the PSC model design effort. While the data analysis task provides the parametric relationships on which to design the PSC model, the parameters, their relationships and their significance are wholly dependent upon the quantity and quality of intelligence contained in the collected data.

The data analysis task, therefore, must ferret out that intelligence, make an assessment of its completeness and quality, and, given that it is adequate, determine the meaningful relationships of the parameters and their significance for use in constructing the PSC model. If, for any reason, the intelligence is determined to be inadequate for any aspect of model development, the data analysis task must identify the area of inadequacy early enough to enable, in coordination with AFWAL/AA, timely collection of additional requisite data. The currently identified data collection task should be fully adequate, however, and the following discussions are based on that assumption.

Refer to the Model Development Roadmap (Figure 19). The primary objectives of the data analysis task should be to:

* uncover the unbiased parametric relationships evidenced in the collected data,

* determine their suitability for use in the PSC model development, both quantitatively and qualitatively,

* use them to substantiate and/or refine the factors and relationships postulated on the basis of Phase I findings (refer to Figure 15 in Section VI), and

* provide them as candidates for development of model algorithms.

In order to accomplish these objectives, the roadmap illustrates that an orderly examination and analysis of the collected data is required. The first step of this process is to organize and categorize the data to perform a preliminary evaluation of the data attributes of comprehensiveness, completeness, representativeness, quantity, quality and usefulnesss. The next step is to create preliminary data "working files" which become the fodder to supply the demand of the detailed data analysis and algorithm development processes, and the basis for development of the comprehensive historical data base. The "working files" are envisioned to consist of both computer based quantitative performance data for easy automated analytical access, and non-computer based information for qualitative considerations.

The final step is to use the "working file" data to perform appropriate statistical and qualitative analyses to identify the most prominent and most significant parameters and their recurring interrelationships for subsequent use in model algorithm development. An interesting branch of this effort will be to compare the unbiased parameter relationships so derived with practices and techniques currently in use, and with other preconceived concepts.

Parameter/Parametric Relationship Determination

Figure 15 on page 78 in Section VI presents a view of the software support cost generation process based on results of Phase I efforts. The Phase II data analysis task should be directed toward supporting, refining and/or modifying the factors and variables and their relationships as illustrated therein. In the figure, the relationships between specified intermediate variables and outputs are fairly well understood, but the influence of the key input variables is not well understood. The analysis task must be oriented toward developing a deeper understanding of all key factors and their relationships. The analysis task must also "look ahead" in an attempt to discern how the software support process may evolve over the next several years in order to assure that other critical factors, perhaps not yet evident, are not left unconsidered.

The main thrust of the data analysis task should center about the quantitative data obtained primarily from the performance data category (Tables 34 and 35), and any additional quantitative data available from the special data category (Table 36). These data should provide the hard core of intelligence from which parametric relationships can be derived using multivariate analysis techniques.

Secondary, but still very important, emphasis should be placed on the qualitative information emanating from the conceptual, regulatory, procedural, process and special data categories. These are the data which provide flavor and fine tuning for the parametric relationships and their significance, and for subsequent model algorithm development and model structure design. Where possible, the qualitative data should be content-analyzed, as part of the data analysis process, to transform the inherently qualitative nature of the information into data that are amenable to numerical manipulations. This will not always be possible.

The overall goal of the data analysis must be to sort through the "working files" to identify the key factors and their interrelationships in an unbiased manner. These factors/interrelationships, then, provide the basis necessary to formulate the PSC model. From a statistical standpoint, the analysis needs can be satisfied using multivariate analysis techniques, of which the following generic areas should be considered:

* factor analysis,

* canonical, partial and multiple correlation analysis, and

* linear discriminant analysis.

Selection of the data analysis technique(s) in Phase II should depend upon the preliminary assessment of attributes of the collected data, especially the quantity and quality. The correlational techniques, which are typically used when the data set contains a set of predictor variables and a set of outcome variables, should comprise the primary analysis tools.

DEVELOP DATA BASE

The operational data base must contain parametric data appropriate for all expected applications of the operational PSC model. The set or sets of parametric data to be used with each model application should be keyed by the input characteristics of the type of software whose support costs are being predicted, and should be automatically accessed by the model for computational use.

Development of the data base should be an evolutionary process starting with the raw quantitative and qualitative data collected during the data collection task, and culminating with an operational data base for use with the PSC model. The data base development process should be an effort parallel (but integrally related) to the data analysis, algorithm development, and model structure design subtasks.

The data base development is envisioned as comprising three basic phases: creating preliminary data "working files," developing "processed data" files, and developing the operational data base. This concept is illustrated in Figure 21.

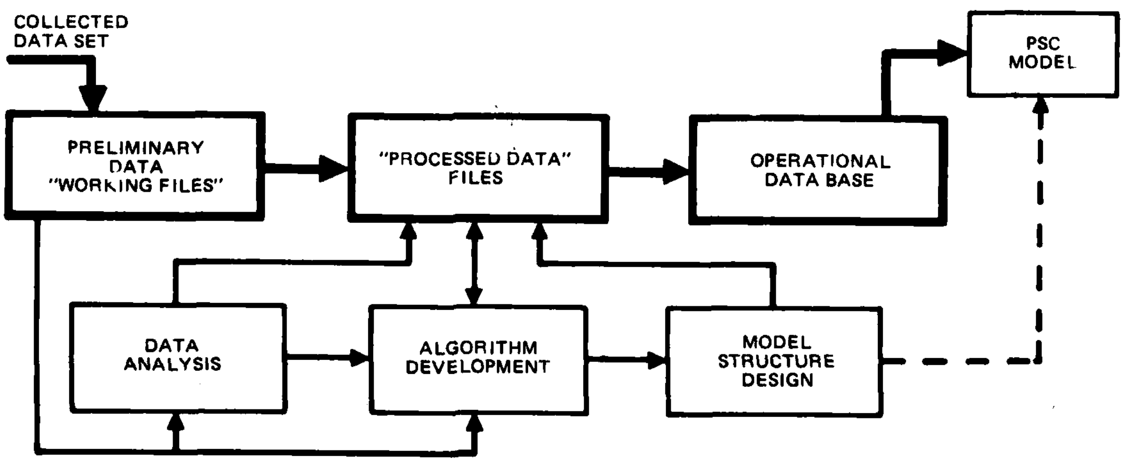


Figure 21. Data Base Development Concept

In order to facilitate explanation of what whould be contained in the "working files," the "processed data" files, and the operational data base, the following hypothetical illustration is offered. The illustration should aid in understanding the concept of the operational data base and its development.

An equation for the PSC model can be expressed in hypothetical form as:

$$(1) \qquad C_{Tj} = \sum_{i=1}^{n} K_{ij} C_i \qquad (j = 1, 2, \ldots, m)$$

where n = number of prediction cost categories,

m = number of software package types,

C_{Tj} = total predicted software support cost for the jth software package type,

C_i = predicted cost of the ith cost category, and

K_{ij} = parametric constant for the ith cost category of the jth software package type.

The first algorithm can then be defined as $K_{1j}C_1$, the second algorithm as $K_{2j}C_2$, etc.

Expanding the illustration, the algorithms become:

Algorithm 1:

$$(2) \qquad K_{1j}C_1 = A_{1j}F_1 + A_{2j}F_2 + \ldots + A_{pj}F_p$$

where

F_p = equations/relationships that comprise cost category 1, and

A_{pj} = cost category 1 parametric constant for the pth relationship of the jth software package type;

Algorithm 2:

$$(3) \qquad K_{2j}C_2 = B_{1j}G_1 + B_{2j}G_2 + \ldots + B_{qj}G_q$$

where

G_q = equations/relationships that comprise cost category 2, and

B_{qj} = cost category 2 parametric constant for the qth relationship of the jth software package type;

and similarly for n algorithms representing all cost categories to be predicted.

Further expanding the illustration, the functions comprising algorithm 1 become, for example:

(4) $F_1 = a_1 f(x) + a_2 f(y) + a_3 f(x,y) + \ldots,$

(5) $F_2 = b_1 f(x,y) + b_2 f(y,z) + b_3 f(z) + \ldots,$

(6) $F_3 = c_1 f(x,y,z) + c_2 f(x,z) + c_3 f(y,z) + \ldots,$

.

.

.

.

$F_p = \ldots,$

where f's = functions of elemental input variable(s), and

a's, b's & c's = parametric constants for the functions (f).

The same type of expansion can be made for algorithms 2 through n.

With this hypothetical illustration as a basis, let us define the concept of the data base development process.

Preliminary Data "Working Files"

These files should contain the categorized and organized raw data elements from the data collection task, both quantitative and qualitative. These files are the elemental values of x, y and z illustrated in equations (4), (5) and (6), and the intelligence that enables determining their relationships.

The "working files" re envisioned as the data elements from which the (F) relationships and the values of the a's, b's and c's are derived in the detailed data analysis subtask. Refer to the roadmap of Figure 19. Some of the values (a's, b's or c's) may be standards, such as $N/flight hour; others could be derived along with the (f) relationships from, for example, multiple correlation analysis of various data elements.

"Processed Data" Files

These files should initially contain the sets of values for a's, b's and c's provided from results of the detailed data analysis subtask. From these values and from additional intelligence in the "working files," the relationships C_i and the values of A's, B's, ..., and K's (equations (1), (2) and (3)) are derived as part of the detailed data analysis and algorithm development subtasks. These values should then be added to the "processed data" files.

The values and/or forms of the A's, B's, ..., and K's represent the relative importance and the interrelationship of the functions that make up the algorithms and of the algorithms that make up the cost categories. Some of the values may be derived by knowledgable, systematic interpretation and application of qualitative data, but all should have traceable bases.

Operational Data Base

Simply, the operational data base should contain all the developed parameter sets from the "processed data" files, properly structured and coded for automatic access and use by the PSC model.

Data Base Concept Summary

The raw data "working files" will be the collected data elements from which parametric relationships (F's, G's, etc.) and parametric constants (a's, b's, c's, etc.) are derived during the detailed data analysis subtask.

The "processed data" files should contain the parametric constants (a's, b's, c's, etc.) from which the cost category relationships (C_i) and the values of their modifiers (A's, B's, ..., and K's) are derived during the detailed data analysis, algorithm development, and model structure design subtasks (refer to the roadmap of Figure 19).

The operational data base will contain the formalized structure of the parametric data sets developed within the processed data files, keyed for automatic access and use by the PSC model.

