

AU/AFIT/GCA/LAS/97S-7

GRADUATE SCHOOL OF LOGISTICS AND ACQUISITION MANAGEMENT

AIR FORCE INSTITUTE OF TECHNOLOGY

CALIBRATION AND VALIDATION OF THE CHECKPOINT
MODEL TO THE AIR FORCE ELECTRONIC SYSTEMS
CENTER SOFTWARE DATABASE

by

Thomas C. Shrum, B.S.
Lieutenant, USAF

A Thesis Submitted to the Faculty

In Partial Fulfillment of the Requirements for the Degree of

Master of Science in Cost Analysis

Wright-Patterson Air Force Base, Ohio

September 1997

Disclaimer

The views expressed in this academic research paper are those of the author(s) and do not reflect the official policy or position of the US government or the Department of Defense. In accordance with Air Force Instruction 51-303, it is not copyrighted, but is the property of the United States government.

Contents

	<i>Page</i>
DISCLAIMER	ii
LIST OF TABLES	vi
PREFACE.....	vii
ABSTRACT	ix
SOFTWARE DATABASE	1
Introduction	1
General Issue.....	1
Specific Problem	3
Research Objective	5
Scope of Research.....	6
Importance of Research	6
Thesis Structure	7
LITERATURE REVIEW	9
Overview	9
Software Cost Estimation.....	9
What is a Software Cost Estimate?	10
What Constitutes a Successful Cost Estimate?.....	11
Cost Estimating Approaches	11
The Origin of Parametric Estimating Models.....	12
Macro-Estimation v Micro-Estimation.....	14
The Need for Model Calibration.....	15
Previous Calibration/Validation Research	17
The Thibodeau Study	17
The Illinois Institute of Technology Study.....	18
The Kemerer Study	18
The Ourada Study	20
The Septuagint Study	21
The SMC Software Database (SWDB).....	21
Separation into Calibration and Validation Subsets.....	22
Evaluation of Results.....	23
The Mertes CHECKPOINT Study.....	26
What is a Function Point?.....	27

History of Function Points.....	28
Function Point Analysis (FPA).....	29
Feature Points.....	30
Current Status of Function Points	31
The SLOC/Function Point Debate	31
Function Point Research.....	32
Language Levels	34
CHECKPOINT Version 2.3.1	36
CHECKPOINT Features	36
Sizing Logic.....	36
Summary.....	38
METHODOLOGY.....	39
Overview	39
The USAF Electronic Systems Center Software Database	40
Data Description.....	40
Database Limitations	41
Previous AFIT Studies Using ESC Data.....	42
Assumptions	42
Database Stratification	43
Development Contractor	44
Programming Language	45
Separation into Calibration and Validation Subsets.....	45
Relevant Range Determination.....	47
CHECKPOINT Estimation Process.....	49
Estimating Sequence	49
CHECKPOINT Templates	49
Calibration Procedure	50
Normalization of Effort.....	51
Revised Instructions	52
Validation Procedure.....	56
Statistical Analysis Measures	57
Magnitude of Relative Error (MRE).....	58
Mean Magnitude of Relative Error (MMRE).....	59
Root Mean Square Error (RMS).....	59
Relative Root Mean Square Error (RRMS).....	59
Prediction at Level k/n (PRED (l)).....	60
Conte's Criteria.....	60
Summary.....	61
FINDINGS AND ANALYSIS	64
Overview	64
Data Reduction	64
Database Stratification Results	65
Stratification by Language.....	67
Calibration Results.....	68
Analysis of Results.....	71

Validation Results	81
Analysis of Results.....	83
Comparison of Calibration v Validation Accuracy Results.....	90
Summary.....	91
CONCLUSIONS AND RECOMMENDATIONS	93
Overview	93
Review of Research Objective.....	94
Discussion of Research Activities.....	94
Limitations of Study	95
Conclusions	97
Recommendations for Follow-on Research	100
APPENDIX A: CHECKPOINT CALIBRATION/VALIDATION PROCEDURE.....	102
Calibration Instructions	102
I. Initial Setup.....	102
II. Generation of Default Estimates.....	103
III. Template Construction.....	105
Validation Instructions	106
APPENDIX B: REDUCED ESC DATABASE	108
APPENDIX C: NORMALIZED EFFORT FOR PROJECTS	112
APPENDIX D: CALIBRATION AND VALIDATION RESULTS.....	116
GLOSSARY	139
BIBLIOGRAPHY.....	141

Tables

	<i>Page</i>
Table 1. Software Cost Estimation Techniques.....	12
Table 2. Summary of the AFIT Septuagint Study	24
Table 3. Accuracy Statistics Summary.....	58
Table 4. Database Stratification Results.....	66
Table 5. Calibration Data Range Summary.....	70
Table 6. Calibration Results Summary.....	73
Table 7. Comparison of Stratification Level on Calibration Results.....	81
Table 8. Validation Data Range Summary	82
Table 9. Validation Results Summary.....	84
Table 10. Comparison of Stratification Level on Validation Results	89
Table 11. General Comparison of Model Accuracy	90
Table 12. Calibration and Validation Summary (Calibrated Model).....	98

Preface

This study determined the default and calibrated accuracy of the CHECKPOINT software estimating model in predicting effort for software projects at Electronic Systems Center, Hanscom AFB MA. It also provided a complete methodology for calibrating CHECKPOINT to other software development environments.

I greatly appreciate the support and guidance offered by several key individuals. First, I would like to thank my thesis advisor, Professor Daniel V. Ferens, for his immense assistance—not only for recommending me to perform this research effort, but also for his insight and feedback during the entire research process. I would also like to thank my thesis reader, Dr. David Christensen, for his valuable insight and suggestions in directing my research, particularly for recommending a method to improve the statistical strength of the calibration results. I would also like to thank Capers Jones and Mark Pinis of Software Productivity Research, Inc. for the use of the CHECKPOINT model, the accompanying manual, and their assistance and insights regarding CHECKPOINT’s “finer points.” In addition, I owe a debt of gratitude to two AFIT classmates, Capt David Marzo and Lt Wayne Bernheisel, who collaborated on sections of this thesis and provided valuable support. Dave, thanks for everything. Finally, Capt Karen Mertes and Capt Stephen Southwell deserve much thanks for providing a solid foundation for this research with their 1996 theses.

The love and infinite patience of my wife, Dixie, and my sons Stephen and Daniel were indispensable to the completion of this thesis. I deeply appreciate their unceasing tolerance and support throughout not only this effort, but also my entire time at AFIT.

—Thomas C. Shrum

Abstract

Shrinking DOD budgets, coupled with increasing software demands, have brought close attention to DOD's general inability to accurately estimate software costs for software-intensive systems. The use of uncalibrated or improperly calibrated software cost models is a significant source of these inaccurate cost estimates. This research effort focused on the calibration and validation of CHECKPOINT Version 2.3.1, a computerized software cost estimating tool, to the USAF Electronic Systems Center (ESC) software database. This thesis is a direct follow-on to a 1996 CHECKPOINT study at the Air Force Institute of Technology, which successfully calibrated and validated the CHECKPOINT model to the SMC software database. While this research generally parallels the methodology in the aforementioned thesis, it offers several advancements in the CHECKPOINT calibration and validation procedure, and it refines the data stratification process and the statistical analyses employed. After stratifying the ESC software database into ten usable data sets, the author calibrated and validated the CHECKPOINT model on each data set. Although the results of this study exhibited occasional improvements in estimating accuracy for both the calibration and validation data, the model generally failed to satisfy the accuracy criteria used to assess overall calibration success and estimating accuracy ($MMRE \leq 0.25$, and $PRED(0.25) \geq 0.75$). The calibration effort was unsuccessful for nine of ten calibration data sets, and the model failed to accurately estimate all ten validation data sets. Thus, the CHECKPOINT

model was not successfully calibrated or validated to the 1997 version of the ESC database. The results of this study illuminate the need for complete, accurate and homogeneous data as a requirement for a successful calibration and validation effort.

Chapter 1

Software Database

Introduction

General Issue

In recent years, software has become a crucial aspect of virtually all U.S. Department of Defense (DOD) programs. Software comprises an increasing proportion of the more than \$250 billion contracted for DOD weapons, command and control, and management information (MIS) systems (Department of the Air Force, Guidelines, 1996: 2-15). For example, “DOD spends an estimated \$42 billion annually for the development and maintenance of its computer systems, but only \$7 billion of this sum buys hardware” (Brown, 1996: 7). Since software has become such an important cost, schedule, and performance driver, it is usually on the critical path of a software-intensive system (Department of the Air Force, Guidelines, 1996: 3-5). Thus, “It has become alarmingly apparent to DOD executives and the U.S. Congress that the software ‘tail’ wags the system ‘dog’” (Brown, 1996: 7).

Why has software become so important to DOD? Software accomplishes the following:

1. It provides the flexibility to adjust to previously unknown threats.
2. It allows us to do more with less.

3. It increases the capabilities of airmen, soldiers, sailors, engineers, managers, and battlefield commanders alike.
4. It provides the versatility and leverage we need to compete and win (Department of the Air Force, Guidelines, 1996: 2-3, 2-4).

Simply, software provides the “brains” of DOD’s weapon systems. Lieutenant General Robert Ludwig stated that the fly-by-wire F-16C aircraft, without software, would be nothing more than “a 15-million dollar lawn dart” (Department of the Air Force, Guidelines, 1996: 2-5).

Unfortunately, DOD’s ability to effectively manage the development and maintenance of large-scale software systems has not kept pace with demand (Brown, 1996: 7). In particular, the overall inability to accurately estimate the cost and/or schedule of a new weapon system continues to plague DOD, and the ensuing cost overrun or schedule slippage can ultimately result in the cancellation of a high profile program (e.g., the U.S. Navy A-12 program). This is especially evident in the realm of software supporting these systems. “Software costs have been particularly troublesome, possibly because they have been difficult to estimate and track” (Space Systems Cost Analysis Group, 1995: 2-1). The shrinking DOD budget, coupled with spiraling software costs, necessitates a means to more accurately estimate the software aspect of these programs.

There are several reasons inherent to the DOD environment that create difficulties in generating an accurate estimation of software costs. First, DOD contracts out most software development projects to a myriad of independent contractors, each with its own project management style (Ferens, 1996: 28). In addition, DOD suffers from a shortage of qualified cost analysts to perform these estimates, as well as a lack of technical experts on the aspects of the development efforts to be estimated (Ferens, 1996: 28). Also, the

analysts are often compelled to perform these estimates in a very short period of time (e.g., evaluation of contractor bids during a source selection). Finally, there is a dearth of relevant, reliable historical software project data to provide the foundation for an estimation methodology (Ferens, 1996: 29).

These circumstances generate a requirement for an estimation tool that is fast, systematic, easy to use, and is capable of providing an estimate in the early stages of development. Accordingly, computerized parametric estimating models are often the preferred solution in DOD (Ferens, 1996: 29). Models such as COCOMO, PRICE-S, SEER-SEM, and CHECKPOINT are among the most popular computerized estimating models utilized in DOD.

Obtaining useful estimates from these models has proven difficult for a variety of reasons. First, these models are often inaccurate and unstable in many situations (Ferens, 1996: 29). Also, input parameters, terminology, and underlying algorithms can differ greatly between models (Coggins and Russell, 1993). In addition, the models often have been developed from project databases dissimilar to that where they are used. Despite these limitations, DOD relies heavily on these models for generating software estimates.

Specific Problem

It is crucial that a parametric model such as CHECKPOINT be calibrated to a specific user environment if the desired accuracy of estimates is to be realized. Model calibration is a procedure that adapts a cost estimating model to a specific user environment using actual historical data from past projects in that environment. A sound calibration procedure first requires the analyst to be thoroughly familiar with the strengths and limitations of the model, the underlying algorithms and assumptions

utilized, as well as the precise definitions of all input terminology (Galonsky, 1995: 1-1, 1-2).

This thesis adds to the “Septuagint Study,” a body of research (in the form of seven independent theses) performed from 1995-1996 at the Air Force Institute of Technology (AFIT). This research stream focused on the calibration of various parametric software estimating models to a large database within the Air Force (these studies will be discussed extensively in Chapter 2). The previous calibration efforts have yielded mixed results, and it is hoped this study will provide additional insight into this area of research, as well as practical knowledge to be utilized in the field.

Specifically, this research effort is a direct follow-on to a 1996 AFIT thesis, *Calibration of the CHECKPOINT Model to the Space and Missile Systems Center (SMC) Software Database (SWDB)* by Capt Karen Mertes. The Mertes study will be referenced extensively throughout this thesis. The pronounced increases in estimating accuracy demonstrated by the Mertes study are part of the motivation for this effort. If the marked improvements in CHECKPOINT’s estimating accuracy through calibration can be duplicated in another environment, specifically the USAF Electronic Systems Center (ESC), this may lend credence to more widespread usage of the CHECKPOINT model throughout DOD.

The CHECKPOINT Model. CHECKPOINT is a commercial proprietary model developed by T. Capers Jones of Software Productivity Research, Inc. (SPR) and is based on actual historical information from approximately 4,700 software projects (SPR, 1996: 1-4). A unique aspect of the CHECKPOINT model is that it is based on *function points*,

which measure software size based on the functionality of the software program, as opposed to the mere size of the software in lines of code (LOC).

Research Objective

The primary objective of this research is to effectively calibrate CHECKPOINT to the Electronic Systems Center (ESC) software database, and this overall objective is supported by three important elements. First, this effort will develop a revised and expanded version of the 1996 CHECKPOINT calibration methodology (Mertes, 1996). Next, the model will be calibrated for several project categories, such as environmental platform, programming language, and development contractor; this calibration procedure will be directed toward estimating development cost in terms of effort. Finally, calibration and validation success will be evaluated for each category using several statistical accuracy measures consistent with other software model calibration efforts.

This research is being conducted to support ESC's objective of obtaining more accurate software cost estimates using the CHECKPOINT model. The CHECKPOINT for Windows Version 2.3.1 User's Guide claims "CHECKPOINT is the software industry's most comprehensive and accurate software estimating package" (SPR, 1996: 1-4). Obviously, such a software estimating tool would be of considerable interest to the Air Force. While the results of the Mertes study are encouraging, and have even been labeled "groundbreaking," there is much attention as to whether its success can be repeated utilizing an analogous methodology to a different environment and its corresponding database of software programs.

The overall research effort will encompass the following activities:

1. Understand the CHECKPOINT model, its functions, and the procedures for generating estimates and templates for calibration.
2. Evaluate the ESC database, which consists of the following steps:
 - a. Eliminate observations lacking complete data required for the calibration process.
 - b. Stratify the database into data sets by category, such as language, platform/application, and contractor/management maturity level. (Several of these concepts will be explained in further detail in Chapter 3, Methodology.)
 - c. Separate the data sets into calibration and validation subsets.
3. Use the CHECKPOINT model to estimate the costs of all projects and determine the default (pre-calibration) accuracy for the calibration and validation data sets.
4. Calibrate the CHECKPOINT model on the selected calibration subsets, employing a revised and expanded CHECKPOINT calibration methodology.
5. Analyze the calibrated model results through various statistical accuracy measures, report improvements in accuracy for all calibration and validation subsets, and state conclusions.

Scope of Research

This research effort will focus on the development aspect of software cost estimating. Despite the fact that software support costs constitute over 70% of software life cycle costs (Ferens, 1992: 4), development costs are by no means trivial. In 1993, DOD software development costs were estimated at over \$30 billion (Christensen and Ferens, 1995: 156). Another reason for this development focus is that the ESC database consists only of software development data for programs.

Importance of Research

This thesis builds upon Mertes' research primarily through the use of the different database for the CHECKPOINT calibration process, providing an important opportunity to support or repudiate the exceptional results of that study. Also, a complete calibration and validation technique will not only be performed, but reported in its entirety. Some of the specific calibration efforts in the Septuagint Study did not perform all necessary steps to calibrate and validate a model, and others may have performed a more comprehensive

effort but did not adequately report the results or document the methodology. It is hoped this research effort will not only bolster the knowledge of model calibrations to specific databases within USAF, but also will provide a template for statistically robust calibrations, which may ultimately yield more accurate estimates.

Since early cost estimates (i.e., in the development phases of a software project) form the basis for fiscal budgetary allocations and other executive decisions, the importance of these estimates to DOD decision makers is paramount. According to Dr. David Christensen of AFIT, once 15% of the scheduled work for a DOD program is completed, there is little or no possibility of recovery from a projected cost or schedule overrun, based on a database of over 700 DOD acquisition programs (Christensen, 1996). Thus, any hope of positively influencing a program based on information gained from an estimate must come in the early stages.

Thesis Structure

This research effort is directed at answering the primary research objective stated above. Ideally, the information gained by fulfilling this objective will not only provide a basis for more accurate software cost estimates at ESC, but will also foster an improved understanding of the parametric model calibration process in general. Chapter 2, *Literature Review*, reviews recent research and publications in the software cost estimating field with an emphasis on accuracy, function points, and model calibration. Chapter 3, *Methodology*, explains how this effort was structured to meet the research objective in accordance with sound research principles. Chapter 4, *Findings and Analysis*, examines the information obtained through the calibration and validation process. Chapter 5, *Conclusions and Recommendations*, draws an overall conclusion

regarding the success of the calibration and validation effort, discusses limitations of the findings, and recommends areas for future research.

Chapter 2

Literature Review

Overview

In 1987, the Defense Science Board studied the causes of software problems (e.g., cost overruns). They concluded that “today’s problems with military software development are not technical problems, but management problems” (Brown, 1996: 7). If software project managers could rely upon consistent, accurate software estimates, at least one such management problem would certainly be improved. This chapter reviews research and literature relevant to software cost estimating and software cost model calibration. Primary areas of focus are the basics of software cost estimation, software cost model calibration, and the ideals for achieving estimating accuracy. Supporting discussion includes a review and evaluation of previous calibration efforts. Also, this chapter will review the concepts, background and current status of function points, and it will provide a general description of the CHECKPOINT model.

Software Cost Estimation

Despite its increased importance, “software cost estimation will never be an exact science. Too many variables—human, technical, environmental, political—can affect the ultimate cost of software and the effort needed to develop it” (Navlakha, 1990: 255).

Brooks' Law provides another example of how software cost estimates may defy reality: Due to the need for additional training and increased communication, "adding manpower to a late software project makes it later" (Brooks, 1975: 14-16). Thus, before one can begin to address the intricacies involved in software cost estimation, some basic concepts underlying this area must be discussed and understood.

What is a Software Cost Estimate?

SPR defines software cost estimating as "predicting the staff, effort, schedule, cost, value, and risk of a project" (SPR, 1996: 20-7). While this is seemingly an accurate definition of an estimate, it excludes one key element that differentiates an estimate from an "educated guess." In his book *Cost Estimation for Software Development*, Bernard Londeix provides an eloquent and more comprehensive definition of a cost estimate:

An estimate, although a judgment, is still an opinion about something. As such, the process of estimating makes use not only of expertise and knowledge but also a set of rules that are universally recognized. The difference between an estimate and an educated guess is that an estimate pretends to be true within certain limits. To arrive at this result, the estimating activity must be based on an estimating method. Estimating and costing always go together, as costing is the purpose of estimating. We cannot cost without estimating. Estimating a software development involves estimating both the size and the cost. By cost we mean the effort spent in man years to develop the software. (Londeix, 1987: 3).

As stated previously, there is one key element of the above citation that hits upon the motivation for this research. A project manager must state "certain limits" for an estimate to be considered a successful management tool; furthermore, these limits are useless without a sound, replicable method to attain them. Therein lies the dilemma facing software cost estimators in DOD and the software cost estimating community at large—the inability to consistently attain useful estimating limits.

What Constitutes a Successful Cost Estimate?

Although the actual criteria will certainly vary with the situation, estimators must have some general idea of what constitutes a successful estimate (one that is useful to management). According to Londeix, a software estimating method is successful when:

1. The early estimate is within $\pm 30\%$ of the actual final cost: This is the accuracy currently obtainable at an early stage of development.
2. The method allows refinement of the estimate during the Software Life Cycle. A higher accuracy can be achieved by monitoring and re-estimating the development each time more information is available.
3. The method is easy to use for an estimator. This enables a quick re-estimate whenever it is necessary; for example, during a progress meeting, the evaluation of alternatives in strategic choices.
4. The rules are understood by everybody concerned. Management feels more secure when the estimating procedures are easily understandable.
5. The method is supported by tools and documented. The availability of tools increases the effectiveness of the method, mainly because results can be obtained more quickly and in a standard fashion.
6. The estimating process can be trusted by software development teams and their management. This helps in gaining the participation of everybody concerned with the estimate. (Londeix, 1987: 3)

Also, an estimating model could be interpreted to be successful based simply on measured predictive accuracy or PRED (*l*). For example, PRED (*l*) limits could define the model as accurate “if the estimates fall within $\pm 25\%$ of the actual cost, 75% of the time” (Conte, Dunsmore, and Shen, 1986: 173). This particular measure of accuracy is used in this research effort and will be discussed in Chapter 3.

Cost Estimating Approaches

There are five principal cost estimating methods considered to be acceptable in the cost estimating community: algorithmic, bottom-up, top-down, expert judgment, and analogy (Boehm, 1981: 342). These cost estimating techniques (as well as two unacceptable alternatives) are discussed in depth in *Software Engineering Economics*, a

seminal work on software cost estimating by Dr. Barry Boehm. The five acceptable techniques (briefly summarized in Table 1) can often be employed in concert to develop a system-level software cost estimate. Each estimating method from Table 1 has inherent capabilities and limitations that have been covered extensively in the literature (Boehm, 1981; Boehm, 1984; Ferens, 1996), as well as each thesis in the Septuagint Study. Rather than revisit these discussions, this review will focus on the capabilities and limitations of a hybrid estimating category: parametric estimating models.

Table 1. Software Cost Estimation Techniques

ESTIMATING METHOD	DESCRIPTION
Acceptable	
ALGORITHMIC	One or more algorithms produces a software cost estimate as a function of a number of variables considered to be the major cost drivers
BOTTOM-UP	Each component of the software job is separately estimated, and the results are aggregated to produce an estimate for the overall job
TOP-DOWN	A cost estimate for the total system is derived; the total cost is then split up among various elements
EXPERT JUDGMENT	Involves consulting one or more experts; relies on their judgment
ANALOGY	Based on actual completed projects—relates the costs of similar past projects to estimate the new project
Unacceptable	
PARKINSON	The Parkinson principle (work expands to fill the available schedule) is used to equate the cost estimate to the available resources
PRICE-TO-WIN	The cost estimate is based on whatever cost is considered necessary to win the contract

(Boehm, 1981: 329-330)

The Origin of Parametric Estimating Models

Since algorithmic and top-down estimating methods are similar in many respects, Ferens has recommended combining these two categories into the category of parametric

models (Ferens, 1996: 29). Since much of the relevant literature uses the terms “parametric” and “algorithmic” interchangeably when referring to computerized estimating models such as COCOMO and CHECKPOINT, algorithmic and parametric models are considered to be synonymous for the purposes of this research. However, as will be discussed later in this chapter, CHECKPOINT, although a parametric estimating model, is not a top-down model.

Parametric models have several capabilities that enamor them to DOD agencies: 1) They are relatively fast and easy to use; 2) They require little input data; 3) They may provide default values in the absence of data; and 4) They can be used very early in a software program’s development cycle (Ferens, 1996: 29).

These capabilities have accompanying limitations, however. First, parametric models are not always accurate (thereby providing the motivation for this research). Despite developers’ claims of accuracy, few of the models evaluated for accuracy have disproved this limitation (many of these accuracy studies are reviewed later in this chapter). Second, the models are often unstable; some input parameters may be overly sensitive, meaning a small change in an input could cause a dramatic change to effort (Ferens, 1996: 29). Third, since program size is a critical input to almost all of parametric models, an inaccurate size input will likely result in an inaccurate cost estimate (Ferens, 1996: 29). In this case, even an otherwise accurate estimating model would be rendered inaccurate by inaccurate inputs. Dr. James Skinner (Assistant Professor of Computer Science at AFIT) concurs, stating “sizing is the weakest link in software estimating” (Skinner, 1997: 6).

Genuchten and Koolen offer three related limitations that are specific to computerized parametric estimating tools:

1. The commercially available models do not originate from the environment in which they are to be used. This is applicable to both location and time.
2. It is extremely difficult to predict the level of certain input variables for a future project (e.g., size).
3. No studies confirm the accuracy and usability of the models; the studies that have been made give very disappointing results. (Genuchten and Koolen, 1991: 38-39)

The limitations stated by Ferens and those by Genuchten and Koolen show considerable overlap in that they reflect a prevalent opinion in the software research community regarding parametric models. These limitations or others along these themes are widely cited in some form across much of the relevant literature. Also, it is interesting to note that six more years of accuracy-related studies have done little to disprove Genuchten and Koolen's third limitation. The 1996 Mertes CHECKPOINT study may stand alone as the only research to refute this claim (Mertes, 1996).

Macro-Estimation v Micro-Estimation

Previously in this chapter, parametric estimating models were mentioned as related to or synonymous with top-down estimating models. Actually, the major commercial software estimation tools will utilize one of two major varieties of internal estimating logic: macro-estimation or micro-estimation (Jones, 1996: 20). Obviously, these terms are related to Boehm's top-down and bottom-up approaches; however, since macro- and micro-estimation are discussed purely in the context of computerized parametric estimating tools, these definitions may be more relevant.

Macro-Estimation. "The estimating equations are aimed at completed software projects. Once the effort and schedule for the overall project are predicted, the estimate is divided into relative amounts for each phase, such as requirements, design, coding, and

so on” (Jones, 1996: 20). Jones states that macro-estimation is easier to perform, but that errors are propagated throughout the entire estimate. For example, an error of 15% will be present in all phases, since the entire project was the basis for the estimate (Jones, 1996: 20). He recommends macro-estimating models for quick, preliminary estimates only (Jones, 1996: 20).

Micro-Estimation. “The estimating equations deal with specific activities, such as requirements, design, coding, unit test, integration, user documentation, and so on” (Jones, 1996: 20). The activity-specific estimates are then aggregated to obtain the project estimate. Micro-estimation is more complex, but offers a main advantage; errors will be contained to specific activities since each activity is estimated separately, thus resulting in a more precise estimate (Jones, 1996: 20). Jones recommends micro-estimating models “for serious business purposes, such as software development contracts” (Jones, 1996: 20).

The Need for Model Calibration

There have been a number of accuracy-related studies that demonstrate a parametric model’s inability to provide accurate estimates when utilized in a different environment from that in which the model was developed (Genuchten and Koolen, 1991, 38). One prominent study is summarized below. The results of these studies, supported by the bad experiences of USAF software development program offices at the Space and Missile Systems Center (SMC) and Electronic Systems Center (ESC), provide ample basis to convince any project manager of the need for calibration. (This realization provided the motivation for SMC’s sponsorship of the AFIT 1995-1996 Septuagint Study.)

