NOTICE

THIS DOCUMENT HAS BEEN REPRODUCED FROM MICROFICHE. ALTHOUGH IT IS RECOGNIZED THAT CERTAIN PORTIONS ARE ILLEGIBLE, IT IS BEING RELEASED IN THE INTEREST OF MAKING AVAILABLE AS MUCH INFORMATION AS POSSIBLE
Deep Space Network Software Cost Estimation Model

Robert C. Tausworthe

April 15, 1981

National Aeronautics and Space Administration
Jet Propulsion Laboratory
California Institute of Technology
Pasadena, California
Deep Space Network Software Cost Estimation Model

Robert C. Tausworthe

April 15, 1981

National Aeronautics and Space Administration

Jet Propulsion Laboratory
California Institute of Technology
Pasadena, California
The research described in this publication was carried out by the Jet Propulsion Laboratory, California Institute of Technology, under contract with the National Aeronautics and Space Administration.
## Contents

I. Introduction .......................................................... 1

II. Model Description .................................................. 2
   A. Estimation of Program Size .................................... 2
   B. Estimation of Productivity .................................... 5
   C. Estimation of Implementation Task Duration ............... 7
   D. Average Staff ................................................... 7
   E. Documentation Sizing and Cost ................................ 7
   F. Computer Resources ............................................. 7
   G. Variance Computation .......................................... 7
   H. Modified Rayleigh-Norden-Putnam Model ..................... 8
   I. Standard Work Breakdown Structure Model .................. 11
   J. Operational Modes ............................................. 11
   K. Risk-Bics Computation ........................................ 12

III. Selection and Calibration of Parameters ....................... 12
   A. Productivity Statistics ....................................... 12
   B. Staff Size Statistics ......................................... 13
   C. Project Duration Statistics ................................... 14
   D. Effort Factors of Inherited Code Utilization .............. 14
   E. High-Level Language Adjustment .............................. 14
   F. Capacity and Timing Constraint Adjustment ................ 16
   G. DSN Documentation Statistics ................................. 16

IV. Typical Example of the Model Operation ......................... 18

V. Summary and Conclusion ........................................... 18

References ............................................................. 36

Appendix A. DSN Software Cost Model Factors .................... 38

Appendix B. DSN Software Cost Model Standard Work Breakdown Structure Skeleton .......... 43
Figures

1. The software cost and resource estimation model data flow ........... 3
2. Software implementation tasks related to new and inherited code activities ........................................... 4
3. Industry estimates of effort vs source lines of code delivered .......... 13
4. Average staffing required vs project effort ........................................... 13
5. Average duration of software projects vs the effort required ........... 14
6. Blank form for software cost model inputs ........................................... 19
7. Example of input form usage ........................................... 22
8. Hard copy format input parameter record ........................................... 25
9. Output of SWAG estimator using the "typical software project" parameters ........................................... 28
10. Full PERT/CPM work breakdown structure and schedule table output of the software cost model ........................................... 29
11. Short-form output of the PERT/CPM WBS schedule data ............. 31
12. Gantt chart output of the software cost model ........................................... 33

Tables

1. Standard work breakdown structure ........................................... 12
2. Inherited code effort requirement estimates ........................................... 15
3. Assembly language effort requirement estimates ........................................... 16
4. Capacity constraint effort requirement estimates ........................................... 17
5. Time critical module effort estimates ........................................... 17
6. DSN documentation statistics ........................................... 18
Abstract

This report presents a parametric software cost estimation model prepared for JPL Deep Space Network (DSN) Data Systems implementation tasks. The resource estimation model modifies and combines a number of existing models, such as those of the General Research Corp., Doye Associates, IBM (Walston-Felix), Rome Air Force Development Center, University of Maryland, and Rayleigh-Norden-Putnam. The model calibrates the task magnitude and difficulty, development environment, and software technology effects through prompted responses to a set of approximately 50 questions. Parameters in the model are adjusted to fit JPL software life-cycle statistics. The estimation model output scales a standard DSN Work Breakdown Structure, which is then input to a PFRT/CPM system, producing a detailed schedule and resource budget for the project being planned.
I. Introduction

The early-on estimation of required resources and schedule for the development and maintenance of software has probably been the least precise aspect of the software life cycle, and yet, an orderly and rational attempt must be made in order to plan and organize an implementation effort. The existence of an orderly and rational approach implies the existence of a resource and schedule model that accepts as input the technical requirements to be achieved—the magnitude of the task; the physical, environmental, human, and management constraints assumed or known to be in effect; the history base of similar and dissimilar experience; the means, alternatives, and technology available to the task; and a theory which is capable of correlating these parameters with measured results.

The least precise of such models is one which relies entirely on experience, intuition, and luck. It is sometimes referred to as a "WAG", or "Wildly Aspiring Guess." (More often, the acronym is somewhat differently derived, but with the same general connotation.) When a more formalized, mathematical model with some statistical verification can be formulated, the model appellation is upgraded to "Scientific WAG", or "SWAG."

The prediction of human behavior is the problem of estimating events in a stochastic process governed by an unknown probability function. The goal of a SWAG model is therefore to predict the events in such a way as to produce minimum variance (or risk). The optimum SWAG model can predict only to the limit imposed by the statistical distribution actually characterizing the human activity.

The optimal SWAG model would require the precise quantification of all technical, environmental, and human behavioristic parameters, and would combine these into a mathematical formula producing maximum likelihood or least mean square error results. Lacking this precise quantification, the best that one may hope for in a SWAG model is that it accommodate the principal factors affecting the estimate variance (or risk) in a way that reduces the variance (or risk) from what it would be, had that factor not been included.

There are a number of SWAGs in existence. Fourteen software cost estimation models are summarized in Ref. 1, and nine are evaluated for JPL use in Ref. 2. None of these, by itself, seemed to the author to contain sufficient range of application and adaptability to the diverse kinds of software being produced at JPL to quantify the relevant cost factors and risks with sufficient accuracy to be useful. Taken all together, however, these models seemed to possess, in their union, the potential for as good a SWAG as could be obtained at the current state of the art.

An IBM study (Walston-Felix, Ref. 3) reported the analysis of 60 software projects with respect to 68 variables believed to influence productivity. Of these, 29 showed a significant, high correlation with productivity and were included in their estimation model.
A number of models reported in Ref. 1 (General Research Corp., Doty Associates, TRW, Air Force Electronic Systems Division, Tecolote, Aerospace Corp., etc.), as well as statistics from the University of Maryland (Ref. 4), provide productivity data with best-fit curves using many fewer parameters.

Still other models, notably the Rayleigh-Norden Putnam model (Refs. 5, 6) presuppose a few-parameter, specific mathematical model, which is then calibrated using available industry data to provide tradeoffs between effort, duration, and risk.

Several models (e.g., Wolverton, Ref. 7) proceed to detail resource expenditures into the various phases of activity. Some of the models are fully automated, such as PRICE-8 (Ref. 8), SLIM (Ref. 9), and SLACE (Ref. 10). The others appear to be calculative, or perhaps small programs.

The software cost model described in this report is fully automated; it borrows and extends features from many of the models above. It utilizes 7 factors from the GRC model, 29 factors from, or similar to, the Walston-Felix model, and incorporates an inherited (or existing) code model due to the author, exposed herein. It utilizes the “PERT” technique to estimate the expected size and variance of the software to be produced. It utilizes a modification of the Rayleigh-Norden- Putnam model to check on the feasibility of resource estimates. It applies the estimated effort, staff, and duration to a standard Work Breakdown Structure (WBS) developed for DSN software tasks, and automatically produces a task plan and schedule to be used at the initial system/subsystem planning, software implementation planning, or software maintenance planning stages of a project.

There are 68 parameters within the model which relate to productivity, duration, staffing level, documentation, and computer resources. Another 69 parameters divide the total estimated effort among each WBS subtask; an additional 69 relate total duration to subtask duration. Subtask precedences are adjustable, and drive a PERT/Critical-Path-Method scheduling algorithm.

The outputs of the model include estimates and variance values for program size, staff productivity, effort, duration, staffing, documentation, and computer resources, together with complete scheduling early/late start/finish and float-time data, plus a Gantt chart (schedule bar chart) of the planned activities.

All parameters in the automated model are easily altered by a simple text editing process, without recompilation of the programs. For all its seeming complexity, the model itself is simple to use. Only a series of questions relating to size and environment need to be answered.

The ensuing sections of this report describe the model and its parameters, and discuss the source or deviation of parameter values and other elements of the model. The values of parameters cited in this report are subject to modification and refinement as further calibration and usage of the model proceeds. The reader interested in a current set of values at any time subsequent to the publication of this report may consult the Software Specification Document (Ref. 11).

Concerning accuracy: at this writing, insufficient data has been collected from DSN projects to optimize the parameters of the model to fit DSN productivity, duration, etc. Therefore, the model accuracy is unknown, as pertaining to DSN predictions. However, the model does fit industry statistics (or can be made to fit any of the cited source models) by proper parameter selection. For this reason, it is felt that the few JPL data points that have been factored in will yield accuracy figures as good as, or perhaps better than, the other models in their stated environments.

II. Model Description

The Software Cost and Resource Estimation Model (the acronym SCAREM is not used) overview appears in Fig. 1 in data-flow-diagram format. Program size and staff productivity factors are separately estimated and then combined to estimate effort, staffing, duration, documentation, and computer resources. The model produces unflated dollar costs for documentation and computer resources. Both estimated mean and variance values for all resources are output.

Estimated values are presented in the automated model to the user as advice. The user is admonished to use these figures with sufficient risk biases to ensure project completion within a desired confidence value. The model then accepts any two of the three parameters: effort, average staff, and duration.

These entries are checked against a model akin to the Rayleigh-Norden-Putnam model, but altered to conform with power-law fits to measured data. Warnings are issued if the entries are unreasonable. The user may alter the input estimates, if desired, for another check.

Once acceptable resources and duration have been decided, the model proceeds to produce a standard DSN software implementation work breakdown structure and schedule without further input from the user.

A. Estimation of Program Size

The size of the software task is measured in “equivalent” Kilo-Source-Lines of Executable Code (KSLCE). A source line of executable code is defined basically as a source language
Fig. 1. The software cost and resource estimation model data flow.
statement occupying one physical line in the source (display) medium that results in generation of object code, reservation of storage, or definition of data type. Comments are excluded, as are statements merely defining labels and equivalences of identifiers. If several basic statements may appear on one physical source line, each such statement should be added separately into the KSL/EC count.

Source lines of new code are weighted differently than lines of reused code, in proportion to the relative amount of effort required to adapt the inherited code to the current task. Even deleted lines of code contribute to the programming effort, and therefore increase the "equivalent" KSL/EC count.

The programming tasks involved with the generation of new code and reuse of existing code are depicted in Fig. 2. The effort to specify, produce, document, and test a new line of code is normalized to unity; the lines of code added, changed, deleted, and retested-only contribute to the equivalent line count according to relative effort. The extent of existing-module modification is measured by the number of lines added in, the number changed, and the number deleted. The number of equivalent lines of code produced is then defined to be

\[ I_{eq} = I_{new} + h I_{mod} + c I_{add} + d I_{chg} + e I_{det} \\
+ g I_{removed} + h I_{retest} \]  

(1)

\[ I_{eq} = I_{new} + 0.27 I_{mod} + 0.53 I_{add} + 0.24 I_{chg} \\
+ 0.15 I_{del} + 0.11 I_{remove} + 0.27 I_{retest} \]  

(2)

Each of the code-size parameters in Eq. (2) is estimated by the "PERT" technique (Ref. 12). This technique presumes that guesses for \( I \) are governed by a beta distribution (Ref. 13).

\[ p(x) = B(s + 1, r + 1)x^s(1 - x)^r \text{ for } 0 \leq x \leq 1 \]  

(3)

The assumptions with respect to each component are the following:

1. New code is subjected to the entire standard implementation process.
2. The recognition of the reuse of existing code is made in the architectural phase, so code added, changed or deleted from modules goes only through subsequent phases.
3. Added code takes the same effort as new code in corresponding phases where activity takes place.
4. Changed code requires the same design and testing effort as new code, but less documentation and coding effort.
5. Deleted lines from existing modules require reduced design, coding, and documentation effort, and no testing of the deleted lines.
6. Any module changed is completely retested and requalified.
7. Deleted modules require less architectural, interface, and detailed design considerations than new code; only that coding effort required to remove the unwanted code contributes to the coding time; no testing of the deleted code is possible; and documentation effort involves removal of entire sections of existing material and minor cleanup. Retesting is covered in modules which interfaced with the deleted module.
8. Retested unmodified code requires revalidation of the interface design and retesting efforts only.

The analysis in Section III D produces the following estimated typical values for the DSN environment:

\[ I_{eq} = I_{new} + 0.31 I_{mod} + 0.53 I_{add} + 0.24 I_{chg} \]

\[ = 0.15 I_{del} + 0.11 I_{remove} + 0.27 I_{retest} \]  

(2)
where $R(t, \cdot)$ is the beta function (Ref. 14),

$$\nu = \frac{I - I_{\min}}{I_{\max}}$$

and $v = s + t$. This distribution has mean value

$$I_{\text{avg}} = \frac{I_{\min} + I_{\max} + v I_0}{v + 2}$$

(4)

in which $I_0$ is the value of $I$, yielding the maximum value of $\nu(v)$.

The variance of $I$ about $I_{\text{avg}}$ is

$$\text{var}(I) = \frac{(v+1)(I_{\max} - I_{\min})^2 + v^2(I_0 - I_{\min})(I_{\max} - I_0)}{(v+2)^2(v+3)}$$

(5)

For a given value of $v$, the variance of $I$ is maximum when

$$s = t = \frac{v}{2}$$

giving

$$\text{Max} [\text{var}(I)] = \frac{(I_{\max} - I_{\min})^2}{4(v+3)}$$

(6)

The value $v = 4$ is quite often used (Ref 6) and is adopted here. The estimators adopted in the model are calculated from guesses for $I_{\max}$, $I_{\min}$, and $I_0$:

$$I_{\text{est}} = \frac{I_{\max} + 4 I_0 + I_{\min}}{6}$$

(7)

$$I_{\text{var}} = \frac{s (I_{\max} - I_{\min})^2 + 16 (I_{\max} - I_0)(I_0 - I_{\min})}{282}$$

(8)

The variance value with $s = t = 2$ gives a standard deviation of

$$I_{\text{sigma}} = \frac{I_{\max} - I_{\min}}{5.29}$$

(9)

The denominator value of Eq. (9) usually published in the literature (Ref. 6) is 6, which underestimates the variance by about 13%.

The estimated value for $I_{\text{avg}}$ is composed of the usual weighted sum of the $I_{\text{est}}$ for each parameter, and its standard deviation is the square-weighted root-sum-square of the individual deviations. The weights in Eq. (1) are used.

**B. Estimation of Productivity**

In this model, the productivity $P$ is defined as total equivalent KSLFC (here denoted $I$) produced, divided by the total staff effort (here denoted $W$),

$$P = \frac{I}{W}, \text{ KSLFC staff-month}$$

(10)

A number of data bases (e.g., Refs. 3, 15, 16) have shown that $I$ and $W$ are correlated through a power-law relationship

$$W = \frac{I^a}{P_1}$$

(11)

where $P_1$ is the average KSLFC productivity rate. The productivity at other values of $I$ is given approximately by

$$P \approx P_1 I^{1+\alpha}$$

(12)

The value of $P_1$ is set primarily by technology and environment. In fact, industry studies show that $P_1$ may vary by as much as 50:1 as a function of such factors. The value of $\alpha$, however, in each environment where data is available, shows a relative constancy, at a value near unity.

