1986

A decision support system for planning, control and auditing of DOD software cost estimation

Sullivan, Anne Nell Neely.;Darabond, James M.

NAVAL POSTGRADUATE SCHOOL
Monterey, California

THESIS

A DECISION SUPPORT SYSTEM FOR PLANNING, CONTROL AND AUDITING OF DOD SOFTWARE COST ESTIMATION

by

Anne Nell Neely Sullivan

and

James M. Darabond

March 1986

Advisors: Tung Bui
Norman R. Lyons

Approved for public release; distribution is unlimited.

CURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

. REPORT SECURITY CLASSIFICATION	1b. RESTRICTIVE MARKINGS
. SECURITY CLASSIFICATION AUTHORITY	3 DISTRIBUTION / AVAILABILITY OF REPORT
. DECLASSIFICATION / DOWNGRADING SCHEDULE	Approved for public release; distribution is unlimited
PERFORMING ORGANIZATION REPORT NUMBER(S)	5. MONITORING ORGANIZATION REPORT NUMBER(S)

. NAME OF PERFORMING ORGANIZATION	6b. OFFICE SYMBOL *(If applicable)*	7a. NAME OF MONITORING ORGANIZATION	
aval Postgraduate School	54	Naval Postgraduate School	
ADDRESS *(City, State, and ZIP Code)*		7b. ADDRESS *(City, State, and ZIP Code)*	
onterey, CA 93943		Monterey, CA 93943	
NAME OF FUNDING / SPONSORING ORGANIZATION	8b. OFFICE SYMBOL *(If applicable)*	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER	
ADDRESS *(City, State, and ZIP Code)*		10. SOURCE OF FUNDING NUMBERS	
		PROGRAM ELEMENT NO.	PROJECT NO. / TASK NO. / WORK UNIT ACCESSION NO.

TITLE *(Include Security Classification)*

DECISION SUPPORT SYSTEM FOR PLANNING, CONTROL AND AUDITING OF DOD SOFTWARE COST STIMATION

PERSONAL AUTHOR(S)

ULLIVAN, Anne Nell Neely and DARABOND, James M.

TYPE OF REPORT	13b. TIME COVERED	14 DATE OF REPORT *(Year, Month, Day)*	15 PAGE COUNT
aster's Thesis	FROM _____ TO _____	1986 March	321

SUPPLEMENTARY NOTATION

COSATI CODES			18. SUBJECT TERMS *(Continue on reverse if necessary and identify by block number)*
FIELD	GROUP	SUB-GROUP	Decision Support System, DDS COCOMO, Software Engineering

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

hesis takes an overivew of past software development estimation problems and practices, urveys the present situation, and provides recommendations. Results from a Department of efense (DOD) wide survey on software development estimation factors are examined for rends using statistical analysis techniques.

asic and Intermediate models of the Constructive Cost Model (COCOMO) are implemented using software engineering approach for development and maintenance estimations. Documentation or this DSS is contained in the appendices.

. DISTRIBUTION / AVAILABILITY OF ABSTRACT	21. ABSTRACT SECURITY CLASSIFICATION	
☒ UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT. ☐ DTIC USERS	unclassified	
. NAME OF RESPONSIBLE INDIVIDUAL	22b. TELEPHONE *(Include Area Code)*	22c. OFFICE SYMBOL
orman F. Lyons	408 646-2666	54Lb

) FORM 1473, 84 MAR 83 APR edition may be used until exhausted
All other editions are obsolete

SECURITY CLASSIFICATION OF THIS PAGE

Approved for public release; distribution is unlimited.

A Decision Support System for Planning, Control, and Auditing of DOD Software Cost Estimation

by

Anne Nell Neely Sullivan
Lieutenant Commander, United States Navy
B.A., University of Kansas, 1973
M.B.A., Bryant College, 1979

and

James M. Darabond
Lieutenant, United States Navy
B.S., University of Utah, 1978

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN INFORMATION SYSTEMS

from the

NAVAL POSTGRADUATE SCHOOL
March 1986

ABSTRACT

This thesis takes an overview of past software development estimation problems and practices, surveys the present situation, and provides recommendations. Results from a Department of Defense (DOD) wide survey on software development estimation factors are examined for trends using statistical analysis techniques.

Basic and Intermediate models of the Constructive Cost Model (COCOMO) are implemented using a software engineering approach for development and documentation. This Decision Support System (DSS) is developed as a prototype for possible use in DOD for software development and maintenance estimations. Documentation for this DSS is contained in the appendices.

THESIS DISCLAIMER

The reader is cautioned that computer programs developed in this research may not have been exercised for all cases of interest. While every effort has been made, within the time available, to ensure that the programs are free of computational and logic errors, they cannot be considered validated. Any application of these programs without additional verification is at the risk of the user.

TABLE OF CONTENTS

LIST OF TABLES

I. BACKGROUND AND INTRODUCTION

The Department of Defense spends billions of dollars annually on computer processing resources. The majority of these resources are for maintaining and developing software [Ref. 1]. Software development and maintenance costs are difficult to control. Many software projects have been behind schedule, over cost, and short of expectations. In an attempt to improve software cost estimation in the Department of Defense (DOD), the Office of the Secretary of Defense (OSD) selected the automated Software Life Cycle (SLIM) cost estimation model as the tentative DOD methodology. The OSD funded a DOD-wide license for SLIM as well as training sessions in the use of SLIM at the Department of Defense Computer Institute (DODCI) [Ref. 2].

Computer software costs are increasing while hardware costs continue to drop. Many software cost estimation models exist to assist a manager with costs and schedules for software development projects. These models vary in accuracy and completeness. Variances are due to difficulties in software cost estimation. Difficulties include inaccurate date, missing information, problems with contractors, lack of automated tools and inappropriate cost drivers. Accuracy of software cost predictors is of extreme import to the Navy due to the number of dollars involved and

the external budgetary constraints. A project manager for software development bears numerous responsibilities which permit little or no time for computation and evaluation of detailed cost/schedule alternatives. Thus, reliable data is essential for project planning and control. Not only must a software cost estimation model have an acceptable degree of accuracy, but also, it must be flexible enough to adapt to a rapidly changing environment. Software cost/schedule packages currently available on the market are based on data bases that would not fit the DOD types of software and do not allow for adjustment of various cost drivers to fit the changing nature of the software development world. It is, therefore, necessary to have an adaptable, circumstance-shaped Decision Support System to remedy these shortcomings.

Besides SLIM, there are numerous other software cost estimation models used by the DOD. The RCA PRICE S model is frequently used in U. S. Air Force software cost estimates. Barry Boehm's Constructive Cost Estimation Model (COCOMO) was recently selected for use in estimating costs of developing the WIS (World Wide Military Command and Control System Information System) in the Ada language [Ref.3].

Boehm's model estimates both development and maintenance costs. Both estimates are of high budgetary interest. Maintenance makes up the major portion of the software life cycle effort [Ref. 4]. Boehm's COCOMO model has three

levels: Basic, Intermediate, and Detailed. Their ability to estimate within 20% of project actual is 25%, 68%, and 70% respectively [Ref. 5]. The COCOMO estimates were based on a limited dispersal of software projects. To increase its predictive capability with Department of Defense software, it would be important to better understand the characteristic and profile of the set of software under study.

Appropriate use of COCOMO requires two prerequisites: a good estimation of KDSI, and a good knowledge of the profile of the cost drivers. Lines of code are difficult to estimate. In the past, errors in estimating size have been as large as a factor of three [Ref. 6]. However, the COCOMO model could prove valuable to the DOD in numerous cases where software must be converted from one language to another as lines of code estimates are more accurate on software conversions than on new developments. Another application for the Intermediate COCOMO model is in conjuction with other cost estimation models. A cost estimation model which does not require the number of lines of code as an input, such as Estimax, could be applied first. Resultant estimated lines of code could then be used as an input to the Intermediate COCOMO model. A third application would be as a cross-validator for other cost estimation models.

The objectives of this thesis are to design, develop, and implement a Decision Support System for the Basic and Intermediate models, and to better understand the nature of the DOD software. A survey will be used to gain a better understanding of DOD software development estimation. Survey results are examined for trends in Chapter III. Chapter IV addresses the Decision Support System (DSS) implementation of the Basic and Intermediate COCOMO models using a software engineering approach. Supporting documentation for the software, including the program listings and Users Manual, are found in the appendices.

II. COCOMO SOFTWARE COST ESTIMATION MODEL

This chapter summarizes the COCOMO Basic and Intermediate software cost estimation models. Maintenance, which is used with both the Basic and Intermediate models, is also discussed.

A. OVERVIEW

The Basic COCOMO model takes as parameters the estimated number of source instructions (KDSI) and the development mode. The development mode parameter indicates what type of project is being developed, ranging from relatively small projects loosely coupled with their operating environment ("organic") to large, complex systems with rigidly specified interfaces, real-time performance constraints, and high reliability requirements ("embedded"). The Basic model calculates man-months of effort and months of schedule, along with productivity in number of delivered source instructions per man-month and annual development cost. For example, a typical result for a 2 KDSI project might be 6.6 man-months of effort required, 5.1 month schedule and approximately 301 required lines of codes per man-month. Distribution of effort and schedule are also calculated, e.g., of the 5.1 months of development time, the model will tell you that 0.97 months would be spend in product design, 3.23 in programming and unit testing, and 0.92 in

integration testing. Requirements analysis are not included in COCOMO estimates, however, product activity distribution by phase for effort is computed. For example, calculated product design for effort would be farther subdivided into requirements analysis, product design, programming, test planning, verification/validation, project office time, quality assurance and documentation development time. Likewise, programming and integration testing would also be subdivided into these same categories.

The Intermediate COCOMO model builds on the Basic model by adding cost drivers, which are measures of various attributes of the product, project, computer and personnel. The product of these cost drivers multiply the calculated effort man-months to produce an adjusted nominal man-month figure. For example, one driver (denoted PCAP) measures Programmer Capability. The PCAP multiplier can range from 0.70 (very high programmer capability) to 1.42 (very low programmer capability). In the example above, if very high quality programmers were available, the estimated development time would be reduced to 4.62 man-months (6.6 x 0.70) provided the rest of the cost drivers remained at a nominal value of 1.0. Cost drivers give a more comprehensive picture of the product and the environment in which it is to be developed, with resulting greater accuracy of prediction.

The COCOMO models are calibrated using data collected for 63 projects completed by TRW between 1964 and 1979. Numeric parameters were not determined solely by statistical curve fitting, but were influenced by the judgment of project managers. The Basic COCOMO model does not have particularly good accuracy; Boehm reports that estimates for the calibration data are within a factor of 2 of the actual effort only 60% of the time. The added parameters of cost drivers in Intermediate COCOMO give it much improved accuracy. Estimates with the Intermediate COCOMO model are within 20% of actual effort 68% of the time.

B. THE BASIC MODEL

The Basic model's parameters are estimated thousands of delivered source instructions (KDSI) and development mode. Source instructions are defined as lines of code, including declarative statements and job control language but excluding comments. Development modes are characterized as follows:

1. Organic
    a. Generally stable development environment
    b. Minimal need for innovation in architectures or algorithms
    c. Relatively small size
    d. Relatively low premium on early completion of the project

e. Software project range usually not greater than 50 KDSI

f. Loose coupling with external systems

2. Semidetached

a. Mixture of organic and embedded characteristics

b. Intermediate level of experience with related systems

c. Wide mix of experienced and inexperienced people

d. Some experience with aspects of system under development

e. Software project range usually not greater than 300 KDSI

3. Embedded

a. Much innovation required

b. Integral part of some larger system with inflexible

c. Interface requirements

d. High required reliability

e. Development within tight time and cost constraints

The basic effort development estimation formula by mode are:

```
Organic:          MM = 2.4(KDSI)**1.05

Semidetached:     MM = 3.0(KDSI)**1.12

Embedded:         MM = 3.6(KDSI)**1.20

where

   MM    = man-months of development effort

   KDSI  = estimated thousands of delivered
           source instructions
```

Another result obtainable from the Basic COCOMO model is development time, i.e., how many months the project will take to complete. These schedule formula by mode are:

Organic:	TDEV = 2.5(MM)**0.38
Semidetached:	TDEV = 2.5(MM)**0.35
Embedded:	TDEV = 2.5(MM)**0.32

where

TDEV = development time in months

MM = effort in man-months calculated above

Besides effort and schedule calculations other data which can be computed and are model and mode independent are:

Average number of personnel = MM/TDEV

Productivity = (1000*KDSI)/MM

Annual cost = Personnel cost/MM * MM

The Basic model also provides information on how the effort and schedule are distributed over the phases of the project. These tabulated percentages are a function of the product size and mode. The product sizes occur for standard KDSI values of 2, 8, 32, 128, and 512. KDSI values occurring between these standard figures are considered nonstandard and must have the closest lowest and highest percentages to it interpolated to produce the proper result. KDSI values below and above 2 and 512 KDSI respectively are beyond the boundaries of the COCOMO model and are not used

as the model formula for effort and schedule are calibrated only for this range. Values for the phase distribution of effort are computed by multiplying each percentage by the prior computed MM number. Phase distribution for schedule is also computed in a similar way except each schedule percentage is multiplied by the calculated TDEV value.

In addition to the phase distribution computations, activity distribution by phase can also be calculated. These percentages are again product and mode dependent and provide more detail about the product design, programming, and test integration values computed for phase distribution of effort. Calculation of the values for this area occurs by multiplying the man-months value obtained for phase distribution product design, programming, and test integration by the respective percentages under each appropriate column. For example, to obtain the values for activity distribution in the organic mode for product design, the product design value computed in the phase distribution would be multiplied by each percentage under the product design column to generate the necessary activity phase distribution for product design.

C. THE INTERMEDIATE MODEL

The key feature which the Intermediate model adds to the Basic model is a set of 15 cost driver attributes. These cost drivers have a default nominal value of 1.0, however, these values can be varied depending on the environment in

which the project is being created. The product of these 15 cost drivers is called the Effort Adjustment Factor (EAF).

Development modes for the Intermediate model are the same as those for the Basic model. However, the effort development estimation formula vary slightly from the Basic model and are:

Organic:	MMn = 3.2(KDSI)**1.05
Semidetached:	MMn = 3.0(KDSI)**1.12
Embedded:	MMn = 2.8(KDSI)**1.20

where

MMn = Nominal man-months

The cost drivers are factored in by multiplying the nominal man-months by the EAF:

MMadj = MMn * EAF

where

MMadj = man-months adjusted

Schedule formula by mode are the same as for the Basic model. Average number of personnel, productivity, annual cost, phase distribution of effort and schedule, and phase activity distribution are also computed in the same manner as for the Basic model.

For a large system it is likely that the cost driver values will vary for different parts of the system. Estimation accuracy can therefore be improved by dividing

the system into components. The nominal man-months are allotted to the components in proportion to their size, and the appropriate set of multipliers are then applied to each component separately. The resulting component estimates are then summed to obtain the overall system estimates.

D. MAINTENANCE MODEL

The process of modifying existing operational software while leaving its primary functions intact is defined as software maintenance. Calculations for the effort and annual cost of this maintenance are also performed in both the Basic and Intermediate models and are mode independent. A new term in this area is called the Annual Change Traffic (ACT). It is the fraction of the software product's source instructions which undergo change during a typical year, either through addition or modification. The value of this factor ranges between 1.00 for complete change to 0 for no change at all to the software. The formulae for ACT is:

$$ACT = \frac{\text{DSI added} + \text{DSI modified}}{\text{Total DSI}}$$

where

ACT = Annual change traffic

DSI = Delivered source instructions

Maintenance formula used with the Basic model are:

(MM)am = MM * ACT

Average maintenance personnel = (MM)am/12

Annual maintenance cost = Maintenance personnel cost/MM * (MM)am

where

(MM)am = Basic annual maintenance effort

MM = Effort in man-months

ACT = Annual change traffic

Calculations for the Intermediate model again vary slightly from the Basic model in that 14 maintenance cost drivers are used to increase the model accuracy. The value for each maintenance cost driver is defaulted to a nominal value of 1.00, but can be varied according to the environment. The product of these cost drivers is called the Maintenance Effort Adjustment Factor (EAFM). Formula for the Intermediate model are:

(MM)nam = MMn * ACT * EAFM

Average maintenance personnel = (MM)nam/12

Annual maintenance cost = Maintenance personnel cost/MM * (MM)nam

The product activity distribution by phase percentages are multiplied by either the annual maintenance effort, (MM)am, value in the Basic model or the nominal annual maintenance effort, (MM)nam, value in the Intermediate model to obtain the maintenance activity distribution by phase.

III. SURVEY RESULTS

A. SURVEY

The survey is designed to learn about the nature of Department of Defense (DOD) software and to be able to profile the cost drivers. The survey, Appendix A, requests service branch, nature of application, models/methods used for cost estimation, percent accuracy on cost, schedule and effort predictions, average size, and information which could be used to predict effort and schedule with the Intermediate COCOMO model. The Intermediate COCOMO model was chosen for the DSS tool because it is much less complicated than the detailed model and only 2% less accurate. The Intermediate model is similar to the Basic model miltiplied by an effort adjustment factor (EAF) which contains cost driver attributes.

The Department of Defense Computer Institute (DODCI) was contacted for a mialing list. The DODCI list was expanded telephonically and 107 surveys were mailed out. The number of surveys sent out by service was 6 Army, 25 Navy, 55 Air Force, 20 Marine Corps, and 1 Coast Guard. Some of the addressees copied and redistributed the survey to various software development shops. A total of 48 surveys were returned: 4 Army, 7 Navy, 23 Air Force, 11 Marine Corps, 1 Coast Guard, 1 Joint Service, and 1 survey of unknown service branch origin.

B. DELIVERED SOURCE INSTRUCTIONS (DSI)

The KDSI (thousands of DSI) size ranges from 1 to 7000 with four respondents not giving/guessing an average KDSI size. Apparently most average KDSI's were estimates since most respondents selected one of the standard KDSI sizes. Of the given KDSI's, 8 were outside of the 2-512 KDSI range required by the COCOMO model. Half of the outer values were too small and the other half were too large. One respondent gave the average KDSI as ranging anywhere from 2 to 512 KDSI, this response was defaulted to 32 KDSI. There were 36 responses giving KDSI's within the COCOMO range. If the sample is random and representative, then approximately 80% of the DOD software falls within the COCOMO model's effective range of prediction. For the total sample, the average KDSI was 342.13 with a standard deviation of 1191. For the sample within 2-512 KDSI, the mean was 84.722 with a standard deviation of 166.

1. KDSI Partitions

Table I partitions the data file. The partitions are used in subsequent files. Each table entry gives the partition description, count, mean, standard deviation and mean to standard deviation ratio.

TABLE I

KDSI MEANS, STANDARD DEVIATIONS, AND RATIOS BY PARTITIONS

PARTITION	N	MEAN	ST.DEV.	RATIO
Total Sample	44	342.13	1119.	0.306
2-512 KDSI	36	84.7	141.	0.606
Organic mode	13	69.1	137.	0.504
Semidetached	18	97.1	162.	0.599
Embedded	5	80.0	64.7	1.24
2-5 KDSI	10	3.00	1.33	2.25
75-128 KDSI	5	11.8	5.50	2.15
32-56 KDSI	10	34.4	7.59	4.53
8-20 KDSI	7	120.	20.0	6.00
2-5 KDSI	4	443.	129.	3.43

The table entries listed are for the total sample and the fraction of the sample with KDSI's within range for the COCOMO model. The latter is further partitioned by mode and by range of KDSI. Since the total sample represents a cross-sectrion of kinds and sizes of DOD software development applications, the larger samples are expected to have low ratios. The ratios for the organic and semidetached mode partitions are also low. This is quite likely due to computer software development still being a new field and earlier applications tending to be organic. One observation in the organic mode was an anomaly at 500 KDSI. In a few years, the ratios for KDSI in the organic

and semidetached modes will probably increase. Some of the survey respondents indicated an ongoing conversion of modes.

2. KDSI Partritions by Mode and by Service

Table II gives the KDSI partitions by mode and by service. The survey responses by service by mode for the partition with a KDSI within the range of the COCOMO model are: Army 100% in semidetached mode; Navy 50% organic and 50% semidetached; USAF 44.4% organic, 33.3% semidetached, and 22.2% embedded; USMC 37.5% organic, 50% semidetached, and 12.5% embedded. The one Coast Guard observation is in the semidetached mode as well as one observation from an unknown orginator. The N = 36 sample is 34.1% organic, 50% semidetached, and 13.9% embedded. While the total sample (N = 48) is 39.6% organic, 45.8% semidetached, and 14.6% embedded.

The very big KDSI (VBKDSI) partition, 250-512K, is 25% organic and 75% semidetached. The big KDSI (BKDSI) partition, 75-128K, is 28.6% organic, 28.6% semidetached and 42.9% embedded. The medium KDSI (MKDSI) range, 32-56K, is 30% organic and 70% semidetached. The small KDSI (SKDSI) range, 8-20K, is 60% organic, 20% semidetached and 20% embedded. The very small KDSI (VSKDSI), 2-5K, is 40%

TABLE II
KDSI PARTITIONS BY MODE AND BY SERVICE BRANCH

	VBKDSI	BKDSI	MKDSI	SKDSI	VSKDSI
MODE:					
Organic	1	2	3	4	4
Semidetached	3	2	7	1	5
Embedded	0	3	0	1	1
Total	4	7	10	6	10
SVBR:					
ARMY	0	1	2	0	1
NAVY	0	0	1	1	2
USAF	4	5	4	4	2
USMC	0	0	3	1	4
USCG	0	0	0	0	1
JOINT	0	0	0	0	0
OTHER	0	1	0	0	0

organic, 50% semidetached and 10% embedded. The VBKDSI is 100% USAF projects. This is very likely due to the small sample size. The BKDSI is 14.3% Army, 71.4% USAF and 14.3% unknown service. The MKDSI is 20% Army, 10% Navy, 40% USAF and 20% USMC. The SKDSI is 20% Navy, 60% USAF,and 20% USMC. The VSKDSI is 10% Army, 20% Navy, 20% USAF, 40% USMC and 10% USCG.

C. COST DRIVERS

The given cost drivers are assigned table values. Table III shows the cost driver rating data for the entire sample

while Table IV gives the data for the subset of the sample with average KDSI's within the 2-512K range. The average for each attribute is within one standard deviation of the nominal except complexity (CPLX) in both Tables III and IV and data in Table III.

TABLE III

COST DRIVERS FOR N = 48

ATTRIBUTE	N	MEAN	ST.DEV	RATIO
RELY	48	1.0994	0.156	7.05
DATA	42	1.0810	0.0708	15.3
CPLX	47	1.2002	0.173	6.94
TIME	46	1.1735	0.238	4.93
STOR	48	1.1316	0.176	6.43
VIRT	47	0.97660	0.113	8.64
TURN	46	0.96761	0.0785	12.3
ACAP	48	0.91312	0.149	6.13
AEXP	48	0.96896	0.0906	10.7
PCAP	48	0.90646	0.141	6.43
VEXP	47	0.98489	0.0907	10.9
LEXP	48	0.97583	0.0392	24.9
MODP	48	0.93708	0.0737	12.7
TOOL	48	0.97271	0.0948	10.3
SCED	47	1.0234	0.0337	30.4

For both the sample set and the subset, the required software reliability (RELY), data base size (DATA), product

complexity (CPLX), execution time constraint (TIME), main storage constraint (STOR), and required development schedule (SCED), are greater than nominal and will increase costs. Tactical/technology based computers would tend to drive the average required reliability (RELY) and complexity (CPLX) multipliers up. Increased data base size (DATA), higher required utilization of execution time (TIME) and main storage (STOR) would increase the initial outlay. The system would also be set up for possible capacity problems which could bring the system down at an inopportune time, possibly driving up long-range costs as well as development costs. A spreadsheet capacity problem was encountered with the software application written in conjunction with this thesis. In resolving the problem, the development time was at least doubled.

The average required schedule (SCED) is slightly longer than nominal. If the spare time is spent on improving documentation, this is a benefit, since the quality of documentation impacts the subsequent maintenance costs.

The mean virtual machine volatility (VIRT) is rated lower than nominal which means that the frequency of major machine changes in the DOD is slower. Some respondents needed an even lower response category. Since Boehm's tables only go down to low for VIRT, the low category was selected for lack of a better option.

Computer turnaround time (TURN) is lower than the nominal, probably due to the number of interactive machines. Perhaps there should be two different drivers/scales for batch and interactive machines. All interactive machines had the same ratings. Yet, tactical interactive computers tend to have turnaround constraints that drive up costs.

Analyst capability (ACAP), applications experience (AEXP), programmer capability (PCAP), virtual machine experience (VEXP), and programming language experience (LEXP) all averaged above nominal which would lower costs. The programmer capability (PCAP) table may need adjustment to account for the sometimes magnitude of difference in programmer productivity. The use of modern programming practices (MDOP) and the use of software tools (TOOL) both averaged slightly higher than nominal which would reduce costs.

Table IV is within one standard deviation of all values in Table III. The DATA attribute is within one standard deviation of the nominal for Table IV, unlike Table III. The difference may be due to projects greater than 512 KDSI not being included in Table IV.

The only other notable difference between the two tables is that the ratio for programming language experience is much lower for the total sample. There was a lower standard deviation in LEXP in the 2-512 KDSI range partition. The lower demoninators lead to lower ratios.

TABLE IV

COST DRIVERS FOR 2-512 KDSI OBSERVATIONS

ATTRIBUTE	N	MEAN	ST.DEV	RATIO
RELY	36	1.0997	0.164	6.71
DATA	33	1.0739	0.0761	14.1
CPLX	35	1.1823	0.155	7.63
TIME	34	1.1750	0.235	5.00
STOR	36	1.1292	0.166	6.80
VIRT	35	0.97486	0.112	8.70
TURN	34	0.95735	0.0783	12.2
ACAP	36	0.91167	0.164	5.56
AEXP	36	0.97417	0.0971	10.0
PCAP	36	0.91139	0.150	6.08
VEXP	35	0.99400	0.0967	10.3
LEXP	36	0.97028	0.0287	33.8
MODP	36	0.93306	0.0706	13.2
TOOL	36	0.97556	0.102	9.56
SCED	35	1.0229	0.0337	30.4

D. EFFORT ADJUSTMENT FACTOR (EAF)

Table V gives the average EAF for the various data partitions. In obtaining the average adjustment factors, all values not given were recorded as '* '. This modification is equivalent to assuming all unknown cost drivers to be nominal. The average effort adjustment factor was 1.4239, with a standard deviation of 1.14, for the 48

sample survey and 1.4236, with a standard deviation of 1.21, for the 36 sample portion of the survey.

TABLE V

EAF MEANS, STANDARD DEVIATIONS, AND RATIOS BY PARTITIONS

PARTITION	N	MEAN	ST.DEV.	RATIO
Total Sample	48	1.4239	1.14	1.25
2-512 KDSI	36	1.4236	1.21	1.18
Organic Mode	13	1.3670	1.02	1.34
Semidetached	18	1.4365	1.38	1.04
Embedded	5	1.5238	1.22	1.25
2-5 KDSI	10	0.83053	0.467	1.78
8-20 KDSI	5	1.5356	1.32	1.16
32-56 KDSI	10	2.1718	1.79	1.21
75-128 KDSI	7	1.1188	0.446	2.51
250-512 KDSI	4	1.4286	0.868	1.65
32 KDSI	9	2.2710	1.87	1.21
128 KDSI	6	1.1678	0.467	2.50

The average effort adjustment factor was 1.4239 with a standard deviation of 1.14 for the 48 sample survey and 1.4236 with a standard deviation of 1.21 for the 36 sample portion of the survey. Nevertheless, when the data base was partitioned by mode and by KDSI, the size of the standard deviation decreased. There was a slight increase in the mean as the partition modes went from organic to

semidetached to embedded. The partitions also overlapped one standard deviation away from the mean.

The EAF for the very small KDSI range, 2-5K, was significantly smaller than for the other larger KDSI ranges. There was an increase in average EAF's with increase in KDSI until the 75-128K range, then the average EAF dropped before increasing again with the number of KDSI. The small sample sizes probably attribute to mid-KDSI range peak in EAF. The 32-56 KDSI sample included an observation with the highest EAF and another extremely high EAF. The small KDSI sample also had two out of five EAF's very high. A contributing factor to the dispersion of high EAF's with respect to the KDSI might be correlated with the KDSI range of certain natures of software applications. The 32 KDSI sample includes 9 of the 10 observations in the mid-KDSI range and has an EAF larger than the 128K, large KDSI range, for the same reasons.

E. NOMINAL MAN-MONTHS

Table VI give the nominal man-month data by partitions. Nominal man-months is the number of man-months of effort required given all cost drivers are nominal. The Appendix B data definition of MMnom gives the equation for effort for each mode. Since KDSI is needed to compute MMnom, the sample for this section has 36 observations.

Of the 36 observations, 39.6% were in the organic mode, 45.6% in the semidetached mode and 14.6% in the embedded mode.

The average nominal effort increased as the mode went from organic to semidetached to embedded. The most marked increase for the partition by mode is between the organic and the semidetached modes.

TABLE VI

MMNOM DATA BY PARTITIONS

PARTITION	N	MEAN	ST.DEV.	RATIO
2-512 KDSI	36	470.24	830.	0.567
Organic mode	13	289.02	597.	0.484
Semidetached	18	569.02	1035.	0.550
Embedded	5	585.79	493.	1.19
2-5 KDSI	10	10.486	5.31	1.97
8-20 KDSI	5	46.826	24.2	1.93
32-56 KDSI	10	151.07	44.1	3.43
75-128 KDSI	7	706.67	241.	2.93
250-512 KDSI	4	2533.1	877.	2.89
32 KDSI	9	137.6	11.9	11.6
128 KDSI	6	761.49	211.	3.61

A much more dramatic increase in nominal effort occurred with the increase in KDSI ranges thus indicating that KDSI has a greater impact on nominal effort than mode. The 32 KDSI sample had the highest ratio. It was a 9 observation

sample with uniform KDSI and two-thirds of the observations in the semidatached mode while the other third of the observations were in the organic mode.

F. EFFORT (MAN-MONTHS (MM))

Table VII gives the effort data by partitions. Effort or MM is equal to the nominal man-months multiplied by the effort adjustment factor.

Again the average effort and standard deviations increased markedly with an increase in the KDSI range. The increases were at a much greater rate for MM than the increases for MMnom due to the multiplier effect of the cost

TABLE VII

EFFORT DATA BY PARTITIONS

PARTITION	N	MEAN	ST.DEV.	RATIO
2-512 KDSI	36	696.54	1475.	0.472
Organic mode	13	473.92	1170.	0.404
Semidetached	18	847.11	1828.	0.463
Embedded	5	773.29	669.	1.16
2-5 KDSI	10	8.1730	4.68	1.75
8-20 KDSI	5	77.932	99.9	0.780
32-56 KDSI	10	316.66	250.	1.27
75-128 KDSI	7	824.41	500.	1.65
250-512 KDSI	4	3916.6	2847.	1.38
32 KDSI	9	313.14	265.	1.18
128 KDSI	6	909.85	488.	1.86

drivers, or to the EAF. The increase in standard deviations caused a decrease in the ratios.

In the partitions by mode, the effort increased from organic to semidetached, then dropped off some from semidetached to embedded. The ratios increase as the mode went from organic to semidetached to embedded.

G. SCHEDULE (TDEV)

Table VIII gives the schedule by data partitions. The equations for the number of months required for development are in the Appendix B data dictionary under the entry for Schedule. There is a separate equation for each mode. MM is needed to compute TDEV, thus the total possible observations are 36.

The mean schedule and the standard deviation both increased significantly with an increase in the KDSI range partitions. The greater rate of increase is caused by a multiplier effect. TDEV is computed from MM which is in turn computed from KDSI.

The 128K partition and the high KDSI partition, whose 7 observations contained the six 128K observations, had the highest ratios. The high KDSI range contained several embedded observations while the very high KDSI range, 250-512K, did not have any observations in the embedded range. The distribution of embedded mode and KDSI's coupled with the small sample size probably contributed to the distribution of the high ratios.

TABLE VIII

SCHEDULE DATA BY PARTITIONS

PARTITION	N	MEAN	ST.DEV.	RATIO
2-512 KDSI	36	17.838	13.6	1.31
Organic mode	13	16.971	15.4	1.10
Semidetached	18	18.423	14.0	1.32
Embedded	5	17.984	8.48	2.12
2-5 KDSI	10	5.1272	1.16	4.42
8-20 KDSI	5	10.178	3.09	3.29
32-56 KDSI	10	18.780	4.68	4.01
75-128 KDSI	7	24.159	3.96	6.10
250-512 KDSI	4	45.771	16.5	2.77
32 KDSI	9	18.712	4.96	3.77
128 KDSI	6	25.077	3.42	7.33

The mean TDEV increased slightly from organic to semidetached mode partitions then decreased from the semidetached to embedded mode partitions. This is expected behavior. TDEV should follow the pattern of the MM because it is computed from the MM.

H. PRODUCTIVITY (PROD)

Productivity is the number of delivered source instructions divided by the effort. Table IX gives the productivity by partition.

TABLE IX

PRODUCTIVITY DATA BY PARTITIONS

PARTITION	N	MEAN	ST.DEV.	RATIO
2-512 KDSI	36	274.22	225.	1.22
Organic mode	13	306.0	206.	1.49
Semidetached	18	273.02	253.	1.08
Embedded	5	193.93	187.	1.04
2-5 KDSI	10	455.39	267.	1.71
8-20 KDSI	5	345.31	307.	1.12
32-56 KDSI	10	153.04	82.6	1.85
75-128 KDSI	7	189.25	91.0	2.08
250-512 KDSI	4	184.08	144.	1.28
32 KDSI	9	152.18	87.5	1.74
128 KDSI	6	180.70	96.6	1.87

The productivity, delivered source instructions per man-month, fell off as the mode went from organic to semidetached to embedded, and also with an increase in the KDSI range. The drop in productivity can be attributed, in part, to increased complexity and increased overhead of communications with an increase in the number of persons working on the project.

For all partitions, the ratios were small due to large standard deviations.

I. FULL-TIME PERSONNEL (FSP)

Full-time personnel, FSP, is equal to effort divided by schedule. For these computations, fractional FSP was used. Table X gives the FSP by data partitions.

TABLE X

FSP DATA BY PARTITIONS

PARTITION	N	MEAN	ST.DEV.	RATIO
2-512 KDSI	36	20.526	27.6	0.744
Organic mode	13	12.304	19.1	0.644
Semidetached	18	23.268	32.9	0.707
Embedded	5	32.035	23.8	1.35
2-5 KDSI	10	1.4849	0.574	2.59
8-20 KDSI	5	6.2941	6.23	1.01
32-56 KDSI	10	15.393	7.63	2.02
75-128 KDSI	7	33.168	17.5	1.90
250-512 KDSI	4	76.631	43.3	1.77
32 KDSI	9	15.107	8.03	1.88
128 KDSI	6	35.910	17.4	2.06

There was a significant increase in both the mean and the standard deviation for FSP and the data partitions as KDSI increased. An increase in FSP as the mode went from organic to semidetached to embedded also occurred.

The ratios for very small KDSI, medium KDSI and the 128K partition were all above 2. The 75K observation in the high

range KDSI had a low enough FSP to bring the mean for the partition below 2. Of the 7 observations in the high KDSI range, 6 were 128K.

J. MODELS/METHODS

Table XI gives the number of reported observations for

TABLE XI

NUMBER OF REPORTED USAGES BY MODEL

ESTIMATION METHOD	NUMBER OF OBSERVATIONS* N = 36	N = 48
EXPERIENCE/NONE	13	17
COCOMO	5	7
SLIM	7	11
RCA PRICE S	4	7
ARMY TB 18-116	2	2
ESTIMAX/ESTIPLAN	2	2
PSL/PSA-SAGE/APS	1	1
STRADIS	1	1
PAC II/ARTEMIS	1	1
MICROREP	0	1
PC/70	1	1
MANUAL/$/LINE	3	3
ECONOMIC ANALYSIS	1	1
OTHER	6	7

*SOME RESPONDENTS REPORTED MORE THAN ONE CATEGORY.

Most of the estimation methods were supplied by the respondents. The offered responses to the model method question was COCOMO, SLIM, and OTHER_____. Some respondents selected the other option or wrote "various" without listing the model(s) used. Thus, less hard data was collected. Practically every one of the respondents could have listed experience as the method used to estimate the software development costs. Therefore, experience was placed in the same category as the "none" response. SLIM, COCOMO, and RCA PRICE S seemed to be the most widely used models with the Federal Conversion Center Manual/$ per lines of code fourth in order of preference. Again, the small sample size and not obtaining all model/method names greatly increases the chances of error on the order of preference of software models/methods. From telephone conversations, it seemed that PRICE S was the most popular model with the Air Force and the Marines preferred SLIM. Some Marines were using ESTIMAX for front end estimates.

K. APPLICATION NATURE

Table XII gives the number of observations of each type of application.

Most of the software application categories were supplied by the respondents. To reduce the number of categories, logistics was combined with supply and real estate management was combined with maintenance. The posed question offered FINANCIAL, SUPPLY, and OTHER_____.

TABLE XII

NUMBER OF APPLICATIONS BY NATURE

NATURE	N = 36	N = 48
FINANCIAL	13	22
PERSONNEL	3	5
SUPPLY	9	12
C-CUBE	2	3
SIMULATION	1	2
TRANSPORTATION	1	1
SYSTEMS SOFTWARE	0	1
WEAPONS	4	4
TRAINING	1	1
STRATEGY	4	4
ENGR/SCIENTIFIC	1	1
OPER. READINESS	1	1
MANAGEMENT	4	5
MED/SAFETY/JAG	2	2
MAINTENANCE	2	3
UNKNOWN	2	2

*SOME RESPONDENTS REPORTED MORE THAN ONE CATEGORY.