A 1987 study by Chris Kemerer on model estimating accuracy concluded that algorithmic models, when used outside their original environments, do not work very well uncalibrated (Kemerer, 1987: 427). In this study, average error rates (MMRE, as detailed in Chapter 3 of this thesis) ranged from 85% to 772% of project actuals. Kemerer theorized that this variation is most likely caused by the degree to which the productivity rates in the model development data match the productivity rates in the target environment (Kemerer, 1987: 427). This study illuminates the need for the Air Force, if it continues to utilize algorithmic estimating tools, to collect detailed historical data on its projects in order to calibrate the models to specific development environments.

Not all researchers believe the right software cost model, a sound calibration procedure, and an extensive historical database will resolve the technical problems associated with software cost estimation (SCE) for development efforts. According to Heemstra, the human aspects have more influence on delivering the software in time and within budget than the use of rigid calculations (Heemstra, 1992: 627-639).

In addition, it is not universally believed that an uncalibrated model is completely useless to the estimating process. Skinner sees a fundamental benefit to the use of computerized parametric estimating models, despite their questionable accuracy and regardless of calibration/validation success. He believes that the software estimation process benefits simply from the structure provided by the model (Skinner, 1997). Kemerer agreed with this opinion, in that the act of providing inputs to the models requires the software project manager to carefully consider many aspects of the upcoming project (Kemerer, 1987: 427).

Previous Calibration/Validation Research

Since calibration will ideally improve parametric model estimating accuracy, there have been several independent research efforts attempting to demonstrate the effectiveness of model calibration. Generally, the independent studies (i.e., those not sponsored by the model developers) have shown the models to be accurate to within 25% of cost or schedule only about half of the time (Ferens and Christensen, 1995: 1). This section briefly discusses some early calibration efforts, and then addresses the AFIT Septuagint Study in more detail.

The Thibodeau Study

In 1981, Robert Thibodeau of General Research Corporation calibrated nine software cost models to three discrete software databases, including an Air Force database of COBOL MIS programs. The results showed that model calibration could improve estimating accuracy by a factor of five. He also concluded that “model performance, given the limitations of procedures, definitions and understanding of cost driving factors, is very much environment dependent” (Thibodeau, 1981: 6-7 - 6-13).

An important aspect of the Thibodeau study is that upon initial evaluation of his analyses of the models’ accuracy, some model developers disparaged the data quality, citing deficiencies such as inconsistent SLOC definitions, incomplete data records, and small database sizes (Thibodeau, 1981: 6-12). He rebutted their evaluation by citing instances where most software development data, including the databases used to develop cost models, may be even more suspect. He recommended that the Air Force increase software development data requirements for all contracted efforts (Thibodeau, 1981: 7-4).

Fifteen years later, however, similar data quality issues still concern the DOD software cost estimating community.

The Illinois Institute of Technology Study

In 1989, the Illinois Institute of Technology Research Institute (IITRI) calibrated seven parametric models including SPQR/20, the forerunner of CHECKPOINT. The models were calibrated using a database of eight Ada projects from commercial and government environments, including three from Electronic Systems Division (ESD, which became ESC in 1994). It is important to note that all projects were sized in SLOC (defined in this study as statements terminated by a semicolon); thus, SPQR/20 was calibrated using SLOC sizing data (IITRI, 1989: 3-4). The results showed only slight overall improvements in accuracy, even when validating the calibrated models with the same data sets. The IITRI study demonstrated SPQR/20 to be one of two models that were most accurate for Ada command and control applications (IITRI, 1989: 3-21).

A common limitation of the Thibodeau and IITRI research efforts (which is also a limitation of many accuracy studies) is that both studies utilized the data sets from the calibration procedure to validate the models (Ferens, 1996: 32).

The Kemerer Study

This 1987 study (already discussed briefly) evaluated the estimating accuracy of four algorithmic models (SLIM, COCOMO, Albrecht's Function Points, and ESTIMACS) on a database of 15 business data-processing projects. This study utilized Magnitude of Relative Error (MRE) and linear regression to test the accuracy of these models. Kemerer found that the best of these models, when calibrated, could explain up to 88% of the behavior of the actual effort in the data set (Kemerer, 1987: 427). He concluded this

finding alone justifies the use of an algorithmic estimating method for an organization with sufficient data to enable calibration to his or her environment. An important aspect of the Kemerer study, as it is related to this research effort, is that it successfully validated the original Albrecht Function Point model on an independent data set (Kemerer, 1987: 427).

Limitations of Research. Several limitations of the Kemerer study must be noted. First, the 15-project database originated from a single firm, which severely limits generalizability of the results. Second, the SLOC-based size estimates for the projects were obtained *ex post* (after the fact); obviously, such accurate measures of size could not be expected for a new project. Third, it is unclear from the published research whether the data set used for validation was also the calibration data set (Kemerer, 1987: 427-428).

Perhaps the most important limitations of this study are associated with Kemerer's failure to completely test several assumptions associated with ordinary least squares regression (or at least document the results of these tests). Matson *et al.* state that despite the high R^2 , "the model fit is inappropriate in several respects" (Matson, Barrett and Mellichamp, 1994: 278). For instance, Kemerer detected an outlying data point that highly influenced the regression model, but he included it based on the rationale that excluding this observation would lower the R^2 for all models (Kemerer, 1987: 426). Matson *et al.* pointed out that "reliance on R^2 alone for assessing the appropriateness and strength of a regression model is erroneous" (Matson, Barrett and Mellichamp, 1994: 280). In addition, Kemerer gave no mention to analysis of residuals or error terms, which is an important model diagnostic (Matson, Barrett and Mellichamp, 1994: 278). Based

on their evaluation, Matson *et al.* concluded that Kemerer's "gross violation of model assumptions renders the resulting inferences virtually meaningless" (Matson, Barrett and Mellichamp, 1994: 278).

The Ourada Study

A 1991 AFIT thesis by Gerald Ourada calibrated and validated four models (REVIC, SASET, SEER-SEM, and COSTMODL) to the Space Systems Division (SSD, which is now SMC) software database. Ourada identified 28 data points suitable for the research effort, and separated the data into two 14-point subsets—one for calibration and one for validation. Ourada's research, although it showed that these models were not accurate for estimating other projects in the SSD database, is still important for three reasons (the third reason is particularly relevant to this research effort):

1. The study used separate data subsets for calibration and validation (which allows more generalizable inferences about model accuracy from the results).
2. It utilized similar statistical measures of accuracy to those utilized in the bulk of the Septuagint Study and in this research.
3. Ourada discovered several potential problems regarding the 1991 edition of the ESC database, such as incomplete data for completed projects. Also, projects never completed had been "estimated" and included in the database as actual data points. (Ourada, 1991: 4.1-4.2)

Limitation of Research. Ourada had originally intended to use the ESD database for the calibration/validation procedure. However, Ourada judged the database to be unusable because of the data unreliability discussed above, and he fully justified this decision in his thesis. Once Ourada obtained a credible database from SSD, he employed an overall calibration/validation methodology similar to that of the Septuagint Study, which his thesis preceded by five years.

The Septuagint Study

As stated previously, this research effort follows in the footsteps of a group of masters' theses performed at AFIT during the years 1995-1996; these seven related research efforts were nicknamed the Septuagint Study by Professor Daniel Ferens, the thesis advisor for all seven efforts. Due to the somewhat standardized presentation format, methodology, and end user motivations for all research therein, the Septuagint Study is the primary focus of this review of previous calibration efforts.

The SMC Software Database (SWDB)

In 1994, SMC, located at Los Angeles AFB, had recently expanded and automated a massive database of completed software development projects. The project information was obtained from various government agencies including NASA, SMC projects, and Air Force Materiel Command (AFMC), as well as industry sources including major aerospace companies and software estimating model developers (Southwell, 1996: 12). The SMC database had been specifically designed to enable use with four software programs widely used in DOD: REVIC, SEER-SEM, PRICE-S, and SASET. Intense efforts were made to standardize the data collection and entry procedures, and to ensure consistent interpretation of all input data (for example, all SLOC counts were defined as logical lines, as opposed to physical lines) (Southwell, 1996: 12).

SMC was interested to see what improvements to estimating accuracy could be attained by model calibration, so in August 1994, it requested AFIT calibrate five cost models most widely used in the Air Force to its new SWDB. Thus, five AFIT graduate students calibrated PRICE-S, REVIC, SASET, SEER-SEM, and SLIM to the 1994 edition of the SMC SWDB. One year later, two additional models, SOFTCOST-R and

CHECKPOINT, were calibrated at AFIT, also using the 1994 SMC SWDB. These seven independently researched but related efforts comprise the Septuagint Study, the results of which are summarized in Table 2. These results were gleaned directly from each independently written AFIT thesis in the study and are completely referenced as such in this researcher's bibliography.

The SMC SWDB consisted of 2,638 software projects, although it was discovered that only 444 included values for effort (Southwell, 1996: 12). For the 1995 research efforts, the database was then stratified into six categories: 1) Unmanned Space, 2) Military Avionics, 3) Missile, 4) Military Mobile, 5) Military Ground - Command and Control, and 6) Military Ground - Signal Processing (Mertes, 1996: 23). Three additional categories were considered in the 1996 efforts: 1) COBOL Projects, 2) Management Information Systems (MIS), and 3) Ground in Support of Space. The project categories used in each research effort are shown in Table 2 under the heading "Application Type."

Separation into Calibration and Validation Subsets

The 1995 studies divided each data set into calibration and validation subsets as follows:

1. If there are fewer than 8 data points, then use all the points for calibration only.
2. If there are at least 8 but fewer than 12 data points, then use 8 for calibration and the rest for validation.

If there are at least 12 data points, then use 1/3 for validation. (Ferens and Christensen, 1995: 7). On the advice of SMC and MCR, this methodology was revised for the 1996

Southwell and Mertes studies as follows:

1. If the data set contains fewer than 8 data points, then the data set is unusable to generate meaningful results and is eliminated.
2. If the data set contains at least 8 data points, one-half are used for calibration and one-half are used for validation. (Mertes, 1996: 29)

Evaluation of Results

When evaluating the results of the Septuagint Study as shown in Table 2, it is important to consider several inconsistencies or limitations which may hinder across the board evaluation of the models' estimating accuracy.

Table 2. Summary of the AFIT Septuagint Study

Study	Cost Model	Application Type	Cal.	Val.	Default Accuracy			Validated Accuracy			
					MMRE	RRMS	Pred (0.25)	MMRE	RRMS	Pred (0.25)	
Galonsky (95)	PRICE-S	Mil Ground	X	X	not reported			0.52			
		Unmanned Space	X	X	not reported			0.36			
		Missile	X		not reported			0.75			
		Mil Mobile	X		not reported			0.38			
Kressin (95)	SLIM	Mil Ground - MIS	X		0.962	n/r	0.00	0.157	n/r	0.83	
		Mil Ground - All	X		n/r	n/r	n/r	2.166	n/r	0.08	
		Command & Control	X	X	0.621	n/r	0.00	0.666	n/r	0.00	
Rathmann (95)	SEER-SEM	Avionics	X	X	0.923	1.472	0.00	0.243	0.240	1.00	
		Command & Control	X	X	0.531	1.031	0.43	0.311	0.296	0.29	
		Signal Processing	X	X	1.440	1.082	0.29	2.092	1.610	0.43	
		Mil Mobile	X	X	2.802	3.711	0.11	0.462	0.342	0.25	
Vegas (95)	SASET	Mil Ground	X	X	10.04	n/r	0.00	5.820	n/r	0.38	
		Unmanned Space	X	X	5.54	n/r	0.23	0.940	n/r	0.00	
		Avionics	X	X	1.760	n/r	0.00	0.220	n/r	1.00	
		Military Mobile	X	X	5.610	n/r	0.25	3.570	n/r	0.00	
Weber (95)	REVIC	Mil Ground	X	X	1.21	1.13	0.00	0.86	0.68	0.00	
		Unmanned Space	X	X	0.44	0.62	0.50	0.32	0.34	0.50	
Mertes (96)	CHECKPOINT	MIS - COBOL		X	0.542	0.101	0.67	0.018	0.010	1.00	
		(f.p.)	Mil Mobile - Ada		X	1.384	0.412	0.25	0.192	0.057	0.75
		(f.p.)	Avionics		X	0.817	0.685	0.50	0.158	0.111	0.75
		(sloc)	Command & Control		X	0.193	0.145	0.50	0.165	0.156	0.50
		(sloc)	Signal Processing		X	0.090	0.081	1.00	0.090	0.081	1.00
		(sloc)	Unmanned Space		X	0.048	0.050	1.00	0.040	0.055	1.00
		(sloc)	Ground (spt. space)		X	0.050	0.058	1.00	0.050	0.058	1.00
		(sloc)	COBOL Projects		X	0.050	0.051	1.00	0.049	0.051	1.00
Southwell (96)	SOFTCOST-R	Mil Ground	X	X	1.895	3.433	0.00	0.519	0.870	0.83	
		Signal Processing	X	X	0.430	0.612	0.11	0.282	0.634	0.44	
		Unmanned Space	X	X	0.557	1.048	0.20	0.480	0.923	0.20	
		Ground (spt. space)	X	X	2.734	3.125	0.13	1.802	1.966	0.20	
		Military Mobile	X	X	0.635	0.514	0.20	0.420	0.395	0.40	
		Avionics	X	X	0.713	0.758	0.20	0.846	0.568	0.20	

(Note: Table 2 was constructed through a collaborative effort between Capt David Marzo, Lt Wayne Bernheisel, and the author in the Spring of 1997 and will also be reported in the 1997 Marzo and Bernheisel theses, heretofore unpublished.)

First, the separation of data into calibration and validation subsets followed the two differing procedures described above. Thus, employing the 1995 methodology, it was

possible to obtain validation results based on one data point. For example, the SEER-SEM results indicate a PRED (0.25) improvement from 0% with the default model to 100% in the calibrated model on the avionics validation subset (Rathmann, 1995: 34). While this seemingly indicates a remarkable improvement in accuracy, any claims are severely hindered by the fact that this result is based on one validation data point. This type of result prompted the 1996 methodology, which required all validation subsets to contain at least four observations.

Second, not all of the data sets in each effort contained enough observations to enable validation of the calibrated model using a separate validation subset. In Table 2, validation using the calibration data subset is indicated by an X in the “Cal.” column. Validation using an independent validation subset is indicated by an X in the “Val.” column. Thus, Table 2 shows that some models were validated for certain application types using only the calibration data sets. As stated previously in this chapter, this practice limits the efficacy of conclusions regarding the applicable model’s accuracy for these application types.

Third, not all students employed the same breadth of statistical tests for accuracy, despite efforts to standardize the analysis of all research results. Incomplete accuracy results are indicated by “not reported” or “n/r” in Table 2.

A fourth limitation of this study is that not all of the SMC database, even the portion that contained effort information, was suitable for each model. Thus, the students were sometimes forced to eliminate portions of the database from consideration, which often severely reduced the overall usable data points and in turn the statistical robustness of results (Ferens and Christensen, 1995: 9).

Despite these limitations, some elements of the Septuagint Study provide viable evidence supporting accuracy improvements from model calibration. This supports the assertion that a cost analyst may frequently obtain better software estimates with a calibrated model (Ferens and Christensen, 1995: 15). Since the CHECKPOINT study is particularly relevant to this research, it will be discussed in further detail.

The Mertes CHECKPOINT Study

Although the Septuagint Study as a whole showed only mixed results regarding improvements in model estimating accuracy, the 1996 CHECKPOINT calibration effort by Capt Mertes yielded marked improvements in estimating accuracy for the function point-based data sets. As can be seen in Table 2, MMRE for the three applications with function point sizing data improved significantly (e.g., an improvement of approximately 25:1 after calibration for MIS, Military Mobile, and Avionics projects).

There are two major possibilities why the SLOC based applications in Table 2 did not reflect this improvement. First, the default estimates were already quite accurate, leaving less margin for improvement. For example, the default model MMRE was only 9% for the Military Ground - Signal Processing application (Mertes, 1996: 89). Second, although CHECKPOINT accepts SLOC-based sizing data, it uses SLOC/function point conversion tables to estimate effort, then converts back to SLOC as an output (these tables are discussed later in this chapter). Thus, the accuracy of resultant estimates may be limited by the accuracy of these conversion ratios, even after calibration.

A primary limitation of this study is the lack of documentation regarding the results of the calibration procedure on the calibration data set. Although these results are not suitable for inferences of accuracy for new projects, they can still provide useful insight

into the success of the model calibration. For instance, a model might be considered unsuccessfully calibrated if it cannot meet the $\pm 25\%$ accuracy, 75% of the time criterion for even the calibration data set. The 1996 SoftCost-R calibration effort included complete documentation for both the calibration and validation subsets of each application type (Southwell, 1996). Using those results, Capt Southwell was able to draw conclusions regarding the success of the calibration for each application type.

The Mertes study departed from the other efforts in that some application types were also calibrated for schedule estimation. Since the bulk of the Septuagint Study (as well as this research effort) calibrated only to cost (i.e., effort), schedule results are beyond the immediate scope of this research effort. Thus, schedule calibration results are not reported in Table 2 and are not discussed.

Finally, this study provided an important contribution that is not reflected in Table 2: a step-by-step model calibration procedure specific to CHECKPOINT (Mertes, 1996: 33-34). This research will utilize a revision of the aforementioned procedure, which will be discussed in detail in Chapter 3.

What is a Function Point?

The CHECKPOINT User's Guide defines function points as "a concept for computing software size from five attributes: external inputs, external outputs, external inquiries, external interfaces, and internal files" (SPR, 1996: 1-3). According to Garmus and Herron, "Function points are a vehicle to measure the functionality being delivered to the end-user. They provide a sizing mechanism that allows us to measure that functionality in specific and consistent terms" (Garmus and Herron, 1996: 16). Allan Albrecht, the inventor of function points, described them as a sort of 'Dow Jones Index'

of the size of the system (Symons, 1991: 15). When comparing these definitions, it becomes apparent that function points are defined more in terms of what they (ideally) do, not what they are.

Function points have two primary uses. First, they provide an objective measure for software project managers to assess levels of productivity (Matson, Barrett, and Mellichamp, 1994: 276). The second use of function points is central to this research; they are used in the estimation of software development cost.

History of Function Points

Prior to the late 1970's, the standard metric for software output was "cost per line of source code," which does not correlate to the economic definition of output, "goods or services produced per unit of labor and expense" (SPR, 1996: 17-2). According to Jones, the problem with the SLOC metric was that it did not adequately capture the substantial percentage of fixed costs associated with a development effort (SPR, 1996: 17-2). As more powerful programming languages were used, the total number of lines necessary for a given program would decrease. But when the fixed costs (e.g., requirements, user documents, specifications) were added to the total program cost, the cost per line of code metric would increase instead of decrease (Jones, 1995: 1).

In the late 1970's Albrecht took the position that there should be a measure of economic output for software that would be valid for all languages, and should represent topics of concern to the users of the software. In short, he wished to measure the functionality of the software. (SPR, 1996: 17-4)

Thus, when Allan Albrecht of IBM published his Function Point metric in 1979, there was finally a definition of economic output for a software project (Mertes, 1996: 17).

Albrecht believed that the measurable functionality of software consisted of five items: 1) the inputs to the application, 2) the outputs from it, 3) inquiries by users, 4) data files that would be updated by the application, and 5) interfaces to other applications (SPR, 1996: 17-4). Using trial and error, he developed empirical weighting factors for the five items, as well as a complexity adjustment factor of $\pm 25\%$. The number of external inputs (EI) was weighted by 4, external outputs (EO) by 5, external inquiries (EQ) by 4, internal data file updates (ILF) by 10, and external interfaces (EIF) by 7 (Mertes, 1996: 18). Thus, the basic function point equation is:

$$\boxed{\text{BFP} = (4\text{EI} + 5\text{EO} + 4\text{EQ} + 10\text{ILF} + 7\text{EIF}) * \text{Complexity Adjustment} \quad (1)}$$

where BFP is the number of Basic Function Points.

In 1986, the non-profit International Function Point User's Group (IFPUG) was formed to assist in transmitting data and information about function points (SPR, 1996: 17-6). In 1990, IFPUG published the Function Point Counting Practices Manual, available to all IFPUG members. This guideline provided the rules for function point counting, enabling users to size software programs through analysis of the functionality of the software to be developed (SPR, 1996: 17-6). "Function point counting is one of the fastest growing management techniques in the software industry today. It is used by a variety of organizations across numerous industry types" (Garmus and Herron, 1996: 1).

Function Point Analysis (FPA)

FPA is merely a formal term to describe a methodology for counting function points. The IFPUG World Wide Web Home Page defines FPA as "a software sizing measure that quantifies the functions contained within software. From this measure, cost and productivity information can be readily derived" (IFPUG, 1996).

Brian Dreger authored what is considered to be the first practical layman's guide and instruction to FPA. *Function Point Analysis* was published in the fall of 1989 and was generally accepted as a standard for counting function points (Dreger, 1989). "Such companies as AT&T, Motorola, Hewlett-Packard, and Boeing were among the early users of the methodology" (Garmus and Herron, 1996: 2).

Feature Points

In 1986, SPR developed an experimental method for applying function point logic to system software such as operating systems, telephone switching systems, etc. (Jones, 1995: 4). Since function points were originally invented for MIS systems, function points were not necessarily perceived as optimal for the following applications:

- Real time software, such as missile defense systems
- Systems software, such as operating systems
- Embedded software, such as radar navigation packages
- Communications software, such as telephone switching systems
- Process control software, such as refinery drivers (Jones, 1995: 4)

The problem was that Albrecht's function points appeared to underestimate projects that were high in algorithmic complexity, but sparse in inputs and outputs (SPR, 1996: 17-7). As a result, a new parameter, number of algorithms, was introduced to the basic equation and assigned a default weight of 3. The SPR Feature Point Method also reduces the empirical weights for internal files (ILF) from 10 down to 7. Thus, when applied to more complex forms of systems software, feature point counts will be significantly higher (Jones, 1995: 5).

It is apparent that the Feature Point Method may offer more applicability to DOD type applications than more conventional FPA. However, feature points are still experimental and are undergoing field trials (SPR, 1996: 17-10). The Feature Point

Method may never be formalized, since Capers Jones of SPR supports IFPUG efforts to develop a single standard for FPA that incorporates facets of the Feature Point Method (Jones, 1997).

Current Status of Function Points

The propagation of function point variants (e.g., Albrecht's Function Points, IBM Function Points, SPR Function Points & Feature Points, and Mark II FPA) has led to an effort by IFPUG to develop a comprehensive, worldwide ISO standard for counting function points (IFPUG, 1997). However, it is unknown when such a standard will be adopted, and which function point variant the new standard would most closely resemble.

The SLOC/Function Point Debate

Opinions regarding the use of function points in software cost estimating range from ringing endorsements to broad condemnation. The book *Measuring the Software Process* provides the following perspective on function points:

Function points are not a panacea for the ills of the software community. Often, instead of describing what function points do, consultants find themselves describing what function points don't do. This is not the result of any shortcomings in the function point counting process itself but more from general misunderstandings that exist in the marketplace today. (Garmus and Herron, 1996: 2-3)

Capers Jones of SPR claims that function point metrics are far superior to SLOC-based metrics for expressing normalized productivity data. "As real costs decline, cost per Function Point also declines. As real productivity goes up, Function Points per person month also goes up" (Jones, 1995: 3). It is important to recall that Albrecht's basic function point metric was developed to solve measurement problems with classical MIS software, and may not be suitable to all software applications (SPR, 1996: 17-7).

According to Tom DeMarco, there is no general solution for all environments. He does assert, however, that “lines of code is not a good basis for cost estimating,” and that “it is much better to measure functionality.” Finally, he adds, “anyone who wants to estimate cost must collect data for his/her own environment” (DeMarco, 1996).

Function Point Research

The Kemerer study found that the non-SLOC models (Albrecht’s Function Points and ESTIMACS) were more accurate than SLOC-based models (COCOMO and SLIM), in terms of MRE results. He attributed this to the similarity between the non-SLOC models and the target database used in the study, rather than an inherent advantage of non-SLOC measures over SLOC (Kemerer, 1987: 427). However, the regression results showed the SLOC-based models to be more accurate. These results were tempered by the fact that “the SLOC counts (for these model inputs) were obtained *ex post*, and were likely far more accurate than SLOC counts predicted before a project begins” (Kemerer, 1987: 427).

Kemerer also performed regression analyses on function point counts as a predictor of KSLOC. He concluded that the R^2 of 0.751, which was similar to Albrecht’s initial claims, “is likely to be good enough to be of use to the software manager” (Kemerer, 1987: 427). Kemerer regarded this result as an independent validation of the Albrecht function point model.

In 1994, Kemerer’s conclusions regarding function point analysis were rebutted by Matson *et al.*, who stated that Kemerer’s findings were weak in two areas. First, they revealed several inadequacies in the regression analysis procedure (as previously discussed in this chapter), rendering the models suspect (Matson, Barrett, and

Mellichamp, 1994: 278-280). Second, the models were based on small samples with large mean squared errors (MSE's). This resulted in regression coefficients with extremely wide confidence intervals (Matson, Barrett, and Mellichamp, 1994: 280).

The Gurner Study. This 1991 AFIT thesis evaluated the estimating accuracy of three function point based models on two commercial databases (Gurner, 1991). The Software Program Acquisition Network Simulations (SPANS) model, developed by Tecolote Research, Inc. under contract to Standard Systems Center (SSC) at Gunter AFB, was the focus of the study. Two other models were used for comparison: CHECKPOINT (SPR, Inc.), and Costar (based on Boehm's COCOMO). Gurner utilized regression analysis, the Wilcoxon Test for bias, and percentage error to evaluate the models. While the regression analysis showed a relatively equal relationship ($R^2 = 0.80$) between estimates and actuals for all three models, CHECKPOINT showed the least bias (biased high at 95% confidence level), and it exhibited the lowest percentage error at 0.21. Gurner also reported MMRE for the three models, and CHECKPOINT was again significantly lower at 0.46 (Gurner, 1991: 43).

Although CHECKPOINT outperformed the other models in the Gurner study, an MMRE of 0.46 is not generally considered to be acceptable, as will be explained in Chapter 3. However, it also must be stated that Gurner did not calibrate the models to the applicable databases prior to obtaining the estimates; rather, he evaluated the models' default accuracy for this study.

Other studies have shown that function points-based models can predict software costs in commercial applications; these studies reported estimating accuracy using either Conte's criteria of $MRE \pm 25\%$ in 75% of cases, or analysis of variance from the results

of linear regression. One such study showed that function points accounted for 74-82% of the variation in software development effort for integrated computer aided software engineering (ICASE) (Subramian and Zarnich, 1996: 143). Conversely, function points may not be ideal for all business environments. A study by Gao and Lo used FPA to size computer assisted learning (CAL) systems for development effort estimation; the results showed that function points could not effectively size CAL systems for effort estimation without the introduction of additional considerations (Gao and Lo, 1996: 212).