It seems intuitive, all other things being equal, that $P$ should not increase as a program's size rises; however, the least-square power-law fits to data bases yield values of 0.91 (IBM, Ref. 3), 0.9091 (Doty, Ref. 1), 0.986 (University of Maryland, Ref. 4), and 0.975 (RAIDC, Ref. 16). One reason for the indicated increase in productivity for larger programs may be the increased usage of higher technology to develop these larger programs. Whatever the reason, the consistency of these figures seems to indicate that a linear power-law relationship, if anything, is slightly conservative.

The model presented here has compensations for the use of higher technology. The value for $\alpha$ assumed by this model, therefore, is unity. However, the model implementation keeps this value parametric, and $\alpha$ can be changed, if desired.

Several models have contributed to the formula by which $P_1$ is calculated, principally those of GRC (Ref. 17) and IBM (Ref. 3). The form of $P_1$ is

$$P_1 = P_0 A_1 A_2$$

(13)
where \( P_0 \) is a constant factor, and \( A_1 \) and \( A_2 \) are difficulty, technology, and environmental adjustment factors. The value of \( A_1 \) is computed on the basis of parameters judged by GRC to be significant,

\[
A_1 = [1 + A_{\text{lang}} + A_{t \text{crit}} + A_{c \text{crit}}] A_{\text{diff}} A_{\text{stable}} A_{\exp} ]^W \tag{14}
\]

The component adjustments are as follows:

(a) Language Adjustment, \( A_{\text{lang}} \):

\[
A_{\text{lang}} = f_{\text{lang}} L_{\text{asy}}
\]

where \( f_{\text{asy}} \) is an assembly language factor, and \( L_{\text{asy}} \) is the amount of assembly language used.

The DSN value, \( f_{\text{asy}} = 0.9 \), is derived in Section IIIF. GRC uses a value of \( f_{\text{asy}} = 3.5 \).

(b) Time-Critical-Code Adjustment, \( A_{t \text{crit}} \):

\[
A_{t \text{crit}} = f_{t \text{crit}} L_{t \text{crit}}
\]

where \( f_{t \text{crit}} \) is a factor which compensates for design/code/test time for code in which timing is critical, and \( L_{t \text{crit}} \) is the amount of such code. The DSN value, \( f_{t \text{crit}} = 0.7 \), is derived in Section IIIF. The GRC value is 1.0.

(c) Capacity-Criticality Adjustment, \( A_{c \text{crit}} \):

\[
A_{c \text{crit}} = f_{c \text{crit}} L_{c \text{crit}}
\]

where \( f_{c \text{crit}} \) is a factor which compensates for capacity-constrained portions of the program, and \( L_{c \text{crit}} \) is the amount of code which contributes to the need for extra effort in solving the capacity problem. The DSN value, \( f_{c \text{crit}} = 1.0 \), is derived in Section IIIF. The GRC value is 7.

(d) Difficulty Adjustment, \( A_{\text{diff}} \):

\[
A_{\text{diff}} = 1 + f_{\text{hard}} L_{\text{hard}} + f_{\text{easy}} L_{\text{easy}}
\]

where \( f_{\text{hard}} \) and \( f_{\text{easy}} \) are factors which relate to the relative effort required for "hard" and "easy" parts of the program, and \( L_{\text{hard}} \) and \( L_{\text{easy}} \) are the amounts of code adjudged to be "hard" and "easy," respectively. The DSN values, \( f_{\text{hard}} = 1.2 \) and \( f_{\text{easy}} = 0.8 \), are taken from data published in TRW reports (Ref. 6).

(c) Requirements and Design Stability Adjustment, \( A_{\text{stable}} \):

\[
A_{\text{stable}} = l_{\text{none}} + f_{\text{base}} l_{\text{base}} + f_{\text{many}} l_{\text{many}}
\]

\[
f_{\text{free}} = \frac{l_{\text{free}}}{L_{\text{free}}}
\]

Here, \( l_{\text{none}} \) is the amount of code which is expected to derive from well-understood, stable requirements, not expected to change. Factors \( f_{\text{base}} \) and \( l_{\text{base}} \) relate to effort and amount of code resulting from requirements expected to change moderately, but which are kept under baseline control. Factors \( f_{\text{many}} \) and \( l_{\text{many}} \) derive from the effort and amount of code for which requirements are expected to produce many changes, but, again, will do so under baseline control. The parameters \( f_{\text{free}} \) and \( l_{\text{free}} \) compensate for the accommodation of requirements allowed to change freely, not baseline-controlled. The values used in the DSN model, \( f_{\text{base}} = 1.35 \), \( f_{\text{many}} = 1.0 \), and \( f_{\text{free}} = 2.3 \), are taken directly from the GRC model.

(f) Experience Adjustment, \( A_{\exp} \):

\[
A_{\exp} = 1 - f_{\exp} T_{\exp} \text{ if } T_{\exp} < \frac{1 - f_{\text{fullup}}}{f_{\exp}}
\]

\[
= f_{\text{fullup}} \text{ if } T_{\exp} > \frac{1 - f_{\text{fullup}}}{f_{\exp}}
\]

Here \( T_{\exp} \) is the average staff training in years, \( f_{\exp} \) is the training rate, and \( f_{\text{fullup}} \) is the fully trained productivity factor. The values \( f_{\exp} = 0.06 \) and \( f_{\text{fullup}} = 0.4 \) derive from the GRC model.

The second productivity adjustment factor, \( A_2 \), derives from the Walston-Felix data (Ref. 3) published by IBM:

\[
A_2 = \prod_{i}^{N} \frac{P_{h_{i}(\cdot)}}{P_{l_{i}(\cdot)}} \exp \left[ w \sum_{i}^{N} x_{i} \log \frac{P_{h_{i}(\cdot)}}{P_{l_{i}(\cdot)}} \right]
\]

in which \( N \) factors that correlate with productivity and their effects on productivity were considered. \( P_{h_{i}(\cdot)} \) values were measured when factor \( i \) was applied and contributed to productivity, and \( P_{l_{i}(\cdot)} \) resulted when factor \( i \) was not applied and thus lowered productivity. The use of \( x_{i} \) relates to the
extent factor \( f \) is present in the application being estimated. The values used for \( x \) are either +1, 0, or -1. The parameter \( v \) was chosen to best fit the data collected.

The various factors contributing to the adjustment, the acceptable responses, and the log-ratio values used in the DSN model were taken, for the most part, directly from Walston-Felix data. A few others have been added, due to their variability in the DSN environment. The exponent value \( w \) was chosen to give a 50:1 variation between the extreme values.

The factors, responses, and log-ratio values used in the DSN model are listed in Appendix A. The value for \( v \) is computed by

$$
W = \frac{v_{w_f}}{\sum_{l=1}^{M} \log \frac{P_{h(t)}}{P_{l(t)}}}
$$

in which \( v_{w_f} = 1.05 \) produces the 50:1 spread in productivity adjustment. The value \( M \) extends the sum over both the IBM and GRC model factors.

IBM and other data show that the total effort adjustment may be expected to deviate, at a given \( z \), from the estimated value by a standard deviation factor of about \( \sigma_{\text{model}} = 1.73 \).

C. Estimation of Implementation Task Duration

The IBM, University of Maryland, and RADC statistics suggest that the average duration \( T \) for a KSLEC and \( w \) staff-months effort is approximated by

$$
T = T_1 w^f
$$

where \( T_1 \) is the 1-KSLEC average duration, and \( f \) is a time-factor exponent found from industry statistics. The value used in the DSN model, \( T_1 = 4.8 \), was adjusted to fit limited available DSN data, and \( f \approx 0.356 \) was the average power-law for the more extensive IBM, RADC, and GRC data.

D. Average Staff

The average staff, in persons, results from manipulation of the duration equation,

$$
S = \frac{W}{T} = \frac{1}{T_1} \frac{v}{f_I}
$$

The staffing exponent, \( 1 - f_I = 0.644 \), implied in the DSN model compares with measured values averaging 0.629 across industry.

E. Documentation Sizing and Cost

IBM statistics showed a nearly linear relationship between pages of documentation and lines of code, whereas the University of Maryland measured almost a square-root relationship. DSN experience over six Mark-III Data System programs revealed an exponent about midway in between (0.83). A study of Software Specification Document (SSD) user needs (Ref. 18) recommended that this document be about 40-50 pages per KSLEC for programs in the 30 KSLEC vicinity. The formula used for the model for the number of pages of documentation is

$$
D_{pp} = D_1 L^{e_{doc}}
$$

The model uses \( D_1 = 90 \) and \( e_{doc} = 0.83 \) to match the DSN experience and SSD guidelines.

The documentation cost is found by a straight dollar-per-page rate; a figure of $30/page is used in the current model.

F. Computer Resources

IBM and TRW give statistical figures for computer time costs as functions of lines of code and total effort, and also as a fraction of total cost. The DSN, however, mostly has dedicated minicomputers for which operational costs are not assessed to the implementation task on a usage basis. TRW does, however, also estimate a linear relationship between CPU time required per machine code instruction of about \( c_1 = 25.2 \) CPU hours per thousand instructions.

The DSN model computes CPU resources as

$$
T_{cpu} = C_1 (L_{assy} + (L - L_{assy}) f_{HOL})^{c_{cpu}}
$$

The higher-order-language expansion factor, \( f_{HOL} = 2.4 \), used in the DSN model is the GRC figure, based on JOVIAL. The exponent value \( c_{cpu} = 0.96 \), given by Walston and Felix (who give dollar costs, rather than CPU time), is adopted to account for the general trend of CPU time with program size.

If CPU dollar cost is relevant, the model computes this at a straight dollar-per-CPU-hour figure (zero in the DSN model, but a parameter is available for other applications).

G. Variance Computation

We assume in this model that measurements of values of a quantity \( y \) for a given \( x \) satisfy a power-law formula of the form

$$
y = kx^v
$$
in which \( v \) is a known, fixed value, but \( k \) exhibits some statistical distribution about a best-power-law fit,

\[ y_0 = k_0 x^v \]  

(28)

The values of \( k_0 \) and \( v \) are chosen to minimize the logarithmic error over the observed data.

\[
\text{err} [\ln(y) | x] = \sum_{i=1}^{N} (\ln y_i - \ln y_0^2) 
\]

(29)

The variation in \( y \) is expressed as a \((X^v)\) factor to be applied to \( y_0 \) at a given \( x \).

If, in a particular instance, there is uncertainty about the value of \( x \), then the \( y \) variation is described by

\[
\text{var} (\ln(y)) = \epsilon \left( \ln \frac{k}{k_0} + v \ln \frac{x}{x_0} \right)^2
\]

(30)

where \( \ln(x_0) \) is the mean of \( \ln(x) \).

If \( k_2 \) and \( x_1 \) are defined as those values such that

\[
\ln^2 \frac{k_1}{k_0} = \epsilon \left( \ln^2 \frac{k}{k_0} \right) 
\]

\[
\ln^2 \frac{x_1}{x_0} = \epsilon \left( \ln^2 \frac{x}{x_0} \right)
\]

(31)

then we can write

\[
y = k_0 x^v \left( \frac{x}{k_2} \right)
\]

(32)

where \( k_2 \) is a standard deviation factor

\[
k_2 = \exp \left[ \left( \ln^2 \frac{k_1}{k_0} + v^2 \ln^2 \frac{x_1}{x_0} \right)^{1/2} \right]
\]

(33)

Factors such as this are applied to the model to each power law, using \( k_2/k_0 \) values measured from industry statistics, and \( x_1/x_0 \) values assumed to be one-sigma estimations of the \( x \)-parameter. The \( x \)-parameter in the model ultimately derives from the program length, \( L \), whose variance was addressed in Section IIA.

### H. Modified Rayleigh-Norden-Putnam Model

Guided by the SWAG-estimated values, the user of the model enters the effort, duration, and staff parameters intended for use. As a check on reasonableness, comparisons of these parameters with the Rayleigh-Norden-Putnam model are made. However, some adjustments in this model have been made in order to make it fit the average power-law variations described above.

The basic differential equation describing the average work effort, \( w \), is taken to be of the form

\[ w' = A\epsilon \left( K - w \right) \]

(34)

in which \( w = w(t) \) is the cumulative effort expended up to time \( t \) (months), \( K \) is the total life-cycle staff-months effort, \( r \) is an exponent that accounts for learning curve and sets the “pace” of the work, and \( A \) is a constant scaling time.

Norden and Putnam assume \( r = 1 \) (a linear learning curve); however, this value of \( r \) is not consistent with the observed power-law relationships between \( L \), \( T \), and \( W \).

The solution to the modified equation is

\[
w = K \left\{ 1 - \exp \left[ - \left( \frac{r}{r+1} \right) \left( \frac{t}{t_0} \right)^{r+1} \right] \right\}
\]

(35)

where the value \( A \) has been replaced by \( r/t_0^{r+1} \), in which \( t_0 \) is the value at which \( w' \) is maximum (time of maximum staffing).

We use \( T \) to denote the time of acceptance test completion and transfer to operations (in months), whence \( w(T) = W \). Then

\[
T/t_0 = \left[ \frac{r+1}{r} \ln \left( \frac{1}{1 - W} \right)^{1/(r+1)} \right]
\]

(36)

As in the Putnam work, a “difficulty index,” \( D \) may be defined as

\[
D = \frac{K r}{t_0 r+1}
\]

(37)

and a “difficulty gradient factor” can be similarly defined as

\[
\nabla D = \frac{D}{W}
\]

(38)
These reduce to the Putnam expressions when \( r = 1 \). Note: Putnam expresses these values using time units of years, so comparisons require a scale change,

\[
D_{\text{Put}} = 12^r D
\]

\( \nabla D_{\text{Put}} = 12^{r+1} \nabla D \) \hspace{1cm} (39)

The number of lines of code is taken to be of the same general form as Putnam's "software equation."

\[
L = c_p K^p t_0^q
\] \hspace{1cm} (40)

(Putnam takes \( p = 1/3, q = 4/3, \) a 1:4 relationship.) The values for \( p \) and \( q \) used in the DSN software cost model are chosen to satisfy both power-law productivity constraints and time-effort tradeoffs.

The ratio \( W/K \) is the implementation-effort/total-life-cycle-effort ratio, which Putnam estimated to be constant, at about 0.95, and NASA measurements (Ref. 19) found to be about 0.88. Such ratios set the proportionality between \( T \) and \( t_0 \) to produce the equations

\[
L = c_p W^p T^q = c_p W^{p+q} S^{-q} = c_p T^{p+q} S^p
\] \hspace{1cm} (41)

for an appropriately defined technology constant \( c_p \) related to the productivity coefficient \( P_1 \) evaluated by the model.

