Systems Engineering Cost Estimating:
Approaches and Impacts

By

Belinda Brown

A MASTER OF ENGINEERING REPORT

Submitted to the College of Engineering at

Texas Tech University in

Partial Fulfillment of

The Requirements for the

Degree of

MASTER OF ENGINEERING

Approved

_____

Dr. J. Borrelli

_____

Dr. A. Ertas

_____

Dr. T. Maxwell

_____

Dr. M. Tanik

October 12, 2002

ACKNOWLEDGEMENTS

This report would not have been possible without the support of many people, to whom I am deeply grateful. First, I wish to thank Raytheon for giving me the opportunity to participate in the Texas Tech University Master's program. In particular, I wish to thank my boss Bill Shieldes for believing in me and cheering me on. A personal thanks to the Raytheon Systems Engineering Cost Estimating Team, in particular, Alan Perkowski, Don Larsen, Jim Hinderer, and Bill Kelberlau, without their guidance and direction this report would not have been possible. A special thanks to my classmates for making this program a truly rewarding and enjoyable experience, Sue Armitage for her humor, Julian Parker for putting up with me, and Kurt Himmelreich for his words of encouragement.

A special thanks to the entire Texas Tech staff and guest speakers, they are Masters of their fields and an inspiration to us all. I looked forward to every class with anticipation of what I was going to learn next. I wish to thank Dr. Ertas and Dr. Tanik for their ideas, keeping me focused, and bringing out the best in me.

I would like to thank my family and friends for their support throughout this learning process. Thank you mom and my sisters Pam and Kim and my friend Sheila for believing in me and keeping me grounded. Finally, I wish to thank God because without him this report would not have been possible.

Table of Contents

ACKNOWLEDGEMENTS ..... **II**
ABSTRACT ..... **V**
LIST OF FIGURES ..... **VII**
LIST OF TABLES ..... **VIII**
DISCLAIMER ..... **IX**
CHAPTER-I INTRODUCTION **1**
CHAPTER-II BACKGROUND **4**
2.1 Commercial Models 4
2.1.1 Hardware Models 4
2.1.2 Software Models 6
2.2 Industry Survey 8
2.3 COSYSMO Project 12
CHAPTER-III WHAT IS SYSTEMS ENGINEERING? **13**
3.1 Systems Engineering Defined 13
3.2 Systems Engineering Roles And Responsibilities 14
3.3 Systems Engineering Process 14
CHAPTER-IV COST ESTIMATING APPROACHES **17**
4.1 Introduction 17
4.2 Expert Opinion 18
4.3 Analogy 20
4.4 Parametric 20
4.5 Industrial Engineering 21
4.6 Conclusions 22
CHAPTER-V BUILDING A PARAMETRIC MODEL **24**
5.1 Introduction 24
5.2 Mathematical Modeling Overview 25
5.2.1 Parametric Cost Estimating Overview 26
5.3 Parametric Cost Model Development Process 27
5.3.1 Define Requirements 28
5.3.2 Identify Cost Drivers 30
5.3.3 Establish Data Collection Methodology 35
5.3.4 Develop SECET Data Flow Diagrams 37
5.3.4.1 Context Diagram 38
5.3.4.2 DFD Level-0 39
5.3.4.3 DFD Level-1 39
5.3.5 Assign Parameter Values 41
5.3.6 Develop Model Equations 45
5.3.6.1 Parametric Equation Overview 46
5.3.6.2 Regression Analysis Overview 46
5.3.6.3 Regression Analysis Applied 47

5.3.6.3.1 Equation Coefficients 49
5.3.6.3.2 R^2 Goodness of Fit Measure 50
5.3.6.4 Equations for Products, Components, and Special Products 53
5.3.7 Establish Model Software Structure 53
5.3.8 Calibration/Validation 54
5.3.8.1 Calibration 54
5.3.8.2 Validation 54
5.4 Implementation 55
5.4.1 Prediction And Confidence Factors 55
5.4.2 Monte-Carlo Simulation Results 57
5.5 Conclusions 58
CHAPTER-VI COMPARISON TO OTHER MODELING TECHNIQUES 59
6.1 Introduction 59
6.2 Size Drivers 62
6.2.1 SLOC 62
6.2.2 Function Point Analysis (FPA) 63
6.3 Application to Sizing Systems Engineering Effort? 65
CHAPTER-VII ADDRESSING THE DEATH SPIRAL 68
7.1 Pay Me Now or Pay Me Later 68
7.2 Reliability Case Study 70
7.3 Testability Case Study 72
CHAPTER-VIII SUMMARY AND CONCLUSIONS 74
REFERENCES 77
APPENDIX A ATTRIBUTE DEFINITIONS 80
APPENDIX B SECET WBS OUTPUT EXAMPLE 89

ABSTRACT

There are many motivators for the development of systems engineering cost estimating processes and tools. First, it makes good business sense to have a cost estimating process and tool accessible to systems engineers that is accurate, easy to use, maintain, repeatable and defendable. Too often systems engineers present the project cost to management, find it difficult to defend and find themselves walking out of the meeting with their budget reduced and the scope remained untouched. With this reduction in budget, follow-on contracts suffer due to improper implementation of systems engineering processes during the design and development phase of a program. For example, if reliability processes are severely modified during the design and development phase to meet the reduced systems engineering budget then production problems can arise that were not planned for or budgeted for thus increasing production costs and jeopardizing schedule, not to mention putting customer satisfaction at risk. Also, operating and support costs can increase significantly when systems engineering processes are not adhered to. And finally, the inception of the Capability Maturity Model Integration (CMMI) by the U.S. Department of Defense (DoD), the Software Engineering Institute (SEI) at Carnegie Mellon University, and the National Defense Industrial Association (NDIA) in 1997, is a big motivator. The systems engineering cost estimating process is one small part of CMMI but without this process it could impede an organization's CMMI achievement which in turn can result in not winning programs. In order to reach higher levels of CMMI a systems engineering cost estimating process must be developed and implemented across an organization.

This report addresses cost estimating approaches, in particular, the development process for Raytheon's Systems Engineering Cost Estimating Tool (SECET) is described, providing in detail

the techniques used to derive equations and assess risk. Comparisons are made to commonly used commercial hardware and software cost models and the techniques used to estimate systems engineering effort. Discussion of the "Death Spiral" provides case studies demonstrating the impact to support costs when appropriate funding is not available during the design and development phase of a program to implement sound systems engineering processes.

LIST OF FIGURES

Figure 1. Weapon Systems Budget....2

Figure 2. Seven Stages of Raytheon's IPDP....15

Figure 3. Estimating Techniques During the Acquisition Cycle [Michaels, Wood, 1989]....18

Figure 4. Parametric Cost Model Development Process....27

Figure 5. Systems Engineering WBS Structure....30

Figure 6. SECET's Context Diagram....38

Figure 7. SECET's DFD Level-0....39

Figure 8. DFD Level-1 Calculate System Labor Hours....40

Figure 9. General Equation Form....45

Figure 10. Parametric Curve Example....46

Figure 11. Equation Coefficients....50

Figure 12. System IPDP Level-2 Tasks Derived Equations....52

Figure 13. Monte-Carlo Simulation Results....57

Figure 14. Generic Cost Estimating Model....59

Figure 15. Systems Engineering Opportunity to Influence LCC [Blanchard, 1992]....69

Figure 16. Reliability Growth Test Impact on PBL Cost....71

Figure 17. Testability Impact on Support Cost....73

List of Tables

Table 1. Cost Estimating Approach Summary 23

Table 2. SECET Top Level Requirements 29

Table 3. Systems Engineering Effort "*System*" Cost Drivers 32

Table 4. Systems Engineering Effort "*Product*" Cost Drivers 33

Table 5. Systems Engineering Effort "*Special Products*" Cost Drivers 34

Table 6. System Parameter Values 42

Table 7. System Parameter Values (*Go, No-Go*) 42

Table 8. Product Parameter Values 43

Table 9. Product Parameter Values (*Go, No-Go*) 44

Table 10. Special Products Parameter Values 44

Table 11. Special Products Parameter Values (*Go, No-Go*) 45

Table 12. System Parameter Matrix (historical data) 48

Table 13. Coefficient of Determination (R^2) 51

Table 14. System IPDP Level-2 Tasks Observed Values 52

Table 15. System Parameters Selected by Wideband Delphi 56

Table 16. Parameter Confidence Factors 56

Table 17. SECET WBS Output Example (notional data) 89

DISCLAIMER

The opinions expressed in this report are strictly those of the author and are not necessarily those of Raytheon, Texas Tech University, nor any U.S. Government agency.