As a result many different categories were obtained and some respondents marked "other" but did not write what "other" was. One survey participant did not answer the nature of application question, however, since the return address was

a financial center, the response was defaulted to financial. Most of the survey participants dealt with financial/supply applications.

Perhaps some of the categories could have been combined or maybe the categories used by the SLIM package should have been used. Applications sharing the same nature should have some overlap/transferability of modules.

L. PERFORMANCE PERCENTAGES

TABLE XIII gives the reported performance percentages.

TABLE XIII

AVERAGE REPORTED PERCENTAGES

VARIABLE	N	MEAN	ST.DEV.	RATIO
Total Sample:				
COST%	37	0.65081	0.306	2.13
SCHEDULE%	37	0.64730	0.321	2.17
EFFORT%	35	0.63143	0.306	2.06
2-512 KDSI:				
COST%	32	0.65406	0.313	2.09
SCHEDULE%	32	0.65312	0.328	1.99
EFFORT%	31	0.63871	0.319	2.00

Several respondents noted both on the survey and telephonically that the questions dealing with what the percentage estimated cost, schedule and effort, were of the actuals was confusing. A few of the respondents seemed to have treated the question as if percent error was the

requested information. It was intended that the question be worded to make the percentages smaller, hence, less embarrassing.

Most of the percentages were divisible by 5. This may be an indication either of rounding or that the differences between estimates and actuals for software development estimation are either not closely monitored or not available to the respondent.

Since many of the respondents said their percentages were "swagged", the reported percentages of estimated/actual cost, schedule, and effort may have no real significance. The reported percentages all averaged around 65% with a small standard deviation. The percentages were correlated with all the other data and no significant correlations were found.

M. CORRELATIONS

1. Nature and Method

Correlation between types of applications and the model(s)/method(s) used was anticipated. However, the sample size was small with respect to the number of types of applications and the number of models/methods in use. Thus, the correlations between nature and methods were not significant. There was some observed correlation between the PRICE S and the COCOMO model. Both models were mildly correlated with weapons application. The correlations may be due to the small sample sizes.

2. Cost Drivers with Total Sample

Table XIV gives correlations between cost drivers, KDSI, and mode. Cost driver data was read into the Minitab spreadsheet. Missing cost drivers were recoded from 99 to "*" before correlation.

A strong correlation exists between anaylst capability and programmer capability in the survey sample. Whether good analysts train programmers, or vice versa, or many analysts are also programmers, or it's planned, or it's the luck of the draw is unknown. However, since programmer capability (PCAP) and analyst experience (AEXP) are correlated almost to a significant level, it appears possible that either the analysts train the programmers or that inexperienced analysts are never assigned to the experienced programmers. The correlation between programmer capability (PCAP) and TOOL and analyst capability (ACAP) and TOOL is almost slgnificant, which would imply that capable programmers and analysts employ software engineering techniques.

The correlation between TIME and STOR is almost significant. The machines with higher main storage constraints required faster programs so that less storage will be used and the chances of a capacity problem are reduced. Required reliability also has correlation coefficients which are almost significant with STOR. Higher reliability generally requires software engineering and

testing and uses more storage. Software engineering is generally needed in larger applications because there are more personnel working on them who must communicate and documentation must be done to allow for maintainability. TOOL and MODP are also almost significantly correlated. The use of software tools and required development schedule seem to be related.

Milder correlations exist between reliability and complexity, reliability and time constraints, complexity and

TABLE XIV

COST DRIVERS CORRELATED WITH TOTAL SAMPLE

MODE	0.258															
RELY	0.023	0.296														
DATA	0.059	0.085	0.288													
CPLX	0.108	0.227	0.480	0.439												
TIME	0.050	0.358	0.442	0.210	0.403											
STOR	0.044	0.236	0.560	0.248	0.382	0.676										
VIRT	0.116	-0.129	0.333	0.282	0.249	0.373	0.229									
TURN	0.149	-0.139	0.061	0.126	0.067	0.099	0.035	0.278								
ACAP	0.074	-0.133	-0.281	-0.391	-0.346	-0.280	-0.353	-0.154	-0.033							
AEXP	0.039	-0.028	0.111	-0.114	-0.246	-0.125	0.172	0.100	-0.254	0.366						
PCAP	-0.032	-0.117	-0.266	-0.409	-0.427	-0.292	-0.201	-0.066	-0.084	0.816	0.504					
VEXP	-0.021	-0.225	-0.140	-0.265	-0.187	-0.062	0.008	-0.090	-0.313	0.468	0.383	0.427				
LEXP	0.008	-0.196	0.201	0.013	-0.028	-0.077	-0.005	0.116	0.030	0.277	0.291	0.334	0.356			
MODP	0.028	-0.130	-0.017	-0.004	0.007	-0.115	0.022	-0.110	0.124	0.343	0.313	0.476	0.217	0.183		
TOOL	-0.085	-0.224	-0.254	-0.360	-0.241	-0.239	-0.052	-0.138	-0.244	0.501	0.417	0.590	0.449	0.029	0.583	
SCED	0.338	0.202	0.001	-0.074	-0.003	-0.028	-0.067	0.147	-0.285	0.182	0.163	0.214	0.237	0.091	-0.110	0.019

time, complexity and programmer capability, data and complexity, data and programmer capability, programmer capability and modern programming practices, programmer capability and virtual machine experience, and virtual machine experience and use of software tools. It follows that to increase reliability requires more complex, faster programs. Speeding up programs tends to make them more complex. Larger data bases frequently require more complex programs. It generally takes more capable programmers to work with larger data bases and more complex programs. The

more capable programmers tend to use modern programming practices, software tools and their expertise with the virtual machine to increase their productivity.

3. Cost Drivers with 2-512K Partition

Table XV gives correlations of the cost drivers with KDSI, mode, MM, and TDEV.

TABLE XV

COST DRIVERS CORRELATED WITH 2-512 KDSI SAMPLE

KDSI	0.055																		
MM	0.088	0.895																	
TDEV	0.037	0.897	0.869																
RELY	0.332	0.125	0.282	0.253															
DATA	0.171	0.087	0.016	0.116	0.279														
CPLX	0.453	0.146	0.152	0.216	0.574	0.409													
TIME	0.287	0.001	0.137	0.236	0.449	0.289	0.471												
STOR	0.120	0.130	0.358	0.285	0.430	0.336	0.460	0.535											
VIRT	-0.198	-0.092	0.045	0.090	0.271	0.275	0.326	0.575	0.410										
TURN	-0.167	-0.351	-0.359	-0.302	-0.001	0.086	0.007	0.119	0.032	0.123									
ACAP	-0.116	-0.066	-0.044	0.103	-0.292	-0.474	-0.460	-0.226	-0.309	-0.249	-0.099								
AEXP	-0.159	0.104	0.164	0.228	0.130	-0.069	-0.126	-0.172	0.227	0.094	-0.306	0.411							
PCAP	-0.116	0.085	0.142	0.208	-0.293	-0.418	-0.361	-0.268	-0.147	-0.171	-0.198	0.888	0.523						
VEXP	-0.321	0.172	0.205	0.331	-0.154	-0.298	-0.256	-0.026	0.078	-0.038	-0.274	0.487	0.476	0.515					
LEXP	-0.157	0.251	0.296	0.363	0.101	-0.255	-0.036	0.083	0.233	0.016	0.026	0.326	0.527	0.455	0.496				
MODP	-0.104	0.155	0.137	0.306	0.022	0.069	-0.032	-0.126	0.077	-0.233	0.077	0.466	0.346	0.567	0.420	0.566			
TOOL	-0.226	0.050	0.084	0.144	-0.247	-0.342	-0.400	-0.315	-0.077	-0.169	-0.295	0.609	0.672	0.706	0.621	0.358	0.562		
SCED	0.080	0.130	0.145	0.276	-0.022	-0.096	0.115	0.084	0.046	0.102	-0.637	0.150	0.155	0.199	0.192	-0.015	-0.061	0.160	

The partition within the COCOMO KDSI range displayed an even stronger correlation between analyst capability and programmer capability and an almost significant relation between programmer capability and analyst experience as well as programmer capability and virtual machine experience. The correlation between virtual machine experience and modern software tools, modern programming practices and use of software tools, language experience and use of modern programming practices, and modern programming practices and programmer capability, imply that the capable programmer tends to be one who uses modern programming practices and software tools and is experienced with the programming language and the virtual machine.

There are very strong correlations between KDSI and MM, KDSI and TDEV, and TDEV and MM. The correlation can be predicted from the formula for effort and schedule.

A significant correlation exists between PCAP and TOOL and an almost signficant correlation exists between ACAP and TOOL. This indicates that the more capable programmers and analysts are likely to use software tools.

The correlation of TIME with STOR and TIME and VIRT is almost significant. Those applcations requiring fast execution time generally also required more of storage. The relation between VIRT and STOR may be caused by updating or upgrading equipment to handle the storage requirements without creating a capacity problem.

The almost significant correlation of RELY with STOR and RELY with complexity are probably due to the increased program coding and storage requirements imposed by increased reliability.

Milder correlations exist between reliability and time constraints, complexity and time, complexity and storage, data and complexity, mode and complexity, time and storage, turnaround and schedule, and analyst capability and virtual machine experience. To increase reliability generally requires more complex, faster programs. Speeding up programs tends to make them more complex. Larger data bases frequently require more complex programs. As the mode is changed from organic to semidetached to embedded the

programs are increasing in complexity. More capable analysts generally exploit their familiarity with the virtual machine to improve performance.

N. COMMENTS

1. Survey

Some of the comments by respondents could be used to modify the model/survey. One respondent deleted part of a mode definition to fit a particular software shop. Another found the mode selection a tough choice due to the restriction to "in-house" personnel for the organic mode. Respondents seemed to have a problem with the reliability response. Many are used to having reliability expressed in a percentage range and some wanted an extra high response for reliability. One respondent commented on some problems which the COCOMO model does not adequately address which tend to drive the costs through the ceiling.

> . . . . Large defense systems (e.g. Early warning, command & control, aircraft avionics/fire control/ automatic test equipment, and electronic warfare) are complex systems involving embedded and stand-alone processors in all size categories. The software is complex particularly related to systems and subsystems interfaces. Early program estimates of cost, schedule, complexity and resources are strongly hampered by inadequate requirements definition, extremely long acquisition/design cycles that are pushing state-of-the-art techniques and equipment, and political environments. The bottom line is we have a very small data base of information relative to the use or accuracy of software cost estimation for these types of programs. Data is hard to get and often not adequately contracted for from the actual development contractor. We use a number of costing models as does the development contractor. These are essential and must mature through

enforced usage if we are to get a handle on software costs . . . but we have a ways to go. Systems I have Systems I have worked on over the past ten years required high reliability (.998 or better), were real time systems, with specified 25% memory and processing time reserves (but generally delivered with no or very little reserves.), subject to continuous software upgrades/enhancements, involved large mainframe ground processing as well as embedded micro/mini cababilities.

2. Telephone Interviews

Prior to distributing the survey, many phone calls were made in an attempt to put together a distribution list. A few lessons were learned that did not appear on the survey. On hearing that one installation was obtaining excellent results using a tuned SLIM model, the installation was contacted. The application was financial/supply related with a large historical database. A telephone interview reported specific numbers between 90% and 100% for the percentages the estimation was of the actual schedule and effort. When the survey was returned, another individual had completed it and all three responses on the question dealing with percentages were marked 100%. The installation had an operations research specialist tune the SLIM model to the historical data base. If other installations have need of the same kind of application/tuning, it would be advisable to have software/techniques exchanges.

One interviewee described a software cost estimation shop which was using several models as cross-validators. The results of this shop and others like it could be a prime source of data.

IV. COCOMO TOOL

A. INTRODUCTION

Much work is already accomplished in the area of software engineering techniques. From a general systems development approach, [Refs. 7 & 8], to a specific systems approach, [Refs.9 & 10], much is presented on the methods of software development. While the approach used in these methods for implementation varies with each author, the elements of requirements analysis and design are considered basic to the proper development of software. Pressman, [Ref. 11], addresses these basic elements in a manner which attempts to integrate various software concepts into a concise guide for analysts and programmers alike. The presentation of requirements analysis and design for the COCOMO TOOL in this chapter incorporates Pressman's guidelines and serves a twofold purpose. First, a general model provides a foundation to start from for those who have little or no idea where to begin. Second, analysis and design of the COCOMO TOOL enhances comprehension of the automated COCOMO model. Information and functional descriptions, processing narrative, design constraints, validation criteria, and special considerations are all expanded on in the first section of specific requirements analysis for the COCOMO TOOL. The second section of design

presents the COCOMO TOOL scope, design descriptions, and module descriptions. These sections give an overview of the mechanisms which drive the program development.

B. REQUIREMENTS ANALYSIS

1. Information Description

a. Data dictionary - Appendix B

b. Data structure charts - Appendix C

2. Functional Description

a. Inputs:

(1) Model selection - Basic or Intermediate

(2) Mode selection - Organic, Semidetached, or Embedded

(3) Estimated thousands of delivered source instructions (KDSI) for the software development project

(4) Monthly personnel costs for software development

(5) Software development effort multipliers for Intermediate model only

(6) Annual Change Traffic (ACT) for software maintenance

(7) Monthly personnel costs for software maintenance

(8) Software maintenance effort multipliers for Intermediate models only

b. Calculations in the COCOMO TOOL use static, single variable, mode dependent formula for computing effort and maintenance man-months and months respectively.

c. Outputs:

(1) Effort in man-months for Basic model

(2) Nominal and adjusted effort for Intermediate model

(3) Effort adjustment factor for Intermediate model

(4) Schedule in months

(5) Productivity in delivered source instructions per man-month

(6) Full time equivalent software personnel

(7) Annual software development cost

(8) Maintenance effort adjustment factor

(9) Maintenance effort in man-months

(10) Annual maintenance cost

(11) Phase distribution of effort and schedule

(12) Activity distribution by phase

(13) Activity distribution by phase for maintenance

3. Processing Narrative

After initiating the program, a choice of two models is made: Basic or Intermediate. Within each model an Organic, Semidetached, or Embedded mode is selected. Input values for cost driver attributes (Intermediate model only), KDSI, and personnel cost per man-month are entered. The program calculates and displays effort adjustment factor (Intermediate model only), estimated effort, schedule, annual cost, productivity, and number of full time software personnel for software development. Options from this point are to continue in the development branch of the program or to enter the maintenance branch. Continuing in the

development branch allows for program calculation and display of phase distribution of effort as well as activity distribution by phase. Results for the phase distribution of effort include product design, programming (detailed design, and unit testing), and integration and testing. The activity distribution by phase produce eight results. These eight results consist of requirements analysis, product design, programming, test planning, verification and validation, project office, quality assurance and manual development.

Selection of the maintenance branch option requires inputs of maintenance effort cost driver attributes (Intermediate model only), maintenance personnel cost per man-month, and annual change traffic values. Results calculated and displayed include estimated effort, schedule, and annual cost for maintenance. An additional option in the maintenance branch produces and displays maintenance phase distribution of effort.

Copies of the prior computed values for either the software development or maintenance calculations are optionally saved during each session. These saved iterations are viewed for comparison either on the computer screen or on a printer output as desired.

4. Design Constraints

   a. Tables used in the COCOMO TOOL for phase distribution of effort and schedule, and activity distribution by phase for effort and maintenance are based on the following values of KDSI: 2, 8, 32, 128, and 512. KDSI values which fall between these standard KDSI figures are interpolated. Values of KDSI lower than 2 and greater then 512 are beyond the range of the COCOMO TOOL and be receive an error message.

   b. The program is interactive.

5. Validation Criteria

   a. Performance bounds

      (1) Calculations computed and displayed in less then 1.5 minutes

      (2) Calculated results accurate to at least one decimal place.

      (3) Inputs are checked for errors and properly indicated when found.

   b. Classes of tests

      (1) Unit testing of module interfaces, local data structures, and important module execution paths, error paths and boundary conditions.

      (2) Top-down integration testing to check interface integrity, functional validity, and information content.

      (3) Validation testing to verify all software requirements are met.

6. Special considerations include providing a user's manual to assist with program execution, error handling, and program maintenance.

C. DESIGN

1. Scope

a. Objective - development of an interactive decision support system (DSS) to implement the Basic and Intermediate COCOMO models.

b. Hardware - Selection of hardware is driven by on-site equipment resources and RAM/hard disk availability to support the selected software. Micro-computers are selected over mainframes because of the desire for software transportability and system availability. Due to the proliferation of IBM compatibles an IBM PC-XT with 640K of RAM, color monitor and a 10 megabyte hard disk is determined to be appropriate for the software development.

c. Major software functions

(1) Table/database capacity

(2) Spreadsheets

(3) Screen generator capability

(4) Report generator capability

(5) Graphing capability

(6) Error prompting messages

(7) Word processing ability

(8) Color manipulation

d. Software - Integration of the above major software functions into a single package is desirable. This feature makes extra coding to interface dissimilar packages avoidable. Knowledgeman from MDBS is selected for these reasons and because it contains all of the above major software functions.

e. Human interfaces - The COCOMO TOOL program is menu driven with selections made from function keys.

2. Design Description

   a. Data description - Appendix B provides a data dictionary of terms used with the COCOMO model.

   b. Derived software structure - Appendix C displays the top-down hierarchy of the COCOMO TOOL program. All modules shown are highly cohesive. Each module is either sequentially or functionally cohesive in that the output data from one module is passed directly into the module or the module takes inputs and produces outputs.

   c. Software structure interfaces - The modules shown in Appendix C also have relatively low coupling. Data coupling occurs because only the necessary data is passed between two modules. Control coupling is also necesary as control flags are passed to maintain program status.

3. Module Descriptions - To enhance readibility and reduce duplication of effort all modules are colocated with the program listings in Appendix D. Each description provides a processing narrative, sample call, input received, output produced, and indicates any submodule which may be called.

V. SUMMARY, CONCLUSIONS AND RECOMMENDATIONS

A. SUMMARY

1. KDSI has a greater impact on nominal effort than mode has.

2. The mean to standard deviation ratios for effort increase as the mode goes from organic to semidetached to embedded.

3. Productivity dropped both as the mode went from organic to semidetached to embedded and as the KDSI range increased.

4. The number of full-time personnel increases as the mode changes from organic to semidetached to embedded as well as with an increase in the number of KDSI.

5. Survey responses indicated that SLIM, COCOMO, RCA PRICE S, and the Federal Conversion Center Manual $/lines of code were the most widely used estimation models/methods.

6. Most applications were financial/supply related.

7. The survey results indicate a strong correlation between analyst capability and programmer capability.

8. Many software development shops do not keep track of number of lines of code and estimated cost, schedule and effort, nor do they match estimates with actuals.

9. There is some correlation between software and reliability, complexity, execution time constraints and storage constraints.

10. Modern programming practices and the use of software tools seem to be related.

11. There are very strong correlations between KDSI and MM, KDSI and TDEV, and TDEV and MM.

B. CONCLUSIONS

1. Survey

No change in the trend for increased demand for software by DOD is anticipated. There will be a continued need for improved software cost estimation. There will probably be no one model which is the panacea for all nature of applications. The successful cost estimation shops will probably use a battery of models. Some day a model or set of models will probably be deemed optimal for specific types of applications. Nevertheless, a data base must be built before a determination can be made. The Intermediate COCOMO model shows promise both as a software development and maintenance estimation tool. Intermediate COCOMO will probably become a valued asset in situations where a good estimation of the number of lines of delivered source instructions is made. Software shops will probably be converting some applications into ADA in the future.

2. Decision Support System

Development of a DSS covers not just one area but rather encompasses several factors which must be closely integrated to produce an effective system. These factors deal with hardware, software, data, procedures, and personnel. The selection of a hardware system for DSS development is often constrained by the resources already available in the software shop which drives the software selection criteria. Networking of microcomputers also

increases the productivity of project development due to file sharing capabilities.

DSS software is only as good as the software applications package foundation on which it executes. Features which should be basic to any applications package for DSS development include tables, spreadsheets, graphing, report and screen generators, and color manipulation. Documentation and vendor support are other attributes that also must be considered. Technically oriented documentation must be tempered with many examples and lessons for the software development practitioner to obtain the full benefits of the package. Anything less then this inhibits the full scale potential that could be realized. Vendor responsiveness for clarification of ambiguous problems becomes very important when there are no other "experts" in the local area. Waiting for a return call for a problem called in is less than satisfactory, especially when a deadline is approaching. The KnowledgeMan applications package contains all of the basic elements listed above and is used in the support of the COCOMO TOOL. This package contains a full compliment of tools whose use is limited only by the creativity of the developer. However, response time for computations and file manipulations are slower than desired. Another limitation of the package is that it allows useage of only 192K of RAM even if a machine contains a higher capacity RAM. For a large size program this may

cause "insufficient memory" errors to be generated which crash the program. To get around this dilemma each module is separately loaded, processed, and released within the program. This produces a speed reduction in the program due to file manipulation. The documentation, while plentiful and excellent for the professional, is challenging for the beginner. Supplemental material from other sources is often a solution as a clearer writing style with more examples is all that is needed.

Proper data development serves only to enhance the product end-result. This is achieved by using rigorous software engineering techniques such as requirements analysis and design. Even though these techniques add extra time to the front end of the development, it is time well spent. Coding time and program maintainability benefit greatly from this preceding work. In addition, data presentation, whether it is input or output, plays just as vital a role as analysis and design. Proper data display is faster to learn, easier to work with, and reduces errors if it is presented in a consistent format. This is where the double responsibility of the programmer comes into play. Not only must the programmers view the product from the viewpoint of the designer, but they must also be able to see the product through the eyes of the user--not a position which can be accomplished by many. Development of the COCOMO TOOL is the result of software engineering

techniques. The program uses menu driven screens, customized function keys, and succinct error messages to produce an effective DSS system.

Procedures are as natural to software development as breathing is to a human. Properly established procedures eliminate disorganization and maintain the essence of productivity. Backup procedures to save completed work and committee procedures to maintain project direction and prevent goal diversification are minimally required in every software development effort. Of course, too many procedures are just as bad as no procedures at all. Excessive procedural detail leaves no room for creativity and flexibility, a bane to software developers. Procedure uses during COCOMO TOOL development prevented loss of program data due to a failed hard disk and enabled program development to procede at a steady pace.

Essential to every software development project are the personnel. Communication among team members before and during the project are necessary for successful project implementation. Lack of communications creates program divergence resulting in time wasted to correct what should have been done correctly in the first place. Experience of personnel with programming and the applications package in use increases productivity and saves time. However, once an applications package is learned there is a great deal of inertia to overcome when deciding to switch to a new

applications package. While we all had prior programming experience, not one of us had any experience with the KnowledgeMan applications package. This single factor was a major contributor to program development schedule increases.

C. RECOMMENDATIONS

1. Survey

It is recommended that the Department of Defense Computer Institute (DODCI) track all the software projects in the DOD. A survey method such as the Delphi approach can be used to improve the data gathering effort. This would produce data compatible with all models tested and to correct any noted deficiencies. The survey should be easy to complete, i.e., be objective. Responses should require only pencil marks with a separate comment sheet supplied. The survey input to the data base should be optically scanned. Photocopies of any rejects should be returned to the originator along with a replacement survey form and an explanation form letter.

DODCI should collect and load data from the initial estimates and from the actual results. Analysis of the data should determine which shops are performing well with particular types of applications. After determining what model(s) or techniques are in use for the successful shops, arrange to have the information/expertise transferred to other sites using similar applications. Any elimination of reinvention of the wheel could both save dollars and improve

performance. A vehicle for transfer could be a mobile training team. The training team could identify ingredients of success, copy and distribute any software tools used, as well as training other teams. With DODCI tracking the performance, there may be competition and some competitors may not want to share their successes. If this is the case, a mobile training team could be used to educate the software centers on the benefits of shared successes and perhaps even assist with the transfer of technology to others.

A cost driver should be added to the COCOMO model for implementation of new R&D, or, perhaps estimated effort, schedule and cost could be multiplied by a number. The cost driver tables should be modified so that all ratings are from very low to extra high. Wherever possible, the ratings should have descriptive numbers. For example, RELY, should also be categorized by percentages. There may also need to be separate cost driver ratings for the limits. The PCAP rating should be evaluated. The current ratings may not account for differences which may be as much as a magnitude.

2. Decision Support System

The use of the COCOMO TOOL is recommended for all DOD software development shops. This tool can be used for software development and maintenance estimation in those shops that do not have any software estimation tools. Shops that have other estimation tools can use the COCOMO TOOL either as a supplement to those tools or as a means for

cross-checking the other tools. While copies of the COCOMO TOOL can be obtained from the Naval Postgraduate School (Professor Bui via the Department of Administrative Sciences), each software shop must provide its own KnowledgeMan package to run this program as distribution of this copyrighted material is unlawful unless some form of license is obtained.

To provide more standardization in DOD for requirements analysis and design of software projects, it is also recommended that all senior level programmer and management personnel be required to attend a requirements analysis and design course offered by DODCI.

APPENDIX A

SURVEY OF SOFTWARE COST ESTIMATION PRACTICES

Organization: Phone:
Mailing Address:
Person conducting survey:

1. What is the nature of your software?

   a. Financial
   b. Supply
   c. Other: _____

2. What is your average program size in thousands of lines of code?
   (Circle a number or write a number in the blank provided.)

   a. 2 b. 8 c. 32 d. 128 e. 512
   OTHER: _____

3. Software development modes can be classified into three types:

   a. Organic - Relatively small teams develop the software. Most team members are from in-house and have extensive experience in working with other related systems within the organization. There is minimal need for innovative algorithms. Software is generally under 50K lines of code. Larger organic mode products may be built using existing software.

   b. Embedded - The software is embedded in a tightly coupled complex of hardware, software, regulations and operational procedures. For example, air traffic control systems, electronic funds transfer systems, etc.

c. Semi-detached - May have a mixture of the organic characteristics. Teams consist of a wide mixture of experienced and inexperienced people; team members have an intermediate or incomplete level of experience with related systems to be developed.

Circle the mode that best applies to your organization.

a. Organic b. Embedded c. Semi-detached

4. Which software cost estimation method(s) or model(s) do you use?

a. COCOMO b. SLIM OTHER:_____

5. Of all projects which you have been involved with for the past five years:

a. of average actual costs, what percentage is estimated costs? _____

b. of average actual schedule, what percentage is estimated schedule? _____

c. of average actual effort, what percentage is estimated effort? _____

6. Profile characteristics of your software projects. For each attribute, circle the category which applies.

ATTRIBUTES

PRODUCT ATTRIBUTES:

RELY: Required software reliability	Effect: slight inconvenience	Low. Easily recoverable losses	Moderate. Recoverable losses	High. Financial loss	Risk to human life	
	Very Low	Low	Nominal	High	Very High	
DATA: Data base size		DB bytes / Prog DSI	D / P	D / P	D / P	
		< 10	10 < - < 100	100 < - < 1000	- > 1000	
		Low	Nominal	High	Very High	
CPLX: Product complexity	Straightline code	Single file. No data structure	Standard routines. Simple structure changes and edits	Special purpose routines. Complex data restructure	Difficult but structured routines	Difficult and unstructured routines
	Very Low	Low	Nominal	High	Very High	Extra High

COMPUTER ATTRIBUTES:

TIME: Execution time constraint	< 50% use of available execution time	70%	85%	95%	
	Nominal	High	Very High	Extra High	
STOR: Execution storage constraint	< 50% use of availiable storage	70%	85%	95%	
	Nominal	High	Very High	Extra High	

VIRT: Virtual machine volatility *	Major change every 12 months Minor: 1 month	Major: 6 months Minor: 2 weeks	Major: 2 months Minor: 1 week	Major: 2 weeks Minor: 2 days
	Low	Nominal	High	Very High

TURN: Computer turnaround time	Interactive	Average turnaround < 4 hours	4 -12 hours	> 12 hours
	Low	Nominal	High	Very High

PERSONNAL ATTRIBUTES:

ACAP: Analyst capability **	15th percentile	35th percentile	55th percentile	75th percentile	90th percentile
	Very Low	Low	Nominal	High	Very High
AEXP: Applications experience	< 4 months experience	1 year experience	3 years experience	6 years experience	12 years experience
	Very Low	Low	Nominal	High	Very High
PCAP: Programmer capability **	15th percentile	35th percentile	55th percentile	75th percentile	90th percentile
	Very Low	Low	Nominal	High	Very High

VEXP: Virtual machine experience *	< 1 month experience	4 months experience	1 year experience	3 years experience
	Very Low	Low	Nominal	High
LEXP: Programming language experience	< 1 month experience	4 months experience	1 year experience	3 years experience
	Very Low	Low	Nominal	High

PROJECT ATTRIBUTES:

MODP: Use of modern programming practices	No use	Beginning use	Some use	General use	Routine use
	Very Low	Low	Nominal	High	Very High

TOOL:	Use of software tools	Basic micro-processor tools	Basic mini tools	Basic midi/maxi tools	Strong maxi pprogramming test tools	Add requirements, design, management, documentation tools
		Very Low	Low	Nominal	High	Very High
SCED:	Required development schedule	75% of nominal	85%	100%	130%	160%
		Very Low	Low	Nominal	High	Very High

7. Would you like to be included on the report findings distribution list?

_____ Yes

_____ No

APPENDIX B

Data Dictionary

DATA DEFINITION
Name: ACT Format: Real Range: 0.00 - 1.00 Field/Cell in Files: CBO.ICF, CBS.ICF, CBE.ICF, CIO.ICF, CIS.ICF, CIE.ICF, BESM.ITB, BESMAD.ITB, IESM.ITB, IESMAD.ITB. Located in spreadsheet cell #M6. Processing: The user enters the value of the ACT onto the spreadsheet. ACT is used to calculate nominal maintenance effort, MMnam, by multiplying by the development effort EFnom (or MMnom) in in a spreadsheet. This is done by CALCMDAT.IPF. Description: Annual Change Traffic. This is the fraction of the software product's source instructions which undergo change during a typical year, either through addition or modification. $$ACT = \frac{(KDSI\ ADDED + KDSI\ MODIFIED)}{(ORIGINAL\ KDSI)}$$

DATA DEFINITION

Name: ACTIVITY DISTRIBUTION BY PHASE

Variables:

Product Design Phase:

RAPD, PDPD, PROGPD, TESTPD, VVPD, POPD, CQPD, MANPD

Programming Phase:

RAPROG, PDPROG, PROGPROG, TESTPROG, VVPROG, POPROG, CQPROG, MANPROG

Integration and Testing Phase:

RAIT, PDIT, PROGIT, TESTIT, VVIT, POIT, CQIT, MANIT

Description:

Activity Distribution by Phase. This is calculated for effort. Example:

Activity Distribution by Phase for Effort = Phase Distribution for Effort x Activity Distribution %

DATA DEFINITION
Name: ACTIVITY DISTRIBUTION OF MAINTENANCE Varibles: MRA, MPD, MPROG, MTEST, MVV, MPO, MCQ, MMAN Description: Activity Distribution of Maintenance Effort = Man-months x Maintenance Activity distribution %.

DATA DEFINITION

Name: ACOST

Format: Real

Range: positive numbers

Field/Cell in Files: CBO.ICF, CBS.ICF, CBE.ICF, CIO.ICF, CIS.ICF, CIE.ICF, BES.ITB, BESP.ITB, BESPAD.ITB, BESM.ITB, BESMAD.ITB, IES.ITB, IESP.ITB, IESPAD.ITB, IESM.ITB, IESMAD.ITB. Placed in spreadsheet cell #D16.

Processing: Calculated by DEVPARBA.IPF for the basic models and DEVPARMS for the intermediate models.

Description:

Annual Personnel cost during development:

ACOST = PCOST x MM for the basic model.

ACOST = PCOST x MMadj for the intermediate model.

DATA DEFINITION

Name: AMC

Format: Real

Range: positive numbers

Field/Cell in Files: CBO.ICF, CBS.ICF, CBE.ICF, CIO.ICF, CIS.ICF, CIE.ICF, BESM.ITB, BESMAD.ITB, IESM.ITB, IESMAD.ITB.
Placed in spreadsheet cell #M14.

Processing: Annual maintenance cost is computed in CALCMDAT.IPF.

Description:

Annual Maintenance Cost:

AMC = (MMam) x (MPCOST)

The intermediate model uses MMnam.

DATA DEFINITION

Name: Code and Unit Test

Description:

Coding and Unit Testing is a subset of the phase distribution percentage for programming for the basic and intermediate models.

DATA DEFINITION

Name: COST DRIVERS

Format: Real

Range: 0.00 - 1.00

Field/Cell in Files: (see data dictionary entry for the cost drivers listed below.)

Description:

1. DEVELOPMENT COST DRIVERS

The 15 factors which affect software development: ERELY, EDATA, ECPLX, ETIME, ESTOR, EVIRT, ETURN, EACAP, EAEXP, EPCAP, EVEXP, ELEXP, EMODP, ETOOL, and ESCED. There is a multiplier for each factor. When multiplied together, these 15 factors form EAF, the effort adjustment factor.

2. MAINTENANCE COST DRIVERS

The 14 factors which affect software maintenance: RELY, DATA, CPLX, TIME, STOR, VIRT, TURN, ACAP, AEXP, PCAP, VEXP, LEXP, MODP,and TOOL. There is a multiplier for each factor. When multiplied together, these 14 factors represent the EAFm, maintenance EAF. These are identical to the factors for development with the exception of MODP and RELY which have different values for maintenance efforts. There is no maintence driver for schedule.

DATA DEFINITION

Name: CQIT

Format: Real

Range: 0.00-0.99

Field/Cell in Files: CBO.ICF, CBS.ICF, CBE.ICF, CIO.ICF, CIS.ICF, CIE.ICF, BESPAD.ITB, IESPAD.ITB Placed in spreadsheet cell #H72

Processing: Computed by CALCDPAD.IPF and placed into spreadsheet cell #H72 for display and possible graphing or reports.

Description: The activity distribution % of Effort devoted to the Configuration Management and Quality Assurance, CM/QA, activity of the integration and testing phase of development.

DATA DEFINITION

Name: CQPD

Format: Real

Range: 0.00-0.99

Field/Cell in Files: CBO.ICF, CBS.ICF, CBE.ICF, CIO.ICF, CIS.ICF, CIE.ICF, BESPAD.ITB, IESPAD.ITB. Located in spreadsheet cell #D72.

Processing: Computed by CALCDPAD.IPF and placed into spreadsheet cell #D72 for display and possible graphing or reports.

Description: The activity distribution % of Effort that is devoted to the Configuration Management and Quality Assurance, CM/QA, activity during the product design phase.

DATA DEFINITION

Name: CQPROG

Format: Real

Range: 0.00-0.99

Field/Cell in Files: CBO.ICF, CBS.ICF, CBE.ICF, CIO.ICF, CIS.ICF, CIE.ICF, BESPAD.ITB, IESPAD.ITB. Located in spreadsheet cell #F72.

Processing: Computed by CALCDPAD.IPF and placed into spreadsheet cell #F72 for display and possible graphing or reports.

Description: The activity distribution % of Effort that is devoted to the Configuration Management and Quality Assurance, CM/QA, activity during the programming phase of development.

DATA DEFINITION

Name: CUT

Format: Real

Range: 0.00 - 1.00

Field/Cell in Files: CBO.ICF, CBS.ICF, CBE.ICF, CIO.ICF, CIS.ICF, CIE.ICF, BESP.ITB, BESPAD.ITB, IESP.ITB, IESPAD.ITB. Located in spreadsheet cell #D49.

Processing: Computed by CALCEFSC.IPF and placed into a spreadsheet cell #D49 for later reports or graphing.

Description: Phase Distribution of Effort allocated to Coding and Unit Testing. This is a subset of the Programming Phase for Effort.

DATA DEFINITION

Name:
Detailed Design

Description:

Detailed Design is a subset of the phase distribution percentage for for programming for the basic and intermediate models.

DATA DEFINITION

Name: DETDES

Format: Real

Range: 0.00 - 1.00

Field/Cell in Files: CBO.ICF, CBS.ICF, CBE.ICF, CIO.ICF, CIS.ICF, CIE.ICF, BESP.ITB, BESPAD.ITB, IESP.ITB, IESPAD.ITB. Located in spreadsheet cell #D48.

Processing: Computed by CALCEFSC.IPF and placed into a spreadsheet cell #D48 for later reports or graphing.

Description: Phase Distribution of Effort allocated to Detailed Design. This is a subset of the Programming Phase for Effort.

DATA DEFINITION

Name: EACAP

Format: Real

Range: 0.00 - 1.00

Field/Cell in Files: CIO.ICF, CIS.ICF, CIE.ICF, IES.ITB, IESP.ITB, IESPAD.ITB. Located in spreadsheet cell #F107.

Processing: The value is defaulted to 1.0 (nominal) on the spreadsheet. The user can change this to another value displayed on the spreadsheet. Used by REDEVDAT.IPF to compute the Effort Adjustment Factor, (EAF).

Description:

Analyst Capability attribute. One of the 15 cost driver multipliers used in obtaining an EAF.

DATA DEFINITION

Name: EAEXP

Format: Real

Range: 0.00 - 1.00

Field/Cell in Files: CIO.ICF, CIS.ICF, CIE.ICF, IES.ITB, IESP.ITB, IESPAD.ITB, IESM.ITB, IESMAD.ITB. Located in spreadsheet cell #F108.

Processing: See EACAP.