Language Levels

A critical input to project sizing is source code language. All software languages conform to a specified “language level,” which originated in the 1960’s and was defined as the number of statements in Basic Assembly Language (BAL) required to equal one statement in some other source language. Thus, COBOL was given a level of 3.00, since it took three Assembly statements to equal one COBOL statement. The CHECKPOINT User’s Guide defines language level as the number of source statements it takes to implement one function point. Under this definition, COBOL averages about 106 statements per function point (SPR, 1996: 4-20).

This brings to light an important element of this research effort, in that all software projects in the ESC database are sized in terms of SLOC. No function point data is available in the ESC database. This internal conversion also requires the source code language, so the appropriate SLOC/function point ratio is selected from the SPR Programming Language Table, version 6.3 (SPR, 1996: 17-21 - 17-33). This language table contains over 400 programming languages, and it includes the 10 languages most commonly used throughout DOD.

The validity of this table and other language tables has been questioned by independent researchers, and even SPR admits that SLOC/FP conversion is more accurate for some languages than others. For example, the range of ratios for Ada to function points is relatively narrow at 71-73 lines per function point across three language tables (Ferens, 1997). However, the range for COBOL can vary as much as \pm 50% (SPR, 1996: 17-7).

A 1992 AFIT study by Henderson reported an even wider variability for COBOL, with SLOC/FP ratios ranging from 13 to 165 across three databases (Henderson, 1992: 96-97). As a result, he concluded “these conversion factors should only be used on programs that are very similar (same developer, same time frame, or same type of application) to the database from which they were developed” (Henderson, 1992: 97). Ferens states that based on these findings, SLOC/FP ratios may be meaningless (Ferens, 1997). Consequently, if SLOC/FP ratios are not known with perfect certainty, then using SLOC as an input inserts another potential source of error into the estimate, thereby increasing the overall uncertainty associated with the resulting estimate.

Despite the continued widespread use of the function point methodology, it is an unfortunate fact that function points have not been widely used in DOD, due to the aforementioned doubts regarding function points’ applicability for use with real time systems. Consequently, there is a dearth of function point based sizing data on DOD projects (Skinner, 1997). Thus, the CHECKPOINT model, which specializes in software estimates based on function points and now offers options for military applications, is still questioned when used with SLOC-based military data.

CHECKPOINT Version 2.3.1

CHECKPOINT for Windows Version 2.3.1 is a computerized, parametric software cost estimating model that “integrates sizing, planning, scheduling, estimating, measurement, risk analysis, value analysis, and technology assessment in a single package” (SPR, 1996: 1-3). The original CHECKPOINT for Windows, Version 2.2.0, was released in early 1993 and has been updated five times since then. CHECKPOINT evolved based on the strengths of its predecessor, SPQR/20 (Software Productivity, Quality, and Reliability), which was also developed by Capers Jones in 1985 (Jones, 1996: 22).

According to the CHECKPOINT User’s Guide, as well as with Jones’ definitions, CHECKPOINT is a micro-estimator, “using input variables to produce size, scope, and production rate estimates for selected tasks. Phase effort is never measured directly, only as a sum of the member tasks” (SPR, 1996: 3-13b). CHECKPOINT’s algorithms are derived from empirical measurements of approximately 4,000 software projects (SPR, 1996: 12-3).

CHECKPOINT Features

Sizing Logic

CHECKPOINT supports three alternative types of software sizing metrics: 1) function points, 2) feature points (systems and embedded software), and 3) source code statements. An important detail regarding CHECKPOINT’s sizing logic is that it will always internally revert to functional metrics. Therefore, even if the sizing input is

entered in terms of SLOC, CHECKPOINT will convert the input to function points using a technique called “backfiring” (SPR, 1996: Appendix A).

An important distinction regarding CHECKPOINT is that it handles complexity differently than Albrecht’s equations. Albrecht’s complexity adjustment factor consisted of 14 aspects of complexity that were rated from 1 to 5 and then added (Symons, 1991: 18). CHECKPOINT handles complexity in a different manner, using just three factors rated from 1 to 5: Problem Complexity, Code Complexity, and Data Complexity. The CHECKPOINT method of complexity adjustment thus allows for more variability ($\pm 40\%$) than Albrecht’s basic complexity adjustment (SPR, 1996: 4-12). For example, Problem and Data Complexity are used in calculating adjusted SPR Function Points and Feature Points. IFPUG Function Point counts are not affected by these parameters. Code Complexity is used in “backfiring,” which establishes a relationship between source code size and function points or feature points. All three complexity responses are used in addition to adjust effort and schedule relationships (SPR, 1996: 4-12).

CHECKPOINT is calibrated through the creation and use of templates, which represent the model’s internal means to incorporate organization-specific historical data into internally developed knowledge bases for estimation of quality and productivity results (SPR, 1996: 4-3).

CHECKPOINT also features a Quick Estimate mode that is intended for projects so early in development or so uncertain that only a few known facts exist. “Even if little or nothing is known about a software project, a Quick Estimate can provide useful information in less than a minute” (SPR, 1996: 4-5). Although CHECKPOINT offers an extremely detailed estimating mode with over 100 inputs, the Quick Estimate mode was

used for this calibration effort. This is primarily because the ESC database lacks sufficient information to perform CHECKPOINT detailed estimates; the contents of the ESC database will be discussed in further detail in Chapter 3.

Summary

Throughout the literature reviewed in this chapter, it is made abundantly clear that there are a myriad of management, technical and measurement difficulties relevant to software cost estimating and software cost model calibration. Software sizing has been cited as both a critical input to virtually all parametric cost estimating models (if accurate estimates are desired), as well as one of the most difficult inputs to measure at the inception of a project. Thus, the software estimator is left with a dilemma: Parametric models, both SLOC and function point-based, are attractive tools for use in the early stages of a project. Yet both types of sizing approaches are rife with difficulties, especially in these early stages.

Regardless of the sizing approach employed by the parametric model of choice, several studies reviewed in this chapter showed that uncalibrated parametric estimating tools were inaccurate for a wide range of environments, applications, and languages. Many studies also demonstrated that calibration to a relevant database may improve model accuracy. Thus, it is hoped this research can build upon these findings to calibrate CHECKPOINT to the ESC database following a sound, well-documented, and replicable methodology.

Chapter 3

Methodology

Overview

If a fundamentally sound estimating methodology is established, then “software project estimation can be transformed from a black art into a series of systematic steps that provides estimates with an acceptable degree of risk” (Navlaka, 1990: 255). In keeping with the above theme, the main objective of this chapter is to provide a logical, well-documented description of the complete methodology employed in this research effort. Ideally, the goal is to enable replication and/or revision of this methodology for future research, as well as practical usage.

First, this chapter provides a brief background of the ESC database of defense software development projects; also, underlying assumptions regarding the data are stated. The next section discusses the procedures used to stratify these projects into data sets by application type, and then into calibration and validation data subsets. The methodology employed to calibrate the CHECKPOINT model to the ESC database is largely based on a procedure developed by Capt Mertes (Mertes, 1996: 31-34). The actual CHECKPOINT calibration procedure, refined for this effort, is described in a step-by-step format. Finally, this chapter describes the complete statistical analyses that were utilized to evaluate all default, calibration, and validation results.

The USAF Electronic Systems Center Software Database

Data Description

¹The current version of the ESC database is a collection of 52 completed software development projects from 32 defense contractors and dates back to 1974. These software projects were developed as either stand-alone efforts or to be integrated with military systems procured by ESC at Hanscom Air Force Base. Typical ESC systems involve high-technology communications and signal processing functions such as those provided by the Airborne Warning and Control System (AWACS). The data have been collected through a collaborative effort between the developing contractors, ESC and the MITRE Corporation (a government support contractor located near Hanscom AFB) (Wells, 1997).

ESC has recently updated its software database and plans to use it to validate several parametric estimating models including REVIC, PRICE-S, SEER-SEM, and COCOMO 2.0. ESC is also using the database to derive its own software cost estimating relationships (CERs) (Wells, 1997).

Detailed information for each of these projects is provided at the computer software configuration item (CSCI) level. A CSCI can be considered a mini-project in that each CSCI is developed to perform a distinct end-use function (Ferens, 1997). In all, the 52 project database consists of information for 312 CSCIs. The size of these CSCIs ranges from 411 Executable Deliverable Source Instructions (EDSI) to 448,523 EDSI. The database contains no function point-based sizing data for projects.

The data is provided in a *Microsoft Excel* spreadsheet. For each CSCI, over 60 columns of information are provided and include information such as the contractor, project effort, CSCI size, schedule, system constraints, and developing contractor capabilities. Since the ESC database was originally constructed for use with SEER-SEM,

the information and ratings provided for each CSCI correlate closely with the terminology of SEER-SEM inputs. Since the development contractor is identified for each CSCI, the ESC database is proprietary in nature (Wells, 1997).

Database Limitations

.¹Several potential limitations of the data have been identified. These limitations were discussed in a May 1997 telephone interview with Ms. Peggy Wells, the ESC database manager, who provided information regarding ESC's endeavors to remove or at least reduce these limitations from the database.

First, many people, who may have interpreted the exact meaning of each data category in significantly different ways, actually collected the data. Therefore, the data represents a certain degree of subjectivity injected by this collection process. This subjectivity has been mitigated to an extent by a normalization process in which one person has actually entered the data in order to enforce a standardized and consistent assignment of ratings and measures for each CSCI (Wells, 1997).

Another limitation was identified in the 1991 Ourada thesis, in which Ourada indicated that some of the information for each project was incomplete (Ourada, 1991: 4.1). This is still true in the current version of the database. Information regarding software application type for each CSCI is present for only a small number of projects, which will severely hinder stratification efforts. ESC is currently updating its database to provide complete information for this data category (Wells, 1997).

A serious concern cited by Ourada was that several of the data points represent projects that were never completed but were estimated for completion. Therefore, these

projects do not represent actual size or effort (Ourada, 1991: 4.1). According to Peggy Wells, all such project data has been removed from the database (Wells, 1997).

Previous AFIT Studies Using ESC Data

There have been two related research efforts at AFIT involving earlier versions of the ESC database. A 1990 study by Daly analyzed the schedule estimation accuracy of five parametric models (PRICE-S, REVIC, SEER-SEM, System-4, and SPQR/20). This study is relevant only in its discussion of the ESD database. Daly stated that the ESD database contained insufficient detail to run the selected models, so he obtained data on 26 ESD projects which had been collected by the MITRE Corporation (Daly, 1990: 50-51). The 1991 Ourada study was reviewed in Chapter 2, and his findings regarding the ESC database were discussed previously in this chapter.

Assumptions

Before inferences regarding the results of this research can be made, three assumptions regarding the data must be stated:

1. Computation or transcription errors in the ESC database, or data entry errors by this researcher, are assumed to be normally distributed, and as such, will have little or no effect on the results.
2. The project data is assumed to be reliable and accurate enough for the results, whether positive or negative, to be considered valid.
3. The categories of project data are assumed to be collected according to standard, consistent definitions throughout the duration of data collection (i.e., the definition of an input remains the same for all observations in the ESC database).

The third assumption is critical to the validity of the data, particularly for the project size parameter. Earlier in this chapter, concerns by ESC regarding the consistency of its data definitions were discussed. As explained in Chapter 2, it is important to have a consistent definition for program size across a given database, and size is a well-known

parameter for inconsistency in definition, attributable to a lack of standardization across programming languages (Coggins and Russell, 1993: 11-12).

A fourth assumption concerns the SPR Programming Languages Table discussed in Chapter 2, where independent research has questioned the accuracy of the SLOC/FP ratios contained therein:

4. It is assumed that inaccuracies within this language table (which is used internally by CHECKPOINT for each project sized in SLOC) will not insert so much error into the calibration procedure as to totally invalidate the research results.

Database Stratification

Since there is sufficient detail at the CSCI level, each of the 312 CSCIs in the ESC database was evaluated as an independent project. Therefore, these CSCI-level data that comprise the various data sets will be referred to as “projects” or “data points” for the remainder of this thesis. Total system-level data for projects will not be used in any calibration or validation procedures.

An early intent of this research effort was to stratify the ESC database in accordance with the software application categories developed by MCR and employed in the Septuagint Study (listed in Chapter 2). Five of the seven application types were thought to be applicable to ESC programs, although ESC does not develop space-related software. Despite the fact that a “platform/application” data category exists as an input to the ESC database, very few data points contain this information. It seemed plausible that anyone with corporate knowledge regarding the projects in the ESC database would be able to at least identify the appropriate operating environment for the projects; however, repeated attempts by this researcher to obtain this information were unsuccessful.

As a result of this limitation, the database was stratified along two other principal data categories that contain more complete data: programming language and development contractor. Also, the database was examined to determine if there are sufficient data points to create data sets stratified by both parameters simultaneously (e.g., language then contractor, or vice versa). As explained in Chapter 2, a detailed stratification process will yield more homogeneous data sets, hypothetically improving the predictive ability of a model for similar projects after calibration.

Development Contractor

Stratifying the database by contractor is an important element of this research effort, because it will either support or repudiate a general opinion in the software estimating community that a model can more accurately estimate effort for a particular developer once the model has been calibrated to that developer (Kemerer, 1987; Genuchten and Koolen, 1991; Herron and Garmus, 1996). The premise for this opinion follows: If sufficient data points are available, calibrating a model to a specific contractor should incorporate that contractor's relative productivity level into the calibrated equations, yielding more accurate estimates for a new project by the same developer. This provides a valuable opportunity in DOD, since the SMC SWDB used in the Septuagint Study contains no contractor data. Although stratification by application type may yield more homogeneous data sets, gains in accuracy may be offset by the mixing of development contractors with varying productivity levels.

Conversely, a concern particular to this research is that although contractor data are available, the mixing of software platforms/application types may reduce overall accuracy. Problems of this nature are inherent to much DOD cost estimating research, in

that there are seldom enough data to permit adequate data stratification while maintaining an adequate sample size.

Since the ESC database contains proprietary information, all contractor names were encoded by an arbitrary letter designation (e.g., contractor A, contractor J, etc.); the key to the contractor codes is not published with this research. It is important to note that some of the contractor designations actually represent multiple contractors who comprised software development teams for some projects.

Programming Language

Projects were stratified by any programming language with sufficient records to provide an adequate sample size. Since the Mertes study calibrated CHECKPOINT on COBOL programs from SMC, this should enable a direct comparison of CHECKPOINT's estimating accuracy for COBOL programs in the ESC versus the SMC development environments.

Separation into Calibration and Validation Subsets

To remain consistent with the latter two efforts in the Septuagint Study, each identified category must contain at least eight data points to be included in the calibration/validation procedure. For project categories that meet this initial criterion, the data set was divided by two, with one-half comprising the calibration subset, and the other half comprising the validation subset. In the instance that a data set has an odd number of projects, the extra data point was added to the calibration subset (Mertes, 1996: 29). Each data set was randomly divided into calibration and validation subsets using the random number generator in *Microsoft Excel*, Version 7.0.

Resampling Technique. As discussed in Chapter 2, small data sets of completed projects are a common limitation when calibrating parametric estimating models to a specific user environment. When working with an extremely small population, it can be difficult to obtain a random, statistically representative sample. This hinders the ability to draw meaningful inferences about the population from a sample (Newbold, 1995: 227-228). Placing model calibration in this statistical context, a data set represents the population for a particular project category, and its (randomly determined) calibration subset can be considered a sample from that population.

By repeatedly drawing random samples from that population, different calibration subsets can be obtained; in turn, each independent subset will possess differing statistics that results in different calibration parameters. Thus, each sample results in a different calibrated model that will yield different levels of estimating accuracy for the validation subset and for new projects. Once the entire calibration/validation procedure has been performed on all randomly generated subsets, calibrated model accuracy can be determined from an average of the results for the validation subset samples.

This technique can provide more meaningful results when working with smaller data sets than otherwise acceptable by allowing each data point a chance to be utilized in different, independent samples from the same population. The independent sample means from these combinations can then be averaged to provide “a mean of the means.” There is also a degree of flexibility with resampling, in that the estimator can vary the number of samples drawn and analyzed, as well as the size of the sample.

The data sets to be resampled were identified according to the following criteria (which are patterned after the overall database stratification criteria outlined previously in this chapter):

1. There must be between eight and 12 data points in the data set.
2. The data set will be separated according to the 50/50 separation ratio described previously in this chapter.
3. Four independent samples (calibration subsets) will be drawn from each data set, and the statistical measures of accuracy will be averaged.

For this research, the decision to resample was made after the ESC database had already been reduced according to the overall stratification methodology described previously (which was consistent with the 1996 efforts in the Septuagint Study). As a result, the resampling criteria were determined by this researcher after considering the existing stratification procedure, the sizes of the data sets already obtained, and the time available to perform the additional procedures and analyses.

Relevant Range Determination

The relevant size range of CHECKPOINT's estimating capability is another important consideration when evaluating project data for inclusion into the calibration/validation procedure. A CER cannot be assumed to maintain the same level of accuracy if the predictor variables lie outside the range of observations upon which it was developed (Neter *et al.*, 1996: 84). Likewise, the underlying algorithms for CHECKPOINT, or any parametric model, are most appropriate within their specific relevant range. Therefore, when the level of a predictor variable (e.g., size) falls far beyond the range of past data, inferences regarding model outputs should be viewed with extreme caution (Neter *et al.*, 1996: 85).

CHECKPOINT's knowledge base was derived from projects ranging in size from 10 function points to 90,000 function points (SPR, 1996: 12-13). A cursory evaluation of the ESC database, using the SLOC/FP ratios provided in the SPR Programming Language Table, showed that many of the smaller CSCIs may exceed the lower bounds of CHECKPOINT's knowledge base (again, this assumes the SPR Language Table is reasonably accurate). Because of this relevant range consideration, all projects smaller than 2K SLOC, regardless of language, were eliminated from the stratification process. From this initial evaluation, it appears that none of the project sizes in the ESC database would exceed 90,000 function points; thus, none of the larger CSCIs were eliminated from consideration.

The range of applications from which CHECKPOINT's productivity factors were derived is 100 to 500 function points (SPR, 1996: 12-11). This is a much more limiting relevant range; for projects outside this range, the CHECKPOINT User's Guide recommends "adjusting productivity upward for the small project and downward for the large project" (SPR, 1996: 12-11). This relevant range was purposely ignored in this research for two reasons: 1) Since the Quick Estimate Mode of CHECKPOINT automatically calculates productivity, there was no means to manually modify the productivity level; and 2) Eliminating projects outside this range, merely because of productivity considerations, would effectively eliminate most of the ESC database from consideration for this study.

CHECKPOINT Estimation Process

Estimating Sequence

Although CHECKPOINT's exact algorithms are proprietary, the User's Guide does describe the basic steps CHECKPOINT goes through in estimating each activity in a software project. This provides insight into why the model's required inputs are necessary for even the Quick Estimate Mode (SPR, 1996: 12-9). This estimating sequence is shown below:

1. CHECKPOINT analyzes the Project Classification information (nature, scope, class, and type). It uses these factors to select the kinds of deliverables (e.g., source code, documentation, and test cases) and activities that are most likely to occur.
2. CHECKPOINT predicts from its knowledge base the sizes of the deliverables that will be produced.
3. CHECKPOINT predicts the initial staffing, effort, schedules, and the costs of producing the project's deliverables.
4. CHECKPOINT tunes the initial results in response to the optional project attributes' data inputs. This data defines the staff, tools, methods, and other factors which influence projects. If none of the project attributes have been input, then the estimating sequence omits this step.
5. Finally, the task-level results are aggregated to produce the higher level activities, phases, and final project total (SPR, 1996: 12-9).

CHECKPOINT Templates

SPR has developed a clever means of enabling users to calibrate CHECKPOINT to specific development environments without allowing the user any insight into its proprietary algorithms: the *estimation template*. The CHECKPOINT User's Guide states that templates "allow the measurement function in a software development organization to create a truly programmable project estimator," and it also stresses the importance of reliable baseline data to the successful use of templates (SPR, 1996: 13-2).

A template is developed by first entering required data for a related group of completed software projects into CHECKPOINT using its measurement mode. Then each data set of projects is saved into a *portfolio*. Finally, a template is created from the portfolio, and the template can be selected to estimate new projects of that type. In this study, the data sets used to create the templates for various categories will consist of the calibration data subsets. (A step-by-step procedure for constructing a CHECKPOINT template will be explained in the following section of this chapter.)

The primary variables affected by an estimation template are:

1. QUALITY
 - a. Defect Potential
 - b. Defect Removal
2. PRODUCTIVITY
 - a. Deliverable Sizes (e.g., source code)
 - b. Assignment Scopes
 - c. Production Rates (e.g., SLOC or FP per person-month) (SPR, 1996: 13-5)

Essentially, this means that CHECKPOINT considers quality and productivity to be important cost drivers which are somehow crucial to its estimating algorithms. Thus, estimation templates replace a portion of CHECKPOINT's knowledge base with quality and productivity values derived from the portfolio of projects (SPR, 1996: 13-2).

Calibration Procedure

When strictly following the step-by-step CHECKPOINT calibration procedure outlined in the Mertes study, this researcher found it extremely difficult to duplicate the template construction process (Mertes, 1996: 33-34). To rectify this problem, several areas in need of clarification were reexamined. It is hoped that these step-by-step instructions, now revised and expanded, will allow complete replication of the actual calibration methodology employed for CHECKPOINT Version 2.3.1. Although there are

many differences, it is important to credit Capt Mertes for the original procedure from which these instructions are developed.

The revised instructions provided in this chapter have been annotated with commentary and details describing how each step was followed in this research effort. A more generalizable, condensed set of instructions, which may be applied to other CHECKPOINT calibration and validation efforts, is provided in Appendix B. The instructions in Appendix B are intended as an *unofficial* supplement to the CHECKPOINT for Windows Version 2.3.1 User's Guide. As such, they do not offer a complete description of how to use the CHECKPOINT model to obtain an estimate, but instead focus on the specific steps necessary to calibrate and validate the model to any specific development environment, as determined by this researcher.

Normalization of Effort

To achieve more accurate estimates, CHECKPOINT calculates effort at the task level and aggregates the task-level effort values to arrive at an effort total for the project. If specified by the estimator, it then normalizes the effort values by phase to arrive at the percentage of total effort for each phase, which is reported in the estimate. These percentages may vary by project, according to inputs such as programming language and program complexity.

CHECKPOINT also allows manual input of effort by task or phase for users with access to this level of information for their development environment. In the Mertes study, effort was normalized according to values recommended by MCR (Mertes, 1996: 31). In the ESC database, however, very little data is available regarding percentages of effort by phase. Therefore, the normalized effort percentages from each project's default

estimate were used to calculate effort by phase, and the phase-level effort figures were entered into each project using CHECKPOINT's measurement mode. Ideally, phase-level effort data for the projects used in template construction would enable the model to more accurately normalize effort for similar projects, using the applicable templates.

Revised Instructions

A few important details regarding these instructions must be understood before the actual procedure is initiated.

- CHECKPOINT operates in a Windows 95 style environment, so all items referred to in this procedure are either drop down menu labels, menu items, or data field labels.
- For ease of reference, menus, menu items, and data field labels are stated exactly as shown in CHECKPOINT; all such items are indicated by Arial font in these instructions.
- Once the information relevant to a particular window has been entered, the window should be closed; this will not close the project file. The steps in these revised instructions have been grouped so this will be more intuitive.
- The annotations that accompany each step in these instructions describe how the instructions were followed in this specific research effort. They are provided for the purpose of clarity, and to enable complete replication of this research. Therefore, the specific inputs or selections provided in these annotations may not reflect the optimum selections for other development environments.

Initial Setup. For the first project only, some preliminary information that is common to all data records must be entered. To begin,

Under the **SETUP** menu:

1. Go to Project Settings and:
 - a. Under Project Mode, identify type of estimate. (In this study, Measure was used.)
 - b. Under Work Metric, identify the time units. (In this study, Months was selected.)
 - c. Under Data Entry Level, select Phase.
2. Go to Time Accounting and:
 - a. Under Average Employees Year, change values to reflect the work environment. (In this study, Non-project days was changed to 0.)
 - b. Go to Project Accounting and:

- 1) Enter values for Accounting hours per business day and Productive hours per project day. (In this study, 8.00 for each category was used.)
 - 2) Enter values for Overtime hours per project day and Overtime hours per non-business day. (In this study, 0.00 for each category was used.)
 - 3) If the remaining fields are left at their default values, the Output (Per Year) will be 251 Business days, 231 Project days, and 1,848 Productive hours.
3. Next, go to Software Metric and select the appropriate sizing metric for both Output software metric and New code software metric. (In this study, Lines of code (KLOC) was used for both items, since no function point data was available.)

Under the INPUT menu, select Required Input and do the following:

4. Go to Project Description and enter appropriate information. Dates are required in the Current Date and Project Start fields, but are not used to constrain the schedule in any fashion. (In this study, the date the project was entered into CHECKPOINT was used for both fields.)
5. Next, go to Project Classification and:
 - a. Under Project nature, identify the project. (In this study, New program development was used for all projects.)
 - b. Under Project Scope, choose from the available entries. (Program(s) within a system was used in this study.)
 - c. Under Project Class, identify the project. (External program, developed under military contract was used in this study.)
 - d. If appropriate, check the box labeled Strict military specification (this was selected for all projects in this study).
 - e. Under Project type, identify the project. (Embedded or real time program was used for both the primary and secondary fields in this study, since it is known that the majority of the projects are embedded. Undoubtedly, MIS projects are included in the ESC database, but it was impossible to identify these projects for this study.)
6. Go to Project Goals and identify accordingly. Find the standard estimate of
5. schedule, staff, and quality was used in this study.
7. Finally, go to Project Complexity and select an appropriate New problem
6. complexity, New code complexity, and New data complexity. (The default values of 3.00 were used for all projects in this study.)
8. Close all windows still open at this point.

These first eight steps comprise the entry of preliminary information that is generally assumed to be common to all projects in the ESC database.

Generation of Default Estimates. Once the initial setup has been completed, specific projects can be entered into CHECKPOINT to obtain nominal estimates of effort. To enter the first project,

Under the INPUT menu:

9. Go to Required Input and:
 - a. Select Function Sizing or Source Code, depending on the software sizing metric to be entered. If KLOC was selected for the software metric (Step 3 above), then Function Sizing will not be available.
 - b. If Function Sizing was selected, enter the number of function points for each of the five Function Types.
 - c. If Source Code was selected, choose the appropriate programming language(s) by clicking on the Choose Languages box that is aligned with the appropriate Code Class. (Languages were chosen under the New code class for this study.)
 - 1) When the Choose Languages box is selected, a New Code Multiple Languages window appears. Under Enter New Languages as, select Lines of Code (KLOC).
 - 2) From the New Language menu, select the appropriate language. (Pages 7-32 to 7-34 of the CHECKPOINT User's Guide provide complete information on how to enter multiple development languages for a single project.)
 - 3) Enter the size in KLOC for the first project in the Measured KLOC field and click OK.
10. Close the Source Code window.
11. Return to Required Input menu and:
 - a. Unless specific information is available, leave all fields under the Project Cost submenu at their default values. (Since project effort in this study is estimated in terms of Person-Months (PM), the Project Cost values are not relevant to the estimates and were ignored.)
 - b. Under Task Selection, click off the box labeled Automatic selection of tasks for development. Ensure the Phase button is selected.
 - 1) Select only the development phases which correspond to the phases of development included in the effort data for the projects. (In this study, five phases were selected: Requirements, External Design, Internal Design, Coding, and Integration and Test.)
 - 2) If information regarding User Involvement in Tasks is known, then click off the Automatic selection of tasks box and select the appropriate phases; otherwise, leave all fields in this section at the default values. (In this study, User Involvement was not included in the actual or estimated total effort, the default values were accepted.)
 - 3) Close the Task Selection window.