CHAPTER-I
INTRODUCTION

Under the Clinton Administration, Department of Defense (DoD) budgets decreased thus decreasing the funds available for acquiring new weapon systems. For this reason, along with fierce competition, contractors were finding ways to reduce design and development and system price to the customer without fully understanding the impacts to the total ownership cost. This lack of understanding is due in part to the lack of adequate cost estimating processes and tools. In his remarks to the Association of the U.S. Army, the Honorable Jacques S. Gansler, Under Secretary of Defense (Acquisition and Technology) stated:

"Unfortunately, we are trapped in a "death spiral." The requirement to maintain our aging equipment is costing us much more each year: in repair costs, down time, and maintenance tempo. But we must keep this equipment in repair to maintain readiness. It drains our resources—resources we should be applying to the modernization of the traditional systems and development and deployment of the new systems. So, we stretch out our replacement schedules to ridiculous lengths and reduce the quantities of the new equipment we purchase—raising their costs and still further delaying modernization." [U.S. Air Force, 2000] With operating and support cost absorbing 80% of the weapon systems budget it leaves very little left to modernize our weapon systems (Figure 1). In order to have more funds available to modernize our weapon systems we need to first understand the cost drivers during the Operating and Support (O&S) phase of the life cycle. Clearly, design decisions made during Concept Exploration (CE) and Engineering and Manufacturing Development (EMD) drive weapon systems' production and O&S costs. Along with the customer's funding profile, these design decisions are driven by adequate funding and implementation of systems engineering processes. To receive adequate

funding for adequate systems engineering process implementation, it is imperative to understand these processes and know how to accurately predict the cost of implementing these processes.

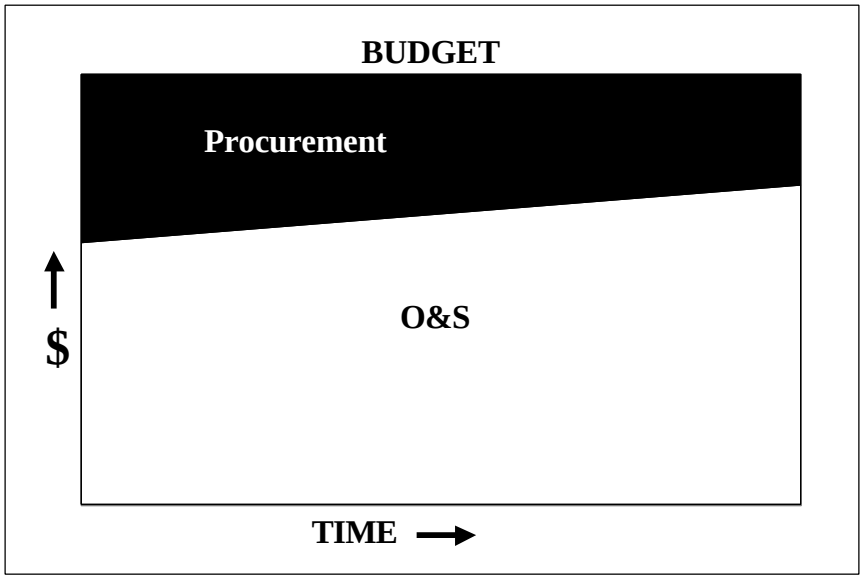

Figure 1. Weapon Systems Budget

Without adequate commercial systems engineering cost estimating processes or tools such as in software cost estimating, many companies are scrambling to develop in-house processes and tools. Today, there exist DoD and Industry Working Groups trying to resolve this issue. Defining what systems engineering is on a program is the first hurdle they must climb. Different companies view systems engineering functions and tasks differently. What is systems engineering responsible for? Does this include reliability, maintainability, and supportability design tasks or does it just include the core systems engineering roles? What are the core systems engineering roles? What are the common cost estimating approaches? Should parametric models be used or bottom-up or both? How are parametric models built? How does the software community predict costs? Should sizing techniques such as Source Lines Of Code

(SLOC) or Function Point Analysis (FPA) be used for systems engineering cost estimating? How does adequate cost estimating and funding during the design and development phase of a program impact other life cycle phases' cost? These questions and others will be addressed in the following chapters. First a little background is given on what other companies are doing in regards to cost estimating, in particular parametric cost estimating. Also, a brief overview of commercial hardware and software cost models is given.

CHAPTER-II
BACKGROUND

This chapter gives a brief overview of a few commonly used commercial hardware and software parametric models and the results of a Raytheon conducted industry survey.

2.1 Commercial Models

2.1.1 Hardware Models

PRICE H – Hardware Model

"PRICE H is used to estimate costs, resources, and schedules for hardware projects such as electronic, electro-mechanical, and structural assemblies. It can be used to estimate hardware projects of any scale, from the smallest individual component to the complex hardware assemblies of a complete aircraft, a ship or a space station." [PRICE, 2002] This model uses a parametric approach to cost estimating based on Cost Estimating Relationships (CERs) that use characteristics that can be quantified, such as weight and size. Cost estimates are generated in three steps: 1) Primary estimate representing a normalized cost is generated from core CERs based on weight and manufacturing complexity, 2) Secondary estimate factors the primary estimate based on complicating factors such as design reuse, specification level, technology maturity, etc., 3) Segregates the estimate into labor, material, and other direct costs based on user's labor rates experience etc.. [DoD, 1999] This model relies on industry data to develop equations and should be calibrated to the user's experience.

SEER-H Model

SEER-H is a hardware estimation, planning, project control, life-cycle cost analysis, and operations and support decision-support tool. [SEER, 2002] "SEER-H uses a combination of

metrics mapping and analytic techniques." [DoD, 1999] Analogic baseline estimates are derived by mapping the system against previous experience (real-world data) and then CERs are used to adjust costs to project specific parameters. Calibration factors are percentage adjustments made to the baseline hours or material costs. A labor allocation table, consisting of factors, allows the labor estimate to be allocated to user defined labor categories or activities. Also, the user can build a custom mapping database. [DoD, 1999]

NAFCOM

"The NASA/Air Force Cost Model (NAFCOM) is an automated parametric cost-estimating tool that uses historical space data to predict the development and production costs of new space programs. It uses parametric relationships to estimate subsystem or component level costs for any aerospace hardware including: earth orbital spacecraft, manned spacecraft, launch vehicle, upper stages, liquid rocket engines, scientific instruments, or planetary spacecraft." [NASA/AF, 2002] There is a Government-use-only version and an unrestricted release version. The model is intended to initially estimate costs at the subsystem or component levels and then rolls these costs up to a project total using system level CERs. Also, NAFCOM allows user defined complexity factors. [DoD, 1999]

ParaModel

"ParaModel, (marketed by Mainstay Software Corporation), provides cost and schedule estimating support for projects that require development, production, modification, integration, or testing of hardware, or software, or both. ParaModel is not database driven, it is a parametric estimating tool." [ParaModel, 2002] It uses many CERs based on data captured and analyzed

from many industries over many years and can be calibrated to a companies experience. Also, staffing requirements are computed for the engineering phase of the development effort. Parameters are linked to one or more other inputs in computing model results. [DoD, 1999]

2.1.2 Software Models

COCOMO

"The original COCOMO model was first published by Dr. Barry Boehm in 1981, and reflected the software development practices of the day. In the ensuing decade and a half, software development techniques changed dramatically. These changes included a move away from mainframe overnight batch processing to desktop-based real-time turnaround; a greatly increased emphasis on reusing existing software and building new systems using off-the-shelf software components; and spending as much effort to design and manage the software development process as was once spent creating the software product." [USC, 2002]

"COCOMO 81 is a model that allows one to estimate the cost, effort, and schedule when planning a new software development activity, according to software development practices that were commonly used in the 1970s through the 1980s." [USC, 2002] It is a regression-based model that considers 63 programs in three modes: embedded, semi-detached, and organic. "Separate equations relating Level Of Effort (LOE) in man-months to program size in thousands of delivered source instructions (KDSI) are established for each mode." [DoD, 1999] The primary input is program size in KDSI along with 15 attributes classified into the categories of product, computer, personnel, and project.