DESIGN PSC MODEL

The model design effort should be the focal point of the PSCM development program. This effort must formulate the results of the data analysis task, based on the historical data recorded in the PSCM data base, into a computer based PSCM. As shown in Figure 19, model design should encompass three major design subtasks: algorithm development, model structure design, and model automation implementation.

Algorithm Development

Algorithm development starts with the set of parametric relationships derived during the data analysis efforts, shapes them into candidate algorithm forms, and culminates with the selection of the set of algorithms to be used in structuring the software support cost prediction model. This set of algorithms, properly structured, forms the software support cost generation system.

The developed algorithms should reflect and encompass the factors and relationships indicated in Figure 15 in Section VI, dynamically reinforcing/modifying them as the analysis suggests. The illustrated system is a network of complex relationships between three postulated types of data, namely, inputs, intermediate values, and outputs. Inputs refer to the data required by the software designer/engineer early in the program life cycle, characterized as data reflecting prime system hardware, prime system software and requirements changes. Intermediate variables specify such things as the degree of software maintainability and support center productivity, and quantify software and hardware support requirements. Outputs refer to the major cost categories of software support cost.

Algorithms describing the network of data relationships must be constructed from the basic set of parametric relations compiled during the data analysis effort. However, since quantitative information may be limited in some areas, supplementing subjective data reflecting the opinions and suggestions expressed by the software professionals should be used when required. AFLC software professionals can be an invaluable souce of the information needed to quantify and measure qualitative software characteristics.

The algorithms generated during the Phase II study will probably have the form:

INTERMEDIATE VARIABLE = f (INPUT),
or OUTPUT = f (INPUT, INTERMEDIATE VARIABLE).

Algorithm development, then, requires a certain degree of parameter definition standardization and the establishment of parameter quantification methods.

Each newly constructed algorithm must undergo a test and refinement process. Only those algorithms should be considered that are sensitive to the change in direction and magnitude of the independent parameter. The final algorithm form should be the one yielding the best mathematical correlation with minimum error. A concerted effort must be made to assure model completeness while eliminating costing redundancy to provide the set of final algorithms for PSCM implementation. Standard rates and default values needed to compute intermediate variables, cost drivers and cost predictions must be part of the operational data base.

The above approach to algorithm development should result in a model that uses data readily available during the early software life cycle to predict software support costs. In addition, by computing the intermediate variables, the model would allow the software designer to evaluate the proposed software design in terms of software maintainability.

<u>Model Structure Design</u>

Structuring the model design requires determining the relative importance of the selected algorithms and their interrelationships, and combining the algorithms into a complete dynamic mathematical representation of the PSC model. Further, the subtask requires matching the mathematical nature of the model to the target computer characteristics.

The resulting model design should reflect and conform to user-oriented needs which include:

* Self-containment in the sense that minimal user participation is required to operate the model,
* Traceability of all cost estimates,
* Acceptable levels of accuracy,
* Adaptability for a wide range of avionics programs in the sense that changing technology does not necessitate reprogramming, and
* Modifiability to permit rapid and effective updating of algorithms and data input/output.

The set of algorithms resulting from the algorithm development subtask is necessarily the key driver of the model structure design process. Data requirements analyses of individual algorithms, and of the set of algorithms as a unit, must identify all parameters needed to compute the software support cost estimates. This list of parameters must include standard costs and rates and well as parameters which vary from system to system.

Model output reports also impact the model parameter list. These reports must present the computational results in an easily understood format. Parameters must be defined in order to prepare an adequate set of cost reports. These reports should include labeled cost summaries, cost profiles, and optionally, the intermediate variables and the user's input data. The following list of reports illustrates typical basic information a user needs to interpret model cost estimates and perform software design tradeoff analyses:

Input Reports should include a list of the user supplied raw data and the computer interpreted input data set default values (if any). These reports should present the complement of parameter values used by the model such as:

* User's raw input data
* Model default values
* Total input data set used to compute cost estimate
* Ground rules and assumptions

Output Reports should present the results of the model computations. The reports should include model computations in varying degrees of detail, such as:

* Cost summary; total support cost
* Cost estimate for each major algorithm
* Cost profile; total and major algorithms
* Detailed cost estimates
* Intermediate computation results

The combination of input and output reports must provide the required traceability for model calibration and verification.

The parameters needed for computation and report generation should be analyzed and used to generate a PSCM Dictionary. The dictionary could contain parameter definitions and characteristics (for example, $/hour as the units characteristic of a parameter representing labor rate.) The dictionary must ensure compatibility of parameters used in more than one algorithm.

A suggested concept of data flow for the PSCM design is presented in Figure 22. Major data interfaces are highlighted and include those between the data sources (user and PSCM Data Base) and the computer program; between the various program modules (data access and interpretation, cost computations, and report generation); and between the model and the user. Data flow diagrams serve to highlight data requirements and commonality of data between the model components. Detailed data flow diagrams should be generated to aid designing the optimal model structure.

A CYBER 175 computer at Wright-Patterson Air Force Base is the target computer for PSCM use. Since model structure is influenced by the programming languages and data management systems supported by the development and target computers, PSCM development should be accomplished on a target computer type. Developing the PSCM on a CYBER 175 supporting the same programming languages and data management systems should ensure compatibility and minimize the effort required to transport the model from the development computer to the target computer.

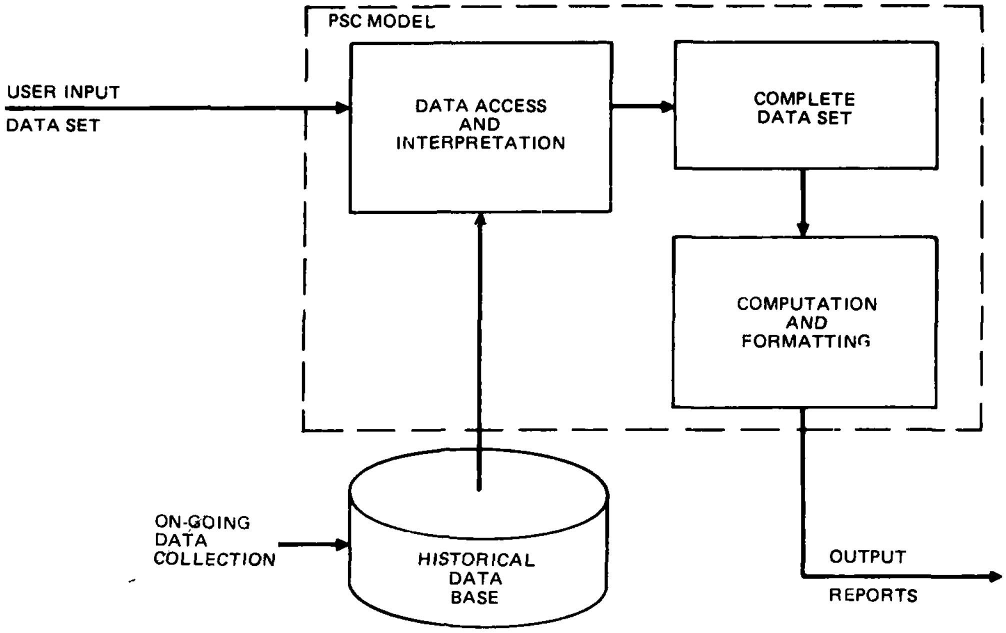


Figure 22. Basic Data Flow

Structure of the algorithms, data requirements and interfaces, programming language, and data access methods should be the basic elements of the model design.

Model Automation Implementation

The implementation subtask requires programming and coding of the mathematical representation of the PSC model. This is the final transition from a "paper" model to a computer-resident model.

Detailed program flowcharts must be developed from the structured model design, and coded using the requisite programming language. The program should then undergo iterative debugging to assure program internal correctness and consistency with respect to the "paper" model.

Corrections and refinements to the model should then be made in accordance with results of verification and validation testing. The process will constitute iterative model testing and re-design until an operational PSC model demonstrating desired results is achieved.

TEST PSC MODEL

The model testing process must play the primary role in establishing PSCM credibility. Model testing should be perfomed in two phases, model verification and model validation. Verification measures the internal consistency of the model, the extent to which it produces expected results based upon known inputs and pre-calculated outputs, and whether it correctly provides the desired output formats. Validation assesses the prediction accuracy of the model in the "real world" using data other than that from which the model was derived, if possible. The verification and validation (V & V) testing should be iterative in nature, providing feedback for refining the design of the operational PSC model and corresponding operational data base. This testing and feedback feature is illustrated in the roadmap of Figure 19.

Model Verification

A PSC model verification plan must be generated which describes the verification testing methodology, provides verification test procedures, details criteria for use in evaluating verification test results, and provides for feedback into the PSC model design.

Inputs to the model for verification testing should be the input set of characteristics which describe one of the specific software packages collected for the study. The PSC model would then automatically interface with the operational data base to select and use those parameter values appropriate for the in-test software package type, and compute the predicted support costs. Results should then be compared with known pre-computed values, and with actual support cost experience for the specific software package. These comparisons should then be evaluated to determine what refinements, if any, need to be made to the model design and to the operational data base. The test should be repeated until the evaluation criteria are satisfied.

The verification test should be applied to each software package type identified and contained in the operational data base.

Model Validation

For model validation, a validation plan must be developed. The plan must be similar to the verification plan, but should provide for inputs from sources other than those used to construct the PSC model and operational data base, i.e., "external" inputs. However, this may not be readily achievable since additional "external" data may not be available. Most of the appropriate relevant data may have been collected and used for determining parameter values and designing the model.

However, one or more additional software packages not in the collected sample should be identified, if possible, for validation testing purposes. The package(s) should to be representative of the package types included in this study.

Having accomplished this, validation testing will be conducted in a manner similar to verification testing, but with the input set of characteristics describing the "external" software package. The model would again select and use the appropriate set of parameter values from the operational data base, and compute the predicted support costs. Again, model results should be compared with actual experience for the "external" software package (to the extent it is available), and refinements should be made to the model as appropriate.

If additional "external" software packages are not available for validation testing, alternate validation testing techniques should be considered.

One alternate approach might be to exercise the model using extreme values in the set of input characteristics to determine model behavior under such conditions. This technique could provide insight into the reasonableness of the model, and a qualitative assessment of confidence to be placed in its computations.

