

A 283 661


IDA PAPER P-2830

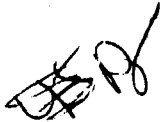
ESTIMATING SOFTWARE DEVELOPMENT COSTS AND SCHEDULES FOR SPACE SYSTEMS

James Bui
Neang I. Om

May 1994

DTIC
ELECTE
AUG 17 1994
S G D

Prepared for
Ballistic Missile Defense Organization



94-25909

Approved for public release; distribution unlimited.

94 8 16 039



INSTITUTE FOR DEFENSE ANALYSES
1801 N. Beauregard Street, Alexandria, Virginia 22311-1772

DTIC QUALITY INSPECTION

IDA Log No. HQ 93-43514

DEFINITIONS

IDA publishes the following documents to report the results of its work.

Reports

Reports are the most authoritative and most carefully considered products IDA publishes. They normally embody results of major projects which (a) have a direct bearing on decisions affecting major programs, (b) address issues of significant concern to the Executive Branch, the Congress and/or the public, or (c) address issues that have significant economic implications. IDA Reports are reviewed by outside panels of experts to ensure their high quality and relevance to the problems studied, and they are released by the President of IDA.

Group Reports

Group Reports record the findings and results of IDA established working groups and panels composed of senior individuals addressing major issues which otherwise would be the subject of an IDA Report. IDA Group Reports are reviewed by the senior individuals responsible for the project and others as selected by IDA to ensure their high quality and relevance to the problems studied, and are released by the President of IDA.

Papers

Papers, also authoritative and carefully considered products of IDA, address studies that are narrower in scope than those covered in Reports. IDA Papers are reviewed to ensure that they meet the high standards expected of refereed papers in professional journals or formal Agency reports.

Documents

IDA Documents are used for the convenience of the sponsors or the analysts (a) to record substantive work done in quick reaction studies, (b) to record the proceedings of conferences and meetings, (c) to make available preliminary and tentative results of analyses, (d) to record data developed in the course of an investigation, or (e) to forward information that is essentially unanalyzed and unevaluated. The review of IDA Documents is suited to their content and intended use.

The work reported in this document was conducted under contract MDA 903 BS C 0003 for the Department of Defense. The publication of this IDA document does not indicate endorsement by the Department of Defense, nor should the contents be construed as reflecting the official position of that Agency.

IDA PAPER P-2830

ESTIMATING SOFTWARE DEVELOPMENT COSTS AND SCHEDULES FOR SPACE SYSTEMS

James Bui
Neang I. Om

May 1994

Accession For	
NTIS	CRA&I ☒
DTIC	TAB ☐
Unannounced	☐
Justification	
By	
Distribution /	
Availability Codes	
Dist	Avail and/or Special
A-1	

Approved for public release; distribution unlimited.



INSTITUTE FOR DEFENSE ANALYSES

Contract MDA 903 89 C 0003

Task T-R2-597.12

PREFACE

This paper was prepared by the Institute for Defense Analyses (IDA) for the Ballistic Missile Defense Organization (BMDO) under a task entitled "Strategic Defense System Phase One Engineering and Technical Support (POET)." The objective of the task was to provide analyses and recommendations to BMDO for defining a baseline concept for the Phase One Strategic Defense System (SDS). In support of that objective, IDA examined the cost and schedule estimates for the software involved in SDS architectures. Part of that work involved developing methods for estimating the costs and schedules of SDS software. This paper presents these methods and explains how they were developed.

This work was reviewed by Beth Springsteen, Thomas P. Frazier, and J. Richard Nelson of IDA and William Kuhn of the MITRE Corporation.

CONTENTS

Executive Summary	S-1
I. Introduction	I-1
A. Background	I-1
B. Approach	I-2
C. Scope	I-2
D. Report Organization	I-3
II. Cost Estimation	II-1
A. Introduction	II-1
B. Method	II-1
C. Database	II-2
D. Data Evaluation and Normalization	II-2
E. Equivalent Source Lines of Code	II-5
F. CER Development	II-5
G. Results	II-6
1. Space Mission Ground Segment CERs	II-6
2. Space Mission Embedded Flight CER	II-8
H. Effort Not Accounted for by the CERs	II-9
I. Issues	II-10
III. Schedule Estimation	III-1
A. Introduction	III-1
B. Method	III-1
C. Database	III-2
D. Results	III-3
E. Schedule Not Accounted for by the TER	III-5
IV. Conclusion	IV-1
Appendix A: Model Application	A-1
Appendix B: Analysis of Outliers	B-1
References	C-1
Abbreviations	D-1

FIGURES

II-1. Software Development Phases	II-3
II-2. Database Classification	II-3
III-1. Estimates of Application and Operating System Software CSCI Development Duration	III-4
III-2. Estimates of Support Software CSCI Development Duration	III-5

TABLES

II-1. Sources and Numbers of Cost Data Points by Software Type	II-4
II-2. ESLOC Ranges	II-9
II-3. Effort Phase Distribution for Semi-Detached and Embedded Modes	II-10
III-1. Sources and Numbers of Schedule Data Points by Software Type	III-2
III-2. Schedule Database Averages and Ranges	III-3
III-3. Schedule Phase Distribution for Semi-Detached and Embedded Modes	III-6

EXECUTIVE SUMMARY

The Ballistic Missile Defense Organization (BMDO) asked the Phase One Engineering Team (POET) to investigate the factors that drive software development costs and schedules. In particular, the BMDO asked the POET¹ to obtain and analyze historical data from past space system programs to identify the characteristics that influence software development costs and schedules.

Many previous research efforts have shown that software size, measured by source lines of code, drives software development costs and schedules. This report addresses the factors that affect software development costs and schedules for space systems. The study focuses on software for both the embedded flight and ground segments.

We used data from the Air Force Space and Missile Systems Center (SMC) database. The data in the SMC database were collected from the Space Systems Cost Analysis Group (SSCAG) participants. The SSCAG is an industry and government group formed to enhance space system cost analysis. Another data source was the NASA Goddard Space Flight Center database.

We evaluated and normalized the databases into a consistent format. We used data at the level of the computer software configuration item (the next level of information down from a software project). The data were segregated by basing mode, software type, mission equipment (manned and unmanned), and user (DoD and NASA). Multiple regression was used to develop cost- and time-estimating relationships.

In analyzing software costs, we examined the following cost drivers: size, basing mode (ground and flight), software type, mission equipment type, and user. The schedule analysis examined size and staffing level as schedule drivers. The following is a summary of our findings.

- Cost:
 - Software size is still a good predictor of software development cost. However, software type (application and support) and user (DoD and NASA) are also important cost factors.

¹ POET is a conglomerate of Federally Funded Research and Development Centers (including IDA) that supports BMDO.

- Ground segment support software costs about 20% to 25% less to develop than application software.
- The DoD's software development costs are higher than NASA's—about 60% more for the ground segment and 40% more for embedded flight software.
- The DoD's embedded flight software development costs are on average five times higher than ground segment software costs.
- Embedded flight software development exhibits more diseconomies of scale than does ground segment software development.
- Schedule:
 - Software size, staffing level, and basing modes are the drivers of software development schedules.
 - Adding staff shortens software development duration at a decreasing rate, because inefficiency is a by-product of larger staff sizes.

We make the following recommendations to improve the BMDO's software cost and schedule estimating.

- Examine the effects of the Ada programming language on software size, cost and schedule. (Our study, which did not include Ada programs, could be updated when Ada data points are available.)
- Examine historical data on ground-based battle management and command, control, and communications programs.² (Our study did not include data points from programs of this nature.)

² The BMDO's Command and Control Element (C2E) is the most software-intensive element. It performs battle management and command, control, and communications functions that are unprecedented in terms of functionality and complexity.

I. INTRODUCTION

A. BACKGROUND

Department of Defense (DoD) software expenditures for weapon systems have grown tremendously during the last twenty years [1]. Weapon systems in the early 1970s did not have any software, while current systems have over one million source lines of code (SLOC).