We may use the power-law formula for \( L \) and the definition of \( D \) to eliminate \( T \). When difficulty is held constant at a value \( D_0 \) corresponding to the power-law, or "average difficulty" of projects, we find that the following relationship must hold:

\[
\frac{1}{a} = p + \frac{q}{r+1}
\] \hspace{1cm} (42)

Use of the power-law expression for average \( T \) similarly produces the relationships

\[
f_t = \frac{1}{r+1}
\]

\[c_p = (P_0 A_1 A_2)^{-1/a} T_1^{-q}
\] \hspace{1cm} (43)

\[D_0 = r K/W \left( \frac{T}{t_0} \right)^{r+1} T_1^{-(r+1)}
\]

\[\nabla D_0 = (r + 1) \frac{T}{t_0} \frac{D_0}{2T}
\]

For a given \( L \), the ratio \( q/p \) sets the time vs effort tradeoff relationship. Use of the factor \( f = q/p = -\log(W_1/W)/\log(T_1/T) \) permits \( p \) and \( q \) to be expressed as

\[
p = \frac{1}{a(1 + f_p)}
\]

\[
q = \frac{f}{a(1 + f_p)}
\] \hspace{1cm} (44)

The model contains an input parameter \( W_1/W = W_{1/2} \), defined to be the effort ratio for which \( T_{1/2}^T = T \). This parameter is used to specify the value of \( f \) as \( f = \log_2(W_{1/2}) \).

The equation for \( L \) is thereby

\[
L = (P_0 A_1 A_2)^{-1/a} \left( \frac{D_0^{q/(r+1)}}{D} \right)^{W^{1/a}}
\] \hspace{1cm} (45)

and the expression for productivity \( P \) as

\[
P = (P_0 A_1 A_2)^{-1/a} \left( \frac{D_0^{q/(r+1)}}{D} \right)^{W^{1/a}-1}
\] \hspace{1cm} (46)

If \( f = q/p \) were set to the Putnam value (viz., 4) then a value \( W_{1/2} = 16 \) would result. While the best value of \( f \) for DSN tasks has not been determined, the consensus of managers and programmers is that a 16:1 effort factor to reduce duration by a factor of 2 is too extreme, and not within their experience. Rather, a \( W_{1/2} = 1.5 \) value is probably closer to actuality in the DSN environment. This assumption produces the values

\[
f = 0.585
\]

\[
p = 0.828
\] \hspace{1cm} (47)

\[
r = 0.484
\]

\[
c_p = 0.0621/(A_1 A_2)
\]

The value \( W/K = 0.88 \) mentioned earlier produces

\[
\frac{T}{T_1} = 1.53
\]

\[
D_0 = 0.788, \text{ or } D_{\text{Put}} = 7.06
\] \hspace{1cm} (48)

\[
\nabla D_0 = 0.169/T, \text{ or } \nabla D_{\text{Put}} = 182/T
\]
Difficulty and gradient calculations are not currently used in the model, however.

When \( L \) and \( W \) are given, \( T \) and \( S \) are then given by

\[
T = \left( \frac{L}{c_p W^p} \right)^{1/p} \tag{49}
\]

\[
S = \frac{W}{T}
\]

When \( L \) and \( T \) are given, \( W \) and \( S \) are

\[
W = \left( \frac{L}{c_p T^p} \right)^{1/p} \tag{50}
\]

\[
S = \frac{W}{T}
\]

and when \( L \) and \( S \) are given, \( W \) and \( T \) are

\[
W = \left( \frac{L S^q}{c_p} \right)^{1/(p+q)} \tag{51}
\]

\[
T = \frac{W}{S}
\]

That is, only one of the average \( W, T, S \) parameters need be specified for a given \( L \). The others can then be computed (and are in the model).

If any pair of the \( W, T, S \) parameters is proposed, the lines-of-code value determined from the “software equation” represents the average number of lines of code that can be supported by the proposed resources. The difference between the size-model estimate and the software equation estimate is a margin on which confidence levels may be calculated.

One should note, however, that the software equation does not remain valid for tasks so small that average staff drops below unity. In this case, we revert back to the power-law equation to yield the needed effort and solve for \( T \) and \( S \) from the relation \( W = T S \).

Confidence in completing the software task within resources and budget when staff is variable is

\[
C(W, T) = P \{ t \leq T, w \leq W \} \tag{52}
\]

where \( P(\cdot) \) is the probability function. That is, the confidence level is the probability that both effort \( W \) and duration \( T \) are not exceeded.

Under the presumptions that the conditional densities \( p(T \mid W, L) \) and \( p(W \mid L) \) are log-normal (which are fairly well borne out by RADC data, see Ref. 16), and that \( p(L) \) can be estimated by a log-normal density as well, then the confidence factor is readily calculated. The log of the software equation provides the conditional mean for \( \log(T) \) in the first; the log of the power-law formula provides the conditional mean for \( \log(W) \) in the second; and the code size estimator provides the mean of \( \log(L) \) in the third. The result is the integral

\[
C(W, T) = \frac{1}{2\pi} \int_{-\infty}^{\infty} \left[ 1 + \text{erf} \left( \frac{u + \nu x}{\sqrt{2q}} \right) \right] \exp(-x^2) \, dx \tag{53}
\]

in which

\[
X = \frac{\ln \left( \frac{T}{T} \right)}{s_T} \frac{1/2}{1/2}
\]

\[
u = \frac{p s_w / q s_T}{(Q)^{1/2}}
\]

\[
Y = 1 + \frac{s_t^2}{p^2 s_w^2}
\]

\[
Q = Y + \frac{s_s^2}{q^2 s_T^2}
\]

\[
Z = Q + \frac{p^2 s_w^2}{q^2 s_T^2}
\]

The values \( t \) and \( w \) are duration and effort values on the software equation curve producing the expected number of lines of code; \( s_t, s_w, s_T, s_T \) represent the standard deviation in \( \log(L), \log(W), \) and \( \log(T) \), respectively.
The \( C(W; T) \) appears to be a function of arbitrary parameters \( t, w \) on the curve \( I_{\text{ref}} = c_n w^p t^q \). The triple, \((I_{\text{ref}}, w, t)\), represents the point from which margins are measured to \((L; W; T)\), the resource values to be used in the project. The model chooses \( w, t \) to maximize \( C(W; T) \). The optimum values of \( w \) and \( t \) satisfy the equation

\[
\left\{ 1 + c_n (R_f + G_f) \right\} \exp \left( \frac{A^2 n^2}{2} \right)
\]

\[
= H \left[ 1 + c_n (J_f + K_f) \right] \exp \left( \frac{A^2 n^2}{2} \right)
\]

where

\[
x = \frac{\ln (T_f)}{s_f} \]

\[
y = \frac{\ln (W_f)}{s_w} \]

\[
A = \left( \frac{1}{J} \right)^{1/2} \]

\[
F = \left( \frac{Y}{Z} \right)^{1/2} \]

\[
G = \frac{ps_w}{qs_r} Y^{1/2} \]

\[
H = \frac{Y q s w}{qs_r Y^{1/2}} \]

\[
J = \frac{ps w}{qs r} Y^{1/2} \]

\[
K = \frac{C}{Z} \]

Equation (55) is solved by Newton's method, subject to the constraint \( (J, H, t) = (p_s w, q_s r, v) \) and the confidence integral is evaluated by numeric integration.

I. Standard Work Breakdown Structure Mode

The DSN Data Systems Section has adopted a standard WBS format for its software implementation tasks. The format of subtasks is shown in Table 1. The table also shows the average reported effort in 31 recent upgrade tasks for each line item, adjusted for future tasks based on an analysis of the problems encountered. Requirements and subsystem design resources were not reported with the subsystem implementation activities; however, estimates for these have been factored into the WBS model.

The model permits selection of the starting point for WBS generation: at the System/Subsystem Functional Requirements (FRD) commencement; at the beginning of the Software Requirements Definition (SRD); or at the beginning of the Software Design Definition (SDD), at which time the software architectural considerations are developed.

Once a starting point has been selected by the user, the model apportions staff effort over the remaining subtasks (using percentage parameters in the standard WBS data base), and calculates subtask durations (also via percentage figures in the data base) to conform with the selected overall effort duration. These percentages are currently chosen to produce a relatively constant effort profile over the implementation project. This profile is a trend in the small DSN efforts currently under way. The percentages are alterable, however, to fit any assumed profile for other tasks.

The standard WBS data base also contains precedence information among subtasks, which is used by the PERT/CPM scheduling algorithm. Such precedences were set by logical and management criteria to form a rational progression of activities and milestones commensurate with DSN standard software implementation practices. Precedences for each task are specified in the form of the decimalized number of the task(s) which must complete before that particular task may begin.

All items in the standard WBS data base can be altered to fit another project by a mere text editing process.

J. Operational Modes

The DSN Software Cost Model software has a size/productivity estimation mode, which is optional; if estimates of staff effort, average staff size, and task duration are known a priori, the estimation mode may be skipped.

Regardless of the means by which amount of code staff effort, average staffing, and project duration are estimated, these figures are used to calculate a confidence level. The user may adjust estimates, as desired, until a suitable set of resource parameters is obtained before proceeding to the schedule generation process.
### K. Risk-Bias Computation

The size and resource estimates produced by the model are mean-estimate figures; however, variance figures are also given for risk-bias computations, as appropriate. If a schedule is generated using the mean estimates given, then the confidence factor for performing the implementation on time and within budget is only about 25%. That is, only about one-quarter of the projects using the mean estimates should expect to deliver within schedule and budget. If a project desires a confidence factor higher than 25%, then resource estimates must be increased by some amount to produce the higher confidence level.

Output is offered on hard copy or computer files, at user selection. Schedules are output beginning with FRD, SRD, or SDD activity.

### III. Selection and Calibration of Parameters

This section contains a data base of statistics and computations used in the formulation of the DSN Software Cost Model.

#### A. Productivity Statistics

The following formulas represent best-fit parametric models of observed data on KSLEC vs effort. In most cases, the unit of measure was not actually KSLEC, but something probably linearly related, such as delivered source lines (including comments) or object instructions.

\[
\ln(l) = 5.2 L^{0.91} \quad \text{(IBM)}
\]

\[
\ln(l) = 1.27 L^{0.986} \quad \text{(University of Maryland)}
\]
Fig. 3. Industry estimates of effort vs source lines of code delivered

\[ W = 10.1 L^{0.79} \]  
\[ W = 4.86 L^{0.976} \]  
\[ W = 5.25 L^{1.057} \]

(GRC, high-order languages)  
(RADC)  
(Doty)

These are shown in Fig. 3.

The figures asterisked (*) are computed from other measured statistics by the same company listed. For example, if project duration was given as a function of effort \( W \), then the staffing could be computed by \( S = W / T \) for comparison to the best-fit data. These staffing best-fit data are plotted in Fig. 4.

The exponent for \( L \) in the DSN model was chosen as unity for the following reasons: (1) the proximity of all the measured exponent values to unity, (2) the wide scatter of points in each case, and (3) the intuitive feeling that productivity cannot rise, all other things being the same, when the program size increases.

B. Staff Size Statistics

Staffing level industry statistics yielded least variation among best-fit models. Some of the results are

\[ S = 0.54 W^{0.6} \]  
\[ = 0.409 W^{0.65} \]  
\[ S = 0.198 W^{0.786} \]  
\[ = 0.196 W^{0.79} \]  
\[ S = 0.989 L^{0.609} \]  
\[ = 0.369 W^{0.624} \]  
\[ = 0.388 W^{0.641} \]

(IBM)  
(*)  
(University of Maryland)  
(*)  
(RADC)  
(*)

Fig. 4. Average staffing required vs project effort
The DSN model computes $S$ from effort $W$ and duration $T$ directly,

$$S = \frac{W}{T^p}$$

C. Project Duration Statistics

Duration statistics seem to produce slightly different effort exponents across the industry estimates, but seem to converge in the 100 staff-months effort region. The fits are

$$T = 2.47 W^{0.35} \quad \text{(IBM)}$$
$$= 2.13 W^{0.4} \quad \text{(*)}$$

$$T = 5.10 W^{0.2} \quad \text{(University of Maryland)}$$
$$= 5.19 W^{0.206} \quad \text{(*)}$$

$$T = 4.55 L^{0.349} \quad \text{(RADE)}$$
$$= 2.59 L^{0.358} \quad \text{(*)}$$

As in the previous section, the figures marked by asterisk (*) are not direct measurements, but computed from direct-measurement power-law fits. These duration data are presented in Fig. 5.

E. High-Level Language Adjustment

The statistics for usage of high-level languages vs assembly language are widely diverse in their effects on project effort. GRC reports (Ref. 15) best-fit curves to measured statistics of

$$W' = 9.38 L^{0.79} \quad \text{(assembly language sources)}$$
$$W' = 10.12 L^{0.79} \quad \text{(high-order language source*)}$$

The (*) statistic is derived from the measured ratio of object instructions to high-level source statements. These two equations suggest nearly a 1:1 ratio of effort on a source line basis. However, GRC also proposes a 4.5:1 cost factor in their Military Sales Software Cost Model (Ref. 17). Therefore, an analysis similar to that of Section 3.4 was undertaken to estimate the high-level vs assembly-language tradeoff. The assumptions were:

(a) Design, coding and unit testing activities would increase proportionately to the code-expansion ratio of the language. A value of 2.4:1 (GRC) was used.

(b) Documentation of the code would also increase in the same ratio, to keep the same relative degree of detail (pages/KSLEC).
Table 2. Inherited code effort requirement estimates

<table>
<thead>
<tr>
<th>Activity</th>
<th>New module</th>
<th>Added code</th>
<th>Deleted code</th>
<th>Changed code</th>
<th>Same code</th>
<th>Deleted module</th>
<th>Retested code</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Requirement and design</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FRD phase</td>
<td>6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FDD phase</td>
<td>7</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SDD phase</td>
<td>10</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SSD phase</td>
<td>13</td>
<td>13</td>
<td>4</td>
<td>13</td>
<td>4</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>SIS activity</td>
<td>10</td>
<td>10</td>
<td>3</td>
<td>10</td>
<td>3</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>SSD phase</td>
<td>22</td>
<td>22</td>
<td>5</td>
<td>6</td>
<td>5</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>Special</td>
<td>6</td>
<td>6</td>
<td></td>
<td>3</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Other</td>
<td>3</td>
<td>3</td>
<td></td>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Coding</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SSD phase</td>
<td>23</td>
<td>23</td>
<td>3</td>
<td>7</td>
<td>3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Special</td>
<td>6</td>
<td>6</td>
<td></td>
<td>3</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Other</td>
<td>4</td>
<td>4</td>
<td></td>
<td>2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Testing</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Integration</td>
<td>12</td>
<td>12</td>
<td></td>
<td>12</td>
<td>12</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>Verification</td>
<td>12</td>
<td>12</td>
<td></td>
<td>12</td>
<td>12</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>Contingency</td>
<td>2</td>
<td>2</td>
<td></td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>Acceptance test</td>
<td>10</td>
<td>10</td>
<td></td>
<td>10</td>
<td>10</td>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td>Demonstration</td>
<td>10</td>
<td>10</td>
<td></td>
<td>10</td>
<td>10</td>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td><strong>Documentation</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FRD</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FDD</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SRD</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SSD</td>
<td>3</td>
<td>3</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SSD</td>
<td>31</td>
<td>31</td>
<td>4</td>
<td>10</td>
<td>4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SOM</td>
<td>3</td>
<td>3</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SS1</td>
<td>6</td>
<td>6</td>
<td></td>
<td>3</td>
<td>1</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>STT</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Management</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FRD review</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FDD review</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SRD review</td>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Management plan</td>
<td>7</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SDD review</td>
<td>3</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>High level design review</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Acceptance review</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Transfer</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>CDF SSD phase</td>
<td>9</td>
<td>9</td>
<td>2</td>
<td>9</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>CDF test phase</td>
<td>3</td>
<td>3</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td>242</td>
<td>194</td>
<td>29</td>
<td>123</td>
<td>65</td>
<td>26</td>
<td>41</td>
</tr>
<tr>
<td>Factor</td>
<td>1.0</td>
<td>0.80</td>
<td>0.12</td>
<td>0.51</td>
<td>0.27</td>
<td>0.11</td>
<td>0.17</td>
</tr>
</tbody>
</table>

(c) Acceptance tests, being functional tests, would be the same for the two. The reduced number of functions to be tested per KSLEC of assembly code as compared to the higher-order language was felt to be negated by the larger number of errors likely to be found.