A second alternative might be to develop hypothetical, representative software package characteristics, in cooperation with AFAL, and compute all intermediate and output values for joint evaluation. These values could then be evaluated for "reasonableness" by experienced software engineers within AFLC. While this approach would be highly qualitative in nature, it would provide a measure of whether the model could and should be used for trade study purposes, and whether it provides results which appear to be "correct."

Hopefully, additional "external" data will be available, and alternate verification approaches will not be required.

The overall objectives of the V & V testing should be to provide fine tuning for the model design through iterative feedback, and to demonstrate:

* ease of model use, i.e., operation with minimal input data sets,

* proper model interface with the operational data base,

* computational accuracy, and

* software support cost prediction capability.

DELIVER FINAL PSC MODEL

The final task in the development of the PSCM is the delivery of the model to AFWAL/AA at WPAFB. This involves three steps: installation on the AF CYBER 175 computer, production of final documentation, and training of specified users.

Installation

The model must be installed on the CYBER 175 computer at WPAFB and throughly checked out to assure its proper functioning. This checkout should include a limited rerun of selected validation tests to verify that the model produces the proper results.

Documentation

All documentation necessary for understanding and maintenance of the model must be developed and delivered to the customer. This documentation should include:

* Programmer's Manual - Description of model theory of operation, rationale for algorithms, model structure, flowcharts, code listings, data base description, data item dictionary, and details of CYBER 175 implementation.

* User's Manual - Description of model theory of operation, data item definitions, description of model input process and running procedure, and limitations of model.

* Test Report - Description of V&V test plans, and results of tests.

Training

In addition to a general briefing to the AF project engineer and other interested persons, specific training should be conducted as follows:

* Programmer Training - A one-day training session for the individual(s) designated as the responsible programmer(s) for the Predictive Software Cost Model.

* User Training - Two half-day sessions for those individuals designated as users of the PSCM.

ESTIMATE OF RESOURCE REQUIREMENTS FOR PHASE II

Estimates of personhours and other resources required for conducting the Phase II tasks identified in the PSC model development roadmap are presented below. The tasks to be performed are:

* collect data
* analyze data
* develop data base
* design PSC model
* test PSC model
* deliver PSC model

A brief description of each task is presented along with the estimate of resources required.

Collect Data

The data collection task requires a significantly large expenditure of effort since it is foundational to the performance of the rest of the Phase II tasks. It will be necessary to visit each ALC, WPAFB and RADC to obtain additional available quantitative data (sometimes by manual file search), and to conduct in-depth interviews with software professionals to obtain process and other qualitative data.

At each ALC: as a minimum, two two-person trips will be required. For each trip, at least one week will be required to collect additional identified quantitative and qualitative data, plus one week after returning home in order to compile and summarize the collected data. Total of 40 personweeks.

At RADC: one two-person trip should be made to obtain their data which will be useful in helping to understand general software relationships. In particular, their data on software reliability should be examined. Again, one week at RADC plus one week after returning home is estimated for the two persons. Total of 4 personweeks.

At WPAFB: three two-person trips will be required. Two two-person trips of one week duration will be for program coordination; one two-person trip will be to study the AFWAL/AA cost estimating process (one week at WPAFB plus one week after returning home). Total of 8 personweeks.

The total estimated labor requirements for the data collection task are:

Number of Person Trips	Destination	Personhours
4	OO-ALC	320
4	SA-ALC	320
4	OC-ALC	320
4	SM-ALC	320
4	WR-ALC	320
2	RADC	160
6	WPAFB	320
28		2080 Total

Commensurate travel expense will be required.

Analyze Data

The data analysis task begins with the preliminary organization and evaluation of the total collection of data in all its forms to create data "working files", and proceeds to perform detailed statistical and qualitative analyses to determine the functional forms of the various candidate parametric relationships for software support cost estimating.

During Phase I, personhours required for data analysis were slightly less than those required for data collection, but no detailed statistical analysis was performed. It seems reasonable during Phase II, therefore, to require slightly more than the quantity of personhours estimated for data collection, i.e., 2080 hours, plus 10%, or 2288 hours. In addition, computer time will be required to perform the analyses, estimated to be 6 CPU hours (based on Hughes use of Amdahl 470; this can vary widely depending on the computing facility used).

Develop Data Base

This task involves establishing the data base structure required to support the PSC model, creating the historical data base (including coding and entering data), and associated debugging efforts. Labor requirements are estimated to be:

	Personhours
Data base structure	160
Create data base	80
Debug	40
	280 Total

Required computer time is estimated to be 0.4 CPU hour (again based on use of Amdahl 470).

Design PSC Model

The model design task involves developing the final set of estimating algorithms, designing the model structure and data flows, coding the model and refining the model as a result of validation and verification testing. Algorithm development, given the basic functional forms derived in the data analysis task, should require approximately 26 personweeks. Structure design, including analysis of the CYBER 175 requirements, should require about 9 personweeks. Model coding/debug and iterative refinements should each require approximately 7 personweeks. In summary, estimated labor requirements for the model design task are:

	Personhours
Algorithm development	1040
Model structure design	360
Coding and debug	280
Refinements	280
	1960 Total

Computer time required for this task is estimated to be 3 CPU hours (Amdahl 470).

Test PSC Model

The test task consists of validation and verification (V & V) of the PSC model. V & V requires operation of the model and testing against pre-established criteria. Efforts include developing the V & V plans, developing test data and running the model, and evaluating the results. Labor requirements are estimated to be:

	Personhours
V & V plans	160
Testing	160
Evaluation	160
	480 Total

Required computer time is estimated to be 0.6 CPU hour (Amdahl 470).

Deliver PSC Model

The model delivery task involves production of final documentation, installation of the model on the AF CYBER 175 computer at AFWAL/AA, and training user personnel. Final documentation should include a Programmers' Manual, a Users' Manual, and a Test Report as a minimum. Model installation will require visits of two persons to AFWAL/AA for a two-week period;

access to the AF CYBER 175 computer on an expeditious basis will be required during that time period to effect the model installation. Training will require two personweeks for development of training materials, and one week at AFAL for conduct of the training.

The total estimated labor requirements for the model delivery task are:

	Number of Person Trips	Destination	Personhours
Documentation			
* Programmer's Manual	-	-	700
* User's Manual	-	-	360
* Test Report	-	-	200
Model installation	2	WPAFB	160
Training	1	WPAFB	120
		Total	1540

Commensurate travel expense will be required. Computer time (CYBER 175) during model installation must be provided by the Air Force.

Resource Requirements Summary

The preliminary estimate of resource requirements for performing Phase II efforts is summarized in Table 37. Total engineering labor (8628 personhours) constitutes slightly more than 4.5 person years of effort; associated clerical support is estimated at an additional 10% of engineering effort. ODC requirements for travel must be commensurate with identified person-trips for data collection and model delivery tasks. ODC requirements for computer time reflect CPU hours using Hughes' Amdahl 470 computing facilities; use of other computer facilities would require an equivalent amount of computer resource requirements.

PHASE II RISKS

As with all worthwhile endeavors, conducting the Phase II PSC model development effort is not without risk. Certain risks are real entities, but if they are recognized for what they are, and if the endeavor is undertaken with their foreknowledge, the effort can be conducted so as to minimize the risks by addressing them directly rather than pretending they do not exist.

For Phase II, the primary risks center about three areas, in decreasing order of importance, i.e.,

* data upon which the study is based,
* the approach to the model design, and
* model usage.

TABLE 37. PHASE II RESOURCE REQUIREMENT ESTIMATE

Task	Labor (personhours)	Other Resources	
		Computer Time (CPU hours)	Travel
Collect data	2080	None	28 person-trips to ALCs, RADC and WPAFB
Analyze data	2288	6.0	None
Develop data base	280	0.4	None
Design PSC model	1960	3.0	None
Test PSC model	480	0.6	None
Deliver PSC model	1540	AF supplied (CYBER 175)	3 person-trips to WPAFB
Clerical support	863	none	none
TOTAL	9491	10.0 plus CYBER 175	31 person-trips

These risks, however, are not peculiar to the approach outlined in the model development roadmap, the subject of this report section. They are risks that would have to be faced whatever the model development approach.

The model development approach recommended for Phase II attempts to overcome these risks by (1) collecting the best data feasibly available, (2) utilizing a modelling approach which is not totally dependent on a large volume of homogeneous data samples, and (3) modularizing the model and data base designs.

A continuing series of updates is indicated to incorporate new data which will become available as software packages are transitioned to post-PMRT status, and to provide more and better quantitative data to improve model accuracy and completeness.

Data Risks

The data risks relate to the availability of requisite data, and whether the available data satisfies the attributes of comprehensiveness, completeness and representativeness, is pertinent, and is adequate in quantity and quality.

1) The data, while accurate, may not contain the required intelligence. The quantitative data may not be of sufficient quantity and quality to determine meaningful results by unbiased analyses.

   * Proper analysis techniques may not provide the relationships that are expected.

   * Rigorous and correct statistical analysis techniques may show poor or insignificant correlations among independent and dependent variables.

   * Root cause/effect relationships may not be there ... except through qualitative (opinions, feelings, etc.) means. (The effort then becomes a model representation of the modeller's biases and pre-conceived views.)

2) The data, while complete, may not be accurate. For example:

   * Wrong manhours may have been recorded against the software support tasks and subtasks.

   * Significant labor may have been cross-recorded among tasks, or cross-recorded with tasks unrelated to software support.

   * There may have been cross-recording or mis-recording of other costs related to software support.

   * How much of the data represents actuals versus how much represents allocations?

3) The data may be too heavily qualitative. While some judicious use of qualitative data is recognized to be required, paucity of quantitative data may necessitate too great a dependence on the opinions (and other biases) of practitioners and the modeller to fill the quantitative data voids.

Approach Risks

The model design approach risks relate to constraints imposed on model usage. The approach, while satisfying the minimum input/ease of use requirements, is also constrained by them.

Software concepts/designs which are candidates for early program cost predicting must be reducible to a description in terms of characteristics compatible with the PSC model input requirements, i.e., the minimum, easy input. If a candidate software package type is significantly different from those upon which the model is based, it may not be readily represented by model-compatible input characteristics, and exercise by the model may be limited or impossible.

The ability to update the model based on new performance data is a constraint imposed by the model design approach. The model will have been based on the precise set of data from the data collection task. New performance data can be handled in one of several ways: add the new data to the existing data and redesign the model; design a new model based on the new data; or provide modularity in the model and data base designs to accommodate modular add-ons/modifications. The PSC model design approach outlined in this section provides for the latter method of handling new performance data.