Under the VIEW menu:

12. Select Development and Task analysis to obtain the default estimate and percentages of effort by phase (if not using previously obtained normalization percentages) for the first project.
 - a. The total default estimate in PM is listed under the Effort Months column heading and should have an asterisk (*) to indicate the number has been estimated by CHECKPOINT, not input directly. (If the asterisk is not present, then under INPUT, Measurement Data Entry, and Development Effort, click the Estimate All button, then Yes.) Record the default estimate.
 - b. If using the phase-level Development Schedule/Effort Percentages estimated by CHECKPOINT, record these numbers for use in Step 14. (In this study, only the numbers under the Effort % column were used.)

Under INPUT:

13. Select Measurement Data Entry and Development Effort.
14. Enter the appropriate Effort Months values for the selected phases. (These values are calculated by multiplying total actual effort for the project by normalization figures available to the organization or by using the percentages obtained in Step 12-b.) Click the Next button to advance to each phase.
15. Once all effort percentages have been entered, click the Measure All button, then Yes. The estimated effort for the project (previously described in Step 12-a) will now reflect the actual effort for the completed project. (The asterisk will now be absent from the effort total, which shows that the effort months were input into the model rather than estimated.)
16. Close the Development Task Analysis window.
17. Under the FILE menu, select Save as and assign a filename that adequately describes the first project. Under the Version label field, type "default" to indicate this version as the default estimate.

Now that the first project has been entered and saved, all remaining projects can be entered by opening the prior one, inputting values for a new project (e.g., differing size, effort, and language information), and saving it with a new name. Remember to toggle between the Estimate all and the Measure all buttons to obtain the default estimates and then to save the projects. To open a previously saved project:

18. Under the FILE menu, select Open and Project.
19. Highlight the desired project and click OK.
20. Under Open Version, select the version of the project to be opened and click

OK. (In this study, there was no reason to produce multiple versions of a project.)

21. Close the Project Description window and continue with steps 9-17. (Step 11 was not repeated in this study, since all estimates were based on the same five development phases.)

Template Construction. Once all projects have been entered, a portfolio can be created to group homogeneous projects together according to the previously determined stratification categories. For each portfolio, select only the projects that are considered to be part of the calibration subset for that project category.

1. Under FILE, select New and Portfolio.
2. Highlight the records to be included in the portfolio by holding down the CTRL key while selecting projects. After all applicable projects have been selected, accept all default options in this window and click OK.
3. Review the contents of the Versions window to verify that the correct projects have been selected. Under FILE, select Save as, assign a filename and portfolio description that appropriately describes the data set and click OK. (Portfolios will be identifiable in the CHECKPOINT directory by a .prt file suffix.)

The final step in the CHECKPOINT calibration process is to create a template for each portfolio. To accomplish this,

4. Under FILE, select New and Template.
5. In the New Template window, leave all default values as they appear on the screen and click OK. (CHECKPOINT will default to the portfolio filename and description for saving; unless another name and description is desired, accept this name. Templates will be identifiable by a .kb file suffix.)
6. Repeat Steps 1-5 to create a template for each stratification category.

Validation Procedure

Once a template has been created for each stratified data set using its applicable calibration subset, all projects must be estimated again, now using the template for that data set. The step-by-step procedure for using calibration templates to obtain new effort estimates for projects is as follows:

1. Under the FILE menu, select Open and Project.
2. Highlight the desired project and click OK.

3. Close the Project Description window. Under the VIEW menu, select Development and Task Analysis to open the window containing the existing effort estimate for the project. Leave this window open for the remaining steps.
4. Under the INPUT menu, select Measurement Data Entry and Development Effort. Click the Estimate all button, then Yes, and review the Task Analysis window to ensure an asterisk now appears next to the effort estimate.
5. Under the FILE menu, select Use Template, highlight the appropriate template for the project, and click OK. When the Template Options window appears, click OK to accept the default values. (In this study, the default values were used.)
6. The effort estimate in the Task Analysis window will now have a plus (+) sign to indicate that the estimate was calculated using a calibration template. Record this estimate (which will be used either to evaluate the success of calibration, or to validate improvements in CHECKPOINT's estimating accuracy).
7. Repeat Steps 1-6 for all remaining projects.

When this procedure has been completed, there will be a default estimate as well as a calibrated estimate for each project in both the calibration and validation subsets. This allows a direct comparison of default vs. calibrated model accuracy with respect to each subset. As stated previously, the success of the model calibration can be evaluated by determining accuracy improvements in the calibration subsets. However, inferences regarding estimating accuracy for new projects should be based solely upon results from the validation subsets.

Statistical Analysis Measures

To facilitate comparison with the majority of the Septuagint Study, as well as other ongoing 1997 calibration/validation efforts, five statistical measures of accuracy were used. Each measure was used to validate the estimating accuracy of the default and calibrated models to the calibration and validation subsets for each data set. These measures are summarized in Table 3.

Table 3. Accuracy Statistics Summary

Statistic	Description	Accuracy Criteria
MRE	The degree of estimating error in an individual estimate	As MRE approaches 0, accuracy improves
MMRE	The average degree of estimating error in a data set	MMRE \leq 25%
RMS	The model's ability to accurately forecast the individual actual effort	As RMS approaches 0, accuracy improves
RRMS	The model's ability to accurately forecast the average actual effort	RRMS \leq 25%
PRED (0.25)	The percentage of estimates that are within 25% of the actual results	PRED (0.25) \geq 75%

(Conte, Dunsmore and Shen, 1986: 276; Southwell, 1996: 33)

Magnitude of Relative Error (MRE)

“If a model cannot perfectly estimate a set of projects, then two types of errors could result: 1) overestimates, where the estimated effort exceeds the actual effort; and 2) underestimates, where the actual effort exceeds the estimated effort” (Kemerer, 1987: 420). Since both errors can seriously impact software projects, Magnitude of Relative Error (MRE) is a good statistical test to capture the seriousness of overestimates and underestimates (Conte, Dunsmore and Shen, 1986: 172). The MRE test, which prevents the two types of errors from canceling each other out when an average is taken, is calculated by using the following equation:

$$MRE = \left| \frac{E_{act} - E_{est}}{E_{act}} \right| (2)$$

where E_{act} is the actual effort for each project in the ESC database, and E_{est} is effort estimated by CHECKPOINT. To define MRE in terms of a percentage, simply multiply the above equation by 100%. In this study, MRE was calculated from the estimates for each project in the calibration and validation subsets, before and after calibration.

Mean Magnitude of Relative Error (MMRE)

The MMRE, which indicates the average degree of estimating error in a data set, was calculated for each calibration and validation subset, before and after calibration. MMRE was calculated using the following equation:

$$MMRE = \frac{1}{n} \sum_{i=1}^n MRE_i \quad (3)$$

Thus, as MMRE decreases, the model's ability to accurately predict the data set (on the average) increases. An MMRE of 25% or lower is considered to be acceptable for effort prediction models (Conte, Dunsmore and Shen, 1986: 172).

Root Mean Square Error (RMS)

RMS, which measures the model's ability to forecast the actual effort for each individual project, is calculated using the following equation:

$$RMS = \sqrt{\frac{1}{n} \sum_{i=1}^n (E_{act} - E_{est})^2} \quad (4)$$

Thus, as RMS decreases, the model's ability to forecast actual effort for each specific project increases. Conte *et al.* state that RMS is only meaningful for regression-based models (Conte, Dunsmore and Shen, 1986: 174). Since it is unclear exactly how CHECKPOINT's algorithms were developed, it is assumed regression was involved; therefore, RMS was measured and reported as a valid statistic.

Relative Root Mean Square Error (RRMS)

RRMS, which measures the model's ability to forecast the average actual effort, is calculated using the following equation:

$$RRMS = \frac{RMS}{\frac{1}{n} \sum_{n=1}^n E_{act}} \quad (5)$$

Thus, as RRMS decreases, the model's ability to forecast average actual effort increases. An RRMS of 25% or lower is considered to be acceptable model performance (Conte, Dunsmore and Shen, 1986: 175).

Prediction at Level k/n (PRED (l))

This measure is called the Percentage Method, and it is often used as a criterion in determining the acceptability of a model's predictive ability (Conte, Dunsmore and Shen, 1986: 173). Conte *et al.* state that an acceptable criterion for an effort estimation model is PRED (0.25) for at least 75% of the data set. They are also careful to point out that an acceptable PRED (l) does not necessarily preclude the possibility of an extremely poor prediction for a single, new project (Conte, Dunsmore and Shen, 1986: 173). PRED (l) is calculated using the following equation:

$$PRED(l) = \frac{k}{n} \quad (6)$$

where k is the number of projects with MRE of less than l , in a data set of n projects.

Conte's Criteria

According to Conte *et al.*, a model can be considered to be acceptably accurate when it satisfies the three following criteria (Conte, Dunsmore and Shen, 1986: 276):

$MMRE \leq 25\%$	(7)
$RRMS \leq 25\%$	(8)
$PRED(0.25) \geq 75\%$	(9)

In the book *Software Engineering Metrics and Models*, Conte *et al.* discussed a few concerns regarding the above criteria. Since some models perform better in some

environments, these criteria will not always be in agreement for a set of models and projects (Conte, Dunsmore and Shen, 1986: 276). This may be especially true for heterogeneous data sets (this is certainly applicable to this study, due to the inability to stratify by application type). Since RRMS is much more conservative, it will reject far more projects than MMRE for a diverse data set, making it difficult to satisfy equations (7), (8), and (9) simultaneously. Thus, Conte *et al.* recommended a more reasonable set of criteria:

$$\boxed{MMRE \leq 0.25 \text{ and } PRED(0.25) \geq 0.75} \quad (10)$$

Due to the diversity in nature of the projects examined in this study and the inability to completely stratify similar projects, the criteria in Equation 10 were used to validate the accuracy of CHECKPOINT on each data set.

Summary

This chapter provided a complete methodology for calibrating and validating the CHECKPOINT model to the ESC database. Beyond this immediate intent, it is hoped that this methodology can be utilized to broaden understanding of the calibration and validation process in general, and to facilitate the calibration of CHECKPOINT to other software development environments.

First, the ESC database was explored to evaluate possible means of stratification into data sets; programming language and development contractor are the main categories upon which the database was stratified. Additional discussion regarding the database covered the separation of each data set into calibration and validation subsets. This chapter described the CHECKPOINT calibration process, focusing on how it utilizes templates to produce more accurate estimates. Next, a generalizable, step-by-step

procedure for calibrating CHECKPOINT was explained, and additional details were given on how these instructions were followed for this research. Finally, this chapter outlined the validation process, and it defined five statistical measures used to validate model accuracy. The results of all procedures described in this chapter will be provided in Chapter 4.

Notes

¹The noted sections describing the ESC database represent a collaborative information-gathering and writing effort between Capt David Marzo and the author. Capt Marzo is currently calibrating the SAGE software cost model to the ESC database as part of another 1997 AFIT thesis. Thus, this information will also appear in the 1997 Marzo thesis (heretofore unpublished), virtually as shown.

Chapter 4

Findings and Analysis

Overview

This chapter presents the findings and associated analyses generated from the procedures described in Chapter 3, *Methodology*. The structure of the discussion parallels the main sections of Chapter 3. The first section, entitled “Data Reduction,” outlines the discoveries and decisions made after an in-depth examination of the data. The next section, entitled “Database Stratification Results,” discusses the process of further evaluating the ESC database for homogenous project categories and the resultant stratification into usable data sets. The third section, entitled “Calibration Results,” discusses the outcome of the CHECKPOINT calibration effort with regard to statistical and qualitative analysis of calibration success. The fourth section, entitled “Validation Results,” presents the project estimation results and statistical analysis of model estimating accuracy before and after calibration. All of these sections are aimed toward fulfilling the research objectives outlined in Chapter 1, *Introduction*.

Data Reduction

After a thorough examination of the ESC database, it was discovered that many of the projects lacked CSCI-specific effort totals. Rather, effort for many records was listed

under an aggregate system- or project-level effort total, and some projects had no recorded effort whatsoever. This finding eliminated many projects from inclusion in this study, since actual effort was a critical element for the model calibration.

Of the 312 CSCI-level records in the ESC database, 102 records contained the minimum information necessary for a viable calibration effort (e.g., size, effort, and programming language, as described Chapter 2) and simultaneously met the stratification criteria. In Chapter 3, several assumptions regarding the database were stated; rather than make additional assumptions about the missing data, the calibration procedure was based solely on these 102 records. Although the original database contained approximately 60 input parameters, the database was further reduced to include only those parameters pertinent to this particular procedure (i.e., parameters that effectively distinguished between projects and could be input into CHECKPOINT). The reduced database used in the stratification process is shown in Appendix C.

As mentioned previously, it was necessary to encode the contractors by arbitrary letter designations (e.g., contractor A, contractor J, etc.). Since several ESC projects were developed by contractor teams, some of the contractor designations may actually represent multiple contractors.

Database Stratification Results

Project records were stratified along two principal project categories with sufficient information to provide an adequate number of data points for this study: programming language and development contractor. When enough data points were available, the data were also stratified by language and contractor simultaneously. As mentioned previously, the lack of information regarding software application type causes an

unavoidable inconsistency with the stratification methodology employed throughout the Septuagint Study. (Although ESC is currently updating the database to include project-specific environmental platform/application type, this information was unavailable in time for the stratification process.)

The records in Appendix C were stratified according to the criteria described in Chapter 3, and this resulted in the creation of ten data sets. The reduced database was stratified into two data sets by contractor, four data sets by programming language, and four data sets by both contractor and language. These data sets are summarized in Table 4, which lists the stratification category, the number of data points in each data set, the number of points used for the calibration and validation subsets, and the number of different samples drawn for the subsets.

Table 4. Database Stratification Results

Stratification Category	#of Projects	# used for Calibration	# used for Validation	# of Samples
Contractor				
Contractor B	9	5*	4*	4
Contractor J	22	11	11	1
Programming Language				
Ada	17	9	8	1
Assembly	23	12	11	1
Fortran	24	12	12	1
Jovial	14	7	7	1
Contractor and Language				
Ada & Contractor R	10	5*	5*	4
CMS2/Assembly (90/10) & Contractor M	10	5*	5*	4
Fortran & Contractor A	15	8	7	1
Jovial & Contractor J	13	7	6	1

Of the ten data sets created, three met the resampling criteria described in Chapter 3.

From these three data sets, the subsets marked by an asterisk (*) denote the size of the

different subsets obtained using the resampling technique. Four subset samples were drawn for each of the three data sets that qualified for resampling. (As detailed in Chapter 3, resampling was conducted on the smaller data sets to increase the statistical strength of the accuracy measures.)

Stratification by Language

Many of the records in Appendix C (as well as other records that did not qualify for the final reduced database) indicated that multiple programming languages were used in development. For the purposes of this study, it is assumed that a language mix of greater than 95% to 5% (given two languages) is indistinguishable from sole use of the predominant language (in terms of effort required). Thus, projects with greater than a 95% to 5% language ratio were included in data sets of the predominant language. This assumption allowed additional projects to be included in the language-stratified data sets, in turn providing larger, more statistically robust data sets.

From an integration standpoint, the preceding assumption may seem faulty, since a project involving multiple development languages of different levels (i.e., a high-order language and a machine language) could conceivably incur integration costs beyond the norm. Since CHECKPOINT allows multiple development languages to be input into a project estimate, it was possible to test this assumption. For example, Project #29 from Appendix C was developed using 99% Jovial and 1% Assembly (% of total KLOC); this project was estimated according to this ratio and also as a 100% Jovial project. The resultant CHECKPOINT estimates from this experiment were virtually identical, supporting the assumption that multiple languages may be insignificant if the ratio is extremely high, at least with respect to CHECKPOINT's estimating algorithms.

Calibration Results

According to the step-by-step CHECKPOINT calibration procedure described in Chapter 3, default effort estimates with phase-level effort percentages were generated and recorded prior to entering the actual effort for each project into the model. After the default estimate for each project was calculated (reported in Appendix E by category), the total actual effort for each project was multiplied by the normalized effort percentages yielded with the default estimate. These proportions were then input back into each project file, to be saved for template construction. (To enable complete replication of this procedure, the normalization figures estimated by CHECKPOINT and the corresponding proportions of actual effort are shown in their entirety in Appendix D.)

Once all projects had been set equal to their actual effort and saved as measurement points, a template was constructed from each calibration subset sample indicated in Table 4. Thus, four different calibration templates were constructed for each of the stratification categories that were analyzed using the resampling technique, resulting in a total of 19 templates for the ten data sets in the study.

Essentially, successfully calibrating CHECKPOINT entails indirectly manipulating parameters in the proprietary equations to provide a best fit solution to the calibration data subsets; each best fit solution is then stored in the template applicable to that data set. The complete results of the calibration process for all data sets, including multiple iterations for those data sets that were resampled, are shown in Appendix E. Each calibration subset is represented by a separate table, consisting of rows showing the record numbers used, and columns showing the default effort, calibrated effort, default

MRE, and calibrated MRE for each record. In addition, default and calibrated MMRE, RMS, RRMS, and PRED (0.25) are shown for each calibration subset.

Table 5 provides information regarding the range of input data, as well as MRE (by record) for the default and calibrated model. These results are taken from the 19 calibration subsets under the ten project categories in Appendix E. Although the ESC database was initially reduced to include only those projects within the relevant range of CHECKPOINT's knowledge base (Chapter 3), Table 5 is useful for other reasons. First, it provides valuable insight into the variation of input and initial output data for each data record. Also, the results from the resampled data sets (intentionally shown for each iteration), specifically the significant range changes across samples, demonstrate the value of resampling and averaging the results.

Table 5. Calibration Data Range Summary

Stratification Category	Size Range (KLOC)	Mean Size (KLOC)	Actual Effort Range (PM)	Mean Actual Effort (PM)	MRE Range (Default model)	MRE Range (Calibrated model)
Contractor B*	8.0 - 104.6	35.8	101.9 - 646.6	259.6	0.51 - 0.91	0.03 - 0.73
	15.9 - 250.0	85.9	90.1 - 2,301.3	672.2	0.03 - 0.85	0.28 - 1.80
	22.6 - 250.0	111.4	101.9 - 2,301.3	894.8	0.51 - 0.70	0.13 - 0.26
	22.6 - 152.2	52.3	90.1 - 1,218.2	345.5	0.03 - 0.70	0.17 - 1.67
Contractor J	5.4 - 79.3	36.3	28.0 - 1,456.9	413.4	0.50 - 0.89	0.11 - 1.53
Ada	18.1 - 184.0	47.0	124.8 - 1,000.0	319.5	0.03 - 0.78	0.00 - 0.94
Assembly	3.0 - 40.7	23.4	14.7 - 1,010.5	347.0	0.55 - 0.99	0.09 - 4.68
Fortran	2.3 - 72.3	22.3	28.3 - 1,456.9	298.9	0.12 - 0.92	0.04 - 3.74
Jovial	2.7 - 181.4	36.3	18.0 - 1,791.0	369.2	0.24 - 0.87	0.05 - 5.45
Ada & Contractor R*	3.3 - 53.0	27.4	81.0 - 406.0	230.1	0.38 - 0.91	0.21 - 0.76
	3.3 - 184.0	58.4	81.0 - 1,000.0	410.4	0.03 - 0.91	0.12 - 0.94
	18.1 - 184.0	57.8	178.0 - 1,000.0	376.12	0.03 - 0.78	0.17 - 0.75
	18.1 - 184.0	57.2	216.6 - 1,000.0	377.5	0.03 - 0.78	0.13 - 0.82
CMS2/ Assembly & Contractor M*	16.4 - 52.4	32.2	295.9 - 1,063.5	694.3	0.80 - 0.94	0.10 - 0.99
	3.6 - 52.4	28.5	67.6 - 1,063.5	587.0	0.80 - 0.94	0.12 - 0.94
	2.9 - 31.6	15.3	67.6 - 1,063.5	449.4	0.92 - 0.97	0.00 - 0.48
	3.6 - 52.4	31.4	67.6 - 927.5	681.8	0.84 - 0.93	0.08 - 0.63
Fortran & Contractor A	2.3 - 9.8	6.5	28.3 - 89.0	64.1	0.75 - 0.90	0.02 - 0.43
Jovial & Contractor J	5.4 - 73.1	26.3	18.0 - 430.3	187.1	0.24 - 0.86	0.07 - 3.19

Although MRE range is not a formal criterion for evaluating calibration success or estimating accuracy, changes in MRE range may provide insights into these areas, particularly with regard to the existence of questionable data points. As will be discussed later in this chapter, *large* increases in MRE range (i.e., an MRE range greater than 1) may indicate instances where the default model has grossly underestimated the majority of projects in a data set, but may have accurately estimated one project. In an effort to provide a best fit solution for the majority of projects in the data set, the calibrated model will then greatly overestimate the project that had initially been accurately estimated.

Analysis of Results

Seven of the ten data sets in this study exhibited an improved MMRE after calibration, although only the Fortran/Contractor A data set met the 0.25 accuracy criterion (shown in Equation 7). RRMS improved for seven of ten data sets, but none of the improvements met the RRMS accuracy criterion of 0.25 (Equation 8). PRED (0.25) improved for nine of ten data sets, but again, only the Fortran/Contractor A data set achieved the desired $\geq 75\%$ accuracy criterion (Equation 9). As explained in Chapter 3, calibration success for each data set was evaluated using just the MMRE and PRED (0.25) criteria (Equation 10), due to the heterogeneous nature of the data. Only one of ten data sets simultaneously met these criteria, indicating a general failure to successfully calibrate CHECKPOINT to the ESC database. As shown in previous research (discussed in Chapter 2), a parametric model will typically achieve better levels of accuracy on the data to which it was calibrated than when using new data. Therefore, if CHECKPOINT could not achieve the specified accuracy criteria for even the calibration data from which the templates were constructed, it is even less likely it will be achieved for the validation subsets.

Table 6 summarizes the calibration effort, providing MMRE, RRMS, and PRED (0.25) results for both the default model and the calibrated model (using the calibration template applicable to each data set). Specifically, these figures are obtained from each calibration subset in Appendix E. An asterisk (*) indicates the three data sets that qualified for resampling; thus, the reported figures for these data sets reflect an average of the four procedural iterations for each data set. Figures in bold type denote calibration

success according to the criteria shown in Equation 10. Following Table 6, a brief supporting analysis in narrative format is provided for each data set.

Table 6. Calibration Results Summary

Stratification Category	Default MMRE	Calibrated MMRE	Default RRMS	Calibrated RRMS	Default PRED (0.25)	Calibrated PRED (0.25)
Contractor B*	0.56	0.47	0.83	0.44	0.10	0.40
Contractor J	0.74	0.51	1.24	0.52	0.00	0.27
Ada	0.54	0.35	0.43	1.01	0.11	0.44
Assembly	0.80	1.98	1.37	1.16	0.00	0.08
Fortran	0.73	0.63	1.67	1.12	0.08	0.42
Jovial	0.72	1.12	1.22	1.57	0.14	0.14
Ada & Contractor R*	0.57	0.50	0.44	0.84	0.15	0.25
CMS2/Assembly & Contractor M*	0.90	0.32	1.06	0.29	0.00	0.50
Fortran & Contractor A	0.82	0.21	0.87	0.28	0.00	0.75
Jovial & Contractor J	0.67	0.79	0.81	0.63	0.14	0.29

Contractor B. This data subset consisted of five records for each of the four resampling iterations; in each sample, a different combination of five records was used. The complete results of all iterations are provided in Appendix E. Projects varied widely in size (8.0 to 250.0 KLOC) and effort (90.1 to 2,301.3 PM). Default MRE ranged from 0.03 to 0.91; calibrated MRE ranged from 0.03 to 1.80. MMRE improved in two of the four samples and met the MMRE criterion in one of four samples. Average default MMRE was 0.56, and average calibrated MMRE was 0.47, both unacceptable according to Equation 7. RRMS improved in three of four samples and met the RRMS criterion in one of four samples. Average RRMS improved from 0.83 to 0.44, failing to meet the RRMS criterion. PRED (0.25) improved in three of four samples and met the 0.75 accuracy level in one sample. Average PRED (0.25) improved from 0.10 to 0.40, falling short of the required 0.75 accuracy.

Thus, this calibration subset did not meet the accuracy criteria in Equation 10. Despite significant improvements in the accuracy measures, this fact suggests the data set may be too heterogeneous to enable successful calibration. In two of the four iterations, the default estimates for Record #7 were very accurate (MRE of 0.03); however, the default model greatly underestimated the remainder of the calibration records. The calibrated model, in an effort to fit its algorithms to the data, overestimated Record #7 (MRE of 1.80 and 1.67), resulting in an increased MMRE for those iterations. Upon investigating the complete record, it was discovered Record #7 represents the only project developed in Pascal language by Contractor B. Although results such as this may warrant removing an anomalous data point such as Record #7, one of the objectives of this research was to determine if stratification by contractor alone would yield sufficiently homogeneous data records for successful calibration. Also ESC representatives could provide no project-specific information (beyond that present in the database), that warranted removal of apparent data outliers. Thus, all records behaving in this fashion were retained (no projects were removed from any data sets).

Contractor J. This calibration subset consisted of 11 projects with a wide variation of size (5.4 to 79.3 KLOC) and effort (28.0 to 1,456.9 PM). A variety of languages was represented in the calibration subset, which further weakened the prospect of overall record homogeneity. Default MRE ranged from 0.50 to 0.89; calibrated MRE ranged from 0.11 to 1.53. MMRE improved from 0.74 in the default model to 0.51 after calibration, failing to meet the MMRE criterion. RRMS improved from 1.24 to 0.52, a significant improvement, but insufficient to meet the RRMS criterion. PRED (0.25) improved from 0.00 to 0.27, falling far short of the required 0.75 accuracy.

The summary results indicate an unsuccessful calibration effort to this data subset. Again, one project (Record #27) had both the best default MRE (0.50) and the worst MRE (1.53) after calibration, although the degree to which this record increases the calibrated MMRE is less extreme than in other cases.