When this research effort began in 1992, the estimate for the total size of software in the elements of the former Strategic Defense Initiative Organization (SDIO) was over ten million SLOC. [The SDIO elements in 1992 were Brilliant Pebbles, Brilliant Eyes, Ground-based Radar, Battle Management and Command, Control, and Communications (BM/C³), etc.] SDIO is now the Ballistic Missile Defense Organization (BMDO), and although its mission has changed from strategic to tactical missile defense, software is still an important part of the system.

Two offices within BMDO oversee software size and cost estimation. The System Engineering and Integration Directorate is responsible for reviewing the software size estimates provided by the various BMDO element project offices. These software size estimates are included in the Cost Analysis Requirements Documents (CARD). The Cost Estimating and Analysis Directorate is responsible for estimating software costs and schedules from information contained in the CARDS.

To perform these analyses, analysts must identify and understand the technical parameters that influence the cost of future software-intensive BMDO elements. Insights into these relationships permit independent assessments of software estimates provided by the element project offices.

SDIO asked the Phase One Engineering Team (POET) to investigate technical parameters that drive software development size, cost, and schedule for future space-based systems. The POET, a conglomerate of Federally Funded Research and Development Centers that supports BMDO, performed similar work in 1990 where software development costs were found to vary significantly by the accommodating hardware location (ground, air, and space) [2].

As part of POET's effort, the Institute for Defense Analyses (IDA) studied software development costs and schedules, and The Aerospace Corporation studied software size. This paper documents IDA's portion of the overall POET effort. The entire POET effort, including the contributions by analysts of The Aerospace Corporation, is documented in a separate report [3].

B. APPROACH

The focus of the study was to analyze existing databases that contain robust samples of historical software development efforts. First, we analyzed the Space Systems Cost Analysis Group (SSCAG) database [4]. The SSCAG, sponsored by the U.S. Air Force Space and Missile Systems Center (SMC) and the National Aeronautics and Space Administration (NASA), is a government and industry working group formed to advance space systems cost analysis. The SSCAG database contains software development information of past programs submitted by contractors and data collected by SMC and NASA. Management Consulting and Research, Incorporated, maintains the SSCAG database for SMC. We also used the NASA Goddard Space Flight Center [5] software development database. This database contains data that do not overlap the SSCAG database.

We developed cost-estimating relationships (CERs) at the level of the computer software configuration item (CSCI). We developed separate CERs for ground segment and embedded flight software. We also investigated cost drivers based on the mission equipment type (unmanned and manned), software type (application, support, and operating system), and user (DoD and NASA). To estimate software schedule duration, we developed time-estimating relationships (TERs).

C. SCOPE

We examined software size for DoD and NASA space missions for both embedded flight and ground segment software. The CERs and TERs developed estimate software development efforts from product design through CSCI integration and test. Factors are available to estimate the other activities (system requirements and system-level integration and tests) not addressed by the CERs and TERs in our study. Independent verification and validation before system deployment is also not included in our analysis.

The programs included in our database used the following non-Ada programming languages: FORTRAN, Assembly, Jovial, and PL1.

The methods derived from our analysis could be used to estimate software development costs and schedules of a satellite system. The data used in this study did not include command and control systems with intensive BM/C³ functions.

D. REPORT ORGANIZATION

This report is divided into four chapters. Following this introduction (Chapter I), Chapter II documents the development of the CERs. Chapter III describes the schedule analysis, and Chapter IV summarizes the findings of our research and presents a list of recommendations to improve BMDO software cost estimation. Appendix A presents examples of application of the models presented in the text, and Appendix B describes the analysis of residuals to eliminate outliers in the data.

II. COST ESTIMATION

A. INTRODUCTION

This chapter documents the CERs developed for embedded flight and ground segment software. It first describes the method used to derive the CER and then discusses the data sources and data sample. Section D addresses the data evaluation and normalization process. Section E explains the notion of equivalent source lines of code (ESLOC), and Section F discusses the CER development process. Section G presents the CERs and results. Section H provides factors to be used to account for activities not included in the CERs. Finally, Section I explores issues surrounding BMDO software cost estimating.

B. METHOD

The method used to develop the CERs to estimate software development cost followed previous work by IDA for BMDO [4]. The basic framework traditionally used to derive software development CERs assumes that costs are related to software size in an exponential form [6]:

$$\text{Effort} = A \times (\text{Size})^B. \quad (\text{II-1})$$

In this equation, effort is a measure of the number of man-months required to develop the software.¹ The coefficient A is the intercept term derived through a log transformation regression. The input, size, is measured by the number of source lines of code. The exponent B is derived from the regression analysis.

In addition to the traditional size cost driver, we examined the type of equipment used for the mission (manned and unmanned space systems) and the software type (application, support, and operating system) as potential cost drivers. Cost differences associated with the different users (DoD and NASA) were also investigated because DoD programs tend to be operational with long lifetimes while NASA programs tend to be experimental and short-lived.

¹ Software cost is measured in man-months of effort, which includes management, design, programming, test and simulation, training, and database administration.

We analyzed the residuals to eliminate outliers that influenced the fit of the regression function in our model. The methods used were Hat Diagonal Matrix, RSTUDENT, and Cook's Distance analyses (see Appendix B).

C. DATABASE

The data in our analysis came from three sources: the Space Systems Cost Analysis Group (SSCAG) [4], the Jet Propulsion Lab (JPL) [7],² and the Goddard Space Flight Center [5]. The data in the SSCAG database contains software used in various space programs, including the Space Shuttle, and both ground segment and embedded flight software. It contains data from 22 member companies, including the Space and Missile Systems Center (SMC). The SSCAG database contains software development programs at CSCI and project levels were measured in source lines of code and in man-months. NASA data included in the SSCAG database were from JPL programs and the Space Shuttle program.

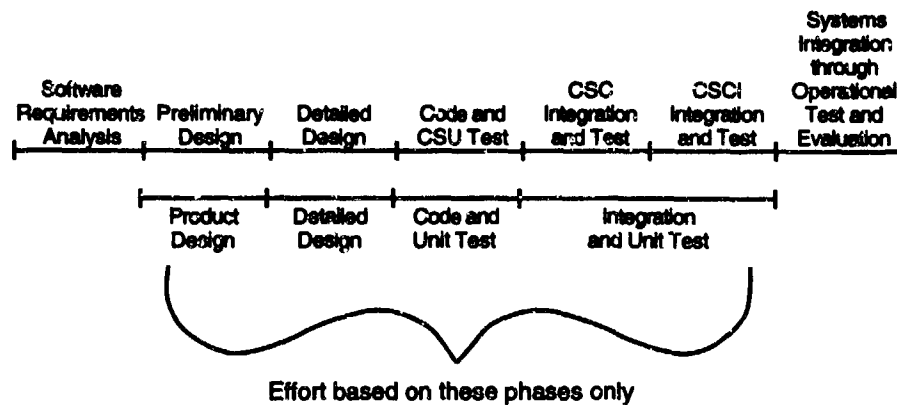
The Goddard database comprises flight and ground segment software for space programs managed by Goddard. Except for the Spacelab data points, all of the Goddard data are from unmanned space missions such as the High Energy Astrophysical Observatory, Solar Max, and the Earth Radiation Budget Satellite.

D. DATA EVALUATION AND NORMALIZATION

We evaluated each database for data content and possible overlap. Each data point was checked for correct basing mode (where the software resides), software type (application, support, operating system), and size. The data points that did not have all development phases (preliminary design through CSCI integration and test, as shown in Figure II-1) were excluded. We also verified the information in the database through meetings with SMC and Management and Consulting Research, Incorporated, and review of source documents [5, 6, and 8]. Only actual values were used for the analysis (estimated values were excluded).

First, we divided the data into two basing modes: ground segment and embedded flight. Within each basing mode the data were further classified by mission type (unmanned space and manned space missions). We then segregated the data into three software types (application, support, and operating system) to test the hypothesis that support software is the least expensive of the three. Figure II-2 depicts our scheme for classifying the data.

² The JPL database is included in the SSCAG database.



Sources: References [6 and 8].