Description:

Applications experience. One of the 15 cost driver multipliers used in obtaining an EAF.

DATA DEFINITION

Name: EAF

Format: Real

Range: positive numbers

Field/Cell in Files: CIO.ICF, CIS.ICF, CIE.ICF, IES.ITB, IESP.ITB, IESPAD.ITB. Located in spreadsheet cell #H11

Processing: REDEVDAT.IPF multiplies the cost driver inputs from the spreadsheet cells together to obtain the EAF which is stored in spreadsheet cell #H11.

Description: Effort Adjustment Factor. This is the product of all 15 cost-drivers for development effort.

DATA DEFINITION

Name: EAFm

Format: Real

Range: positive numbers

Field/Cell in Files: CBO.ICF, CBS.ICF, CBE.ICF, CIO.ICF, CIS.ICF, CIE.ICF, IES.ITB, IESP.ITB, IESPAD.ITB. Located in spreadsheet cell #012.

Processing: CALCEAFM.IPF multiplies the cost driver inputs from the spreadsheet cells together to obtain the EAFm which is stored in spreadsheet cell #012.

Description: Maintenance Effort Adjustment Factor. This is the product of all14 cost-drivers for maintenance effort.

DATA DEFINITION

Name: ECPLX

Format: Real

Range: 0.00 - 1.00

Field/Cell in Files: CIO.ICF, CIS.ICF, CIE.ICF, IES.ITB, IESP.ITB, IESPAD.ITB. Located in spreadsheet cell #F102.

Processing: See EACAP.

Description:

Project Complexity attribute. One of the 15 cost driver multipliers used in obtaining an EAF.

DATA DEFINITION

Name: EDATA

Format: Real

Range: 0.00 - 1.00

Field/Cell in Files: CIO.ICF, CIS.ICF, CIE.ICF, IES.ITB, IESP.ITB, IESPAD.ITB. Placed in spreadsheet cell #F101.

Processing: See EACAP.

Description:

Data Base Size attribute. One of the 15 cost driver multipliers used in obtaining an EAF.

DATA DEFINITION

Name: Effort Coefficients

Format: Real

Range: positive number

Located in Files: DEVPARBA.IPF for the basic model and DEVPARMS.IPF for intermediate.

Description:

Coefficients for the effort equations. These vary by model/mode and can be changed by the user in the respective IPF files (BASIC: DEVPARBA.IPF, INTERMEDIATE: DEVPARMS.IPF) to tune the model to historical project data gathered by an organization.

DATA DEFINITION

Name: Effort Exponents

Format: Real

Range: positive number

Located in Files: DEVPARBA.IPF for the basic model and DEVPARMS.IPF for intermediate.

Description:

Exponents for the effort equations. See Effort Coefficients.

DATA DEFINITION

Name: ELEXP

Format: Real

Range: 0.00 - 1.00

Field/Cell in Files: CIO.ICF, CIS.ICF, CIE.ICF, IES.ITB, IESM.ITB, IESMAD.ITB, IESP.ITB, IESPAD.ITB. Located in spreadsheet cell #F111.

Processing: See EACAP.

Description:

Programming Language Experience. One of the 15 cost driver multipliers used in obtaining EAF.

DATA DEFINITION

Name: EMODP

Format: Real

Range: 0.00 - 1.00

Field/Cell in Files: IESM.ITB, IESMAD.ITB, CIO.ICF, CIS.ICF, CIE.ICF, IES.ITB, IESPAD.ITB, IESP.ITB. Located in spreadsheet cell #F112.

Processing: See EACAP.

Description:

Use of Modern Programming Practices. One of the 15 cost driver multipliers used in obtaining EAF.

DATA DEFINITION

Name: EPCAP

Format: Real

Range: 0.00 - 1.00

Field/Cell in Files: IESP.ITB, IESPAD.ITB, CIO.ICF, CIS.ICF, CIE.ICF, IES.ITB, IESM.ITB, IESMAD.ITB. Placed in spreadsheet cell #F109.

Processing: See EACAP.

Description:

Programmer Capability. One of the 15 cost driver multipliers used in obtaining EAF.

DATA DEFINITION

Name: ERELY

Format: Real

Range: 0.00 - 1.00

Field/Cell in Files: CIO.ICF, CIS.ICF, CIE.ICF, IESP.ITB, IESM.ITB, IESMAD.ITB, IESPAD.ITB, IES.ITB. Located in spreadsheet cell #F100.

Processing: See EACAP.

Description:

Required Software Reliability. One of the 15 cost driver multipliers used in obtaining EAF.

DATA DEFINITION

Name: ESCED

Format: Real

Range: 0.00 - 1.00

Field/Cell in Files: CIO.ICF, CIS.ICF, CIE.ICF, IESP.ITB, IESM.ITB, IESMAD.ITB, IESPAD.ITB, IES.ITB. Located in spreadsheet cell #F114.

Processing: See EACAP.

Description:

Required Development Schedule. One of the 15 cost driver multipliers used in obtaining EAF.

DATA DEFINITION

Name: ESTOR

Format: Real

Range: 0.00 - 1.00

Field/Cell in Files: CIO.ICF, CIS.ICF, CIE.ICF, IESP.ITB, IESM.ITB, IESMAD.ITB, IESPAD.ITB, IES.ITB. Located in spreadsheet cell #F104.

Processing: See EACAP.

Description:

Main Storage Constraint. One of the 15 cost driver multipliers used in obtaining EAF.

DATA DEFINITION

Name: ETOOL

Format: Real

Range: 0.00 - 1.00

Field/Cell in Files: CIO.ICF, CIS.ICF, CIE.ICF, IESP.ITB, IESM.ITB, IESMAD.ITB, IESPAD.ITB, IES.ITB. Located in spreadsheet cell #F113.

Processing: See EACAP.

Description:

Use of Software Tools. One of the 15 cost driver multipliers used in obtaining EAF.

DATA DEFINITION

Name: ETIME

Format: Real

Range: 0.00 - 1.00

Field/Cell in Files: CIO.ICF, CIS.ICF, CIE.ICF, IESP.ITB, IESM.ITB, IESMAD.ITB, IESPAD.ITB, IES.ITB. Located in spreadsheet cell #F103.

Processing: See EACAP.

Description:

Execution Time Constraint. One of the 15 cost driver multipliers used in obtaining EAF.

DATA DEFINITION

Name: ETURN

Format: Real

Range: 0.00 - 1.00

Field/Cell in Files: CIO.ICF, CIS.ICF, CIE.ICF, IESP.ITB, IESM.ITB, IESMAD.ITB, IESPAD.ITB, IES.ITB. Located in spreadsheet cell #F106.

Processing: See EACAP.

Description:

Computer Turnaround Time. One of the 15 cost driver multipliers used in obtaining EAF.

DATA DEFINITION

Name: EVEXP

Format: Real

Range: 0.00 - 1.00

Field/Cell in Files: CIO.ICF, CIS.ICF, CIE.ICF, IESP.ITB, IESM.ITB, IESMAD.ITB, IESPAD.ITB, IES.ITB. Located in spreadsheet cell #F110.

Processing: See EACAP.

Description:

Virtual Machine Experience. One of the 15 cost driver multipliers used in obtaining EAF.

DATA DEFINITION

Name: EVIRT

Format: Real

Range: 0.00 - 1.00

Field/Cell in Files: CIO.ICF, CIS.ICF, CIE.ICF, IESP.ITB, IESM.ITB, IESMAD.ITB, IESPAD.ITB, IES.ITB. Located in spreadsheet cell #F105.

Processing: See EACAP.

Description:

Virtual Machine Volatility. One of the 15 cost driver multipliers used in obtaining EAF.

DATA DEFINITION

Name: FSP

Format: Real

Range: positive number

Field/Cell in Files: CBO.ICF, CBS.ICF, CBE.ICF, CIO.ICF, CIS.ICF, CIE.ICF, BES.ITB, BESP.ITB, BESPAD.ITB, IES.ITB, IESP.ITB, IESPAD.ITB. Placed in spreadsheet cell #D15.

Processing: Calculated by DEVPARBA.IPF for the basic model and DEVPARMS.IPF for the intermediate model.

Description: Average staffing of Personnel

FSP = MM/SCHEDULE for the basic model.

FSP = MMadj/SCHEDULE for the intermediate model.

DATA DEFINITION

Name: FSPm

Format: Real

Range: positive numbers

Field/Cell in Files: CBO.ICF, CBS.ICF, CBE.ICF, CIO.ICF, CIS.ICF, CIE.ICF, BESM.ITB, BESMAD.ITB, IESM.ITB, IESMAD.ITB. Placed in spreadsheet cell #M13.

Processing: Calculated in CALCMDAT.IPF.

Description: Average staffing for maintenance.

FSPm = MMam/12 for the basic model.

FSPm = MMnam/12 for the intermediate model.

DATA DEFINITION

Name: Integration and Testing

Description:

Integration and Testing is a phase distribution percentage by mode for the basic and intermediate models.

DATA DEFINITION

Name: IT

Format: Real

Range: 0.00 - 1.00

Field/Cell in Files: CBO.ICF, CBS.ICF, CBE.ICF, CIO.ICF, CIS.ICF, CIE.ICF, BESP.ITB, BESPAD.ITB, IESP.ITB, IESPAD.ITB. Located in spreadsheet cell #D50.

Processing: Computed by CALCEFSC.IPF and placed into spreadsheet cell #D50 for later reports or graphing.

Description: Phase Distribution of Effort allocated to Integration and Testing.

DATA DEFINITION

Name: KDSI

Format: Real

Range: 2.0-512.0

Field/Cell in Files: CBO.ICF, CBS.ICF, CBE.ICF, CIO.ICF, CIS.ICF, CIE.ICF, BES.ITB, BESP.ITB, BESPAD.ITB, BESM.ITB, BESMAD.ITB, IES.ITB, IESP.ITB, IESPAD.ITB, IESM.ITB, IESMAD.ITB. Placed in spreadsheet cell #D5 by the user.

Processing: The user inputs the KDSI value. After the other values are loaded, the user presses a function key which performs DEVPARBA or DEVPARMS for the basic or intermediate models respectively. The program checks to ensure that the KDSI is in the allowable range of 2-512. If not, an error message is displayed and the user is allowed to input the data again. KDSI is used to compute man-months and productivity in mode dependent formulae. KDSI is evaluated by EVALKDSI.IPF to determine whether the KDSI is standard (2, 8, 32 ,128, 512), or ,if non-standard, between which values it falls. KDSI is use in table lookups on the spreadsheet. Values for non-standard KDSI's must be interpolated by INTERPOL.IPF.

Description: Thousands of lines of Delivered Source Instructions or Lines of Code.

DATA DEFINITION

Name: MACAP

Format: Real

Range: 0.00 - 1.00

Field/Cell in Files: CIO.ICF, CIS.ICF, CIE.ICF, IESM.ITB, IESMAD.ITB. Located in spreadsheet cell #F126.

Processing: The value is defaulted to 1.0 (nominal) on the spreadsheet. The user can change this to another value displayed on the spreadsheet. Used by CALCEAFM.IPF to compute the maintenance effort adjustment factor, EAFm.

Description:

Analyst Capability Attribute. One of the 14 cost driver multipliers used in obtaining EAFm.

DATA DEFINITION

Name: MAEXP

Format: Real

Range: 0.00 - 1.00

Field/Cell in Files: CIO.ICF, CIS.ICF, CIE.ICF, IESM.ITB, IESMAD.ITB. Located in spreadsheet cell #F127.

Processing: See MACAP.

Description:

Applications experience. One of the 14 cost driver multipliers used in obtaining EAFm.

DATA DEFINITION

Name: Man-Months

Description:

Man-Months of Effort. The MM variables in this data dictionary include MM, MMadj, MMam, MMnam, and MMnom.

DATA DEFINITION

Name: MANIT

Format: Real

Range: 0.00-0.99

Field/Cell in Files: CBO.ICF, CBS.ICF, CBE.ICF, CIO.ICF, CIS.ICF, CIE.ICF, BESPAD.ITB, IESPAD.ITB. Placed in spreadsheet cell #H73

Processing: Computed by CALCDPAD.IPF and placed into spreadsheet cell #H73 for display and possible graphing or reports.

Description: The activity distribution % of Effort devoted to developing/ maintaining manuals in the integration and testing phase of development.

DATA DEFINITION

Name: MANPD

Format: Real

Range: 0.00-0.99

Field/Cell in Files: CBO.ICF, CBS.ICF, CBE.ICF, CIO.ICF, CIS.ICF, CIE.ICF, BESPAD.ITB, IESPAD.ITB. Located in spreadsheet cell #D73.

Processing: Computed by CALCDPAD.IPF and placed into spreadsheet cell #D73 for later reports or graphing.

Description: The activity distribution % of Effort that is devoted to the developing/maintaining Manuals during the product design phase.

DATA DEFINITION

Name: MANPROG

Format: Real

Range: 0.00-0.99

Field/Cell in Files: CBO.ICF, CBS.ICF, CBE.ICF, CIO.ICF, CIS.ICF, CIE.ICF, BESPAD.ITB, IESPAD.ITB. Located in spreadsheet cell #F73.

Processing: Computed by CALCDPAD.IPF and placed into spreadsheet cell #F68 for later reports or graphing.

Description: The activity distribution % of Effort that is devoted to developing/maintaing manuals during the programming phase.

DATA DEFINITION

Name: MCPLX

Format: Real

Range: 0.00 - 1.00

Field/Cell in Files: CIO.ICF, CIS.ICF, CIE.ICF, IESM.ITB, IESMAD.ITB. Located in spreadsheet cell #F121.

Processing: See MACAP.

Description:

Project Complexity attribute. One of the 14 cost driver multipliers used in obtaining EAFm.

DATA DEFINITION

Name: MCQ

Format: Real

Range: 0.00-0.99

Field/Cell in Files: CBO.ICF, CBS.ICF, CBE.ICF, CIO.ICF, CIS.ICF, CIE.ICF, BESMAD.ITB, IESMAD.ITB. Located in spreadsheet cell #D90.

Processing: Computed by CALCMAPA.IPF and placed into spreadsheet cell #D90.

Description: The activity distribution % of Effort devoted to the Configuration Management and Quality Assurance, CM/QA, activity of maintenance.

DATA DEFINITION

Name: MDATA

Format: Real

Range: 0.00 - 1.00

Field/Cell in Files: CIO.ICF, CIS.ICF, CIE.ICF, IESM.ITB, IESMAD.ITB. Located in spreadsheet cell #F120.

Processing: See MACAP.

Description:

Data Base Size attribute. One of the 14 cost driver multipliers used in obtaining EAFm.

DATA DEFINITION

Name: MLEXP

Format: Real

Range: 0.00 - 1.00

Field/Cell in Files: CIO.ICF, CIS.ICF, CIE.ICF, IESM.ITB, IESMAD.ITB. Located in spreadsheet cell #F130.

Processing: See MACAP.

Description:

Programming Language Experience. One of the 14 cost driver multipliers used in obtaining EAFm.

DATA DEFINITION

Name: MM

Format: Real

Range: Positive Numbers

Field/Cell in Files: CBO.ICF, CBS.ICF, CBE.ICF, BESPAD.ITB, BES.ITB, BESP.ITB, BESM.ITB, BESMAD.ITB. Placed in spreadsheet cell #D12.

Processing: Calculated in DEVPARBA.IPF for the basic model.

Description: Development effort. Varies by mode:

Organic:	MM = 2.4(KDSI) to the 1.05 power
Semidetached:	MM = 3.0(KDSI) to the 1.12 power
Embedded:	MM = 3.6(KDSI) to the 1.20 power

DATA DEFINITION

Name: MMadj

Format: Real

Range: positive number

Field/Cell in Files: CIO.ICF, CIS.ICF, CIE.ICF, IESPAD.ITB, IESP.ITB, IESM.ITB, IES.ITB, IESMAD.ITB. Placed in spreadsheet cell #D11.

Processing: Calculated in DEVPARMS.IPF.

Description: Man-month average adjusted effort.

MMadj = MMnom x EAF

DATA DEFINITION

Name: MMam

Format: Real

Range: positive numbers

Field/Cell in Files: CBO.ICF, CBS.ICF, CBE.ICF, BESMAD.ITB, BESM.ITB. Placed in spreadsheet cell #M12.

Processing: Computed by CALCMDAT.IPF

Description: Annual Maintenance Effort.
MMam = ACT x MM for the basic model

DATA DEFINITION

Name: MMAN

Format: Real

Range: 0.00-0.99

Field/Cell in Files: CBO.ICF, CBS.ICF, CBE.ICF, CIO.ICF, CIS.ICF, CIE.ICF, BESPAD.ITB, IESPAD.ITB. Located in spreadsheet cell #D91

Processing: Computed by CALCMAPA.IPF and placed into spreadsheet cell #D91.

Description: The activity distribution % of Effort that is devoted to developing/maintaining manuals for maintenance.

DATA DEFINITION

Name: MMnam

Format: Real

Range: positive numbers

Field/Cell in Files: CIO.ICF, CIS.ICF, CIE.ICF, IESM.ITB, IESMAD.ITB, IESPAD.ITB. Placed in spreadsheet cell #M12.

Processing: Calculated by CALCMDAT.IPF.

Description: Nominal Annual Maintenance Effort.

MMnam = ACT x MMnom for the intermediate model.

DATA DEFINITION

Name: MMnom

Format: Real

Range: Positive Numbers

Field/Cell in Files: CIO.ICF, CIS.ICF, CIE.ICF, IESPAD.ITB, IES.ITB, IESP.ITB, IESM.ITB, IESMAD.ITB. Placed in spreadsheet cell #D12.

Processing: Calculated in DEVPARMS.IPF for the intermediate model.

Description: Nominal development effort. Sometimes called MMnom.

Varies by mode:

Organic:	MMnom = 3.2(KDSI) to the 1.05 power
Semidetached:	MMnom = 3.0(KDSI) to the 1.12 power
Embedded:	MMnom = 2.8(KDSI) to the 1.20 power

DATA DEFINITION

Name: MMODP

Format: Real

Range: 0.00 - 1.00

Field/Cell in Files: CIO.ICF, CIS.ICF, CIE.ICF, IESM.ITB, IESMAD.ITB. Located in spreadsheet cell #F131.

Processing: See MACAP.

Description:

Use of Modern Programming Practices. One of the 14 cost driver multipliers used in obtaining EAFm.

DATA DEFINITION

Name: MODE

Format: Real

Range: 1,2, or 3

Field/Cell in Files: CBO.ICF, CBS.ICF, CBE.ICF, CIO.ICF, CIS.ICF, CIE.ICF, BES.ITB,BESP.ITB, BESPAD.ITB,BESM.ITB, BESMAD.ITB, IES.ITB, IESP.ITB, IESPAD.ITB, IESM.ITB, IESMAD.ITB. Located in spreadsheet cell #B20

Processing: User inputs mode selection via a function key in SETUPBAS.IPF for the basic model or SETUPINT.IPF for the intermediate model. The file loads the spreadsheet loading file, SSLODBAS.IPF or SSLODINT.IPF with the selected mode. SSLOD*.IPF selects the correct spreadsheet for the model-mode combination. (CBO.,CBS.,CBE.,CIO.,CIS., or CIE.ICF) The spreadsheets contain table values specific to each model-mode combination. The mode is also used DEVPARMS/DEVPARBA to select the set of equations used.

Description: Modes of software development defined by characteristics:

<1> Organic: small team, in-house.

<2> Semidetached: combination of embedded and organic modes.

<3> Embedded: large team, tight schedule

DATA DEFINITION

Name: MODEL

Format: Real

Range: 1,2

Field/Cell in Files: CBO.ICF, CBS.ICF, CBE.ICF, CIO.ICF, CIS.ICF, CIE.ICF, BES.ITB,BESP.ITB, BESPAD.ITB,BESM.ITB, BESMAD.ITB, IES.ITB, IESP.ITB, IESPAD.ITB, IESM.ITB, IESMAD.ITB. Located in spreadsheet cell #A20

Processing: The user select the model via predefined function keys in the COCO.IPF file. The function selected performs the next file, SETUPBAS.IPF or SETUPINT.IPF, depending on whether the basic or intermediate model was chosen.

Description: Model characteristics:

<1> Basic: KDSI, PCpMM, MODE, and ACT are inputs.

<2> Intermediate: Similar to Basic. Also includes cost drivers and different equation coefficients/exponents.

DATA DEFINITION

Name: MODELMOD

Format: Literal

Range:
Model: Basic or Intermediate

Mode: Organic, Semidetached, or Embedded

Field/Cell in Files: CBO.ICF, CBS.ICF, CBE.ICF, CIO.ICF, CIS.ICF, CIE.ICF, BES.ITB, BESP.ITB, BESPAD.ITB, BESM.ITB, BESMAD.ITB, IES.ITB, IESP.ITB, IESPAD.ITB, IESM.ITB, IESMAD.ITB. Located in spreadsheet cell #D20

Processing: Result of user model and mode selection via function keys at the start of the program or of another iteration of the program.

Description: The model-mode combination is one of the following:

<1> Basic Organic <2> Intermediate Organic

<3> Basic Semidetached <4> Intermediate Semidetached

<5> Basic Embedded <6> Intermediate Embedded

DATA DEFINITION

Name: MPCAP

Format: Real

Range: 0.00 - 1.00

Field/Cell in Files: CIO.ICF, CIS.ICF, CIE.ICF, IESM.ITB, IESMAD.ITB. Located in spreadsheet cell #F128

Processing: See MACAP.

Description:

Programmer Capability. One of the 14 cost driver multipliers used in obtaining EAFm.

DATA DEFINITION

Name: MPCOST

Format: Real

Range: positive number

Field/Cell in Files: CBO.ICF, CBS.ICF, CBE.ICF, CIO.ICF, CIS.ICF, CIE.ICF, BESM.ITB, BESMAD.ITB, IESM.ITB, IESMAD.ITB. Placed in spreadsheet cell #M5.

Processing: Input by the user.

Description: Average personnel cost per Man-month (MM) during maintenance.

DATA DEFINITION

Name: MPD

Format: Real

Range: 0.00 - 1.00

Field/Cell in Files: CBO.ICF, CBS.ICF, CBE.ICF, CIO.ICF, CIS.ICF, CIE.ICF, BESMAD.ITB, IESMAD.ITB. Located in spreadsheet cell #D85.

Processing: Computed by CALCMAPA.IPF and placed into spreadsheet cell #D85.

Description:

The Activity Distribution % that is devoted to product design for maintenance.

DATA DEFINITION

Name: MPO

Format: Real

Range: 0.00 - 1.00

Field/Cell in Files: CBO.ICF, CBS.ICF, CBE.ICF, CIO.ICF, CIS.ICF, CIE.ICF, BESMAD.ITB, IESMAD.ITB. Located in spreadsheet cell #D89.

Processing: Computed by CALCMAPA.IPF and placed into spreadsheet cell #D89.

Description:

The Activity Distribution % for maintenance that is devoted to the project office.

DATA DEFINITION

Name: MPROG

Format: Real

Range: 0.00 - 1.00

Field/Cell in Files: CBO.ICF, CBS.ICF, CBE.ICF, CIO.ICF, CIS.ICF, CIE.ICF, BESMAD.ITB, IESMAD.ITB. Located in spreadsheet cell #D86.

Processing: Computed by CALCMAPA.IPF and placed into spreadsheet cell #D86.

Description:

The percent of the activity distribution of maintenance allocated to programming.

DATA DEFINITION

Name: MRA

Format: Real

Range: 0.00-0.99

Field/Cell in Files: CBO.ICF, CBS.ICF, CBE.ICF, CIO.ICF, CIS.ICF, CIE.ICF, BESMAD.ITB, IESMAD.ITB. Located in cell #D84.

Processing: Computed by CALCMAPA.IPF and placed into spreadsheet cell #D84.

Description: The activity distribution % of Effort that is devoted to the Requirements Analysis for maintenance.

DATA DEFINITION

Name: MRELY

Format: Real

Range: 0.00 - 1.00

Field/Cell in Files: CIO.ICF, CIS.ICF, CIE.ICF, IESM.ITB, IESMAD.ITB, Placed in spreadsheet cell #F119.

Processing: See MACAP.

Description:

Required Software Reliability. One of the 14 cost driver multipliers used in obtaining EAFm.

DATA DEFINITION

Name: MSTOR

Format: Real

Range: 0.00 - 1.00

Field/Cell in Files: CIO.ICF, CIS.ICF, CIE.ICF, IESM.ITB, IESMAD.ITB. Placed in spreadsheet cell #F123.

Processing: See MACAP.

Description:

Main Storage Constraint. One of the 14 cost driver multipliers used in obtaining EAFm.

DATA DEFINITION

Name: MTEST

Format: Real

Range: 0.00-0.99

Field/Cell in Files: CBO.ICF, CBS.ICF, CBE.ICF, CIO.ICF, CIS.ICF, CIE.ICF, BESMAD.ITB, IESMAD.ITB. Located in spreadsheet cell #D87.

Processing: Computed by CALCMAPA.IPF and placed into spreadsheet cell #D87.

Description: The activity distribution % of Maintenance devoted to the Test Planning activity.

DATA DEFINITION

Name: MTIME

Format: Real

Range: 0.00 - 1.00

Field/Cell in Files: CIO.ICF, CIS.ICF, CIE.ICF, IESM.ITB, IESMAD.ITB. Placed in spreadsheet cell #F122.

Processing: See MACAP.

Description:

Execution Time Constraint. One of the 14 cost driver multipliers used in obtaining EAFm.

DATA DEFINITION

Name: MTOOL

Format: Real

Range: 0.00 - 1.00

Field/Cell in Files: CIO.ICF, CIS.ICF, CIE.ICF, IESM.ITB, IESMAD.ITB, Placed in spreadsheet cell #F132.

Processing: See MACAP.

Description:

Use of Software Tools. One of the 14 cost driver multipliers used in obtaining EAFm.

DATA DEFINITION

Name: MTURN

Format: Real

Range: 0.00 - 1.00

Field/Cell in Files: CIO.ICF, CIS.ICF, CIE.ICF, IESM.ITB, IESMAD.ITB. Placed in spreadsheet cell #F125.

Processing: See MACAP.

Description:

Computer Turnaround Time. One of the 14 cost driver multipliers used in obtaining EAFm.

DATA DEFINITION

Name: MVV

Format: Real

Range: 0.00-0.99

Field/Cell in Files: CBO.ICF, CBS.ICF, CBE.ICF, CIO.ICF, CIS.ICF, CIE.ICF, BESMAD.ITB, IESMAD.ITB. Located in spreadsheet cell #D88.

Processing: Computed by CALCMAPA.IPF and placed into spreadsheet cell #D88.

Description: The activity distribution % of Maintenance devoted to Verification and Validation.

DATA DEFINITION

Name: MVEXP

Format: Real

Range: 0.00 - 1.00

Field/Cell in Files: CIO.ICF, CIS.ICF, CIE.ICF, IESM.ITB, IESMAD.ITB. Placed in spreadsheet cell #F129.

Processing: See MACAP.

Description:

Virtual Machine Experience. One of the 14 cost driver multipliers used in obtaining EAFm.

DATA DEFINITION

Name: MVIRT

Format: Real

Range: 0.00 - 1.00

Field/Cell in Files: CIO.ICF, CIS.ICF, CIE.ICF, IESP.ITB, IESMAD.ITB. Placed in spreadsheet cell #F124.

Processing: See MACAP.

Description:

Virtual Machine Volatility. One of the 14 cost driver multipliers used in obtaining EAFm.

DATA DEFINITION

Name: PCOST

Format: Real

Range: positive number

Field/Cell in Files: CBO.ICF, CBS.ICF, CBE.ICF, CIO.ICF, CIS.ICF, CIE.ICF, BES.ITB, BESP.ITB, BESPAD.ITB, BESM.ITB, BESMAD.ITB, IES.ITB, IESP.ITB, IESPAD.ITB, IESM.ITB, IESMAD.ITB. Located in spreadsheet cell #D6.

Processing: Input by the user.

Description: Average personnel cost per Man-month (MM) during development.

DATA DEFINITION

Name: PDIT

Format: Real

Range: 0.00-0.99

Field/Cell in Files: CBO.ICF, CBS.ICF, CBE.ICF, CIO.ICF, CIS.ICF, CIE.ICF, BESPAD.ITB, IESPAD.ITB. Placed in spreadsheet cell #H67

Processing: Computed by CALCDPAD.IPF and placed into spreadsheet cell #H67 for display and possible graphing or reports.

Description: The activity distribution % of Effort devoted to Product Design during the integration and testing phase of development.

DATA DEFINITION

Name: PDPD

Format: Real

Range: 0.00-0.99

Field/Cell in Files: CBO.ICF, CBS.ICF, CBE.ICF, CIO.ICF, CIS.ICF, CIE.ICF, BESPAD.ITB, IESPAD.ITB. Located in spreadsheet cell #D67.

Processing: Computed by CALCDPAD.ITB and placed into spreadsheet cell #D67 for display and possible graphing or reports.

Description: The activity distribution % of Effort that is devoted to product design during the product design phase.

DATA DEFINITION

Name: PDPROG

Format: Real

Range: 0.00-0.99

Field/Cell in Files: CBO.ICF, CBS.ICF, CBE.ICF, CIO.ICF, CIS.ICF, CIE.ICF, BESPAD.ITB, IESPAD.ITB. Located in spreadsheet cell #F67.

Processing: Computed by CALCDPAD.IPF and placed into spreadsheet cell #F67 for later reports or graphing.

Description: The activity distribution % of Effort that is devoted to the product design during the programming phase of development.

DATA DEFINITION

Name: PHASE DISTRIBUTION %

Variable Names: See PHASE DISTRIBUTION OF EFFORT and PHASE DISTRIBUTION OF SCHEDULE.

Description: The percentage of Effort, (MM), or Schedule, (TDEV), devoted to a certain phase of development.

DATA DEFINITION

Name: PHASE DISTRIBUTION OF EFFORT

Variables: PRODES, PROG, DETDES, CUT, and IT. See the respective variables in in this data dictionary for further explanation.

Description: Phase Distribution of Development Effort

for Basic: MM x Phase Distribution %

for Intermediate: MMadj x Phase Distribution %

DATA DEFINITION

Name: PHASE DISTRIBUTION OF SCHEDULE

Variables: SCHEDPD, SPROG, and SIT.

Description: Phase Distribution of Development Schedule = SCHEDULE x Phase Distribution %.

DATA DEFINITION

Name: Plans and Requirements

Description:

Plans and Requirements is the Phase preceeding the Development Phase. It is considered to take 6%. Thus Plans and Requirements and Development are 106% of Development.

DATA DEFINITION

Name: POIT

Format: Real

Range: 0.00-0.99

Field/Cell in Files: CBO.ICF, CBS.ICF, CBE.ICF, CIO.ICF, CIS.ICF, CIE.ICF, BESPAD.ITB, IESPAD.ITB. Placed in cell #H71.

Processing: Computed by CALCDPAD.IPF and placed into spreadsheet cell #H71 for display and possible graphing or reports.

Description: The activity distribution % of Effort that is devoted to the Project Office during the Integration and Testing Phase of development.

DATA DEFINITION

Name: POPD

Format: Real

Range: 0.00-0.99

Field/Cell in Files: CBO.ICF, CBS.ICF, CBE.ICF, CIO.ICF, CIS.ICF, CIE.ICF, BESPAD.ITB, IESPAD.ITB. Located in spreadsheet cell #D71.

Processing: Computed by CALCDPAD.IPF and placed into spreadsheet cell #D71 for later reports or graphing.

Description: The activity distribution % of Effort that is devoted to the project office during the product design phase.

DATA DEFINITION

Name: POPROG

Format: Real

Range: 0.00-0.99

Field/Cell in Files: CBO.ICF, CBS.ICF, CBE.ICF, CIO.ICF, CIS.ICF, CIE.ICF, BESPAD.ITB, IESPAD.ITB. Located in spreadsheet cell #F71.

Processing: Computed by CALCDPAD.ITB and loaded into spreadsheet cell #F71 for later reports or graphing.

Description: The activity distribution % of Effort that is devoted to the project office during the programming phase.

DATA DEFINITION

Name: PRODES

Format: Real

Range: 0.00 - 1.00

Field/Cell in Files: CBO.ICF, CBS.ICF, CBE.ICF, CIO.ICF, CIS.ICF, CIE.ICF, BESP.ITB, BESPAD.ITB, IESP.ITB, IESPAD.ITB. Located in spreadsheet cell #D46.

Processing: Computed in CALCEFSC.IPF and placed into spreadsheet cell #D46 for later reports or graphing.

Description: Phase Distribution of Effort allocated to Product Design.

DATA DEFINITION

Name: PRODUCT

Format: Real

Range: positive numbers

Field/Cell in Files: CBO.ICF, CBS.ICF, CBE.ICF, CIO.ICF, CIS.ICF, CIE.ICF, BES.ITB, BESP.ITB, BESPAD.ITB, BESM.ITB, BESMAD.ITB, IES.ITB, IESP.ITB, IESPAD.ITB, IESM.ITB, IESMAD.ITB. Placed in spreadsheet cell #D13.

Processing: Calculated in DEVPARBA.IPF for the basic model and DEVPARMS.IPF for the intermediate model.

Description: The number of KDSI developed per MM. (KDSI/MM). This is also referred to as the Productivity Index (PI).

DATA DEFINITION

Name: PROG

Format: Real

Range: 0.00 - 1.00

Field/Cell in Files: CBO.ICF, CBS.ICF, CBE.ICF, CIO.ICF, CIS.ICF, CIE.ICF, BESP.ITB, BESPAD.ITB, IESP.ITB, IESPAD.ITB. Located in spreadsheet cell #D47.

Processing: Computed in CALCEFSC.IPF and placed into spreadsheet cell #D47 for later reports or graphing.

Description: Phase Distribution of Effort allocated to Programming. It is the sum of DETDES and CUT.

DATA DEFINITION

Name: PROGIT

Format: Real

Range: 0.00-0.99

Field/Cell in Files: CBO.ICF, CBS.ICF, CBE.ICF, CIO.ICF, CIS.ICF, CIE.ICF, BESPAD.ITB, IESPAD.ITB. Placed in cell #H68.

Processing: Computed by CALCDPAD.IPF and placed into spreadsheet cell #H68 for display and possible graphing or reports.

Description: The activity distribution % of Effort devoted to Programming during the integration and testing phase of development.

DATA DEFINITION

Name: PROGPROG

Format: Real

Range: 0.00-0.99

Field/Cell in Files: CBO.ICF, CBS.ICF, CBE.ICF, CIO.ICF, CIS.ICF, CIE.ICF, BESPAD.ITB, IESPAD.ITB. Located in spreadsheet cell #F68.

Processing: Computed by CALCDPAD.IPF and placed into spreadsheet cell #F68 for later reports or graphing.

Description: The activity distribution % of Effort that is devoted to programming during the programming phase.

DATA DEFINITION
Name: Programming Description: Programming is a phase in the distribution of effort and schedule by mode and also an activity in the distribution by phase.

DATA DEFINITION

Name: PROGPD

Format: Real

Range: 0.00-0.99

Field/Cell in Files: CBO.ICF, CBS.ICF, CBE.ICF, CIO.ICF, CIS.ICF, CIE.ICF, BESPAD.ITB, IESPAD.ITB. Located in spreadsheet cell #D68.

Processing: Computed by CALCDPAD.ITB and placed into spreadsheet cell #D68 for later reports or graphing.

Description: The activity distribution % of Effort that is devoted to the Programming activity during the product design phase.

DATA DEFINITION

Name: Project Office

Description:

The activity distribution % of Effort or Schedule by mode that is devoted to Project Office tasks during development/maintenance.

DATA DEFINITION

Name: RAIT

Format: Real

Range: 0.00-0.99

Field/Cell in Files: CBO.ICF, CBS.ICF, CBE.ICF, CIO.ICF, CIS.ICF, CIE.ICF, BESPAD.ITB, IESPAD.ITB. Placed in cell #H66.

Processing: Computed by CALCDPAD.IPF and placed into spreadsheet cell #H66 for display and possible graphing or reports.

Description: The activity distribution % of Effort devoted to the Requirements Analysis for integration and testing during development.

DATA DEFINITION

Name: RAPD

Format: Real

Range: 0.00-0.99

Field/Cell in Files: CBO.ICF, CBS.ICF, CBE.ICF, CIO.ICF, CIS.ICF, CIE.ICF, BESPAD.ITB, IESPAD.ITB. Located in spreadsheet cell #D66

Processing: Computed by CALCDPAD.ITB and placed into spreadsheet cell #D66 for later reports or graphing.

Description: The activity distribution % of Effort that is devoted to Requirements Analysis during the product design phase.

DATA DEFINITION

Name: RAPROG

Format: Real

Range: 0.00-0.99

Field/Cell in Files: CBO.ICF, CBS.ICF, CBE.ICF, CIO.ICF, CIS.ICF, CIE.ICF, BESPAD.ITB, IESPAD.ITB. Located in spreadsheet cell #F66.

Processing: Computed by CALCDPAD.IPF and placed into spreadsheet cell #F66 for later reports or graphing.

Description: The activity distribution % of Effort that is devoted to requirements analysis during the programming phase.

DATA DEFINITION

Name: Rating

Description:

Cost Drivers for the intermediate and detailed models are given ratings as well as Effort Multipliers by Phase for the detailed model. Ratings range from Very Low to Extra High.

DATA DEFINITION

Name: SCHEDPD

Format: Real

Range: 0.00-1.00

Field/Cell in Files: CBO.ICF, CBS.ICF, CBE.ICF, CIO.ICF, CIS.ICF, CIE.ICF, BESP.ITB, BESPAD.ITB, IESP.ITB, IESPAD.ITB. Located in spreadsheet cell #D53.