Ada. This calibration subset consisted of nine Ada projects varying significantly in size (18.1 to 184.0 KLOC), effort (124.8 to 1,000.0 PM), and contractor (three contractors represented). Default MRE ranged from 0.03 to 0.78; calibrated MRE ranged from 0.00 to 0.94. MMRE improved from 0.54 to 0.35, falling slightly short of the MMRE criterion. RRMS significantly worsened from 0.43 to 1.01. PRED (0.25) improved significantly (0.11 to 0.44), but failed the 0.75 accuracy measure.

These results, again indicating an unsuccessful calibration (Equation 10), merit further discussion regarding the simultaneous improvements in MMRE and worsening of RRMS. How can this occur? Simply, MMRE does not weight projects by actual effort, but RRMS is a weighted accuracy measure. In this case, the largest project with respect to effort, Record #49 (1,000 PM), is over twice as large as the next largest record (406 PM). The MMRE measure treats Record #49 as only one of nine projects, each contributing an equal percentage (11.1%) to MMRE. In the case of RRMS, the estimating error (i.e., actual effort minus estimated effort) for each record is squared and then summed, magnifying the contribution of Record #49 to RMS, and in turn, RRMS. Thus, large projects with significant estimating errors are much more influential to RRMS than MMRE.

Assembly. This calibration subset consisted of 12 projects varying greatly in size (3.0 to 40.7 KLOC), effort (14.7 to 1,010.5 PM), and contractor (six contractors

represented). Default MRE ranged from 0.55 to 0.99; calibrated MRE ranged from 0.09 to 4.68. MMRE greatly worsened from 0.80 to 1.98. RRMS slightly improved from 1.37 to 1.16, far short of the RRMS criterion. PRED (0.25) showed no significant improvement (0.00 to 0.08).

These results indicate a tremendously unsuccessful calibration effort for this stratification category. MRE significantly worsened for eight of the 12 records, but three of the four remaining records (#83, #84, and #88) represented the three largest projects with respect to effort. Despite the general calibration failure, these results provide evidence that the algorithms contained in CHECKPOINT templates are weighted toward larger projects, which supports the assumption that CHECKPOINT's proprietary algorithms may be based (at least in part) on ordinary least squares regression models.

Fortran. This calibration subset consisted of 12 projects with a wide variation of size (2.3 to 72.3 KLOC), effort (28.3 to 1,456.9 PM), and contractor (four represented). Default MRE ranged from 0.12 to 0.92; calibrated MRE ranged from 0.04 to 3.74. MMRE improved slightly from 0.73 to 0.63, failing to meet the MMRE criterion. RRMS improved from 1.67 to 1.12, still far too inaccurate to meet the RRMS criterion. PRED (0.25) improved from 0.08 to 0.42, falling short of the required 0.75 accuracy.

Based on these results, the model failed to successfully calibrate on this data subset. This data category provides another example of an initially accurate estimate (Record #71, with an MRE of 0.12) that became wildly inaccurate in the calibrated model (3.74). In an effort to explore the sensitivity of MMRE to the inclusion of Record #71, a supplementary Fortran template excluding Record #71 was constructed. Although the complete subset results are not published with this research, the modified template

provided a calibrated MMRE of 0.35 for the 11-record calibration subset, still failing to meet the desired accuracy.

Jovial. This calibration subset consisted of seven projects with a wide variation in size (2.7 to 181.4 KLOC) and effort (18.0 to 1,791.0 PM). Default MRE ranged from 0.24 to 0.87; calibrated MRE ranged from 0.05 to 5.45. MMRE worsened from a default value of 0.72 to 1.12 after calibration, and RRMS worsened from 1.22 to 1.57, both significant failures to meet the respective criteria. PRED (0.25) remained constant at 0.14, falling far short of the required 0.75 accuracy.

These results indicate an unsuccessful calibration effort to this data subset. Record #21 strongly influenced the MMRE result with an MRE that worsened from 0.24 to 5.45. Record #14, with actual effort larger than the other six records combined, also decreased in accuracy (0.63 to 0.85), contributing greatly to the increased RRMS in the calibrated model.

Ada and Contractor R. This calibration subset, stratified by two categories simultaneously, consisted of five records for each of the four samples. The projects ranged in size and effort from 3.3 to 184.0 KLOC and 81.0 to 1,000.0 PM respectively. Default MRE ranged from 0.03 to 0.91; calibrated MRE ranged from 0.12 to 0.90. MMRE improved in all four samples; however, MMRE did not meet 0.25 criterion in any of the four samples. Average default MMRE (i.e., the average of the four sample MMREs) was 0.57, and average calibrated MMRE was 0.50, both unacceptable according to Equation 7. RRMS improved in only one of four samples and did not meet the RRMS criterion in that sample; average RRMS worsened from 0.44 to 0.84, primarily due to the results of Record #49 (with the largest actual effort in the data set). Although

PRED (0.25) improved in two of four samples, it never met the 0.75 accuracy level. Average PRED (0.25) improved from 0.15 to 0.25, falling far short of the required 0.75 accuracy.

Thus, the accuracy statistics again failed to meet the criteria for calibration success, as specified in Equation 10. Record #49, which appeared in three of the four calibration subset samples, was particularly influential in driving the negative result, with a default MRE of 0.03 and calibrated MRE ranging from 0.75 to 0.90. Typically, the statistics for this project would prompt the estimator to investigate this record for data entry errors or unusual development circumstances that would warrant its removal from the data set. It is also possible in this effort that Record #49 represents a different application category than the other records in the data subset (e.g., an MIS CSCI in an overall military ground platform).

CMS2/Assembly (90%/10%) and Contractor M. This data subset consisted of five records for each of the four samples. The projects ranged in size and effort from 2.9 to 52.4 KLOC and 67.6 to 1,063.5 PM respectively. Default MRE ranged from 0.80 to 0.97; calibrated MRE ranged from 0.00 to 0.99. MMRE significantly improved in all four samples, meeting the MMRE criterion in one of four samples. Average default MMRE was 0.90, and average calibrated MMRE was 0.32, falling slightly short of the 0.25 accuracy criterion. RRMS improved in all four samples and met the RRMS criterion in one of four samples. Average RRMS improved from 1.06 to 0.29, narrowly failing the RRMS criterion. PRED (0.25) improved in all four samples but failed the 0.75 accuracy level in all samples. Average PRED (0.25) improved from 0.00 to 0.50, falling short of the required 0.75 accuracy.

Despite significant improvements in all accuracy statistics, model calibration (to these subset samples) was unsuccessful according to Equation 10. To reinforce the meaning of Equation 10, the results according to this equation can be stated in a narrative manner as follows: The calibrated model (i.e., template) fit this calibration subset within 32% (MMRE), and predicted project effort within 25% of their actuals 50% of the time, which fails to satisfy the required MMRE of 25% or lower, and the requirement to predict effort within 25% of actual effort for at least 75% of all records in the subset.

Fortran and Contractor A. This calibration subset consisted of eight projects varying in size and effort from 2.3 to 9.8 KLOC and 28.3 to 89.0 PM respectively. Default MRE ranged from 0.75 to 0.90; calibrated MRE ranged from 0.02 to 0.43. After calibration, MRE improved for every record in this subset (which was unique to all calibration subsets that were not resampled). MMRE improved from 0.82 to 0.21, meeting the MMRE criterion. RRMS improved from 0.87 to 0.28, falling slightly short of the RRMS criterion. PRED (0.25) improved from 0.00 to 0.75, meeting the required 0.75 accuracy.

This was the only successful calibration effort of the ten data sets calibrated on CHECKPOINT in this research, providing evidence that stratification by at least programming language and contractor is necessary to produce a sufficiently homogeneous data set (in the absence of information on application category).

Jovial and Contractor J. This calibration subset consisted of seven projects varying widely in size (5.4 to 73.1 KLOC) and effort (18.0 to 430.3 PM). Default MRE ranged from 0.24 to 0.86; calibrated MRE ranged from 0.07 to 3.19. MMRE worsened from 0.67 to 0.79, failing to meet the MMRE criterion. RRMS improved from 0.81 to

0.63, insufficient to meet the RRMS criterion. PRED (0.25) improved from 0.14 to 0.29, falling far short of the required 0.75 accuracy.

These results, although they indicate an unsuccessful calibration effort to this data subset, can provide additional insight into the statistical accuracy improvements gained by additional stratification categories. This is the only data set stratified by two categories that were also used separately, which enables a direct comparison of the accuracy statistics for each calibration subset (Table 7). It is interesting to observe that the default statistics marginally improved for the Jovial/Contractor J data set, but the calibrated model statistics for Jovial/Contractor J were adversely impacted by the poor results yielded by the Jovial data set. Calibrated PRED (0.25) was the only statistic for Jovial/Contractor J that ranked highest in the comparison, and this is likely insignificant due the small data sets considered. A logical inference from this comparison is that additional stratification categories do not guarantee increased chances of calibration success, if one of the categories does not, in itself, somehow contribute to producing a homogeneous data set.

Table 7. Comparison of Stratification Level on Calibration Results

Stratification Category	Default MMRE	Calibrated MMRE	Default RRMS	Calibrated RRMS	Default PRED (0.25)	Calibrated PRED (0.25)
Contractor J	0.74	0.51	1.24	0.52	0.00	0.27
Jovial	0.72	1.12	1.22	1.57	0.14	0.14
Jovial & Contractor J	0.67	0.79	0.81	0.63	0.14	0.29

Validation Results

As described previously in this research, an evaluation of default vs. calibrated estimating accuracy using new projects (i.e., projects excluded from the calibration process) is necessary to make valid inferences regarding a model’s estimating accuracy. In this study, the validation projects were comprised of the records that remained in each data set after random selection of the calibration subsets. The complete results of the validation procedure, organized by validation subset under each stratification category, are shown in Appendix E. The results are presented in a parallel format to the calibration results (including all iterations of data sets that were resampled).

In addition, Table 8 provides information regarding size, actual effort, and MRE ranges (for the default and calibrated model) for each validation subset. As expected, the MRE ranges are significantly wider for most of the validation subsets than for their calibration subset counterparts. For the Assembly validation subset, the high calibrated MRE value of 14.73 shows the calibrated model overestimating a project by 1473%, indicating a project that may have been estimated quite accurately in the default model, but then fell victim to a template providing the best fit solution for the majority of the projects in the calibration subset.

Table 8. Validation Data Range Summary

Stratification Category	Size Range (KLOC)	Mean Size (KLOC)	Actual Effort Range (PM)	Mean Actual Effort (PM)	MRE Range (Default model)	MRE Range (Calibrated model)
Contractor B*	24.0 - 250.0 8.0 - 152.2 8.0 - 35.0 8.0 - 250.0	115.3 52.6 20.8 94.7	90.1 - 2,301.3 101.9 - 1,218.2 90.1 - 211.3 8.0 - 250.0	930.3 414.6 136.3 822.9	0.03 - 0.57 0.52 - 0.91 0.03 - 0.91 0.51 - 0.91	0.49 - 2.28 0.17 - 0.77 0.16 - 1.51 0.12 - 0.78
Contractor J	2.7 - 73.1	24.9	18.0 - 441.0	186.5	0.12 - 0.97	0.02 - 5.45
Ada	3.3 - 148.3	39.3	18.0 - 406.0	189.2	0.52 - 5.12	0.07 - 11.34
Assembly	6.5 - 38.0	17.7	12.5 - 1,413.7	334.1	0.20 - 0.99	0.02 - 14.73
Fortran	2.6 - 116.1	19.2	26.5 - 565.2	166.0	0.03 - 0.95	0.05 - 2.76
Jovial	5.4 - 79.3	34.4	28.0 - 1,241.9	336.3	0.48 - 0.84	0.09 - 1.41
Ada & Contractor R*	21.3 - 184.0 21.3 - 40.7 3.3 - 53.0 3.3 - 53.0	60.7 29.7 30.3 30.9	178.0 - 1,000.0 178.0 - 256.0 81.0 - 406.0 81.0 - 406.0	398.6 218.3 252.6 251.2	0.03 - 0.73 0.38 - 0.73 0.38 - 0.91 0.52 - 0.91	0.13 - 1.67 0.10 - 0.48 0.09 - 0.84 0.13 - 0.84
CMS2/ Assembly & Contractor M*	2.6 - 31.6 2.6 - 31.6 2.6 - 52.4 2.6 - 28.2	13.7 17.4 30.5 14.5	67.6 - 927.5 132.1 - 927.5 147.5 - 1,063.5 132.1 - 604.4	326.5 433.9 571.5 339.0	0.84 - 0.97 0.97 - 0.84 0.97 - 0.80 0.97 - 0.80 0.97	0.16 - 0.72 0.27 - 0.73 0.43 - 2.24 0.24 - 1.02
Fortran & Contractor A	2.6 - 27.7	9.3	26.5 - 162.0	87.7	0.62 - 0.94	0.09 - 1.17
Jovial & Contractor J	2.7 - 79.3	21.6	28.0 - 1,241.9	306.4	0.65 - 0.87	0.11 - 0.56

Analysis of Results

Six of the ten validation subsets in this study showed an improvement in MMRE after calibration; however, none met the accuracy criterion of 0.25 shown in Equation 7. RRMS improved for six of ten subsets, but none of the improvements met the RRMS accuracy criterion of 0.25 (Equation 8). PRED (0.25) improved for eight of ten subsets; however, none achieved the desired $\text{PRED (0.25)} \geq 75\%$ accuracy criterion (Equation 9). As explained previously, overall acceptable accuracy for each data set was evaluated using only the MMRE and PRED (0.25) criteria (Equation 10), due to the suspected heterogeneous nature of the data. None of the ten project categories simultaneously met these criteria, indicating a general failure by CHECKPOINT to accurately estimate effort for projects in the ESC database (utilizing the level of data entry and stratification procedures described in this study).

Table 9 summarizes the validation results in Appendix E, providing MMRE, RRMS, and PRED (0.25) results for both the default model and the calibrated model. An asterisk (*) denotes the three data sets that qualified for resampling; thus, the reported figures for these data sets reflect an average of the validation subset samples. It is obvious from Table 9 that none of the accuracy measures from the validation records met the criteria for model accuracy. Following Table 9, a brief supporting analysis is provided for each validation subset.

Table 9. Validation Results Summary

Stratification Category	Default MMRE	Calibrated MMRE	Default RRMS	Calibrated RRMS	Default PRED (0.25)	Calibrated PRED (0.25)
Contractor B*	0.60	0.64	0.74	0.49	0.13	0.25
Contractor J	0.69	1.33	0.91	1.43	0.18	0.18
Ada	1.21	1.70	1.34	2.54	0.00	0.50
Assembly	0.83	2.05	1.44	1.20	0.09	0.18
Fortran	0.73	0.70	1.12	2.31	0.17	0.17
Jovial	0.71	0.44	1.22	0.68	0.00	0.43
Ada & Contractor R*	0.59	0.39	0.57	0.72	0.05	0.45
CMS2/Assembly & Contractor M*	0.91	0.69	1.13	0.64	0.00	0.10
Fortran & Contractor A	0.82	0.44	0.84	0.88	0.00	0.29
Jovial & Contractor J	0.80	0.37	1.42	0.70	0.00	0.33

Contractor B. This validation subset consisted of four records for each of the four resampling iterations; the complete results of all iterations are provided in Appendix E. Projects varied widely in size (8.0 to 250.0 KLOC) and effort (90.1 to 2,301.3 PM). MMRE improved in two of the four samples but failed to meet the MMRE criterion in any of the four samples. Average default MMRE was 0.60 and average calibrated MMRE was 0.64. RRMS improved in all four samples and met the RRMS criterion in one of four samples. Average RRMS improved from 0.74 to 0.49. PRED (0.25) improved in two of four samples but did not meet the 0.75 accuracy level in any sample. Average PRED (0.25) improved slightly from 0.13 to 0.25, well below the required 0.75 accuracy.

Thus, this validation subset did not meet the accuracy criteria in Equation 10, which indicates that the calibrated model cannot accurately estimate new projects stratified solely by this category. As discussed in the analysis of the calibration subset, Record #7 had an extremely accurate default estimate (MRE of 0.03) that significantly worsened in

the two samples in which it appeared (calibrated MRE of 2.28 and 1.51 in the two samples). This record strongly influenced the negative MMRE result.

Contractor J. This validation subset consisted of 11 projects varying in size and effort from 2.7 to 73.1 KLOC and 18.0 to 441.0 PM respectively. Default MRE ranged from 0.12 to 0.97; calibrated MRE ranged from 0.02 to 5.45. MMRE worsened from 0.69 to 1.33, and RRMS worsened from 0.91 to 1.43, both grossly failing to meet their applicable criteria. PRED (0.25) remained constant at 0.18, falling far short of the required 0.75 accuracy.

These results indicate a significant failure to accurately estimate effort for this subset. Project #21 appears to be a prime candidate for further investigation and possible removal from this subset, with a default MRE of 0.24 and a calibrated MRE of 5.45, significantly impacting the negative MMRE result. Additionally, Records #26 and #71 (the second and third largest projects respectively) had large MREs (1.55 and 4.48) that seriously impacted the RRMS result.

Ada. This validation subset consisted of eight projects with a wide variation of size (3.3 to 148.3 KLOC) and effort (18.0 to 406.0 PM). Default MRE ranged from 0.52 to 5.12; calibrated MRE ranged from 0.07 to 11.34. MMRE worsened from 1.21 to 1.70, and RRMS significantly worsened from 1.34 to 2.54. PRED (0.25) improved from 0.00 to 0.50, falling short of the required 0.75 accuracy.

Although the MMRE result shows a significant failure to accurately estimate effort for this subset, this figure is driven by the MRE for Record #76, with a calibrated MRE of 11.34 (after a default MRE of 5.12). Again, Record #76 may warrant removal if more information could be gained regarding its development circumstances (e.g., a high

percentage of reused LOC). Supplementary analysis revealed that removing Record #76 from the validation subset would result in an MMRE of 0.32, still slightly above the required 0.25 criterion.

Assembly. This validation subset consisted of 11 projects ranging in size and effort from 6.5 to 38.0 KLOC and 12.5 to 1,413.7 PM respectively. Default MRE ranged from 0.20 to 0.99; calibrated MRE ranged from 0.02 to 14.73. MMRE worsened from 0.83 to 2.05, and RRMS slightly improved from 1.44 to 1.20, both failures to meet their applicable criteria. PRED (0.25) improved from 0.09 to 0.18, falling far short of the required 0.75 accuracy.

These results provide another example of an aberrant data point that seriously impacts the MMRE statistic. The MRE for record #11 worsened from 0.20 to 14.73 after calibration. Also, Record #36 worsened from a default MRE of 0.64 to 3.65. Records such as this pose a dilemma for the estimator—in the absence of additional information, should the outlying data point be removed from the data set?

Fortran. This validation subset consisted of 12 projects with a wide variation of size (2.6 to 116.1 KLOC) and effort (26.5 to 565.2 PM). Default MRE ranged from 0.03 to 0.95; calibrated MRE ranged from 0.05 to 2.76. There was no significant improvement in MMRE (from 0.73 to 0.70), and RRMS significantly worsened from 1.12 to 2.31. PRED (0.25) remained constant at 0.17, far below the required 0.75 accuracy. These results indicate a significant failure to accurately estimate effort for this subset.

Jovial. This validation subset consisted of seven projects with a wide variation of size (5.4 to 79.3 KLOC) and effort (28.0 to 1,241.9 PM). Default MRE ranged from 0.48

to 0.84; calibrated MRE ranged from 0.09 to 1.41. MMRE improved from 0.71 to 0.44, and RRMS improved from 1.22 to 0.68, but both results were insufficient to meet the applicable criteria. PRED (0.25) improved from 0.00 to 0.43, falling short of the required 0.75 accuracy.

Despite significant improvements with this validation subset, these results (along with the other three data sets stratified solely by programming language) indicate that stratification by language may not produce sufficiently homogeneous records to provide accurate estimates after calibration.

Ada and Contractor R. This validation subset consisted of five records for each of the four samples. MMRE improved in three of four samples; however, it did not meet the MMRE criterion in any of the four samples. The projects ranged in size from 3.3 to 184.0 KLOC, and effort from 81.0 to 1,000.0 PM. Default MRE ranged from 0.03 to 0.91; calibrated MRE ranged from 0.09 to 1.67. Average MMRE improved from 0.59 to 0.39 after calibration. RRMS improved in three of four samples but did not meet the RRMS criterion in any of the four samples. Average RRMS worsened from 0.57 to 0.72 after calibration. PRED (0.25) improved in three of four samples but never met the 0.75 accuracy level in any of the samples. Average PRED (0.25) improved from 0.05 to 0.45, a significant improvement, but well below the required 0.75 accuracy.

As discussed in the analysis of the calibration subset, Record #49, with the most accurate default estimate, seriously impacted the summary statistics (particularly RRMS, due to the large size of this project). However, this impact was mitigated somewhat by the fact that Record #49 appeared in only one of the four validation samples.

CMS2/Assembly (90%/10%) and Contractor M. This validation subset consisted of five records for each of the four samples. MMRE improved in three of four samples but still failed to meet the MMRE criterion in any sample. The projects ranged in size and effort from 2.6 to 52.4 KLOC and 67.6 to 1,063.5 PM respectively. Default MRE ranged from 0.80 to 0.97; calibrated MRE ranged from 0.16 to 2.24. Average default MMRE was 0.91, and average calibrated MMRE was 0.69, still unacceptable. RRMS improved in two of four samples but failed to meet the RRMS criterion in any sample. Average RRMS improved from 1.13 to 0.64 but failed to meet the RRMS criterion. PRED (0.25) improved in two of four samples but never met the 0.75 accuracy level. Average PRED (0.25) improved only slightly (0.00 to 0.10).

Again, the calibrated model failed to successfully estimate the validation subset. Unlike many of the other data sets, the results for this subsets are almost uniformly poor; across the four validation samples for this data set, no single data point appears to overwhelmingly influence the overall negative validation results.

Fortran and Contractor A. This validation subset consisted of seven projects varying in size and effort from 2.6 to 27.7 KLOC and 26.5 to 162.0 PM respectively. Default MRE ranged from 0.62 to 0.94; calibrated MRE ranged from 0.09 to 1.17. MMRE significantly improved from 0.82 to 0.44, but RRMS slightly worsened from 0.84 to 0.88, both falling short of the applicable criteria. PRED (0.25) improved from 0.00 to 0.29, well below the required 0.75 accuracy.

Although the calibration results demonstrated successful model calibration to this data set, the calibrated model was not successfully validated. As the only project that did not exhibit a significantly reduced MRE on the calibrated model, Record #66 was a

primary reason for this unsuccessful validation. Record #66 was also the largest project in terms of size and actual effort, which heavily influenced the negative RRMS result. Supplementary analysis revealed that removing Record #66 from the validation data set would result in an MMRE of 0.31 and a PRED (0.25) of 0.33, still insufficient to meet the criteria in Equation 10.

Jovial and Contractor J. This data subset consisted of six projects varying widely in size (2.7 to 79.3 KLOC) and effort (28.0 to 1,241.9 PM). Default MRE ranged from 0.65 to 0.87; calibrated MRE ranged from 0.11 to 0.56. MMRE improved from 0.80 to 0.37, failing to meet the MMRE criterion. RRMS improved from 1.42 to 0.70, far above the RRMS criterion. PRED (0.25) improved from 0.00 to 0.33, far below the required 0.75 accuracy.

Despite improvements across all three accuracy measures, these results still do not meet the overall criteria for successful validation of CHECKPOINT to this data set. As with the calibration subset for this data set, additional analysis is possible for this data set in the form of a direct comparison between the validation results of this data set and the separate Jovial and Contractor J data sets.

Table 10. Comparison of Stratification Level on Validation Results

Stratification Category	Default MMRE	Calibrated MMRE	Default RRMS	Calibrated RRMS	Default PRED (0.25)	Calibrated PRED (0.25)
Contractor J	0.69	1.33	0.91	1.43	0.18	0.18
Jovial	0.71	0.44	1.22	0.68	0.00	0.43
Jovial & Contractor J	0.80	0.37	1.42	0.70	0.00	0.33

Unlike its calibration counterpart, the Jovial/Contractor J validation subset exhibited the largest improvements for MMRE and RRMS, despite initially higher default values. This

result lends support to a generally accepted belief in the software estimating community: Increased database stratification will ideally yield more homogeneous data sets for validation, provided each category contributes positively to the stratification process. However, it must be noted that the accuracy results for all three validation subsets in Table 10 are based on templates that did not indicate successful calibration (Table 6). This severely limits the strength of inferences from the results in Table 10.

Comparison of Calibration v Validation Accuracy Results

The results from Table 6 and Table 9 can also be compared to assess the opinion that a model should be validated using data independent of that used to calibrate the model. As discussed in Chapter 2, a parametric model will appear to perform well against Conte’s criteria when validated upon projects that were also used in the model calibration process, but it still may not accurately predict new projects. Therefore, one could expect the calibrated model accuracy statistics to be generally worse for the validation data sets, indicating a poorer fit to the model (poorer estimating accuracy in the validation context). Table 11 shows a direct comparison of average default and calibrated accuracy for all calibration data sets, vs. similar averages for the validation data sets.

Table 11. General Comparison of Model Accuracy

Data Type	Default MMRE	Calibrated MMRE	Default RRMS	Calibrated RRMS	Default PRED (0.25)	Calibrated PRED (0.25)
Calibration Projects (Subset averages)	0.71	0.69	0.99	0.79	0.07	0.35
Validation Projects (Subset averages)	0.79	0.88	1.07	1.16	0.06	0.29

The results in Table 11 were obtained by averaging the accuracy statistics for each data subset (MMRE, RRMS, and PRED) into a single statistic (i.e., for records used in

calibration vs. those used for validation). For example, all calibrated MMRE statistics for the ten categories in the calibration process (Table 6) were averaged to obtain a single calibrated MMRE of 0.69 for all calibration subsets. For resampled data sets, the averages in Tables 6 and 9 were incorporated into the overall averages shown above. Since the data sets did not vary drastically in size, the averages in Table 11 are not weighted by number of records in each data set, nor by number of samples drawn. However, weighting the averages by data set size has been performed in other research (Southwell, 1996: 53), and it could yield significantly different results in this case.

As expected, Table 11 indicates that the calibrated CHECKPOINT model generally estimated projects more accurately when validated on the calibration data than the validation data set. This supports the theory that validating a calibrated model with new data is a more conservative measure of estimating accuracy. Although the default statistics were roughly equal across data type (the slight variations are due to small data set sizes), the calibrated model performed significantly worse on the validation projects than the calibration projects. For example, mean calibrated MMRE was 0.69 for all calibration projects, but was 0.88 for validation projects. Also, mean calibrated RRMS was 0.79 for calibration projects, but was 1.07 for validation projects. Finally, mean PRED (0.25) was 0.35 for calibration projects, but was slightly worse at 0.29 on validation projects.

Summary

After comparing the results of the 1996 Mertes thesis (Table 2) to the calibration and validation results in this chapter, it is immediately apparent that the results of this research do not remotely approach the exceptional results of that study. Simply, the

reasons for the discrepancy of results may lie in two areas: 1) methodology differences between the Mertes study and this research, and 2) the limitations and assumptions associated with the ESC database (see Chapter 3). The conclusions regarding the reasons for these disparate results will be discussed in Chapter 5, along with several limitations associated with these research findings.