Figure II-1. Software Development Phases

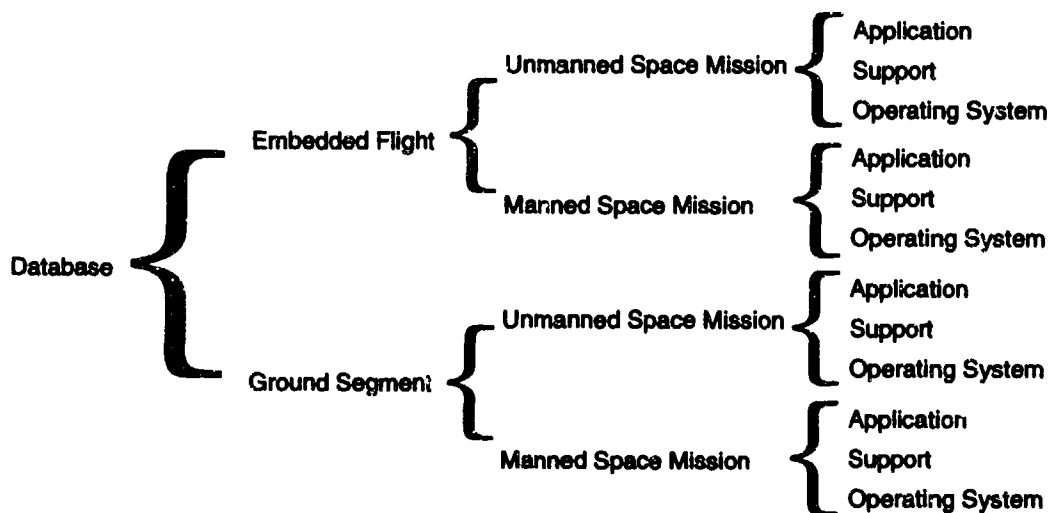


Figure II-2. Database Classification

Application software is specific to satellite or payload missions. Such software is critical to the mission and requires a high degree of real-time processing. Real-time processing provides output that does not delay the user or process [9]. This includes signal processing, mission control, command, control, and communications, and so on.

Support software supports the mission and is not critical to basic operation. Support software could be considered "off-line" because it is not required for real-time processing in order to complete a mission. This includes post processing, simulation, training, database management, maintenance, test, and so on. The difference between application and support software is the degree of real-time processing required to execute the task.

Operating system software manages the hardware resources, including computer or system operations. It is designed to operate, maintain, and control specific computer equipment.

Our original sample included 253 data points, 224 for the ground segment and 29 for embedded flight. The ground segment data included 171 unmanned and 53 manned missions. The distribution of application and support ground segment software was nearly equal. Few data points for operating system software were in the database. The embedded flight software included 17 unmanned missions and 12 manned missions. Almost all of the embedded flight software was classified as application software.

Later in the study, we decided to analyze only data for the unmanned space mission systems for two reasons: (1) the BMDO's main interest was in unmanned space missions, and (2) after we performed residuals analysis to eliminate questionable data points (reducing the number of data points from 253 to 145), too few manned space mission data points remained in the database.

The ground segment data for our reduced data set included 136 unmanned missions. The DoD data points accounted for 6% of the unmanned missions. The segregation between application and support ground segment software was 53% and 46% relatively. Only 1.5% of the data points were operating system software. The embedded flight software for our reduced data set included only 9 unmanned missions. All of the embedded flight data points were application software.

The sources and numbers of data points for the unmanned space ground segment and embedded flight software are shown in Table II-1.

Table II-1. Sources and Numbers of Cost Data Points by Software Type

Software Type	SSCAG Data		Goddard Data	Total
	DoD Mission	NASA Mission		
Ground Segment				
Application	9	16	47	72
Support	0	35	27	62
Operating System	0	1	1	2
Total	9	52	75	136
Embedded Flight				
Application	2	3	4	9
Support	0	0	0	0
Operating System	0	0	0	0
Total	2	3	4	9

E. EQUIVALENT SOURCE LINES OF CODE

Since software size is the primary driver of software cost, a convention to account for the true size of a CSCI, including reused and modified software, is essential. We measured software size in source lines of code. We included data declarations, job-control language, files, tests, simulations, and training, but excluded comment lines, commercial off-the-shelf software, and in-house software. We adjusted the software size for whether the code was reused and/or modified using the method documented in [7]:

$$\text{ESLOC} = \text{New SLOC} + 0.5 \text{ Modified SLOC} + 0.25 \text{ Inherited SLOC}.$$

The term ESLOC is equivalent source lines of code, New SLOC is newly developed code, Inherited SLOC is synonymous to reused code, and Modified SLOC is between new and reused code.

F. CER DEVELOPMENT

We analyzed the ground segment software development cost for the application and operating system software separately from the support software. We also developed a CER for ground segment software for DoD users only. The embedded flight CER contains only application software.

Development cost, measured in man-months (MM) of effort, is the dependent variable in the multiple regression analyses. Candidate cost-driving variables are:

- Size measured in thousands of equivalent source line of code (EKSLOC).
- Embedded flight software indicator variable (FLT). This 1/0 indicator variable has a value of 1 for embedded flight software and a value of 0 otherwise.
- Application software indicator (APP). This 1/0 indicator has a value of 1 for application software and a value of 0 otherwise.
- Operating system software indicator (SYS). This 1/0 indicator has a value of 1 for operating system software and a value of 0 otherwise.
- Support software indicator (SUP). This 1/0 indicator has a value of 1 for support software and a value of 0 otherwise.
- Department of Defense user indicator (DOD). This 1/0 indicator has a value of 1 for DoD users and a value of 0 otherwise.

Our CERs take on the intrinsically linear multiplicative form $Y = A \times B$ as shown in Equation II-1. To estimate the coefficient of this equation, we transformed the equation to a logarithmic form and then applied ordinary least square linear regression. When the equation is transformed from the logarithmic form back to a multiplicative form, the

multiplicative residuals are assumed to be distributed log normally. As the log normal distribution is right-skewed, the expected value and most likely value (mode) of the residuals are no longer equal. Therefore, an adjustment must be made for the multiplicative form to yield the expected value for the dependent variable. We made this adjustment by adding one-half of the regression mean square error to the constant term of the logarithmic equation before it is transformed into the multiplicative form [10]. We then transformed the intercept term into a multiplicative constant, which yields an adjustment factor (adjusted constant term/unadjusted constant term) on the multiplicative form greater than one. In reporting the estimating relationships, we report the adjusted multiplicative equation along with the factor so that the equation can be back-adjusted to yield the most-likely value.

G. RESULTS

We developed cost-estimating relationships for both ground segment and embedded flight space mission software CSCIs. For the ground segment, we analyzed the application and operating system software separately from the support software. For the embedded flight software, we developed only one CER because the data were comprised of application software only.

1. Space Mission Ground Segment CERs

First, we analyzed the application and operating system software using a multiple-input CER with size (EKSLOC) and user type (NASA or DOD) variables. Then we examined the support software using a single-input CER with size (EKSLOC) as the independent variable.

a. Application and System Software CER

The application and operating system software CER for ground-based CSCIs used for unmanned space missions is presented in Equation II-2.

$$MM = 4.3 \times EKSLOC^{1.08} \times 1.57^{DOD} \quad (II-2)$$

(30.9, .000) (3.7, .000)

N = 74 Adjusted R² = 0.93 SEE = 63 Intercept Adjustment = 1.06

The t-scores and probability levels are in parentheses below the parameter estimates.³ N is the number of observations. The adjusted R² indicates 93% of the variation in MM can be explained by the single variable in Equation II-2 in log transformed space.⁴ SEE, the standard error of the estimate, is in the dimension of the independent variable and indicates better fit for smaller SEE values.⁵

The sample average of the independent variable (EKSLOC) in Equation II-2 is 30.8. The range of EKSLOC is 0.8 to 160.5.

That the EKSLOC coefficient is greater than one in Equation II-2 indicates that cost will increase at a greater rate as size increases. Equation II-2 indicates that DoD application and operating system software for the ground segment is 57% more expensive than NASA software. This may have to do with the fact that DoD systems in our database are different than NASA programs in terms of software development and documentation standards and operational requirements.

b. Support Software CER

The support software CER for ground-based CSCIs used in unmanned space missions is shown in Equation II-3.

$$MM = 4.7 \times EKSLOC^{0.98} \quad (II-3)$$

(31.9, .000)

N = 62 Adjusted R² = 0.95 SEE = 40.2 Intercept Adjustment = 1.07

Equation II-3 indicates that the support software development effort increases at a decreasing rate as the size increases, implying that there are no diseconomies of scale in support software. This outcome is intuitive because support software may not have the complex interfaces that tend to make large application CSCIs more expensive.