Model Usage Risks

Characteristics of software used as model inputs in early program cost studies may be changed significantly during subsequent phases of the program. The characteristics must be definable, specifiable as requirements, and enforceable for new software programs. If input characteristics provided for cost analysis in early program trade studies are not controlled, and are changed when software design and development is implemented, predicted software support costs (from the PSC model) would always be much different from resulting actual software support requirements and their costs. This would make the PSC model appear to be "wrong" when, in reality, the conditions (input characteristics) were changed. The model flexibility and sensitivity should handle some change, but not drastic changes in basic software package characteristics.

Model usage risks also relate to interpretation and application of results of studies which utilize the PSC model. No model is the perfect predictor in absolute terms. However, when conditions are held constant for comparing one software alternative with one or more other alternatives, relative values of the predictions can be excellent comparative evaluation tools. In other words, don't hang your hat on the absolute predicted cost values, but put a great deal of stock in relative differences, when the predicting model is sound.

The PSC model that will have been designed in Phase II will be based on experience over the preceeding 4-5 years, tempered by current conceptual prognostications. The model will then be applied to subsequent software programs. If the advances in software technology during the period the model is intended for use are as rapid as they have been in the recent past, will the model be current enough to be effective? What about the new software technology advancements - unknown now, but bound to occur? Here, again, the modular model design approach comes into play.

IX. CONCLUSIONS

Two primary conclusions are drawn from results of Phase I of the PSCM study:

1) A new model directed at avionics software support cost prediction is needed.

2) Development of such a model is feasible.

Other conclusions from the study are secondary in nature, expanding upon and supporting these two.

A new predictive software cost model should be developed to satisfy the Air Force requirement to be able to predict software costs (especially software support costs) in planning the development of advanced avionics systems.

* Existing software cost models do not adequately address the aspects of software support costs.

  - Most models concentrate on estimating costs of software development and give little, if any, attention to costs of software support. As a corollary, the sophistication of treatment for software development aspects far exceeds that given to software support aspects.

  - Models do not generally reflect AFLC support processes, and they do not account for a large portion of the support resources required.

* Good modelling technology exists which can be readily applied to software support estimating.

  - Techniques used in the (primarily) software development cost prediction models may be applicable in addressing software support cost aspects.

* A lack of definitive historical data on software operation and support and their associated costs has hampered support cost estimating and assessment of its contribution to software life cycle costs.

* There is insufficient understanding of the factors that drive and otherwise affect software support costs.

  - Most models consider manpower as the key driver to the virtual exclusion of other important factors.

* Bases upon which most models are built are extremely limited.

- Software program size, or the number of instructions, forms the basis for most software cost models. Other/additional pertinent software program characteristics should be identified, and additional estimating relationships developed for the software support cost application.

* Distinct software packages being supported by AFLC are numerous, and increasing at an ever increasing rate.

  - Over 50,000 separate embedded computer system programs are currently estimated to be in the Air Force inventory, increasing at a rate of 6400 packages per year.

It is feasible to develop a new predictive software cost model which adequately addresses software support cost estimations. While the amount of available quantitative data is marginal, its adequacy can be improved by making use of various special studies and surveys of key software engineers and managers; modelling technology is adequate; only the approach need be established.

* Data from six sample software packages and from other identified data sources indicate there is probably adequate quantity and quality of data upon which to base the desired model design.

  - Some quantitative data on changes and manhours is readily obtainable on the A-7D and F/FB-111 aircraft OFPs. Data on the F-15 and F-16 will become available in the future. Data on other packages and on EW software may be obtainable by a manual search of project files. Other cost data and relationship data, such as the impact of V&V requirements on aircraft flight test requirements, will have to be obtained by special studies and interviews of experienced practitioners in the field.

  - Additional data will have to be collected to update that which has already been collected, and to fill information voids.

* Methods of collecting software support performance data differ among ALCs, with differing emphases that reflect program tailoring and size as well as accounting practices.

  - Software support within the ALCs is generally task oriented.

* The new model should be developed, to the extent possible, based on quantitative historical software support performance data, supplemented by qualitative information from software practitioners.

* Six primary resource categories should be included in the model design.

  - personnel
  - support equipment
  - support software
  - facilities
  - data and documentation
  - flight test

* Five primary support cost drivers determined in the Phase I study should be included in the model design.

  - change requirements, both frequency and size
  - prime system software, including program size, architecture, and hardware constraints
  - prime system hardware
  - software support personnel, including experience, training and productivity
  - test requirements for software changes

* The best modelling methodology of the alternatives evaluated is a modified bottom-up (element estimate) approach.

  - The recommended approach best utilizes available data; it also provides ease-of-use by minimizing user input requirements.
  - The modular feature of the approach provides updating capability for new data, and enables modifications (modular add-ons) to accommodate new requirements.

* The primary risk of the recommended methodology centers about the availability of requisite historical data. The quality and quantity of available data may be less than expected.

X. OBSERVATIONS AND RECOMMENDATIONS

OBSERVATIONS

In conducting the field surveys during this study, several observations were made which are only partially or not directly related to the purposes of this study. They are as follows:

1. An Air Force-wide software support data system analogous to the AFM 66-1 hardware maintenance data collection system has not yet been established, nor are there apparently any immediate plans to do so. Furthermore, it is difficult (if not impossible) to identify the effort related to software development in various system acquisitions. This lack of consistent, uniform historical data imposes a severe handicap on the Air Force's ability to understand, compare and predict software support needs, capabilities and performance.

2. Operational flight programs for training simulators have to be written after the aircraft OFPs are completed. This not only imposes an additional programming burden on the Air Force, but also exacts the undesirable side effect that training simulators normally lag the operational aircraft by about one block change cycle.

3. Several persons expressed the desirability of MMEC (and other software organization) representatives from the various ALC's meeting together periodically to exchange information on various problems, proposed solutions, and possible approaches to technological and managerial improvements in the support of avionics embedded software.

RECOMMENDATIONS

The following recommendations are offered for Air Force consideration:

1. The Air Force Avionics Laboratory should pursue implementation of the plan detailed herein for development of a model to predict the support costs of avionics embedded software.

2. The Air Force Logistics Command should move to develop and implement a system to collect data on the support of avionics embedded software. As a related effort, the Air Force Systems Command should refine the Work Breakdown Structure to clearly delineate, specify and control the software-related efforts on all new acquisitions.

3. The Air Force should investigate the technical feasibility and economic desirability of designing aircraft training simulators so that they can utilize the same operational flight programs as the operational aircraft.

4. The Air Force Logistics Command should sponsor periodic meetings of representatives from all Air Force organizations involved in designing or supporting avionics embedded software, in order to facilitate understanding of common problems and enhance communication of possible solutions and technological improvements.

PSCM BIBLIOGRAPHY

I. Regulations and Procedures

1. MM Operating Instruction 800-14, "Acquisition Management," Warner Robins ALC, Robins AFB, Georgia, 15 November 1978

2. MMEC Operating Instruction 800-14, "Software Change Processing Procedures for Operational Flight Programs," 2 June 1978

3. "F-111 Configuration Management Documentation Guide for MMECP Software," Sacramento ALC, Sacramento, Ca., 14 December 1978

4. "F-111 Operational/Support Configuration Management Procedure for F/FB-111 Operational Software," Sacramento ALC, Sacramento, Ca., 1978

5. Air Force Regulation 800-14, Vol I, "Management of Computer Resources in Systems," 10 May 1974.

6. Air Force Regulation 800-14, Vol. II, "Acquisition and Support Procedures for Computer Resources in Systems," 26 September 1975.

7. Department of Defense Directive 4100.35, "Development of Integrated Logistic Support for Systems/Equipments," October 1, 1970.

8. Department of Defense Guide 4100.35-G, "Integrated Logistic Support Planning Guide for DoD Systems and Equipments," October 15, 1968.

9. MIL-STD-483, "Military Standard Configuration Management Practices for Systems, Equipment, Munitions and Computer Programs," 31 December 1970.

10. MIL-STD-490, "Military Standard Specification Practices," 30 October 1968.

11. Department of Defense Standard 7935.1-S, "Automated Data System Documentation Standards," 13 September 1977.

12. MMR Operating Instruction 800-01, "Software Change Processing/Configuration Management for EW Systems," (Draft), 12 December 1979

13. MMEC Operating Instruction 65-2, "Configuration Management for Avionics Integration Support Facilities," (Draft)

14. MMR Operating Instruction 800-03, "System Verification Test Procedures Using Simulation and Analysis Test Systems," (Draft), 12 December 1979

15. MMR Operating Instruction, "Software Testing Guidelines," (Draft), 12 December 1979

16. MIL-STD-1521A, "Technical Reviews and Audits for Systems, Equipment, and Computer Programs," DoD, 1 June 1976

17. AFR 173-10, Volume I, "USAF Cost and Planning Factors," 6 February 1975, revised 2 May 1977

II. AFWAL/AA and ALC Data and Reports

18. "Embedded Computer System Integrated Support Plan," Vols. I,II & III Oklahoma City ALC, Tinker AFB, Oklahoma, January 1979

19. "Definition of An Approach to Establishment of an F-15 Avionics Software Support Capability," Avionics Engineering Directorate, Aeronautical Systems Division Report ASD-TR-74-30, Wright-Patterson AFB, Ohio, July 1974

20. "Avionics Operational Flight Program Integrated Support Plan," Warner Robins ALC, Robins AFB, Georgia, No. 31073-6041-TU-00, January 1978

21. "Embedded Computer Resources," Engineering Division, Directorate of Material Management, Warner Robins AFB, Georgia, April 1978

22. "F-16 FCC OFP Independent Assessment, Final Report," Simulation Technology Report STI-78-545, 14 July 1978

23. "F-16 Independent Assessment Fire Control Computer Operational Flight Program Validation Plan and Procedures," Simulation Technology Report STI-78-545, September 30, 1978

24. Vendt, B. A., "Software Support For the F-16 Avionic Computers," Air Force Institute of Technology, Report AD-A020-361, Wright Patterson AFB, Ohio, December 1975

25. "Air Force Computer Resources Inventory," ASD Computer Center - CPIN Accession Listing (Sample Printout)

26. Heinger, K. L. and Parnas, D. L., "Software Requirements for the A-7E Aircraft," Naval Research Laboratory NRL Memorandum Report 3876, November 1978