(d) Requirements and architectural design tend to be top-level activities depending on numbers of functions.
Capacity constraints can often be relieved by program optimization or segmentation, when mass storage resources are available, but timing considerations then also sometimes become a factor, and TRW estimates a 2.5:1 factor in effort for such timing constraints. GRC concurs in this figure. TRW also estimates a 3:1 increase in effort for real-time programs over sequential programs.

Inasmuch as capacity and timing constraints often require coding in assembly language, it was felt that the three were correlated in the same data and that the effort factor of all three taken simultaneously was overcompensation. The DSN productivity on the Mark III Data System (10 source lines/day in assembly language, with real-time and severe capacity constraints) tended to favor this hypothesis.

Tables 4 and 5 are estimated resource budgets for capacity- and timing-constrained developments. Since these factors directly affect code design, coding, and testing, the emphasis of the extra effort is applied in these subtasks.

G. DSN Documentation Statistics

The documentation page count of six Mark III Data System tasks representing over 100 KSLEC produced data presented in Table 6. These data gave best-fit power-law statistics of

\[ D_{SRD} = 6.29 L^{0.499} \quad (X/1.50) \]
\[ D_{SDD} = 21.3 L^{0.314} \quad (X/1.63) \]
\[ D_{SOM} = 15.3 L^{0.727} \quad (X/1.21) \]
\[ D_{SSD} = 136 L^{0.832} \quad (X/1.21) \]
\[ D_{SSF} = 7.71 L^{1.09} \quad (X/1.61) \]
\[ D_{tot} = 175 L^{0.829} \quad (X/1.21) \]

It was generally felt, however, that the SRD and SDD documentation represented by these figures was not detailed enough, whereas the SSD was too detailed in some areas, and yet not detailed enough in others. A study (Ref. 18) recommended about 46 SSD pages/KSLEC (without listings), down about 43% from the average of 80 pages/KSLEC actually produced for the SSDs, or down about 33% from the total.

The power law curves above were therefore adjusted up 10% for the SRD and SDD, and down by about 45% for the SSD to arrive at the recommended figure

\[ D_{tot} = 120 L^{0.823} \quad (X/1.2) \]
Table 4. Capacity constraint effort requirement estimates

<table>
<thead>
<tr>
<th>Activity</th>
<th>FRD start</th>
<th>SRD start</th>
<th>SDD start</th>
</tr>
</thead>
<tbody>
<tr>
<td>Requirements and design</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FRD phase</td>
<td>6</td>
<td></td>
<td></td>
</tr>
<tr>
<td>FDD phase</td>
<td>7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SRD phase</td>
<td>10</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>SSD phase</td>
<td>x 2</td>
<td>26</td>
<td>26</td>
</tr>
<tr>
<td>SIS activity</td>
<td>x 2</td>
<td>20</td>
<td>20</td>
</tr>
<tr>
<td>SSD phase</td>
<td>x 5</td>
<td>110</td>
<td>110</td>
</tr>
<tr>
<td>Special</td>
<td>x 4</td>
<td>24</td>
<td>24</td>
</tr>
<tr>
<td>Other</td>
<td>x 4</td>
<td>12</td>
<td>12</td>
</tr>
<tr>
<td>Coding</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SSD phase</td>
<td>x 2</td>
<td>46</td>
<td>46</td>
</tr>
<tr>
<td>Special</td>
<td>x 2</td>
<td>12</td>
<td>12</td>
</tr>
<tr>
<td>Other</td>
<td>x 2</td>
<td>8</td>
<td>8</td>
</tr>
<tr>
<td>Testing</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Integration</td>
<td>x 2</td>
<td>24</td>
<td>24</td>
</tr>
<tr>
<td>Verification</td>
<td>x 2</td>
<td>24</td>
<td>24</td>
</tr>
<tr>
<td>Contingency</td>
<td>x 2</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>Acceptance test</td>
<td>x 2</td>
<td>20</td>
<td>20</td>
</tr>
<tr>
<td>Demonstration</td>
<td>x 2</td>
<td>20</td>
<td>20</td>
</tr>
<tr>
<td>Documentation</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FRD</td>
<td>3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>FDD</td>
<td>3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SRD</td>
<td>3</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>SSD</td>
<td>3</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>SSD</td>
<td>31</td>
<td>31</td>
<td>31</td>
</tr>
<tr>
<td>SOM</td>
<td>3</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>STT</td>
<td>6</td>
<td>6</td>
<td>6</td>
</tr>
<tr>
<td>Management</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FRD review</td>
<td>2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>FDD review</td>
<td>2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SRD review</td>
<td>5</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>Management plan</td>
<td>7</td>
<td>7</td>
<td>7</td>
</tr>
<tr>
<td>SDD review</td>
<td>3</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>High level design review</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Acceptance review</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Transfer</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>CDF SSD phase</td>
<td>x 1.3</td>
<td>12</td>
<td>12</td>
</tr>
<tr>
<td>CDF test phase</td>
<td>x 1.3</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>Total</td>
<td>463</td>
<td>440</td>
<td>422</td>
</tr>
<tr>
<td>Factor</td>
<td>1.91</td>
<td>2.01</td>
<td>2.10</td>
</tr>
</tbody>
</table>

Table 5. Time critical module effort estimates

<table>
<thead>
<tr>
<th>Activity</th>
<th>FRD start</th>
<th>SRD start</th>
<th>SDD start</th>
</tr>
</thead>
<tbody>
<tr>
<td>Requirements and design</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FRD phase</td>
<td>6</td>
<td></td>
<td></td>
</tr>
<tr>
<td>FDD phase</td>
<td>7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SRD phase</td>
<td>10</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>SSD phase</td>
<td>x 1.5</td>
<td>20</td>
<td>20</td>
</tr>
<tr>
<td>SIS activity</td>
<td>x 1.5</td>
<td>15</td>
<td>15</td>
</tr>
<tr>
<td>SSD phase</td>
<td>x 3</td>
<td>66</td>
<td>66</td>
</tr>
<tr>
<td>Special</td>
<td>x 3</td>
<td>18</td>
<td>18</td>
</tr>
<tr>
<td>Other</td>
<td>x 4</td>
<td>9</td>
<td>9</td>
</tr>
<tr>
<td>Coding</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SSD phase</td>
<td>x 2</td>
<td>46</td>
<td>46</td>
</tr>
<tr>
<td>Special</td>
<td>x 2</td>
<td>12</td>
<td>12</td>
</tr>
<tr>
<td>Other</td>
<td>x 2</td>
<td>8</td>
<td>8</td>
</tr>
<tr>
<td>Testing</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Integration</td>
<td>x 2</td>
<td>24</td>
<td>24</td>
</tr>
<tr>
<td>Verification</td>
<td>x 2</td>
<td>24</td>
<td>24</td>
</tr>
<tr>
<td>Contingency</td>
<td>x 2</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>Acceptance test</td>
<td>x 2</td>
<td>20</td>
<td>20</td>
</tr>
<tr>
<td>Demonstration</td>
<td>x 2</td>
<td>20</td>
<td>20</td>
</tr>
<tr>
<td>Documentation</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FRD</td>
<td>3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>FDD</td>
<td>3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SRD</td>
<td>3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SDD</td>
<td>3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SSD</td>
<td>31</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SOM</td>
<td>3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>STT</td>
<td>6</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Management</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FRD review</td>
<td>2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>FDD review</td>
<td>2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SRD review</td>
<td>5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Management plan</td>
<td>7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SDD review</td>
<td>3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>High level design review</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Acceptance review</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Transfer</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CDF SSD phase</td>
<td>x 1.3</td>
<td>12</td>
<td>12</td>
</tr>
<tr>
<td>CDF test phase</td>
<td>x 1.3</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>Total</td>
<td>399</td>
<td>376</td>
<td>358</td>
</tr>
<tr>
<td>Factor</td>
<td>1.649</td>
<td>1.72</td>
<td>1.78</td>
</tr>
</tbody>
</table>
Table 6. DSN documentation statistics

<table>
<thead>
<tr>
<th>Subsys stem</th>
<th>KSIC</th>
<th>SRD</th>
<th>SDD</th>
<th>SSD</th>
<th>SOM</th>
<th>SIT</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPA OP I A</td>
<td>274</td>
<td>63</td>
<td>128</td>
<td>1974</td>
<td>193</td>
<td>431</td>
<td>2784</td>
</tr>
<tr>
<td>CPA OP I</td>
<td>493</td>
<td>43</td>
<td>58</td>
<td>3678</td>
<td>236</td>
<td>736</td>
<td>4748</td>
</tr>
<tr>
<td>MDA OP D</td>
<td>151</td>
<td>22</td>
<td>33</td>
<td>919</td>
<td>85</td>
<td>53</td>
<td>1112</td>
</tr>
<tr>
<td>DST OP C</td>
<td>37</td>
<td>18</td>
<td>36</td>
<td>380</td>
<td>36</td>
<td>52</td>
<td>524</td>
</tr>
<tr>
<td>MON OP C</td>
<td>122</td>
<td>11</td>
<td>73</td>
<td>1318</td>
<td>128</td>
<td>162</td>
<td>1692</td>
</tr>
<tr>
<td>CMI OP D</td>
<td>118</td>
<td>17</td>
<td>23</td>
<td>1340</td>
<td>95</td>
<td>66</td>
<td>1541</td>
</tr>
</tbody>
</table>

The SRD, SDD, and SSD recommended best-fit sizes are:

- $D_{SRD} = 7.1 / 0.492 \ (x/1.5)$
- $D_{SDD} = 23.4 / 0.313 \ (x/1.6)$
- $D_{SSD} = 78.4 / 0.831 \ (x/1.2)$
- $D_{SOM} = 15.3 / 0.727 \ (x/1.2)$
- $D_{SIT} = 7.71 / 1.09 \ (x/1.8)$

IV. Typical Example of the Model Operation

The software prototype implementation of the DSN Software Cost Model was programmed for the JPL Automated Office Data Center/Terminal Work Station (AODC/TWS, Ref. 20), which utilizes a Z-80 CPU, 64K-bytes RAM, diskette storage, and a CP/M-compatible operating system.

The user merely enters the program name (SOFTCOST) and answers prompted questions. A blank form is available (see Fig. 6) for data entry by an operator, should the user so desire. The example on the completed form shown in Fig. 7 produces the outputs shown in Figs. 8 through 12. Outputs are selectable: all or none can be generated.

V. Summary and Conclusion

The Software Cost Model reported here is the first of a series of refinements. As the model is used and as performance data are collected, no doubt changes will be made: adjustments of parameters, alterations of formulas, modifications of formats, added and deleted input requirements, additional kinds of outputs, and so forth. One extension currently envisioned is the automated transfer of the WBS data base generated by the model into the WBS-based project control system currently used in the DSN Data Systems implementation tasks, by means of networking the software model host computer with the project control system host.

If the model presented seems complex, it is justly so, for the factors which affect human performance are generally complex and unpredictable, except in statistical terms. One sample function chosen from a stochastic ensemble is hardly ever "average" or "typical". One must expect variations between actual behavior and predictions by the model.

The directions for the future are to refine the model for greater accuracy (within the human performance estimation capacity limits), to extend the utility of the model throughout the entire software life cycle, and to provide the basis for indicating needed new computer technology, software methodology, and tools.
Answer the following items to the best of your estimation.

1. How much new code is to be produced (completely new modules)?
   - Maximum value, kilo-lines executable source (99% confidence level)?
   - Expected value, kilo-lines executable source?
   - Minimum value, kilo-lines executable source (99% confidence level)?

2. How much code exists in modules requiring modification?
   - Maximum value, kilo-lines executable source (99% confidence level)?
   - Expected value, kilo-lines executable source?
   - Minimum value, kilo-lines executable source (99% confidence level)?

3. How much code will be deleted from these existing modules?
   - Maximum value, kilo-lines executable source (99% confidence level)?
   - Expected value, kilo-lines executable source?
   - Minimum value, kilo-lines executable source (99% confidence level)?

4. How much code will be added to these existing modules?
   - Maximum value, kilo-lines executable source (99% confidence level)?
   - Expected value, kilo-lines executable source?
   - Minimum value, kilo-lines executable source (99% confidence level)?

5. How much code will be changed in other ways in these modules?
   - Maximum value, kilo-lines executable source (99% confidence level)?
   - Expected value, kilo-lines executable source?
   - Minimum value, kilo-lines executable source (99% confidence level)?

6. How much code will be deleted as entire modules from existing code?
   - Maximum value, kilo-lines executable source (99% confidence level)?
   - Expected value, kilo-lines executable source?
   - Minimum value, kilo-lines executable source (99% confidence level)?

7. How much of the remaining existing code must be retested?
   - Maximum value, kilo-lines executable source (99% confidence level)?
   - Expected value, kilo-lines executable source?
   - Minimum value, kilo-lines executable source (99% confidence level)?