Processing: Computed by CALCEFSC.IPF and placed into spreadsheet cell #D53 for later reports or graphing.

Description: Phase Distribution of Schedule allocated to Product Design.

DATA DEFINITION

Name: SCHEDULE

Format: Real

Range: positive number

Field/Cell in Files: CBO.ICF, CBS.ICF, CBE.ICF, CIO.ICF, CIS.ICF, CIE.ICF, BESPAD.ITB, IESPAD.ITB, BES.ITB, IES.ITB, IESP.ITB, BESP.ITB, BESM.ITB, IESM.ITB, BESMAD.ITB, IESMAD.ITB. Located in spreadsheet cell #D14.

Processing: Calculated by DEVPARBA.IPF for the basic model and DEVPARMS.IPF for the intermediate model. Also referred to as TDEV.

Description: Total Development Schedule. Sometimes referred to as TDEV or Schedule. Varies by mode:

Organic:	2.5(MM) to the 0.38 power
Semidetached:	2.5(MM) to the 0.35 power
Organic:	2.5(MM) to the 0.32 power

DATA DEFINITION

Name: Schedule Coefficient

Format: Real

Range: positive numbers

Located in Files: DEVPARBA.IPF for the basic model and DEVPARMS.IPF for the intermediate.

Description:

Coefficient for the equation for SCHEDULE. See Effort Coefficients.

DATA DEFINITION

Name: Schedule Exponent

Format: Real

Range: positive numbers

Located in Files: DEVPARBA.IPF for the basic model and DEVPARMS.IPF for the intermediate model.

Description:

Exponent for the equation for SCHEDULE. See Effort Coefficients.

DATA DEFINITION

Name: SIT

Format: Real

Range: 0.00-0.99

Field/Cell in Files: CBO.ICF, CBS.ICF, CBE.ICF, CIO.ICF, CIS.ICF, CIE.ICF, BESP.ITB, BESPAD.ITB, IESP.ITB, IESPAD.ITB. Located in spreadsheet cell #D55.

Processing: Computed by CALCEFSC.IPF and placed into spreadsheet cell #D55 for later reports or graphing.

Description: Phase Distribution of Schedule allocated to Integration and Testing.

DATA DEFINITION

Name: SPROG

Format: Real

Range: 0.00-0.99

Field/Cell in Files: CBO.ICF, CBS.ICF, CBE.ICF, CIO.ICF, CIS.ICF, CIE.ICF, BESP.ITB, BESPAD.ITB, IESP.ITB, IESPAD.ITB. Located in spreadsheet cell #D54.

Processing: Computed by CALCEFSC.IPG and placed into spreadsheet cell #D54 for later reports or graphing.

Description: Phase Distribution of Schedule allocated to Programming.

DATA DEFINITION

Name: TDEV

Description:

Total Development Schedule. Also referred to as Schedule. See the Data Definition for Schedule.

DATA DEFINITION

Name: Test Planning

Description:

The activity distribution % of Effort or Schedule by mode that is devoted to Test Planning during development/maintenance.

DATA DEFINITION

Name: TESTIT

Format: Real

Range: 0.00-0.99

Field/Cell in Files: CBO.ICF, CBS.ICF, CBE.ICF, CIO.ICF, CIS.ICF, CIE.ICF, BESPAD.ITB, IESPAD.ITB. Placed in cell #H69.

Processing: Computed by CALCDPAD.IPF and placed into spreadsheet cell #H69 for display and possible graphing or reports.

Description: The activity distribution % of Effort devoted to Test Planning during the integration and testing phase of development.

DATA DEFINITION

Name: TESTPD

Format: Real

Range: 0.00-0.99

Field/Cell in Files: CBO.ICF, CBS.ICF, CBE.ICF, CIO.ICF, CIS.ICF, CIE.ICF, BESPAD.ITB, IESPAD.ITB. Located in spreadsheet cell #D69.

Processing: Computed by CALCDPAD.ITB and placed into spreadsheet cell #D69 for later reports or graphing.

Description: The activity distribution % of Effort that is devoted to test planning during the product design phase.

DATA DEFINITION

Name: TESTPROG

Format: Real

Range: 0.00-0.99

Field/Cell in Files: CBO.ICF, CBS.ICF, CBE.ICF, CIO.ICF, CIS.ICF, CIE.ICF, BESPAD.ITB, IESPAD.ITB. Located in spreadsheet cell #F69.

Processing: Computed by CALCDPAD.IPF and placed into spreadsheet cell #F69 for later reports or graphing.

Description: The activity distribution % of Effort that is devoted to the test planning during the programming phase.

DATA DEFINITION

Name: Verification and Validation

Description:

The activity distribution % of Effort or Schedule by mode that is devoted to Software Verification and Validation during development or maintenance.

DATA DEFINITION

Name: VVIT

Format: Real

Range: 0.00-0.99

Field/Cell in Files: CBO.ICF, CBS.ICF, CBE.ICF, CIO.ICF, CIS.ICF, CIE.ICF, BESPAD.ITB, IESPAD.ITB. Placed in cell #H70.

Processing: Computed by CALCDPAD.IPF and placed into spreadsheet cell #H70 for display and possible graphing or reports.

Description: The activity distribution % of Effort that is devoted to Verification and Validation during the Integration and Testing Phase of development.

DATA DEFINITION

Name: VVPD

Format: Real

Range: 0.00-0.99

Field/Cell in Files: CBO.ICF, CBS.ICF, CBE.ICF, CIO.ICF, CIS.ICF, CIE.ICF, BESPAD.ITB, IESPAD. Located in spreadsheet cell #D70.

Processing: Computed by CALCDPAD.ITB and placed into spreadsheet cell #D70 for later reports or graphing.

Description: The activity distribution % of Effort that is devoted to verification and validation during the product design phase.

DATA DEFINITION

Name: VVPROG

Format: Real

Range: 0.00-0.99

Field/Cell in Files: CBO.ICF, CBS.ICF, CBE.ICF, CIO.ICF, CIS.ICF, CIE.ICF, BESPAD.ITB, IESPAD.ITB. Located in spreadsheet cell #F70.

Processing: Computed by CALCDPAD.IPF and placed into spreadsheet cell #F70 for later reports or graphing.

Description: The activity distribution % of Effort that is devoted to the verification and validation during the programming phase.

APPENDIX C

COCOMO TOOL STRUCTURE CHARTS

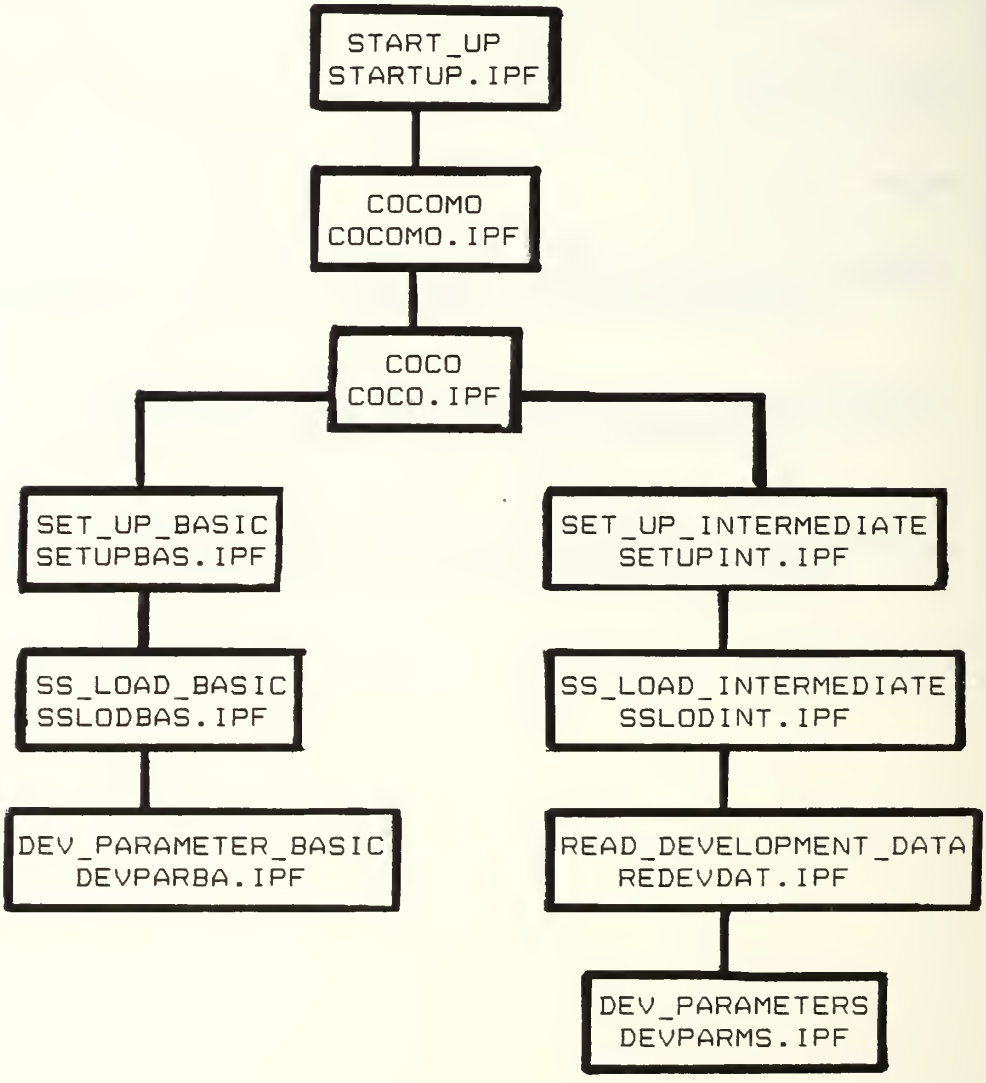


Figure C1: COCOMO Program Initiation

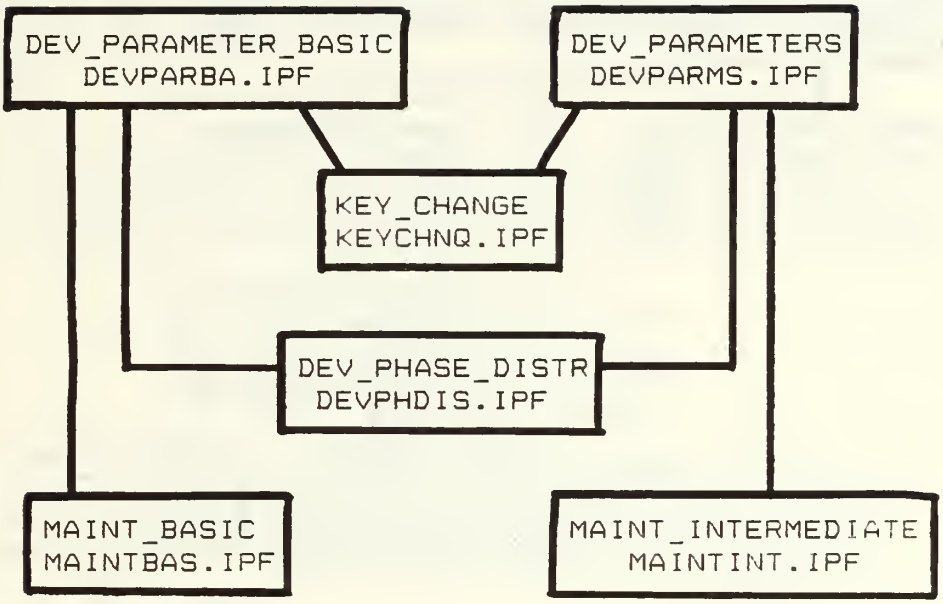
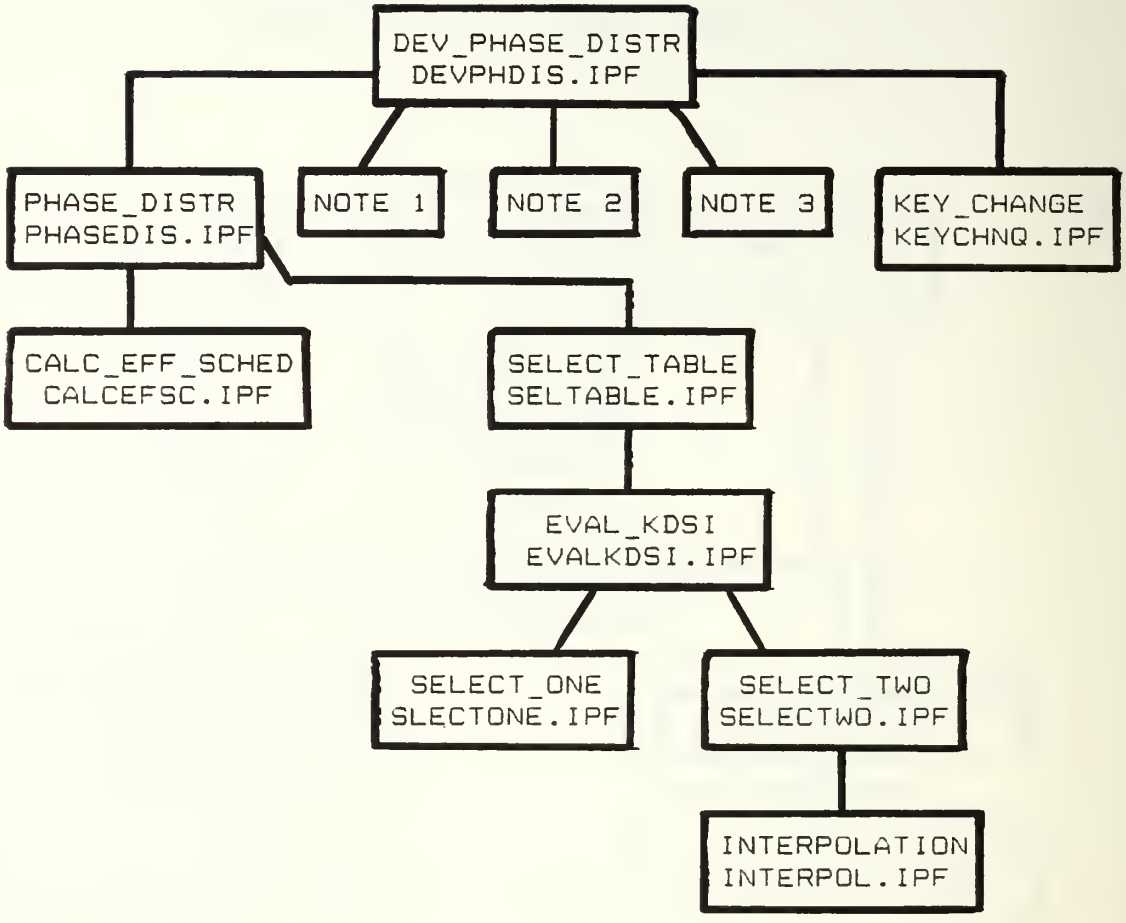
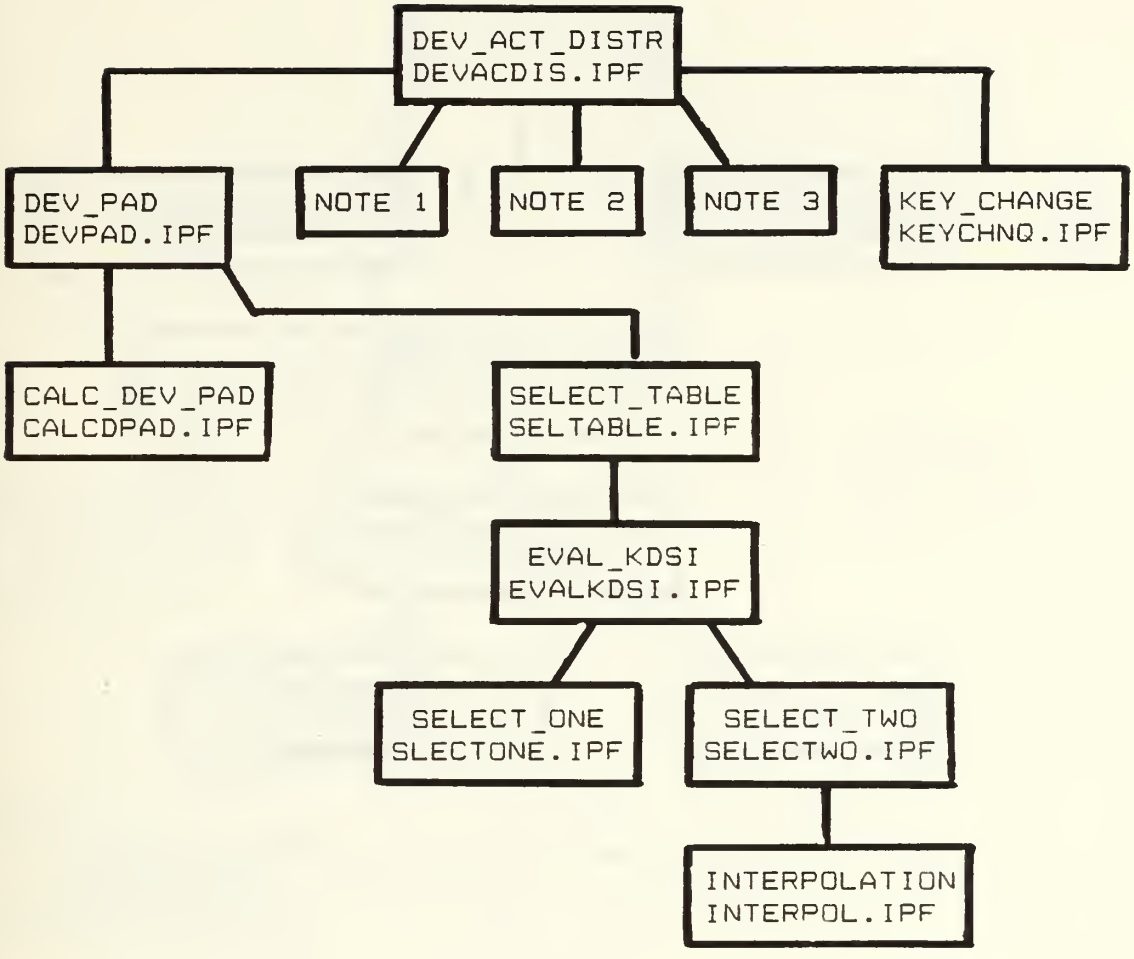


Figure C2: Effort/schedule computations and options



NOTES:
1. GRAF_PHASE_EFFORT
   GRAFPHE.IPF
2. GRAF_PHASE_SCHEDULE
   GRAFPHS.IPF
3. DEV_ACT_DISTR
   DEVACDIS.IPF

Figure C3: Development phase computations and options



NOTES:
1. GRAF_ACT_DIST_PD
   GRAFADPD.IPF
2. GRAF_ACT_DIST_PHASE
   GRAFADP.IPF
3. GRAF_ACT_DIST_INTEST
   GRAFADIT.IPF

Figure C4: Development activity computations and options

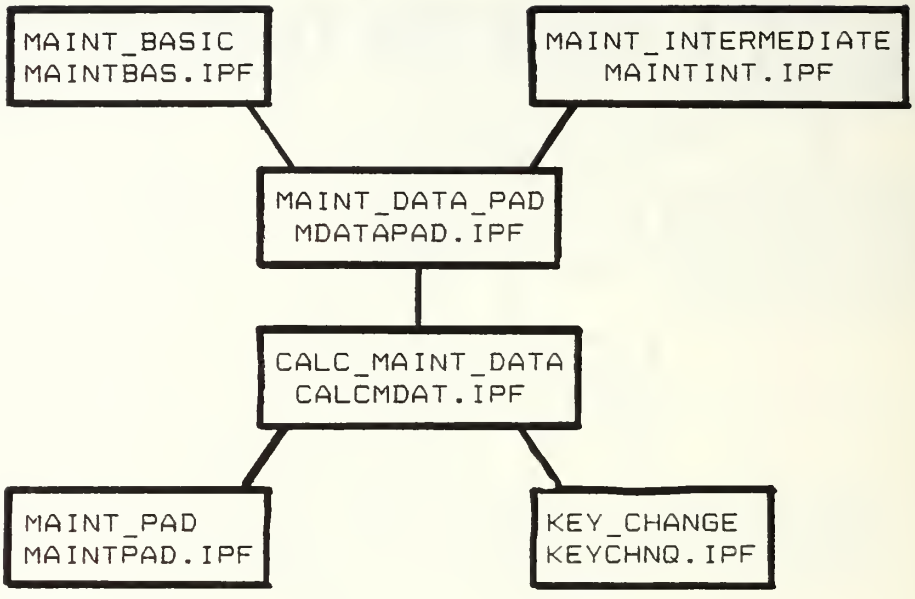
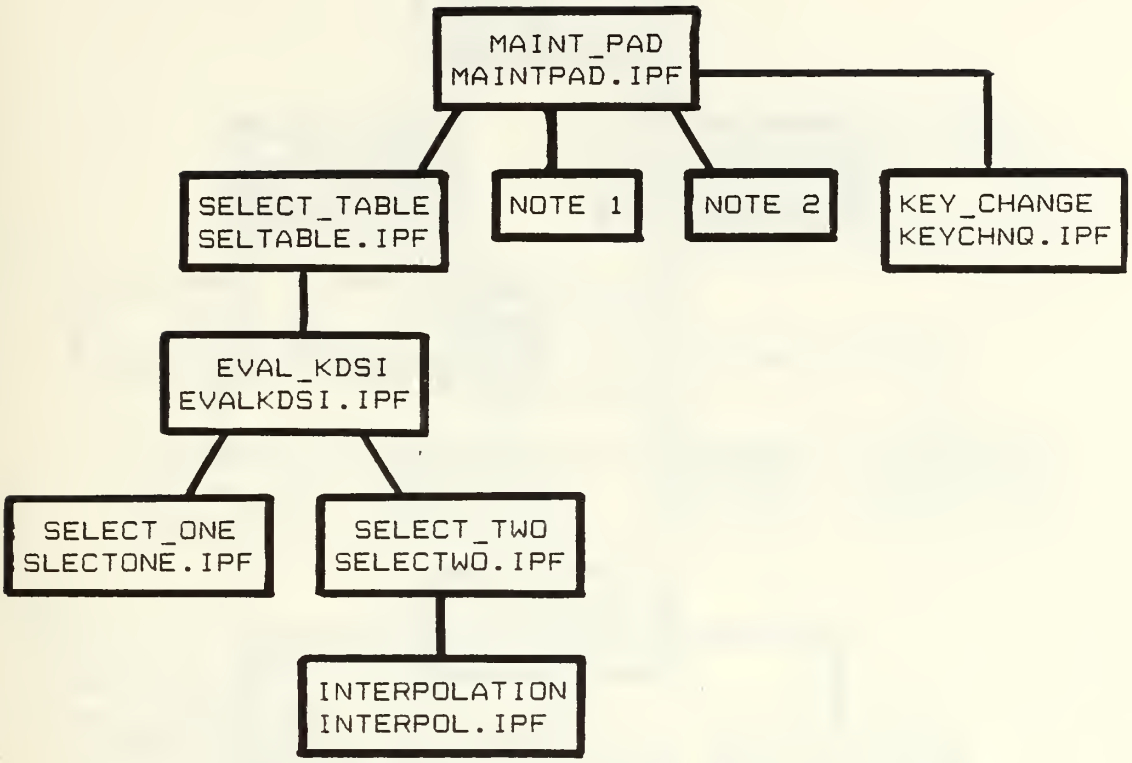


Figure C5: Maintenance computations and options



NOTES:
1. CALC_MAINT_PAD
   CALCMAPA.IPF
2. GRAF_PD_MAINT
   GRAFPDM.IPF

Figure C6: Maintenance phase activity computations/options

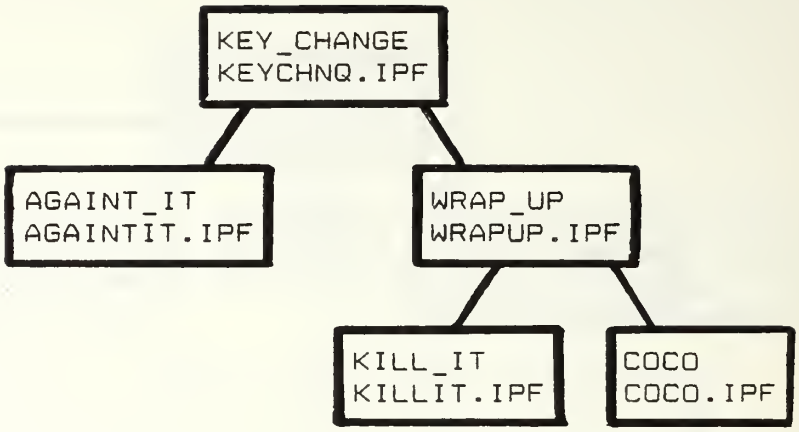


Figure C7: Program iteration and termination options

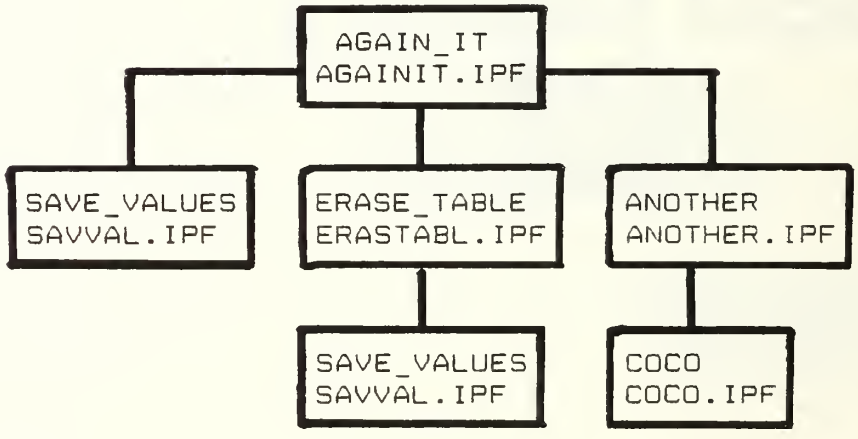
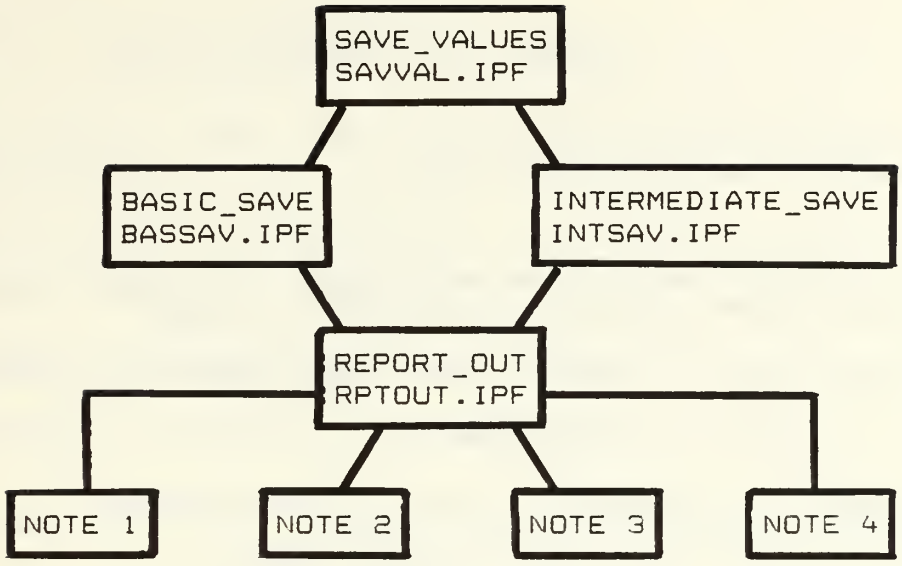


Figure C8: Program iteration options



NOTES:

1. BASIC_RPT_ONE
   BRPTONE.IPF
2. BASIC_RPT_ALL
   BRPTALL.IPF
3. INTERMEDIATE_RPT_ONE
   IRPTONE.IPF
4. INTERMEDIATE_RPT_ALL
   IRPTALL.IPF
5. Each module listed in notes 1 - 4 has the option of choosing either WRAPUP.IPF or ANOTHER.IPF