27. "Shopping List" ... a listing of hardware and software required for operational (maintenance) support of the A-7D Computer Laboratory, dated April 9, 1979

28. "Data Reduction Subroutine Analysis"... tabular summary of A-7 OFP and support software routines identifying lines of code, calls, branches, and variables and relative time (for a programmer or systems engineer) to learn

29. "TM7503-291 on Revised Project Plan for Joint NRL-NWC Avionics Software Structuring Demonstration (W/A-7D OFP Maintenance History Milestones Chart)," 15 November 1978

30. "A-7 OFP/Pave Penney Program Assembly Revision History, Program Change Flows, and Program Listings, Air Force 2 Pave Penney Baseline AF2P-2," January 6, 1979

31. A-7 Change Control Methods Outline and Sample Source Change Listing

32. A-7 Software Guideline, May 21, 1979. (Software documentation and coding techniques requirements)

33. "Operational Support Flight Manual," USAF Series, A-7D, No. T.O. 1A-7D-1S-103, 7 May 1973

34. "A-7E TC-2/2A Software Tools Reference Manual," NWC Technical Memorandum 3595, NWC, China Lake, Ca., September 1978

35. "A-7D OFP TC-2/2A Handbook Assembler," NWC Technical Memorandum 3597, NWC, China Lake, Ca., September 1978

36. "A-7D OFP TC-2/2A Assembly," NWC Technical Memorandum 3596, NWC, China Lake, CA., September 1978

37. F-15 Avionics Software Computer Resources Integrated Support Plan (CRISP), March 1978

38. WR-ALC Computer Resources (MMEC), Work Center Comments and Time Standard Description (with MMEC Work Flow Diagrams)

39. WR-ALC F-15 Avionics Integrated Support Facility Peynomic Simulator System Specification

40. Smithhisler, W. L., "F-15 SAR Radar Life Cycle Cost Analysis Report," Hughes Aircraft, IDC (orig. 28-57-00), 2 November 1978

41. "F-111 Charter for the MMECP Steering Committee (listing printout)," Sacramento ALC, Sacramento, Ca., 1978

42. F-111 OFP Change Summary (listings), 9-74 through 19-79

43. "Computer Update for F-111 Aircraft," SM-ALC/MME, Briefing Charts, 18 November 1978 (Note: includes OFP change analysis data, cost estimates, program sizing trade-offs, including F-111A/E NAV-BOMB analysis based on AFAL-TR-79-1043 cited in references III.2, and IV.5)

44. "DOT File Listing (computer/MIS printout)," F-111 OFP and Support Software Programs, Sacramento ALC, Sacramento, Ca., 1979

45. F-16 OFP Summary Project Analysis, F-16 Block IMMA-D76-003, Period August 28, 1979 (change history listing printout)

III Technical Reports and Papers

46. Cost Reporting Elements and Activity Cost Tradeoffs for Defense System Software, General Research Corporation Report FSD-TR-77-262, May 1977

47. F-111 A/E Digital BOMB-NAV System Software Analysis, Battelle-Columbus Laboratories Report (AFAL-TR-79-1043, January 1979

48. Herd, J. H., et. al., Software Cost Estimation Study, Doty Associates Report AD/A-042 264, June 1977

49. Software Cost Estimating Methodology, AFAL-TR-77-66, August 1977

50. "Quantitative Software Models," Software Engineering Research Review, SRR-1, Data and Analysis Center for Software, Rome Air Development Center, Griffiss AFB, N.Y., March 1979

51. Summary of Technical Progress, Investigation of Software Models, RADC Report TR-79-188, Rome Air Development Center, July 1979

52. Embedded Computer Resources, Engineering Division, Directorate of Materiel Management, Warner Robins ALC, Robins AFB, Georgia, April 1978

53. Devenny, Capt. T. J., An Exploratory Study of Software Cost Estimating at the Electronic Systems Division, Air Force Institute of Technology, Wright-Patterson AFB, Ohio, September 1976 (Masters Thesis)

54. "Quantitative Software Models," Software Engineering Research Review, SRR-1, Data & Analysis Center for Software, March 1979

55. Bourdon, Captain G. A. and Captain J. A. Duquette, A Computerized Model for Estimating Software Life Cycle Costs (Model Concept), Comptroller, Electronics Systems Division, Hanscom Air Force Base, ESO-TR-77-253, Vol I, April 1978

56. Herd, J. H., et. al., Software Cost Estimation Study, Vol I, Doty Associates Inc., TR No. 151, Rockville, Md., AD/A-042264, June 1977

57. Highland, R. W., "Exploratory Study of Software Support Costs," IDC Ref 281742/111, Hughes Aircraft Company, 24 July 1979

58. James, Lt. T. G. Jr., Software Cost Estimating Methodology, Air Force Avionics Laboratory, Wright-Patterson AFB, Ohio, AFAL-TR-77-66, August 1977

59. Junk, W. S., and J. A. McCall, Software Cost Estimation and Evaluation Handbook, General Electric, 29 August 1978

60. Clapp, J. A., A Review of Software Cost Estimation Methods, The MITRE Corporation, ESD-TR76-271, August 1976

61. Freiman, F. R., "PRICE Software Model - Overview," RCA PRICE Systems, Cherry Hill, N. J., November 1977

62. Stone, A. J., "1976 IR&D Year-End Report on Software Life Cycle Cost Analysis," Hughes Aircraft Company, 1976

63. Schane, R. N., "Potential Effects of Standardization on Avionics Software Life-Cycle Cost," 1979 NAECON Proceedings, Logicon Report TR-79-101, 29 June 1979, pp. 558-567

64. Babiak, Captain N. J. and L. D. Parriott, Jr., "Management of Embedded Computer Support Facilities," Proceedings of (AIAA) Computers in Aerospace Conference, Los Angeles, Ca., November 1977, pp 286-296

65. Bridgman, M. S., A. F. Hitt, and T. R. Cork, "Assessment of Software Development and Maintenance Costs Due to Retrofit of Embedded Avionics Computer," Proceedings of 2nd Computers in Aerospace Conference, October 1979, pp 17-22

66. Ferens, D. V., and R. L. Harris, "Avionics Computer Software Operation and Support Cost Estimation," Proceedings of the IEEE 1979 NAECON, Dayton, Ohio, 15-17 May 1979

67. Norden, P. V., "Project Life Cycle Modeling," Software Phenomenology - Working Papers of the Software Life Cycle Management Workshop, Airle House, August 1977

68. Putnam, L. H., "A General Empirical Solution to the Macro Software Sizing and Estimating Problem," IEEE Transactions on Software Engineering, Vol SE-4, July 1978

69. Stone, Harold S., "Life Cycle Cost Analysis of Instruction-Set Architecture Standardization for Military Computer Systems," Computer, April 1979, pp 35-47

70. Walston, C. E., and C. P. Felix, "A Method of Programming Measurement and Estimation," IBM Systems Journal, Volume Sixteen, Number one, 1977, pp 54-75

71. Merrill, D. H., "Software Costing Models," IDC Ref. 7555.31/24, Hughes Aircraft Company, 7 September 1979

72. Metrics of Software Quality, McDonnell Douglas Astronautics Company Report MDC-G7517, July 1978

73. Norden, P. V., "Useful Tools for Project Management," Management of Production, M. K. Starr (Ed.), Penguin Books, 1970

74. Modular Life Cycle Cost Model for Advanced Aircraft Systems Phase II, Vol I, Cost Methodology Development and Application, AFFDL-TR-78-40, Grumman Aerospace Corporation, Bethpage, New York, April 1978

75. Irvine, A. and M. Mckenzie, Evaluation of the DSN Software Methodology, DSN Progress Report 42-48, September and October 1978

76. Dalkey, N. C., "The Delphi Method: An Experimental Study of Group Opinion," the Rand Corporation, RM-5888-PR; Santa Monica, California, 1969.

77. DeRoze, B. C., Defense System Software Management Plan, Report AD/A-022 558, National Technical Information Service, Springfield, Virginia, 1976.

78. DeRoze, B. C. and Nyman, T. H., "The Software Life Cycle," SIGNAL, Nov/Dec. 1977

79. DoD Weapon Systems Software Management Study, The Johns Hopkins University Applied Physics Laboratory, Maryland, 1975.

80. Findings and Recommendations of the Joint Logistics Commanders Software Reliability Work Group (SRWG Report) (2 volumes), Air Force Systems Command, Andrews Air Force Base, District of Columbia, 1975

81. McCall, J. L., et. al., Factors in Software Quality, RADC-TR-77-369, Rome Air Development Center, Griffiss Air Force Base, New York, 1977

82. Myers, G. J., Software Reliability Principles and Practices, New York: John Wiley and Sons, 1976

83. Ortega, L. H., "Documentation Standards Final Report," Vol VII, Structured Programming Series, Report RADC-TR-74-300, Rome Air Development Center, Griffiss Air Force Base, New York, 1974

84. Smith, R. L., "Estimating Software Project Resource Requirements," Vol. XI, Structured Programming Series, Report RADC-TR-74-300, Rome Air Development Center, Griffiss Air Force Base, NeW York, 1975

85. Patterson, A. F., and E. M. Long, "The F-111 OFP Dynamic Simulation System," Proceedings of the IEEE 1976 NAECON, Dayton, Ohio, 18-20 May 1976.