³ The t-score is the statistic that tests the null hypothesis that the coefficient B in Equation II-1 is equal to zero against the alternative hypothesis that B is not equal to zero [11]. The t-score is the ratio of the regression coefficient to its standard error. A t-score of about 2.0 implies ~95% confidence that the coefficient is significant. Higher t-scores imply greater confidence in the coefficient significance. An analogy to this statistic might be the signal-to-noise ratio. The probability level statistic shows the confidence level that the estimated coefficient is equal to zero. Lower probability values indicate greater statistical significance.

⁴ The R² is a measure of the fit of a regression equation. An adjustment is made to lessen the effect of increasing the R² value through the addition of independent variables. The adjusted R² modifies the R² to penalize the model containing additional variables when compared with alternative regression models [12, p. 365]. An R² of 1.00 indicates a perfect fit.

⁵ Reference [12, p. 118].

c. DoD Ground Segment CER

For the DoD user, we developed a separate regression model to estimate the cost of ground segment software for military space missions for application software only. The model is shown in Equation II-4.

$$MM = 7.2 \times EKSLOC^{1.04} \quad (II-4)$$

(10.6, .000)

N = 9 Adjusted R² = 0.97 SEE = 53 Intercept Adjustment = 1.007

The average of the independent variable (EKSLOC) in Equation II-4 is 38.1. The range of EKSLOC is 6.8 to 116.8.

Because of the smaller sample size, the adjusted R² improved to 97%. However, software type was not a cost driver. The intercept (A in Equation II-1) in Equation II-4 (7.2) is 67% higher than the multiplier in Equation II-2 (4.3). This is consistent with the estimate for the DoD dummy variable parameter (1.57) in Equation II-2.

The exponent of the variable EKSLOC in both Equations II-2 and II-4 are greater than one. This suggests diseconomies of scale and is in accordance with conventional CERs. However, CERs have been developed that show economies of scale [13, 14, and 15]. Given the description in Reference [7], our ground segment CER could apply to semi-detached CSCIs.

2. Space Mission Embedded Flight CER

Due to insufficient data for the embedded flight software, we combined the operating system and application software data for the CER analysis. To develop the embedded flight CER, we tested all the cost drivers that were used for the ground segment CERs. However, only software size and DoD user proved to be significant variables, as shown in Equation II-5.

$$MM = 8.3 \times EKSLOC^{1.47} \times 1.38^{DOD} \quad (II-5)$$

(22.84, .000) (2.75, .033)

N = 9 Adjusted R² = 0.988 SEE = 58 Intercept Adjustment = 1.001

The average of the independent variable EKSLOC for unmanned missions in Equation II-5 is 13 and the range is 3 to 32.

Equation II-5 indicates that DoD embedded flight software costs 38% more than NASA software. That the EKSLOC coefficient in Equation II-5 (1.47) is greater than in Equation II-2 (1.08) implies more diseconomies of scale in embedded flight software

development than in ground segment software development. The multiplier in Equation II-5 (8.3) is about two times larger than the ground segment CER multiplier in Equation II-2 (4.3), which also indicates higher cost in the embedded flight software development compared with ground segment software. This higher cost in embedded flight software can be attributed to a high degree of real-time processing, ultra-high reliability, interfaces with other equipment besides computer hardware, and computer hardware obsolescence⁶ [16 and 17]. These complexity factors may also explain the diseconomies of scale associated with this type of software.

When doing cost estimates using parametric relationships, analysts must understand the relevant range of the data with which the relationships were developed. The ranges of ESLOC (CSCIs level) used for our models are provided in Table II-2.

Table II-2. ESLOC Ranges

ESLOC	Minimum Value	Average Value	Maximum Value
Ground Application and Operating System	400	26,000	192,700
Ground Support	800	31,00	160,500
Embedded Flight	3,000	13,000	32,000

H. EFFORT NOT ACCOUNTED FOR BY THE CERS

As mentioned previously, our database captures cost from product design through CSCI integration and test. This does not include planning, requirements analysis, system integration, and test activities. Analysts should use factors associated with their programs when adjusting the CERs. If program data are not available, use the effort phase distribution shown in Table II-3. Analysts can adjust the CERs to include the missing phases by multiplying the CERs by the factors shown in the table.

Independent verification and validation (when an independent contractor or agency verifies and validates software being developed by another contractor or agency) is also not accounted for in the CERs. According to [6], this additional effort could range from 20% to 40% of the development effort, depending on the reliability of the software. This factor should be applied only to flight CSCIs and mission critical ground CSCIs.

⁶ Programmers must use space-qualified computer hardware. Because it is time-consuming for computers to be space qualified, this leads to the use of equipment that is several years behind the commercial market in terms of performance and design [16].

**Table II-3. Effort Phase Distribution for Semi-Detached and Embedded Modes
(Very Large Project >512 EKSLOC)**

Mode	Phase	Percentage
Semi-detached (Ground Segment)	Plans and Requirements	7
	Product Design	16
	Programming	48
	System Integration	29
	Percentage of Effort Accounted by CER	64
	Percentage of Effort Not Accounted by CER	36
Embedded (Flight)	Plans and Requirements	7
	Product Design	17
	Programming	44
	System Integration and Test	32
	Percentage of CSCI Effort Accounted by CER	61
	Percentage of CSCI Effort Not Accounted by CER	39

Source: Reference [6, p. 90].

Note: The numbers in this table were normalized so that they add to 100%.

An additional factor of 40% should be applied to the cost estimates generated by the CERs to include system level plans, requirements, integration and test activities.

I. ISSUES

We intended to test cost effects due to the Ada programming language; however, our database did not include data points for Ada programs with all the development phases included. We are aware of research in this area [18], but have not encountered space system CERs with Ada data points.

During our discussions with BMDO officials, the question often asked was whether the CERs can be used to estimate software cost for BM/C³ systems. The data points in our analysis did not include systems of this nature. Our CERs can be used to estimate embedded flight software and ground segment software at ground entry points for satellite systems. BM/C³ systems might have a higher degree of real-time processing than our ground segment data points.

III. SCHEDULE ESTIMATION

A. INTRODUCTION

Previous research has addressed the question of how long it takes to develop software [6 and 13]. The traditional method of estimating software development schedule is to derive an equation that uses development effort (man-months) as the single independent variable. Our analysis also used effort specified as average staffing level (man-month/duration), and included a second variable, software size (EKSLOC), to the equation [19]. Our approach will help answer the question of how much program duration can be shortened with added staff while holding project size (EKSLOC) constant.

The next section discusses the method used to develop TERs. Section C describes the database, and Section D presents the results. Section E provides factors to be used to account for activities not included in the TERs.

B. METHOD

We followed the same method used to derive the CERs to develop the time-estimating relationships. Development time (duration), measured in months from product design through CSCI integration and test, is the dependent variable. The candidate schedule-driving variables are:

- Size measured in thousands of equivalent source line of code (EKSLOC).
- Average staff level (AVG_MM) measured in man-months (total man-months divided by duration).
- Embedded flight software indicator (FLT). This 1/0 indicator has a value of 1 for embedded flight and a value of 0 otherwise.
- Application software indicator (APP). This 1/0 indicator has a value of 1 for application software and a value of 0 otherwise.
- Operating system software indicator (SYS). This 1/0 indicator has a value of 1 for operating system software and a value of 0 otherwise.
- Support software indicator (SUP). This 1/0 indicator has a value of 1 for support software and a value of 0 otherwise.

C. DATABASE

Although derived from the same sources as the data used for the CER analysis, the data sample used in our TER analysis contained only 98 software development programs. These data points were mostly from the Goddard database and the NASA data points in the SSCAG database. The small sample is due to the limited amount of software CSCI schedule data for military systems in the SSCAG database. We used only the data points that included all the development phases considered (product design through CSCI integration and test).