Figure C9: Program data save/report generation

APPENDIX D

COCOMO TOOL PROGRAM LISTING

```
/* STARTUP.IPF - STARTUP MODULE                                    */

/* This module acts as an auto-exec.  It begins the COCOMO*/
/* program when KMAN is invoked by the user.                       */

/* Sample Call:  None (Invoked when KMAN is entered)               */

/* Input: KMAN typed by user                                       */

/* Output: Title screen of the COCOMO program                      */

/* Sub-module: COCOMO.IPF (COCOMO MODULE)                          */

Load perform "COCOMO"

Perform "COCOMO"
```

```
/* COCOMO.IPF - COCOMO MODULE                                          */

/* This module prints the coverpage and displays the                   */
/* banner page.                                                        */

/* Sample call: PERFORM COCOMO                                         */

/* Input:   Called by STARTUP.IPF module                               */

/* Output:  Coverpage and banner page                                  */

/* Sub-module:  COCO.IPF (COCO MODULE)                                 */


/Let e.deci = 2          ! Sets spreadsheet decimal places

Clear                    ! Clears screen

Form COVERFRM            ! Coverpage
 At 3,30 put "-----------"
 At 4,30 put "C O C O M O"
 At 5,30 put "-----------"
 At 7,28  put "COnstructive COst MOdel"
 At 9,28 put  "          for "
 At 11,22 put "Estimating Software Development Cost"
 At 13,22 put "Dec 1985"
 At 15,27 put "Naval Postgraduate School"
 At 16,27 put "     Monterey, CA"
 At 23,32 put "PRESS SPACE BAR" with "B"
 At 1,1 to 24,80 put "FBBW"
 At 2,3 to 23,78 put "FWBU"
 At 23,31 to 23,48 put "FOBU"
Endform

Putform COVERFRM;              ! displays coverpage.

Wait
```

```
Form BANNERCO
 At 3,33 put "COCOMO PROGRAM"
 At 5,12 put "This decision support system program automates"
 At 6,12 put "the COCOMO method of software engineering for"
 At 7,12 put "development and maintenance.  It enables the"
 At 8,12 put "user to select one of two models (Basic or"
 At 9,12 put "Intermediate), and one of three modes"
 At 10,12 put "(Organic, Semidetached, or Embedded) for the"
 At 11,12 put "computation of development and or maintenance"
 At 12,12 put "data for a given KDSI input.  Options include"
 At 13,12 put "phase distribution calculations for development"
 At 14,12 put "or maintenance, activity distribution by phase"
 At 15,12 put "for development, graphs, reports and model/mode"
 At 16,12 put "iterations.  Iterations of data can be saved"
 At 17,12 put "for report generation.  Data can be saved"
 At 18,12 put "or erased before the program is terminated."
 At 23,28 put "PRESS SPACE BAR TO BEGIN"
 AT 1, 1 TO 24, 80 PUT "FOBW"
 AT 2, 2 TO 23, 79 PUT "FOBU"
 At 23,27 to 23,53 put "FWBU"
Endform

Putform BANNERCO;            ! Displays banner.

Release COVERFRM;

Wait

Load perform "COCO"

Release BANNERCO;

Perform "COCO"    ! Redefines function keys and loads menu
```

```
/* COCO.IPF - COCO MODULE                                              */

/* This program begins the COCOMO process. redefines                  */
/* function keys and displays the screen for COCOMO model             */
/* selection.                                                         */

/* Sample call: PERFORM COCO                                          */

/* Input: Called by COCOMO.IPF                                        */

/* Output: F key selection of COCOMO model                            */
/*         F1 - Basic model                                           */
/*         F2 - Intermediate model                                    */

/* Submodules:  SETUPBAS.IPF  (SET_UP_BASIC MODULE)                   */
/*              SETUPINT.IPF  (SET_UP_INTERMEDIATE MODULE)            */


Let e.serr = true
Release perform "COCOMO"
Let e.serr = false

/* Function keys redefined for model selection */

Redefine function 1 "  Perform \"SETUPBAS\" \13"
Redefine function 2 "  Perform \"SETUPINT\"  13"
Redefine function 3 "  \"WRONG KEY\" \13"
Redefine function 4 "  \"WRONG KEY\" \13"
Redefine function 5 "  \"WRONG KEY\" \13"
Redefine function 6 "  \"WRONG KEY\" \13"
Redefine function 7 "  \"WRONG KEY\" \13"
Redefine function 8 "  \"WRONG KEY\" \13"
Redefine function 9 "  \"WRONG KEY\" \13"
Redefine function 10 " Perform \"WRAPUP\" \13"
```

```
Form MODLFORM          ! Model selection form
 At 2,33 put "------------"
 At 3,33 put "C O C O M O"
 At 4,33 put "------------"
 At 6,23 put "TO SELECT A MODEL DEPRESS ONE"
 At 8,26 put "OF THE FOLLOWING F KEYS:"
 At 13,26 put "F1 BASIC"
 At 15,26 put "F2 INTERMEDIATE"
 At 18,26 put "F10 END PROGRAM"
 At 1,1 to 24,80 put "FBBW"
 At 2,3 to 23,78 put "FWBU"
 At 18,25 to 18,41 put "FRBU"
Endform

Load perform "SETUPBAS"
Load perform "SETUPINT"
Load perform "WRAPUP"

Putform MODLFORM;            ! Display model selection scre
```

```
/* SETUPBAS.IPF - SET_UP_BASIC MODULE                                    */

/* This module redefines F keys to load one of three                     *
/* spreadsheets for the Basic COCOMO model.                              */

/* Sample call: PERFORM "SETUPBAS"                                       */

/* Input: Function key to select development mode                        */
/*             F1 - Organic.                                             */
/*             F2 - Semidetached.                                        */
/*             F3 - Embedded.                                            */

/* Output: basic model spreadsheet                                       */

/* Sub-module: SSLODBAS  (SS_LOAD_BASIC MODULE)                          -

 MODE = 0                    ! Defines and initializes variable

/* Function keys redefined to select proper basic mode                  */
Redefine function 1 "  MODE = 1; PERFORM \"SSLODBAS\"
                                  USING \"MODE\" \13"
Redefine function 2 "  MODE = 2; PERFORM \"SSLODBAS\"
                                  USING \"MODE\" \13"
Redefine function 3 "  MODE = 3; PERFORM \"SSLODBAS\"
                                  USING \"MODE\" \13"
Redefine function 10 " \"WRONG KEY\"  \13"

Form MODEFORM           ! Displays mode selection
 At 2,33 Put "-------------"
 At 3,33 Put "C O C O M O"
 At 4,33 Put "-------------"
 At 6,23 Put "TO SELECT A MODE DEPRESS ONE"
 At 8,26 Put "OF THE FOLLOWING F KEYS:"
 At 12,26 Put "F1 BASIC ORGANIC"
 At 16,26 Put "F2 BASIC SEMIDETACHED"
 At 20,26 Put "F3 BASIC EMBEDDED"
 At 1,1 to 24,80 Put "FBBW"
 At 2,3 to 23,78 Put "FWBU"
Endform

Release perform "COCO"
Release perform "WRAPUP"
Release MODLFORM
Load perform "SSLODBAS"

Putform MODEFORM; at 24,1    ! Display mode selection screen.
```

```
/* SSLODBAS.IPF - SS_LOAD_BASIC MODULE                              */

/* This module loads a basic model spreadsheet for the             */
/* organic, semidetached, or embedded modes.                        */

/* Sample call: PERFORM "SSLODBAS" USING "MODE"                     */

/* Inputs:  MODE = Organic, Semidetached or Embedded, as            */
/*                 user selected by F key                           */

/* Output:  COCOMO Basic spreadsheet, by mode                       */

/Let e.deci = 2     ! Sets spreadsheet decimal places

Form SSLOADBAS
  At 22,26 Put "LOADING BASIC MODEL" WITH "B"
  At 22,25 to 22,46 Put "FOBU"
Endform;

Putform SSLOADBAS; at 24,1

Redefine function 1 " PERFORM \"DEVPARBA\" \13"
Redefine function 2 " \"DEPRESS F1 FIRST\" \13"
Redefine function 3 " \"DEPRESS F1 FIRST\" \13"
Redefine function 10 " PERFORM \"KEYCHNO\" \13"

Load perform "DEVPARBA"
Load perform "KEYCHNO"

Release MODEFORM
Release perform "SETUPBAS"

MODE = #A;

If MODE = 1 then                ! Organic spreadsheet
   Load "CBO" with "C";         ! "C" loads cell definitions
Else
   If MODE = 2 then             ! Semidetached spreadsheet
      Load "CBS" with "C";
   Else
      /* MODE = 3 */            ! Embedded spreadsheet
         Load "CBE" with "C";
   Endif;
Endif;
```

```
Calc = #A1                    ! Displays effort/schedule page
Calc = #D5                    ! Moves cursor to KDSI input cell
Calc                          ! Initiates the spreadsheet mode

Release SSLOADBAS
Release MODE
```

```
/* DEVPARBA.IPF - MODULE DEV_PARAMETER_BASIC                        */

/* This module calculates effort and schedule criteria for*/
/* the basic COCOMO model and its' three modes: organic,  */
/* semidetached and embedded.  The results of these       */
/* calculations are displayed to the user.                */

/* Sample call:  PERFORM "DEVPARBA" (Invoked by F-1)       */

/* Input:  KDSI = Number of thousands of delivered source */
/*                instructions                            */
/*         PCOST = Personnel costs per man-month          */

/* Output:  MAN-MONTH - effort                            */
/*          PRODUCTIVITY - delivered source instructions  */
/*                         man-month                      */
/*          TDEV - effort schedule in months              */
/*          FSP  - full time software personnel           */
/*          ANNUAL COST - development cost per year       */


Local I
Local J
Local MAXNUM

Let MAXNUM = 200    ! Number for error message delay counter

Form COMPFORM       ! Computing effort/schedule form
  At 12,24 put "COMPUTING EFFORT/SCHEDULE" with "B"
  At 12,23 to 12,51 put "fobu"
Endform;

Putform compform; at 24,1

Form LARGKDSI
  At 12,28 put "KDSI IS GREATER THEN 512" WITH "B"
  At 13,30 put "KDSI >2 OR <512 ONLY"
  At 12,26 to 13,55 put "fwbr"
Endform;

Form SMALKDSI
  At 12,30 put "KDSI IS LESS THEN 2" WITH "B"
  At 13,30 put "KDSI >2 OR <512 ONLY"
  At 12,26 to 13,55 put "fwbr"
Endform;
```

```
ERRFLAG = 0
KDSI    = #D5          ! User input KDSI
FCOST   = #D6          ! User input cost/man-month

If KDSI < 2 then

  Putform SMALKDSI; AT 24,1   ! KDSI is too small

  Let I = 0
  While I < MAXNUM do          ! Error message delay
    I = I + 1
  Endwhile

  CALC = #D5                   ! Cursor back to KDSI input
  CALC                         ! Clears error message
  ERRFLAG = 1                  ! Don't perform calculations

Else

  If KDSI > 512 then

    Putform LARGKDSI; AT 24,1   ! KDSI is too large

    Let J = 0
    While J < MAXNUM do          ! Error message delay
      J = J + 1
    Endwhile

    CALC = #D5                   ! Cursor back to KDSI input
    CALC                         ! Clears error message
    ERRFLAG = 1                  ! Don't perform calculations

  Endif;
Endif;

Let e.serr = true

* Compute development parameters. *

If ERRFLAG = 0 then  ! KDSI is >2 and <512

   #C20 = "1"
```

```
Load perform "DEVPHDIS"   ! Phase calculations
Load perform "MAINTBAS"   ! Maintenance calculations
Load perform "KEYCHNG"

Redefine function 2 " Perform \"DEVPHDIS\" \13"
Redefine function 3 " Perform \"MAINTBAS\" \13"
Redefine function 10 "Perform \"KEYCHNG\" \13"


If MODE = 1 then                ! organic mode

   #D12 = 2.4 * (EXP (1.05 * LN (KDSI)))          /* mm */

   #D13 = (1000 * KDSI) / #D12          /* productivity */

   #D14 = 2.5 * (EXP (.38 * LN (#D11)))          * tdev */

   #D15 = #D12 / #D14                            /* fsp */

   #D16 = PCOST * #D12                    /* annual cost */

   Calc;                ! Displays updated spreadsheet

Else

   If MODE = 2 then                ! semidetached mode

      #D12 = 3.0 * (EXP (1.12 * LN (KDSI)))       /* mm */

      #D13 = (1000 * KDSI) / #D12       /* productivity */

      #D14 = 2.5 * (EXP (.35 * LN (#D11)))       /* tdev */

      #D15 = #D12 / #D14                          /* fsp */

      #D16 = PCOST * #D12                 /* annual cost */

      Calc;                ! Displays updated spreadsheet
```

```
      ELSE

            ! embedded mode

            #D12 = 3.6 * (EXP (1.20 * LN (KDSI)))     /* mm */

            #D13 = (1000 * KDSI) / #D12     /* productivity */

            #D14 = 2.5 * (EXP (.32 * LN (#D11)))    /* tdev */

            #D15 = #D12 / #D14                          /* fsp */

            #D16 = PCOST * #D12                 /* annual cost */

            Calc;                  ! Displays updated spreadsheet

        Endif;
    Endif;
Endif;

Let e.serr = false

Release COMPFORM
Release LARGKDSI
Release SMALKDSI
Release ERRFLAG
Release KDSI
Release PCOST
```

```
/* SETUPINT.IPF - SET_UP_INTERMEDIATE MODULE                          */

/* This module redefines F keys to load one of three                  */
/* Spreadsheets for the Intermediate COCOMO model                     */

/* Sample call: PERFORM "SETUPINT"                                    */

/* Input: Function key to select development mode                     */
/*             F1 - Organic                                           */
/*             F2 - Semidetached                                      */
/*             F3 - Embedded                                          */

/* Output: Intermediate model spreadsheet                             */

/* Sub-module: SSLODINT (SS_LOAD_INTERMEDIATE)                        */

 MODE = 0                    ! Defines and initializes variable

Redefine function 1 "  MODE = 1; PERFORM \"SSLODINT\"
                               USING \"MODE\" \13"
Redefine function 2 "  MODE = 2; PERFORM \"SSLODINT\"
                               USING \"MODE\" \13"
Redefine function 3 "  MODE = 3; PERFORM \"SSLODINT\"
                               USING \"MODE\" \13"
Redefine function 10 " \"WRONG KEY\"  \13"

Form MODEFORM           ! Displays mode selection
 At 2,33 Put "-----------"
 At 3,33 Put "C O C O M O"
 At 4,33 Put "-----------"
 At 6,23 Put "TO SELECT A MODE DEPRESS ONE"
 At 8,26 Put "OF THE FOLLOWING F KEYS:"
 At 12,26 Put "F1 INTERMEDIATE ORGANIC"
 At 16,26 Put "F2 INTERMEDIATE SEMIDETACHED"
 At 20,26 Put "F3 INTERMEDIATE EMBEDDED"
 At 1,1 to 24,80 Put "FBBW"
 At 2,3 to 23,78 Put "FWBU"
Endform

Release perform "COCO"
Release perform "WRAPUP"
Release MODLFORM
Load perform "SSLODINT"

Putform MODEFORM; at 24,1    ! Display mode selection screen
```

```
/* SSLODINT.IPF - SS_LOAD_INTERMEDIATE MODULE                    */

/* This module loads an intermediate model spreadsheet for*/
/* the organic, semidetached, or embedded mode.                 */

/* Sample call: PERFORM "SSLODINT" USING "MODE"                  */

/* Inputs:  MODE = Organic, Semidetached or Embedded, as  */
/*                 user selected by F key                        */

/* Output:  COCOMO Intermediate spreadsheet, by mode       */

Redefine function 1 " PERFORM \"REDEVDAT\" \13"
Redefine function 2 " \"WRONG KEY\" \13"
Redefine function 3 " \"WRONG KEY\" \13"

Form SSLOADINT
  At 22,26 Put "LOADING INTERMEDIATE MODEL" WITH "B"
  At 22,25 to 22,53 Put "FOBU"
Endform;

Putform SSLOADINT; at 24,1

Release MODEFORM
Release perform "SETUPINT"

Load perform "REDEVDAT"

Let e.deci = 2      ! Sets spreadsheet decimal places

MODE = #A;

If MODE = 1 then                ! Organic spreadsheet
   Load "CIO" with "C";         ! "C" loads cell definitions
Else
   If MODE = 2 then             ! Semidetached spreadsheet
      Load "CIS" with "C";
   Else
      /* MODE = 3 */            ! Embedded spreadsheet
         Load "CIE" with "C";
   Endif;
Endif;
```

```
Calc= #A96                  ! Displayes effort cost driver page
Calc = #F100                ! Moves cursor to first cost driver
Calc                        ! Initiates the spreadsheet mode

Release SSLOADINT
Release MODE
```

```
/* REDEVDAT.IPF - READ_DEVELOPMENT_DATA MODULE                    */

/* This module reads cost driver inputs, and calculates           */
/* and validates the effort adjustment factor (EAF).              */

/* Sample call:  PERFORM "REDEVDAT"                               */

/* Input:  15 effort cost drivers                                 */

/* Output:  Effort Adjustment Factor (EAF)                        */


Form EAFCOMP
  At 23,33  Put "COMPUTING EAF" WITH "B"
  At 23,32 to 23,47 Put "FOBU"
Endform

Putform EAFCOMP; AT 24,1

Local DR1
Local DR2
Local DR3
Local DR4
Local DR5
Local DR6
Local DR7
Local DR8
Local DR9
Local DR10
Local DR11
Local DR12
Local DR13
Local DR14
Local DR15
Local MAXCOUNT
Local I
```

```
DR1     = #F100         ! RELY
DR2     = #F101         ! DATA
DR3     = #F102         ! CPLX
DR4     = #F103         ! TIME
DR5     = #F104         ! STOR
DR6     = #F105         ! VIRT
DR7     = #F106         ! TURN
DR8     = #F107         ! ACAP
DR9     = #F108         ! AEXP
DR10    = #F109         ! PCAP
DR11    = #F110         ! VEXP
DR12    = #F111         ! LEXP
DR13    = #F112         ! MODP
DR14    = #F113         ! TOOL
DR15    = #F114         ! SCED

Form BLANKOUT
  At 23,33 Put "                "
  At 23,32 to 23,47 Put "FUBU"
Endform;

Form EAFERR             ! EAF has a negative value
  At 23,22 to 23,57 Put "FWBR"
  At 23,23 Put "CAN'T USE NEGATIVE OR ZERO VALUES" WITH "B"
Endform

/* Compute EAF */
EAF = DR1 * DR2 * DR3 * DR4 * DR5 * DR6 * DR7 * DR8 * DR9 \
      * DR10 * DR11 * DR12 * DR13 * DR14 * DR15;

/* Validate cost driver input */
If EAF <= 0 then
   Putform EAFERR; at 24,1   ! Displays error message
   I = 0
   MAXCOUNT = 200
   While I < MAXCOUNT do     ! Delay to display error message
      I = I + 1
   Endwhile
   Calc = #F100    ! Cursor placed in first cost driver cell
   Calc            ! Redisplays cost driver page
```

```
Else
   Redefine function 1 " PERFORM \"DEVPARMS\" \13"
   Redefine function 2 " \"DEPRESS F1 FIRST\" \13"
   Redefine function 3 " \"DEPRESS F1 FIRST\" \13"
   Redefine function 10 " PERFORM \"KEYCHNQ\" \13"
   Load perform "DEVPARMS"
   #H11 = EAF     ! EAF displayed on KDSI input page
   Putform BLANKOUT; at 24,1
   Release EAFCOMP
   Release EAFERR
   Calc = #A1     ! Displays effort/schedule input page
   Calc = #D5     ! Places cursor into KDSI input cell
Endif;
```

```
/* DEVPARMS.IPF - DEV_PARAMETERS MODULE                              */


/* This module calculates effort and schedule criteria for*/
/* the intermediate COCOMO model and its' three modes:     */
/* organic, semidetached, and embedded.  The results of    */
/* these calculations are displayed to the user.
        */

/* Sample call:  PERFORM "DEVPARMS" (Invoked by F1)        */

/* Input:  KDSI - Estimated number of thousands of         */
/*                delivered source instructions            */
/*         PCOST - Personnel costs per man-month for       */
/*                 development                             */

/* Output: MAN-MONTH (Nominal) - Effort                    */
/*         MAN-MONTH (Adjusted)- Effort x EAF              */
/*         PRODUCTIVITY - Delivered source instructions/mm*/
/*         TDEV - Effort schedule in months                */
/*         FSP  - Full time software personnel             */
/*         ANNUAL COST - Development cost per year         */


Local I
Local J
Local MAXNUM
Local MODE

Let MAXNUM = 200    ! Number for error message delay counter

Form COMPFORM       ! Computing effort/schedule form
  At 12,27 Put "COMPUTING EFFORT/SCHEDULE" WITH "B"
  At 12,26 to 12,53 Put "FOBU"
Endform;

Putform COMPFORM; at 24,1

Form LARGKDSI
  At 12,28  Put "KDSI IS GREATER THEN 512" WITH "B"
  At 13,30 Put "KDSI >2 OR <512 ONLY"
  At 12,26 to 13,55 Put "FWBR"
Endform;
```

```
Form SMALKDSI
  At 12,30 Put "KDSI IS LESS THEN 2" WITH "B"
  At 13,30 Put "KDSI >2 OR <512 ONLY"
  At 12,26 to 13,55 Put "FWBR"
Endform;

ERRFLAG = 0
KDSI    = #D5          ! User input KDSI
PCOST   = #D6          ! User input cost/man-month
EAF     = #H11         ! Cost driver product value
MODE    = #B20         ! COCOMO Mode

If KDSI < 2 then
  Putform SMALKDSI; at 24,1   ! KDSI is too small
  Let I = 0
  While I < MAXNUM do         ! Error message delay
    I = I + 1
  Endwhile
  Calc = #D5                  ! Cursor back to KDSI input
  Calc                        ! Clears error message
  ERRFLAG = 1                 ! Don't perform calculations
Else
  If KDSI > 512 then
    Putform LARGKDSI; AT 24,1   ! KDSI is too large
    Let J = 0
    While J < MAXNUM do         ! Error message delay
      J = J + 1
    Endwhile
    Calc = #D5                  ! Cursor back to KDSI input
    Calc                        ! Clears error message
    ERRFLAG = 1                 ! Don't perform calculations
  Endif;
Endif;

/* Compute development parameters. */
If ERRFLAG = 0 then  ! KDSI is >2 and <512
   #C20 = "1"       ! Program performed effort calculations
   Load perform "DEVPHDIS"
   Load perform "MAINTINT"
   Load perform "KEYCHNQ"
   Redefine function 2 " PERFORM \"DEVPHDIS\" \13"
   Redefine function 3 " PERFORM \"MAINTINT\" \13"
   Redefine function 10 "PERFORM \"KEYCHNQ\" \13"
```

```
   If MODE = 1 then                ! Organic mode
      #D12 = 3.2 * (EXP (1.05 * LN (KDSI)))    !  MM(nom)
      #D11 = #D12 * EAF                        !  MM(adj)
      #D13 = (1000 * KDSI) / #D11              !  Productivity
      #D14 = 2.5 * (EXP (.38 * LN (#D11)))     !  TDEV
      #D15 = #D11 / #D14                       !  FSP
      #D16 = PCOST * #D11                      !  Annual cost
      Calc;               ! Displays updated spreadsheet
   Else
      If MODE = 2 then             ! Semidetached mode
         #D12 = 3.0 * (EXP (1.12 * LN (KDSI))) ! MM(nom)
         #D11 = #D12 * EAF                     ! MM(adj)
         #D13 = (1000 * KDSI) / #D11           ! Productivity
         #D14 = 2.5 * (EXP (.35 * LN (#D11)))  ! TDEV
         #D15 = #D11 / #D14                    ! FSP
         #D16 = PCOST * #D11                   ! Annual cost
         Calc;               ! Displays updated spreadsheet
      Else
                             ! Embedded mode
           #D12 = 2.8 * (EXP (1.20 * LN (KDSI)))! MM(nom)
           #D11 = #D12 * EAF                    ! MM(adj)
           #D13 = (1000 * KDSI) / #D11          ! Product.
           #D14 = 2.5 * (EXP (.32 * LN (#D11))) ! TDEV
           #D15 = #D11 / #D14                   ! FSP
           #D16 = PCOST * #D11                  ! Annual cost
           Calc;          ! Displays updated spreadsheet
      Endif;
    Endif;
Endif;

Release COMPFORM
Release LARGKDSI
Release SMALKDSI
Release ERRFLAG
Release EAF
Release KDSI
Release PCOST
```

```
/* KEYCHNQ.IPF - KEY_CHANGE MODULE                                    */

/* This module redefines F keys, displays a selection      */
/* screen on the spreadsheet, and dependent on the user    */
/* selection, either selects another program iteration or */
/* ends the program.                                                  */

/* Sample call: PERFORM "KEYCHNG"       (Invoked by F10)    */

/* Input:  F key selection by user                                    */

/* Output: Program end or program reiteration                         */

/* Submodule:  AGAINIT.IPF (AGAIN_IT MODULE)                          */

Let e.serr = true
Release perform "DEVPARMS"
Release perform "DEVPARBA"
Release perform "DEVPHDIS"
Release perform "GRAFPHE"
Release perform "GRAFPHS"
Release perform "DEVACDIS"
Release perform "GRAFADPD"
Release perform "GRAFADP"
Release perform "GRAFADIT"
Release perform "MAINTPAD"
Release perform "GRAfPDM"
Let e.serr = false
Load perform "AGAINIT"
Load perform "WRAPUP"

/* Function keys redefined for spreadsheet use */
Redefine function 1 " PERFORM \"AGAINIT\" \13"
Redefine function 10 " PERFORM \"WRAPUP\" \13"

Form FINIS
 At 5, 24 Put "BEFORE QUITTING . . . . . ."
 At 8, 13 Put "SELECT an option:"
 At 11, 19 Put "<F1> Another iteration/Save values/Reports"
 At 13, 19 Put "<F10> End program"
 At 1, 1 to 24, 80 Put "FBBW"
 At 2, 3 to 23, 78 Put "FWBU"
 At 13, 18 to 13, 37 Put "FRBU"
Endform;
Putform FINIS; at 24,1
```

```
/* MAINTBAS.IPF - MAINT_BASIC MODULE                                  */

/* This module begins the basic maintenance calculation by*/
/* displaying the maintenance effort and schedule page.   */

/* Sample call: PERFORM "MAINTBAS"  (Invoked by F3)       */

/* Input: None                                                        */

/* Output: Maintenance effort and schedule page                       */


Release perform "DEVPARBA"
Release perform "SSLODBAS"
Release perform "DEVPHDIS"
Release MODE

Load perform "MDATAPAD"

Redefine function 1 " PERFORM \"MDATAPAD\" \13"
Redefine function 2 " \"DEPRESS F1 FIRST\" \13"
Redefine function 3 " \"DEPRESS F1 FIRST\" \13"

MODSTAT = 0    ! Basic maintenance

Calc = #I1     ! Displays maintenance effort\schedule page

Calc = #M5     ! Places cursor in cost per man-month cell
```

```
/* MAINTINT.IPF - MAINT_INTERMEDIATE MODULE                    */

/* This module begins the intermediate maintenance             */
/* calculation by displaying the maintenance cost driver       */
/* table for inputs.                                           */

/* Sample call: PERFORM "MAINTINT"  (Invoked by F3)            */

/* Input: None                                                 */

/* Output: Maintenance cost driver table                       */


Release perform "DEVPARMS"
Release perform "SSLODINT"
Release perform "DEVPHDIS"
Release perform "REDEVDAT"
Release MODE

Load perform "CALCEAFM"

Redefine function 1 " PERFORM \"CALCEAFM\" \13"
Redefine function 2 " \"WRONG KEY\" \13"
Redefine function 3 " \"WRONG KEY\" \13"

Calc = #A115  ! Displays maintenance cost drivers

Calc = #F119  ! Places cursor onto first maint. cost driver
```

```
/* DEVPHDIS.IPF - DEV_PHASE_DISTR MODULE                          */


/* This is the control module for the calculation of the          */
/* effort and schedule distribution by phase.                     */

/* Sample call: PERFORM "DEVPHDIS"  (Invoked by F2)               */

/* Input: KDSI - estimated number of thousands of                 */
/*                delivered source instructions                   */
/*        MM - adjusted development man-months                    */
/*        TDEV - development schedule                             */

/* Output: Same as input                                          */
/*         Effort and schedule phase distributions                */

/* Submodule:   PHASEDIS.IPF (PHASE_DISTRE MODULE)                */


Let e.serr = true

Release perform "REDEVDAT"
Release perform "DEVPARMS"
Release perform "DEVPARBA"
Release perform "SSLODBAS"
Release perform "SSLODINT"
Release perform "MAINTBAS"
Release perform "MAINTINT"

Let e.serr = false

Redefine function 1 " PERFORM \"GRAFPHE\" \13"
Redefine function 2 " PERFORM \"GRAFPHS\" \13"
Redefine function 3 " PERFORM \"DEVACDIS\" \13"

Form COMPHASE
  At 16,22 Put "COMPUTING PHASE DISTRIBUTION" WITH "B"
  At 16,21 to 16,57 Put "FOBU"
Endform;

Calc = #A39  ! Displays spreadsheet phase distribution

Putform COMPHASE; AT 24,1;   ! Computing message
```

```
Load perform "PHASEDIS"
Perform "PHASEDIS";
Release perform "PHASEDIS"

#C20 = "2"     ! Program calculated activity distributions
#D21 = #D46    ! PD value for graphing
#D22 = #D48    ! DD value for graphing
#D23 = #D49    ! CUT value for graphing
#D24 = #D50    ! IT value for graphing

Load perform "DEVACDIS"
Load perform "GRAFPHE"
Load perform "GRAFPHS"

Release COMPHASE
Release KDSI
Release MM
Release TDEV
Release TABLEROW
Release STARTCOL
Release TEMPCOL
Release DISTTYPE

Calc           ! Displays spreadsheet with calculated values
```

```
/* PHASEDIS.IPF - PHASE_DISTR MODULE                                  */

/* This module controls the calculation of effort and                 */
/* schedule distribution by phase.                                    */

/* Sample call: PERFORM "PHASEDIS"                                    */

/* Input: KDSI - Estimated number of thousands of                     */
/*                delivered source instructions                       */
/*        MM - Adjusted development man-months                        */
/*        TDEV - Development schedule                                 */

/* Output: Effort phase distributions                                 */

/* Submodules:SELTABLE.IPF (SELECT_EFF/SCHED_TABLE MODULE)            */
/*            CALCEFSC.IPF (CALC_EFF_SCHED MODULE)                    */


Local MODEL
MODEL = #A20

If MODEL = 1 then      ! Basic model
  MM    = #D12
Else                     ! Intermediate model
  MM   = #D11
Endif;

KDSI = #D5;
TDEV = #D14;
DISTTYPE = 1;       ! Phase distribution.

Load perform "SELTABLE"
Perform "SELTABLE" using "DISTTYPE", "KDSI";
Release perform "SELTABLE"

Load perform "CALCEFSC"
Perform "CALCEFSC" using "MM", "TDEV";
Release perform "CALCEFSC"

Return;
```

```
/* SELTABLE.IPF - SELECT_TABLE MODULE                              */


/* Based on the type of distribution, this module selects         */
/* the top left cell of a table within the spreadsheet.           */
/* This top left cell is used as a starting point for             */
/* selection of phase percentages.                                */

/* Sample call:  PERFORM "SELTABLE" USING "#A", "#B"              */

/* Input: #A = DISTTYPE - type of distribution:                    */
/*              PHASE - phase distribution of effort and          */
/*                      schedule                                  */
/*              PRODDES - activity distribution of product        */
/*                        design                                  */
/*              PROGING - activity distribution of                */
/*                        programming                             */
/*              INTTEST - activity distribution of                */
/*                        integration and test                    */
/*              MAINT - activity distribution by phase for        */
/*                      maintenance                               */
/*        #B = KDSI - estimated # of thousands of                 */
/*                    delivered source instructions               */

/* Output: KDSI - Same as input                                   */
/*         TABLEROW - cell row # of table top row                 */
/*         STARTCOL - cell column # of selected tables'           */
/*                    left-most column                            */
/*         TEMPCOL - cell column number of the temporary         */
/*                   percentage location                          */
/*         Cell # and value of percentage                         */

/* Submodule: EVALKDSI.IPF (EVAL_KDSI MODULE)                     */


DISTTYPE = #A;
KDSI     = #B;

TABLEROW = 0;
STARTCOL = 0;
TEMPCOL  = 0;
```

```
/* Based on type of distribution, determines location of  */
/* top left cell of table within the spreadsheet and the  */
/* temporary location for percentages.                    */

Test DISTTYPE
   Case 1:                                ! Phase distribution
      TABLEROW = 46
      STARTCOL = 9                        ! Column I
      TEMPCOL = 24                        ! Column X
         Break;
   Case 2:                                ! Product design
      TABLEROW = 66
      STARTCOL = 9                        ! Column I
      TEMPCOL  = 24                       ! Column X
         Break;
   Case 3:                                ! Programming
      TABLEROW = 66
      STARTCOL = 14                       ! Column N
      TEMPCOL  = 25                       ! Column Y
         Break;
   Case 4:                                ! Integration & test
      TABLEROW = 66
      STARTCOL = 19                       ! Column S
      TEMPCOL  = 26                       ! Column Z
         Break;
   Otherwise:                             ! Maintenance
      TABLEROW = 84
      STARTCOL = 9                        ! Column I
      TEMPCOL  = 24                       ! Column X
Endtest

Load perform "EVALKDSI"
Perform "EVALKDSI" using "KDSI", "TABLEROW", "STARTCOL", \
                         "TEMPCOL";
Release perform "EVALKDSI"

Return;        ! Returns control to the PHASEDIS modules
```

```
/* EVALKDSI.IPF - EVAL_KDSI MODULE                                    */

/* This module evaluates KDSI for standard values (2,8,32,*/
/* 128, or 512).  If the KDSI > 2 and < 512, it is          */
/* determined to be nonstandard.                            */

/* Sample call: PERFORM "EVALKDSI" USING "#A", "#B", "#C",*/
/*                                       "#D";              */

/* Input: #A = KDSI - estimated # of thousands of           */
/*                    delivered source instructions.        */
/*        #B = TABLEROW - cell row # of the table top row */
/*        #C = STARTCOL - cell column # of the selected     */
/*                        table's left most column.         */
/*        #D = TEMPCOL (temporary storage) - cell column #*/
/*                     of temporary percentage location.   */

/* Output: Same as the above and the cell # and value of   */
/*         percentages.                                     */

/* Submodules: SLECTONE.IPF (SELECT_ONE MODULE)             */
/*             SELECTWO.IPF (SELECT_TWO MODULE)             */

 KDSI     = #A;
 TABLEROW = #B;
 STARTCOL = #C;
 TEMPCOL  = #D;

 If KDSI = 2 or KDSI=8 or KDSI = 32 or KDSI = 128 or \
    KDSI = 512 then      ! Standard KDSI
     Load perform "SLECTONE"
     Perform "SLECTONE" using "KDSI", "TABLEROW", \
                              "STARTCOL", "TEMPCOL";
     Release perform "SLECTONE"
 Endif;

If KDSI > 2 and KDSI < 512 and KDSI ne 8 and KDSI ne 32 \
   and KDSI ne 128 then      ! Non-standard KDSI
    Load perform "SELECTWO"
    Perform "SELECTWO" using "KDSI", "TABLEROW", \
                             "STARTCOL", "TEMPCOL":
    Release perform "SELECTWO"
Endif;

Return;
```

```
/* SLECTONE.IPF - SELECT_ONE MODULE                                    */

/* This module selects one column of percentages from a    */
/* table.                                                              */

/* Sample call: PERFORM "SLECTONE" USING "#A", "#B". "#C",*/
/*                                        "#D"                         */

/* Input: #A = KDSI - estimated # of thousands of          */
/*                    delivered source instructions                    */
/*        #B = TABLEROW - cell row # of table top row      */
/*        #C = STARTCOL - cell column # of selected        */
/*                        table's left-most column                     */
/*        #D = TEMPCOL - cell column # of temporary        */
/*                       percentage                                    */

/* Output: Cell # and value of the selected percentage.    */
/*         All selected percentages are stored in the      */
/*         same row as the original percentage and in the */
/*         following cell columns based on type of         */
/*         distribution selected:                                      */
/*         phase distribution      -   column 24 = X       */
/*         activity distribution:                                      */
/*            product design       -   column 24 = X       */
/*            programming          -   column 25 = Y       */
/*            integration & test   -   column 26 = Z       */
/*            maintenance          -   column 24 = X       */


KDSI        = #A;
TABLEROW    = #B;
STARTCOL    = #C;
TEMPCOL     = #D;

Local I;
Local PERCOL;
Local MAXAMT;
MAXAMT = 8;
```

```
If KDSI = 2 then
  PERCOL = STARTCOL                    ! 1st column of table
Else
  If KDSI = 8 then
    PERCOL = STARTCOL + 1              ! 2nd column of table
  Else
    If KDSI = 32 then
      PERCOL = STARTCOL + 2            ! 3rd column of table
    Else
      If KDSI = 128 then
        PERCOL = STARTCOL + 3          ! 4th column of table
      Else
        /* KDSI = 512 */
        PERCOL = STARTCOL + 4          ! 5th column of table
      Endif;
    Endif;
  Endif;
Endif;

I = 0;

/* Percentages moved from table to column */
While I < MAXAMT do
  #(TABLEROW,TEMPCOL) = #(TABLEROW,PERCOL)
  I = I + 1;
  TABLEROW = TABLEROW + 1;    ! Increments to the next row.
Endwhile;

Return;
```

```
/* SELECTWO.IPF - SELECT_TWO MODULE                                    */


/* Based on KDSI, this module selects two columns of                   */
/* percentages from a table and passes these percentages               */
/* to INTERPOL for interpolation.                                      */

/* Sample call: PERFORM "SELECTWO" USING "#A", "#B", "#C",             */
/*                                       "#D";                         */

/* Input:  #A = KDSI - estimated # of thousands of                     */
/*                     delivered source instructions                   */
/*         #B = TABLEROW - cell row # of table's top row               */
/*         #C = STARTCOL - cell column # of the selected               */
/*                         tables' left-most column                    */
/*         #D = TEMPCOL (temporary column) - cell column               */
/*                      number of the temporary percentage             */
/*                      location after interpolation                   */

/* Output: KDSI - same as input                                        */
/*         LOWKDSI - standard KDSI which is less than KDSI             */
/*                   entered                                           */
/*         HIGHKDSI - standard KDSI which is greater than              */
/*                    KDSI entered                                     */
/*         LOWPER - low percentage to be interpolated                  */
/*         HIGHPER - high percentage to be interpolated                */
/*                   HIGHKDSI column of the selected table             */
/*         TABLEROW - same as input                                    */
/*         TEMPCOL - same as input                                     */

/* Submodule:  INTERPOL.IPF (INTERPOLATION MODULE)                     */


 KDSI       = #A;
 TABLEROW   = #B;
 STARTCOL   = #C;
 TEMPCOL    = #D;

 Local I;       ! I is a local counter to this module
 Local MAXAMT; ! Maximum number of percentages in a column
 MAXAMT = 8;
```

```
/* Based on KDSI, select variables for passing to INTERPOL*/
 If KDSI > 2 and KDSI < 8 then
   LOWKDSI =     2
   HIGHKDSI =    8
   LOWPERCOL =   STARTCOL                ! 1st column of table
   HIGHPERCOL = STARTCOL + 1             ! 2nd column of table
 Else
   If KDSI > 8 and KDSI < 32 then
     LOWKDSI =     8
     HIGHKDSI =    32
     LOWPERCOL =   STARTCOL + 1          ! 2nd column of table
     HIGHPERCOL = STARTCOL + 2           ! 3rd column of table
   Else
     If KDSI > 32 and KDSI < 128 then
          LOWKDSI =     32
          HIGHKDSI =    128
          LOWPERCOL =   STARTCOL + 2     ! 3rd column of table
          HIGHPERCOL = STARTCOL + 3     ! 4th column of table
     Else
       /* KDSI > 128 and KDSI < 512 */
       LOWKDSI =     128
       HIGHKDSI =    512
       LOWPERCOL =   STARTCOL + 3      ! 4th column of table
       HIGHPERCOL = STARTCOL + 4      ! 5th column of table
     Endif;
   Endif;
 Endif;

/* Selects pairs of percentages from the adjacent columns */
/* of the table and calls INTERPOL.  The selections are   */
/* made from the top to the bottom row.  There are eight  */
/* rows per table.                                        */

Load perform "INTERPOL"

I = 0;
```

```
While I < MAXAMT do
  LOWPER  = #(TABLEROW,LOWPERCOL)
  HIGHPER = #(TABLEROW,HIGHPERCOL)
  PERFORM "INTERPOL" USING "KDSI", "LOWKDSI", "HIGHKDSI", \
                           "LOWPER","HIGHPER","TABLEROW", \
                           "TEMPCOL";
   I = I + 1;
   TABLEROW = TABLEROW + 1;  ! Increments to the next row
   Release LOWPER
   Release HIGHPER
Endwhile;

Release perform "INTERPOL"

Return;
```

```
/* INTERPOL.IPF - INTERPOLATION MODULE                              */

/* This module interpolates the two columns of percentages*/
/* selected from the percentage tables.                             */

/* Sample call: PERFORM "INTERPOL" USING "#A", "#B", "#C",*/
/*                                       #D", "#E", "#F", "#G" */

/* Input: #A = KDSI -Estimated # of thousands of                    */
/*                   delivered source instructions                  */
/*        #B = LOWKDSI - standard KDSI which is less than */
/*                      KDSI entered                                */
/*        #C = HIGHKDSI - standard KDSI which is greater  */
/*                        than KDSI entered.                        */
/*        #D = LOW% -  cell # and percentage from the      */
/*                   LOWKDSI column of the selected table */
/*        #E = HIGH% -  cell # and percentage from the     */
/*                   HIGHKDSI column of the selected table*/
/*        #F = TABLEROW - rows of low and high percentages*/
/*        #G = TEMPCOL (temporary column) - location in   */
/*                   the X column of the spreadsheet where*/
/*                   the interpolated percentages are     */
/*                   placed.                                        */

/* Output: Interpolated percentages                                 */

KDSI      = #A;
LOWKDSI   = #B;
HIGHKDSI  = #C;
LOWPER    = #D;
HIGHPER   = #E;
TABLEROW  = #F;
TEMPCOL   = #G;

/* Interpolation of low and high percentages                        */
TEMPCELL = HIGHPER+(((KDSI-LOWKDSI)/(HIGHKDSI-LOWKDSI))* \
           (LOWPER-HIGHPER));

/* Interpolated percentages assigned to X column in                 */
/* spreadsheet                                                      */

#(TABLEROW,TEMPCOL) = TEMPCELL;

Return;
```

```
/* CALCEFSC.IPF - CALC_EFF_SCHED MODULE                          */

/* This module calculates the phase distribution of effort*/
/* by multiplying MM by a phase distribution percentage   */
/* and the phase distribution of schedule by multiplying  */
/* TDEV by a phase distribution percentage. It also places*/
/* the calculated values in cells for display.            */

/* Sample call: PERFORM "CALCEFSC" USING "#A","#B";       */

/* Input: #A = MM - adjusted development man-month        */
/*        #B = TDEV - development schedule                */

/* Output: Effort and schedule phase distributions.       */


MM   = #A;
TDEV = #B;

Local I;              ! Counters
Local J;
Local DISPEROW        ! Row & column number displays
Local DISPSROW
Local DISPCOL
Local MAXEFFRT
Local MAXSCHED
Local TEMPEROW
Local TEMPSROW
Local TEMPCOL

DISPEROW = 46; ! Top row for display of effort distr.
DISPSROW = 53; ! Top row for display of schedule distr.
DISPCOL  =  4; ! Column for display of effort/schedule distr

MAXEFFRT = 5;  ! Maximum # of percentages for effort distr.
MAXSCHED = 3;  ! Maximum # of percentages for schedule distr

TEMPEROW = 46; ! Top temp storage row for effort percentage
TEMPSROW = 51; ! Top temp storage row for sched percentage
TEMPCOL  = 24; ! Temporary column; which is X.
```

```
/* Calculates effort distribution and displays results.    */
I = 0;

While I < MAXEFFRT do
  #(DISPEROW,DISPCOL) = #(TEMPEROW,TEMPCOL) * MM
  DISPEROW = DISPEROW + 1
  TEMPEROW = TEMPEROW + 1
  I = I + 1
Endwhile;


/* Calculates schedule distribution and displays results  */
J = 0;

While J < MAXSCHED do
  #(DISPSROW,DISPCOL) = #(TEMPSROW,TEMPCOL) * TDEV
  DISPSROW = DISPSROW + 1
  TEMPSROW = TEMPSROW + 1
  J = J + 1
Endwhile;

#H43 = #D5          ! Displays KDSI

Return;             ! Returns control to PHASEDIS module
```

```
/* GRAFPHE.IPF - GRAF_PHASE_EFFORT MODULE                          */

/* This module displays an instruction screen and a pie           */
/* chart for the phase distribution of effort.  It is             */
/* optionally called by the user via a function key after         */
/* computing phase distribution.  This module returns to          */
/* the spreadsheet at the phase distribution location.            */

/* Sample call:  PERFORM "GRAFPHE"     (Invoked by F1)            */

/* Input:  Effort phase distribution calculations                 */

/* Output: Pie chart displaying input values                      */


Let e.deci = 1

Clear

Form SHTFORM
 At 9, 20 Put " Press the  ENTER  key when you are"
 At 9, 31 Put "ENTER"
 At 9, 32 Put "ENTER"  WITH "R"
 At 11, 20 Put "   ready to continue and again when "
 At 13, 20 Put "      finished viewing the graph"
 At 6, 13 to 17, 64 Put "FABC"
Endform;

Putform SHTFORM;

Wait

#Title = "EFFORT (in Man-Months)"
Plot labeled % PIE from #C21 to #D24

Release SHTFORM

Let e.deci = 2

Calc

Return;
```

```
/* GRAFPHS.IPF - GRAF_PHASE_SCHEDULE MODULE                    */

/* This module displays an instruction screen and a pie       */
/* chart for the phase distribution of schedule.  It is       */
/* optionally called by the user via a function key after     */
/* computing phase distribution.  This module returns to      */
/* the spreadsheet at the phase location.                     */

/* Sample call: PERFORM "GRAFPHS"      (Invoked by F2)        */

/* Input:  Schedule phase distribution calculations           */

/* Output: Pie chart display of the input values              */


Let e.deci = 1

Clear

Form SHTFORM
 At 9, 20 Put " Press the  ENTER  key when you are"
 At 9, 31 Put "ENTER"
 At 9, 32 Put "ENTER"  WITH "R"
 At 11, 20 Put "   ready to continue and again when"
 At 13, 20 Put "      finished viewing the graph"
 At 6, 13 to 17, 64 Put "FABC"
Endform;

Putform SHTFORM;

Wait

#Title = "SCHEDULE (in months)"
Plot labeled % PIE from #C53 to #D55

Release SHTFORM

Let e.deci = 2

Calc

Return;
```

```
/* DEVACDIS.IPF - DEV_ACT_DISTR MODULE                              */

/* This is the control module for the calculation of the           */
/* activity distribution by phase.  It invokes DEV_PAD             */
/* and receives activity distribution computations.                */

/* Sample call: PERFORM "DEVACDIS"  (Invoked by F3)                */

/* Input: KDSI - estimated number of thousands of                  */
/*                delivered source instructions                    */
/*        MM - adjusted development man-months                     */

/* Output: Phase activity distributions                            */

/* Submodule:   DEVPAD.IPF (DEV_PAD MODULE)                        */

#C20 = "3"     ! Program at activity distribution

Let e.serr = true
Release perform "DEVPHDIS"
Release perform "GRAFPHE"
Release perform "GRAFPHS"
Let e.serr = false

Redefine function 1 " PERFORM \"GRAFADPD\" \13"
Redefine function 2 " PERFORM \"GRAFADP\" \13"
Redefine function 3 " PERFORM \"GRAFADIT\" \13"

Form COMPACT
  At 21,37 Put "COMPUTING ACTIVITY DISTRIBUTION" WITH "B"
  At 21,36 to 21,69 Put "FOBU"
Endform;

Calc = #A58  ! Displays activity distribution

Putform COMPACT; at 24,1;    ! Computing message

Load perform "DEVPAD"
Perform "DEVPAD";
Release perform "DEVPAD"
```

```
Calc

Release COMPACT

Load perform "GRAFADPD"
Load perform "GRAFADP"
Load perform "GRAFADIT"

Return;
```

```
/* DEVPAD.IPF - DEV_PAD MODULE                                        */

/* Based on the development activity distribution type,    */
/* this module selects a table which contains distribution*/
/* percentages by activity                                            */

/* Sample call: PERFORM "DEVPAD"                                      */

/* Input: KDSI - Estimated number of thousands of          */
/*               delivered source instructions             */
/*               Phase distributions of effort             */

/* Output: Same as input.                                             */
/*         DISTYPE1 - product design activity distribution*/
/*         DISTYPE2 - programming activity distribution    */
/*         DISTYPE3 - intergration and test activity       */
/*                    distribution                         */
/*         Development activity distribution               */
/*         Phase distribution of effort:                   */
/*                PRODEFFT - product design                */
/*                PROGEFFT - programming                   */
/*                INTEFFT - intergration and test          */

/* Submodules:                                                        */
/*            SELTABLE.IPF (SELECT_EFF/SCHED_TABLE MODULE)*/
/*            CALCDPAD.IPF (CALC_DEV_PAD MODULE)          */