86. Freedman, R. J., "Organic Support of the A-7E Airborn Computer System," Proceedings of the Aeronautical Systems Software Workshop, National Technical Information Service Report AD/A-004-547, July 1974, pp. 261-286

87. Teichgraeben, R. D., and D. M. DeMoss, "Real Time Simulation as a Tool for F-16 Operational Flight Program Development and Testing," Proceedings of the AIAA 2nd Digital Avionics Systems Conference, November, 1977

88. Anderson, P. G., "Redundancy Techniques for Software Quality," 1978 Proceedings Annual Reliability and Maintainability Symposium, Los Angeles, Ca., 17-19 January 1978

89. Belady, L. A., "Software Complexity," Software Phenomenology - Working Papers of the Software Life Cycle Management Workshop, Airle House, August 1977

90. Belady, L. A., "Measures and Empirical Studies," Working Papers of the Second Software Life Cycle Management Workshop, IEEE, Atlanta, Georgia, August 21-22, 1978

91. Boehm, B. W., "Quantitative Evaluation of Software Quality," Software Phenomenology-Working Papers of the Software Life Cycle Management Workshop, Airle House, August 1977

92. Boehm, B. W., "Software and Its Impact: A Quantitative Assessment," Datamation, May 1973, pp 48-59

93. Brooks, F. P. Jr., "The Mythical Man-Month," DATAMATION, December 1974, pp 45-52

94. Curtis, B., et. al., "Some Distinctions Between the Psychological and Computational Complexity of Software," Working Papers of the Second Software Life Cycle Management Workshop, IEEE, August 1978

95. Cave, W. C., and A. B. Salisbury, "Controlling the Software Life Cycle - The Project Management Task," IEEE Transactions on Software Engineering, Volume SE-4, Number 4, July 1978

96. Ferrell, J., "Logical Planning for Automatic Test Equipment Software Requirements," Computers in Aerospace Conference, Los Angeles, Ca., November 1977 pp 260-267

97. Gardiner, P. C., "Multi-Attribute Utility Measure for Logistics Decision Making," Parts I and II, SPECTRUM, Winter 1978 and Summer 1979

98. Goodenough, J. B., and S. L. Gerhart, "Toward a Theory of Test Data Selection," IEEE Transaction on Software Engineering, Vol SE-1, No 2, June 1975

99. Griffith, V. G., "Can HOL Use In Avionics Pay for Itself?," Aeronautical Systems Software Workshop, July 1974

100. Henry, R. C., "Notes on Software Reliability and Quality," Software Phenomenology - Working Papers of the Software Life Cycle Management Workshop, Airle Home, August 1977

101. "Control Data Cyber 170 Series," 70C-263-12A Computer, Datapro Research Corporation, Delran, N. J., June 1977, revised May 1978

102. "JPL Improves Software Development Process," COMPUTER IEEE, February 1978, pp 22-25

103. Kline, M. B., "Software and Hardware R & M: What Are the Differences?," 1980 Proceedings Annual Reliability and Maintainability Symposium, pp 179-185

104. Manley, J. H., "Software Life Cycle Management: Dynamics Theory", Working Papers of the Second Software Life Cycle Management Workshop, Atlanta, Georgia, August 21-22, 1978

105. McCabe, T. J., "Software Complexity Measurement," Working Papers of the Second Software Life Cycle Management Workship, IEEE, August 1978

106. Miller, C. R., "Software Maintenance and Life Cycle Management," Software Phenomenology - Working Papers of the Software Life Cycle Management Workship, Airle House, 21-23 August 1977

107. Myers, W. (Contributing Editor), "The Need for Software Engineering," Computer, IEEE, February 1978

108. Peercy, D. E., "A Software Maintainability Evolution Methodology," AIAA 2nd Computers in Aerospace Conference, October 1979, pp 315-325

109. Rubey, R. J., J. A. Dana, and P. W. Biche, "Quantitative Aspects of Software Validation," IEEE Transactions on Software Engineering, Vol SE-1, No 2, June 1975

110. De Roze, B. C., "The Software Life Cycle - A Management and Technology Challenge in the Department of Defense," IEEE Transactions on Software Engineering, Vol SE-4, No 4, July 1978, pp 309-317

111. Snyders, J., "Slashing Software Maintenance Costs," Computer Decisions, July 1979, Pp 44-50

112. Tansworth, R. C., "The Workbreakdown Structure in Software Project Management," Working Papers of the Second Software Life Cycle Management Workshop, IEEE, August 1978

113. Walters, G. F., and J. A. McCall, "The Development of Metrics for Software R & M," Proceedings of the 1978 Annual Reliability and Maintainability Symposium, Los Angeles, California, 17-19 January 1978

114. Wolverton, R. W., "Software Life Cycle Management - Dynamics Practice," Working Papers of the Second Software Life Cycle Management Workshop, IEEE, Atlanta, Georgia, August 21-22, 1978

115. Boehm, B. W., "The High Cost of Software," Proceedings of a Symposium on the High Cost of Software, Monterey, California, 1973 (published by Stanford Research Institute, Menlo Park, California)

116. Boehm, B. W., et. al., Characteristics of Software Quality, TRW Software Series TRW-SS-73-09, December 1973

117. Drezner, S. M., et. al., "The Computer Resources Management Study," The Rand Corporation R-1855/1-PR., Santa Monica, California, 1976

118. Dunn, R. H. and Ullman, R. S., "A Workable Software Quality/Reliability Plan," Proceedings of the 1978 Annual Reliability and Maintainability Symposium, Los Angeles, Ca., 17-19 January 1978

119. Howley, P. P., "Software Quality Assurance for Reliable Software," Proceedings of the 1978 Annual Reliability and Maintainability Symposium, Los Angeles, Ca., 17-19 January 1978

120. "Implications of Using Modular Programmins," Guide Nr. 1, John Hoskyns and Company, Ltd., Hoskyns Systems Research, Inc., 600 Third Avenue, New York, New York, 1973

121. Lindhorst, W. M., "Scheduled Maintenance of Applications Software," Datamation, May 1973

122. McCarthy, R., "Applying the Technique of Configuration Management to Software," Defense Management Journal, October 1975

123. Mooney, J. W., "Organized Program Maintenance," Maintenance," Datamation, February 1975

124. Munson, J. B., "Designing for Software Maintainability - A Major Concern for Life Cycle Costing," 1978 Conference on Design to Cost/Life Cycle Cost, State of the Art Seminars, Los Angeles, California

125. Proceedings of a Symposium on the High Cost of Software, Monterey, California, 1973 (published by Stanford Research Institute, Menlo Park, California)

126. Shooman, M. L. and Bolskey, M. I., "Software Errors: Types, Distribution, Test and Correction Times," International Conference on Reliable Software, 1975

127. Sukert, A. N., "An Investigation of Software Reliability Models," Proceedings of the 1977 Annual Reliability and Maintainability Symposium, The Institute of Electrical and Electronic Engineers, Inc

128. Swanson, E. B., "The Dimensions of Maintenance," Proceedings of the Second International Conference on Software Engineering, San Francisco, 1976

129. Trainor, W. L., "Software - From Satan to Saviour," USAF Avionics Laboratory, Wright-Patterson Air Force Base, Ohio; paper presented at National Aerospace Electronics Conference, 1973

130. Wolverton, R. W., "The Cost of Developing Large-Scale Software," IEEE Transactions on Computers, Vol C-23, Nr. 6, June 1974

131. Wulf, W. A., "Programming Methodology," Proceedings of a Symposium on the High Cost of Software, Monterey, California 1973

132. Parnas, D. L., "Designing Software for Ease of Extension and Contraction," IEEE Transactions on Software Engineering, Vol. SE-5, No. 2, March 1979

133. Mills, Capt. B. S., "Simplified Procedures for Performing Life Cycle Cost Analyses," Proceedings of the IEEE 1979 NAECON, Dayton, Ohio, 15-17 May 1979

134. Patterson, A. E., "Operational Flight Programs: Change and Control," Conference Proceedings on Managing of Weapon Systems Software, USAF University, Maxwell AFB, ALA, May 1976.

135. Patterson, A.E., and Capt. R. G. Algire, "An Approach to Modern Avionics Integration Support," Procedings of the IEEE 1976 NAECON, Dayton, Ohio, 18-20 May 1976.

APPENDIX A. DESCRIPTION OF CPIN SYSTEM

The Technical Order Section of the Operations and Support Branch (MMEDU) at Oklahoma City ALC is responsible for the CPIN (Computer Program Identification Number) system and the AFCR (Air Force Computer Resources) inventory.

The CPIN system provides for the identification, indexing, requisition, distribution and follow-on requirement of all Air Force computer programs and associated documentation for computer systems embedded in or supporting weapon systems. All Air Force computer software acquired, developed, managed, or used under the AFR 800-2 program management concept will be included in the CPIN system. Major types are operational, test or support programs applicable to Operational Flight Programs (OFP), Electronic Warfare (EW), Ground C-E-M, Simulator or Aircrew Training Devices (ATD), Automatic Test Equipment (ATE) and certain Command and Control (C & C) systems. ECS software for new weapon systems currently under development will be included in the CPIN system. Applicable software currently in other data systems, such as the Technical Order (TO) System (-CT check tapes), are being phased into CPIN on a system by system basis (reference TO 00-5-2, Section IV, Para 4-1).

CPIN Compendiums are similar to TO Numerical Index and Requirements Table (NI&RT). CPINs are assigned to each computer program, or aggregate of computer programs designated as configured items (CPCI), and to the related documentation package. CPINs and descriptive data are indexed in the compedium. The compendium will be updated, published and distributed monthly to all ECS software managers and software users who establish a requirement for the publication. Command compendiums consisting of ECS software unique to or managed by a specific command will also be available at a later date.

Separate compendia will be published for each CPIN category, which are related to types of aerospace systems or equipment. CPIN categories, description, and compendium numbers are:

Category	Description	Compedium Number
81	Aircraft	80-1-81
82	Missiles	80-1-82
83	Ground C-E-M	80-1-83
84	Simulator/Trainer	80-1-84
85	Test Stations/Tester	80-1-85
87	General Purpose Computer	80-1-87
88	Other	80-1-88
91	Command and Control	80-1-91

CPIN 80-1-81

USAF

COMPUTER PROGRAM IDENTIFICATION NUMBER SYSTEM

COMPENDIUM

OF

AIRCRAFT

EMBEDDED COMPUTER SYSTEM SOFTWARE

SAMPLE

FIRST EDITION

PUBLISHED UNDER AUTHORITY OF THE SECRETARY OF THE AIR FORCE

1 JUNE 1979

TABLE OF CONTENTS

PART I - ACTIVE

COMPUTER PROGRAM IDENTIFICATION NUMBERS (CPIN)

CATEGORY 81 - AIRCRAFT

A OPERATIONAL FLIGHT

ARN101 - AN/ARN-101 - NAVIGATION SET, LORAN

*** 81A-ARN101-F001-00A REV 11, 12 FEB 79 (U) ABD 00

OFP for NC of F-4E DMAS ARN-101, Provides navigation using INS/LORAN inputs, calculates various weapons delivery modes, 1 punched mylar tape, NO. PROG/CPC 1, LANGUAGE Assembly, OPERATOR MANUAL TO Not prepared, SUP COM IBM 360/65, SUP PROG Assembler, Linkage Editor, Simulator, Translator, Mag Tape Utility
APPL SYS: F-4E.
APPL SUBSYS: AN/ARN-101.