Chapter 5

Conclusions and Recommendations

Overview

As a research endeavor within an academic environment, this study did not focus solely upon the pursuit of an improved software cost estimating tool. This thesis also stressed the importance of a viable calibration and validation process in general, sought to explore and enhance the process, and it attempted to foster in the reader a deeper understanding regarding this process. Therefore, the reader should consider that this process-oriented focus, within the framework of a typically results-oriented objective (i.e., model calibration in an actual business environment), may have significantly impacted the research results, thus influencing the conclusions stated herein.

This chapter provides a brief review of the research objectives and discusses the fulfillment of those objectives. Also, it discusses several limitations of this research effort, from limiting assumptions to limitations of data and resources. Since the research results were discussed extensively in Chapter 4, the conclusions in this chapter are stated largely from a qualitative viewpoint; these conclusions include possible reasons for the negative calibration and validation results. Finally, this chapter provides several recommendations for follow-on research.

Review of Research Objective

The primary objective of this research effort was to calibrate the CHECKPOINT model to the Electronic Systems Center software database and to validate improvements in estimating accuracy using the calibrated model. Upon successful validation, this study would offer ESC and other DOD agencies a means of obtaining more accurate software cost estimates using the CHECKPOINT model in specific software development environments.

A corollary goal was to provide results that could be compared to the exceptional results of the 1996 Mertes CHECKPOINT study. Unfortunately, the results did not indicate successful calibration or validation of the model to the ESC database. Thus, the Mertes study still stands alone as the only AFIT calibration effort to achieve such groundbreaking increases in estimating accuracy as a result of model calibration.

Discussion of Research Activities

In support of the primary objective, several research activities were identified in Chapter 1. These activities are repeated here, along with a brief summary discussing the outcome of the activity:

1. Understand the CHECKPOINT model, its functions, and the procedures for generating estimates and templates for calibration.
2. The researcher has a greatly increased understanding of the elements listed above, and it is hoped that a similar understanding will be conveyed to readers of this study.
3. Evaluate the ESC database, which consists of the following steps:
 - a. Eliminate observations lacking complete data required for the calibration process.
This was performed as a result of the database evaluation conducted according to the methodology in Chapter 3. The rationale for the data reduction was documented in Chapter 4, and the reduced database is included in Appendix C.
 - b. Stratify the database into data sets by category, such as language, platform/application, and contractor/management maturity level. Due to

data limitations, the ESC database could not be stratified according to platform/application, which constitutes a primary limitation to this research. Stratification by contractor, language, and both in tandem was accomplished, resulting in ten data categories. The database stratification methodology is discussed in Chapter 3.

- c. Separate the data sets into calibration and validation subsets. The methodology for this procedure was discussed in Chapter 3, and the resulting data sets are presented in Appendix E. Note: The small data sets produced by this research activity resulted in the implementation of the formal resampling procedure described in Chapter 3 and reported in Appendix E.
4. Use the CHECKPOINT model to estimate the costs of all projects and determine the pre-calibration accuracy for the calibration and validation data sets.
5. The default estimates for all projects and the accompanying accuracy statistics are presented in Appendix E.
6. Calibrate the CHECKPOINT model on the selected calibration subsets, employing a refined CHECKPOINT calibration methodology.
7. This was performed as described in Chapter 3, and a step-by-step instruction for this procedure is included in Appendix B.
8. Analyze the calibration results and report improvements in accuracy for all calibration and validation subsets. Analyze results through various statistical measures, report results, and state conclusions.
9. The complete results, in the form of accuracy statistics before and after calibration, are presented in Chapter 4 and Appendix E. The conclusions regarding these results are stated in this chapter.

As indicated by the outcomes above (and in keeping with the process-oriented focus of this study), this research was not altogether unsuccessful, in that several activities were accomplished in support of the overall research objective. In particular, the detailed calibration instructions should provide a useful aid to other software estimators who wish to calibrate CHECKPOINT to a specific development environment.

Limitations of Study

There were several limitations discovered during the conduct of this research that must be considered before viable conclusions can be drawn from the research results:

1. Lack of detailed effort data (granularity): In a 1997 correspondence, Capers Jones of SPR stated that “data which is not granular to at least the phase level is not reliable enough for benchmarking and is dangerous to use” (Jones, 1997). It is important to note that SPR cautions against the use of templates created from

- projects measured at any level higher than task, especially at the project level (Pinis, 1997). Although project (CSCI) effort totals in the ESC database were normalized by phase and then input into CHECKPOINT, this does not constitute true phase-level recorded data, but rather an attempt by the researcher to logically estimate phase-level project effort. As such, these projects can only be considered as measured at the project level.
2. Proprietary nature of model: Since CHECKPOINT is a commercial model, its estimating algorithms are proprietary and unavailable, even to registered owners of the model. This greatly inhibits insight by the calibrator into the underlying logic of CHECKPOINT's estimating equations; thus, it is difficult to assess exactly how a template is modifying these equations. This is a significant limitation when calibrating the CHECKPOINT model, as compared to the complete control over the estimating equations in non-proprietary parametric models such as REVIC (Weber, 1995).
 3. Process-oriented focus of research: The research objective was to determine if the methodology stated herein would yield an acceptably accurate calibrated model, rather than to "massage the data" in pursuit of the optimum accuracy results. Thus, all records with sufficient input data for this study were indiscriminately retained, regardless of empirical evidence of nonconformance. As discussed in Chapter 4, supplementary analyses showed potential improvements in overall accuracy if certain projects were eliminated, but still did not elevate the accuracy to acceptable levels.
 4. Weakness of assumptions: Some of the assumptions (originally stated in Chapter 3) regarding the data used in this study are understood by the author to be tenuous, but were deemed as necessary premises for several procedures in the overall research effort. In particular, the database was constructed solely of SLOC-based projects, which CHECKPOINT must convert to function points, and then back to KLOC, using the technique called backfiring (Chapter 2; Appendix A). Since these SLOC/function point ratios have been questioned in previous research (see discussion in Chapter 2), this backfiring technique potentially inserts another source of variation into the calibrated model. However, it was assumed any variation inserted by this backfiring technique was acceptable.
 5. Other database limitations: The inability to stratify by environmental platform/application category has been stated throughout this research effort; essentially, each estimate was differentiated by estimating on size, total effort, and language only. Also, the ESC database, having been developed for another parametric model, contains information that is not fully compatible for data entry into CHECKPOINT. Another limitation is the incomplete nature of the data records; although the database contained 60 data categories (inputs), few of these were reported in sufficient quantity to be useful. This resulted in small data set sizes, which weakened the robustness of all associated statistics. (As discussed in Chapter 2, small data set size is a common limitation of research in this area).
 6. Researcher background: A final limitation (which is external to the quality of the data, the viability of the CHECKPOINT model, or the overall calibration methodology employed) is the expertise of the researcher. The researcher is a novice to the field of software development, which limited his insight into areas

such as common applications for programming languages, or anomalous data that may be apparent to an expert. In addition, the researcher had only recently learned to use the CHECKPOINT model (self-taught), and errors in his application of model functions for this effort are certainly possible.

10.

Conclusions

After calibration, the CHECKPOINT model generally failed to satisfy the accuracy criteria generally accepted by the software estimating community (stated in Equations 7, 8, and 9). Due to the unavoidable heterogeneity of the data sets used, these criteria were reduced, resulting in Equation 10. Still, the calibration effort was unsuccessful in nine of ten categories for the calibration data, and it was unsuccessful in all ten categories for the validation data. However, it should be noted that the model did exhibit occasional improvements in accuracy, and it performed slightly better for the data sets stratified by both contractor and language simultaneously. The overall accuracy results, by project category, of the model after calibration are summarized in Table 12:

Table 12. Calibration and Validation Summary (Calibrated Model)

Data Type Stratification Category	Calibration Projects				Validation Projects			
	MMR E	RRM S	PRE D (0.25)	Cal. Success	MMR E	RRM S	PRE D (0.25)	Val. Success
Contractor B	0.47	0.44	0.40	No	0.64	0.49	0.25	No
Contractor J	0.51	0.52	0.27	No	1.33	1.43	0.18	No
Ada	0.35	1.01	0.44	No	1.70	2.54	0.50	No
Assembly	1.98	1.16	0.08	No	2.05	1.20	0.18	No
Fortran	0.63	1.12	0.42	No	0.70	2.31	0.17	No
Jovial	1.12	1.57	0.14	No	0.44	0.68	0.43	No
Ada & Contractor R	0.50	0.84	0.25	No	0.39	0.72	0.45	No
CMS2/Assem bly & Contractor M	0.32	0.29	0.50	No	0.69	0.64	0.10	No
Fortran & Contractor A	0.21	0.28	0.75	Yes	0.44	0.88	0.29	No
Jovial & Contractor J	0.79	0.63	0.29	No	0.37	0.70	0.33	No

Due to the limitations summarized earlier in this chapter, it is not cogent to conclusively state that the unsuccessful calibration effort was due to inherent flaws in the CHECKPOINT model (i.e., the assumptions upon which its estimating algorithms are based), especially since those algorithms are unknown. These limitations also weaken conclusions (whether in support or refutation) comparing the methodology and results of the Mertes study to those obtained through this research. Such a comparison was an important element of the primary research objective in this study.

It is the opinion of the author that the overriding cause of CHECKPOINT's unacceptable estimating accuracy is the coarse nature of the ESC database, in terms of its applicability to the model. CHECKPOINT is an extremely sophisticated parametric model with over 100 available inputs (in the detailed estimating mode). Although many inputs were selected to describe the nature of the projects in the ESC database (e.g.,

project goals, classification, complexity, as detailed in Chapter 3), these inputs were applied universally to all projects. These inputs were based on a general knowledge of ESC software development efforts, which is an extreme generalization, given the diversity of software applications even within ESC. This was done for two reasons:

1. It was assumed that these selections would apply to a majority of the projects, in spite of the virtual certainty that they could not possibly apply without exception.
2. An identical assumption (even down to the same input selections) had been universally applied to the SMC projects in the 1996 CHECKPOINT study, which yielded exceptional calibration and validation results (Mertes, 1996). To provide a true comparison of results, the same assumption was necessary in this study.

As a result, these input selections did not differentiate between projects, thereby implying all projects were identical with respect to those inputs. The only data inputs that effectively differentiated between projects were software size, actual effort, and programming language, as shown in Appendix C. Thus, the CHECKPOINT model was forced to estimate each project on these three inputs, plus the collection of inputs that were applied to all ESC projects used in the calibration and validation procedures.

These conclusions strongly reinforce the need for improved software project data collection practices in DOD. Almost two decades ago, Thibodeau stressed the importance of more detailed, complete project tracking and reporting measures for the purposes of building better historical software databases (Thibodeau, 1981: 7-4). He suggested this could be accomplished by increasing data requirements for contracted efforts. Despite the overall size of the SMC and ESC databases amassed since that time, the database stratification process invariably results in small data sets. Obviously, this problem is still prevalent throughout the AFIT studies of the two USAF software databases, and is particularly exemplified by this research. The effects of this trade-off

between project homogeneity and data set size can only be mitigated by more complete project data, enabling more projects to be included in the calibration process.

Recommendations for Follow-on Research

There are several possibilities for follow-on research, which could be accomplished as independent studies or incorporated into other similar research efforts. One early intent of this research effort was to calibrate CHECKPOINT using data stratified by developers' management maturity level, according to the Software Engineering Institute's Capability Maturity Model (Garmus and Herron, 1996: 19). Although not part of their software databases, perhaps this information is available elsewhere in SMC or ESC; alternatively, the researcher (in cooperation with the sponsor) could make a subjective determination of each contractor's level.

In support of several planned in-house calibration efforts, ESC is updating its software database to include many new projects, as well as a complete classification by platform/application type. It was the intention of this researcher to utilize this updated database, but it was unavailable in time to be included in this study. It may be very revealing to explore how additional stratification by application, when coupled with other stratification categories, could improve the homogeneity of the data sets, in turn yielding improvements in CHECKPOINT's estimating accuracy.

There are several other research possibilities which would augment this study, as well as similar research:

1. Perform calibration/ validation procedure employing the detailed estimating mode of CHECKPOINT, if sufficient detailed data is available in the updated version of the ESC database. This also assumes the data can be made compatible with CHECKPOINT input parameters.

2. Use regression analysis to further explore the ESC database for highly correlated projects, which may provide evidence for an improved means of stratifying the ESC database (or similar DOD databases).
3. Calibrate and validate other models to the updated ESC database.
4. Thoroughly investigate anomalous projects in the ESC database, such as those mentioned in Chapter 4 of this study (perhaps revealed by statistical tests for outlying data points). This could accompany a subsequent, optimal results-oriented calibration/validation effort.

As can be seen from the items above, several of these recommendations originate from the limitations stated in this chapter. Given more time and information, these limitations could easily provide the genesis for additional research opportunities.

Appendix A

CHECKPOINT Calibration/Validation Procedure

These instructions are intended as an *unofficial* supplement to the CHECKPOINT for Windows Version 2.3.1 User's Guide and focus on the specific steps necessary to calibrate and validate the model to any specific development environment, as determined by this researcher.

Calibration Instructions

I. Initial Setup

These steps comprise the entry of preliminary information generally assumed to be common to a group of projects in the software development environment. If project classifications, goals, or complexities differ, repeat Steps 4-7 for each group of similar projects prior to beginning Part II for those projects.

Under the **SETUP** menu:

1. Go to Project Settings and:
 - a. Under Project Mode, identify type of estimate.
 - b. Under Work Metric, identify the time units.
 - c. Under Data Entry Level, select Phase.
2. Go to Time Accounting and:
 - a. Under Average Employees Year, change values to reflect the work environment.
 - b. Go to Project Accounting and:
 - 1) Enter values for Accounting hours per business day and Productive hours per project day. (Example: Enter 8.00 for each category.)

- 2) Enter values for Overtime hours per project day and Overtime hours per non-business day. (Example: Enter 0.00 for each category.)
- 3) If using the above example values and leaving the remaining fields at their default values, the Output (Per Year) will be 251 Business days, 231 Project days, and 1,848 Productive hours.
Next, go to Software Metric and select the appropriate sizing metric for both Output software metric and New code software metric.

Under the INPUT menu, select Required Input and do the following:

4. Go to Project Description and enter appropriate information. Dates are required in the Current Date and Project Start fields, but are not used to constrain the schedule in any fashion.
5. Next, go to Project Classification and:
 - a. Under Project nature, identify the project.
 - b. Under Project Scope, choose from the available entries.
 - c. Under Project Class, identify the project.
 - d. If appropriate, check the box labeled Strict military specification.
 - e. Under Project type, identify the project.
6. Go to Project Goals and identify accordingly.
7. Finally, go to Project Complexity and select an appropriate New problem complexity, New code complexity, and New data complexity.
8. Close all windows still open at this point.

II. Generation of Default Estimates

Once Part I has been completed, specific projects can be entered into CHECKPOINT to obtain nominal estimates of effort. To enter the first project,

Under the INPUT menu:

9. Go to Required Input and:
 - a. Select Function Sizing or Source Code, depending on the software sizing metric to be entered. If KLOC was selected for the software metric (Step 3 above), then Function Sizing will not be available.
 - b. If Function Sizing was selected, enter the number of function points for each of the five Function Types.
 - c. If Source Code was selected, choose the appropriate programming language(s) by clicking on the Choose Languages box that is aligned with the appropriate Code Class.
 - 1) When the Choose Languages box is selected, a New Code Multiple Languages window appears. Under Enter New Languages as, select the appropriate metric (e.g., Lines of Code (KLOC)).
 - 2) From the New Language menu, select the appropriate language. (Pages

7-32 to 7-34 of the CHECKPOINT User's Guide provide complete information on how to enter multiple development languages for a single project).

- 3) Enter the size in KLOC for the first project in the Measured KLOC field and click OK.
10. Close the Source Code window.
 11. Return to Required Input menu and:
 - a. Unless specific information is available, leave all fields under the Project Cost submenu at their default values. If the projects are to be estimated in person-months (PM), then this information is not relevant and should be ignored.
 - b. Under Task Selection, click off the box labeled Automatic selection of tasks for development. Ensure the appropriate data entry level button is selected (e.g., if entering data at the phase level, select the Phase button).
 - 1) Select only the development tasks or phases which correspond to the tasks or phases of development included in the effort data for the projects.
 - 2) If information regarding User Involvement in Tasks is known, then click off the Automatic selection of tasks box and select the appropriate phases; otherwise, leave all fields in this section at the default values.
 - 3) Close the Task Selection window.

Under the VIEW menu:

12. Select Development and Task analysis to obtain the default estimate and percentages of effort by phase or task (if not using previously obtained normalization percentages) for the first project. (Note: If actual phase- or task-level effort data is known, then it is not necessary to record these effort percentages.)
 - a. The total default estimate in PM is listed under the Effort Months column heading and should have an asterisk (*) to indicate the number has been estimated by CHECKPOINT, not input directly. (If the asterisk is not present, then under INPUT, Measurement Data Entry, and Development Effort, click the Estimate All button, then Yes.) Record the default estimate.
 - b. If using the phase-level Development Schedule/Effort Percentages estimated by CHECKPOINT, record these numbers for use in Step 14.

Under INPUT:

13. Select Measurement Data Entry and Development Effort.
14. Enter the appropriate Effort Months values for the selected phases. (These values can be calculated by multiplying total actual effort for the project by normalization figures available to the organization, or by using the percentages obtained in Step 12-b.) Click the Next button to advance through each phase.
15. Once all effort percentages have been entered, click the Measure All button, then Yes. The estimated effort for the project (previously described in Step 12-a) will now reflect the actual effort for the completed project. (The asterisk will now be

absent from the effort total, which shows that the effort months were input into the model rather than estimated.)

16. Close the Development Task Analysis window.
17. Under the FILE menu, select Save as and assign a filename that adequately describes the first project. Under the Version label field, type “default” to indicate this version as the default estimate.

Now that the first project has been entered and saved, all remaining projects can be entered by opening the prior one, inputting values for a new project (e.g., differing size, effort, and language information), and saving it with a new name. Remember to toggle between the Estimate all and the Measure all buttons to obtain the default estimates and then to save the projects. To open a previously saved project:

18. Under the FILE menu, select Open and Project.
19. Highlight the desired project and click OK.
20. Under Open Version, select the version of the project to be opened and click OK.
21. Close the Project Description window and continue with steps 9-17.

III. Template Construction

Once all projects have been entered, a portfolio can be created to group homogeneous projects together according to the previously determined stratification categories. For each portfolio, select only the projects that are considered to be part of the calibration subset for that project category.

1. Under FILE, select New and Portfolio.
2. Highlight the records to be included in the portfolio by holding down the CTRL key while selecting projects. After all applicable projects have been selected, accept all default options in this window and click OK.
3. Review the contents of the Versions window to verify that the correct projects have been selected. Under FILE, select Save as, assign a filename and portfolio description that appropriately describes the data set and click OK. (Portfolios will be identifiable in the CHECKPOINT directory by a .prt file suffix.)

The final step in the CHECKPOINT calibration process is to create a template for each portfolio. To accomplish this,

4. Under FILE, select New and Template.

5. In the **New Template** window, leave all default values as they appear on the screen and click **OK**. (**CHECKPOINT** will default to the portfolio filename and description for saving; unless another name and description is desired, accept this name. Templates will be identifiable by a **.kb** file suffix.)
6. Repeat Steps 1-5 to create a template for each stratification category.

Validation Instructions

Once a template has been created for each stratified data set using its applicable calibration subset, all projects must be estimated again, now using the template for that data set. The step-by-step procedure for using calibration templates to obtain new effort estimates for projects is as follows:

1. Under the **FILE** menu, select **Open and Project**.
2. Highlight the desired project and click **OK**.
3. Close the **Project Description** window. Under the **VIEW** menu, select **Development and Task Analysis** to open the window containing the existing effort estimate for the project. Leave this window open for the remaining steps.
4. Under the **INPUT** menu, select **Measurement Data Entry and Development Effort**. Click the **Estimate all** button, then **Yes**, and review the **Task Analysis** window to ensure an asterisk now appears next to the effort estimate.
5. Under the **FILE** menu, select **Use Template**, highlight the appropriate template for the project, and click **OK**. When the **Template Options** window appears, click **OK** to accept the default values. (In this study, the default values were used.)
6. The effort estimate in the **Task Analysis** window will now have a plus (+) sign to indicate that the estimate was calculated using a calibration template. Record this estimate (which will be used either to evaluate the success of calibration, or to validate improvements in **CHECKPOINT**'s estimating accuracy).
7. Repeat Steps 1-6 for all remaining projects.

Appendix B

Reduced ESC Database

Record #	KTR Code	EDSI (KLOC)	Programming Language(s)	Actual Total Effort (PM)
1	B	104.6	CMS 2	646.6
2	B	22.6	CMS 2	101.9
3	B	152.2	CMS 2L	1,218.2
4	B	250.0	CMS 2	2,301.3
5	B	27.8	CMS 2	205.9
6	B	15.9	CMS 2	211.3
7	B	35.0	PASCAL	90.1
8	B	24.0	FORTRAN	111.5
9	D	10.7	ADA	114.8
10	E	17.9	ASSEMBLY	56.0
11	E	13.6	ASSEMBLY	12.5
12	E	27.1	ASSEMBLY	72.4
13	F	21.3	CMS 2M	241.0
14	N/R	181.4	JOVIAL	1,791.0
15	J	11.3	JOVIAL	80.7
16	J	6.1	C	85.9
17	J	79.3	JOVIAL	1,241.9
18	J	6.5	ASSEMBLY	211.8
19	J	57.8	FORTRAN 71%, ASSEMBLY 29%	441.0
20	J	21.8	JOVIAL	360.2
21	J	11.7	JOVIAL	18.0
22	J	11.5	ASSEMBLY	169.3
23	J	63.4	JOVIAL 85%, ASSEMBLY 15%	429.9
24	J	9.1	JOVIAL 95%, ASSEMBLY 5%	115.3
25	J	13.3	JOVIAL 99%, ASSEMBLY 1%	131.5
26	J	73.1	JOVIAL	385.2
27	J	36.4	JOVIAL 86.4%, ASSEMBLY 13.6%	161.0
28	J	5.8	JOVIAL 95.6%, ASSEMBLY 4.4%	28.0
29	J	5.4	JOVIAL 99%, ASSEMBLY 1%	56.8
30	J	8.8	JOVIAL	94.4
31	K	22.9	FORTRAN	122.6
32	K	20.5	FORTRAN	335.7
33	C	13.4	FORTRAN	565.2
34	W	40.3	FORTRAN	1,259.8
35	L	20.1	ADA	124.8

Record #	KTR Code	EDSI (KLOC)	Programming Language(s)	Actual Total Effort (PM)
36	O	17.3	ASSEMBLY	55.1
37	O	17.7	ASSEMBLY	60.5
38	O	34.1	ASSEMBLY	96.8
39	O	3.0	ASSEMBLY	14.7
40	P	3.5	ADA	18.0
41	R	21.3	ADA	224.0
42	R	33.4	ADA	335.2
43	R	3.3	ADA	81.0
44	R	53.0	ADA	406.0
45	R	22.0	ADA	216.6
46	R	18.1	ADA	230.0
47	R	27.4	ADA	178.0
48	R	37.2	ADA	256.0
49	R	184.0	ADA	1,000.0
50	R	40.7	ADA	217.0
51	B	8.0	ASSB 49%, FORTRAN 44%	132.4
52	U	35.0	ADA	261.7
53	U	22.4	ASSEMBLY	152.3
54	U	39.5	ASSEMBLY	301.0
55	A	9.8	FORTRAN	83.5
56	A	8.5	FORTRAN	89.0
57	A	8.7	FORTRAN	63.0
58	A	2.6	FORTRAN	26.5
59	A	3.1	FORTRAN	89.0
60	A	7.7	FORTRAN	71.5
61	A	5.1	FORTRAN	87.0
62	A	11.3	FORTRAN	90.5
63	A	3.7	FORTRAN	53.0
64	A	6.6	FORTRAN	60.0
65	A	5.4	FORTRAN	83.0
66	A	27.7	FORTRAN	162.0
67	A	2.8	FORTRAN	30.3
68	A	11.2	FORTRAN	110.0
69	A	2.3	FORTRAN	28.3
70	J	72.3	FORTRAN	1,456.9
71	J	54.7	FORTRAN	125.7
72	J	2.7	JOVIAL	33.4
73	J	49.6	FORTRAN 75%, ASSY 25%	369.1
74	J	18.9	JOVIAL	172.3

Record #	KTR Code	EDSI (KLOC)	Programming Language(s)	Actual Total Effort (PM)
75	J	52.9	JOVIAL	430.3
76	X	148.3	ADA	119.0
77	Y	6.6	FORTRAN	51.0
78	Z	40.7	ASSEMBLY	333.7
79	Z	6.5	ASSEMBLY	461.1
80	N/R	26.9	ASSEMBLY	298.9
81	N/R	20.5	ASSEMBLY	278.9
82	N/R	14.8	ASSEMBLY	1,413.7
83	N/R	14.8	ASSEMBLY	936.8
84	N/R	11.0	ASSEMBLY	850.5
85	N/R	25.2	ASSEMBLY	300.0
86	N/R	12.8	ASSEMBLY	175.8
87	N/R	21.5	ASSEMBLY	212.6
88	Z	32.1	ASSEMBLY	1,010.5
89	N/R	38.0	ASSEMBLY	364.2
90	G	23.1	ADA	201.8
91	G	56.9	ADA 99%, C 1%	405.5
92	M	16.4	CMS-2 90%, ASSB 10%	515.3
93	M	2.6	CMS-2 90%, ASSB 10%	147.5
94	M	22.3	CMS-2 90%, ASSB 10%	604.4
95	M	27.7	CMS-2 90%, ASSB 10%	357.9
96	M	28.2	CMS-2 90%, ASSB 10%	295.9
97	M	3.6	CMS-2 90%, ASSB 10%	67.6
98	M	41.7	CMS-2 90%, ASSB 10%	1,063.5
99	M	31.6	CMS-2 90%, ASSB 10%	927.5
100	M	2.9	CMS-2 90%, ASSB 10%	132.1
101	M	52.4	CMS-2 90%, ASSB 10%	992.5
102	T	116.1	FORTRAN	445.5