Originally, our TER database had 141 software development programs, which we categorized according to space mission type (manned space and unmanned space) and by basing mode (ground segment and embedded flight). For the same reasons as in our CER analysis (BMDO's interest in unmanned space missions and our residuals analysis) our database was reduced from 141 to 98 data points. Of the 98 data points, 91 were ground segment and 7 were embedded flight CSCIs.

We also categorized the data sample by software type: application, support, and operating system. The ground-based data points had 39 application CSCIs, 51 support CSCIs, and 1 operating system CSCI. All embedded flight data points were application software. A breakdown of our database is shown in Table III-1. Table III-2 shows the averages and ranges of the schedule database.

Table III-1. Sources and Numbers of Schedule Data Points by Software Type

Software Type	SSCAG Data		Goddard Data	Total
	DoD Mission	NASA Mission		
Ground Segment				
Application	0	4	35	39
Support	0	25	26	51
Operating System	0	0	1	1
Total	0	29	75	91
Embedded Flight				
Application	2	3	2	7
Support	0	0	0	0
Operating System	0	0	0	0
Total	2	3	2	7

Table III-2. Schedule Database Averages and Ranges

Variable	Basing Mode	Average	Minimum	Maximum
EKSLOC	Applications System	27.8	1.5	130
EKSLOC	Ground Support	27.6	0.4	192.7
AVG_MM	Applications System	6.4	0.6	16.3
AVG_MM	Ground Support	2.9	0.2	16.5

D. RESULTS

Due to the small sample size of the embedded flight data points, we pooled the ground segment and embedded flight data points for our regression. We tested several different specifications in developing TERs to estimate software development duration. We analyzed the application and operating system software separately from the support software.

1. Application and Operating System Software TER

Software size, staff level, and basing mode proved to be significant explanatory variables, as shown in Equation III-1.

$$\text{Duration} = 7.2 \times \text{EKSLOC}^{0.67} \times \text{AVG_MM}^{-0.48} \times 2.9 \text{ FLT} \quad (\text{III-1})$$

(7.9, .000) (5.4, .000) (6.9, .000)

$$N = 47 \quad \text{Adjusted } R^2 = 0.53 \quad \text{SEE} = 10.2 \quad \text{Intercept Adjustment} = 1.05$$

The t-scores and probability levels are in parentheses below the parameter estimates. N is the number of observations.

2. Support Software TER

The same explanatory variables were significant for support software as for application and operating system software.

$$\text{Duration} = 5.4 \times \text{EKSLOC}^{0.76} \times \text{AVG_MM}^{-0.68} \quad (\text{III-2})$$

(18.4, .000) (12.4, .000)

$$N = 51 \quad \text{Adjusted } R^2 = 0.86 \quad \text{SEE} = 9.4 \quad \text{Intercept Adjustment} = 1.004$$

Equations III-1 and III-2 indicate software development duration decreases at a decreasing rate as staff size increases (as denoted by the negative exponent values of -.48 and -.68 on average staff level). This is due to the inefficiencies of a larger staff size discussed in [20].

Equation III-1 suggests embedded flight software takes 2.9 times longer to develop (holding size and staff level constant), which is consistent with our CER findings. Again, this is due primarily to a high degree of real-time processing, ultra-high reliability, interfaces with other equipment besides computer hardware, computer hardware obsolescence, and so on [16 and 17]. However, the data in our database show that, on average, embedded flight software takes only 25% longer to develop because: (1) embedded flight software is smaller in size (67% less) than ground segment software and (2) more manpower (25% more) is assigned to develop flight software than ground segment software. Figure III-1 depicts estimates of software CSCI development duration predicted by Equation III-1.

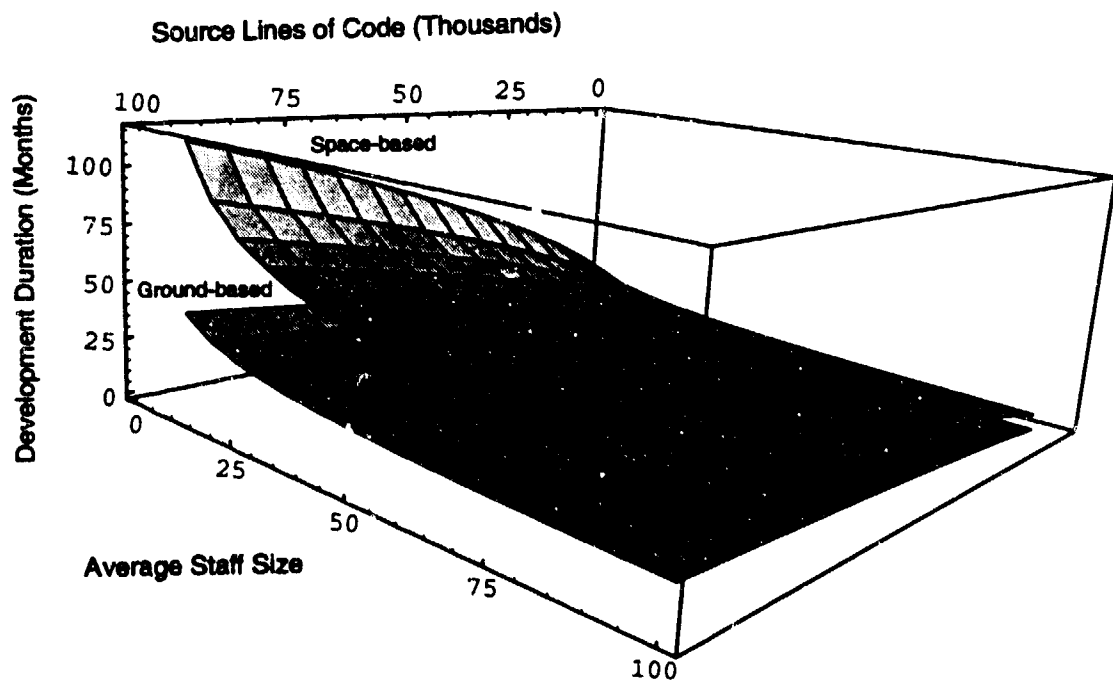


Figure III-1. Estimates of Application and Operating System Software CSCI Development Duration

Equation III-2 suggests that support software takes less time to develop than application and system software. As expected, due to the low degree of real-time processing, the support software development duration decreases at a higher rate than the application and system software (-68 versus -48) with the application of additional staff. Figure III-2 depicts estimates of software CSCI development duration predicted by Equation III-2.

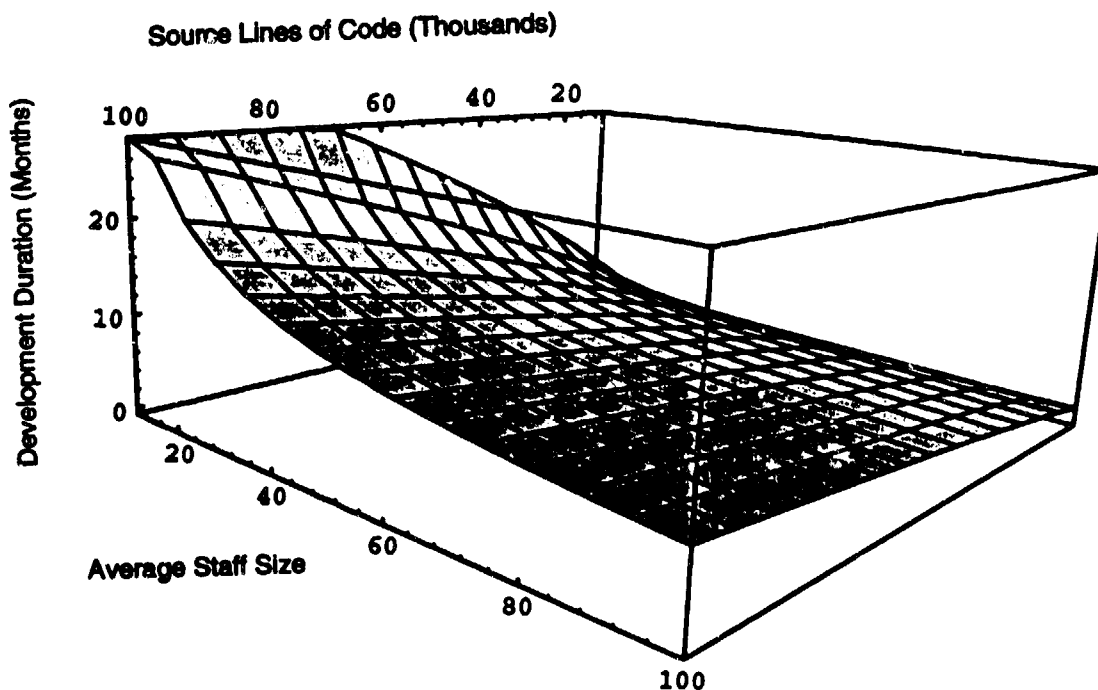


Figure III-2. Estimates of Support Software CSCI Development Duration

In developing our models, we also checked for the presence of multicollinearity using variance inflation factors (VIFs).¹ This is to check the dependency between the two variables in our models, software size and staff level. The application software has a VIF of less than 3.5, and the support software has a VIF of less than 2.5, an indication of no multicollinearity in the models.