*** 81A-ARN101-F001-00D REV 11, 12 FEB 79 (U) ABD 00

DOCUMENTATION PACKAGE CONTAINS: Computer Program Development Spec for F-4E NC CB1001-0, Computer Program Product Spec for F-4E NC DD1001-0, Programmers Notebook, Version Description Document, Configuration Index, Change Status Report, Computer Program Manual SE, Cat I Test Plan/Procedures SE, Cat I Test Report SE, Test Requirements Document, Spec Change Notice

81A-ARN101-F002-00A REV 3, 2 FEB 79 (U) D 00

OFP for NC or RF-4C DMAS ARN-101, Provides navigation using INS/LORAN inputs, performs reconnaissance functions, 1 punched mylar tape, NO. PROG/CPC 1, LANGUAGE Assembly, OPERATOR MANUAL TO Not prepared, SUP COMP IBM 360/65, SUP PROG Assembler, Linkage Editor, Simulator, Translator, Mag Tape Utility
APPL SYS: RF-4C.
APPL SUBSYS: AN/ARN-101.

81A-ARN101-F002-00D REV 3, 2 FEB 79 (U) D 00

DOCUMENTATION PACKAGE CONTAINS: Computer Program Development Spec for RF-4C NC CB1001-00, Computer Program Product Spec for RF-4C NC CC1001-00, Programmers Notebook, Version Description Document, Configuration Index, Change Status Report, Computer Program Manual SE, Cat I Test Plans/Procedures SE, Cat I Test Report SE, Test Requirements Document, Spec Change Notice

ASN91 - AN/ASN-91(V) - COMPUTER SET, TACTICAL

* 81A-ASN91-U001-00A (U) B OC

Control, EQUIP/UUT ID 216-01940-1, 6870000-11, 6870200-8, TCS, Memory diagnostics test, signal diagnostics, I/O diagnostics repair verification test, 1 punched tape, NO. PROG/CPC 1, LANGUAGE TC-2 Assembler, CTL COMP/TEST STA AN/ASM-403(V)1 IBM, OPERATOR MANUAL TO 5N5-13-13-8-1, OFFLINE
APPL SYS: A-7D.
APPL SUBSYS: AN/ASN-91(V).

B ELECTRONIC WARFARE

A(MX6770A) - MX-6770A/A - INTERFERENCE BLANKER

** 81B-A(MX6770A)-U001-00A REV 6, 4 FEB 79 (U) D WR

Checkout Interference Blanker, EQUIP/UUT ID 1E4000G3 Interference Blanker, Test interference blanker P/N 1E4000G3 on GPATS, 1 mylar tape, NO. PROG/CPC 1, LANGUAGE Hexadecimal, CTL COMP/TEST STA GSM204-V6 GPATS, UUT INTERFACE TEST ADAPTER 7234476, OPERATOR MANUAL TO 51P7-2-2-8-3, OFFLINE
APPL SYS: F-111A, F-111E.
APPL SUBSYS: MX-6700A/A.

CATEGORY 81 - AIRCRAFT

PART II - CANCELLED SOFTWARE

81J-ASB9A(ASB16)-U001-00A
81J-ASB9A(ASB16)-U001-00D
81K-ASG19-U001-00A
81K-ASG19-U001-00D
81K-MA1/ASQ25FDT-T001-00A
81L-GSM24-U026-00A
81M-PNN16-U009-00A
81M-PNN16-U009-00D
81Q-VBC(APN16)-T003-00A
81Q-VBC(APN16)-T003-00D

PART III - RESCINDED SOFTWARE

81J-ASB-4A(ASB9A)-U001-00A
81J-ASB-4A(ASB9A)-U001-00D
81K-ASG24-U004-00A
81K-ASG24-U004-00A

PART IV - REIDENTIFIED CPINS

ORIGINAL CPIN	NEW CPIN
81J-ASB4A(ASB9A)-S022-00A	81M-ASB4A(ASB9A)-S002-00A
81J-ASB4A(ASB9A)-S022-00D	81M-ASB4A(ASB9A)-S002-00D

CPIN/AFCRI DATA AND CONTROL RECORD – PART I

1. TO: OC-ALC/MMEDU
TINKER AFB, OK 73145
(THROUGH CHANNELS)

2. CONTROLLING ALC/MMEC
CONTACT NAME/ADDRESS

3. [260] REQUESTING AGENCY
CONTACT NAME & ADDRESS

4. SECURITY CLASSIFICATION
CPCI TITLE

5. DATE PREPARED

PHONE
MMEC CONTROL NO.

PHONE
AGENCY CONTROL NO.

6. [110] TYPE ACTION (CIRCLE ONE)

A–NEW CPCI	F–NEW CPCI VER	K–NEW CPCI REV	Q–ADD DATA
B–NEW DOC	G–NEW DOC VER	L–NEW DOC REV	T–DELETE DATA
D–NEW CPCI/DOC	H–NEW CPCI/DOC VER	M–NEW CPCI/DOC REV	V–REPLACE DATA
E–CANCEL	J–RESCIND	P–REINSTATE	W–INQUIRY (AFCRI)

Z–OTHER (EXPLAIN IN REMARKS)

7. 040 TYPE SOFTWARE
- [ ] F - OPERATIONAL
- [ ] S - SUPPORT
- [ ] T - TEST
- [ ] U - UUT

8. CURRENT NUMBERS REV
CPCI
DOC

9. NEW NUMBERS ASSIGNED REV
DOC
TO USER MANUAL

10. REMARKS

11. [100] [110] DOC IDENT | TO | FROM | CAT | CPIN NUMBER | ACT | REV NO

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34

H

12. [080] MAJOR FUNCTION

13. [030] SOFTWARE GROUP

14. TR CODE | PROG | CLASS | MANAGER CODES: SM/IM | PRIME ALC/MMEC | COMMAND: CD, ROUTING | TYPE | SERIES | JOINT SERVICES | NO FOREIGN | OFFICIAL USE | SYMBOLS: A L D E H M F P | RESERVED

35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80

J A N | P | A | H

15. TR CODE | SUBSYSTEM/SYS ID NO | CPCSB DATE | CPCI COST | LEVEL | RESERVED

P 0 1

16. TR CODE | SUBSYSTEM/SYSTEM TITLE

P 0 2

17. TR CODE | CPCI PROGRAM TITLE

P 0 4

P 0 5

18. [070] TR CODE | EQUIPMENT/UUT IDENTIFICATION

P 0 7

19. TR CODE | CPCI PROGRAM DESCRIPTION

P 0 9

P 1 0

20. [100] [200] TR CODE | UNITS | TYPE MEDIA | PROG LANGUAGE

35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80

P 1 4

21. [300] TR CODE | CONTROL COMPUTER TEST STATION IDENTIFICATION

P 1 5

22. TR CODE | UUT INTERFACE TEST ADAPTER | TO USERS MANUAL

P 1 6

AFLC FORM OCT 78 505 PREVIOUS EDITION IS OBSOLETE

23. TR CODE | ON-OFF LINE | SUPPORT COMPUTER IDENTIFICATION

310
320

35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80

P 1 7

24. TR CODE | SUPPORT PROGRAMS REQUIRED

P 1 8

25. TR CODE | SPECIAL NOTES

P 2 0

P 2 1

26. TR CODE | ALL APPLICABLE SYSTEMS

060

P 2 5

P 2 6

27. TR CODE | ALL APPLICABLE SUBSYSTEMS

060

P 2 8

P 2 9

28. TR CODE | DESCRIPTION OF MEDIA UNITS

35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80

P 3 2

P 3 3

P 3 4

P 3 5

29. TR CODE | DOCUMENTATION PACKAGE

R 0 1

R 0 2

R 0 3

R 0 4

R 0 5

R 0 6

30. CONTROL COMPUTER

OPERATING SYSTEM	MILITARY DESIGNATION	D.P. INSTL	ADPE PLN
3 1 0	3 2 0	3 4 0	3 5 0

NATIONAL STOCK NUMBER (NSN)	MEMORY SIZE
3 3 0	3 6 0

31. CONTINUED DATA YES ________ NO ________ NUMBER OF PAGES ____________

32. OC-ALC/MMEDU ACTION

☐ NUMBERING ACTION COMPLETED

☐ RETURNED FOR

OC-ALC/MMEDU CONTACT | DATE

CPIN/AFCRI DATA AND CONTROL RECORD - PART II

DATE	PRIME ALC/MMEC	MMEC CTL NO	NUMBER OF PROGRAMS	ACCESSION NUMBER
			OF	0 1 0

A - SUBMIT ONE FORM FOR EACH CPCI COMPUTER PROGRAM

COMPUTER PROGRAM

TITLE

0 2 0

ACRONYM	LOCAL I.D. NUMBER	ENTRY-DATE
1 3 0	1 4 0	1 5 0

STATUS	VERSION	REVISION	VALIDATION DATE
1 6 0	1 7 0	1 8 0	1 9 0

CONTRACTOR

NAME

2 3 0

NUMBER	CONTRACT NUMBER
2 4 0	2 5 0

ABSTRACT

3 7 0

AFLC FORM OCT 78 506 PREVIOUS EDITION IS OBSOLETE.

CPIN DATA (CONTINUED)

	TR CODE																																												
35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79	80
35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79	80

APPENDIX B. TYPICAL POSITION DESCRIPTION OF AISF PERSONNEL DEVELOPING AND INTEGRATING SOFTWARE CHANGES

Below is the official position description for a GS-12 Electronic Engineer (Computer Systems). This description outlines the basic requirements of the work to be done, whether performed by Civil Service or contractor personnel.

I. INTRODUCTION

Incumbent of this position serves as an Avionics System Engineer responsible for accomplishing software and systems engineering projects/tasks for avionics embedded computer systems, their resident Operational Flight Program (OFPs) and their support systems for assigned prime aircraft systems.

II. DUTIES AND RESPONSIBILITIES

1. Develops, coordinates and carries through to completion blocks of work of large scope containing many phases of which two or more phases each contain several complex features. Plans and conducts research, development, or other work for which precedent data, criteria, methods or techniques are significantly inadequate, are controversial, or contain critical gaps. Develops or originates completely new features, in additon to improving, extending, or validating currently known precedents, data methods or techniques. In accomplishing the above, incumbent is responsible for the development of modifications and changes to complex aircraft digital avionics systems, their Operational Flight Program (OFPs), and laboratory support systems (e.g., Avionics Integration Support Facility (AISF) software. In addition, incumbent is responsible for the investigation, analysis, evaluation and reporting on avionics system performance, problems and new requirements.