Appendix C

Normalized Effort for Projects

Record #	Estimated Effort % by Phase					Person-Months of Effort by Phase				
	Requirements	External Design	Internal Design	Coding	Integrat. and Test	Requirements	External Design	Internal Design	Coding	Integrat. and Test
1	7.4	18.1	18.2	30.7	25.6	47.8	117.0	117.7	198.5	165.5
2	9.4	18.3	15.4	34.1	22.8	9.6	18.6	15.7	34.7	23.2
3	10.3	20.3	16.6	27.3	25.5	125.5	247.3	202.2	332.6	310.6
4	9.8	19.5	16.0	25.5	29.2	225.5	448.8	368.2	586.8	672.0
5	9.0	18.0	15.4	34.9	22.6	18.5	37.1	31.7	71.9	46.5
6	9.3	17.5	15.4	35.4	22.4	19.7	37.0	32.5	74.8	47.3
7	9.5	19.5	16.9	29.9	24.2	8.6	17.6	15.2	26.9	21.8
8	9.2	18.0	15.1	35.2	22.5	10.3	20.1	16.8	39.2	25.1
9	11.1	20.9	17.9	25.9	24.1	12.7	24.0	20.5	29.7	27.7
10	5.1	8.4	7.3	62.8	16.4	2.9	4.7	4.1	35.2	9.2
11	5.1	8.3	7.2	62.6	16.7	0.6	1.0	0.9	7.8	2.1
12	4.8	8.7	7.9	63.2	15.5	3.5	6.3	5.7	45.8	11.2
13	9.5	17.8	15.4	34.6	22.7	22.9	42.9	37.1	83.4	54.7
14	10.1	20.1	16.4	26.7	26.7	180.9	360.0	293.7	478.2	478.2
15	9.4	16.9	15.8	34.8	23.1	7.6	13.6	12.8	28.1	18.6
16	9.3	15.0	12.1	41.1	22.5	8.0	12.9	10.4	35.3	19.3
17	7.8	18.0	16.8	32.2	25.1	96.9	223.5	208.6	399.9	311.7
18	5.2	8.6	6.3	63.5	16.4	11.0	18.2	13.3	134.5	34.7
19	7.7	16.0	14.4	38.9	22.9	34.0	70.6	63.5	171.5	101.0
20	9.4	18.2	15.4	34.3	22.7	33.9	65.6	55.5	123.5	81.8
21	9.3	17.0	15.8	34.7	23.1	1.7	3.1	2.8	6.2	4.2
22	5.1	8.3	7.0	62.9	16.7	8.6	14.1	11.9	106.5	28.3
23	8.0	17.4	15.4	35.5	23.6	34.4	74.8	66.2	152.6	101.5
24	9.5	17.0	15.1	35.2	23.1	11.0	19.6	17.4	40.6	26.6
25	9.6	17.3	15.7	34.7	22.7	12.6	22.7	20.6	45.6	29.9
26	8.0	18.1	17.0	32.4	24.5	30.8	69.7	65.5	124.8	94.4
27	8.3	17.2	14.8	36.5	23.1	13.4	27.7	23.8	58.8	37.2
28	10.0	16.6	13.7	36.3	23.3	2.8	4.6	3.8	10.2	6.5
29	10.1	16.6	13.6	36.1	23.6	5.7	9.4	7.7	20.5	13.4
30	9.6	17.0	15.0	35.4	23.1	9.1	16.0	14.2	33.4	21.8
31	9.2	18.0	15.1	35.3	22.4	11.3	22.1	18.5	43.3	27.5
32	9.5	17.6	15.4	34.6	22.9	31.9	59.1	51.7	116.2	76.9
33	9.6	17.3	15.7	34.8	22.6	54.3	97.8	88.7	196.7	127.7
34	9.0	18.4	16.1	33.0	23.4	113.4	231.8	202.8	415.7	294.8
35	10.8	21.7	18.1	25.0	24.4	13.5	27.1	22.6	31.2	30.4
36	5.1	8.4	7.3	63.1	16.2	2.8	4.6	4.0	34.8	8.9
37	5.0	8.4	7.2	63.0	16.3	3.0	5.1	4.4	38.1	9.9
38	4.9	8.7	8.8	62.2	15.4	4.7	8.4	8.5	60.2	14.9

Record #	Estimated Effort % by Phase					Person-Months of Effort by Phase				
	Require-ments	External Design	Internal Design	Coding	Integrat. and Test	Require-ments	External Design	Internal Design	Coding	Integrat. and Test
39	5.3	7.7	6.4	63.8	16.8	0.8	1.1	0.9	9.4	2.5
40	11.9	19.3	15.6	27.6	25.7	2.1	3.5	2.8	5.0	4.6
41	10.7	21.6	18.2	24.8	24.7	24.0	48.4	40.8	55.6	55.3
42	10.2	21.3	18.7	24.4	25.4	34.2	71.4	62.7	81.8	85.1
43	12.0	19.4	15.3	27.5	25.7	9.7	15.7	12.4	22.3	20.8
44	9.4	21.7	19.9	23.4	25.5	38.2	88.1	80.8	95.0	103.5
45	10.4	21.5	17.8	25.7	24.5	22.5	46.6	38.6	55.7	53.1
46	10.9	21.7	18.1	25.2	24.2	25.1	49.9	41.6	58.0	55.7
47	10.5	21.4	18.4	24.8	24.9	18.7	38.1	32.8	44.1	44.3
48	10.0	21.7	18.7	24.0	25.4	25.6	55.6	47.9	61.4	65.0
49	11.1	22.3	18.0	18.4	30.2	111.0	223.0	180.0	184.0	302.0
50	10.0	21.3	18.9	24.7	25.1	21.7	46.2	41.0	53.6	54.5
51	8.1	13.0	10.8	47.0	21.1	10.7	17.2	14.3	62.2	27.9
52	10.2	21.6	18.7	24.4	25.1	26.7	56.5	48.9	63.9	65.7
53	5.0	8.7	7.7	62.5	16.1	7.6	13.3	11.7	95.2	24.5
54	4.9	8.9	9.0	62.0	15.2	14.7	26.8	27.1	186.6	45.8
55	9.3	16.8	15.2	35.2	23.4	7.8	14.0	12.7	29.4	19.5
56	9.5	17.0	14.8	35.3	23.4	8.5	15.1	13.2	31.4	20.8
57	9.6	17.0	15.0	35.3	23.1	6.0	10.7	9.5	22.2	14.6
58	9.9	16.5	11.9	37.4	24.3	2.6	4.4	3.2	9.9	6.4
59	10.3	16.3	12.4	36.9	24.1	9.2	14.5	11.0	32.8	21.4
60	9.6	16.8	14.4	35.8	23.4	6.9	12.0	10.3	25.6	16.7
61	10.2	16.5	13.5	36.4	23.4	8.9	14.4	11.7	31.7	20.4
62	9.4	16.8	15.8	34.9	23.1	8.5	15.2	14.3	31.6	20.9
63	10.1	16.3	12.7	37.2	23.7	5.4	8.6	6.7	19.7	12.6
64	9.7	16.6	13.8	36.4	23.6	5.8	10.0	8.3	21.8	14.2
65	10.1	16.5	13.6	36.3	23.6	8.4	13.7	11.3	30.1	19.6
66	9.1	18.0	15.4	34.9	22.7	14.7	29.2	24.9	56.5	36.8
67	9.9	16.3	12.1	38.1	23.6	3.0	4.9	3.7	11.5	7.1
68	9.4	16.7	15.8	35.0	23.1	10.3	18.4	17.4	38.5	25.4
69	10.0	16.3	11.8	37.3	24.6	2.8	4.6	3.3	10.5	6.9
70	8.0	18.1	17.0	32.4	24.5	116.6	263.7	247.7	472.0	356.9
71	8.4	18.2	15.9	33.5	24.1	10.6	22.9	20.0	42.1	30.3
72	9.9	16.3	12.2	37.8	23.9	3.3	5.4	4.1	12.6	8.0
73	8.1	16.5	14.7	38.3	22.4	29.9	60.9	54.3	141.4	82.7
74	9.1	17.6	15.4	35.0	22.8	15.7	30.3	26.5	60.3	39.3
75	8.4	18.2	15.9	33.6	23.9	36.1	78.3	68.4	144.6	102.8
76	11.6	23.0	18.5	19.0	27.9	13.8	27.4	22.0	22.6	33.2

Record #	Estimated Effort % by Phase					Person-Months of Effort by Phase				
	Require-ments	External Design	Internal Design	Coding	Integrat. and Test	Require-ments	External Design	Internal Design	Coding	Integrat. and Test
77	9.7	16.6	13.9	36.0	23.8	4.9	8.5	7.1	18.4	12.1
78	4.8	8.7	8.9	62.1	15.5	16.0	29.0	29.7	207.2	51.7
79	5.3	8.6	6.3	63.4	16.5	24.4	39.7	29.0	292.3	76.1
80	4.8	8.6	7.8	63.4	15.5	14.3	25.7	23.3	189.5	46.3
81	5.0	8.6	7.6	62.7	16.1	13.9	24.0	21.2	174.9	44.9
82	5.1	8.3	7.1	63.2	16.3	72.1	117.3	100.4	893.5	230.4
83	5.1	8.3	7.1	63.2	16.3	47.8	77.8	66.5	592.1	152.7
84	5.1	8.3	6.8	63.2	16.7	43.4	70.6	57.8	537.5	142.0
85	4.8	8.5	7.7	63.4	15.5	14.4	25.5	23.1	190.2	46.5
86	5.2	8.3	7.2	62.6	16.6	9.1	14.6	12.7	110.1	29.2
87	5.0	8.7	7.6	62.6	16.1	10.6	18.5	16.2	133.1	34.2
88	4.9	8.6	8.6	62.6	15.4	49.5	86.9	86.9	632.6	155.6
89	5.0	8.6	9.0	62.1	15.3	18.2	31.3	32.8	226.2	55.7
90	10.4	21.5	18.0	25.5	24.6	21.0	43.4	36.3	51.5	49.6
91	9.3	22.0	20.1	23.1	25.5	37.7	89.2	81.5	93.7	103.4
92	9.1	17.0	15.1	36.9	21.9	46.9	87.6	77.8	190.1	112.9
93	9.7	15.8	11.2	39.3	23.9	14.3	23.3	16.5	58.0	35.3
94	9.2	17.3	15.0	36.1	22.4	55.6	104.6	90.7	218.2	135.4
95	8.8	17.5	15.0	36.6	22.0	31.5	62.6	53.7	131.0	78.7
96	8.8	17.6	15.0	36.5	22.1	26.0	52.1	44.4	108.0	65.4
97	9.9	15.7	11.8	39.1	23.5	6.7	10.6	8.0	26.4	15.9
98	8.5	17.4	14.7	36.3	23.0	90.4	185.0	156.3	386.1	244.6
99	8.7	17.6	15.1	36.2	22.4	80.7	163.2	140.1	335.8	207.8
100	9.6	15.9	11.8	39.3	23.4	12.7	21.0	15.6	51.9	30.9
101	8.3	17.3	15.5	35.4	23.5	82.4	171.7	153.8	351.3	233.2
102	10.5	20.6	16.7	28.0	24.2	46.8	91.8	74.4	124.7	107.8

Appendix D

Calibration and Validation Results

Stratification Category: Contractor B

(1ST ITERATION)

Calibration Subset						
Record #	Size (KLOC)	Actual Effort (PM)	Default Effort (PM)	Calibrated Effort (PM)	MRE (Default)	MRE (Calibrated)
1	104.6	646.6	317.48	1,053.91	0.51	0.63
2	22.6	101.9	48.44	158.83	0.52	0.56
5	27.8	205.9	62.03	199.23	0.70	0.03
6	15.9	211.3	32.28	105.52	0.85	0.50
51	8.0	132.4	11.26	35.95	0.91	0.73
			<u>Default</u>	<u>Calibrated</u>		
MMRE			0.70	0.49		
RMS			189.00	194.77		
RRMS			0.73	0.75		
PRED (0.25)			0.00	0.20		

Validation Subset						
Record #	Size (KLOC)	Actual Effort (PM)	Default Effort (PM)	Calibrated Effort (PM)	MRE (Default)	MRE (Calibrated)
3	152.2	1,218.2	534.30	1,819.81	0.56	0.49
4	250.0	2,301.3	988.38	3,444.70	0.57	0.50
7	35.0	90.1	92.43	295.71	0.03	2.28
8	24.0	111.5	52.77	169.52	0.53	0.52
			<u>Default</u>	<u>Calibrated</u>		
MMRE			0.42	0.95		
RMS			662.56	585.65		
RRMS			0.71	0.63		
PRED (0.25)			0.25	0.00		

Contractor B (cont.)

(2ND ITERATION)

Calibration Subset						
Record #	Size (KLOC)	Actual Effort (PM)	Default Effort (PM)	Calibrated Effort (PM)	MRE (Default)	MRE (Calibrated)
1	104.6	646.6	317.48	895.28	0.51	0.38
4	250.0	2,301.3	988.38	2,951.81	0.57	0.28
6	15.9	211.3	32.28	89.18	0.85	0.58
7	35.0	90.1	92.43	252.32	0.03	1.80
8	24.0	111.5	52.77	145.27	0.53	0.30
			<u>Default</u>	<u>Calibrated</u>		
MMRE			0.50	0.67		
RMS			611.16	324.77		
RRMS			0.91	0.48		
PRED (0.25)			0.20	0.00		

Validation Subset						
Record #	Size (KLOC)	Actual Effort (PM)	Default Effort (PM)	Calibrated Effort (PM)	MRE (Default)	MRE (Calibrated)
2	22.6	101.9	48.44	134.10	0.52	0.32
3	152.2	1,218.2	534.30	1,574.67	0.56	0.29
5	27.8	205.9	62.03	170.34	0.70	0.17
51	8.0	132.4	11.26	30.42	0.91	0.77
			<u>Default</u>	<u>Calibrated</u>		
MMRE			0.67	0.39		
RMS			318.10	167.20		
RRMS			0.77	0.40		
PRED (0.25)			0.00	0.25		

Contractor B (cont.)

(3RD ITERATION)

Calibration Subset						
Record #	Size (KLOC)	Actual Effort (PM)	Default Effort (PM)	Calibrated Effort (PM)	MRE (Default)	MRE (Calibrated)
1	104.6	646.6	317.48	802.48	0.51	0.24
2	22.6	101.9	48.44	120.74	0.52	0.18
3	152.2	1,218.2	534.30	1,380.09	0.56	0.13
4	250.0	2,301.3	988.38	2,599.61	0.57	0.13
5	27.8	205.9	62.03	152.53	0.70	0.26
			<u>Default</u>	<u>Calibrated</u>		
MMRE			0.57	0.19		
RMS			681.67	168.94		
RRMS			0.76	0.19		
PRED (0.25)			0.00	0.80		

Validation Subset						
Record #	Size (KLOC)	Actual Effort (PM)	Default Effort (PM)	Calibrated Effort (PM)	MRE (Default)	MRE (Calibrated)
6	15.9	211.3	32.28	79.82	0.85	0.62
7	35.0	90.1	92.43	226.00	0.03	1.51
8	24.0	111.5	52.77	129.60	0.53	0.16
51	8.0	132.4	11.26	27.75	0.91	0.79
			<u>Default</u>	<u>Calibrated</u>		
MMRE			0.58	0.77		
RMS			100.18	96.99		
RRMS			0.73	0.71		
PRED (0.25)			0.25	0.25		

Contractor B (cont.)

(4TH ITERATION)

Calibration Subset						
Record #	Size (KLOC)	Actual Effort (PM)	Default Effort (PM)	Calibrated Effort (PM)	MRE (Default)	MRE (Calibrated)
2	22.6	101.9	48.44	128.87	0.52	0.26
3	152.2	1,218.2	534.30	1,422.05	0.56	0.17
5	27.8	205.9	62.03	161.70	0.70	0.21
7	35.0	90.1	92.43	240.25	0.03	1.67
8	24.0	111.5	52.77	137.82	0.53	0.24
			<u>Default</u>	<u>Calibrated</u>		
MMRE			0.47	0.51		
RMS			314.56	116.17		
RRMS			0.91	0.34		
PRED (0.25)			0.20	0.60		

Validation Subset						
Record #	Size (KLOC)	Actual Effort (PM)	Default Effort (PM)	Calibrated Effort (PM)	MRE (Default)	MRE (Calibrated)
1	104.6	646.6	317.48	824.64	0.51	0.28
4	250.0	2,301.3	988.38	2,587.30	0.57	0.12
6	15.9	211.3	32.28	85.69	0.85	0.59
51	8.0	132.4	11.26	29.18	0.91	0.78
			<u>Default</u>	<u>Calibrated</u>		
MMRE			0.71	0.44		
RMS			612.99	167.29		
RRMS			0.74	0.20		
PRED (0.25)			0.00	0.25		

Stratification Category: Contractor J

Calibration Subset						
Record #	Size (KLOC)	Actual Effort (PM)	Default Effort (PM)	Calibrated Effort (PM)	MRE (Default)	MRE (Calibrated)
16	6.1	85.9	9.44	48.46	0.89	0.44
17	79.3	1,241.9	226.27	1,106.02	0.82	0.11
23	63.4	429.9	169.40	768.80	0.61	0.79
24	9.1	115.3	16.36	84.91	0.86	0.26
27	36.4	161.0	80.35	407.73	0.50	1.53
28	5.8	28.0	9.92	51.18	0.65	0.83
29	5.4	56.8	9.32	48.16	0.84	0.15
70	72.3	1,456.9	198.99	972.31	0.86	0.33
73	49.6	369.1	110.81	540.78	0.70	0.47
74	18.9	172.3	39.28	203.10	0.77	0.18
75	52.9	430.3	135.59	664.62	0.68	0.54
			<u>Default</u>	<u>Calibrated</u>		
MMRE			0.74	0.51		
RMS			511.48	216.85		
RRMS			1.24	0.52		
PRED (0.25)			0.00	0.27		

Contractor J (cont.)

Validation Subset						
Record #	Size (KLOC)	Actual Effort (PM)	Default Effort (PM)	Calibrated Effort (PM)	MRE (Default)	MRE (Calibrated)
15	11.3	80.7	21.32	111.09	0.74	0.38
18	6.5	211.8	6.73	33.92	0.97	0.84
19	57.8	441.0	130.62	640.77	0.70	0.45
20	21.8	360.2	46.48	238.31	0.87	0.34
21	11.7	18.0	22.25	116.10	0.24	5.45
22	11.5	169.3	12.70	63.90	0.92	0.62
25	13.3	131.5	25.78	134.40	0.80	0.02
26	73.1	385.2	201.30	984.17	0.48	1.55
30	8.8	94.4	15.74	81.66	0.83	0.13
71	54.7	125.7	140.65	689.12	0.12	4.48
72	2.7	33.4	4.43	22.76	0.87	0.32
			<u>Default</u>	<u>Calibrated</u>		
MMRE			0.69	1.33		
RMS			169.79	267.06		
RRMS			0.91	1.43		
PRED (0.25)			0.18	0.18		

Stratification Category: Ada Projects

Calibration Subset						
Record #	Size (KLOC)	Actual Effort (PM)	Default Effort (PM)	Calibrated Effort (PM)	MRE (Default)	MRE (Calibrated)
35	20.1	124.8	56.46	111.07	0.55	0.11
41	21.3	224.0	60.53	119.03	0.73	0.47
45	22.0	216.6	63.86	123.87	0.71	0.43
46	18.1	230.0	50.35	99.23	0.78	0.57
48	37.2	256.0	119.90	236.40	0.53	0.08
49	184.0	1,000.0	974.90	1,944.36	0.03	0.94
50	40.7	217.0	135.42	265.43	0.38	0.22
90	23.1	201.8	67.49	130.97	0.67	0.35
91	56.9	405.5	204.04	406.39	0.50	0.00
			<u>Default</u>	<u>Calibrated</u>		
MMRE			0.54	0.35		
RMS			138.02	322.57		
RRMS			0.43	1.01		
PRED (0.25)			0.11	0.44		

Validation Subset						
Record #	Size (KLOC)	Actual Effort (PM)	Default Effort (PM)	Calibrated Effort (PM)	MRE (Default)	MRE (Calibrated)
9	10.7	114.8	27.21	53.56	0.76	0.53
40	3.5	18.0	7.61	14.86	0.58	0.17
42	33.4	335.2	105.71	207.86	0.68	0.38
43	3.3	81.0	7.14	13.94	0.91	0.83
44	53.0	406.0	187.45	370.29	0.54	0.09
47	27.4	178.0	84.86	165.28	0.52	0.07
52	35.0	261.7	111.11	218.95	0.58	0.16
76	148.3	119.0	728.13	1,468.62	5.12	11.34
			<u>Default</u>	<u>Calibrated</u>		
MMRE			1.21	1.70		
RMS			253.98	480.78		
RRMS			1.34	2.54		
PRED (0.25)			0.00	0.50		

Stratification Category: Assembly Projects

Calibration Subset						
Record #	Size (KLOC)	Actual Effort (PM)	Default Effort (PM)	Calibrated Effort (PM)	MRE (Default)	MRE (Calibrated)
10	17.9	56.0	20.32	265.87	0.64	3.75
12	27.1	72.4	32.32	407.86	0.55	4.63
37	17.7	60.5	20.04	262.28	0.67	3.34
38	34.1	96.8	41.80	549.39	0.57	4.68
39	3.0	14.7	3.12	40.52	0.79	1.76
53	22.4	152.3	25.46	333.24	0.83	1.19
54	39.5	301.0	49.38	649.06	0.84	1.16
78	40.7	333.7	52.33	687.32	0.84	1.06
81	20.5	278.9	23.30	304.86	0.92	0.09
83	14.8	936.8	16.72	215.22	0.98	0.77
84	11.0	850.5	12.07	157.98	0.99	0.81
88	32.1	1,010.5	38.97	511.61	0.96	0.49
			<u>Default</u>	<u>Calibrated</u>		
MMRE			0.80	1.98		
RMS			476.53	401.20		
RRMS			1.37	1.16		
PRED (0.25)			0.00	0.08		

Assembly Projects (cont.)

Validation Subset						
Record #	Size (KLOC)	Actual Effort (PM)	Default Effort (PM)	Calibrated Effort (PM)	MRE (Default)	MRE (Calibrated)
11	13.6	12.5	15.06	196.62	0.20	14.73
18	6.5	211.8	6.73	88.50	0.97	0.58
22	11.5	169.3	12.70	166.02	0.92	0.02
36	17.3	55.1	19.58	256.31	0.64	3.65
79	6.5	461.1	6.71	88.28	0.99	0.81
80	26.9	298.9	31.98	403.40	0.89	0.35
82	14.8	1,413.7	16.72	215.15	0.99	0.85
85	25.2	300.0	29.90	375.60	0.90	0.25
86	12.8	175.8	14.18	184.99	0.92	0.05
87	21.5	212.6	24.40	319.65	0.89	0.50
89	38.0	364.2	47.27	621.87	0.87	0.71
			<u>Default</u>	<u>Calibrated</u>		
MMRE			0.83	2.05		
RMS			479.78	399.96		
RRMS			1.44	1.20		
PRED (0.25)			0.09	0.18		

Stratification Category: Fortran Projects

Calibration Subset						
Record #	Size (KLOC)	Actual Effort (PM)	Default Effort (PM)	Calibrated Effort (PM)	MRE (Default)	MRE (Calibrated)
31	22.9	122.6	50.19	216.26	0.59	0.76
34	40.3	1,259.8	96.16	412.87	0.92	0.67
55	9.8	83.5	18.18	79.93	0.78	0.04
57	8.7	63.0	15.72	68.61	0.75	0.09
60	7.7	71.5	13.66	59.63	0.81	0.17
63	3.7	53.0	6.27	27.48	0.88	0.48
66	27.7	162.0	61.83	267.65	0.62	0.65
68	11.2	110.0	21.12	93.54	0.81	0.15
69	2.3	28.3	3.86	16.79	0.86	0.41
70	72.3	1,456.9	198.99	833.49	0.86	0.43
71	54.7	125.7	140.65	595.97	0.12	3.74
77	6.6	51.0	11.59	50.64	0.77	0.01
			<u>Default</u>	<u>Calibrated</u>		
MMRE			0.73	0.63		
RMS			497.83	335.19		
RRMS			1.67	1.12		
PRED (0.25)			0.08	0.42		

Fortran Projects (cont.)

Validation Subset						
Record #	Size (KLOC)	Actual Effort (PM)	Default Effort (PM)	Calibrated Effort (PM)	MRE (Default)	MRE (Calibrated)
7	35.0	90.1	92.43	228.75	0.03	1.54
32	20.5	335.7	43.25	189.92	0.87	0.43
33	13.4	565.2	25.83	115.39	0.95	0.80
56	8.5	89.0	15.31	66.50	0.83	0.25
58	2.6	26.5	4.24	18.33	0.84	0.31
59	3.1	89.0	5.17	22.55	0.94	0.75
61	5.1	87.0	8.73	38.13	0.90	0.56
62	11.3	90.5	21.34	94.58	0.76	0.05
64	6.6	60.0	11.45	50.06	0.81	0.17
65	5.4	83.0	9.35	41.00	0.89	0.51
67	2.8	30.3	4.60	19.97	0.85	0.34
102	116.1	445.5	389.85	1,675.33	0.12	2.76
			<u>Default</u>	<u>Calibrated</u>		
MMRE			0.73	0.70		
RMS			185.26	383.48		
RRMS			1.12	2.31		
PRED (0.25)			0.17	0.17		

Stratification Category: Jovial Projects

Calibration Subset						
Record #	Size (KLOC)	Actual Effort (PM)	Default Effort (PM)	Calibrated Effort (PM)	MRE (Default)	MRE (Calibrated)
14	181.4	1,791.0	661.10	3,311.87	0.63	0.85
20	21.8	360.2	46.48	219.24	0.87	0.39
21	11.7	18.0	22.25	116.10	0.24	5.45
24	9.1	115.3	16.36	71.75	0.86	0.38
30	8.8	94.4	15.74	68.89	0.83	0.27
72	2.7	33.4	4.43	19.11	0.87	0.43
74	18.9	172.3	39.28	181.01	0.77	0.05
			<u>Default</u>	<u>Calibrated</u>		
MMRE			0.72	1.12		
RMS			448.75	578.84		
RRMS			1.22	1.57		
PRED (0.25)			0.14	0.14		

Validation Subset						
Record #	Size (KLOC)	Actual Effort (PM)	Default Effort (PM)	Calibrated Effort (PM)	MRE (Default)	MRE (Calibrated)
15	11.3	80.7	21.32	95.15	0.74	0.18
17	79.3	1,241.9	226.27	1,046.18	0.82	0.16
25	13.3	131.5	25.78	119.01	0.80	0.09
26	73.1	385.2	201.30	926.53	0.48	1.41
28	5.8	28.0	9.92	42.88	0.65	0.53
29	5.4	56.8	9.32	40.38	0.84	0.29
75	52.9	430.3	135.59	627.09	0.68	0.46
			<u>Default</u>	<u>Calibrated</u>		
MMRE			0.71	0.44		
RMS			408.74	230.19		
RRMS			1.22	0.68		
PRED (0.25)			0.00	0.43		

Stratification Category: Ada Projects by Contractor R

(1ST ITERATION)

Calibration Subset						
Record #	Size (KLOC)	Actual Effort (PM)	Default Effort (PM)	Calibrated Effort (PM)	MRE (Default)	MRE (Calibrated)
43	3.3	81.0	7.14	19.39	0.91	0.76
44	53.0	406.0	187.45	517.82	0.54	0.28
45	22.0	216.6	63.86	171.37	0.71	0.21
46	18.1	230.0	50.35	137.58	0.78	0.40
50	40.7	217.0	135.42	367.09	0.38	0.69
			<u>Default</u>	<u>Calibrated</u>		
MMRE			0.66	0.47		
RMS			151.97	99.41		
RRMS			0.66	0.43		
PRED (0.25)			0.00	0.20		

Validation Subset						
Record #	Size (KLOC)	Actual Effort (PM)	Default Effort (PM)	Calibrated Effort (PM)	MRE (Default)	MRE (Calibrated)
41	21.3	224.0	60.53	165.63	0.73	0.26
42	33.4	335.2	105.71	290.29	0.68	0.13
47	27.4	178.0	84.86	228.69	0.52	0.28
48	37.2	256.0	119.90	328.70	0.53	0.28
49	184.0	1,000.0	974.90	2,669.23	0.03	1.67
			<u>Default</u>	<u>Calibrated</u>		
MMRE			0.50	0.53		
RMS			146.44	748.28		
RRMS			0.37	1.88		
PRED (0.25)			0.20	0.20		

Ada Projects by Contractor R (cont.)