KDSI     = #D5;

PRODEFFT = #D46;       ! Product design.
PROGEFFT = #D47;       ! Programming.
INTEFFT  = #D50;       ! Integration & testing.

DISTTYPE = 2;          ! Product design.

Load perform "SELTABLE"
Perform "SELTABLE" using "DISTTYPE", "KDSI";
Release perform "SELTABLE"

Load perform "CALCDPAD"
Perform "CALCDPAD" using "DISTTYPE", "PRODEFFT";
Release perform "CALCDPAD"
```

```
DISTTYPE = 3;          ! Programming.

Load perform "SELTABLE"
Perform "SELTABLE" using "DISTTYPE", "KDSI";
Release perform "SELTABLE"

Load perform "CALCDPAD"
Perform "CALCDPAD" using "DISTTYPE", "PROGEFFT";
Release perform "CALCDPAD"

DISTTYPE = 4;          ! Integration & testing.

Load perform "SELTABLE"
Perform "SELTABLE" using "DISTTYPE", "KDSI";
Release perform "SELTABLE"

Load perform "CALCDPAD"
Perform "CALCDPAD" using "DISTTYPE", "INTEFFT";
Release perform "CALCDPAD"

Return;   ! Returns control to DEVACDIS module.
```

```
/* GRAFADPD.IPF - GRAF_ACT_DIST_PD MODULE                          */

/* This module displays an instruction screen and a pie           */
/* chart for the activity distribution of product design.          */
/* It is optitionally called by the user via a function           */
/* key after computing activity distribution.                     */

/* Sample call:  PERFORM "GRAFADPD"     (Invoked by F1)           */

/* Input:  Activity distr. product design calculations            */

/* Output: Pie chart display of the input values                  */


Let e.deci = 1

Clear

Form SHTFORM
 At 9, 20 Put " Press the  ENTER  key when you are"
 At 9, 31 Put "ENTER"
 At 9, 32 Put "ENTER"  WITH "R"
 At 11, 20 Put "   ready to continue and again when"
 At 13, 20 Put "      finished viewing the graph"
 At 6, 13 to 17, 64 Put "FABC"
Endform;

Putform SHTFORM;

Wait

#Title = "ACTIVITY DISTRIBUTION for PRODUCT DESIGN"
Plot labeled % PIE from #C66 to #D73

Release SHTFORM

Let e.deci = 2

Calc

Return;
```

```
/* GRAFADP.IPF - GRAF_ACT_DIST_PHASE MODULE                    */

/* This module displays an intruction screen and a pie        */
/* chart for the activity distribution of programming.        */
/* It is optionally called by the user via a function key     */
/* after computing activity distribution.                     */

/* Sample call:  PERFORM "GRAFADP"   (Invoked by F2)          */

/* Input:  Activity distribution programming calculations     */

/* Output: Pie chart display of the input values.             */


LET E.DECI = 1

Clear

Form SHTFORM
 At 9, 20 Put " Press the  ENTER  key when you are"
 At 9, 31 Put "ENTER"
 At 9, 32 Put "ENTER"  WITH "R"
 At 11, 20 Put "   ready to continue and again when"
 At 13, 20 Put "      finished viewing the graph"
 At 6, 13 to 17, 64 Put "FABC"
Endform;

Putform SHTFORM;

Wait

#Title = "ACTIVITY DISTRIBUTION for PROGRAMMING"
Plot labeled % PIE from #E66 to #F73

Release SHTFORM

Let e.deci = 2

Calc

Return;
```

```
/* GRAFADIT.IPF - GRAF_ACT_DIST_INTEST MODULE                    */

/* This module displays an instruction screen and a pie         */
/* chart for the activity distribution of integration and       */
/* and testing.  It is optionally called by the user via a      */
/* function key after computing activity distribution.          */

/* Sample call: PERFORM "GRAFADIT"     (Invoked by F3)          */

/* Input:  Activity distr. integration/test calculations        */

/* Output: Pie chart display of the input values                */


Let e.deci = 1

Clear

Form SHTFORM
 At 9, 20 Put " Press the  ENTER  key when you are"
 At 9, 31 Put "ENTER"
 At 9, 32 Put "ENTER"  WITH "R"
 At 11, 20 Put "   ready to continue and again when"
 At 13, 20 Put "      finished viewing the graph"
 At 6, 13 to 17, 64 Put "FABC"
Endform;

Putform SHTFORM;

Wait

#Title = "ACTIVITY DISTRIBUTION for INTEGRATE/TEST"
Plot labeled % PIE from #G66 to #H73

Release SHTFORM

Let e.deci = 2

Calc

Return;
```

```
/* CALCDPAD.IPF - CALC_DEV_PAD MODULE                              */

/* This module calculates the phase activity distribution */
/* by multiplying phase distribution of effort by an      */
/* activity distribution percentage.                      */

/* Sample call: PERFORM "CALCDPAD" USING "#A", "#B";      */

/* Input: #A = DISTTYPE - type of activity distribution:  */
/*                PRODDES - product design                */
/*                PROGING - programming                   */
/*                INTTEST - integration & testing         */
/*         #B = DIST - phase distribution or effort       */

/* Output: Phase activity distributions                   */

DISTTYPE = #A;
DIST     = #B;

Local I;            ! Counter

DISPLROW = 66; ! Top row for display of activity distr.

MAXAMNT  = 8;  ! Max # of percentages for activity distr.

TEMPROW  = 66; ! Top temp storage row for activity percent

/* Based on activity distribution type, */
/* set-up columns in spreadsheet.*/

If DISTTYPE = 2 then                ! Product design
  DISPLCOL = 4                      ! Column D
  TEMPCOL  = 24                     ! Column X
Else
  If DISTTYPE = 3 then              ! Programming
    DISPLCOL = 6                    ! column F
    TEMPCOL  = 25                   ! column Y
  Else
    /* DISTTYPE = "INTTEST" */
    DISPLCOL = 8                    ! column H
    TEMPCOL = 26                    ! column Z
  Endif;
Endif;
```

```
/* Calc activity distr and place in cells for display */

I = 0;

While I < MAXAMNT do
  #(DISPLROW,DISPLCOL) = #(TEMPROW,TEMPCOL) * DIST;
  DISPLROW = DISPLROW + 1
  TEMPROW  = TEMPROW + 1
  I = I + 1
Endwhile;

#H62 = #D5          ! KDSI value displayed

Return;             ! Returns control to DEVACDIS module
```

```
/* CALCEAFM.IPF - CALC_EAF_MAINT MODULE                          */


/* This module reads in the maintenance cost drivers and         */
/* computes the effort adjustment factor (EAFM) for              */
/* maintenance.                                                  */

/* Sample call:  PERFORM "CALCEAFM"                              */

/* Input:  MCODR1 - 14: 14 maintenance cost drivers              */

/* Output: EAFM - maintenance effort adjustment factor           */


Form EAFMCOMP  ! EAFM calculation message
  At 23,28 Put "COMPUTING MAINTENANCE EAF" with "B"
  At 23,27 to 23,54 Put "FOBU"
Endform;

Putform EAFMCOMP; at 24,1

Form NEGVALCD
  At 23,23 Put "CAN'T USE NEGATIVE OR ZERO VALUES" WITH "B"
  At 23,22 to 23,59 Put "FWBR"
Endform;

Local MCODR1
Local MCODR2
Local MCODR3
Local MCODR4
Local MCODR5
Local MCODR6
Local MCODR7
Local MCODR8
Local MCODR9
Local MCODR10
Local MCODR11
Local MCODR12
Local MCODR13
Local MCODR14
Local EAFM1
Local EAFM2
Local I
Local MAXNUM
```

```
Let MAXNUM = 200

      /* Calculation of EAFM */

/* Read maintenance cost driver values */

MCODR1  = #F119;     ! RELY
MCODR2  = #F120;     ! DATA
MCODR3  = #F121;     ! CPLX
MCODR4  = #F122;     ! TIME
MCODR5  = #F123;     ! STOR
MCODR6  = #F124;     ! VIRT
MCODR7  = #F125;     ! TURN
MCODR8  = #F126;     ! ACAP
MCODR9  = #F127;     ! AEXP
MCODR10 = #F128;     ! PCAP
MCODR11 = #F129;     ! VEXP
MCODR12 = #F130;     ! LEXP
MCODR13 = #F131;     ! MODP
MCODR14 = #F132;     ! TOOL

EAFM1 = MCODR1*MCODR2*MCODR3*MCODR4*MCODR5*MCODR6*MCODR7;
EAFM2 = MCODR8*MCODR9*MCODR10*MCODR11*MCODR12*MCODR13;

EAFM = EAFM1 * EAFM2 * MCODR14;

/* Input validation */

If EAFM <= 0 then
  Putform NEGVALCD; at 24,1 ! Can't use neg values or zero
  Let I = 0
  While I < MAXNUM do  ! Error message delay
    I = I + 1
  Endwhile
  Calc = #F119     ! Cursor in first maint cost driver cell
  Calc             ! Redisplays maintenance cost drivers
```

```
Else
   Release perform "MAINTINT"
   Release EAFMCOMP
   Release NEGVALCD
   Redefine function 1 " PERFORM \"MDATAPAD\" \13"
   Redefine function 2 " \"DEPRESS F1 FIRST\" \13"
   Load perform "MDATAPAD"
   #012 = EAFM  ! Displays EAFM
   Calc = #I1;
   Calc = #M5;
Endif;

Release EAFM
```

```
/* MDATAPAD.IPF - MAINT_DATA_PAD MODULE                              */


/* This module controls the calculation and display of               */
/* nominal annual maintenance, full-time-equivalent                  */
/* software personnel for maintenance, maintenance cost              */
/* per man-month, and project activity  distribution by              */
/* phase for maintenance.                                            */

/* Sample call:  PERFORM "MDATAPAD"    (Invoked by F1)               */

/* Input:  KDSI - estimated thousands of delivered source            */
/*                instructions                                       */
/*         MMNOM - nominal effort                                    */

/* Output:  Display of calculated maintenance effort data            */

/* Submodule:  CALCMDAT.IPF  (CALC_MAINT_DATA MODULE)                */


Let e.serr = true
Release perform "MAINTBAS"
Release EAFM
Release perform "CALCEAFM"
Let e.serr = false

Local I
Local J
Local MAXNUM
Local ERRFLAG

Let MAXNUM = 200

Form MCOMP     ! Maint values computing
  At 12,30 Put "COMPUTING MAINTENANCE" WITH "B"
  At 12,29 to 12,51 Put "FOBU"
Endform;

Putform MCOMP; at 24,1

Form NEGVALMC  ! Neg or zero value error msg for maint cost
  At 12,23 Put "CAN'T USE NEGATIVE OR ZERO VALUES" WITH "B"
  At 12,22 to 12,57 Put "FWBR"
Endform;
```

```
Form MINVAL          ! ACT boundary error message
  At 12,29 Put "ACT RANGE 0 TO 1 ONLY" WITH "B"
  At 12,28 to 12,52 Put "FWBR"
Endform;

MCOST = #M5
ACT   = #M6
ERRFLAG = 0

If MCOST <= 0 then
  Putform NEGVALMC; at 24,1
  Let I = 0
  While I < MAXNUM do    ! Error message delay
    I = I + 1
  Endwhile
  Calc = #M5             ! Cursor placed into MCOST cell
  Calc                   ! Redisplays maint page
  ERRFLAG = 1            ! Don't perform calculations
Else.
  If ACT < 0 or ACT > 1 then
    Putform MINVAL; at 24,1
    Let J = 0
    While J <  MAXNUM do     ! Error message delay
      J = J + 1
    Endwhile
    Calc = #M6               ! Cursor placed into ACT cell
    Calc                     ! Redisplays maint page
    ERRFLAG = 1              ! Don't perform calculations
  Endif;
Endif;

If ERRFLAG = 0 then          ! Inputs validated
  Let e.serr = true
  Redefine function 2 " PERFORM \"MAINTPAD\" \13"
  Load perform "MAINTPAD"
  Load perform "CALCMDAT"
  Perform "CALCMDAT" using "MODSTAT" ! Calculates maint data
  Release perform "CALCMDAT"
  Let e.serr = false
Endif;
```

```
Calc            ! Redisplays maint effort page

Release ACT
Release MCOST
Release MCOMP
Release NEGVALMC
Release MINVAL
```

```
/* CALCMDAT.IPF - CALC_MAINT_DATA MODULE                          */

/* This module computes annual maintenance effort (MMNAM),*/
/* full-time-equivalent software personnel for maintenance*/
/* (FSPM), and annual maintenance cost (AMC).                     */

/* Sample call:  PERFORM "CALCMDAT" USING "#A"                    */

/* Input: #A = MODSTAT                                            */
/*              0 = Basic maintenance                             */
/*              1 = Intermediate maintenance                      */
/*         ACT - Annual change traffic                            */
/*         MPCOST - Maint personnel cost per man-month            */
/*         MM - Effort in man-months                              */
/*         MMNOM - Nominal effort in man-months                   */
/*         EAFM - Maintenance effort adjustment factor            */
/*                (for the intermediate model)                    */

/* Output: MMAM - Annual maint effort for basic model             */
/*          MMNAM - Nominal annual maintenance effort             */
/*          FSPM -  Average staffing level for maintenance        */
/*          AMC -   Annual maintenance cost                       */


MODSTAT = #A;

#C20 = "4"     ! Program calculated effort and maint values

/* Maintenance parameter calculations */

If MODSTAT = 0 then      ! Basic maintenance
   MPCOST = #M5;
   ACT    = #M6;
   MM     = #D12;
   MMAM   = ACT * MM;
   FSPM   = MMAM/12;
   AMC    = MPCOST * MMAM;
   #M12   = MMAM;
```

```
Else                          ! Intermediate maintenance
    EAFM   = #012;
    MPCOST = #M5;
    ACT    = #M6;
    MMNOM  = #D12;
    MMNAM  = ACT * MMNOM * EAFM;
    FSPM   = MMNAM/12;
    AMC    = MPCOST * MMNAM;
    #M12   = MMNAM;
Endif;

/* Display maintenance parameters */

#M13 = FSPM;
#M14 = AMC;

Let e.serr = true
Release MM
Release MMAM
Release MMNAM
Release MMNOM
Release FSPM
Release AMC
Let e.serr = false

Return;
```

```
/* MAINTPAD.IPF - MAINT_PAD MODULE                                    */


/* This module controls the percentage selection from the */
/* maintenance activity table as determined by mode and   */
/* KDSI.  It also calculates project activity distribution*/
/* (PAD) for the adjusted annual maintenance effort.      */

/* Sample call: PERFORM "MAINTPAD"     (Invoked by F2)    */

/* Input: KDSI - estimated number of thousands of         */
/*               delivered source instructions.           */
/*        MMNOM - nominal annual maintenance effort       */

/* Output: Maintenance activity distr. values displayed   */

/* Submodules:                                            */
/*            SELTABLE.IPF (SELECT_EFF/SCHED_TABLE MODULE)*/
/*            CALCMAPA.IPF (CALC_MAINT_PAD MODULE)        */


Redefine function 1 " PERFORM \"GRAFPDM\" \13"
Redefine function 2 " \"WRONG KEY\" \13"

Form MPHCOMP   ! Computing maintenance phase values
  At 20,21 Put "COMPUTING MAINTENANCE PHASE" WITH "B"
  At 20,20 to 20,49 Put "FOBU"
Endform;

Release perform "MDATAPAD"
Release MODSTAT

#C20 = "5"     ! Program calculated maintenance phase values
KDSI = #D5;
MMNOM = #D12;
DISTTYPE = 5            !  Maintenance

Calc = #A77

Putform MPHCOMP; at 24,1 ! Computing message displayed
```

```
Load perform "SELTABLE"
Perform "SELTABLE" using "DISTTYPE", "KDSI";
Release perform "SELTABLE"

Load perform "CALCMAPA"
Perform "CALCMAPA" using "MMNOM";
Release perform "CALCMAPA"

Load perform "GRAFPDM"

Release KDSI
Release MMNOM
Release DISTTYPE

Calc
```

```
/* CALCMAPA.IPF - CALC_MAINT_PAD MODULE                          */

/* This module computes adjusted annual maintenance effort*/
/* (MMnam), full-time-equivalent software personnel for    */
/* maintenance (FSPm), and annual maintenance cost         */
/* (MAINT COST).                                           */

/* Sample call: PERFORM "CALCMAPA" USING "#A"              */

/* Input: #A = MM -  nominal annual maintenance effort     */

/* Output: Maintenance activity distribution calculations */

MM = #A;

Local I;       ! Counter

DISPMROW = 84;! Top row for display of maint activity dist
DISPMCOL =  4;! Column for display of maint activity dist

MAXMAINT =  8;! Max # of percentages for maint activity dist

TEMPMROW = 84;! Top temp storage row for maint act percent
TEMPMCOL = 24;! Temporary X column

/* Calculate maintenance activity distribution */

I = 0;

While I < MAXMAINT do
  #(DISPMROW, DISPMCOL) = #(TEMPMROW, TEMPMCOL) * MMNOM
  DISPMROW = DISPMROW + 1
  TEMPMROW = TEMPMROW + 1
  I = I + 1
Endwhile;

Release MPHCOMP
Release DISPMROW
Release DISPMCOL
Release MAXMAINT
Release TEMPMROW
Release TEMPMCOL

Return;  ! Returns control back to MAINTPAD module.
```

```
/* GRAFPDM.IPF - GRAF_PD_MAINT MODULE                              */

/* This module displays an instruction screen and a pie           */
/* chart for the phase distribution of maintenance.  It is*/
/* optionally called by the user via a function key after */
/* computing activity distribution.                               */

/* Sample call:  PERFORM "GRAFPDM"     (Invoked by F1)            */

/* Input:  Calculated maintenance pad values                      */

/* Output: Pie chart display of maintenance pad values            */


Let e.deci = 1

Clear

Form SHTFORM
 At 9, 20 Put " Press the  ENTER  key when you are"
 At 9, 31 Put "ENTER"
 At 9, 32 Put "ENTER"  WITH "R"
 At 11, 20 Put "   ready to continue and again when"
 At 13, 20 Put "      finished viewing the graph"
 At 6, 13 to 17, 64 Put "FABC"
Endform;

Putform SHTFORM;

Wait

#Title = "PHASE DISTRIBUTION of MAINTENANCE"
Plot labeled % PIE from #C84 to #D91

Release SHTFORM

Let e.deci = 2

Calc

Return;
```

```
/* AGAINIT.IPF - AGAIN_IT MODULE                                    */


/* This module allows the user to perform another                   */
/* iteration, save prior computed values, or to erase               */
/* tables of other values saved and create a new table of           */
/* prior computed values.                                           */

/* Sample call: PERFORM "AGAINIT"      (Invoked by F1)              */

/* Input: F key selection by user                                   */

/* Output: One of the above selected options                        */

/* Submodules: SAVVAL.IPF  (SAVE_VALUES MODULE)                     */
/*             ERASTABL.IPF  (ERASE_TABLE MODULE)                   */
/*             ANOTHER.IPF  (ANOTHER MODULE)                        */


Release perform "KEYCHNQ"
Release FINIS

Load perform "SAVVAL"
Load perform "ERASTABL"
Load perform "ANOTHER"

Redefine function 1 " PERFORM \"SAVVAL\" \13"
Redefine function 2 " PERFORM \"ERASTABL\" \13"
Redefine function 3 " PERFORM \"ANOTHER\" \13"
Redefine function 10 " \"WRONG KEY\" \13"

Form CHOICE
 At 7, 20 Put "BEFORE PERFORMING ANOTHER ITERATION:"
 At 11, 20 Put "<F1> SAVE prior computed values"
 At 13, 20 Put "<F2> ERASE other computed values and"
 At 14, 20 Put "     START a new table"
 At 16, 20 Put "<F3> Perform another iteration WITHOUT"
 At 17, 20 Put "     saving prior computed values."
 At 1, 1 to 24, 80 Put "FBBW"
 At 2, 3 to 23, 78 Put "FWBU"
Endform;

Putform CHOICE; at 24,1
```

```
/* WRAPUP.IPF - WRAP_UP MODULE                                        */


/* This module permits the user to either save calculated */
/* values or to erase all calculated values before ending */
/* the program.                                                       */

/* Sample call: PERFORM "WRAPUP"      (Invoked by F10)     */

/* Input: F key selection by user                                     */

/* Output: <F1> Save values and end program                           */
/*         <F2> Erase values and end program                          */
/*         <F3> Continue program                                      */

/* Submodules: COCO.IPF  (COCO MODULE)                                */
/*             KILLIT.IPF  (KILL_IT MODULE)                           */

Form LASTFRM
 At 9, 20 Put "BEFORE QUITTING ..."
 At 11, 20 Put "<F1> End program"
 At 13, 20 Put "<F2> Erase calculated values and end program"
 At 15, 20 Put "<F3> Continue program"
 At 1, 1 to 24, 80 Put "FWBR"
 At 2, 2 to 23, 79 Put "FRBW"
Endform

Load perform "KILLIT"
Load perform "COCO"

Redefine function 1 " BYE \13"
Redefine function 2 " PERFORM \"KILLIT\" \13"
Redefine function 3 " PERFORM \"COCO\" \13"
Redefine function 10 " \"WRONG KEY\" \13"

Putform LASTFRM; at 24,1
```

```
/* KILLIT.IPF - KILL_IT MODULE                                    */

/* This module erases all values from both the  basic and */
/* intermediate tables and terminates the program.        */

/* Sample call: PERFORM "KILLIT"       (Invoked by F2)    */

/* Input: F key selection by user                         */

/* Output: Basic and Intermediate model table values      */
/*         erased and program terminated.                 */


Release perform "WRAPUP"
Release perform "COCO"

Form ERASALL
 At 21, 22 Put "Erasing All Tables" with "B"
 At 20, 1 to 22, 80 Put "FRBW"
Endform

Putform ERASALL; at 24,1

Use BES              ! Erase all Basic table values
Mark all
Compress BES
Finish BES

Use BESP
Mark all
Compress BESP
Finish BESP

Use BESPAD
Mark all
Compress BESPAD
Finish BESPAD

Use BESM
Mark all
Compress BESM
Finish BESM
```

```
Use BESMAD
Mark all
Compress BESMAD
Finish BESMAD

Use IES                ! Erase all Intermediate table values
Mark all
Compress IES
Finish IES

Use IESP
Mark all
Compress IESP
Finish IESP

Use IESPAD
Mark all
Compress IESPAD
Finish IESPAD

Use IESM
Mark all
Compress IESM
Finish IESM

Use IESMAD
Mark all
Compress IESMAD
Finish IESMAD

Clear

Bye
```

```
/* SAVVAL.IPF - SAVE_VALUES MODULE                                    */


/* This module determines whether basic or intermediate   */
/* values are to be saved.  The decision is based on which*/
/* model is selected at the program beginning by the user.*/

/* Sample call: PERFORM "SAVVAL"                                      */

/* Input: Basic or intermediate model selection                       */

/* Output: Basic or intermediate values saved                         */

/* Submodules: BASSAV.IPF (BASIC_SAVE MODULE)                         */
/*               INTSAV.IPF (INTERMEDIATE_SAVE MODULE)                */


Form WAITBAS
  At 20,30 Put "Saving BASIC Values" with "b"
  At 21,30 Put "     Please Wait"
  At 19,1 to 22,80 Put "fubw"
Endform

Form WAITINT
  At 20,27 Put "Saving INTERMEDIATE Values" with "B"
  At 21,27 Put "         Please Wait"
  At 19,1 to 22,80 Put "fubw"
Endform

Form NOVAL
  At 20,28 Put "No values were computed" with "B"
  At 21,28 Put "   Select <F3> only"
  At 19,1 to 22,80 Put "fwbr"
Endform

Local PROGSTAT
Local MODEL
Local FLAG

FLAG = 0
MODEL = #A20      ! Model # pucked from spreadsheet cell #A20
PROGSTAT = #C20   ! Point from which exited program
```

```
If PROGSTAT = 0 then  ! No values computed prior to quiting
   Putform NOVAL; at 24,1
   FLAG = 1
Endif

If MODEL = 1 and FLAG = 0 then
   Putform WAITBAS; at 24,1
   Let e.serr = true
   Release perform "ERASETABL"
   Release perform "AGAINIT"
   Release CHOICE
   Release NOVAL
   Let e.serr = false
   Load perform "BASSAV"      ! Basic model values saved
   Perform "BASSAV"
Else
   If MODEL = 2 and FLAG = 0 then
      Putform WAITINT; at 24,1
      Let e.serr = true
      Release perform "ERASTABL"
      Release perform "AGAINIT"
      Release CHOICE
      Release NOVAL
      Let e.serr = false
      Load perform "INTSAV"
      Perform "INTSAV"
   Endif;
Endif
```

```
/* ERASTABL.IPF - ERASE_TABLE MODULE                                  */


/* This module erases values from all the basic or                    */
/* intermediate tables depending upon which model the user            */
/* is currently using.                                                */

/* Sample call: PERFORM "ERASTABL"      (Invoked by F2)               */

/* Input: MODEL number from cell #A20 in the current                  */
/*        spreadsheet                                                 */
/*        1 - Basic model,  2 - Intermediate model                    */

/* Output: Basic or Intermediate model table values erased            */
/*         and new table values from prior calculation                */
/*         saved                                                      */

/* Submodule:  SAVVAL.IPF  (SAVE_VALUES MODULES)                      */


Form WAITBERA
 At 20, 27 Put "Erasing Basic Table Values" with "B"
 At 19, 1 to 21, 80 Put "FRBW"
Endform

Form WAITIERA
 At 20, 23 Put "Erasing Intermediate Table Values" with "B"
 At 19, 1 to 21, 80 Put "FRBW"
Endform

Local MODEL

MODEL = #B20

If MODEL = 1 then  ! Basic table values erased

   Putform WAITBERA; at 24,1

   Use BES
   Mark all
   Compress BES
   Finish BES
```

```
    Use BESP
    Mark all
    Compress BESP
    Finish BESP

    Use BESPAD
    Mark all
    Compress BESPAD
    Finish BESPAD

    Use BESM
    Mark all
    Compress BESM
    Finish BESM

    Use BESMAD
    Mark all
    Compress BESMAD
    Finish BESMAD

Else                    ! Intermediate table values erased

     Putform WAITIERA; at 24,1

    Use IES
    Mark all
    Compress IES
    Finish IES

    Use IESP
    Mark all
    Compress IESP
    Finish IESP

    Use IESPAD
    Mark all
    Compress IESPAD
    Finish IESPAD

    Use IESM
    Mark all
    Compress IESM
    Finish IESM
```

```
    Use IESMAD
    Mark all
    Compress IESMAD
    Finish IESMAD

Endif

Release WAITBERA
Release WAITIERA

Perform "SAVVAL"
```

```
/* ANOTHER.IPF - ANOTHER MODULE                                    */

/* This module loads coco.ipf which redefines function            */
/* keys and displays the model selection form so that             */
/* another iteration can be performed.                            */

/* Sample call: PERFORM "ANOTHER"                                 */

/* Input: F key selection on various menus to perform             */
/*        another iteration.                                      */

/* Output: COCOMO model selection options                         */

/* Submodule:  COCO.IPF  (COCO MODULE)                            */


Let e.serr = true

Release CHOICE
Release perform "BASSAV"
Release perform "RPTOUT"
Release perform "INTSAV"
Release perform "BRPTONE"
Release perform "BRPTALL"
Release perform "IRPTONE"
Release perform "IRPTALL"
Release perform "AGAINIT"
Release perform "SAVVAL"
Release perform "ERASTABL"
Release perform "KEYCHNQ"

Let e.serr = false

Load perform "coco"

Perform "coco"

Wait

Stop
```

```
/* BASSAV.IPF - BASIC_SAVE MODULE                                    */

/* This module saves Basic COCOMO values.                            */

/* Sample call: PERFORM "BASSAV"                                     */

/* Input: Effort, phase and activity distributions, and              */
/*        maintenance and maintenance phase distributions            */
/*        depending on where user exited from the                    */
/*        computation program.                                       */

/* Output: Input values placed into one of five tables.              */

/* Submodule:  RPTOUT.IPF  (REPORT_OUT MODULE)                       */


Local PROGSTAT

PROGSTAT = #C20     ! Indicates where user quit program

! Set environment variables
Let e.supd = true
Let e.stat = false
Let e.lmod = false
Let e.deci = 2

Test PROGSTAT

   Case "1":  ! Effort computations saved
      Use BES
      Attach 1
      MODELMOD = #D20; KDSI = #D5; PCOST = #D6; MM = #D12;
      PRODUCT = #D13; SCHEDULE = #D14; FSP = #D15;
      ACOST = #D16;
      Finish BES
      Break;
```

```
Case "2":  ! Effort & phase computations saved
   Use BESP
   Attach 1
   MODELMOD = #D20; KDSI = #D5; PCOST = #D6; MM = #D12;
   PRODUCT = #D13; SCHEDULE = #D14; FSP = #D15;
   ACOST = #D16;
   PRODES = #D46; PROG = #D47; DETDES = #D48; CUT = #D49;
   IT = #D50; SCHEDPD = #D53; SPROG = #D54; SIT = #D55;
   Finish BESP
   Break;

Case "3":  ! Effort, phase & activity computations saved
   Use BESPAD
   Attach 1
   MODELMOD = #D20; KDSI = #D5; PCOST = #D6; MM = #D12;
   PRODUCT = #D13; SCHEDULE = #D14; FSP = #D15;
   ACOST = #D16;
   PRODES = #D46; PROG = #D47; DETDES = #D48; CUT = #D49;
   IT = #D50; SCHEDPD = #D53; SPROG = #D54; SIT = #D55;
   RAPD = #D66; PDPD = #D67; PROGPD = #D68; TESTPD = #D69;
   VVPD = #D70; POPD = #D71; CQPD = #D72; MANPD = #D73;
   RAPROG = #F66; PDPROG = #F67; PROGPROG = #F68;
   TESTPROG = #F69;
   VVPROG = #F70; POPROG = #F71; CQPROG = #F72;
   MANPROG = #F73;
   RAIT = #H66; PDIT = #H67; PROGIT = #H68; TESTIT = #H69;
   VVIT = #H70; POIT = #H71; CQIT = #H72; MANIT = #H73;
   Finish BESPAD
   Break;

Case "4":  ! Effort & maintenance computations saved
   Use BESM
   Attach 1
   MODELMOD = #D20; KDSI = #D5; PCOST = #D6; MM = #D12;
   PRODUCT = #D13; SCHEDULE = #D14; FSP = #D15;
   ACOST = #D16;
   MPCOST = #M5; ACT = #M6; MMAM = #M12; FSPM = #M13;
   ACM = #M14;
   Finish BESM
   Break;
```

```
   Otherwise:  ! Effort, maint & maint phase values saved
      Use BESMAD
      Attach 1
      MODELMOD = #D20; KDSI = #D5; PCOST = #D6; MM = #D12;
      PRODUCT = #D13; SCHEDULE = #D14; FSP = #D15;
      ACOST = #D16;
      MPCOST = #M5; ACT = #M6; MMAM = #M12; FSPM = #M13;
      ACM = #M14;
      MRA = #D84; MPD = #D85; MPROG = #D86; MTEST = #D87;
      MVV = #D88;
      MPO = #D89; MCQ = #D90; MMAN = #D91;
      Finish BESMAD
      Break;
Endtest

! Reset environmental variables
Let e.supd = false
Let e.stat = true
Let e.lmod = true

Redefine function 1 " SEL = 1;PERFORM \"RPTOUT\" USING
                              \"SEL\" \13"
Redefine function 2 " SEL = 2;PERFORM \"RPTOUT\" USING
                              \"SEL\" \13"
Redefine function 3 " PERFORM \"ANOTHER\" \13"
Redefine function 10 " PERFORM \"WRAPUP\" \13"
Load perform "RPTOUT"

Form RPTCON
 At 9,26 Put "BASIC values have been saved"
 At 13, 22 Put "<F1> Display LAST computed BASIC values"
 At 15, 22 Put "<F2> Display ALL computed BASIC values"
 At 17, 22 Put "<F3> Continue Program"
 At 19, 22 Put "<F10> End Program"
 At 1, 1 to 24, 80 Put "FBBW"
 At 2, 3 to 23, 78 Put "FWBU"
 At 19, 21 to 19, 40 Put "FRBU"
Endform

Putform RPTCON; at 24,1
Release WAITBAS
Release WAITINT
Release perform "SAVVAL"
Return;
```

```
/* INTSAV.IPF - INTERMEDIATE_SAVE MODULE                          */

/* This module saves intermediate COCOMO values.                  */

/* Sample call: PERFORM "INTSAV"                                  */

/* Input: Effort, phase and activity distributions, and           */
/*        maintenance and maintenance phase distributions         */
/*        depending on where user exited from the                 */
/*        computation program.                                    */

/* Output: Input values placed into one of five tables.           */

/* Submodule:  RPTOUT.IPF  (REPORT_OUT MODULE)                    */


Local PROGSTAT

PROGSTAT = #C20     ! Indicates where user quit program

! Set environment variables
Let e.supd = true
Let e.stat = false
Let e.lmod = false
Let e.deci = 2

Test PROGSTAT

   Case "1":  ! Effort computations saved
      Use IES
      Attach 1
      MODELMOD = #D20; ERELY = #F100; EDATA = #F101;
      ECPLX = #F102; ETIME = #F103; ESTOR = #F104;
      EVIRT = #F105; ETURN = #F106; EACAP = #F107;
      EAEXP = #F108; EPCAP = #F109; EVEXP = #F110;
      ELEXP = #F111; EMODP = #F112; ETOOL = #F113;
      ESCED = #F114; EAF = #H11; KDSI = #D5; PCOST = #D6;
      MMADJ = #D11; MMNOM = #D12; PRODUCT =#D13;
      SCHEDULE = #D14; FSP = #D15; ACOST = #D16;
      Finish IES
      Break;
```

```
Case "2":  ! Effort & phase computations saved
   Use IESP
   Attach 1
   MODELMOD = #D20; ERELY = #F100; EDATA = #F101;
   ECPLX = #F102; ETIME = #F103; ESTOR = #F104;
   EVIRT = #F105; ETURN = #F106; EACAP = #F107;
   EAEXP = #F108; EPCAP = #F109; EVEXP = #F110;
   ELEXP = #F111; EMODP = #F112; ETOOL = #F113;
   ESCED = #F114; EAF = #H11; KDSI = #D5; PCOST = #D6;
   MMADJ = #D11; MMNOM = #D12; PRODUCT =#D13;
   SCHEDULE = #D14; FSP = #D15; ACOST = #D16;
   PRODES = #D46; PROG = #D47; DETDES = #D48; CUT = #D49;
   IT = #D50; SCHEDPD = #D53; SPROG = #D54; SIT = #D55;
   Finish IESP
   Break;

Case "3":  ! Effort, phase & activity computations saved
   Use IESPAD
   Attach 1
   MODELMOD = #D20; ERELY = #F100; EDATA = #F101;
   ECPLX = #F102; ETIME = #F103; ESTOR = #F104;
   EVIRT = #F105; ETURN = #F106; EACAP = #F107;
   EAEXP = #F108; EPCAP = #F109; EVEXP = #F110;
   ELEXP = #F111; EMODP = #F112; ETOOL = #F113;
   ESCED = #F114; EAF = #H11; KDSI = #D5; PCOST = #D6;
   MMADJ = #D11; MMNOM = #D12; PRODUCT =#D13;
   SCHEDULE = #D14; FSP = #D15; ACOST = #D16;
   PRODES = #D46; PROG = #D47; DETDES = #D48; CUT = #D49;
   IT = #D50; SCHEDPD = #D53; SPROG = #D54; SIT = #D55;
   RAPD = #D66; PDPD = #D67; PROGPD = #D68; TESTPD = #D69;
   VVPD = #D70; POPD = #D71; CQPD = #D72; MANPD = #D73;
   RAPROG = #F66; PDPROG = #F67; PROGPROG = #F68;
   TESTPROG = #F69; VVPROG = #F70; POPROG = #F71;
   CQPROG = #F72; MANPROG = #F73; RAIT = #H66;
   PDIT = #H67; PROGIT = #H68; TESTIT = #H69;
   VVIT = #H70; POIT = #H71; CQIT = #H72; MANIT = #H73;
   Finish IESPAD
   Break;
```

```
    Case "4":  ! Effort & maintenance computations saved
       Use IESM
       Attach 1
       MODELMOD = #D20; ERELY = #F100; EDATA = #F101;
       ECPLX = #F102; ETIME = #F103; ESTOR = #F104;
       EVIRT = #F105; ETURN = #F106; EACAP = #F107;
       EAEXP = #F108; EPCAP = #F109; EVEXP = #F110;
       ELEXP = #F111; EMODP = #F112; ETOOL = #F113;
       ESCED = #F114; EAF = #H11; KDSI = #D5; PCOST = #D6;
       MMADJ = #D11; MMNOM = #D12; PRODUCT =#D13;
       SCHEDULE = #D14; FSP = #D15; ACOST = #D16;
       MRELY = #F119; MDATA = #F120; MCPLX = #F121;
       MTIME = #F122; MSTOR = #F123; MVIRT = #F124;
       MTURN = #F125; MACAP = #F126; MAEXP = #F127;
       MPCAP = #F128; MVEXP = #F129; MLEXP = #F130;
       MMODP = #F131; MTOOL = #F132; EAFM = #012;
       MPCOST = #M5; ACT = #M6; MMNAM = #M12;
       FSPM = #M13; ACM = #M14;
       Finish IESM
       Break;

    Otherwise:  ! Effort, maint & maint phase values saved
       Use IESMAD
       Attach 1
       MODELMOD = #D20; ERELY = #F100; EDATA = #F101;
       ECPLX = #F102; ETIME = #F103; ESTOR = #F104;
       EVIRT = #F105; ETURN = #F106; EACAP = #F107;
       EAEXP = #F108; EPCAP = #F109; EVEXP = #F110;
       ELEXP = #F111; EMODP = #F112; ETOOL = #F113;
       ESCED = #F114; EAF = #H11; KDSI = #D5; PCOST = #D6;
       MMADJ = #D11; MMNOM = #D12; PRODUCT =#D13;
       SCHEDULE = #D14; FSP = #D15; ACOST = #D16;
       MRELY = #F119; MDATA = #F120; MCPLX = #F121;
       MTIME = #F122; MSTOR = #F123; MVIRT = #F124;
       MTURN = #F125; MACAP = #F126; MAEXP = #F127;
       MPCAP = #F128; MVEXP = #F129; MLEXP = #F130;
       MMODP = #F131; MTOOL = #F132; EAFM = #012;
       MPCOST = #M5; ACT = #M6; MMNAM = #M12;
       FSPM = #M13; ACM = #M14;
       MRA = #D84; MPD = #D85; MPROG = #D86; MTEST = #D87;
       MVV = #D88; MPO = #D89; MCQ = #D90; MMAN = #D91;
       Finish IESMAD
       Break;
Endtest
```

```
! Reset environmental variables
Let e.supd = false
Let e.stat = true
Let e.lmod = true

Redefine function 1 " SEL = 3;PERFORM \"RPTOUT\" USING
                              \"SEL\" \13"
Redefine function 2 " SEL = 4;PERFORM \"RPTOUT\" USING
                              \"SEL\" \13"
Redefine function 3 " PERFORM \"ANOTHER\" \13"
Redefine function 10 " PERFORM \"WRAPUP\" \13"

Load perform "RPTOUT"

Form RPTCON
 At 9,17 Put "INTERMEDIATE values have been saved"
 At 13, 19 Put "<F1> Display LAST computed INTERMEDIATE values"
 At 15, 19 Put "<F2> Display ALL computed INTERMEDIATE values"
 At 17, 19 Put "<F3> Continue Program"
 At 19, 19 Put "<F10> End Program"
 At 1, 1 to 24, 80 Put "FBBW"
 At 2, 3 to 23, 78 Put "FWBU"
 At 19, 18 to 19, 37 Put "FRBU"
Endform

Putform RPTCON;at 24,1

Release WAITBAS
Release WAITINT
Release perform "SAVVAL"

Return;
```

```
/* RPTOUT.IPF - REPORT_OUT MODULE                                    */

/* This module selects the proper basic or intermediate   */
/* reprots to display the calculated basic or             */
/* intermediate values.                                   */

/* Sample call: PERFORM "RPTOUT" USING "#A"               */

/* Input: #A = SEL:                                       */
/*        1 - Display prior calculated basic values       */
/*        2 - Display all prior calculated basic values   */
/*        3 - Display prior calculated intermediate values*/
/*        4 - Display all prior calculated intermediate   */
/*            values                                      */

/* Output: One or all basic or intermediate values        */

/* Submodules:  BRPTONE.IPF  (BASIC_RPT_ONE MODULE)       */
/*              BRPTALL.IPF  (BASIC_RPT_ALL MODULE)       */
/*              IRPTONE.IPF  (INTERMEDIATE_RPT_ONE MODULE)*/
/*              IRPTALL.IPF  (INTERMEDIATE_RPT_ALL MODULE)*/


SEL = #A

Let e.serr = true
Release perform "BASSAV"
Release perform "INTSAV"
Let e.serr = false

Test SEL

  Case  1:        ! Loads prior basic report module
    Load perform "BRPTONE"
    Redefine function 1 " OPT = 1;PERFORM \"BRPTONE\" USING
                                  \"OPT\",\"PROGSTAT\" \13"
    Redefine function 2 " OPT = 2;PERFORM \"BRPTONE\" USING
                                  \"OPT\",\"PROGSTAT\" \13"
    Redefine function 10 " \"WRONG KEY\" \13"
    Break;
```

```
  Case  2:        ! Loads all prior basic report module
    Load perform "BRPTALL"
    Redefine function 1 " OPT = 1;PERFORM \"BRPTALL\" USING
                                  \"OPT\" \13"
    Redefine function 2 " OPT = 2;PERFORM \"BRPTALL\" USING
                                  \"OPT\" \13"
    Redefine function 10 " \"WRONG KEY\" \13"
    Break;

  Case  3:        ! Loads prior intermediate report module
    Load perform "IRPTONE"
    Redefine function 1 " OPT = 1;PERFORM \"IRPTONE\" USING
                                  \"OPT\",\"PROGSTAT\" \13"
    Redefine function 2 " OPT = 2;PERFORM \"IRPTONE\" USING
                                  \"OPT\",\PROGSTAT\" \13"
    Redefine function 10 " \"WRONG KEY\" \13"
    Break;

  Otherwise:    ! Loads all prior intermediate report module
    Load perform "IRPTALL"
    Redefine function 1 " OPT = 1;PERFORM \"IRPTALL\" USING
                                  \"OPT\" \13"
    Redefine function 2 " OPT = 2;PERFORM \"IRPTALL\" USING
                                  \"OPT\" \13"
    Redefine function 10 " \"WRONG KEY\" \13"
    Break;
Endtest;

PROGSTAT = #C20     ! Program status at point exited

Form SELECTOP
  At 9,23 Put "Select an option to display report"
  At 12, 23 Put "<F1> SCREEN output"
  At 14, 23 Put "<F2> PRINTER output"
  At 15, 23 Put "      (Turn on printer first)"
  At 1, 1 to 24, 80 Put "FBBO"
  At 2, 3 to 23, 78 Put "FOBU"
  At 15,22 to 15,52 Put "FRBU"
Endform

Putform SELECTOP; at 24,1
Wait
Stop
```

```
/* BRPTONE.IPF - BASIC_RPT_ONE MODULE                              */

/* This module displays prior computed COCOMO values on           */
/* either the screen or on a printer for the basic model.         */

/* Sample call: PERFORM "BRPTONE" USING "#A","#B"                  */

/* Input: #A = OPT:                                                */
/*            1 - Setup screen parameters                          */
/*            2 - Setup printer parameters                         */
/*         #B = PROGSTAT - Where calculations terminated           */

/* Output: Prior computed basic COCOMO values.                     */


OPT = #A
PROGSTAT = #B

Let e.stat = false
Let e.supd = true

Let e.serr = true
Release perform "RPTOUT"
Let e.serr = false

Form ROUT
 At 20, 27 Put  "OBTAIN REPORT FROM PRINTER"
 At 19,1 to 21,80 Put "FUBO"
Endform

If OPT = 1 then     ! Set screen parameters
  Let e.pdep = 24
Else                ! Set printer parameters
  Putform ROUT; at 24,1
  Let e.pdep = 60
  Let e.pmar = 7
  Let e.oprn = true
Endif
```

```
Test PROGSTAT

   Case "1":
     Use BES
     Obtain last record
     Report "BESRPT"
     Finish BES
     Break;

   Case "2":
     Use BESP
     Obtain last record
     Report "BESPRPT"
     Finish BESP
     Break;

   Case "3":
     Use BESPAD
     Obtain last record
     Report "BESPADRP"
     Finish BESPAD
     Break;

   Case "4":
     Use BESM
     Obtain last record
     Report "BESMRPT"
     Finish BESM
     Break;

   Otherwise:
     Use BESMAD
     Obtain last record
     Report "BESMADRP"
     Finish BESMAD
     Break;

Endtest

Wait
```

```
! Reset environmental variables
Let e.stat = true
Let e.supd = false
Let e.oprn = false

Form RPTDONE
  At 9,32 Put "REPORT COMPLETED"
  At 12,23 Put "<F1> SCREEN output"
  At 14,23 Put "<F2> PRINTER output"
  At 15,23 Put "      (Turn on printer)"
  At 17,23 Put "<F3> Continue program"
  At 19,23 Put "<F10> End Program"
  At 1,1 to 24,80 Put "FBBW"
  At 2,3 to 23,78 Put "FWBU"
  At 15,22 to 15,52 Put "FRBU"
  At 19,22 to 19,41 Put "FRBU"
Endform

Redefine function 1 " OPT =  1;Perform \"BRPTONE\" USING
                               \"OPT\",\"PROGSTAT\" \13"
Redefine function 2 " OPT =  2;Perform \"BRPTONE\" USING
                               \"OPT\",\"PROGSTAT\" \13"
Redefine function 3 " Perform \"ANOTHER\" \13"
Redefine function 10 " Perform \"WRAPUP\" \13"

Putform RPTDONE; at 24,1
```

```
/* BRPTALL.IPF - BASIC_RPT_ALL MODULE                              */


/* This module displays all prior computed basic COCOMO           */
/* values on either the screen or on a printer.                    */

/* Sample call: PERFORM "BRPTALL" USING "#A"                       */

/* Input: #A - OPT:                                                */
/*            1 - setup screen parameters                          */
/*            2 - setup printer parameters                         */

/* Output: Prior computed basic COCOMO values.                     */


OPT = #A

Let e.stat = false
Let e.supd = true

Let e.serr = true
Release perform "RPTOUT"
Let e.serr = false

Form ROUT
 At 20, 27 Put  "OBTAIN REPORT FROM PRINTER"
 At 19,1 to 21,80 Put "FUBO"
Endform

If OPT = 1 then     ! Set screen parameters
  Let e.pdep = 24
Else                ! Set printer parameters
  Putform ROUT; at 24,1
  Let e.pdep = 60
  Let e.pmar = 7
  Let e.oprn = true
Endif

Let e.serr = true
```

```
Use BES

If Currec(BES) = 0 then
  Finish BES
Else
  Obtain first record
  While #found do
    Report "BESRPT"
    If Eot(BES) then
      Finish BES
      Break
    Endif;
    Obtain next
  Endwhile;
  Wait;
Endif

Use BESP

If Currec(BESP) = 0 then
  Finish BESP
Else
  Obtain first record
  While #found do
    Report "BESPRPT"
    If Eot(BESP) then
      Finish BESP
      Break
    Endif;
    Obtain next
  Endwhile;
  Wait;
Endif

Use BESPAD
```

```
If Currec(BESPAD) = 0 then
  Finish BESPAD
Else
  Obtain first record
  While #found do
    Report "BESPADRP"
    If Eot(BESPAD) then
      Finish BESPAD
      Break
    Endif;
    Obtain next
  Endwhile;
  Wait;
Endif

Use BESM

If Currec(BESM) = 0 then
  Finish BESM
Else
  Obtain first record
  While #found do
    Report "BESMRPT"
    If Eot(BESM) then
      Finish BESM
      Break
    Endif;
    Obtain next
  Endwhile;
  Wait;
Endif

Use BESMAD
```

```
If Currec(BESMAD) = 0 then
  Finish BESMAD
Else
  Obtain first record
  While #found do
    Report "BESMADRP"
    If Eot(BESMAD) then
      Finish BESMAD
      Break
    Endif;
    Obtain next
  Endwhile;
  Wait;
Endif

Let e.serr = false

! Reset environmental variables
Let e.stat = true
Let e.supd = false
Let e.oprn = false

Form RPTDONE
  At 9,32 Put "REPORT COMPLETED"
  At 12,23 Put "<F1> SCREEN output"
  At 14,23 Put "<F2> PRINTER output"
  At 15,23 Put "      (Turn on printer)"
  At 17,23 Put "<F3> Continue program"
  At 19,23 Put "<F10> End Program"
  At 1,1 to 24,80 Put "FBBW"
  At 2,3 to 23,78 Put "FWBU"
  At 15,22 to 15,52 Put "FRBU"
  At 19,22 to 19,41 Put "FRBU"
Endform

Redefine function 1 " OPT =  1;Perform \"BRPTALL\" USING
                               \"OPT\" \13"
Redefine function 2 " OPT =  2;Perform \"BRPTALL\" USING
                               \"OPT\" \13"
Redefine function 3 " Perform \"ANOTHER\" \13"
Redefine function 10 " Perform \"WRAPUP\" \13"

Putform RPTDONE; at 24,1
```

```
/* IRPTONE.IPF - INTERMEDIATE_RPT_ONE MODULE                    */


/* This module displays prior computed COCOMO values on        */
/* either the screen or on a printer for the                    */
/* intermediate model.                                          */

/* Sample call: PERFORM "IRPTONE" USING "#A","#B"               */

/* Input: #A = OPT:                                             */
/*            1 - setup screen parameters                       */
/*            2 - setup printer parameters                      */
/*         #B = PROGSTAT - Where user ended calculations        */

/* Output: Prior computed intermediate COCOMO values.           */


OPT = #A
PROGSTAT = #B

Let e.stat = false
Let e.supd = true

Let e.serr = true
Release perform "RPTOUT"
Let e.serr = false

Form ROUT
 At 20, 27 Put  "OBTAIN REPORT FROM PRINTER"
 At 19,1 to 21,80 Put "FUBO"
Endform

If OPT = 1 then     ! Set screen parameters
  Let e.pdep = 24
Else                ! Set printer parameters
  Putform ROUT; at 24,1
  Let e.pdep = 60
  Let e.pmar = 7
  Let e.oprn = true
Endif
```

```
Test PROGSTAT

   Case "1":
     Use IES
     Obtain last record
     Report "IESRPT"
     Finish IES
     Break;

   Case "2":
     Use IESP
     Obtain last record
     Report "IESPRPT"
     Finish IESP
     Break;

   Case "3":
     Use IESPAD
     Obtain last record
     Report "IESPADRP"
     Finish IESPAD
     Break;

   Case "4":
     Use IESM
     Obtain last record
     Report "IESMRPT"
     Finish IESM
     Break;

   Otherwise:
     Use IESMAD
     Obtain last record
     Report "IESMADRP"
     Finish IESMAD
     Break;

Endtest

Wait
```

```
! Reset environmental variables
Let e.stat = true
Let e.supd = false
Let e.oprn = false

Form RPTDONE
  At 9,32 Put "REPORT COMPLETED"
  At 12,23 Put "<F1> SCREEN output"
  At 14,23 Put "<F2> PRINTER output"
  At 15,23 Put "     (Turn on printer)"
  At 17,23 Put "<F3> Continue program"
  At 19,23 Put "<F10> End Program"
  At 1,1 to 24,80 Put "FBBW"
  At 2,3 to 23,78 Put "FWBU"
  At 15,22 to 15,52 Put "FRBU"
Endform

Redefine function 1 " OPT =  1;Perform \"IRPTONE\" USING
                               \"OPT\",\"PROGSTAT\" \13"
Redefine function 2 " OPT =  2;Perform \"IRPTONE\" USING
                               \"OPT\",\"PROGSTAT\" \13"
Redefine function 3 " Perform \"ANOTHER\" \13"
Redefine function 10 " Perform \"WRAPUP\" \13"

Putform RPTDONE; at 24,1
```

```
/* IRPTALL.IPF - INTERMEDIATE_RPT_ALL MODULE                    */

/* This module displays all prior computed intermediate        */
/* COCOMO values on either the screen or on a printer.          */

/* Sample call: PERFORM "IRPTALL" USING "#A"                    */

/* Input: #A - OPT:                                             */
/*           1 - Setup screen parameters                        */
/*           2 - Setup printer parameters                       */

/* Output: Prior computed intermediate COCOMO values.           */


OPT = #A

Let e.stat = false
Let e.supd = true

Let e.serr = true
Release perform "RPTOUT"
Let e.serr = false

Form ROUT
  At 20, 27 Put  "OBTAIN REPORT FROM PRINTER"
  At 19,1 to 21,80 Put "FUBO"
Endform

If OPT = 1 then     ! Set screen parameters
  Let e.pdep = 24
Else                ! Set printer parameters
  Putform ROUT; at 24,1
  Let e.pdep = 60
  Let e.pmar = 7
  Let e.oprn = true
Endif

Let e.serr = true

Use IES
```

```
If Currec(IES) = 0 then
  Finish IES
Else
  Obtain first record
  While #found do
    Report "IESRPT"
    If Eot(IES) then
      Finish IES
      Break
    Endif;
    Obtain next
  Endwhile;
  Wait;
Endif

Use IESP

If Currec(IESP) = 0 then
  Finish IESP
Else
  Obtain first record
  While #found do
    Report "IESPRPT"
    If Eot(IESP) then
      Finish IESP
      Break
    Endif;
    Obtain next
  Endwhile;
  Wait;
Endif

Use IESPAD
```

```
If Currec(IESPAD) = 0 then
  Finish IESPAD
Else
  Obtain first record
  While #found do
    Report "IESPADRP"
    If Eot(IESPAD) then
      Finish IESPAD
      Break
    Endif;
    Obtain next
  Endwhile;
  Wait;
Endif

Use IESM

If Currec(IESM) = 0 then
  Finish IESM
Else
  Obtain first record
  While #found do
    Report "IESMRPT"
    If Eot(IESM) then
      Finish IESM
      Break
    Endif;
    Obtain next
  Endwhile;
  Wait;
Endif

Use IESMAD
```

```
If Currec(IESMAD) = 0 then
  Finish IESMAD
Else
  Obtain first record
  While #found do
    Report "IESMADRP"
    If Eot(IESMAD) then
      Finish IESMAD
      Break
    Endif;
    Obtain next
  Endwhile;
  Wait;
Endif

Let e.serr = false

! Reset environmental variables
Let e.stat = true
Let e.supd = false
Let e.oprn = false

Form RPTDONE
  At 9,32 Put "REPORT COMPLETED"
  At 12,23 Put "<F1> SCREEN output"
  At 14,23 Put "<F2> PRINTER output"
  At 15,23 Put "      (Turn on printer)"
  At 17,23 Put "<F3> Continue program"
  At 19,23 Put "<F10> End Program"
  At 1,1 to 24,80 Put "FBBW"
  At 2,3 to 23,78 Put "FWBU"
  At 15,22 to 15,52 Put "FRBU"
  At 19,22 to 19,41 Put "FRBU"
Endform

Redefine function 1 " OPT =  1;Perform \"IRPTALL\" USING
                               \"OPT\" \13"
Redefine function 2 " OPT =  2;Perform \"IRPTALL\" USING
                               \"OPT\" \13"
Redefine function 3 " Perform \"ANOTHER\" \13"
Redefine function 10 " Perform \"WRAPUP\" \13"

Putform RPTDONE; at 24,1
```