E. SCHEDULE NOT ACCOUNTED FOR BY TERS

Since our TERS do not include the plans and requirements and system integration and test phases, the factors in Table III-3 can be used to adjust the TERS.

As shown in the table, an additional 45% to 50% factor should be applied to the duration estimate generated by the TER to include system-level plans, requirements, integration, and test activities.

¹ The VIF for each term in the model measures the combined effect of the dependencies among the regressors on the variance of that term. One or more large VIFs indicate multicollinearity. Practical experience indicates that if any of the VIFs exceed 5 or 10, it is an indication that the associated regression coefficients are poorly estimated because of multicollinearity [21].

**Table III-3. Schedule Phase Distribution for Semi-Detached
and Embedded Modes
(Very Large Project >512 EKSLOC)**

<u>Mode</u>	<u>Phase</u>	<u>Percentage</u>
Semi-detached (Ground Segment)	Plans and Requirements	19
	Product Design	23
	Programming	32
	System Integration	26
	Percentage of Effort Accounted by TER	55
	Percentage of Effort Not Accounted by TER	45
Embedded (Flight)	Plans and Requirements	29
	Product Design	27
	Programming	23
	System Integration and Test	21
	Percentage of CSCI Schedule Accounted by TER	50
	Percentage of CSCI Schedule Not Accounted by TER	50

Source: Reference [6, p. 90].

IV. CONCLUSION

We reached the following conclusions based on our analysis of the cost database:

- User type was a cost driver for ground-based application and operating system software, but not for support software.
- DoD software development costs are about 60% higher for ground-based software and 40% higher for embedded flight software than NASA software development costs.
- Development costs for both ground-based and embedded flight application and operating system software increase at an increasing rate with size. However, as size increases, embedded flight software development costs increase at a much higher rate than ground-based software.
- Development costs for military embedded flight software are on average five times higher than ground segment software.
- Development costs for ground-based support software are about 20% to 25% lower than for application software. Costs also increase at a decreasing rate with size.
- Software productivity (measured in EKSLOC per man-month) did not improve over time for the programs studied between 1977-1988. However, software functionality has increased considerably over the same time period.
- Software language cost differences could not be quantified in the CERs for Ada versus non-Ada software. Our databases did not contain any Ada-language data points that included all the development phases.
- Embedded flight software development exhibits more diseconomies of scale than does ground segment software development.

Our findings concerning the schedule database were:

- Basing mode was a schedule driver for application software development, but not for support software.
- Given the same staffing level and size, embedded flight software takes almost three times longer to develop than ground-based software due mainly to stringent reliability requirements, which result in added testing.
- Although adding more staff decreases software development duration, it does so at a decreasing rate because inefficiency is a by-product of larger staff size.

Because future weapon systems are required to use the Ada programming language, historical data from programs using Ada should be examined. Our study did not include Ada programs but could be updated in the future when Ada data points are available. A comparison between Ada and non-Ada software sizes is contained in [22].

This study addressed space mission software for both flight and ground-based software. Another type of software that is critical to the BMDO's mission is command, control, and communications software. Such software is ground-based and involves a high degree of real-time processing. Our study did not include software of this nature. Insights into historical data for programs with similar characteristics would be useful.

APPENDIX A
MODEL APPLICATION

APPENDIX A

MODEL APPLICATION

Analysts can apply the estimating relationships presented in this paper in two ways. The first approach is to use the models in their role as an assessment tool. The second approach is, given a desired model output, to estimate another independent variable in the model. In the examples that follow, we present the application of the models using the first approach for the CERs and both approaches for the TERs. Before applying the estimating relationships for the software development schedule, we need to make some assumptions about the hypothetical program and the spacecraft associated with it. Our hypothetical spacecraft, the SSD-1, is a medium-sized unmanned surveillance spacecraft for DoD, it requires 13 CSCIs to carry out the mission, 4 embedded flight CSCIs for on-board processing, and 9 ground-based CSCIs for the system's ground-control segment. In addition, 4 ground-based support CSCIs are also required. The aggregate software sizes are 125 KSLOC for embedded flight and 1,119 KSLOC for ground-based, of which 695 KSLOC are support software.

SOFTWARE DEVELOPMENT COST

Presented in Tables A-1 and A-2 are the estimates of the software costs for the SSD-1 program for both ground segment and embedded flight. The total estimates include costs not accounted for by the models, system integration and test (29% for the ground segment and 32% for the embedded flight), and plans and requirements (7% for both ground segment and embedded flight). Using Equations II-2 and II-5, respectively, to calculate development effort in man-months, we can get the estimates for the development cost by assuming a cost of \$16,000 per man-month.

Table A-1. SSD-1 Ground Software Cost Estimates

<u>Basing Mode</u>	<u>CSCI Name</u>	<u>Software Type</u>	<u>New SLOC (K)</u>	<u>Effort (MM)</u>	<u>Cost (K) at \$16 K/MM</u>	<u>Cost per SLOC</u>
Ground	Off-Line Data Processing	S	100	429	6,858	69
Ground	Command & Status	A	30	266	4,254	142
Ground	Communications	A	15	126	2,012	134
Ground	Data Processing	A	45	412	6,591	146
Ground	Mission Performance	A	70	664	10,622	152
Ground	Telemetry Processing	A	20	172	2,745	137
Ground	Operator Interface	A	100	976	15,613	156
Ground	Environment Interface	A	90	871	13,934	155
Ground	Test Support	S	80	344	5,511	69
Ground	Simulator	S	400	1,668	26,683	67
Ground	Configuration Control	A	40	363	5,804	145
Ground	Mission Planning	S	115	492	7,865	68
Ground	Signal Processing	A	14	117	1,868	133
Ground Segment Total			1,119	6,898	110,360	99
System Level Integration (29%)					32,004	
EMD Total					142,365	127
DEM/VAL Phase Cost: Plans & Requirements (7%)					7,725	
Ground Segment Grand Total					150,090	134

Table A-2. Embedded Flight Software Cost Estimates

<u>Basing Mode</u>	<u>CSCI Name</u>	<u>Software Type</u>	<u>New SLOC (K)</u>	<u>Effort (MM)</u>	<u>Cost (K) at \$16 K/MM</u>	<u>Cost per SLOC</u>
Space	Operating System	A	50	3,601	57,619	1,152
Space	Communications	A	35	2,132	34,108	975
Space	Diagnostics	A	25	1,300	20,799	832
Space	Signal Processing	A	15	613	9,816	654
Space Flight Total			125	7,646	122,342	979
System Level Integration (32%)					39,149	
EMD Total					161,491	1,292
DEM/VAL Phase Cost: Plans & Requirements (7%)					8,564	
Space Flight Grand Total					170,055	1,360

SOFTWARE DEVELOPMENT SCHEDULES

For the time-estimating relationships, we present examples of two applications: schedule assessment, and staff level estimate. Tables A-3 and A-4 illustrate the application of our TER in its role as schedule assessment for ground-segment and embedded flight shown in Equation III-1. The estimate will include the two phases not accounted by our TER, plans requirements [19% for ground segment and 29% for embedded flight of the engineering and manufacturing development (EMD) schedule] and system integration and test (26% for ground segment and 21% for embedded flight). Note that EMD duration includes development time accounted for by our TER and system integration and testing duration.