8. Expected percentage of code to be developed actually delivered (0-90, 91-99, 100)?

9. How many different kinds of input/output data items per 1000 lines of new or modified code (>80, 16-80, 0-15)?

10. Overall complexity of program and database architecture (high, medium, low)?

11. Complexity of code logical design (high, medium, low)?

12. What percent of the programming task is in Assembly language?

13. What percent of the new or modified code must be storage-optimized?

Fig. 6. Blank form for software cost model inputs
<table>
<thead>
<tr>
<th>Question</th>
<th>Answer</th>
</tr>
</thead>
<tbody>
<tr>
<td>14. What percent of the new or modified code must be timing-optimized?</td>
<td></td>
</tr>
<tr>
<td>15. What percent of the total programming task is 'easy'?</td>
<td></td>
</tr>
<tr>
<td>16. What percent of the total programming task is 'hard'?</td>
<td></td>
</tr>
<tr>
<td>17. When is work to start, on the (FRD/FDD, SRD, SDD)?</td>
<td></td>
</tr>
<tr>
<td>18. What percent of the total program requirements are well established, stable, and will not be altered before delivery?</td>
<td></td>
</tr>
<tr>
<td>19. What percent of the requirements are likely to change slightly before delivery, but will do so under baseline change control?</td>
<td></td>
</tr>
<tr>
<td>20. What percent of the requirements are likely to change more drastically before delivery, but will do so under baseline control?</td>
<td></td>
</tr>
<tr>
<td>21. Complexity of program functional requirements (high, medium, low)?</td>
<td></td>
</tr>
<tr>
<td>22. Expected user involvement in requirements definition (much, some, none)?</td>
<td></td>
</tr>
<tr>
<td>23. Customer experience in application area (much, none, some)?</td>
<td></td>
</tr>
<tr>
<td>24. Customer/implementor organizational interface complexity (high, normal, low)?</td>
<td></td>
</tr>
<tr>
<td>25. Interfaces with other SW development projects or organizations (many, few, none)?</td>
<td></td>
</tr>
<tr>
<td>26. Efficiency of implementing organization (poor, ok, good)?</td>
<td></td>
</tr>
<tr>
<td>27. Overall implementation personnel qualifications and motivation (low, average, high)?</td>
<td></td>
</tr>
<tr>
<td>28. Percentage of programmers doing functional design who will also be doing development (&lt;25, 25-50, &gt;50)?</td>
<td></td>
</tr>
<tr>
<td>29. Previous programmer experience with application of similar or greater size and complexity (minimal, average, extensive)?</td>
<td></td>
</tr>
<tr>
<td>30. What is the average staff experience, in years, obtained from work similar to that required in the task being estimated?</td>
<td></td>
</tr>
<tr>
<td>31. Previous experience with operational computer to be used (minimal, average, extensive)?</td>
<td></td>
</tr>
<tr>
<td>32. Previous experience with programming language(s) to be used (minimal, average, extensive)?</td>
<td></td>
</tr>
<tr>
<td>33. Use of top-down methodology (low, medium, high)?</td>
<td></td>
</tr>
<tr>
<td>34. Use of structured programmer team concepts (low, medium, high)?</td>
<td></td>
</tr>
<tr>
<td>35. Use of Structured Programming (low, medium, high)?</td>
<td></td>
</tr>
</tbody>
</table>

Fig. 6 (contd)
36. Use of design and code inspections (low, QA, peer)? 

37. Classified security environment for computer (yes, no)? 

38. Hardware under concurrent development (much, some, none)? 

39. Percent of work done at primary development site (<70, 70-90, >90)? 

40. Development computer access mode (remote, scheduled, demand)? 

41. Percent of development computer access availability (<30, 30-60, >60)? 

42. Quality of SW development tools and environment (poor, ok, good)? 

43. Maturity of system and support software (buggy, ok, good)? 

44. Overall adverse constraints on program design (severe, average, minimal)? 

45. Is the program real-time, multi-task (chiefly, some, no)? 

46. SW to be adaptable to multiple computer configurations or environments (yes, no)? 

47. Adaptation required to change from development to operational environment (much, some, minimal)? 

Values to be used for planning are: 
- Kilo-lines of code: 
- Effort (person-months): 
- Duration (months): 
- Average staff (persons): 

Is output to be saved in a file? 

Name of output file to be created: 

Schedule start date: 

Select desired outputs and output media, or enter RETURN only for defaults. Defaults are 1A, 2A, and 3A. Choices are: 

1 = Gantt Chart 
2 = PERT data, 132 width 
3 = PERT data, 80 width 

Choice(s):
### Answer the following items to the best of your estimation.

1. **How much new code is to be produced (completely new modules)?**
   - Maximum value, kilo-lines executable source (99% confidence level): 5.5
   - Expected value, kilo-lines executable source: 3.3
   - Minimum value, kilo-lines executable source (99% confidence level): 1.1

2. **How much code exists in modules requiring modification?**
   - Maximum value, kilo-lines executable source (99% confidence level): 6.9
   - Expected value, kilo-lines executable source: 4.3
   - Minimum value, kilo-lines executable source (99% confidence level): 1.3

3. **How much code will be deleted from these existing modules?**
   - Maximum value, kilo-lines executable source (99% confidence level): 4.4
   - Expected value, kilo-lines executable source: 2.2
   - Minimum value, kilo-lines executable source (99% confidence level): 0.2

4. **How much code will be added to these existing modules?**
   - Maximum value, kilo-lines executable source (99% confidence level): 7.7
   - Expected value, kilo-lines executable source: 5.5
   - Minimum value, kilo-lines executable source (99% confidence level): 3.3

5. **How much code will be changed in other ways in these modules?**
   - Maximum value, kilo-lines executable source (99% confidence level): 1.2
   - Expected value, kilo-lines executable source: 0.8
   - Minimum value, kilo-lines executable source (99% confidence level): 0.2

6. **How much code will be deleted as entire modules from existing code?**
   - Maximum value, kilo-lines executable source (99% confidence level): 1.4
   - Expected value, kilo-lines executable source: 1.3
   - Minimum value, kilo-lines executable source (99% confidence level): 1.1

7. **How much of the remaining existing code must be retested?**
   - Maximum value, kilo-lines executable source (99% confidence level): 2.1
   - Expected value, kilo-lines executable source: 1.9
   - Minimum value, kilo-lines executable source (99% confidence level): 1.7

8. **Expected percentage of code to be developed actually delivered (0-90, 91-99, 100)?**
   - 91-99

9. **How many different kinds of input/output data items per 1000 lines of new or modified code (≥60, 16-80, 0-15)?**
   - 16-80

10. **Overall complexity of program and data base architecture (high, medium, low)?**
    - Med

11. **Complexity of code logical design (high, medium, low)?**
    - Low

12. **What percent of the programming task is in Assembly language?**
    - 9

13. **What percent of the new or modified code must be storage-optimized?**
    - 9
14. What percent of the new or modified code must be timing-optimized? 9
15. What percent of the total programming task is 'easy'? 20
16. What percent of the total programming task is 'hard'? 80
17. When is work to start, on the (FRD/FDD, SRD, SDD)?
18. What percent of the total program requirements are well established, stable, and will not be altered before delivery? 80
19. What percent of the requirements are likely to change slightly before delivery, but will do so under baseline change control? 10
20. What percent of the requirements are likely to change more drastically before delivery, but will do so under baseline control? 5
21. Complexity of program functional requirements (high, medium, low)? low
22. Expected user involvement in requirements definition (much, some, none)? much
23. Customer experience in application area (much, none, some)? some
24. Customer/implementor organizational interface complexity (high, normal, low)? normal
25. Interfaces with other SW development projects or organizations (many, few, none)? few
26. Efficiency of implementing organization (poor, ok, good)? good
27. Overall implementation personnel qualifications and motivation (low, average, high)? high
28. Percentage of programmers doing functional design who will also be doing development (<25, 25-50, >50)? 25-50
29. Previous programmer experience with application of similar or greater size and complexity (minimal, average, extensive)? average
30. What is the average staff experience, in years, obtained from work similar to that required in the task being estimated? 6
31. Previous experience with operational computer to be used (minimal, average, extensive)?
32. Previous experience with programming language(s) to be used (minimal, average, extensive)?
33. Use of top-down methodology (low, medium, high)? high
34. Use of structured programmer team concepts (low, medium, high)? high
35. Use of Structured Programming (low, medium, high)? high

Fig. 7 (contd)
<table>
<thead>
<tr>
<th>Question</th>
<th>Answer</th>
</tr>
</thead>
<tbody>
<tr>
<td>36. Use of design and code inspections (low, QA, peer)?</td>
<td>QA</td>
</tr>
<tr>
<td>37. Classified security environment for computer (yes, no)?</td>
<td>no</td>
</tr>
<tr>
<td>38. Hardware under concurrent development (much, some, none)?</td>
<td>none</td>
</tr>
<tr>
<td>39. Percent of work done at primary development site (&lt;70, 70-90, &gt;90)?</td>
<td>70-90</td>
</tr>
<tr>
<td>40. Development computer access mode (remote, scheduled, demand)?</td>
<td>demand</td>
</tr>
<tr>
<td>41. Percent of development computer access availability (&lt;30, 30-60, &gt;60)?</td>
<td>30-60</td>
</tr>
<tr>
<td>42. Quality of SW development tools and environment (poor, ok, good)?</td>
<td>ok</td>
</tr>
<tr>
<td>43. Maturity of system and support software (buggy, ok, good)?</td>
<td>ok</td>
</tr>
<tr>
<td>44. Overall adverse constraints on program design (severe, average, minimal)?</td>
<td>min</td>
</tr>
<tr>
<td>45. Is the program real-time, multi-task (chiefly, some, no)?</td>
<td>no</td>
</tr>
<tr>
<td>46. SW to be adaptable to multiple computer configurations or environments (yes, no)?</td>
<td>no</td>
</tr>
<tr>
<td>47. Adaptation required to change from development to operational environment (much, some, minimal)?</td>
<td>min</td>
</tr>
</tbody>
</table>

Values to be used for planning are:
- Kilo-lines of code: 
- Effort (person-months): 16 months
- Average staff (persons): 2

Is output to be saved in a file? yes
Name of output file to be created: VC_EDIT
Schedule start date: 17 NOV 80

Select desired outputs and output media, or enter RETURN only for defaults. Defaults are 1A, 2A, and 3A. Choices are:
- 1 = Gantt Chart
- 2 = PERT data, 132 width
- 3 = PERT data, 80 width

Choice(s): 1B, 2B, 3A

Fig. 7 (contd)
1. How much new code is to be produced (completely new modules)?
   - Maximum value, kilo-lines executable source (99% confidence level): 3.5
   - Expected value, kilo-lines executable source: 3.3
   - Minimum value, kilo-lines executable source (99% confidence level): 3.1

2. How much code exists in modules requiring modification?
   - Maximum value, kilo-lines executable source (99% confidence level): 6.9
   - Expected value, kilo-lines executable source: 6.6
   - Minimum value, kilo-lines executable source (99% confidence level): 6.3

3. How much code will be deleted from these existing modules?
   - Maximum value, kilo-lines executable source (99% confidence level): .4
   - Expected value, kilo-lines executable source: .3
   - Minimum value, kilo-lines executable source (99% confidence level): .2

4. How much code will be added to these existing modules?
   - Maximum value, kilo-lines executable source (99% confidence level): .7
   - Expected value, kilo-lines executable source: .6
   - Minimum value, kilo-lines executable source (99% confidence level): .4

5. How much code will be changed in other ways in these modules?
   - Maximum value, kilo-lines executable source (99% confidence level): 1.2
   - Expected value, kilo-lines executable source: .9
   - Minimum value, kilo-lines executable source (99% confidence level): .7

6. How much code will be deleted as entire modules from existing code?
   - Maximum value, kilo-lines executable source (99% confidence level): 1.4
   - Expected value, kilo-lines executable source: 1.3
   - Minimum value, kilo-lines executable source (99% confidence level): 1.1

7. How much of the remaining existing code must be retested?
   - Maximum value, kilo-lines executable source (99% confidence level): 2.1
   - Expected value, kilo-lines executable source: 1.9
   - Minimum value, kilo-lines executable source (99% confidence level): 1.5

8. Expected percentage of code to be developed actually delivered (0-90, 91-99, 100)?
   - 91-99

9. How many different kinds of input/output data items per 1000 lines of new or modified code (>80, 16-80, 0-15)?
   - 16-80

10. Overall complexity of program and data base architecture (high, medium, low)?
    - MEDIUM

11. Complexity of code logical design (high, medium, low)?
    - LOW

12. What percent of the programming task is in Assembly language?
    - 9

13. What percent of the new or modified code must be storage-optimized?
    - 9

Fig. 8. Hard copy format input parameter record
<table>
<thead>
<tr>
<th></th>
<th>Question</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>14</td>
<td>What percent of the new or modified code must be timing-optimized?</td>
<td>9</td>
</tr>
<tr>
<td>15</td>
<td>What percent of the total programming task is 'easy'?</td>
<td>20</td>
</tr>
<tr>
<td>16</td>
<td>What percent of the total programming task is 'hard'?</td>
<td>30</td>
</tr>
<tr>
<td>17</td>
<td>When is work to start, on the (FRD/PDD, SRD, SDD)?</td>
<td>FRD/PDD</td>
</tr>
<tr>
<td>18</td>
<td>What percent of the total program requirements will be established and stable before design, and will not be altered before delivery?</td>
<td>80</td>
</tr>
<tr>
<td>19</td>
<td>What percent of the requirements are likely to change slightly before delivery, but will do so under baseline change control?</td>
<td>10</td>
</tr>
<tr>
<td>20</td>
<td>What percent of the requirements are likely to change more drastically before delivery, but will do so under baseline control?</td>
<td>5</td>
</tr>
<tr>
<td>21</td>
<td>Complexity of program functional requirements (high, medium, low)?</td>
<td>LOW</td>
</tr>
<tr>
<td>22</td>
<td>Expected user involvement in requirements definition (much, some, none)?</td>
<td>MUCH</td>
</tr>
<tr>
<td>23</td>
<td>Customer experience in application area (much, none, some)?</td>
<td>SOME</td>
</tr>
<tr>
<td>24</td>
<td>Customer/implementor organizational interface complexity (high, normal, low)?</td>
<td>NORMAL</td>
</tr>
<tr>
<td>25</td>
<td>Interfaces with other SW development projects or organizations (many, few, none)?</td>
<td>FEW</td>
</tr>
<tr>
<td>26</td>
<td>Efficiency of implementing organization (poor, ok, good)?</td>
<td>GOOD</td>
</tr>
<tr>
<td>27</td>
<td>Overall implementation personnel qualifications and motivation (low, average, high)?</td>
<td>HIGH</td>
</tr>
<tr>
<td>28</td>
<td>Percentage of programmers doing functional design who will also be doing development (&lt;25, 25-50, &gt;50)?</td>
<td>25-50</td>
</tr>
<tr>
<td>29</td>
<td>Previous programmer experience with application of similar or greater size and complexity (minimal, average, extensive)?</td>
<td>AVERAGE</td>
</tr>
<tr>
<td>30</td>
<td>What is the average staff experience, in years, obtained from work similar to that required in the task being estimated?</td>
<td>6</td>
</tr>
<tr>
<td>31</td>
<td>Previous experience with operational computer to be used (minimal, average, extensive)?</td>
<td>MINIMAL</td>
</tr>
<tr>
<td>32</td>
<td>Previous experience with programming language(s) to be used (minimal, average, extensive)?</td>
<td>MINIMAL</td>
</tr>
<tr>
<td>33</td>
<td>Use of top-down methodology (low, medium, high)?</td>
<td>HIGH</td>
</tr>
<tr>
<td>34</td>
<td>Use of structured programmer team concepts (low, medium, high)?</td>
<td>HIGH</td>
</tr>
<tr>
<td>35</td>
<td>Use of Structured Programming (low, medium, high)?</td>
<td>HIGH</td>
</tr>
</tbody>
</table>