APPENDIX E

COCOMO TOOL USER'S MANUAL

COCOMO Tool

Constructive Cost Model Tool

Version 1.0

User's Manual

December 1985

Naval Postgraduate School
Monterey, California

The documentation contained herein pertains to Version 1.0 of COCOMO Tool (Constructive Cost Model Tool) as implemented on the IBM PC computer systems at the Naval Postgraduate School. While it is believed that the contents are completely accurate, neither the school nor the authors assume any liability resulting from inaccuracies herein or from the use of this documentation or the use of COCOMO.

TABLE OF CONTENTS

APPENDICES

Chapter 1

Introduction

1.1 General Information

The COCOMO (Constructive Cost Model) Tool is an interactive, decision support system used to apply a software cost estimation technique. This tool is based on the Basic and Intermediate COCOMO model developed by Barry W. Boehm at TRW and explained in detail in his text, "Software Engineering Economics" (Prentice-Hall, 1981). It calculates estimates of man-months of effort, cost, and schedule required for a software project. These estimates are based on the project size expressed in estimated number of thousands of lines of deliverable source instructions (KDSI) entered by the user.

This manual explains how to use the program. Chapter 2 provides a brief description of the Basic and Intermediate COCOMO models as well as COCOMO Maintenance used by both models. The use of the COCOMO Tool is introduced in Chapter 3. Both narrative and screen descriptions demonstrate COCOMO Basic, Intermediate and Maintenance program utilization. Chapter 4 descibes how to obtain a report of the computed values on either the screen or printer. Sample reports are also displayed in this chapter so the user can get an idea of what to expect in the report format.

While this document includes a complete description of the mechanics of using the COCOMO Tool, it does not attempt to provide the background and understanding of the underlying COCOMO model necessary to use it wisely. It is recommended that COCOMO Tool users familiarize themselves with Boehm's book (Chapters 4-9) since the details of the models assumptions, limitations, and accuracy are not reproduced here.

The COCOMO Tool is written in software using the KnowledgeMan application package developed by Micro Data Base Systems, Inc. and is operational on IBM PC systems. It is screen-oriented and menu-driven, and requires the use of a CRT terminal and a hard disk drive.

1.2 COCOMO Tool Characteristics Remarks

The program is memu driven and selections are made with the use of function keys. It is divided into four major functional areas: Model/mode selection, development branch,

maintenance branch, and program termination. In model/mode selection the user enters the program and selects either the Basic or Intermediate model. Next, one mode is choosen from either the organic, semidetached, or embedded modes. Once a mode is selected the user is automatically moved into the development branch. Within this branch, effort and schedule parameters are first computed. From this point a decision must be made to either stay in the development branch or move into the maintenance branch. Once a choice is made it is final. The only way an unchoosen branch can be entered is on another iteration of the model and mode. The development branch will give the options of having phase and activity distributions computed. The maintenance branch involves calculation of maintenance parameters along with the option to also have maintenance phase distributions computed. The program termination function not only allows the program to end, it also provides the option of saving prior computed values from the development or maintenance branches. Another option in this area also permits output of the saved data on either the screen or printer.

Program error messages will be generated if an incorrect function key is depressed or an out-of-range value is inserted for computation. Error messages and appropriate actions are listed alphabetically in Appendix C. Note that KDSI values below 2 KDSI and above 512 KDSI will cause an error message as these are the low and high boundaries of the COCOMO model.

Familiarity with Boehm's text, "Software Engineering Economics" (Chapters 4 - 9), is strongly recommended to gain an understanding of the COCOMO model assumptions, limitations, and accuracy. Use of this program without knowledge of the COCOMO model will limit the full understanding that can be gained from the computed results.

Due to program size and complexity, computations for standard KDSI values (2, 8, 32, 128, and 512) are performed at a moderate rate of speed. However, computations for nonstandard KDSI values will show an increase in the time needed to obtain the desired results. This will be especially evident for the computation of phase and activity distributions due to percentage interpolations.

Pages in the development and maintenance branches each have a header at the page top indicating the model, mode, and page type. Sample pages used in the program explanation will be from the Intermediate model. Differences for the Basic model, other then the model name in the page header, will be explained as each page is shown.

Chapter 2

COCOMO Software Cost Estimation Model

This chapter summarizes the COCOMO Basic and Intermediate software cost estimation models. Readers who are already familiar with these COCOMO models can skip to chapter 3, COCOMO Model Utilization.

2.1 Overview

The Basic COCOMO model takes as parameters the estimated number of source instructions (KDSI) and the development mode. The development mode parameter indicates what type of project is being developed, ranging from relatively small projects loosely coupled with their operating environment ("organic") to large, complex systems with rigidly specified interfaces, real-time performance constraints, and high reliability requirements ("embedded"). The Basic model calculates man-months of effort and months of schedule, along with productivity in number of delivered source instructions per man-month and annual development cost. For example, a typical result for a 2 KDSI project might be 6.6 man-months of effort required, 5.1 month schedule and approximately 301 required lines of codes per man-month. Distribution of effort and schedule are also calculated. e.g., of the 5.1 months of development time, the model will tell you that 0.97 months would be spend in product design, 3.23 in programming and unit testing, and 0.92 in integration testing. Requirements analysis are not included in COCOMO estimates, however, product activity distribution by phase for effort is computed. For example, calculated product design for effort would be farther subdivided into requirements analysis, product design, programming, test planning, verification/validation, project office time, quality assurance and documentation development time. Likewise, programming and integration testing would also be subdivided into these same categories.

The Intermediate COCOMO model builds on the Basic model by adding cost drivers, which are measures of various attributes of the product, project, computer and personnel. The product of these cost drivers multiply the calculated effort man-months to produce an adjusted nominal man-month figure. For example, one driver (denoted PCAP) measures Programmer Capability. The PCAP multiplier can range from 0.70 (very high programmer capability) to 1.42 (very low

programmer capability). In the example above, if very high quality programmers were available, the estimated development time would be reduced to 4.62 man-months (6.6 x 0.70) provided the rest of the cost drivers remained at a nominal value of 1.0. Cost drivers give a more comprehensive picture of the product and the environment in which it is to be developed, with resulting greater accuracy of prediction.

The COCOMO models are calibrated using data collected for 63 projects completed by TRW between 1964 and 1979. Numeric parameters were not determined solely by statistical curve fitting, but were influenced by the judgment of project managers. The Basic COCOMO model does not have particularly good accuracy; Boehm reports that estimates for the calibration data are within a factor of 2 of the actual effort only 60% of the time. The added parameters of cost drivers in Intermediate COCOMO give it much improved accuracy. Estimates with the Intermediate COCOMO model are within 20% of actual effort 68% of the time.

It is important, however, to note that the data described above no longer reflects the profiles (e.g., cost drivers) of current and future software. It is imperative that the data base and estimated parameters are constantly updated to improve the prediction power of the COCOMO Tool database. Detailed discussion of such a recalibration process is provided in Boehm's text.

2.2 The Basic Model

The Basic model's parameters are estimated thousands of delivered source instructions (KDSI) and development mode. Source instructions are defined as lines of code, including declarative statements and job control language but excluding comments. Development modes are characterized as follows:

Organic
- generally stable development environment
- minimal need for innovation in architectures or algorithms
- relatively small size
- relatively low premium on early completion of the project
- software project range usually not greater than 50 KDSI
- loose coupling with external systems

Semidetached
- mixture of organic and embedded characteristics
- intermediate level of experience with related systems
- wide mix of experienced and inexperienced people
- some experience with aspects of system under development
- software project range usually not greater than 300 KDSI

Embedded
- much innovation required
- integral part of some larger system with inflexible interface requirements
- high required reliability
- development within tight time and cost constraints

The basic effort development estimation formula by mode are:

```
Organic:          MM = 2.4(KDSI)**1.05

Semidetached:     MM = 3.0(KDSI)**1.12

Embedded:         MM = 3.6(KDSI)**1.20

where
   MM   = man-months of development effort
   KDSI = estimated thousands of delivered source
          instructions
```

Another result obtainable from the Basic COCOMO model is development time, i.e., how many months the project will take to complete. These schedule formula by mode are:

```
Organic:          TDEV = 2.5(MM)**0.38

Semidetached:     TDEV = 2.5(MM)**0.35

Embedded:         TDEV = 2.5(MM)**0.32

where
   TDEV = development time in months
   MM   = effort in man-months calculated above
```

Besides effort and schedule calculations other data which can be computed and are model and mode independent are:

```
Average number of personnel = MM/TDEV

Productivity = (1000*KDSI)/MM

Annual cost = Personnel cost/MM * MM
```

The Basic model also provides information on how the effort and schedule are distributed over the phases of the project. These tabulated percentages are listed in Appendix A (Table A1) and are a function of the product size and mode. The product sizes shown in Table A1 are for standard KDSI values of 2, 8, 32, 128, and 512. KDSI values occurring between these standard figures are considered nonstandard and must have the closest lowest and highest percentages to it interpolated to produce the proper result. KDSI values below and above 2 and 512 KDSI respectively are beyond the boundaries of the COCOMO model and are not used as the model formula for effort and schedule are calibrated only for this range. Values for the phase distribution of effort are computed by multiplying each percentage by the prior computed MM number. Phase distribution for schedule is also computed in a similar way except each schedule percentage is multiplied by the calculated TDEV value.

In addition to the phase distribution computations, activity distribution by phase can also be calculated. The percentages for the activity distribution are listed in Appendix A (Tables A2 - A4). These percentages are again product and mode dependent and provide more detail about the product design, programming, and test integration values computed for phase distribution of effort. Calculation of the values for this area occurs by multiplying the man-months value obtained for phase distribution product design, programming, and test integration by the respective percentages under each appropriate column. For example, to obtain the values for activity distribution in the organic mode for product design, the product design value computed in the phase distribution would be multiplied by each percentage under the product design column to generate the necessary activity phase distribution for product design.

2.3 The Intermediate Model

The key feature which the Intermediate model adds to the Basic model is a set of 15 cost driver attributes, which are listed in Appendix B (Table B1). These cost drivers have a default nominal value of 1.0, however, these values can be varied depending on the environment in which the project is being created. The product of these 15 cost drivers is called the Effort Adjustment Factor (EAF).

Development modes for the Intermediate model are the same as those for the Basic model. However, the effort development estimation formula vary slightly from the Basic model and are:

```
Organic:          MMn = 3.2(KDSI)**1.05

Semidetached:     MMn = 3.0(KDSI)**1.12

Embedded:         MMn = 2.8(KDSI)**1.20

where
   MMn = Nominal man-months
```

The cost drivers are factored in by multiplying the nominal man-months by the EAF:

```
MMadj = MMn * EAF

where
   MMadj = man-months adjusted
```

Schedule formula by mode are the same as for the Basic model. Average number of personnel, productivity, annual cost, phase distribution of effort and schedule, and phase activity distribution are also computed in the same manner as for the Basic model.

For a large system it is likely that the cost driver values will vary for different parts of the system. Estimation accuracy can therefore be improved by dividing the system into components. The nominal man-months are allotted to the components in proportion to their size, and the appropriate set of multipliers are then applied to each component separately. The resulting component estimates are then summed to obtain the overall system estimates.

2.4 Maintenance Model

The process of modifying existing operational software while leaving its primary functions intact is defined as software maintenance. Calculations for the effort and annual cost of this maintenance are also performed in both the Basic and Intermediate models and are mode independent. A new term in this area is called the Annual Change Traffic (ACT). It is the fraction of the software product's source instructions which undergo change during a typical year, either through addition or modification. The value of this factor ranges between 1.00 for complete change to 0 for no change at all to the software. The formulae for ACT is:

$$ACT = \frac{\text{DSI added} + \text{DSI modified}}{\text{Total DSI}}$$

where
ACT = Annual change traffic
DSI = Delivered source instructions

Maintenance formula used with the Basic model are:

$$(MM)am = MM * ACT$$

$$\text{Average maintenance personnel} = (MM)am/12$$

$$\text{Annual maintenance cost} = \text{Maintenance personnel cost} / MM * (MM)am$$

where
(MM)am = Basic annual maintenance effort
MM = Effort in man-months
ACT = Annual change traffic

Calculations for the Intermediate model again vary slightly from the Basic model in that 14 maintenance cost drivers are used to increase the model accuracy. These maintenance cost drivers are listed in Appendix B (Table B2). The value for each maintenance cost driver is defaulted to a nominal value of 1.00, but can be varied according to the environment. The product of these cost

drivers is called the Maintenance Effort Adjustment Factor (EAFM). Formula for the intermediate model are

```
(MM)nam = MMn * ACT * EAFM

Average maintenance personnel = (MM)nam/12

Annual maintenance cost = Maintenance personnel cost/MM
                          * (MM)nam
```

The product activity distribution by phase percentages are listed in Appendix A (Table A5). These percentages are multiplied by either the annual maintenance effort, (MM)am, value in the Basic model or the nominal annual maintenance effort, (MM)nam, value in the Intermediate model to obtain the maintenance activity distribution by phase.
effort

Chapter 3

COCOMO Model Utilization

This chapter illustrates the use of the COCOMO tool through the display of representative screens that are observed during the program. Beginning with information concerning hardware and software required for this program, the chapter guides the user through steps to use the Basic and Intermediate models. COCOMO maintenance is also illustrated and discussed.

3.1 Systems Requirements and Installation Procedures

To properly run the COCOMO Tool program certain hardware and software requirements must be met. The following is a minimum required list.

Hardware:

Microcomputer with at least 256K of memory
Keyboard with function keys
10 megabyte hard disk
Printer (optional)

Software:

KnowledgeMan package including K-Report and K-Graph
COCOMO Tool Program (see Appendix D)
COCOMO Tool Database (see Appendix D)
DOS 2.1 or higher

3.2 Installation Procedures

After the software listed above is loaded onto the system hard disk, the COCOMO Tool program is invoked by typing "KMAN" in response to the system prompt. This results in a display of the COCOMO Tool title page (Figure 3.1) after several moments.

```
-----------
C O C O M O
-----------

COnstructive COst MOdel

for

Estimating Software Development Cost

Dec 1985

Naval Postgraduate School
Monterey, CA

PRESS SPACE BAR
```

Figure 3.1: Title Page

Depressing the space bar once causes the banner page (Figure 3.2) to appear. After depressing the space bar a second time, a model selection page (Figure 3.3) appears.

```
COCOMO PROGRAM

This decision support system program automates the COCOMO
method of software engineering for development and
maintenance. It enables the user to select one of two
models (Basic or Intermediate), and one of three modes
(Organic, Semidetached, or Embedded) for the computation of
development and or maintenance data for a given KDSI input.
Options include phase distribution calculations for
development or maintenance, activity distribution by phase
for development, graphs, reports and model/mode iterations.
Iterations of data can be saved for report generation. Data
can be saved or erased before the program is terminated.

PRESS SPACE BAR TO BEGIN
```

Figure 3.2: Banner Page

3.3 Model/Mode Selection

F1 and F2 invoke the Basic and Intermediate COCOMO models respectively as shown on the model selection page (Figure 3.3). If for some reason program termination is desired, than depressing F10 will end the program and return the system prompt.

```
          -----------
          C O C O M O
          -----------

TO SELECT A MODEL DEPRESS ONE
  OF THE FOLLOWING F KEYS:

       F1   BASIC

       F2   INTERMEDIATE

       F10  END PROGRAM
```

Figure 3.3: Model Selection

```
          -----------
          C O C O M O
          -----------

TO SELECT A MODE DEPRESS ONE

  OF THE FOLLOWING F KEYS:


  F1   INTERMEDIATE ORGANIC

  F2   INTERMEDIATE SEMIDETACHED

  F3   INTERMEDIATE EMBEDDED
```

Figure 3.4: Mode Selection

After depressing F1 or F2, the next item to appear is the mode selection page (Figure 3.4). The selection of a mode

is dependent upon the environment upon which the project is developed. Guidelines for each mode are presented at the beginning of Chapter 2. After selecting either the organic, semidetached, or embedded mode by depressing F1, F2, or F3, respectively, the message "LOADING X MODEL", where X equals either the Basic or Intermediate model selected, will appear at the bottom of the mode selection page. The reason for the message is that it takes several minutes to load the large spreadsheet from the hard disk into memory. Once this action is completed, an effort/schedule page (Figure 3.5) appears at which point the program is in the development branch for the Basic model. The Intermediate model differs from the Basic model at this point in that it enters the development branch.

```
-------------------------------------------------------------
  INTERMEDIATE COCOMO   ORGANIC MODE   EFFORT/SCHEDULE
-------------------------------------------------------------

                KDSI =          <= Enter KDSI
   Personnel Cost/MM =          <= Enter Monthly Personnel Cost

                  Press <F1> to COMPUTE

     Adjusted Effort (MM) =                  EAF =
       Nominal Effort (MM) =
    Productivity (DSI/MM) =
        Schedule (months) =
      Avg Personnel (FSP) =
                Annual Cost =

  <F2> Phase Distribution  <F3> Maintenance    <F10> Quit
```

Figure 3.5: Effort/Schedule

3.4 Development Branch

Once the spreadsheet is loaded, an effort/schedule page (Figure 3.5) appears. This is the point at which the program is in the development branch for the Basic model. The Intermediate model differs from the Basic model at this point in that it enters the development branch when the effort cost river page (Figure 3.6) is displayed. This occurs prior to Figure 3.5 in the Intermediate model.