Table A-3. Ground-Based Schedule Estimates

Basing Mode	CSCI Name	Software Type	New SLOC (K)	Staff Level	Duration (months)
Ground	Off-Line Data Processing	S	100	12	33
Ground	Command & Status	A	30	4	36
Ground	Communications	A	15	2	32
Ground	Data Processing	A	45	6	39
Ground	Mission Performance	A	70	9	43
Ground	Telemetry Processing	A	20	3	32
Ground	Operator Interface	A	100	15	43
Ground	Environment Interface	A	90	13	43
Ground	Test Support	S	80	6	45
Ground	Simulator	S	400	45	39
Ground	Configuration Control	A	40	5	39
Ground	Mission Planning	S	115	10	42
Ground	Signal Processing	A	14	3	25
Ground Segment Total			1,119	133	45
System Level Integration (26%)					12
EMD Duration					56
DEM/VAL Phase Cost: Plans & Requirements (19%)					11

Table A-4. Embedded Flight Schedule Estimates

Basing Mode	CSCI Name	Software Type	New SLOC (K)	Staff Level	Duration (months)
Space	Operating System	A	50	20	68
Space	Communications	A	35	15	62
Space	Diagnostics	A	25	10	60
Space	Signal Processing	A	15	6	54
Space Flight Total			125	51	68
System Level Integration (21%)					14
EMD Duration					82
DEM/VAL Duration: Plans & Requirements (29%)					24

In estimating the staff level required given a fixed development duration, we assume one CSCIs for each software type: ground application, ground support, and flight application. In order to use our model to estimate the average staff level, we need to specify when the software is required for the program to proceed on schedule. For the SSD-1 program, we require that the embedded flight software be completed in a duration of 68 months. For the ground-segment, we require a development duration of 45 months. The question is: What average staff level do we need to support the software development schedule?

From our TERs for ground-based support software,

$$\text{Duration} = 5.4 (\text{KESLOC})^{0.76} (\text{Average Staff Level})^{-0.68},$$

and application and operating system software,

$$\text{Duration} = 7.2(\text{KESLOC})^{0.67} (\text{Average Staff Level})^{-0.48} (2.9)^{\text{FLT}},$$

we solved for average staff level, yielding:

- Ground-based support software:

$$\begin{aligned} \text{Average Staff Level} &= [5.4 (\text{KESLOC})^{0.76}/\text{duration}]^{1/0.68} \\ &= [5.4 (695)^{0.76}/45]^{1.47} \\ &= 66.3 \end{aligned}$$

- Ground-based application and operating system software:

$$\begin{aligned} \text{Average Staff Level} &= [7.2(\text{KESLOC})^{0.67}/\text{duration}]^{1/0.48} \\ &= [7.2(424)^{0.67}/45]^{2.08} \\ &= 102.2 \end{aligned}$$

- Embedded flight application and operating system software:

$$\begin{aligned}\text{Average Staff Level} &= [7.2 (\text{KESLOC})^{0.67} (2.9)^{\text{FLT/duration}}]^{1/0.48} \\ &= [7.2 (125)^{0.67} (2.9)(1)/68]^{2.08} \\ &= 71.7\end{aligned}$$

For the SSD-1, where the three developments overlap, a combined average staff of 240.2 software engineers is needed.

There are more ways to apply schedule assessment problems to the data and analyses presented in this paper. The purpose of the SSD-1 example is to provide examples so that BMDO analysts can make better use of the analyses provided.

APPENDIX B
ANALYSIS OF OUTLIERS

APPENDIX B

ANALYSIS OF OUTLIERS

Frequently in regression analysis applications, the data set contains some observations that are outlying or extreme. These outliers may involve large residuals and often have dramatic effects on the fitted least squares regression function. However, the fact that an observation is an outlier (that is, an observation that provides a large residual when the chosen model is fitted to the data) does not necessarily mean that the observation is an influential one with respect to the fitted equation. It is important to study the outlying observations carefully and decide whether they should be retained or eliminated. We used three statistics to help identify influential data points that are outlying with respect to their X or Y values: Hat Diagonal Matrix, RSTUDENT, and Cook's distance.

HAT DIAGONAL MATRIX

The Hat Diagonal Matrix is used to identify outlying X observations. The diagonal element of the hat matrix ($H = X(X'X)^{-1}X'$) defined as $h_i = X'_i (X'X)^{-1}X_i$ (where X_i pertains to the i th observation and X'_i is the i th row of the X matrix pertaining to the i th observation) is called the leverage (in terms of the X values) of the i th observation. It indicates whether or not the X values for the i th observation are outlying. Each h_i reflects the influence of an observed data point Y_i on the fitted value Y_i^* . A large value h_i indicates that the i th observation is distant from the center of the X observations. If the i th data point is an outlying X data point (one with a large leverage value h_i) it exercises substantial leverage in determining the fitted value Y_i^* . A leverage value h_i is usually considered to be large if it is more than twice as large as the means leverage value $h = p/n$, where p is the number of regression parameters in the regression function, including the intercept term. Leverage values greater than $2p/n$ indicate outlying observations that may have undue influence on the fit of the regression model [23].

RSTUDENT

The studentized residual, RSTUDENT, is used to detect outlying or extreme Y observations based on an examination of the residuals. When the residuals e_i have

substantially different variances $\sigma^2(e_i)$, the magnitude of e_i relative to $\sigma(e_i)$ should be considered instead of the regression standard error estimate $\sqrt{MSE}$ to give recognition of differences in their sampling errors. The residual mean square MSE is the residual sum of squares SSE divided by its associated degree of freedom $n - 2$. The variance, denoted by $\sigma^2(e_i) = \sigma^2(1 - h_i)$, has an unbiased estimator $s^2(e_i) = MSE(1 - h_i)$, where e_i is the residual $Y_i - \hat{Y}_i$ and h_i is the leverage value. The ratio of e_i to $s(e_i)$ is called the "studentized residual." It is denoted by:

$$e_i^* = \frac{e_i}{s(e_i)}.$$

When the i th observation is deleted, the regression function is fitted to the remaining $n - 1$ observations, and the point estimate of the expected value when the X levels are those of the i th observation, denoted by $\hat{Y}_{(i)}$, will be compared with the actual Y_i observed value. The residual $d_i = Y_i - \hat{Y}_{(i)}$ is called a "deleted residual." Thus, the studentized deleted residual denoted by d_i^* is:

$$d_i^* = \frac{d_i}{s(d_i)}.$$

However, the studentized deleted residuals d_i^* can be calculated without having to fit the regression function with the i th observation omitted. An algebraically equivalent expression for d_i^* is:

$$d_i^* = e_i \left[\frac{n - p - 1}{SSE(1 - h_i) - e_i^2} \right]^{1/2}.$$

Note that the studentized deleted residual d_i^* can be calculated from the residual e_i , the sum of squares SSE , and the leverage value h_i , all for the fitted regression based on the n observations.

To identify outlying Y observations, we examine the studentized deleted residuals for large absolute values and use the appropriate t distribution to ascertain how far in the tails such outlying values fall. The typical criterion for screening is to use 2.0 for RSTUDENT value. Data points with RSTUDENT value greater than 2.0 would be considered influential outliers [24 and 25].

COOK'S DISTANCE MEASURE

The Cook's distance measures the change in estimated regression coefficient vector caused by deletion of an observation; that is, the difference between the vector b of the estimated regression coefficients based on all n observations and the vector $b_{(i)}$ based on the $n - 1$ observations with the i th observation deleted [24]. The Cook's distance measure D_i uses the boundary of the confidence region for all p regression coefficients β_k ($k = 0, \dots, p - 1$) given by $\frac{(b - \beta)' X' X (b - \beta)}{pMSE} = F(1 - \alpha, p, n - p)$ for measuring the combined impact of the differences in the estimated regression coefficients when the i th observation is deleted:

$$D_i = \frac{(b - b_{(i)})' X' X (b - b_{(i)})}{pMSE}$$

D_i can be evaluated by comparing it with an appropriate F distribution. Although D_i does not follow exactly an F distribution, it has been found to be approximately in the tail area probability of the corresponding F distribution. To assess the magnitude of D_i , one should refer to the corresponding $F(p, n - p)$ distribution and ascertain the tail area probability. If the tail probability of the F distribution is beyond the 90th percentile, the distance between the vector b and $b_{(i)}$ should be considered large, meaning the i th observation has a substantial influence on the fit of the regression function.