Fig. 8 (contd)
<table>
<thead>
<tr>
<th>Question</th>
<th>Answer</th>
</tr>
</thead>
<tbody>
<tr>
<td>36. Use of design and code inspections (low, QA, peer)?</td>
<td>QA</td>
</tr>
<tr>
<td>37. Classified security environment for computer (yes, no)?</td>
<td>NO</td>
</tr>
<tr>
<td>38. Hardware under concurrent development (much, some, none)?</td>
<td>NONE</td>
</tr>
<tr>
<td>39. Percent of work done at primary development site (70-90)?</td>
<td>70-90</td>
</tr>
<tr>
<td>40. Development computer access mode (remote, scheduled, demand)?</td>
<td>DEMAND</td>
</tr>
<tr>
<td>41. Percent of development computer access availability (&lt;30, 30-60, &gt;60)?</td>
<td>30-60</td>
</tr>
<tr>
<td>42. Quality of SW development tools and environment (poor, ok, good)?</td>
<td>OK</td>
</tr>
<tr>
<td>43. Maturity of system and support software (buggy, ok, good)?</td>
<td>OK</td>
</tr>
<tr>
<td>44. Overall adverse constraints on program design (severe, average, minimal)?</td>
<td>MINIMAL</td>
</tr>
<tr>
<td>45. Is the program real-time, multi-task (chiefly, some, no)?</td>
<td>NO</td>
</tr>
<tr>
<td>46. SW to be adaptable to multiple computer configurations or environments (yes, no)?</td>
<td>NO</td>
</tr>
<tr>
<td>47. Adaptation required to change from development to operational environment (much, some, minimal)?</td>
<td>MINIMAL</td>
</tr>
</tbody>
</table>

Fig. 8 (contd)
Estimated Overall Parameters:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>+1-sigma</th>
<th>Average Value</th>
<th>-1-sigma</th>
</tr>
</thead>
<tbody>
<tr>
<td>Adjusted Lines of code</td>
<td>6182 SLEC</td>
<td>6182 SLEC</td>
<td>6182 SLEC</td>
</tr>
<tr>
<td>Effort</td>
<td>26.5 person-months</td>
<td>26.5 person-months</td>
<td>26.5 person-months</td>
</tr>
<tr>
<td>Staff productivity</td>
<td>233 SLEC/staff-month</td>
<td>233 SLEC/staff-month</td>
<td>233 SLEC/staff-month</td>
</tr>
<tr>
<td>Duration</td>
<td>15.7 months</td>
<td>15.7 months</td>
<td>15.7 months</td>
</tr>
<tr>
<td>Avg. Staff</td>
<td>1.7</td>
<td>1.7</td>
<td>1.7</td>
</tr>
<tr>
<td>Documentation</td>
<td>537 pages</td>
<td>$16.1K</td>
<td>$13.4K</td>
</tr>
<tr>
<td>Computer CPU time</td>
<td>319 hours</td>
<td>$0.0K</td>
<td>$0.0K</td>
</tr>
</tbody>
</table>

Use these figures to arrive at Effort, Duration, and Staffing requirements. Include factors to provide acceptable risk and confidence levels.

Values specified are:

Kilo-lines of code: 6.18
Effort (person-months): 32.0
Duration (months): 16.0
Average staff (persons): 2.0

For the numbers you have entered, a reasonableness check indicates that the average project would produce 7303 lines of code, using 32 staff-months of resources and 16 months of duration, with an average staff of 2 persons, for a productivity of 228 SLEC/staff-month.

The level of confidence in delivering 6182 lines of code, on-time and within resources= 33 %.

Is output to be saved in a file? Y
Name of output file to be created: VCEDIT
Schedule start date: 17NOV80

Select desired outputs and output media, or enter RETURN only for defaults. Defaults are 1A, 2A, and 3A. Choices are:

1=Gantt Chart
2=PERT data, 132 width
3=PERT data, 80 width

Choice(s): 1B, 2B, 3A

Fig. 9. Output of SWAG estimator using the "typical software project" parameters
<table>
<thead>
<tr>
<th>CODE</th>
<th>TASK</th>
<th>WHO</th>
<th>ORT</th>
<th>EARLY</th>
<th>LATE</th>
<th>EFF</th>
<th>START</th>
<th>FINISH</th>
<th>FLOAT</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.4.4</td>
<td>Function A</td>
<td></td>
<td></td>
<td>18.169</td>
<td>23JUL81</td>
<td>169</td>
<td>23JUL81</td>
<td>186</td>
<td>17AUG81</td>
</tr>
<tr>
<td>4.4.5</td>
<td>Function B</td>
<td></td>
<td></td>
<td>18.186</td>
<td>17AUG81</td>
<td>186</td>
<td>17AUG81</td>
<td>203</td>
<td>11SEP81</td>
</tr>
<tr>
<td>4.4.6</td>
<td>Function C</td>
<td></td>
<td></td>
<td>17.203</td>
<td>11SEP81</td>
<td>203</td>
<td>11SEP81</td>
<td>220</td>
<td>6OCT81</td>
</tr>
<tr>
<td>4.4.7</td>
<td>Function D</td>
<td></td>
<td></td>
<td>17.220</td>
<td>6OCT81</td>
<td>220</td>
<td>6OCT81</td>
<td>227</td>
<td>29OCT81</td>
</tr>
<tr>
<td>4.4.8</td>
<td>Function E</td>
<td></td>
<td></td>
<td>17.237</td>
<td>29OCT81</td>
<td>237</td>
<td>29OCT81</td>
<td>254</td>
<td>23NOV81</td>
</tr>
<tr>
<td>4.4.9</td>
<td>Function F</td>
<td></td>
<td></td>
<td>17.254</td>
<td>23NOV81</td>
<td>254</td>
<td>23NOV81</td>
<td>271</td>
<td>18DEC81</td>
</tr>
<tr>
<td>4.5</td>
<td>Special Tasks</td>
<td></td>
<td></td>
<td>0.139</td>
<td>10JUN81</td>
<td>271</td>
<td>18DEC81</td>
<td>139</td>
<td>10JUN81</td>
</tr>
<tr>
<td>4.5.1</td>
<td>Support software</td>
<td></td>
<td></td>
<td>12.133</td>
<td>2JUN81</td>
<td>133</td>
<td>2JUN81</td>
<td>139</td>
<td>10JUN81</td>
</tr>
<tr>
<td>4.5.2</td>
<td>Other</td>
<td></td>
<td></td>
<td>6.133</td>
<td>2JUN81</td>
<td>133</td>
<td>2JUN81</td>
<td>136</td>
<td>5JUN81</td>
</tr>
<tr>
<td>4.6</td>
<td>Acceptance Readiness Rw</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>271</td>
<td>18DEC81</td>
</tr>
<tr>
<td>5.1</td>
<td>SW Test and Transf</td>
<td></td>
<td></td>
<td>0.332</td>
<td>18MAR82</td>
<td>332</td>
<td>18MAR82</td>
<td>332</td>
<td>18MAR82</td>
</tr>
<tr>
<td>5.2</td>
<td>Contingency</td>
<td></td>
<td></td>
<td>28.273</td>
<td>2DEC81</td>
<td>273</td>
<td>2DEC81</td>
<td>299</td>
<td>1FEB82</td>
</tr>
<tr>
<td>5.3</td>
<td>STT</td>
<td></td>
<td></td>
<td>25.96</td>
<td>9APR81</td>
<td>314</td>
<td>22FEB82</td>
<td>109</td>
<td>28APR81</td>
</tr>
<tr>
<td>5.3.1</td>
<td>Write all sections</td>
<td></td>
<td></td>
<td>0.313</td>
<td>19FEB82</td>
<td>332</td>
<td>16MAR82</td>
<td>313</td>
<td>19FEB82</td>
</tr>
<tr>
<td>5.3.2</td>
<td>Edit and release</td>
<td></td>
<td></td>
<td>2.312</td>
<td>1FEB82</td>
<td>356</td>
<td>10MAR82</td>
<td>313</td>
<td>19FEB82</td>
</tr>
<tr>
<td>5.4</td>
<td>Acceptance tests</td>
<td></td>
<td></td>
<td>22.312</td>
<td>1FEB82</td>
<td>312</td>
<td>18FEB82</td>
<td>312</td>
<td>18FEB82</td>
</tr>
<tr>
<td>5.5</td>
<td>Demonstration tests</td>
<td></td>
<td></td>
<td>22.312</td>
<td>1FEB82</td>
<td>312</td>
<td>18FEB82</td>
<td>312</td>
<td>18FEB82</td>
</tr>
<tr>
<td>5.6</td>
<td>Transfer, CDE to COE</td>
<td></td>
<td></td>
<td>27.327</td>
<td>11MAR82</td>
<td>327</td>
<td>11MAR82</td>
<td>327</td>
<td>11MAR82</td>
</tr>
<tr>
<td>6.1</td>
<td>Mgt Tasks and Milestones</td>
<td></td>
<td></td>
<td>0.19</td>
<td>16DEC80</td>
<td>332</td>
<td>18MAR82</td>
<td>19</td>
<td>16DEC80</td>
</tr>
<tr>
<td>6.2</td>
<td>CDE Activities</td>
<td></td>
<td></td>
<td>37.0</td>
<td>17NOV80</td>
<td>313</td>
<td>19FEB82</td>
<td>19</td>
<td>16DEC80</td>
</tr>
<tr>
<td>6.3</td>
<td>Develop Sys Impl Plan</td>
<td></td>
<td></td>
<td>3.10</td>
<td>3DEC80</td>
<td>10</td>
<td>3DEC80</td>
<td>12</td>
<td>5DEC80</td>
</tr>
<tr>
<td>6.4</td>
<td>Draft Software Impl Plan</td>
<td></td>
<td></td>
<td>6.50</td>
<td>4FEB81</td>
<td>50</td>
<td>4FEB81</td>
<td>55</td>
<td>11FEB81</td>
</tr>
<tr>
<td>6.5</td>
<td>Revise Impl Plan</td>
<td></td>
<td></td>
<td>12.81</td>
<td>19MAR81</td>
<td>81</td>
<td>19MAR81</td>
<td>90</td>
<td>1APR81</td>
</tr>
<tr>
<td>6.6</td>
<td>QA Audit</td>
<td></td>
<td></td>
<td>26.312</td>
<td>1FEB82</td>
<td>318</td>
<td>26FEB82</td>
<td>326</td>
<td>10MAR82</td>
</tr>
</tbody>
</table>

**Fig. 10 (contd)**
<table>
<thead>
<tr>
<th>CODE</th>
<th>TASK</th>
<th>WHO</th>
<th>EFF</th>
<th>E-START</th>
<th>L-FINISH</th>
<th>FLT</th>
</tr>
</thead>
<tbody>
<tr>
<td>*0.</td>
<td>START</td>
<td></td>
<td>0</td>
<td>17NOV80</td>
<td>17NOV80</td>
<td>0</td>
</tr>
<tr>
<td>*1.</td>
<td>Sys Plans, Reqs, &amp; Design</td>
<td></td>
<td></td>
<td>9JAN81</td>
<td>9JAN81</td>
<td>0</td>
</tr>
<tr>
<td>*1.1</td>
<td>Define Subsys Regts</td>
<td>15</td>
<td>17NOV80</td>
<td>3DEC80</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>*1.2</td>
<td>FRD</td>
<td></td>
<td>0</td>
<td>8DEC80</td>
<td>8DEC80</td>
<td>0</td>
</tr>
<tr>
<td>1.2.1</td>
<td>Write all sections</td>
<td>4</td>
<td>17NOV80</td>
<td>5DEC80</td>
<td>9</td>
<td></td>
</tr>
<tr>
<td>1.2.2</td>
<td>Edit and release FRD</td>
<td>2</td>
<td>5DEC80</td>
<td>8DEC80</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>1.3</td>
<td>Level B Review</td>
<td>3</td>
<td>8DEC80</td>
<td>10DEC80</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>1.4</td>
<td>Define Sys Architecture</td>
<td>18</td>
<td>10DEC80</td>
<td>31DEC80</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>*1.5</td>
<td>FDD</td>
<td></td>
<td>0</td>
<td>7JAN81</td>
<td>7JAN81</td>
<td>0</td>
</tr>
<tr>
<td>1.5.1</td>
<td>Write all sections</td>
<td>4</td>
<td>10DEC80</td>
<td>6JAN81</td>
<td>12</td>
<td></td>
</tr>
<tr>
<td>1.5.2</td>
<td>Edit and release</td>
<td>2</td>
<td>6JAN81</td>
<td>7JAN81</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>1.6</td>
<td>Level C Review</td>
<td>3</td>
<td>7JAN81</td>
<td>9JAN81</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>*2.</td>
<td>SW Planning and Reqs</td>
<td>0</td>
<td>16FEB81</td>
<td>16FEB81</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>*2.1</td>
<td>Define Software Reqs</td>
<td>25</td>
<td>9JAN81</td>
<td>4FEB81</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>*2.2</td>
<td>SRD</td>
<td></td>
<td>0</td>
<td>12FEB81</td>
<td>12FEB81</td>
<td>0</td>
</tr>
<tr>
<td>2.2.1</td>
<td>Write all sections</td>
<td>4</td>
<td>9JAN81</td>
<td>11FEB81</td>
<td>21</td>
<td></td>
</tr>
<tr>
<td>2.2.2</td>
<td>Edit and release</td>
<td>2</td>
<td>11FEB81</td>
<td>12FEB81</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>2.3</td>
<td>Level D Review</td>
<td>2</td>
<td>12FEB81</td>
<td>16FEB81</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>*3.</td>
<td>SW Architecture and Design</td>
<td>0</td>
<td>6APR81</td>
<td>6APR81</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>*3.1</td>
<td>Define SW architecture</td>
<td>31</td>
<td>16FEB81</td>
<td>19MAR81</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>*3.2</td>
<td>SDD</td>
<td></td>
<td>0</td>
<td>2APR81</td>
<td>2APR81</td>
<td>0</td>
</tr>
<tr>
<td>3.2.1</td>
<td>Write all sections</td>
<td>4</td>
<td>16FEB81</td>
<td>1APR81</td>
<td>30</td>
<td></td>
</tr>
<tr>
<td>3.2.2</td>
<td>Edit and release</td>
<td>2</td>
<td>1APR81</td>
<td>2APR81</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>3.3</td>
<td>System Interface Design</td>
<td>22</td>
<td>16FEB81</td>
<td>1APR81</td>
<td>20</td>
<td></td>
</tr>
<tr>
<td>3.4</td>
<td>Level E Review</td>
<td>2</td>
<td>2APR81</td>
<td>6APR81</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>*4.</td>
<td>SW Detailed Design &amp; Prod</td>
<td>0</td>
<td>22DEC81</td>
<td>22DEC81</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>4.1</td>
<td>SSD</td>
<td></td>
<td>0</td>
<td>23FEB82</td>
<td>1IMR82</td>
<td>12</td>
</tr>
<tr>
<td>4.1.1</td>
<td>Write Sections 1,2,3</td>
<td>6</td>
<td>6APR81</td>
<td>18DEC81</td>
<td>175</td>
<td></td>
</tr>
<tr>
<td>4.1.2</td>
<td>Write Section 4</td>
<td>18</td>
<td>6APR81</td>
<td>14APR81</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>4.1.3</td>
<td>Write Section 5</td>
<td>43</td>
<td>24APR81</td>
<td>18DEC81</td>
<td>142</td>
<td></td>
</tr>
<tr>
<td>4.1.4</td>
<td>Write Section 6</td>
<td>4</td>
<td>6APR81</td>
<td>18DEC81</td>
<td>176</td>
<td></td>
</tr>
<tr>
<td>4.1.5</td>
<td>Write Section 7</td>
<td>9</td>
<td>6APR81</td>
<td>18DEC81</td>
<td>173</td>
<td></td>
</tr>
<tr>
<td>4.1.6</td>
<td>Edit and release</td>
<td>6</td>
<td>16FEB82</td>
<td>1IMR82</td>
<td>12</td>
<td></td>
</tr>
<tr>
<td>4.2</td>
<td>SOM</td>
<td></td>
<td>0</td>
<td>22FEB82</td>
<td>1IMR82</td>
<td>13</td>
</tr>
<tr>
<td>4.2.1</td>
<td>Write preliminary draft</td>
<td>8</td>
<td>24APR81</td>
<td>4MAY81</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>4.2.2</td>
<td>Complete all sections</td>
<td>9</td>
<td>2JUN81</td>
<td>18DEC81</td>
<td>133</td>
<td></td>
</tr>
<tr>
<td>4.2.3</td>
<td>Edit and release</td>
<td>4</td>
<td>16FEB82</td>
<td>1IMR82</td>
<td>13</td>
<td></td>
</tr>
<tr>
<td>4.3</td>
<td>High-level Design Review</td>
<td>2</td>
<td>29MAY81</td>
<td>2JUN81</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>4.4</td>
<td>Module Production &amp; Integ</td>
<td>0</td>
<td>18DEC81</td>
<td>18DEC81</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>4.4.1</td>
<td>Executive and control</td>
<td>18</td>
<td>4MAY81</td>
<td>29MAY81</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>4.4.2</td>
<td>I/O Modules</td>
<td>18</td>
<td>2JUN81</td>
<td>26JUN81</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>4.4.3</td>
<td>Interface handlers</td>
<td>18</td>
<td>26JUN81</td>
<td>23JUL81</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>