2. Develops and carries through to completion complex changes to the OFPs. Uses the AISF to analyze and evaluate OFP requirements in order to develop optimum implementation. Investigates potential solutions to system problems/change requirements considering tradeoff analyses involving implementation costs, algorithm developments, timing requirements, memory size, hardware/software integration requirements, support equipment, personnel capabilities and limitations, data package development and overall magnitude of the effort; and translates these change requirements into engineering specifications and tasks. Designs the change mechanization and integration; develops the programming code; and debugs, tests and documents the results. At all times assures aircraft system integrity and compatibility; and meets resource allocations, performance criteria, cost and schedule.

3. Establishes formal test requirements for OFPs; develops and implements test plans; conducts detailed tests using the full capabilities of the AISF and instrumented flight test aircraft; and analyzes, evaluates and reports test results.

4. Serves as project engineer for the design and development of changes and modifications to the AISF hardware/software resources and other avionics support systems. Provides system engineering support and assures compatibility with the aircraft avionics, digital computer complexes and OFPs. Establishes change requirements directly with the AISF and avionics support systems users. Prepares change specifications, and plans and schedules the complete development and implementation.

5. Conducts studies and evaluations of systems in acquisition and determines support requirements. Performs studies, prepares Computer Resources Integrated Support Plans (CRISPs) and participates as a member of Computer Resources Working Groups (CRWGs).

6. Prepares contractual engineering proposals and associated specifications and work orders.

7. Monitors and maintains close liaison between contractor and Air Force activities associated with the engineering support of digital avionics, embedded computer systems and OFPs for prime aircraft systems.

8. Reviews, evaluates and advises on the effectiveness, technical adequacy and suitability of work and proposals of others related to digital avionics and OFP support. Evaluates more complex vendor proposed modifications for requirements, feasibility, completeness, accuracy, cost, and operational and logistics impact.

9. Consults, coordinates and attends conferences with other service activities and higher headquarters on matters pertaining to avionics OFP development and support. Makes recommendations to higher authority for changes to policies and practices, based on knowledge, experience, engineering studies, observations, and reports received from service activities, and defends ALC findings and recommendations. Travels to contractor or other government facilities to review engineering data and render opinions and decisions which are normally unreviewed; maintains liaison with other government activites and contractors in order to exchange engineering data and to maintain a current knowledge of the state-of-the-art.

10. Independently determines logical approach to solutions of major associated avionics OFP development and support problems. Carefully weighs the advantages of increased systems reliability, maintainability, etc., against time, cost, compatibility, and safety of flight. Makes and evaluates proposed changes to the system software on the basis of established hardware/software interfaces. Establishes supporting projects with other engineering personnel and directs the integration of auxiliary projects toward the ultimate objective. Scope of project effort is broad in that all projects consider, as applicable: the mission of the aircraft; functions of associated avionics systems (weapon delivery, navigation, reconnaissance, radar, instrumentation, etc.); communication/interface requirements; flight test; computer program documentation and configuration control; and validation/verification of the software. Applied research, special investigations, statistical analysis, etc., are a normal part of the incumbent's effort in accomplishing his duties and responsibilities.

III. CONTROLS OVER WORK

Incumbent is under the supervision of the Section Chief and receives technical direction from the functional group engineers and other senior engineers who give assignments in terms of broad, general objectives and relative priority of work. Extent and limits of assignments are mutually discussed. Incumbent works with considerable freedom from technical control in selecting and establishing the proper methods for attacking and resolving complex features and otherwise carrying assignments through to completion. Controversial policy questions are resolved by joint consideration with the supervisor and functional group engineer. Completed work is reviewed for adequacy in terms of broad objectives of the work and for compliance with Air Force policies and regulations. Decisions and recommendations based upon application of standard engineering practices are rarely changed by higher authority, except for reasons of policy, public relations, or budgetary consideration.

IV. OTHER SIGNIFICANT FACTS

1. Fields of Engineering: Electronic - 55%, Computer Science - 30%, Aerospace - 15%

2. In addition to an extensive academic and professional knowledge of scientific and engineering principles, it will be necessary for the incumbent to possess a special faculty to do successful applied research and establish authoritative criteria based on sound engineering principles used within this discipline by joint consideration with other engineers. At most times, the incumbent will be responsible for several projects requiring difficult and advanced engineering work of a high degree of

originality, therefore incumbent must have a thorough and detailed knowledge of: avionics digital systems, (e.g., inertial navigation systems, fire control radars, stores management systems; digital controls and displays, etc.); aircraft embedded computer systems; real-time operational flight software; laboratory support systems to include real-time simulation systems, host computer systems and avionics system hot mock-ups; software configuration management; software documentation; OFP testing, evaluation, verification and validation; and aircraft performance and operation, specifically in the areas of navigation and weapon delivery. Must be experienced and knowledgeable in real-time programming, mathematical modeling, computer architecture and programming languages.

3. Incumbent must possess a high degree of professional judgment, skill, initiative, planning and leadership ability. Also must possess ability to maintain effective personal work relationships at all levels and to justify and sell his own professional viewpoints in conferences, engineering reviews and with fairly large groups wherein conflicting points of view are represented. Requires an intimate knowledge of functions, organizational structure, jurisdictional responsibilites, etc., of USAF and elements thereof.

4. The incumbent of this position must be capable and willing to perform TDY travel in accordance with the Joint Travel Regulation.

5. Supports and takes affirmative actions in furtherance of Equal Employment Opportunity in all aspects of personnel actions, with special emphasis on Upward Mobility and other special programs.

6. Position requires a security clearance of Secret.

7. Performs other related duties as required.

8. Subject to call during off-duty hours.

9. All personnel will share in the responsibility for a sound industrial safety program. Incumbent is required to comply with all applicable safety directives. Unsafe conditions are to be promptly reported to the immediate supervisor.

LIST OF ACRONYMS

AFB	Air Force Base
AFCRI	Air Force Computer Resources Inventory
AFIT	Air Force Institute of Technology, Wright-Patterson AFB, Ohio
AFLC	Air Force Logistics Command
AFWAL/AA	The Avionics Laboratory of the Air Force Wright Aeronautical Laboratories
AISF	Avionics Integration Support Facility
ALC	Air Logistics Center
ATD	Aircrew Training Device
ATE	Automatic Test Equipment
C^3	Communications, Command & Control
CDR	Critical Design Review
CE	Communication Electronics
CI	Configured Item
CPCI	Computer Program Configuration Item
CPCP	Computer Program Change Proposal
CPCSB	Computer Program Configuration Sub-Board
CPIN	Computer Program Identification Number
CRISP	Computer Resources Integrated Support Plan
D/MM	Director of Material Management
DSARC	Defense Systems Acquisition Review Council
ECCM	Electronic Counter Counter Measures
ECP	Engineering Change Proposal
ECS	Embedded Computer System

LIST OF ACRONYMS

EPROM	Erasable Programmable Read Only Memory
EW	Electronic Warfare
H/W	Hardware
ILSP	Integrated Logistic Support Plan
IR&D	Independent Research and Development
ITE	Integration Test Equipment
MIPR	Military Interdepartmental Purchase Request
MMEC	Computer Resource Branch within the Engineering Division of the Material Management Division at the ALCs
MSRD	Master Software Requirements Document
OC-ALC	Oklahoma City ALC, Tinker AFB, Oklahoma
OC-ALC/MATT	The Software Support Center at Oklahoma City ALC
OC-ALC/MMECZA	The portion of OC-ALC/MMEC located at China Lake Naval Weapons Center, California
OC-ALC/MMEDU	Technical Order Section of the Operations and Support Branch at Oklahoma City ALC
OO-ALC	Ogden ALC, Hill AFB, Utah
OO-ALC/ACDCS	Ogden ALC Comptroller organization providing programming support for support software
OO-ALC/MMECA	Branch responsible for OFP change design and development at Ogden ALC.
OO-ALC/MMETA	Branch providing validation and verification of OFP changes at Ogden ALC.
OT&E	Operational Test and Evaluation
PDR	Preliminary Design Review
PERT	Program Evaluation and Review Technique
PMRT	Program Management Responsibility Transfer

LIST OF ACRONYMS

PSCM	Predictive Software Cost Model
RADC	Rome Air Development Center, Griffiss AFB, New York
RAM	Random Access Memory
RAND	The Rand Corporation, Santa Monica, California
RFP	Request for Proposal
RIW	Reliability Improvement Warranty
ROM	Read Only Memory
SA-ALC	San Antonio ALC, Kelly AFB, Texas
SA-ALC/MATT	Software Support Center at San Antonio ALC
SA-ALC/MMEC	Computer Resources Branch at San Antonio ALC
SA-ALC/MMIM	Logistics Management Branch at San Antonio ALC
SA-ALC/MMIR	Engineering and Reliability Branch at San Antonio ALC
SM-ALC	Sacramento ALC, McClellan AFB, California
SM-ALC/MMECF	Ground Communications, Electronics, and Meteorological Systems Support Branch at Sacramento
SM-ALC/MMECM	Software Management Branch at Sacramento ALC
SM-ALC/MMECP	F/FB-111 Support Branch at Sacramento ALC
SM-ALC/MMECS	Administration Branch at Sacramento ALC
SMART	Simple Multi-Attribute Ranking Technique
SPDD	System Program Description Document
S/W	Software
TCTO	Time Compliance Technical Order
TEWS	Tactical Electronic Warfare System
TO	Technical Order

LIST OF ACRONYMS

UUT	Unit Under Test
V&V	Validation and Verification
VDD	Version Description Document
WPAFB	Wright-Patterson Air Force Base
WR-ALC	Warner-Robins ALC, Robins AFB, Georgia
WR-ALC/MMEC	Computer Resources Branch of Warner-Robins ALC
WR-ALC/MMECA	ATE Acquisition Organization at Warner-Robins ALC
WR-ALC/MMECD	Weapon System Integration Organization at Warner-Robins ALC
WR-ALC/MMR	Electronic Warfare Management Branch at Warner-Robins ALC
WR-ALC/MMRR	Engineering and Reliability Branch for EW Systems at Warner-Robins ALC

*U.S.Government Printing Office: 1980 — 657-084/784