COCOMOII, led by Dr. Boehm, was developed during the mid 1990s by a consortium of organizations to be more compatible with the software design methodologies and practices of 1990s and 2000s. The equations are a revision of COCOMO 81 equations and it includes a new

database consisting of data submitted by the consortium members. There are three stages of estimation. Stage 1, Application Composition, uses object points as its size measurement. Stage 2, Early Design, uses function points or thousands of source lines of code (KSLOC) as the size measure. Stage 3, as in COCOMO 81, uses KSLOC as the primary input with 17 effort multipliers. [DoD, 1999]

PRICE S

PRICE S offers three different analytical instruments for sizing the software. PRICE S Sizing Wizard estimates the number of instructions as source lines of code (SLOC) from three basic inputs: quantitative descriptors, qualitative descriptors, and sizing factors, and delivers this figure to the PRICE S application for developing the overall software cost estimate. "The PRICE S Function Point Sizing Tool assists you in developing a function point count reflecting the size of the software to be estimated. It then translates this count into a corresponding number of source lines of code (SLOC), which is used as an input for the PRICE S application. The PRICE S Predictive Object Point (POP) Sizing Tool lets you determine the magnitude of an object-oriented software project using parameters that relate to object-oriented analysis and design. The POP Sizing Tool, a proprietary instrument developed by PRICE Systems, asks you to describe the software in terms of the weighted methods in classes and the class hierarchy of the design. These inputs are then processed to calculate a project magnitude in Predictive Object Points. This value is then used as the project size input." [PRICE, 2002] The principle inputs are grouped by the following nine categories: Project Magnitude, Program Application, Productivity Factor, Design Inventory, Utilization, Customer Specifications and Reliability Requirements,

Development Environment, Difficulty rating for internal and external integration, and Development Process. [DoD, 1999]

SEER-SEM

"SEER's Software Development Tools (SEER-SEM, SEER-SEM Client & SEER-SSM) are powerful decision-support and process optimization tools that estimate costs, labor, schedules, reliability, and risks associated with information technology, embedded system and commercial software development projects." [SEER, 2002] SEER-SEM inputs are categorized into three groups: Size, Knowledge-Base, and Input Parameters. Size can be SLOC, Function Points, or Proxies that convert to SLOC. Knowledge-Base inputs defaults values into the many input parameters which include: platform, application, acquisition method, development method, personnel capability and experience, product development requirements, and others. "Estimated effort is proportional to size raised to an entropy factor, which is nominally 1.2 (as in embedded COCOMO 81), but can vary based on user selected options. [DoD, 1999]

2.2 Industry Survey

Raytheon conducted an industry wide survey to assess cost estimating approaches with a focus on systems engineering cost estimating. [Kelberlau, 2001] It was concluded from this survey that most companies have experience in using commercial parametric cost estimating tools for hardware production and software development cost predictions and some have developed their own tools for such predictions. Most tools available today project systems engineering effort as a factor of the overall engineering effort and do not project the effort for systems engineering processes and tasks at the system and product levels. These types of models are still in their infancy, data collection, or testing phases. A summary of the survey results is provided below.

Raytheon, Electronic Systems, North Texas

This region uses "bottom-up" as a primary method and PRICE as a secondary method for hardware production cost estimating. SEER and COCMO are used for software development cost estimating. As part of this region's goal of achieving CMMI Level 2 and 3 they assembled a team of senior systems engineers, including the author of this report, to develop a systems engineering cost estimating process and tool. The process is being deployed on programs today, however, the tool is still under development and data is being collected to derive the equations in the tool. Chapter V of this report describes the development process of this tool.

Raytheon, Communications Command and Control (C3I), Garland, Texas

This region uses SEER and PRICE for software and hardware cost predictions respectively. A "bottom-up" approach is used for systems engineering effort estimates. They have recently started a data collection effort for systems engineering labor and are looking at starting metrics.

Raytheon, C3I, Tewksbury, Massachusetts

Compilation of Common Cost Collection Codes (C5) is the primary means of engineering cost estimating. C5 consists of three fundamental pieces: (1) A Standard Work Breakdown Structure (WBS); (2) Historical data extracted from projects and organized in C5 databases in accordance with the WBS; and (3) Methods of bidding tasks organized using the Standard WBS and the historical database.

Raytheon, Portsmouth, New Hampshire

The independent analysis group in Portsmouth uses PRICE and similar tools. This region has developed the N&MIS Engineering Cost Estimating System (NECES), a closed loop system for estimating engineering labor. It is a simplified version of C5 and includes a repository for productivity and cycle time metrics.

Raytheon, Tucson, Arizona

Level of Effort and parametric models are used for cost estimating at this region. Both approaches are supported by the collection of actual data.

Raytheon, El Segundo, California

Over the past five years this site has built a model dedicated to systems engineering estimating. This tool collects and analyzes data based on actual cost for systems engineering processes. Each product area has its own custom data. At this time this model is used primarily for checking bottom-up estimates.

Lockheed Martin Company, Grand Prairie, Texas

This site uses PRICE and tools similar to the C5 approach but with a much enhanced data collection system with data unique to each product area.

Lockheed Martin Company, Fort Worth, Texas

At this site the systems engineering labor estimates are a combination of expert opinion and simple cost estimating relationships based on historical data. PRICE is used on a limited basis.

Lockheed Martin Company, Denver, Colorado

To estimate systems engineering effort they generate the total engineering effort from an extensive data collection system across their product areas and models. From this data they have perform regression analysis to establish CERs that relate systems engineering to the total engineering effort. All of their equations are at a high level and they anticipate breaking it down into process areas as CMMI activities take hold. Also, they employ Delphi techniques to critique their estimates.

Boeing, Company Wide

Although modeling tools and approaches may vary some around the company depending on legacy company influences, cost data collection for supporting proposal and analysis has been part of the Boeing culture since the seventies. Extensive engineering and product data has enabled Boeing to build many models, establish Cost Estimating Relationships (CERs) and perform calibrations. Some parts of the business have used the data to calibrate the PRICE tools but mostly have product area specific models. Their Joint Strike Fighter (JSF) hardware model creates a design engineering estimate then factors the other engineering functions, such as systems engineering. Also, engineering estimates are developed using a bottom-up and engineering judgment approach.

Northrup Grumman, Baltimore, Maryland

Bottoms-up is the primary means of cost estimation with the PRICE tool used for checks on large proposals.

TRW, Los Angeles, California

This site has been collecting data by WBS and job code for all engineering activities for some time. They use this data to generate cost factors for their proposals. The PRICE and SEER models are used to perform cost analysis and to verify proposal inputs.

GDLS, Detroit, Michigan

A WBS is established then senior engineers do similar-to comparisons using historical data to estimate effort. The PRICE tool is used on an irregular basis to check/compare the estimate.

NASA

Over the years NASA has built a parametric model called NASCOM. They also use PRICE and SEER.

GE Jet Engine Division

GE built a very elaborate parametric cost model in the mid 1990s and it is used to estimate all cost categories for all their products.

2.3 COSYSMO Project

The Constructive Systems Engineering Cost Model (COSYSMO) project led by Dr. Barry Boehm, University of Southern California (USC), consists of participants from industry, government and academia. The goal is to build a COnstructive COst MOdel (COCOMO II) like model for estimating effort and duration of systems engineering tasks….to fill the holes not covered by COCOMO II software model. COSYSMO uses "size" and "cost drivers" to generate systems engineering effort and duration. Size drivers include the number of requirements (counted by shalls in system spec), interfaces, Technical Performance Measures (TPMs), operational scenarios and modes, platforms, and algorithms. The initial list of cost drivers includes application factors (requirements and architecture understanding, platform difficulty, etc.) and team factors (stakeholder communities, personnel capability, process maturity, etc.). [Reifer, 2002]

CHAPTER-III
WHAT IS SYSTEMS ENGINEERING?