(2ND ITERATION)

Calibration Subset						
Record #	Size (KLOC)	Actual Effort (PM)	Default Effort (PM)	Calibrated Effort (PM)	MRE (Default)	MRE (Calibrated)
42	33.4	335.2	105.71	201.30	0.68	0.40
43	3.3	81.0	7.14	13.84	0.91	0.83
44	53.0	406.0	187.45	356.37	0.54	0.12
46	18.1	230.0	50.35	97.17	0.78	0.58
49	184.0	1,000.0	974.90	1,900.95	0.03	0.90
			<u>Default</u>	<u>Calibrated</u>		
MMRE			0.59	0.57		
RMS			166.61	413.34		
RRMS			0.41	1.01		
PRED (0.25)			0.20	0.20		

Validation Subset						
Record #	Size (KLOC)	Actual Effort (PM)	Default Effort (PM)	Calibrated Effort (PM)	MRE (Default)	MRE (Calibrated)
41	21.3	224.0	60.53	117.18	0.73	0.48
45	22.0	216.6	63.86	121.01	0.71	0.44
47	27.4	178.0	84.86	159.96	0.52	0.10
48	37.2	256.0	119.90	229.52	0.53	0.10
50	40.7	217.0	135.42	257.52	0.38	0.19
			<u>Default</u>	<u>Calibrated</u>		
MMRE			0.57	0.26		
RMS			129.54	68.14		
RRMS			0.59	0.31		
PRED (0.25)			0.00	0.60		

Ada Projects by Contractor R (cont.)

(3RD ITERATION)

Calibration Subset						
Record #	Size (KLOC)	Actual Effort (PM)	Default Effort (PM)	Calibrated Effort (PM)	MRE (Default)	MRE (Calibrated)
45	22.0	216.6	63.86	112.25	0.71	0.48
46	18.1	230.0	50.35	90.12	0.78	0.61
47	27.4	178.0	84.86	147.17	0.52	0.17
48	37.2	256.0	119.90	211.57	0.53	0.17
49	184.0	1,000.0	974.90	1,754.09	0.03	0.75
			<u>Default</u>	<u>Calibrated</u>		
MMRE			0.51	0.44		
RMS			129.18	347.00		
RRMS			0.34	0.92		
PRED (0.25)			0.20	0.40		

Validation Subset						
Record #	Size (KLOC)	Actual Effort (PM)	Default Effort (PM)	Calibrated Effort (PM)	MRE (Default)	MRE (Calibrated)
41	21.3	224.0	60.53	108.54	0.73	0.52
42	33.4	335.2	105.71	186.62	0.68	0.44
43	3.3	81.0	7.14	12.88	0.91	0.84
44	53.0	406.0	187.45	329.85	0.54	0.19
50	40.7	217.0	135.42	237.26	0.38	0.09
			<u>Default</u>	<u>Calibrated</u>		
MMRE			0.65	0.42		
RMS			166.89	96.18		
RRMS			0.66	0.38		
PRED (0.25)			0.00	0.40		

Ada Projects by Contractor R (cont.)

(4TH ITERATION)

Calibration Subset						
Record #	Size (KLOC)	Actual Effort (PM)	Default Effort (PM)	Calibrated Effort (PM)	MRE (Default)	MRE (Calibrated)
41	21.3	224.0	60.53	112.06	0.73	0.50
45	22.0	216.6	63.86	116.24	0.71	0.46
46	18.1	230.0	50.35	93.24	0.78	0.59
49	184.0	1,000.0	974.90	1,815.34	0.03	0.82
50	40.7	217.0	135.42	245.72	0.38	0.13
			<u>Default</u>	<u>Calibrated</u>		
MMRE			0.52	0.50		
RMS			133.87	376.01		
RRMS			0.35	1.00		
PRED (0.25)			0.20	0.20		

Validation Subset						
Record #	Size (KLOC)	Actual Effort (PM)	Default Effort (PM)	Calibrated Effort (PM)	MRE (Default)	MRE (Calibrated)
42	33.4	335.2	105.71	195.59	0.68	0.42
43	3.3	81.0	7.14	13.07	0.91	0.84
44	53.0	406.0	187.45	345.16	0.54	0.15
47	27.4	178.0	84.86	152.20	0.52	0.14
48	37.2	256.0	119.90	221.64	0.53	0.13
			<u>Default</u>	<u>Calibrated</u>		
MMRE			0.64	0.34		
RMS			163.15	77.01		
RRMS			0.65	0.31		
PRED (0.25)			0.00	0.60		

Stratification Category: CMS2/Assembly (90/10) Projects by Contractor M

(1ST ITERATION)

Calibration Subset						
Record #	Size (KLOC)	Actual Effort (PM)	Default Effort (PM)	Calibrated Effort (PM)	MRE (Default)	MRE (Calibrated)
92	16.4	515.3	31.54	312.08	0.94	0.39
94	22.3	604.4	45.24	454.73	0.93	0.25
96	28.2	295.9	60.21	587.76	0.80	0.99
98	41.7	1,063.5	98.41	959.49	0.91	0.10
101	52.4	992.5	127.07	1,250.44	0.87	0.26
			<u>Default</u>	<u>Calibrated</u>		
MMRE			0.89	0.40		
RMS			675.66	212.71		
RRMS			0.97	0.31		
PRED (0.25)			0.00	0.40		

Validation Subset						
Record #	Size (KLOC)	Actual Effort (PM)	Default Effort (PM)	Calibrated Effort (PM)	MRE (Default)	MRE (Calibrated)
93	2.6	147.5	4.16	41.30	0.97	0.72
95	27.7	357.9	58.86	576.16	0.84	0.61
97	3.6	67.6	5.71	56.52	0.92	0.16
99	31.6	927.5	68.17	671.80	0.93	0.28
100	2.9	132.1	4.56	45.20	0.97	0.66
			<u>Default</u>	<u>Calibrated</u>		
MMRE			0.92	0.49		
RMS			416.78	162.46		
RRMS			1.28	0.50		
PRED (0.25)			0.00	0.20		

CMS2/Assembly (90/10) Projects by Contractor M (cont.)

(2ND ITERATION)

Calibration Subset						
Record #	Size (KLOC)	Actual Effort (PM)	Default Effort (PM)	Calibrated Effort (PM)	MRE (Default)	MRE (Calibrated)
92	16.4	515.3	31.54	303.86	0.94	0.41
96	28.2	295.9	60.21	574.52	0.80	0.94
97	3.6	67.6	5.71	55.04	0.92	0.19
98	41.7	1,063.5	98.41	933.70	0.91	0.12
101	52.4	992.5	127.07	1,224.51	0.87	0.23
			<u>Default</u>	<u>Calibrated</u>		
		MMRE	0.89	0.38		
		RMS	628.29	196.56		
		RRMS	1.07	0.33		
		PRED (0.25)	0.00	0.60		

Validation Subset						
Record #	Size (KLOC)	Actual Effort (PM)	Default Effort (PM)	Calibrated Effort (PM)	MRE (Default)	MRE (Calibrated)
93	2.6	147.5	4.16	39.93	0.97	0.73
94	22.3	604.4	45.24	441.60	0.93	0.27
95	27.7	357.9	58.86	561.32	0.84	0.57
99	31.6	927.5	68.17	658.90	0.93	0.29
100	2.9	132.1	4.56	43.63	0.97	0.67
			<u>Default</u>	<u>Calibrated</u>		
		MMRE	0.92	0.51		
		RMS	485.25	178.57		
		RRMS	1.12	0.41		
		PRED (0.25)	0.00	0.00		

CMS2/Assembly (90/10) Projects by Contractor M (cont.)

(3RD ITERATION)

Calibration Subset						
Record #	Size (KLOC)	Actual Effort (PM)	Default Effort (PM)	Calibrated Effort (PM)	MRE (Default)	MRE (Calibrated)
92	16.4	515.3	31.54	514.84	0.94	0.00
94	22.3	604.4	45.24	733.43	0.93	0.21
97	3.6	67.6	5.71	87.77	0.92	0.30
99	31.6	927.5	68.17	1,084.94	0.93	0.17
100	2.9	132.1	4.56	68.88	0.97	0.48
			<u>Default</u>	<u>Calibrated</u>		
MMRE			0.93	0.23		
RMS			510.93	95.75		
RRMS			1.14	0.21		
PRED (0.25)			0.00	0.60		

Validation Subset						
Record #	Size (KLOC)	Actual Effort (PM)	Default Effort (PM)	Calibrated Effort (PM)	MRE (Default)	MRE (Calibrated)
93	2.6	147.5	4.16	63.06	0.97	0.57
95	27.7	357.9	58.86	938.70	0.84	1.62
96	28.2	295.9	60.21	958.15	0.80	2.24
98	41.7	1,063.5	98.41	1,523.84	0.91	0.43
101	52.4	992.5	127.07	1,984.72	0.87	1.00
			<u>Default</u>	<u>Calibrated</u>		
MMRE			0.88	1.17		
RMS			607.60	629.20		
RRMS			1.06	1.10		
PRED (0.25)			0.00	0.00		

CMS2/Assembly (90/10) Projects by Contractor M (cont.)

(4TH ITERATION)

Calibration Subset						
Record #	Size (KLOC)	Actual Effort (PM)	Default Effort (PM)	Calibrated Effort (PM)	MRE (Default)	MRE (Calibrated)
95	27.7	357.9	58.86	583.64	0.84	0.63
97	3.6	67.6	5.71	57.19	0.92	0.15
98	41.7	1,063.5	98.41	976.55	0.91	0.08
99	31.6	927.5	68.17	682.51	0.93	0.26
101	52.4	992.5	127.07	1,268.18	0.87	0.28
			<u>Default</u>	<u>Calibrated</u>		
MMRE			0.89	0.28		
RMS			708.81	197.31		
RRMS			1.04	0.29		
PRED (0.25)			0.00	0.40		

Validation Subset						
Record #	Size (KLOC)	Actual Effort (PM)	Default Effort (PM)	Calibrated Effort (PM)	MRE (Default)	MRE (Calibrated)
92	16.4	515.3	31.54	314.87	0.94	0.39
93	2.6	147.5	4.16	41.48	0.97	0.72
94	22.3	604.4	45.24	459.88	0.93	0.24
96	28.2	295.9	60.21	596.83	0.80	1.02
100	2.9	132.1	4.56	45.35	0.97	0.66
			<u>Default</u>	<u>Calibrated</u>		
MMRE			0.92	0.60		
RMS			357.50	184.60		
RRMS			1.05	0.54		
PRED (0.25)			0.00	0.20		

Stratification Category: Fortran Projects by Contractor A

Calibration Subset						
Record #	Size (KLOC)	Actual Effort (PM)	Default Effort (PM)	Calibrated Effort (PM)	MRE (Default)	MRE (Calibrated)
55	9.8	83.5	18.18	103.82	0.78	0.24
56	8.5	89.0	15.31	87.42	0.83	0.02
57	8.7	63.0	15.72	89.89	0.75	0.43
60	7.7	71.5	13.66	77.85	0.81	0.09
61	5.1	87.0	8.73	49.72	0.90	0.43
64	6.6	60.0	11.45	65.35	0.81	0.09
67	2.8	30.3	4.60	26.19	0.85	0.13
69	2.3	28.3	3.86	22.00	0.86	0.22
			<u>Default</u>	<u>Calibrated</u>		
MMRE			0.82	0.21		
RMS			55.92	18.21		
RRMS			0.87	0.28		
PRED (0.25)			0.00	0.75		

Validation Subset						
Record #	Size (KLOC)	Actual Effort (PM)	Default Effort (PM)	Calibrated Effort (PM)	MRE (Default)	MRE (Calibrated)
58	2.6	26.5	4.24	24.16	0.84	0.09
59	3.1	89.0	5.17	29.40	0.94	0.67
62	11.3	90.5	21.34	122.23	0.76	0.35
63	3.7	53.0	6.27	35.91	0.88	0.32
65	5.4	83.0	9.35	53.18	0.89	0.36
66	27.7	162.0	61.83	351.50	0.62	1.17
68	11.2	110.0	21.12	121.03	0.81	0.10
			<u>Default</u>	<u>Calibrated</u>		
MMRE			0.82	0.44		
RMS			73.53	77.25		
RRMS			0.84	0.88		
PRED (0.25)			0.00	0.29		

Stratification Category: Jovial Projects by Contractor J

Calibration Subset						
Record #	Size (KLOC)	Actual Effort (PM)	Default Effort (PM)	Calibrated Effort (PM)	MRE (Default)	MRE (Calibrated)
21	11.7	18.0	22.25	75.38	0.24	3.19
24	9.1	115.3	16.36	54.37	0.86	0.53
25	13.3	131.5	25.78	87.14	0.80	0.34
26	73.1	385.2	201.30	677.40	0.48	0.76
29	5.4	56.8	9.32	30.61	0.84	0.46
74	18.9	172.3	39.28	135.77	0.77	0.21
75	52.9	430.3	135.59	459.50	0.68	0.07
			<u>Default</u>	<u>Calibrated</u>		
MMRE			0.67	0.79		
RMS			151.94	117.85		
RRMS			0.81	0.63		
PRED (0.25)			0.14	0.29		

Validation Subset						
Record #	Size (KLOC)	Actual Effort (PM)	Default Effort (PM)	Calibrated Effort (PM)	MRE (Default)	MRE (Calibrated)
15	11.3	80.7	21.32	72.17	0.74	0.11
17	79.3	1,241.9	226.27	763.00	0.82	0.39
20	21.8	360.2	46.48	158.25	0.87	0.56
28	5.8	28.0	9.92	32.53	0.65	0.16
30	8.8	94.4	15.74	52.10	0.83	0.45
72	2.7	33.4	4.43	14.59	0.87	0.56
			<u>Default</u>	<u>Calibrated</u>		
MMRE			0.80	0.37		
RMS			436.04	213.06		
RRMS			1.42	0.70		
PRED (0.25)			0.00	0.33		

Glossary

The brief definitions provided below comprise some of the most important terms and concepts pertinent to this research:

Backfiring. A methodology that first establishes a relationship between source code size and function points (or utilizes a previously established relationship), then converts between these two sizing metrics.

Calibration. The adjustment of a model's equations to induce the model to provide a predicted outcome as close as possible to the actual outcome for a given set of data. (Vegas, 1995: 5)

Computer Software Configuration Item (CSCI). An independently managed subset of a large software development project. A CSCI can be considered a "mini-project" in that each CSCI is developed to perform a distinct end-use function. (Ferens, 1997)

Cost estimation. "The art of collecting and scientifically studying costs and related information on current and past activities as a basis for projecting costs as an input to the decision process for a future activity." (Coggins and Russell, 1993)

Function Point. A concept for measuring the functionality of software by computing software size from five attributes: external inputs, external outputs, external inquiries, external interfaces, and internal files. Used primarily for measuring MIS software. (SPR, 1996: 1-3)

Function Point Analysis (FPA). A software sizing measure that quantifies the functions contained within software. From this measure, cost and productivity information can be readily derived. (IFPUG, 1996)

Macro-estimation. The estimating equations are aimed at completed software projects. Once the effort and schedule for the overall project are predicted, the estimate is divided into relative amounts for each phase; synonymous with top-down estimating. (Jones, 1996: 20)

Micro-estimation. Estimating costs at the task or element level for each specific deliverable and then summarizing the task results into the higher level activities, phases, and final project total. (SPR, 1996: 20-13)

Parametric model. A model that uses one or more cost estimating relationships (CERs) or algorithms to estimate costs associated with the development of a software project. The CERs or algorithms are typically based on the project's technical, physical, or other characteristics. (Weber, 1995)

Software metric. The unit of measure for the size of the program or system being developed. Two primary options are Lines of Code (LOC) and Function Points. (SPR, 1996: 20-15)

Source Lines of Code (SLOC). A measure of software volume consisting chiefly of executable program instructions which are delivered in the final product. Excludes blank lines, comments, unmodified vendor supplied operating system or utility software, or other non-developed code. (Rathmann, 1995: 7)

Stratification. Separation of the observations within a database into data sets consisting of homogeneous projects (e.g., similar platform or application, programming language, or program size). This assumes the observations to be stratified contain all necessary information for the calibration process.

Validation. Testing a specific model using known inputs and establishing a comparison between model output and actuals within some specified error range. This is independent and non-iterative with calibration. Also called *cross-validation* in statistics, since the validation procedure uses a portion of the original database purposely excluded from the model development or calibration procedure. (Ourada, 1991: 1.6)

Bibliography

- Boehm, B. W. *Software Engineering Economics*. Englewood Cliffs NJ: Prentice-Hall, Inc., 1981.
- Brooks, F. P. *The Mythical Man-Month*. Reading MA: Addison-Wesley Publishing Company, 1975.
- Brown, N. "Industrial Strength Management Strategies," *Crosstalk: The Journal of Defense Software Engineering*, 9: 7-13, August 1996.
- Christensen, D. S. Class Lecture for AMGT 600. School of Logistics and Acquisition Management, Air Force Institute of Technology (AU), Summer 1996.
- Christensen, D. S., and D. V. Ferens. "Using Earned Value for Performance Measurement on Software Development Projects," *Acquisition Review Quarterly*: 155-171, Spring 1995.
- Coggins, G. A., and R. C. Russell. *Software Cost Estimating: A Comparative Study of What the Models Estimate*. MS Thesis, AFIT/GCA/LAS/93S-4, School of Systems and Logistics, Air Force Institute of Technology (AU), Wright-Patterson AFB OH, September 1993 (AD-A275989).
- Conte, S. D., H. E. Dunsmore, and V. Y. Shen. *Software Engineering Metrics and Models*. Menlo Park CA: The Benjamin/Cummings Publishing Company, Inc., 1986.
- Daly, B. A. *A Comparison of Software Schedule Estimators*. MS Thesis, AFIT/GCA/LSQ/90S-1. School of Systems and Logistics, Air Force Institute of Technology (AU), Wright-Patterson AFB OH, September 1990 (AD-A229532).
- DeMarco, T. Kent State University Lecture on Function Points. WWWeb, <http://www.cis.ksu.edu/~dag/540fall96/materials/cost/sld053.htm> (4 April 1997).
- Department of the Air Force. Guidelines for Successful Acquisition and Management of Software-Intensive Systems: Weapon Systems, Command and Control Systems, Management Information Systems (Version 2.0). Hill AFB UT: Ogden Air Logistics Center, June 1996.
- Dreger, J. B. *Function Point Analysis*. Englewood Cliffs NJ: Prentice Hall, Inc., 1989.
- Ferens, D. V. "New Perspectives in Software Logistics Support," *Logistics Spectrum*: 4-8, Spring 1992.
- Ferens, D. V. "Software Cost Estimation in the DoD Environment," *American Programmer*, 9: 28-34, July 1996.
- Ferens, D. V., and D. S. Christensen. "Software Cost Model Calibration and Validation - An Air Force Case Study," (unpublished). School of Logistics and Acquisition Management, Air Force Institute of Technology, Wright-Patterson AFB OH. 17 pages, 1995.
- Galonsky, J. C. *Calibration of the PRICE S Software Cost Model*. MS Thesis, AFIT/GCA/LAS/95S-1. School of Logistics and Acquisition Management, Air

- Force Institute of Technology (AU), Wright-Patterson AFB OH, September 1995 (AD-A301377).
- Gao, Xiangzhu, and B. Lo. "A Modified Function Method for CAL Systems With Respect to Software Cost Estimation," *Proceedings - 1996 International Conference Software Engineering: Education and Practice*: 212-219, 1996.
- Garmus, D. and D. Herron. *Measuring the Software Process*. Upper Saddle River NJ: Prentice-Hall, Inc., 1996.
- Genuchten, M. and H. Koolen. "On the Use of Software Cost Models," *Information & Management*, 21: 37-44, 1991.
- Gurner, R. B. *A Comparative Study of the Reliability of Function Point Analysis in Software Development Effort Estimation Models*. MS Thesis, AFIT/GCA/LSY/91S-2. School of Systems and Logistics, Air Force Institute of Technology (AU), Wright-Patterson AFB OH, September 1991 (AD-A244179).
- Heemstra, F. J. "Software Cost Estimation," *Information and Software Technology*, 34: 627-639, 1992.
- Henderson, G. S. *The Application of Function Points to Predict Source Lines of Code for Software Development*. MS Thesis, AFIT/GCA/LSY/92S-1. School of Systems and Logistics, Air Force Institute of Technology (AU), Wright-Patterson AFB OH, September 1992 (AD-A258447).
- Illinois Institute of Technology (IIT) Research Institute. *Test Case Study: Estimating the Cost of Ada Software Development*. Lanham MD: IIT, April 1989.
- International Function Points Users' Group. *IFPUG Home Page*. WWWeb, <http://www.bannister.com/ifpug/home/docs/ifpughome.html> (4 April 1997).
- Jones, C. "What Are Function Points?" *Software Productivity Research Home Page*. WWWeb, <http://www.spr.com/library/funcmet.htm#functionpoints> (March 1995).
- Jones, C. "How Software Estimation Tools Work," *American Programmer*, 9: 20-28, July 1996.
- Jones, C. Chairman, Software Productivity Research, Inc. Electronic mail correspondence to Professor Daniel V. Ferens, Air Force Institute of Technology. 5 June 1997.
- Jones, C. Chairman, Software Productivity Research, Inc. Electronic mail correspondence. 11 July 1997.
- Kemerer, C. F. "An Empirical Validation of Software Cost Estimation Models," *Communications of the ACM*, 30(5): 416-429, May 1987.
- Kressin, R. K. *Calibration of SLIM to the Air Force Space and Missile Systems Center Software Database*. MS Thesis, AFIT/GCA/LAS/95S-6. School of Logistics and Acquisition Management, Air Force Institute of Technology (AU), Wright-Patterson AFB OH, September 1995 (AD-A301603).
- Londeix, B. *Cost Estimation for Software Development*. Wokingham, England: Addison-Wesley Publishing Company, 1987.
- Matson, J. E., B. E. Barrett, and J. M. Mellichamp. "Software Development Cost Estimation Using Function Points," *IEEE Transactions on Software Engineering*, 20(4): 275-287, April 1994.
- Mertes, K. R. *Calibration of the CHECKPOINT Model to the Space and Missile Center (SMC) Software Database (SWDB)*. MS Thesis, AFIT/GCA/LAS/96S-11. School

- of Logistics and Acquisition Management, Air Force Institute of Technology (AU), Wright-Patterson AFB OH, September 1996 (AD-A319518).
- Navakla, J. K. "Choosing a Software Cost Estimation Model for Your Organization: A Case Study," *Information & Management*, 18: 255-261, 1990.
- Neter J., M. H. Kutner, C. J. Nachtsheim, and W. Wasserman. *Applied Linear Regression Models* (3rd Ed.). Chicago: Irwin, 1996.
- Newbold, P. *Statistics for Business and Economics* (4th Ed.). Englewood Cliffs NJ: Prentice-Hall, 1995.
- Ourada, Gerald L. *Software Cost Estimating Models: A Calibration, Validation and Comparison*. MS Thesis, AFIT/GSS/LSY/91D-11. School of Systems and Logistics, Air Force Institute of Technology (AU), Wright-Patterson AFB OH, December 1991 (AD-A246677).
- Pinis, M. Vice-President of Technology, Software Productivity Research, Inc. Electronic mail correspondence. 12 July 1997.
- Rathmann, K. D. Calibration of the System Evaluation and Estimation of Resources Software Estimation Model (SEER-SEM) for the Air Force Space and Missile Center (SMC). MS Thesis, AFIT/GCA/LAS/95S-9. School of Logistics and Acquisition Management, Air Force Institute of Technology (AU), Wright-Patterson AFB OH, September 1996 (AD-A300703).
- Skinner, J. M. "Predicting the Cost of Software." Slides from presentation to the Dayton OH Chapter of the Society of Cost Estimating and Analysis (SCEA). 23 April 1997.
- Skinner, J. M. Assistant Professor of Computer Science, Air Force Institute of Technology (AU). Personal Interview. 15 May 1997.
- Software Productivity Research, Inc. *CHECKPOINT for Windows User's Guide*. (Version 2.3.1). Burlington MA, 1996.
- Southwell, S. V. *Calibration of the SoftCost-R Software Cost Model to the Space and Missile Systems Center (SMC) Software Database (SWDB)*. MS Thesis, AFIT/GSM/LAS/96S-6. School of Logistics and Acquisition Management, Air Force Institute of Technology (AU), Wright-Patterson AFB OH, September 1996 (AD-A319050).
- Space Systems Cost Analysis Group (Novak-Ley, G. and S. Stukes, eds.). *Software Methodology Handbook* (Version 1.0). June 1995.
- Subramian, G. H., and G. E. Zarnich. "An Examination of Some Software Development Effort and Productivity Determinants in ICASE Tool Projects," *Journal of Management and Information Systems*, 12(4): 143-160, Spring 1996.
- Symons, C. R. *Software Sizing and Estimating: Mk II FPA*. West Sussex England: John Wiley & Sons Ltd., 1991.
- Thibodeau, R. *An Evaluation of Software Cost Estimating Models*. Huntsville AL: General Research Corporation, 1981.
- Vegas, C. D. *Calibration of Software Architecture Sizing and Estimation Tool*. MS Thesis, AFIT/GCA/LAS/95S-11. School of Logistics and Acquisition Management, Air Force Institute of Technology (AU), Wright-Patterson AFB OH, September 1995 (AD-A301376).
- Weber, B. G. *A Calibration of the REVIC Software Cost Estimating Model*. MS Thesis, AFIT/GCA/LAS/95S-13. School of Logistics and Acquisition Management, Air

Force Institute of Technology (AU), Wright-Patterson AFB OH, September 1995 (AD-A300694).

Wells, P. Software Database Manager, Electronic Systems Center, Air Force Materiel Command, Hanscom AFB MA. Telephone Interview. 29 May 97.