Fig. 11. Short-form output of the PERT/CPM WBS schedule data
<table>
<thead>
<tr>
<th>CODE</th>
<th>TASK</th>
<th>WHO</th>
<th>EFF</th>
<th>E-START</th>
<th>L-FINISH</th>
<th>FLT</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.4.4</td>
<td>Function A</td>
<td>:</td>
<td>:</td>
<td>18</td>
<td>23JUL81</td>
<td>17AUG81</td>
</tr>
<tr>
<td>4.4.5</td>
<td>Function B</td>
<td>:</td>
<td>:</td>
<td>18</td>
<td>17AUG81</td>
<td>11SEP81</td>
</tr>
<tr>
<td>4.4.6</td>
<td>Function C</td>
<td>:</td>
<td>:</td>
<td>17</td>
<td>11SEP81</td>
<td>6OCT81</td>
</tr>
<tr>
<td>4.4.7</td>
<td>Function D</td>
<td>:</td>
<td>:</td>
<td>17</td>
<td>6OCT81</td>
<td>29OCT81</td>
</tr>
<tr>
<td>4.4.8</td>
<td>Function E</td>
<td>:</td>
<td>:</td>
<td>17</td>
<td>29OCT81</td>
<td>23NOV81</td>
</tr>
<tr>
<td>4.4.9</td>
<td>Function F</td>
<td>:</td>
<td>:</td>
<td>17</td>
<td>23NOV81</td>
<td>18DEC81</td>
</tr>
<tr>
<td>4.5</td>
<td>Special Tasks</td>
<td>:</td>
<td>:</td>
<td>0</td>
<td>10JUN81</td>
<td>18DEC81</td>
</tr>
<tr>
<td>4.5.1</td>
<td>Support software</td>
<td>:</td>
<td>:</td>
<td>12</td>
<td>2JUN81</td>
<td>18DEC81</td>
</tr>
<tr>
<td>4.5.2</td>
<td>Other</td>
<td>:</td>
<td>:</td>
<td>6</td>
<td>2JUN81</td>
<td>18DEC81</td>
</tr>
<tr>
<td>4.6</td>
<td>Acceptance Readiness Revw</td>
<td>:</td>
<td>:</td>
<td>2</td>
<td>18DEC81</td>
<td>22DEC81</td>
</tr>
<tr>
<td>5.1</td>
<td>SW Test and Transfer</td>
<td>:</td>
<td>:</td>
<td>28</td>
<td>22DEC81</td>
<td>1FEB82</td>
</tr>
<tr>
<td>5.2</td>
<td>Contingency</td>
<td>:</td>
<td>:</td>
<td>25</td>
<td>9APR81</td>
<td>11MAR82</td>
</tr>
<tr>
<td>5.3</td>
<td>STT</td>
<td>:</td>
<td>:</td>
<td>0</td>
<td>19FEB82</td>
<td>18MAR82</td>
</tr>
<tr>
<td>5.3.1</td>
<td>Write all sections</td>
<td>:</td>
<td>:</td>
<td>14</td>
<td>2JUN81</td>
<td>1FEB82</td>
</tr>
<tr>
<td>5.3.2</td>
<td>Edit and release</td>
<td>:</td>
<td>:</td>
<td>2</td>
<td>18FEB82</td>
<td>11MAR82</td>
</tr>
<tr>
<td>5.4</td>
<td>Acceptance tests</td>
<td>:</td>
<td>:</td>
<td>20</td>
<td>1FEB82</td>
<td>18MAR82</td>
</tr>
<tr>
<td>5.5</td>
<td>Demonstration tests</td>
<td>:</td>
<td>:</td>
<td>22</td>
<td>18FEB82</td>
<td>11MAR82</td>
</tr>
<tr>
<td>5.6</td>
<td>Transfer, CDE to COE</td>
<td>:</td>
<td>:</td>
<td>7</td>
<td>11MAR82</td>
<td>18MAR82</td>
</tr>
<tr>
<td>6.1</td>
<td>Mgt Tasks and Milestones</td>
<td>:</td>
<td>:</td>
<td>0</td>
<td>16DEC80</td>
<td>18MAR82</td>
</tr>
<tr>
<td>6.2</td>
<td>CDE Activities</td>
<td>:</td>
<td>:</td>
<td>37</td>
<td>17NOV80</td>
<td>18MAR82</td>
</tr>
<tr>
<td>6.3</td>
<td>Develop prelim budget</td>
<td>:</td>
<td>:</td>
<td>3</td>
<td>3DEC80</td>
<td>5DEC80</td>
</tr>
<tr>
<td>6.3</td>
<td>Develop Sys Impl Plan</td>
<td>:</td>
<td>:</td>
<td>3</td>
<td>31DEC80</td>
<td>6JAN81</td>
</tr>
<tr>
<td>6.4</td>
<td>Draft Software Impl Plan</td>
<td>:</td>
<td>:</td>
<td>6</td>
<td>4FEB81</td>
<td>11FEB81</td>
</tr>
<tr>
<td>6.5</td>
<td>Revise Impl Plan</td>
<td>:</td>
<td>:</td>
<td>12</td>
<td>19MAR81</td>
<td>1APR81</td>
</tr>
<tr>
<td>6.6</td>
<td>QA Audit</td>
<td>:</td>
<td>:</td>
<td>26</td>
<td>18FEB82</td>
<td>18MAR82</td>
</tr>
<tr>
<td>FINISH</td>
<td></td>
<td>:</td>
<td>:</td>
<td>0</td>
<td>18MAR82</td>
<td>18MAR82</td>
</tr>
</tbody>
</table>

Fig. 11 (contd)
<table>
<thead>
<tr>
<th>TASK</th>
<th>1980 DEC JAN FEB MAR APR</th>
<th>1981 MAY JUN JUL AUG SEP OCT NOV DEC JAN FEB MAR APR</th>
<th>MAN FINISH</th>
</tr>
</thead>
<tbody>
<tr>
<td>0. START</td>
<td>A</td>
<td></td>
<td>0:17FEB81</td>
</tr>
<tr>
<td>1. Sys Plan, Reqts, &amp; Design</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.1 Define Subsys Reqts</td>
<td>A</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.2 FPD</td>
<td>A</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.2.1 Write all sections</td>
<td>A</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.2.2 Edit and release</td>
<td>A</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.3 Level B Review</td>
<td>A</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.4 Define Sys Architecture</td>
<td>A</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.5 FPD</td>
<td>A</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.5.1 Write all sections</td>
<td>A</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.5.2 Edit and release</td>
<td>A</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.6 Level C Review</td>
<td>A</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2. SW Planning and Reqts</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2.1 Define Software Reqts</td>
<td>A</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2.2 SDD</td>
<td>A</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2.2.1 Write all sections</td>
<td>A</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2.2.2 Edit and release</td>
<td>A</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3. SW Architecture and Design</td>
<td>A</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3.1 Define SW architecture</td>
<td>A</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3.2 SDD</td>
<td>A</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3.2.1 Write all sections</td>
<td>A</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3.2.2 Edit and release</td>
<td>A</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3.3 System Interface Design</td>
<td>A</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3.4 Level E Review</td>
<td>A</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4. SW Detailed Design &amp; Proc</td>
<td>A</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4.1 SDD</td>
<td>A</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4.1.1 Write sections 1,2,3</td>
<td>A</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4.1.2 Write Section 4</td>
<td>A</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4.1.3 Write Section 5</td>
<td>A</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4.1.4 Write Section 6</td>
<td>A</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4.1.5 Write Section 7</td>
<td>A</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4.1.6 Edit and release</td>
<td>A</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4.2 SOM</td>
<td>A</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4.2.1 Write preliminary draft</td>
<td>A</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4.2.2 Complete all sections</td>
<td>A</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4.2.3 Edit and release</td>
<td>A</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4.3 High-level Design Review</td>
<td>A</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4.4 Module Production &amp; Integrate</td>
<td>A</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4.4.1 Test Modules</td>
<td>A</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4.4.2 I/O Modules</td>
<td>A</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4.4.3 Interface handlers</td>
<td>A</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**LEGEND:**
- **A** = ACTIVITY
- **E** = EARLY FINISH
- **L** = LATE START
- **F** = LATE FINISH
- **P** = EVENTS PAST
- **V** = ACTUAL FINISH
- **C** = CRITICAL PATH ITEM

**NOTE:** ABSENCE OF S, A, OR V INDICATES MERGED DATES

---

**Fig. 12.** Gantt chart output of the software cost model
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>4.4</td>
<td>Function A</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>16/17/5/61</td>
</tr>
<tr>
<td>4.4.5</td>
<td>Function B</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>18/19/5/61</td>
</tr>
<tr>
<td>4.4.6</td>
<td>Function C</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>17/16/6/61</td>
</tr>
<tr>
<td>4.4.7</td>
<td>Function D</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>17/28/5/61</td>
</tr>
<tr>
<td>4.4.8</td>
<td>Function E</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>17/23/5/61</td>
</tr>
<tr>
<td>4.4.9</td>
<td>Function F</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>17/18/5/61</td>
</tr>
<tr>
<td>4.5</td>
<td>Special Tasks</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>02/11/5/61</td>
</tr>
<tr>
<td>4.5.1</td>
<td>Support software</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>12/17/5/61</td>
</tr>
<tr>
<td>4.5.2</td>
<td>Other</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>17/18/5/61</td>
</tr>
<tr>
<td>4.6</td>
<td>Acceptance Readiness Rev</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>6/12/5/61</td>
</tr>
<tr>
<td>5.</td>
<td>SW Test and Transfer</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>01/12/5/61</td>
</tr>
<tr>
<td>5.1</td>
<td>Verification tests</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>28/15/5/61</td>
</tr>
<tr>
<td>5.2</td>
<td>Contingency</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>25/14/5/61</td>
</tr>
<tr>
<td>5.3</td>
<td>STT</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>12/14/5/61</td>
</tr>
<tr>
<td>5.3.1</td>
<td>Write all sections</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>14/11/5/61</td>
</tr>
<tr>
<td>5.3.2</td>
<td>Edit and release</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>02/12/5/61</td>
</tr>
<tr>
<td>5.4</td>
<td>Acceptance tests</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>20/12/5/61</td>
</tr>
<tr>
<td>5.5</td>
<td>Demonstration tests</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>22/11/5/61</td>
</tr>
<tr>
<td>5.6</td>
<td>Transfer, CDE to CDE</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>07/11/5/61</td>
</tr>
<tr>
<td>6.1</td>
<td>CDE Activities</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>37/15/5/61</td>
</tr>
<tr>
<td>6.2</td>
<td>Develop prelim budget</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>3/10/5/61</td>
</tr>
<tr>
<td>6.3</td>
<td>Develop Sys Impl Plan</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>03/11/5/61</td>
</tr>
<tr>
<td>6.4</td>
<td>Draft Software Impl Plan</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>06/11/5/61</td>
</tr>
<tr>
<td>6.5</td>
<td>Revise Impl Plan</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>12/10/5/61</td>
</tr>
<tr>
<td>6.6</td>
<td>QA Audit</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>26/11/5/61</td>
</tr>
</tbody>
</table>

**LEGEND:**
- **=-ACTIVITY**
- **A-EARLY FINISH**
- S-LATE START
- V-LATE FINISH
- <=EVENTS PAST
- V=ACTUAL FINISH
- *=CRITICAL PATH ITEM

**NOTE:** ABSENCE OF S, S, A, OR V INDICATES MERGED DATES

Fig. 12 (contd)
<table>
<thead>
<tr>
<th>TASK</th>
<th>1960</th>
<th>1981</th>
<th>1982</th>
<th>LATE FINISH</th>
</tr>
</thead>
<tbody>
<tr>
<td>*1.</td>
<td>Sys Plans, Reqs, &amp; Design</td>
<td>100</td>
<td>100</td>
<td>100</td>
</tr>
<tr>
<td>*2.</td>
<td>SW Planning and Peqs</td>
<td>100</td>
<td>100</td>
<td>100</td>
</tr>
<tr>
<td>*3.</td>
<td>SW Architecture and Design</td>
<td>100</td>
<td>100</td>
<td>100</td>
</tr>
<tr>
<td>*4.</td>
<td>SW Detailed Design &amp; Prod</td>
<td>100</td>
<td>100</td>
<td>100</td>
</tr>
<tr>
<td>*5.</td>
<td>SW Test and Transfer</td>
<td>100</td>
<td>100</td>
<td>100</td>
</tr>
<tr>
<td>*6.</td>
<td>Mgt Tasks and Milestones</td>
<td>100</td>
<td>100</td>
<td>100</td>
</tr>
</tbody>
</table>

LEGEND: **ACTIVITY** A- EARLY FINISH S- LATE START V- LATE FINISH < EVENTS PAST V- ACTUAL FINISH **- CRITICAL PATH ITEM**

NOTE: ABSENCE OF S, S, A, OR V INDICATES MERGED DATES

Fig. 12 (contd)
References


20. As of this writing, there are no externally referenceable documents describing the AODC. User Manuals are available, however, from the AODC User's Library, Information Systems Engineering Section, Jet Propulsion Laboratory, Pasadena, Calif.
### Appendix A
**DSN Software Cost Model Factors**

This Appendix contains a listing of the standard factor file accessed by the software model. A correspondence between factors on the file and those appearing in formulas herein appears below:

<table>
<thead>
<tr>
<th>FILE AND PROGRAM</th>
<th>REPORT SYMBOL</th>
</tr>
</thead>
<tbody>
<tr>
<td>NEWFAC</td>
<td>l (normalized)</td>
</tr>
<tr>
<td>CHMFAC</td>
<td>b</td>
</tr>
<tr>
<td>ADLFAC</td>
<td>c</td>
</tr>
<tr>
<td>CHLFAC</td>
<td>d</td>
</tr>
<tr>
<td>DLNFAC</td>
<td>e</td>
</tr>
<tr>
<td>DMDFAC</td>
<td>g</td>
</tr>
<tr>
<td>RTSFAC</td>
<td>h</td>
</tr>
<tr>
<td>LLEV</td>
<td>f_HOL</td>
</tr>
<tr>
<td>LADJ</td>
<td>f_assy</td>
</tr>
<tr>
<td>CADJ</td>
<td>f_c-crit</td>
</tr>
<tr>
<td>TADJ</td>
<td>f_t-crit</td>
</tr>
<tr>
<td>EADJ</td>
<td>f_easy</td>
</tr>
<tr>
<td>HADJ</td>
<td>f_hard</td>
</tr>
<tr>
<td>BADJ</td>
<td>f_base</td>
</tr>
<tr>
<td>MADJ</td>
<td>f_many</td>
</tr>
<tr>
<td>FADJ</td>
<td>f_free</td>
</tr>
</tbody>
</table>

* Factors in the file used by the model not given a specific symbol in this report.*
<table>
<thead>
<tr>
<th>FILE AND PROGRAM</th>
<th>REPORT SYMBOL</th>
</tr>
</thead>
<tbody>
<tr>
<td>SADJ</td>
<td>$f_{exp}$</td>
</tr>
<tr>
<td>FULLUP</td>
<td>$f_{full-up}$</td>
</tr>
<tr>
<td>KSLECPM</td>
<td>$P_1$</td>
</tr>
<tr>
<td>A</td>
<td>$a$</td>
</tr>
<tr>
<td>AWFFAC</td>
<td>$f_{wf}$</td>
</tr>
<tr>
<td>ELFAC</td>
<td>$A_{\sigma}$</td>
</tr>
<tr>
<td>DFAC</td>
<td>$T_1$</td>
</tr>
<tr>
<td>DEXP</td>
<td>$f_t$</td>
</tr>
<tr>
<td>T1FAC</td>
<td>*</td>
</tr>
<tr>
<td>PPKSLEC</td>
<td>$D_1$</td>
</tr>
<tr>
<td>AD</td>
<td>$C_{doc}$</td>
</tr>
<tr>
<td>PGCOST</td>
<td>*</td>
</tr>
<tr>
<td>D1FAC</td>
<td>*</td>
</tr>
<tr>
<td>CFAC</td>
<td>$C_1$</td>
</tr>
<tr>
<td>AC</td>
<td>$C_{cpu}$</td>
</tr>
<tr>
<td>C1FAC</td>
<td>*</td>
</tr>
<tr>
<td>GPRCOST</td>
<td>*</td>
</tr>
<tr>
<td>S1FAC</td>
<td>*</td>
</tr>
<tr>
<td>WTFAC</td>
<td>$W_{1/2}$</td>
</tr>
<tr>
<td>ATFAC</td>
<td>$W/K$</td>
</tr>
<tr>
<td>LOGRATI**</td>
<td>$-\log(P_{hi(i)}/P_{lo(i)})$</td>
</tr>
</tbody>
</table>

* Factors in the file used by the model not given a specific symbol in this report.

** The LOGRATIO designator does not appear in the file; however it is found as the numeric field following the 3 ANSWER entries.
DSN SOFTWARE COST MODEL STANDARD FACTOR FILE

| VERSION (VERS$) | "Version 1.3 31OCT80"
| WORK DAYS PER MO (DAYSPERMO) | 20
| HDR FIELD 1 (HFL1$) | "TITLE:"
| HDR FIELD 2 (HFL2$) | "CDE:"
| HDR FIELD 3 (HFL3$) | "ECR/ECO:"
| HDR FIELD 4 (HFL4$) | "PROG. ID:"
| HDR FIELD 5 (HFL5$) | "SUBSYS:"
| CLEAR SCREEN (CLSS) | .2,27,43
| NEW CODE FACTOR (NEWFAC) | 1
| CHG MODULE FAC (CHMFAC) | .27
| ADD LINES FAC (ADLFAC) | .80
| CHG LINES FAC (CHLFAC) | .24
| DEL LINES FAC (DLNLFAC) | -.15
| DEL MODULE FAC (DMDFAC) | .11
| RETEST FACTOR (RTSFAC) | .17
| LANGUAGE LEVEL (ILEV) | .24
| LANG ADJ FAC (LADJ) | .82
| CNP ADJ FAC (CNPADJ) | .101
| TIME ADJ FAC (TDJ) | .72
| EASY ADJ FAC (EDJ) | .8
| HARD ADJ FAC (HDJ) | .2
| BASLINE CHG FAC (BDJ) | .35
| MANY CHG BL FAC (MBDJ) | .7
| FREE CHG FAC (FDJ) | .23
| STAFF EXP ADJ FAC (SADJ) | .06
| FULL-UP TRNG FAC (FULLUP) | .4
| KSLEC/MD (KSLCPM) | .237
| DEFAULT ACTIVITY (PDEFLT) | .2
| EFFORT EXPONENT (AMPFAC) | .95
| EFFORT STD DEV FAC (ELFAC) | .72
| DURATION FAC (DFAC) | .488
| DURATION EXP (DEXP) | .356
| DUR STD DEV FAC (DIFAC) | .149
| PAGES DOC/KSLEC (PPKSLCEC) | .120
| DOC EXPONENT (AD) | .923
| COST PER PAGE (FCOST) | .30
| DOC STD DEV FAC (DFAC) | .2
| COMPR HR FAC (CFAC) | .252
| CPTR TIME EXP II (AC) | .96
| CPTR TIM STD DEV FAC (CIFAC) | .15
| COMPR HR COST (CPTHCOST) | .0
| STAFF STD DEV FAC (SIFAC) | .153
| HALF-TIME EFF. FACTOR (HIFAC) | .15
| PUTNAM KA/K FACTOR (KTFAC) | .88
| WBS TASK SKIPS (SKIPS) | .0, 14, 21
| PROMPT1 | "Expected value, kilo-lines executable source"
| PROMPT2 | "Maximum value, kilo-lines executable source(99% confidence level)"
| PROMPT3 | "Minimum value, kilo-lines executable source(99% confidence level)"
| PROMPT4 | "1. How much new code is to be produced (completely new modules)?"
| PROMPT5 | "2. How much code exists in modules requiring modification?"
1. How much code will be deleted from these existing modules?
2. How much code will be added to these existing modules?
3. How much code will be changed in other ways in these modules?
4. How much code will be deleted as entire modules from existing code?
5. How much of the remaining existing code must be retested?
6. How much code will be added to these existing modules?
7. How much code will be changed in other ways in these modules?
8. How much code will be deleted as entire modules from existing code?
9. How much of the remaining existing code must be retested?
10. What percent of the programming task is in Assembly language?
11. What percent of the new or modified code must be storage-optimized?
12. What percent of the new or modified code must be timing-optimized?
13. What percent of the total programming task is 'easy'?
14. What percent of the total programming task is 'hard'?
15. What percent of the total program requirements will be established and stable before design, and will not be altered before delivery?
16. What percent of the requirements are likely to change slightly before delivery, but will do so under baseline change control?
17. What percent of the requirements are likely to change more drastically before delivery, but will do so under baseline control?
18. What is the average staff experience, in years, obtained from work similar to that required in the task being estimated?
19. What percent of the requirements are likely to change slightly before delivery, but will do so under baseline change control?
20. What percent of the requirements are likely to change more drastically before delivery, but will do so under baseline control?
21. What is the average staff experience, in years, obtained from work similar to that required in the task being estimated?
22. What percent of the total program requirements will be established and stable before design, and will not be altered before delivery?
23. What percent of the requirements are likely to change slightly before delivery, but will do so under baseline change control?
24. What percent of the requirements are likely to change more drastically before delivery, but will do so under baseline control?
25. What is the average staff experience, in years, obtained from work similar to that required in the task being estimated?
26. What percent of the total program requirements will be established and stable before design, and will not be altered before delivery?
27. What percent of the requirements are likely to change slightly before delivery, but will do so under baseline change control?
28. What percent of the requirements are likely to change more drastically before delivery, but will do so under baseline control?
29. What is the average staff experience, in years, obtained from work similar to that required in the task being estimated?
30. What percent of the total program requirements will be established and stable before design, and will not be altered before delivery?
31. What percent of the requirements are likely to change slightly before delivery, but will do so under baseline change control?
32. What percent of the requirements are likely to change more drastically before delivery, but will do so under baseline control?
33. What is the average staff experience, in years, obtained from work similar to that required in the task being estimated?
34. What percent of the total program requirements will be established and stable before design, and will not be altered before delivery?
35. What percent of the requirements are likely to change slightly before delivery, but will do so under baseline change control?
36. What percent of the requirements are likely to change more drastically before delivery, but will do so under baseline control?
37. What is the average staff experience, in years, obtained from work similar to that required in the task being estimated?
38. What percent of the total program requirements will be established and stable before design, and will not be altered before delivery?
size and complexity
ANSWERS, minimal, average, extensive, -1.033
PROMPT34, "31. Previous experience with operational computer to be used
ANSWERS, minimal, average, extensive, -0.759
PROMPT35, "32. Previous experience with programming language(s) to be used
ANSWERS, minimal, average, extensive, -1.149
PROMPT36, "33. Use of top-down methodology"
ANSWERS, low, medium, high, -0.493
PROMPT37, "34. Use of structured programmer team concepts"
ANSWERS, low, medium, high, -0.622
PROMPT38, "35. Use of Structured Programming"
ANSWERS, low, medium, high, -0.577
PROMPT39, "36. Use of design and code inspections"
ANSWERS, low, QA, peer, -0.432
PROMPT40, "37. Classified security environment for computer"
ANSWERS, yes, no, -0.617
PROMPT41, "38. Hardware under concurrent development"
ANSWERS, much, some, none, -0.518
PROMPT42, "39. Percent of work done at primary development site"
ANSWERS, <70, 70-90, >90, -1.147
PROMPT43, "40. Development computer access mode"
ANSWERS, remote, scheduled, demand, -0.742
PROMPT44, "41. Percent of development computer access availability"
ANSWERS, <30, 30-60, >60, -0.718
PROMPT45, "42. Quality of SW development tools and environment"
ANSWERS, poor, ok, good, -0.693
PROMPT46, "43. Maturity of system and support software"
ANSWERS, buggy, ok, good, -0.693
PROMPT47, "44. Overall adverse constraints on program design"
ANSWERS, severe, average, minimal, -0.568
PROMPT48, "45. Is the program real-time, multi-task"
ANSWERS, chiefly, some, no, -2.3
PROMPT49, "46. SW to be adaptable to multiple computer configurations or environments"
ANSWERS, yes, no, -0.405
PROMPT50, "47. Adaptation required to change from development to operational environment"
ANSWERS, much, some, minimal, -0.405
Appendix B
DSN Software Cost Model Standard
Work Breakdown Structure Skeleton

The standard Work Breakdown Structure (WBS) skeleton is contained in a file defining a PERT plan of a typical software project. Effort and duration of each task in the WBS skeleton are given in fractions of the total W and T determined by the program.

The first record in the file is merely an identifier, announcing the WBS version number (displayed on the screen when the program starts). The remaining records are of the form

(task code), (task title), (who field), (effort), (duration), (predecessor list), (date)

The (task code) and (task title) fields are strings, while (effort) and (duration) are numeric (fractional values). The (who field) is not used in this version (it appears so as to be compatible with the PERT program). The (predecessor list) is of the form

(task code)

or

(task code), (predecessor list)

The (predecessor list) may extend over several lines; the signal for the end of the list is the period. The (date) field is null; it also appears here to be compatible with the PERT program. When not null, it announces to the PERT program an actual date that the task must complete.
DSN SOFTWARE COST MODEL STANDARD WBS SKELETON

3. SW Architecture and Design Development
   3.1. Define SW architecture, .048, .666, 2,,.
   3.2. SDD, 0, .96, 3, 2, 2,,.
      3.2.1. Write all sections, .00576, .96, 2,,.
      3.2.2. Edit and release, .00288, .654, 3, 2, 2, 3, 3, 2, 3, 6, 5,,
   3.3. System Interface Design, .0346, .96, 2,,
   6.5. Revise Impl Plan, .0192, .655, 3, 1,,
   3.4. Level E Review, .00384, .655, 3, 2, 2, 2, 3, 3, 6, 5,,

4. SW Detailed Design & Production
   4.1. SSD, 0, .96, 4, 1, 2,,.
      4.1.1. "Write Sections 1, 2, 3,, .0096, .96, 3,,
      4.1.2. Write Section 4, .0288, .654, 3,,
      4.1.3. Write Section 5, .0672, .96, 4, 1, 2,,
      4.1.4. Write Section 6, .00672, .96, 3,,
      4.1.5. Write Section 7, .0144, .96, 3,,
      4.1.6. Edit and release, .00874, .96, 4, 1, 1, 2, 1, 3, 4, 1, 4, 4, 1, 5, 5, 4,,
   4.2. SOM, 0, .96, 4, 2, 3,,
      4.2.1. Write preliminary draft, .0125, .654, 4, 1, 2,,
      4.2.2. Complete all sections, .0144, .96, 4, 3,,
      4.2.3. Edit and release, .00605, .96, 4, 2, 2, 5, 4,,
   4.3. High-Level Design Review, .00384, .582, 4, 1, 2, 2, 4, 4, 1, 4, 2, 1,,
   4.4. Module Production & Integration, 0, .96, 4, 4, 9,,
      4.4.1. Executive and control, .0278, .498, 4, 1, 2, 4, 2, 1,,
      4.4.2. I/O Modules, .0278, .498, 4, 1, 2, 4, 3,,
      4.4.3. Interface handlers, .0278, .498, 4, 4, 2,,
      4.4.4. Function A, .0278, .515, 4, 4, 3,,
      4.4.5. Function B, .0278, .515, 4, 4, 4,,
      4.4.6. Function C, .0269, .515, 4, 4, 5,,
      4.4.7. Function D, .0269, .515, 4, 4, 6,,
      4.4.8. Function E, .0269, .515, 4, 4, 7,,
      4.4.9. Function F, .0269, .515, 4, 4, 8,,
   4.5. Special Tasks, 0, .96, 4, 5, 2, 4, 5, 2,,
      4.5.1. Support software, .0192, .963, 4, 3,,
      4.5.2. Other, .0096, .96, 4, 3,,
   4.6. Acceptance Readiness Review, .00384, .582, 4, 2, 2, 2, 4, 5, 4, 1, 1, 4, 1, 3, 4, 1, 4, 4, 1, 5, 4, 4,,

5. SW Test and Transfer
   5.1. Verification tests, .0432, .535, 4,,
   5.2. Contingency, .0384, .96, 4, 1, 2,,
   5.3. STT, 0, .96, 5, 3, 2,,
      5.3.1. Write all sections, .0221, .96, 4, 3,,
      5.3.2. Edit and release, .00384, .96, 5, 4,,
   5.4. Acceptance tests, .0317, .776, 5, 3, 1, 5, 3, 1,,
   5.5. Demonstration tests, .0336, .704, 5, 4,,
   5.6. "Transfer, CDE to ODE", .0115, .704, 5, 5, 5, 3, 2, 5, 2, 4, 2, 4, 1,,

6. Mgmt Tasks and Milestones
   6.1. CDE Activities, .0576, .96, 0,,
   6.6. QA Audit, .04, .922, 5, 4,,