3.1 Systems Engineering Defined

Before we can project the systems engineering effort on a program we first need to define Systems Engineering. Systems Engineering may mean different things to different people, companies, and industries. Benjamin Blanchard states that "broadly defined, systems engineering is the effective application of scientific and engineering efforts to transform an operational need into a defined system configuration through the top-down iterative process of requirements analysis, functional analysis and allocation synthesis, design optimization, test and evaluation and validation". [Blanchard, 1998]

Raytheon defines Systems Engineering as "the selective application of scientific and engineering principles to:

- transform an operational need into a description of the system configuration which best satisfies the operational need according to the measures of effectiveness
- integrate related technical parameters and ensure compatibility of all physical, functional, and technical program interfaces in a manner which optimizes the total system definition and design
- integrate the efforts of all engineering disciplines and specialties into the total engineering effort" [Raytheon, 2002]

So a systems engineer transforms and integrates? Now that we know what systems engineering is…let's cost what the effort would be to apply the definitions above to a program. Not so fast

you say? First, we need to dig a little deeper and discuss the roles and responsibilities of a systems engineer.

3.2 Systems Engineering Roles And Responsibilities

Put simply, a Systems Engineer (SE):

- is responsible for the technical integrity of the system
- ensures the communication and coordination of requirements, design, and interfaces among the implementing disciplines
- ensures requirements precede design
- ensures structured integration and verification against requirements
- ensures the system is tested for stress, not just for success
- coordinates with the customer to ensure that requirements are clear, concise, and verifiable
- acts as the user's advocate to ensure the system meets the user's needs within the scope of the contract. [Kollman, Norby, 2001]

Can we project the systems engineering effort on a program based on the roles and responsibilities defined above? Probably not. Certainly we can project the effort based on the tasks performed by a systems engineer. Sound familiar? Sounds like a bottom-up methodology. What are these tasks? Doesn't the systems engineering process define the tasks?

3.3 Systems Engineering Process

Raytheon's Integrated Product Development Process (IPDP), which governs the development of new products, is a documented, top-level process that integrates all the activities necessary to

plan and execute each phase of a program throughout a system's entire life cycle. [Raytheon IPDP, 2002] The seven stages of IPDP provide the framework for detailed program planning and execution (Figure 2.). The seven stages are: 1) Business Strategy Planning/Execution, 2) Project Planning, Management and Control, 3) Requirements and Architecture Development, 4) Product Design and Development, 5) System Integration, Verification, and Validation, 6) Production and Deployment, and 7) Operation and Support.

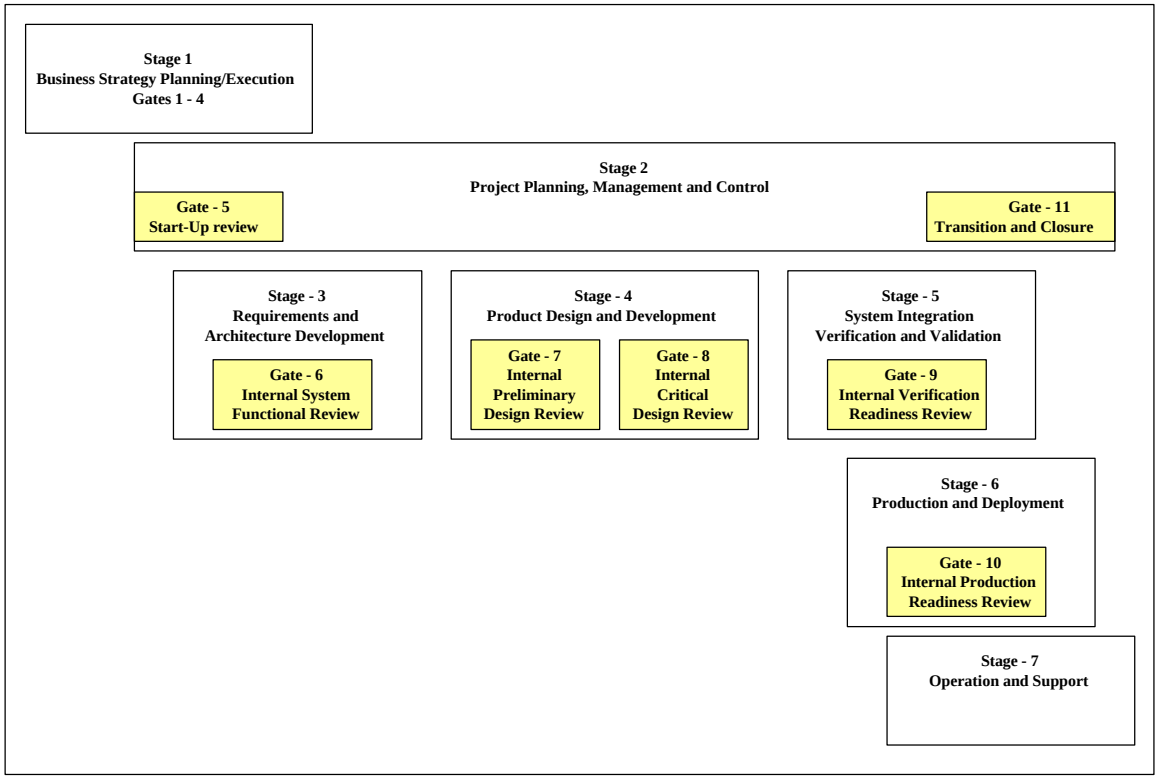

Figure 2. Seven Stages of Raytheon's IPDP

Systems Engineering is involved in all stages of the product development process with varying degrees of effort. As one might expect, systems engineering effort is significantly higher during stages one through five. Clearly, there is systems engineering effort during stages six and seven but if the systems engineering processes are implemented and adhered to during stages one through five then less effort is required during stages six and seven. More discussion on the

impact to stages six and seven can be found in Chapter-VII of this report. What are some of the systems engineering tasks?

Raytheon's Systems Engineering IPDP Level 2 tasks are:

- Systems Engineering Planning & Mgmt
- System Requirements Definition
- System Preliminary Design
- System Integration, Verification & Validation
- Product Requirements Definition
- Product Preliminary Design
- Product Integration, Verification & Validation
- Component Requirements Definition
- Component Preliminary Design Support
- Detail Design Support
- Component Integration and Test

Each Level 2 tasks consist of several levels of tasks with task descriptors. Suppose each Level 2 task contain ten tasks each then there would be 110 tasks. For a bottom-up method one could cost each systems engineering task applicable to their program, however, this would be very time consuming. What if systems engineering effort is charged to the eleven Level 2 tasks listed above and data is collected by these tasks? We could then, along with system and product characteristics, have the information needed to develop a model to project systems engineering effort on a program. Read more about this parametric approach in Chapter-V. The common cost estimating approaches are discussed in the next chapter.

CHAPTER-IV
COST ESTIMATING APPROACHES

4.1 Introduction

Four common approaches to cost estimating are Expert Opinion, Analogy, Parametric, and Industrial Engineering. Estimating approaches vary depending on the program phase and the maturity of the system as shown in Figure 3. [Michaels, Wood, 1989] In most cases during the conceptual phase there is little information available to provide a detailed estimate, so it makes sense to use expert opinion, parametric, or even analogy to estimate cost. Also, more detail does not imply more accuracy. As the saying goes…"measure with a micrometer, mark with chalk, and cut with an axe"…how many times have estimators been in that situation? To reassure ourselves and management, most estimates are prepared using a combination of the four approaches. Also, another methodology used by management is the “Price-To-Win” approach. In this case the estimator provides management with what the project “should cost” using one of the four approaches listed above along with a risk analysis so that management can make an informed decision based on the risk they want to assume and marketing influences. Brief descriptions of the four common approaches and their advantages and disadvantages are discussed within this chapter.