Cook's distance measure D_i can be calculated without fitting a new regression function where the i th observation is deleted. An algebraically equivalent expression is [23]:

$$D_i = \frac{e_i^2}{pMSE} \times \frac{h_i}{(1 - h_i)^2}$$

Note that D_i depends on two factors: (1) the size of the residual e_i and (2) the leverage value h_i . The larger e_i or h_i is, the larger D_i is. Thus, the i th observation can be influential: (1) by having a large residual e_i and only a moderate leverage h_i , or (2) by having a large leverage value h_i with only a moderately sized residual e_i , or (3) by having both a large residual e_i and a large leverage value h_i .

While analysis of outlying and influential observations is a necessary component of good regression analysis, it is neither automatic nor foolproof and requires good judgment by the analyst [23].

REFERENCES

REFERENCES

- [1] U.S. General Accounting Office. "Embedded Computer Systems: Defense Does Not Know How Much It Spends on Software." GAO/IMTEC-92-62BR, July 1992.
- [2] Shyman, Steven R., and T. Keith Blankenship. "Improving Methods For Estimating Software Development Costs," Institute for Defense Analyses, Paper P-2388, June 1990.
- [3] Bui, James, Neang Om, Robin Lyons, and Marilee Wheaton. "Estimating Software Development Sizes, Costs, and Schedules for Space Systems." The POET, March 1994.
- [4] Management Consulting and Research, Inc., and Galorath Associates, Inc., "Space Systems Division (SSD): Software Sizing Database, Final Report." 28 January 1992.
- [5] Goddard Space Flight Center. "Software Cost Estimating Model, Volume II." PRC D2210-H, PRC Systems Services, February 1987.
- [6] Boehm, Barry. *Software Engineering Economics*. Englewood Cliffs, New Jersey: Prentice-Hall, Inc., 1981.
- [7] Jet Propulsion Laboratory, California Institute of Technology. "NASA Flight Project Software Database Data Books, Volumes 1 and 2." January 1989.
- [8] U.S. Department of Defense. "Military Standard: Defense System Software Development". DoD-STD-2167A, 29 February 1988.
- [9] Wertz, J. R., and J. Larson, editors. *Space Mission Analysis and Design*. Boston: Kluwer Academic Publishers, 1991, pp. 542-543.
- [10] Goldberger, A. S. *Topics in Regression Analysis*. London: The Macmillan Company, 1968, pp. 120-123.
- [11] Draper, N. R., and H. Smith. *Applied Regression Analysis*. New York: John Wiley and Sons, Inc., 1966, p. 29.
- [12] Kmenta, Jan. *Elements of Econometrics*. New York: The Macmillan Company, 1971.
- [13] Taub, Audrey E. "A Statistical Analysis of the Cost Analysis Technical Center Software Database." The MITRE Corporation, Bedford, MA, MTR-11001, October 1991.
- [14] Nelson, R. "Software Data Collection and Analysis at RADC." Rome Air Development Center, 1978.
- [15] Walston, C. E., and C.P. Felix. "A Method of Programming Measurement and Estimation," IBM Systems, J, 16, 1, 1977. pp. 54-73.
- [16] Norris, P. "Why Is Space Software Special?" in Proceedings from "The Management of Large Software Projects in the Space Industry Meeting," Logica, CNES, Toulouse, France, 1991.

- [17] Anderson, Christine, and Merlin Dorfman. *Aerospace Software Engineering: A Collection of Concepts, Progress in Astronautics and Aeronautics, Volume 136*. American Institute of Astronautics and Aeronautics, 1991, pp. 149.
- [18] Giallombardo, Robert J. "Effort and Schedules Estimating Model for Ada Software Developments." The MITRE Corporation, Bedford, MA, MTR-11303, May 1992.
- [19] Harmon, Bruce R, and Neang I. Om. "Assessing Acquisition Schedules for Unmanned Spacecraft." Institute for Defense Analyses, Paper P-2766, forthcoming.
- [20] Conte, S. D., H. E. Dunsmore, and V. Y. Shen. *Software Engineering Metrics and Models*. Menlo Park, California: The Benjamin/Cummings Publishing Company, Inc., 1986, pp. 258.
- [21] Montgomery, D. G., and E. A. Peck. "Introduction to Linear Regression Analysis," John Wiley and Sons, Inc., 1982. pp. 149 and 287.
- [22] Bailey, J. W., T. P. Frazier, and M. L. Young. "Comparing Ada and FORTRAN Lines of Code: Some Experimental Results." Institute for Defense Analyses, Paper P-2899, November 1993.
- [23] Neter, John, William Wasserman, and Michael H. Kutner, *Applied Linear Regression Models*. Richard Irwin, Inc., Homewood Illinois, 1983.
- [24] R. Cook, "Detection of Influential Observations in Linear Regression," *Techometrics*, 19 (1977), pp. 15-18.
- [25] David A. Belsey, Edwin Kuh, and Roy E. Welsch. *Regression Diagnostics: Identifying Influential Data and Sources of Collinearity*. New York: John Wiley and Sons, Inc., 1980.

ABBREVIATIONS

ABBREVIATIONS

BM/C ³	Battle Management and Command, Control, and Communications
BMDO	Ballistic Missile Defense Organization
CARD	Cost Analysis Requirements Document
CER	cost-estimating relationship
CSCI	computer software configuration item
DoD	Department of Defense
ESLOC	equivalent source lines of code
EMD	engineering and manufacturing development
IDA	Institute for Defense Analyses
JPL	Jet Propulsion Laboratory
NASA	National Aeronautics and Space Administration
POET	Phase One Engineering Team
RSTUDENT	Studentized Residual
SDIO	Strategic Defense Initiative Organization
SLOC	source lines of code
SMC	Space and Missile Systems Center
SSCAG	Space Systems Cost Analysis Group
TER	time-estimating relationship
VIF	variance inflation factor

UNCLASSIFIED

REPORT DOCUMENTATION PAGE			Form Approved OMB No. 0704-0188	
Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188), Washington, DC 20503.				
1. AGENCY USE ONLY (Leave blank)	2. REPORT DATE May 1994	3. REPORT TYPE AND DATES COVERED Final Report, Oct 1992 - Apr 1994		
4. TITLE AND SUBTITLE Estimating Software Development Costs and Schedules for Space Systems		5. FUNDING NUMBERS MDA 903 89C 0003 T-R2-597.12		
6. AUTHOR(S) James Bui, Neang I. Om				
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Institute for Defense Analyses 1801 N. Beauregard Street Alexandria, VA 22311-1772		8. PERFORMING ORGANIZATION REPORT NUMBER IDA Paper P-2830		
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) BMDO/AQP Room 1E1044, The Pentagon Washington, D.C. 20301		10. SPONSORING/MONITORING AGENCY REPORT NUMBER		
11. SUPPLEMENTARY NOTES				
12A. DISTRIBUTION/AVAILABILITY STATEMENT Approved for public release; distribution unlimited.		12B. DISTRIBUTION CODE		
13. ABSTRACT (Maximum 200 words) This paper analyzes historical software development costs and schedules from military and NASA satellite systems. Cost- and time-estimating relationships were developed for ground segment and embedded flight software. In addition to the traditional size variable, other cost drivers were found: software residence (ground or flight), and software type (application and support). Schedule driving variables included size, staffing level, and residence. The equations developed from the study can be used to estimate future space system software development costs or to cross-check estimates generated by other methods.				
14. SUBJECT TERMS Cost Estimating Relationships, Time Estimating Relationships, Space Systems, Computer Programs			15. NUMBER OF PAGES 47	
			16. PRICE CODE	
17. SECURITY CLASSIFICATION OF REPORT Unclassified	18. SECURITY CLASSIFICATION OF THIS PAGE Unclassified	19. SECURITY CLASSIFICATION OF ABSTRACT Unclassified	20. LIMITATION OF ABSTRACT SAR	

NSN 7540-01-280-5500

Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. Z39-18
298-102

UNCLASSIFIED