INTERMEDIATE COCOMO			COST DRIVERS			ORGANIC MODE	
VLow	Low	Nom	High	VHi	XtraHi		
0.75	0.88	1.00	1.15	1.40	1.40	RELY	1.00<Enter values
0.94	0.94	1.00	1.08	1.16	1.16	DATA	1.00
0.70	0.85	1.00	1.15	1.30	1.65	CPLX	1.00
1.00	1.00	1.00	1.11	1.30	1.66	TIME	1.00
1.00	1.00	1.00	1.06	1.21	1.56	STOR	1.00
0.87	0.87	1.00	1.15	1.30	1.30	VIRT	1.00
0.87	0.87	1.00	1.07	1.15	1.15	TURN	1.00
1.46	1.19	1.00	0.86	0.71	0.71	ACAP	1.00
1.29	1.13	1.00	0.91	0.82	0.82	AEXP	1.00
1.42	1.17	1.00	0.86	0.70	0.70	PCAP	1.00
1.21	1.10	1.00	0.90	0.90	0.90	VEXP	1.00
1.14	1.07	1.00	0.95	0.95	0.95	LEXP	1.00
1.24	1.10	1.00	0.91	0.82	0.82	MODP	1.00
1.24	1.10	1.00	0.91	0.83	0.83	TOOL	1.00
1.23	1.08	1.00	1.04	1.10	1.10	SCED	1.00 Press <F1>

Figure 3.6: Cost Drivers

The cost drivers are all initially defaulted to a nominal value of 1.0. These values can be changed by moving the cell cursor, [ ], to the appropriate row by using the down arrow. Once the cell cursor is at the appropriate row, then a new cost driver value can be entered. Cost driver values are arranged in a table to the left of each cost driver with a range from very low to extra high. Once a value is identified from this table for a particular cost driver, its value is typed in and the 'Enter' key is depressed. The entered value then appears at the cell cursor position. Cost drivers which are not modified will retain their value of 1.0. After all the modifications are completed F1, as shown in the last cost driver row, is depressed. A flashing message, "COMPUTING EAF", is displayed at the bottom of the effort cost driver page to indicate that the Effort Adjustment Factor (EAF) is being computed. When this computation is completed, the effort/schedule page (Figure 3.5) is next displayed.

The effort/schedule page is divided into two sections: the input section and the output section. These sections are divided by the "Press <F1> to COMPUTE" statement with the input section above this statement. The cell cursor is positioned in the blank space before the 'Enter KDSI' arrow.

To enter the KDSI value, type in the value and depress the 'Enter key'. The value will appear in the cell cursor location just after KDSI =. For example, if there are 2000 lines of code in the module then enter a 2. THE ONLY KDSI VALUES THAT CAN BE ENTERED HERE MUST OCCUR IN THE RANGE BETWEEN 2 AND 512 KDSI. Any KDSI values outside of this range will cause a program error message to be generated, and a return of the cell cursor to the KDSI position for a new value to be entered. After entering the KDSI value, move the cell cursor down one position by depressing the down arrow once. At this location the personnel cost/MM figure is entered as indicated by the left pointing arrow and the "Enter Monthly Personnel Cost" statement. Enter this figure by typing in the amount and depressing the Enter key. For example, personnel cost/MM for a module might be $3000.00 so 3000 would be entered. After both the KDSI and personnel cost/MM are entered depress F1 to compute the effort and schedule parameters. A flashing message, "COMPUTING EFFORT/SCHEDULE", will appear which indicates that the parameter computations are in progress. When the computations are completed, the effort/schedule will redraw to display the computed parameters in the lower half of the page. For a Basic model display, there is no adjusted effort or EAF. Another iteration for a different KDSI value can be performed at this point by moving the cell cursor back up to the KDSI input position with the up arrow. After entering a new KDSI value, depress F1 for the next iteration to begin. Besides performing another iteration other options include:

F2 - Phase distribution calculations to continue in the development branch

F3 - Maintenance calculations to enter the maintenance branch

F4 - Quit to enter the program termination branch

```
-------------------------------------------------------------
   INTERMEDIATE COCOMO                          ORGANIC MODE
-------------------------------------------------------------

PHASE DISTRIBUTION OF EFFORT FOR KDSI OF:

---EFFORT---
PRODUCT DESIGN              PD
PROGRAMMING                 P
    DETAIL DESIGN           DD
    CODE/UNIT TEST          CUT
INTEGRATION/TEST            IT

---SCHEDULE---
PRODUCT DESIGN              PD
PROGRAMMING                 P
INTEGRATION/TEST            IT


<F1> Graph Effort  <F2> Graph Schedule  <F10> Quit
<F3> Activity Distribution
```

Figure 3.7: Phase Distribution

The selection of F3 or F4 are described in later sections. Selecting F2 causes the phase distribution page, (Figure 3.7), to be displayed with the flashing message, "COMPUTING PHASE DISTRIBUTION", shown in the center of the screen. When the computations are completed by the program the page will redraw to reveal the computed values for the effort and schedule phase distributions. Options available are indicated at the bottom of the page and include:

F1 - Graph Effort values

F2 - Graph Schedule values

F3 - Activity distribution calculations to continue in the development branch

F10 - Quit to enter the program termination branch

```
---------------------------------------------------------------
   INTERMEDIATE COCOMO                          ORGANIC MODE
---------------------------------------------------------------

PRODUCT ACTIVITY DISTRIBUTION BY PHASE FOR EFFORT -- KDSI OF

                      PRODUCT          PRO-          INTEGRATION
                      DESIGN        GRAMMING            /TEST
REQ/MENT ANALYSIS     RA            RA                  RA
PRODUCT DESIGN        PD            PD                  PD
PROGRAMMING           P             P                   P
TEST PLANNING         TP            TP                  TP
VERIFY/VALIDATE       VV            VV                  VV
PROJECT OFFICE        PO            PO                  PO
CM/QA                 CQ            CQ                  CQ
MANUALS               M             M                   M

<F1> Graph Prod Des  <F2> Graph Prog  <F3> Graph IT
                                      <F10> Quit
```

Figure 3.8: Activity Distribution

Selection of F1 or F2 will display a pie chart of the values for the effort and schedule phase distribution, respectively. Depressing the 'Enter' key will remove the pie chart display and redraw the phase distribution page. With the selection of F3, the activity distribution page, (Figure 3.8), will be drawn with the flashing message, "COMPUTING ACTIVITY DISTRIBUTION", shown near the bottom of the page. This message implies that the activity distribution values are in the process of being computed. Upon completion of the computations this page is redrawn to reveal the calculated values. Options are indicated at the bottom of the page and include:

F1 - Graph product design values

F2 - Graph programming values

F3 - Graph integration/test values

F10 - Quit to enter the program termination branch

Selection of F1, F2, or F3 displays a pie chart of the product design, programming, or integration/test values, respectively. Depressing the 'Enter' key removes the pie

chart display and redraws the activity distribution page. With the selection of F10 the program termination branch is entered. The options available in this branch are described in section 3.5.

3.5 Maintenance Branch

The maintenance branch is entered as a result of selecting F3 on the effort/schedule page, (Figure 3.5). For the Intermediate model the maintenance cost driver page, (Figure 3.9), is displayed. The Basic model, however, skips this page and immediately displays the maintenance effort/schedule page, (Figure 3.10).

INTERMEDIATE COCOMO MAINTENANCE DRIVERS ORGANIC MODE

VLow	Low	Nom	High	VHi	XtraHi		
1.35	1.15	1.00	0.98	1.10	1.10	RELY	1.00<Enter values
0.94	0.94	1.00	1.08	1.16	1.16	DATA	1.00
0.70	0.85	1.00	1.15	1.30	1.65	CPLX	1.00
1.00	1.00	1.00	1.11	1.30	1.66	TIME	1.00
1.00	1.00	1.00	1.06	1.21	1.56	STOR	1.00
0.87	0.87	1.00	1.15	1.30	1.30	VIRT	1.00
0.87	0.87	1.00	1.07	1.15	1.15	TURN	1.00
1.46	1.19	1.00	0.86	0.71	0.71	ACAP	1.00
1.29	1.13	1.00	0.91	0.82	0.82	AEXP	1.00
1.42	1.17	1.00	0.86	0.70	0.70	PCAP	1.00
1.21	1.10	1.00	0.90	0.90	0.90	VEXP	1.00
1.14	1.07	1.00	0.95	0.95	0.95	LEXP	1.00
1.35	1.16	1.00	0.86	0.74	0.74	MODP	1.00
1.24	1.10	1.00	0.91	0.83	0.83	TOOL	1.00 Press <F1>

Figure 3.9: Maintenance Cost Drivers

The maintenance cost driver page uses the same approach as the effort cost driver page. Cost driver values are defaulted to the nominal value of 1.0 and are modified as necessary using the values in the displayed table. Depressing F1 after cost driver modification is completed causes the message, "COMPUTING MAINTENANCE EAF", to be displayed at the page bottom. This indicates the computation of the maintenance EAF is in progress. Upon

completion of this computation by the program, the maintenance effort/schedule page, (Figure 3.10), is drawn.

```
-------------------------------------------------------------
  INTERMEDIATE COCOMO      ORGANIC MODE       MAINTENANCE
-------------------------------------------------------------

Maintenance Personnel Cost/MM =  <=Enter Monthly Maint Cost
  Annual Change Traffic (ACT) =  <=Enter ACT Value

                  Press <F1> to COMPUTE


Annual Maintenance Effort (MM)am =              EAFM =
 Maint Software Personnel (FSP)m =
         Annual Maintenance Cost =


     <F2> Maintenance Phase Distribution   <F10> Quit
```

Figure 3.10: Maintenance Effort/Schedule

The maintenance effort/schedule page is also divided into two sections: input and output. The input section is above the statement, "Press <F1> to COMPUTE". The input cell cursor is located in this section in front of the left pointing arrow indicating "Enter Monthly Maint Cost". Enter the dollar amount in the same manner as for the effort and schedule page. Depress the 'Enter' key. Depress the down arrow once to move the cell cursor to the ACT entry position. This value is defaulted to 1.00 which means that the entire software module/project will be modified over the course of the year. If the entire software module/project is not modified over the course of the year then enter the fraction that will be added/modified. This can be determined by using the ACT equation in Chapter 2. The range of this value must be between 0.1 and 1.00. To modify this value, type in the new ACT figure and depress the 'Enter' key. Depressing F1 next initiates the maintenance effort/schedule parameter computation process which is indicated by the flashing message, "COMPUTING MAINTENANCE EFFORT". Upon completion of this computation the page is redrawn with the computed parameter values displayed in the

lower half of the page. Options at the bottom of the page include:

F2 - Maintenance Phase Distribution calculations to continue in the maintenance branch

F10 - Quit to the program termination branch

```
------------------------------------------------------------
  INTERMEDIATE COCOMO                          ORGANIC MODE
------------------------------------------------------------

  PRODUCT ACTIVITY DISTRIBUTION BY PHASE FOR MAINTENANCE


REQ/MENT ANALYSIS   RA
PRODUCT DESIGN      PD
PROGRAMMING         P
TEST PLANNING       TP
VERIFY/VALIDATE     VV
PROJECT OFFICE      PO
CM/QA               CQ
MANUALS             M

            <F1> Graph Values        <F10> Quit
```

Figure 3.11: Maintenance Activity Distribution

Selection of F2 causes the maintenance activity distribution page, (Figure 3.11), to be drawn with the flashing message, "COMPUTING MAINTENANCE PHASE", displayed at the page bottom. Upon completion of these calculations the page is redrawn with the computed values displayed. Options at the page bottom include:

F1 - Graph values

F10 - Quit to the program termination branch

Selection of F1 displays a pie chart of the calculated values. Depressing the 'Enter' key redisplays the maintenance activity distribution page. An F10 selection imitiates the program termination branch which is discussed in the next section.

3.6 Program Termination Branch

Besides the option of program termination, this branch of the program also provides the opportunity to save prior computed values from the development or maintenance branches. Another option allows output of the saved values in a report format on the screen or printer. This branch is activated by selection of F10 in any of the following figures: 3.5, 3.7, 3.8, 3.10, 3.11. The iteration option page, (Figure 3.12), is displayed as a result of the F10 selection.

```
          BEFORE QUITTING . . . . . .

SELECT an option:

       <F1> Another iteration

       <F10> End program
```

Figure 3.12: Iteration Option

This page allows another iteration to be performed or begins the program termination process. Depressing F10 causes the program termination page, (Figure 3.13), to be displayed.

```
BEFORE QUITTING . . . . . .

  <F1> Save calculated values and end program

  <F2> Erase calculated values and end program

  <F3> Continue program
```

Figure 3.13: Program Termination

Selection of F1 terminates the program and displays the operating system prompt. All values that were saved will remain in their respective tables for the Basic and Intermediate COCOMO models. An F2 selection will erase all saved values from the Basic and Intermediate models, and

display the operating system prompt. If no program termination is desired, then depressing F3 will display the model selection page, (Figure 3.3), so another iteration can be performed.

Returning to the iteration option page, (Figure 3.12), depressing F1, "Another iteration", causes the save value option page, (Figure 3.14), to be displayed.

```
BEFORE PERFORMING ANOTHER ITERATION:

  <F1>  SAVE prior computed values

  <F2>  ERASE other computed values and
        START a new table

  <F3>  Perform another iteration WITHOUT
        saving prior computed values
```

Figure 3.14: Save Value Option

Selection of F3 disguards the values computed in the last session and displays the model selection page, (Figure 3.3), so another iteration can be performed. Depressing F1 instead saves the values calculated in the last session. Selection of F2 also saves values calculated in the last session, however, it first erases values saved from other prior sessions. This feature is model dependent in that it erases only values for the model currently in use. For example, if the Intermediate model is currently selected then only prior saved values for the Intermediate model are erased. No values that were saved in prior sessions for the Basic model are touched. Choosing either F1 or F2 results in the display of the data output page, (Figure 3.15).

```
INTERMEDIATE values have been saved

    <F1> Display LAST computed INTERMEDIATE values
    <F2> Display ALL computed INTERMEDIATE values
    <F3> Continue Program
    <F10> End Program
```

Figure 3.15: Data Output

Selection of F3 causes a display of the model selection page, (Figure 3.3), to begin another iteration. Selection of F4 results in the display of the program termination page, (Figure 3.14). Choosing either F1 or F2 invokes the report generator to provide a formatted report. F1 produces a report for only the values of the last session for the model currently in use. F2 produces multiple reports for the values of ALL the prior sessions for the model currently in use. Selection of either F1 or F2 causes the media output page, (Figure 3.16), to be displayed.

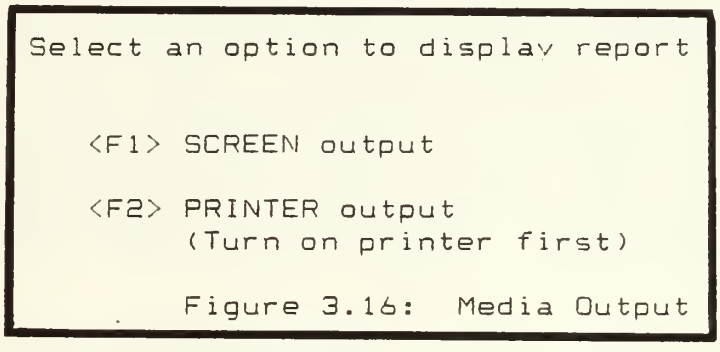


Figure 3.16: Media Output

Selection of F1 produces an output of single or multiple reports, depending on which option was previously selected, on the terminal screen. F2 produces the same formatted reports except they are output on the printer. Depressing the space bar at the end of the last report displays the report completed page, (Figure 3.17).

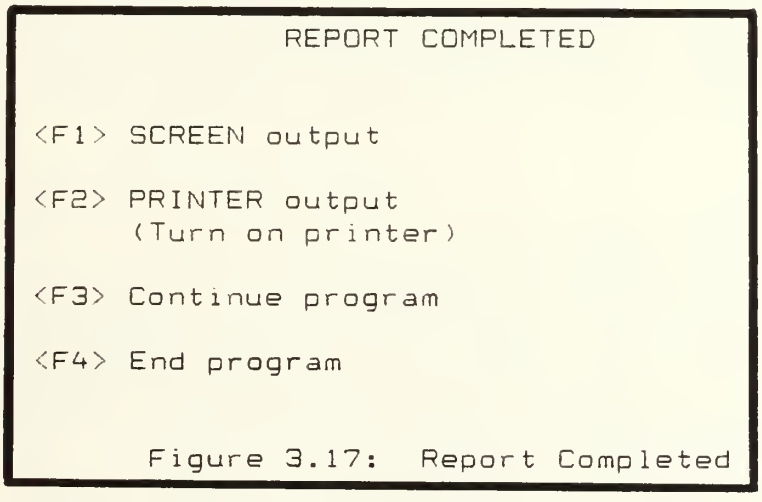


Figure 3.17: Report Completed

Selection of F1 or F2 produce the same results as before. They are included here in order to offer a chance to output the prior generated report in another media from the initial media selected. Selection of F3 or F10 again cause the display of figure 3.3 or 3.14, respectively.

Chapter 4

Model Maintenance

Equations used in the COCOMO model have a large number of empirically derived constants such as coefficients and exponents. In general, these constants provide reasonable accuracy for most software cost-estimation situations. However, users who have appropriate data available may desire to recalibrate these constants in the COCOMO model to better fit the experience of their own organization. Chapter 29 of Boehm's, "Software Engineering Economics" discusses the recalibration procedures. Although the recalibration process itself can become quite complex, altering the constants used in the COCOMO Tool is a relatively easy task. This chapter describes the process of constant modification so once the constants are developed internally they can be changed in the COCOMO Tool.

4.1 Equation Modification

Modification of the COCOMO Tool coefficients and exponents is a simple, straightforward task. These constants reside in two files of the COCOMO program called DEVPARBA.IPF and DEVPARMS.IPF. Using a word processor, such as KEDIT, just call in the above named files one at a time. The equations listed in these files are readily identifiable and can be changed by typing over the coefficients and/or exponents listed. Saving these files back to the hard disk automatically updates the resident files. The COCOMO Tool program is now ready to use with the new modified values.

4.2 Cost Driver Modification

Cost driver modification in the COCOMO Tool requires a little more effort then the equation modification described above. The effort and maintenance cost drivers are located on three spreadsheets in the COCOMO Tool database. These spreadsheets are listed as CIO.ICF, CIS.ICF, and CIE.ICF. To modify the cost driver values listed on the above named spreadsheets follow the sequence of steps below.

1. Rename the file STARTUP.IPF to STOP.IPF while in DOS.
2. Type in "KMAN" in response to the DOS prompt.

3. After obtaining the MDBS header and a "_", type in "CALC" and depress the 'Enter' key.
4. Upon the display of a blank spreadsheet, type, \LOAD FROM "CIO.ICF", and depress the 'Enter' key.
5. The spreadsheet that next appears will display the effort cost drivers for the Intermediate organic model. Type, \BORDER, to obtain letters and numbers across the top and bottom, respectively. The cell cursor location occurs in the upper right part of the screen. For example, #F100 means the cursor is in column F and row 100 of the spreadsheet.
6. Using the left arrow key move the cursor to the left until the cell cursor in the upper right hand corner reads #A100. In this position, the numbers in that row appear across the bottom of the screen. If no numbers are to be changed in that row then move the cell cursor with the down arrow key until the appropriate row where values are to be changed is reached.
7. After reaching the appropriate row where values are to be changed type, \EDIT. Use Control D to move the cursor to the right until it is over the number which needs to be changed. At this point just type in the new number over the old number. Once all the numbers in the displayed row have been changed, depress the 'Enter' key.
8. Move the cell cursor down to the next row to be changed and repeat the process in step 7, if necessary.
9. Maintenance cost drivers can be modified in the same manner as the effort cost drivers. After finishing the effort cost driver modification, type, \#A115, to reach the maintenance cost drivers. Move the cell cursor down to the appropriate row as described in step 6, and make changes as indicated in step 7.
10. Once all of the cost drivers have been modified, the next step is to save these changes. To accomplish this type in the following:

    \#A96 followed by 'Enter'
    \#F100 followed by 'Enter'
    \BYE

    After the "_" appears, type, SAVE TO "CIO.ICF" and depress the 'Enter' key.

11. Upon return of the "_" again repeat steps 3 through 10. The only change is to replace CIO.ICF with CIS.ICF and CIE.ICF for changes to these spreadsheets.
12. After changing all of the cost driver values in all three spreadsheets, type, BYE, in response to the last "_" prompt. This returns the system to DOS. Rename the STOP.IPF file back to STARTUP.IPF. The program is now ready to run.

APPENDIX A

COCOMO Percentages

Effort Distribution

	KDSI SIZE				
	2	8	32	128	512
Organic Mode					
Product Design	16	16	16	16	16
Programming	68	65	62	59	59
Detailed Design	26	25	24	23	23
Code & Unit Test	42	40	38	36	36
Integration & Test	16	19	22	25	25
Semidetached Mode					
Product Design	17	17	17	17	17
Programming	64	61	58	55	52
Detailed Design	27	26	25	24	23
Code & Unit Test	37	35	33	31	29
Integration & Test	19	22	25	28	31
Embedded Mode					
Product Design	18	18	18	18	18
Programming	60	57	54	51	48
Detailed Design	28	27	26	25	24
Code & Unit Test	32	30	28	26	24
Integration & Test	22	25	28	31	34

Schedule Distribution

	KDSI SIZE				
	2	8	32	128	512
Organic Mode					
Product Design	19	19	19	19	19
Programming	63	59	55	51	51
Integration & Test	18	22	26	30	30
Semidetached Mode					
Product Design	24	25	26	27	28
Programming	56	52	48	44	40
Integration & Test	20	23	26	29	32
Embedded Mode					
Product Design	30	32	34	36	38
Programming	48	44	40	36	32
Integration & Test	22	24	26	28	30

Table A1: Effort and Schedule Percentages by Phase

	Product Design				Pro-gramming				Integration & Test			
KDSI SIZE:	S	I	M	L	S	I	M	L	S	I	M	L
Requirements Analysis	15				5				3			
Product Design	40				10				6			
Programming	14				58				34			
Test Planning	5				4				2			
Verification & Validation	6				6				34			
Project Office	11				6				7			
CM/QA	2				6				7			
Manuals	7				5				7			

S = 2 KDSI I = 8 KDSI M = 32 KDSI L = 128 KDSI

Table A2: Activity Distribution by Phase - Organic

	Product Design					Pro-gramming					Integration & Test				
KDSI SIZE:	S	I	M	L	VL	S	I	M	L	VL	S	I	M	L	VL
Requirements Analysis	12.5	<----------			>12.5	4	4	4	4	4	2.5	<----------			>2.5
Product Design	41	<----------			>41	8	8	8	8	8	5	5	5	5	5
Programming	12	12.5	13	13.5	14	56.5	<----------			>56.5	33	35	37	39	41
Test Planning	4.5	5	5.5	6	6.5	4	4.5	5	5.5	6	2.5	2.5	3	3	3.5
Verification & Validation	6	6.5	7	7.5	8	7	7.5	8	8.5	9	32	31	29.5	28.5	27
Project Office	13	12	11	10	9	7.5	7	6.5	6	5.5	8.5	8	7.5	7	6.5
CM/QA	3	2.5	2.5	2.5	2	7	6.5	6.5	6.5	6	8.5	8	8	8	7.5
Manuals	8	8	7.5	7	7	6	6	5.5	5	5	8	8	7.5	7	7

S = 2 KDSI I = 8 KDSI M = 32 KDSI L = 128 KDSI VL = 512 KDSI

Table A3: Activity Distribution by Phase - Semidetached

	Product Design					Pro-gramming					Integration & Test				
KDSI SIZE:	S	I	M	L	VL	S	I	M	L	VL	S	I	M	L	VL
Requirements Analysis	10	10	10	10	10	3	3	3	3	3	2	2	2	2	2
Product Design	42	42	42	42	42	6	6	6	6	6	4	4	4	4	4
Programming	10	11	12	13	14	55	55	55	55	55	32	36	40	44	48
Test Planning	4	5	6	7	8	4	5	6	7	8	3	3	4	4	5
Verification & Validation	6	7	8	9	10	8	9	10	11	12	30	28	25	23	20
Project Office	15	13	11	9	7	9	8	7	6	5	10	9	8	7	6
CM/QA	4	3	3	3	2	8	7	7	7	6	10	9	9	9	8
Manuals	9	9	8	7	7	7	7	6	5	5	9	9	8	7	7

S = 2 KDSI I = 8 KDSI M = 32 KDSI L = 128 KDSI VL = 512 KDSI

Table A4: Activity Distribution by Phase - Embedded

	Organic				Semidetached					Embedded				
KDSI SIZE:	S	I	M	L	S	I	M	L	VL	S	I	M	L	VL
Requirements Analysis	7	7	7	7	6.5	6.5	6.5	6	6	6	6	6	5	5
Product Design	13	13	13	13	12	12	12	12	12	11	11	11	11	11
Programming	45	44	43	42	41.5	41.5	41	41	41	38	39	39	40	41
Test Planning	3	3	3	3	3	3	3.5	4	4.5	3	3	4	5	6
Verification & Validation	10	11	12	13	11	12	13	13.5	14	12	13	14	14	14
Project Office	7	7	7	7	8.5	8	7.5	7	6.5	10	9	8	7	6
CM/QA	5	5	5	5	6.5	6	6	6	5.5	8	7	7	7	6
Manuals	10	10	10	10	11	11	10.5	10.5	10.5	12	12	11	11	11

S = 2 KDSI I = 8 KDSI M = 32 KDSI L = 128 KDSI VL = 512 KDSI

Table A5: Activity Distribution by Phase for Maintenance

APPENDIX B

Cost Drivers

	Very Low	Low	Nominal	High	Very High	Extra High
RELY - required software reliability	0.75	0.88	1.00	1.15	1.40	1.40
DATA - database size	0.94	0.94	1.00	1.16	1.16	1.16
CPLX - product complexity	0.70	0.70	1.00	1.15	1.30	1.65
TIME - execution time constraint	1.00	1.00	1.00	1.11	1.30	1.66
STOR - main storage constraint	1.00	1.00	1.00	1.06	1.21	1.56
VIRT - virtual machine volatility	0.87	0.87	1.00	1.15	1.30	1.30
TURN - computer turnaround time	0.87	0.87	1.00	1.07	1.15	1.15
ACAP - analyst capability	1.46	1.19	1.00	0.86	0.71	0.71
AEXP - applications experience	1.29	1.13	1.00	0.91	0.82	0.82
PCAP - programmer capability	1.42	1.17	1.00	0.86	0.70	0.70
VEXP - virtual machine experience	1.21	1.10	1.00	0.90	0.90	0.90
LEXP - programming language experience	1.14	1.07	1.00	0.95	0.95	0.95
MODP - use of modern prog. practices	1.24	1.10	1.00	0.91	0.82	0.82
TOOL - use of software tools	1.24	1.10	1.00	0.91	0.83	0.83
SCED - required development schedule	1.23	1.08	1.00	1.04	1.04	1.04

Table B1: Effort Cost Drivers

	Very Low	Low	Nominal	High	Very High	Extra High
RELY - required software reliability	1.35	1.15	0.98	1.10	1.10	1.40
DATA - database size	0.94	0.94	1.00	1.16	1.16	1.16
CPLX - product complexity	0.70	0.85	1.00	1.15	1.30	1.65
TIME - execution time constraint	1.00	1.00	1.00	1.11	1.30	1.66
STOR - main storage constraint	1.00	1.00	1.00	1.06	1.21	1.56
VIRT - virtual machine volatility	0.87	0.87	1.00	1.15	1.30	1.30
TURN - computer turnaround time	0.87	0.87	1.00	1.07	1.15	1.15
ACAP - analyst capability	1.46	1.19	1.00	0.86	0.71	0.71
AEXP - applications experience	1.29	1.13	1.00	0.91	0.82	0.82
PCAP - programmer capability	1.42	1.17	1.00	0.86	0.70	0.70
VEXP - virtual machine experience	1.21	1.10	1.00	0.90	0.90	0.90
LEXP - programming language experience	1.14	1.07	1.00	0.95	0.95	0.95
MODP - use of modern prog. practices	1.35	1.16	1.00	0.86	0.74	0.74
TOOL - use of software tools	1.24	1.10	1.00	0.91	0.83	0.83

Table B2: Maintenance Cost Drivers

APPENDIX C

Error Messages

1. ACT RANGE 0 TO 1 ONLY

   Problem: Maintenance ACT value outside 0 to 1 boundary

   Solution: Input an ACT value between 0 and 1. Depress F1 again to compute the maintenance parameters.

2. CAN'T USE NEGATIVE OR ZERO VALUES

   Problem:
   1. An effort or maintenance cost driver value is less than or equal to zero.

   2. Maintenance personnel cost/MM = 0

   Solution:
   1. Input a proper cost driver from the effort or maintenance table. Depress F1.

   2. Input a maintenance cost/MM and depress F1.

3. INSUFFICIENT MEMORY

   Problem: KMAN memory constraints exceeded by the program execution and causing the program to terminate abruptly.

   Solution: Type, BYE and depress 'Enter' key
   Upon receiving the DOS prompt type, KMAN. to begin program execution again.

   NOTE: Values saved prior to the abrupt program termination still exist in the database files.

4. KDSI IS GREATER THAN 512
   KDSI >2 OR <512 ONLY

   Problem: KDSI input value is greater than 512.

   Solution: Reenter a KDSI value between 2 and 512.
   Depress F1.

5. KDSI IS LESS THAN 2
   KDSI >2 OR <512 ONLY

   Problem: KDSI input value is less than 2.

   Solution: Reenter a KDSI value between 2 and 512.
   Depress F1.

6. WRONG KEY

   Problem: An F key not shown on the screen selection list was depressed.

   Solution
   1. If in an input screen (e.g., KDSI input) then check the input values and reenter any which may have been changed. Depress the proper F key.

   2. If not in an input screen then just depress the proper F key.

APPENDIX D

Program/Data Base Listings

1. Program Listings

File Name	Module Name
STARTUP.IPF	START_UP
COCOMO.IPF	COCOMO
COCO.IPF	COCO
SETUPBAS.IPF	SET_UP_BASIC
SSLODBAS.IPF	SS_LOAD_BASIC
DEVPARBA.IPF	DEV_PARAMETER_BASIC
SETUPINT.IPF	SET_UP_INTERMEDIATE
SSLODINT.IPF	SS_LOAD_INTERMEDIATE
REDEVDAT.IPF	READ_DEVELOPMENT_DATA
DEVPARMS.IPF	DEV_PARAMETERS
DEVPHDIS.IPF	DEV_PHASE_DISTR
PHASEDIS.IPF	PHASE_DISTR
CALCEFSC.IPF	CALC_EFF_SCHED
SELTABLE.IPF	SELECT_TABLE
EVALKDSI.IPF	EVAL_KDSI
SLECTONE.IPF	SELECT_ONE
SELECTWO.IPF	SELECT_TWO
INTERPOL.IPF	INTERPOLATION
DEVACDIS.IPF	DEV_ACT_DISTR
DEVPAD.IPF	DEV_PAD
CALCDPAD.IPF	CALC_DEV_PAD
MAINTBAS.IPF	MAINT_BASIC
MAINTINT.IPF	MAINT_INTERMEDIATE
MDATAPAD.IPF	MAINT_DATA_PAD
CALCMDAT.IPF	CALC_MAINT_DATA
MAINTPAD.IPF	MAINT_PAD
KEYCHNQ.IPF	KEY_CHANGE
AGAINTIT.IPF	AGAINT_IT
WRAPUP.IPF	WRAP_UP
KILLIT.IPF	KILL_IT
SAVVAL.IPF	SAVE_VALUES
ERASTABL.IPF	ERASE_TABLE
ANOTHER.IPF	ANOTHER
RPTOUT.IPF	REPORT_OUT
BRPTONE.IPF	BASIC_RPT_ONE
BRPTALL.IPF	BASIC_RPT_ALL
IRPTONE.IPF	INTERMEDIATE_RPT_ONE
IRPTALL.IPF	INTERMEDIATE_RPT_ALL
GRAFPHE.IPF	GRAF_PHASE_EFFORT

GRAFPHS.IPF	GRAF_PHASE_SCHEDULE
GRAFADPD.IPF	GRAF_ACT_DIST_PD
GRAFADP.IPF	GRAF_ACT_DIST_PHASE
GRAFADIT.IPF	GRAF_ACT_DIST_INTEST
GRAFPDM.IPF	GRAF_PD_MAINT

2. Data Base Listings

a. Spreadsheets

File Name	Spreadsheet Name
CBO.ICF	COCOMO_BASIC_ORGANIC
CBS.ICF	COCOMO_BASIC_SEMIDETACHED
CBE.ICF	COCOMO_BASIC_EMBEDDED
CIO.ICF	COCOMO_INTERMEDIATE_ORGANIC
CIS.ICF	COCOMO_INTERMEDIATE_SEMIDETACHED
CIE.ICF	COCOMO_INTERMEDIATE_EMBEDDED

b. Tables

File Name	Table Name
BES.ITB	BASIC_EFFORT_SCHED
BESP.ITB	BASIC_EFFORT_SCHED_PHASE
BESPAD.ITB	BASIC_EFFORT_SCHED_PAD
BESM.ITB	BASIC_EFFORT_SCHED_MAINT
BESMAD.ITB	BASIC_EFFORT_SCHED_MAINT_ACT_DIST
IES.ITB	INT_EFFORT_SCHED
IESP.ITB	INT_EFFORT_SCHED_PHASE
IESPAD.ITB	INT_EFFORT_SCHED_PAD
IESM.ITB	INT_EFFORT_SCHED_MAINT
IESMAD.ITB	INT_EFFORT_SCHED_MAINT_ACT_DIST

LIST OF REFERENCES

1. Artzer, S. P. and Neidrauer, R. A., Software Engineering Basics: A Primer for the Project Manager, Master's Thesis, Naval Postgraduate School, Monterey, CA, June, 1982.

2. Carabello, J. M., Office of the Assistant Secretary of Defense Memorandum for ADP Policy Committee (Program Management) dated 8 May 1981.

3. Douville, A. A., Salasin, J. and Probert, T. H., "The Impact of ADA on COCOMO cost estimates as applied to the World Wide Military Command & Control (WWMCCS) Information System (WIS)", IDA Paper P-1810, Institute for Defense Analysis, Alexandria, VA, January, 1985.

4. Vorgang, B. R., A Macro Approach to Software Resource Estimation and Life Cycle Control, Master's Thesis, Naval Postgraduate School, Monterey, CA, December, 1981.

5. Boehm, B. W., Software Engineering Economics, Prentice-Hall, Inc., Englewood Cliffs, NJ, 1981

6. Green, J. F. and Selby, B. F., Dynamic Planning and Control of Software Maintenance: A Fiscal Approach, Master's Thesis, Naval Postgraduate School, Monterey, CA, December, 1981.

7. Yourdon, E. and Constantine, L., Structured Design, Prentice-Hall, Englewood, NJ, 1979.

8. DeMarco, T., Structured Analysis and System Specification, Prentice-Hall, Englewood, NJ, 1981.

9. Bennett, John L., Building Decision Support Systems, Addison-Wesley Publishing Company, Inc., 1983.

10. Sprague, Ralph H., Jr., and Carlson, Eric D., Building Effective Decision Support Systems, Prentice-Hall, Englewood, NJ, 1982.

11. Pressman, Roger S., Software Engineering: A Practioner's Approach, McGraw-Hill, 1982.

INITIAL DISTRIBUTION LIST

		No. Copies
1.	Defense Technical Information Center Cameron Station Alexandria, Virginia 22304-6145	2
2.	Library, Code 0142 Naval Postgraduate School Monterey, California 93943-5002	2
3.	Department Chairman, Code 54 Department of Administrative Sciences Naval Postgraduate School Monterey, California 93943-5000	1
4.	Computer Technology Programs, Code 37 Naval Postgraduate School Monterey, California 93943-5000	1
5.	Professor Norman R. Lyons, Code 54Lb Department of Administrative Sciences Naval Postgraduate School Monterey, California 93943-5000	1
6.	Professor Tung Bui, Code 54Bd Department of Administrative Sciences Naval Postgraduate School Monterey, California 93943-5000	1
7.	DODCI ATTN: Mrs. Sarah Taylor Bldg 175 Washington Navy Yard Washington, D.C. 20374	1
8.	Commanding Officer ATTN: LCDR A. N. N. Sullivan (Code 73) Naval Recruiting Command 4015 Wilson Blvd Arlington, VA. 22203-1991	2
9.	Commanding Officer ATTN: LT J. M. Darabond (Code 30B) Fleet Intelligence Center, Europe and Atlantic Norfolk, VA. 23511-6690	2