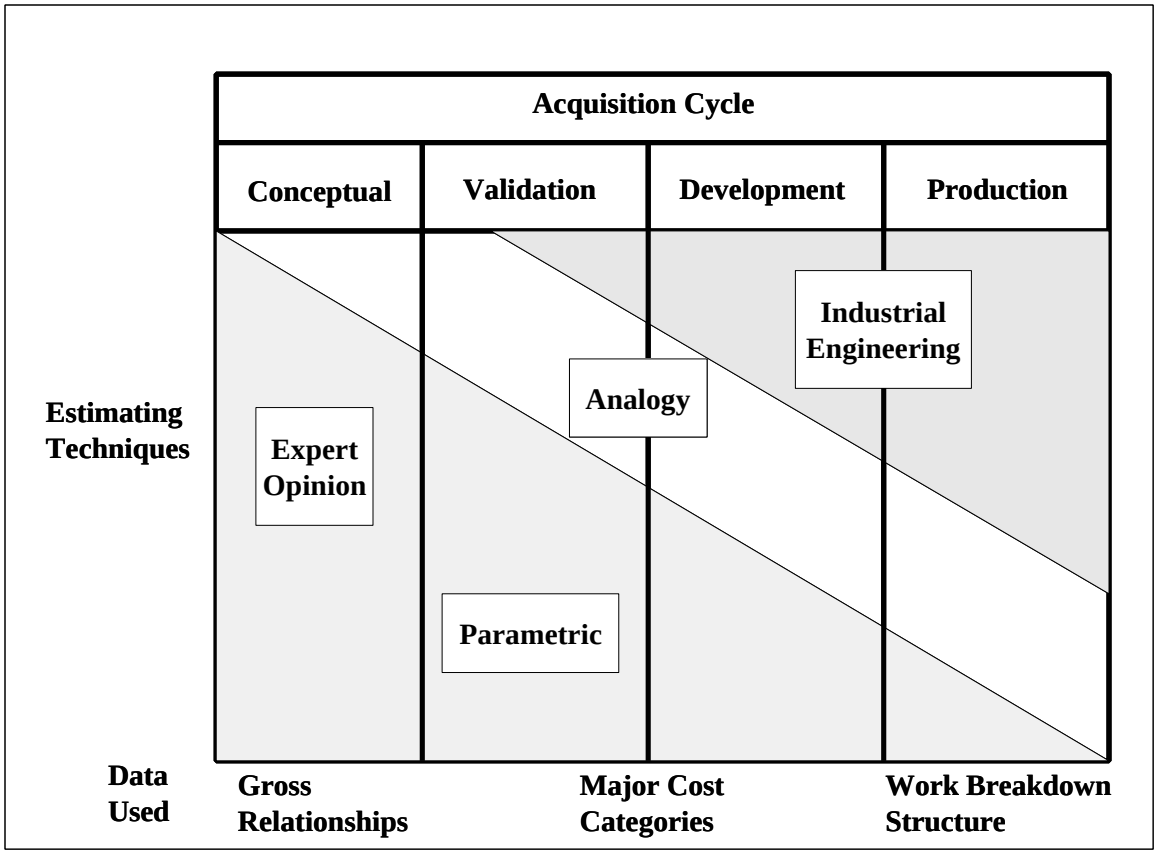

Figure 3. Estimating Techniques During the Acquisition Cycle [Michaels, Wood, 1989]

4.2 Expert Opinion

Expert Opinion or Expert Judgment approach is used by the estimator when data is scarce or nonexistent. As one might expect this approach involves consulting with one or more experts, who with an understanding of the proposed project applies their past experiences to provide the estimator with an estimate.

Advantages

Because of the experiences on previous projects and most new projects are extensions of previous projects, the *Expert* can apply judgment based on technical differences, application, and

complexity to arrive at a reasonable estimate. This approach enables the program to obtain quick estimates when data is scarce and time is short.

Disadvantages

Does the *Expert* have a good memory of the past projects and a good understanding of the future project? Is there a good understanding of the technology and complexity differences? Is the *Expert* overly optimistic, pessimistic, or biased? The approach weighs heavily on the *Expert's* objectivity and expertise. To avoid gross errors in estimating a *Group Consensus* is preferred.

Group Consensus: Wideband Delphi

The Wideband Delphi technique; formulated from the Delphi technique, originated at The Rand Corporation in 1948 as a means of predicting future occurrences; is commonly used in software development cost estimating. [Boehm, 1981] This technique has been highly successful in combining the free discussion of group meetings and anonymous estimation by the participants. The Wideband Delphi technique is listed below.

1. Coordinator presents each expert with a specification and an estimation form.
2. Coordinator calls a group meeting in which the experts discuss estimation issues with the coordinator and each other.
3. Experts fill out forms anonymously.
4. Coordinator prepares and distributes a summary of the estimates.
5. Coordinator calls a group meeting, specifically focusing on having the experts discuss points where their estimates vary widely.

6. Experts fill out forms, again anonymously, and Steps 4 to 6 are iterated for as many rounds as appropriate. [Boehm, 1981]

4.3 Analogy

This approach is similar to the Expert Opinion approach but is based on data. Actual data from a past analogous project is used a baseline and adjusted by experts for technology, application, and complexity differences. Also, applying the Wideband Delphi technique may improve the accuracy of the estimate.

Advantages

Similar to the Expert Opinion approach, analogy estimates are relatively easy and quick to derive. Accuracy may improve since the estimate is based on actual data.

Disadvantages

Data may be easy to obtain but the quality of the data can be questionable. Is the data a good representation or baseline for the future project?

4.4 Parametric

"Parametric cost estimating is a methodology using analytical techniques and historical costs and other program variables such as system physical characteristics or performance characteristics". [Dennedy, 1998] Cost Estimating Relationships (CERs) in the parametric model are typically derived using statistical techniques such as regression analysis. For example, it may be determined that weight and volume are related by a mathematical equation. A detailed example

of the development of a Systems Engineering Cost Estimating parametric model is given in Chapter-V.

Advantages

Once the parametric model has been developed the cycle time to produce an estimate is minimal. CERs are developed from actual data with a focus on cost drivers. They are not influenced by bias, optimism, or a desire to win. One great advantage is that the outcome is repeatable.

Disadvantages

Historical data may not keep up with rapid changing technology causing the outcome to not represent the future accurately. And as in the Expert Opinion and Analogy approaches the parametric model depends on an expert supplying the inputs to the model, which drives the outcome. As the saying goes, "garbage in…garbage out". To overcome this, the Wideband Delphi technique is recommended to derive the inputs to the model.

4.5 Industrial Engineering

The Industrial Engineering approach commonly referred to as “bottom-up” estimating relies on detailed estimates of the lowest levels of a system such as the components of the system. These detailed estimates are derived from supplier quotes, those who will be performing the actual work at the component level, learning curves, etc.. Like the name, one builds the cost estimate from the bottom up and compiles each estimate until you reach the top level, the system level.

Advantages

When detailed information and time is available this approach is the preferred method by most managers because it gives them more accurate costs for each component of the system, making it easier to manage.

Disadvantages

Clearly, this approach is very time consuming and involves many participants. Since this approach is focused on the components of the system many times the system level cost such as integration, specialty engineering, and quality assurance are neglected. Also, detailed information is not always available prior to a proposal submittal.

4.6 Conclusions

Maturity of the system, program phase, data, and time will determine the best cost estimating approach to apply on a program. If data and time is sparse then Expert Opinion is the preferred approach. If data exists on a similar program but time is sparse then the Analogy approach combined with the Expert Opinion approach is the preferred method. If a parametric model exists and time is sparse then maybe the Parametric approach is the way to go. Finally, if there is detailed information available and time is not an issue then maybe the Industrial Engineering approach is the best method. Keep in mind that accuracy may increase as more data becomes available but historical data may not be representative of the future. It is the opinion of the author of this report that the best approach is a combination of the four approaches or at a minimum use two approaches and compare and resolves the differences. Table 1 provides a summary of the advantages and disadvantages of the commonly used four approaches.

APPROACH	ADVANTAGE	DISADVANTAGE
Expert Opinion	Used when data and time is sparse	Relies heavily on objectivity and expertise
Analogy	Used when data is available and time is sparse	Relies on quality data
Parametric	Used when data is available and time is sparse, Objective, Repeatable	Inputs rely on expert opinion and relies on quality data
Industrial Engineering	Used when there is detailed data and time is not an issue	Very time consuming, system level cost may be neglected

Table 1. Cost Estimating Approach Summary

CHAPTER-V
BUILDING A PARAMETRIC MODEL

5.1 Introduction

This chapter gives a brief overview of mathematical modeling, in particular parametric modeling, and provides an example of how to build a Systems Engineering Cost Estimating parametric model. First, what does the government have to say about parametric modeling as a means to develop cost estimates for proposal submittals?

A memorandum from the Office of the Under Secretary of Defense for Directors of Defense Agencies, signed by Eleanor A. Spector, Director, Defense Procurement states: "I fully support the use of properly calibrated and validated parametric cost estimating techniques on proposals submitted to DoD, and I encourage your enthusiastic support. For many procurements, we do not need voluminous bills of material and grass roots engineering estimates of hours which must be audited and updated throughout the course of a lengthy negotiation. Instead, we could rely on parametrics to price early design/development effort, portions of follow-on production buys, or any other effort where verifiable data exists to price parametrically. The cost model, the data used in the model, and the calibration of the model are cost or pricing data required by (Truth in Negotiations Act) TINA." [Spector, 1995]

Virgil Hertling's (Contract Cost/Price Analyst at Headquarters, Air Force Materiel Command, Directorate of Contracting, Pricing and Finance Branch at Wright Patterson Air Force Base, Ohio) paper presented to the International Society of Parametric Analysts and The Society of Cost Estimating and Analysis Joint International Conference in 1998 states, "The Joint

Industry/Government Parametric Cost Estimating Initiative (PCEI) is a key acquisition reform initiative, established to expand the use of parametrics as a primary basis of estimate. Parametric proponents in both industry and government assert that the expanded use of parametric cost estimating techniques on proposal submitted to the government will result in better estimates and more fair and reasonable contract prices, fewer resources to prepare proposals, and reduced contract award cycle time." [Hertling, 1998]

Better estimates, reduced cycle time…it sounds like the government likes it. What is parametric modeling? First let's look at mathematical modeling in general.

5.2 Mathematical Modeling Overview

Modeling is a means in which to describe a system or physical phenomena. Mathematical modeling techniques include: 1) exact and approximate analytical techniques (Ordinary Differential Equations (ODEs) and difference equations, Partial Differential Equations (PDEs), variational principles, stochastic processes); 2) numerical methods (finite differences for ODEs and PDEs, and finite elements); 3) observation models (function fitting, data transforms, network architectures, search techniques, density estimation, filtering and state estimation, liner and nonlinear time series). [Gershenfeld, 1999] This entire report could be devoted to explaining in detail the above techniques but we will focus on a type of observation model…parametric modeling.

Parametric modeling is an observation model in that it forecasts future values of the dependent variable by using historical data to estimate the relationship between the dependent variable and

one or more independent variables. See Section 5.3.6.3 of this Chapter for an example of applying regression analysis to derive the relationships between the variables.

5.2.1 Parametric Cost Estimating Overview

Arlene Minkiewiez, Chief Scientist at Price Systems (cost forecasting/modeling software tools) stated: "Parametric cost estimating is a technique that has been used since the early 1960's by project managers to support cost estimation and decision analysis for products and services. It is an operations research discipline that relies heavily on the collection of historical data and mathematical modeling to develop relationships that predict costs. This method structures a real life situation into a mathematical model, abstracting the essential elements so that a solution relevant to the decisions maker's objectives can be sought.

Originally developed by RCA in the early 1960's, as parametric cost estimating was conceived as an alternative to using bottoms up analysis to estimate the costs of mainly Aerospace and Defense hardware development and production projects. Unlike bottoms up analysis, the parametric model does not require the estimator to complete a thorough decomposition of the product or service being delivered. Instead it applies the values for key cost drivers to cost estimating relationships which then deliver the predicted cost. Although the value of the parametric cost model for a specific type of product or service is that it can be developed only once using data from many organizations, the fact that the model can be calibrated means that it can still deliver very accurate results for many organizations. Parametric cost estimating offers a low cost, time saving solution to the prediction of costs for specific types of products or services." [Minkiewiez, 2002]

So enough talk about what parametric cost estimating is…how is a parametric cost model built?

5.3 Parametric Cost Model Development Process

The following sections describe Raytheon's process for developing the Systems Engineering Cost Estimating Tool (SECET) and the results from implementing the process (Figure 4). Several iterations of each step of the process was conducted, some were conducted in parallel and others sequentially. Notional data will be used throughout for demonstration purposes.

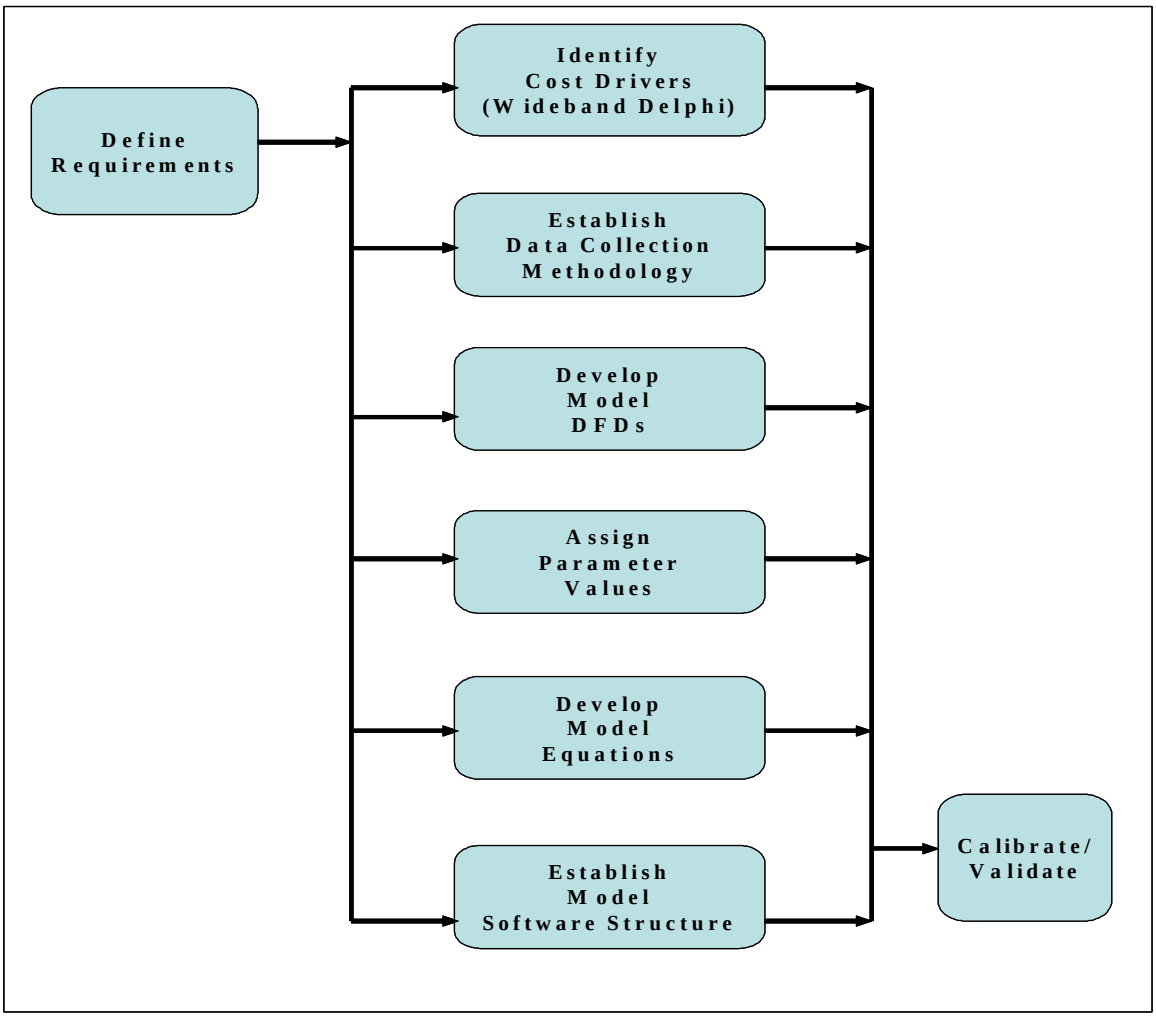

Figure 4. Parametric Cost Model Development Process

5.3.1 Define Requirements

As in the development of any system the requirements need to be well defined up front so that there are no surprises later on. Raytheon assembled a team of senior systems engineers and CMMI experts to develop SECET top level requirements (Table 2). The team decided the output of the model needed to reflect the Systems Engineering Work Breakdown Structure (WBS). This WBS should include the Systems Engineering IPDP Level 2 tasks at the system, product, hardware components, and software components levels; and any special products unique to the program (Figure 5). Also, the team decided to perform a Wideband Delphi to establish the systems engineering cost drivers.

Top Level Requirements		Verification Method			
Ref #	Requirement	Inspect	Analyze/Simulate	Demonstrate	Test
1	SECET shall be easy to use, maintain, repeatable and defendable with the ability to add additional functionality to determine the cost impact to follow-on contracts when systems engineering processes are not implemented during EMD.	X	X	X	X
2	SECET shall be parametric using historical data.		X		X
3	Historical data shall be collected by the categories of RF systems, and EO systems with the option of adding more systems.	X		X	
4	This database shall include the systems and their subsystems, which shall be called products.	X		X	
5	Data shall be collected by the systems engineering labor hours for IPDP level 2 tasks, and special products along with defined system and product attributes.	X		X	
6	Data collected at the system level shall be categorized by; Systems Engineering Planning & Management, System Requirements Definition, System Preliminary Design, and System Integration, Verification and Validation.	X		X	
7	Data collected at the product level shall be categorized by; Product Requirements Definition, Product Preliminary Design, Product Integration, Verification and Validation, Component Requirements Definition, Component Preliminary Design Support, Detail Design Support, and Component Integration and Test Support.	X		X	
8	Data collected at the hardware component level shall be categorized by; Component Requirements Definition, Component Preliminary Design Support, Detail Design Support, and Component Integration and Test Support.	X		X	
9	Data collected at the software component level shall be categorized by; Component Requirements Definition, Component Preliminary Design Support, Detail Design Support, and Component Integration and Test Support.	X		X	
10	Hardware component data collected shall be a roll-up of all hardware components in that product.		X	X	X
11	Software component data collected shall be a roll-up of all hardware components in that product		X	X	X
12	Data collected for Special Products shall be categorized by type and attribute.	X		X	
13	Each system type selected shall have their own screen with IPDP level 2 tasks and system attributes to select from.	X		X	
14	The Product Selection Screen shall give the user a list of products to select from with the ability to toggle from products selected	X		X	
15	Each product selected shall have their own screen with IPDP level 2 tasks at the product and component levels and product attributes to select from.	X		X	
16	Special Products shall be listed on one screen along with their attributes.	X		X	
17	If an IPDP level 2 task is not selected then those hours shall be excluded from the solution.	X			X
18	The WBS defined by the Systems Engineering IPDP level 2 tasks shall be the Labor output from SECET.	X		X	
19	SECET shall be developed to run on a Personal Computer (PC) using commercial languages or software such as Visual Basic 6 (VB6), Microsoft Excel, Microsoft Access, etc..	X			
20	It shall be flexible and dynamic in order to allow the administrator/maintainer to modify data or add additional systems and products as the data becomes available.	X		X	
21	SECET shall be user friendly and provide graphical representation of the labor output.	X		X	
22	SECET shall be developed with the flexibility to add additional functionality to determine the cost impact to follow-on contracts such as production or operating and support when systems engineering IPDP level 2 tasks are not implemented during EMD.	X	X	X	X

Table 2. SECET Top Level Requirements

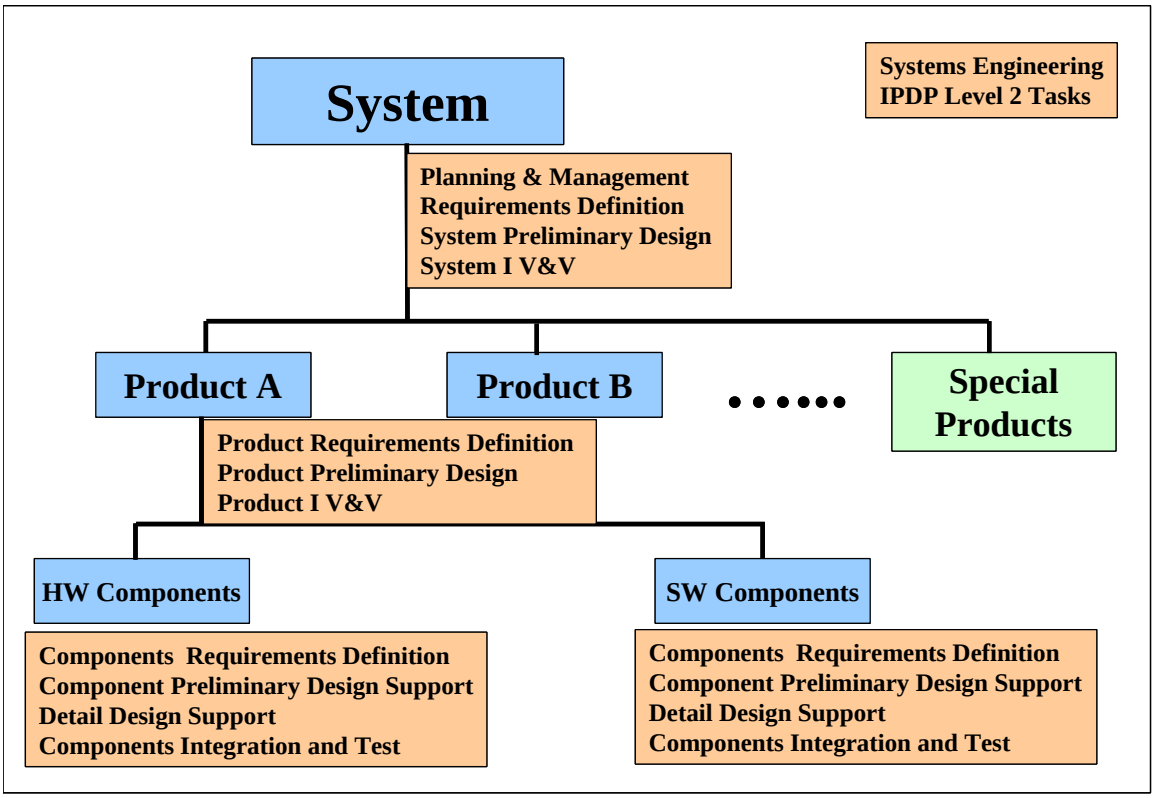

Figure 5. Systems Engineering WBS Structure

5.3.2 Identify Cost Drivers

After the top level requirements were established the team performed a Wideband Delphi to identify the cost drivers. One might expect that in order to identify the cost drivers the team needed to first analyze the data. This is the right approach if quality data existed, however, this was not the case. Program data existed along with the systems engineering labor hours for each program but documented information regarding the complexity (cost drivers) of each program was nonexistent. To overcome this, the team gathered a group of experts from the different business areas to discuss their experiences on what drives the systems engineering cost on programs.

The following gives the steps of the Wideband Delphi process that the team went through.

1. Coordinator called a group meeting in which the experts were given the Systems Engineering Cost Estimating Tool (SECET) requirements, along with discussion of potential drivers of systems engineering effort.
2. Experts developed list of cost drivers anonymously.
3. Coordinator prepared and distributed a summary of the cost drivers.
4. Coordinator called a group meeting focusing on having the experts discuss points where their cost drivers vary widely.
5. Experts revised list of cost drivers, again anonymously, and Steps 3 to 5 were iterated for as many rounds as appropriate.
6. Coordinator called a group meeting to finalize the list of cost drivers.

Tables 3 through 5 give the results of the Wideband Delphi process for identifying the cost drivers of the systems engineering effort on a program. Cost drivers were established for the system, product along with each product's hardware and software components, and special products. Establishing and defining independent cost drivers proved to be very difficult without data to confirm the expert's opinions. All agreed to an explicit definition of each cost driver (Appendix A). [Raytheon CMMI, 2002] Understanding the complexity of this process and the impact of the results, the coordinator reassured the experts that this first list of cost drivers was a baseline and would change to reflect actual data. To that end a data collection methodology was established.

Program Classification
Unclassified
Classified
SAR
Customer Relationship
Indirect Involvement
Direct Involvement
Staff Experience
Extensive Experience
Normal Experience
Inexperienced
Schedule
Compressed
Normal
Stretched
Requirements Definition
Well Defined
Normal
Not Defined
Simulation/Model Environment
Existing Tools
Some Development of Tools
Extensive Dev Required
External Interfaces
Easy
Normal
Complex
System Testing
Formal Qual Test
Field Test
Flight Test

Table 3. Systems Engineering Effort "*System*" Cost Drivers

Product Testing
Prototype Test
Engineering Environmental Test
Component Test
Integrity Test
HALT
RGDT
Requirements Definition
Well Defined
Normal
Not Defined
Design
Simple Mod
Extensive Mod
New Design
Technology Maturity
Existing Technology
Some New Technology
New Technology
Performance Requirements
Easy
Normal
Aggressive
Internal Interfaces
Easy
Normal
Complex
Operational Environment
Benign
Normal
Severe
HW Components
under 10
10 to 20
over 20
SW Components
under 5
5 to 10
over 10
Percent Vender/COTS (components)
68 - 100
31 - 67
0 - 30
WC/Tolerance Analysis Circuits
0-2 Circuits
3-5 Circuits
6 or Greater Circuits
HW Safety/Hazard Analysis
0-1 Hazard Area
2-3 Hazard Areas
4 or more Hazard Areas
SW Safety/Hazard Analysis
No
Yes
RMSS Performance Requirements
Easy
Normal
Aggressive

Table 4. Systems Engineering Effort "*Product*" Cost Drivers

Special Products
FMECA/FMEA
RQT
HF Demonstration/reviews
M/T/S Demo
LORA
LSA
LSAR
Tech Manuals
Training
TRD
FMEA/FMECA (Level)
Signal level
SRU level
LRU level
FMEA/FMECA (functional)
Reliability
Testability
Safety
Tech Manual (type)
Operator
Maintenance
Tech Manual (complexity)
Basic
Detailed
Interactive
Training (type)
Operator
Maintenance
Level of Analysis
LRU only
LRU and SRU
LRU, SRU, SubSRU

Table 5. Systems Engineering Effort *"Special Products"* Cost Drivers

The author of this report struggled with what to call the cost drivers so she read several books and articles which proved to be confusing or inconsistent. Dr. Boehm refers to software cost drivers as attributes; for example, personnel attributes can be personnel capability and experience with the application or computer system and project attributes can be schedule constraints and use of software tools. [Boehm, 1981] William Delaney and Erminia Vaccari state: "The fact that the measurement activity invariably produces a value leads one to infer that the system under

observation “has” a property or *attribute* which is responsible for such a regularity….When a measurement operation yields the same value under all conditions, the variable generates into what we call a (system) *parameter* (the mass of the pendulum bob, the number of departments in the factory, the charge of the electron, etc.)” [Delaney, Vaccari, 1989]. The American Heritage Dictionary defines an attribute as “a quality or characteristic belonging to a person or thing” and a parameter as “a variable or an arbitrary constant appearing in a mathematical expression, each value of which restricts or determines the specific form of the expression”. [The American Heritage Dictionary, 1976] Throughout the literature, parameters are typically thought of as “fixed limits” of the variable. In order to be consistent in this report, from this point forward the cost drivers identified in Tables 3 through 5 will be referred to as attributes and the selections for each attribute will be referred to as parameters (i.e. the cost driver “Staff Experience” is an attribute and Extensive Experience, Normal Experience, and Inexperienced are the parameters of this attribute).

5.3.3 Establish Data Collection Methodology

The data collection methodology is crucial for the accuracy, validity and completeness of SECET. Three categories of data existed; 1) legacy program data, 2) on-going program data, and 3) new program data. Legacy program data represented data from programs that were completed, which existed in various forms. Ongoing program data was similar to legacy program data in that it existed in various forms, however, these programs were not completed. As one might expect new program data represented data from programs that were just awarded or just on contract.

All legacy programs collected cost based on the program WBS. Systems engineering effort was sometimes collected in one WBS element or other times across several WBS elements. Attributes were never collected so the lead systems engineer or key member of the program team was asked to select the parameters of each attribute that best represented their program. Also, they were asked to sub-divide the total systems engineering effort according to the IPDP Level 2 tasks at the system, product, and component levels.

The methodology for collecting data from ongoing programs was similar to that of legacy programs in that the lead systems engineer or key member of the program team was asked to map the systems engineering effort to the IPDP Level 2 tasks and identify the parameters that best represented the attributes. Clearly, this program data would not be used in the algorithm development until the program contract was completed.

Compared to legacy and ongoing programs, new programs were the easiest to get our arms around because we started with a clean slate. These programs were asked to use the systems engineering IPDP Level 2 tasks for the WBS elements, open a charge number for each task, and collect effort and cost for each element along with the attributes and parameters identified in Tables 3 through 5. As with ongoing programs, this data would not be used in algorithm development until the program contract was completed.

Requirements for a database structure were developed consisting of the WBS elements (systems engineering Level 2 tasks) and system, product, hardware components, software components, and special products attributes. This database was to include data categories for system type

(e.g. Radar and Forward Looking Infrared systems (FLIRs)), products (e.g. antenna, transmitter, receiver, power supply), hardware and software components, and special products. Additionally, this database must include a comment field enabling each program to identify any attributes or anomalies not represented in the baseline attribute list to be used in future data analyses and algorithm development.

In conclusion, the data collection methodology consisted of identifying the types of data available, defining how that data would be used, and developing requirements for a database structure. Obviously, the quality of the legacy data was questionable because of the manner in which it was originally collected and then stratified. Because of this and the maturity level of ongoing and new programs data we new it would take some time before quality data was available for algorithm development, however, this data collection methodology set the stage for future success and increased confidence in our systems engineering cost estimating process. Now that we have data, how is this data used to predict systems engineering labor hours?

5.3.4 Develop SECET Data Flow Diagrams

"Data Flow Diagrams (DFDs) have been used for many years prior to the advent of computers. DFDs show the flow of data through a system. The system may be a company, an organization, a set of procedures, a computer hardware system, a software system, or any combination of the proceeding." [Davis, 1993] In our case it was a way to graphically depict the flow of data through our system, SECET, consisting of several layers of diagrams starting with the context diagram then peeling a layer to Level-0 diagram, then Level-1 diagrams, then Level-2 diagrams until the appropriate level was reached. SECET is a system that collects data and uses that data

to develop equations and then based on the user's selections predicts systems engineering labor hours and cost. The following sections provide SECET's DFDs giving a general understanding of the flow of data as well as the processes within SECET.

5.3.4.1 Context Diagram

SECET's context diagram is a system level diagram showing the user inputting system, product, component, and special product selections into the system (Figure 6). These selections are unique to the user's system, the products of the system, the components of the products, and special products of the program. The output is a report showing the total labor hours generated by SECET.

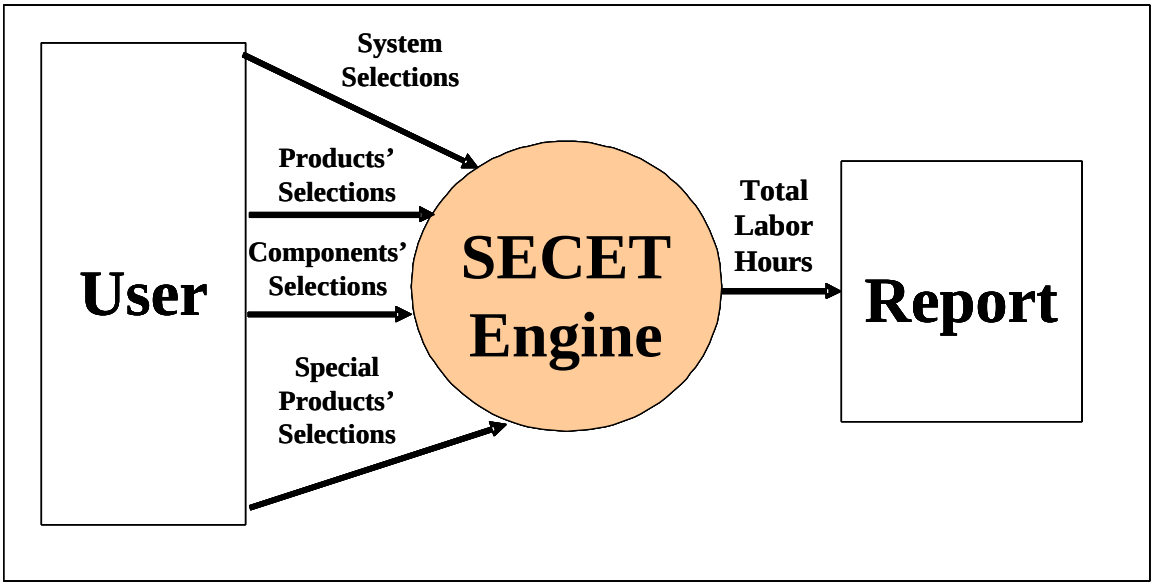

Figure 6. SECET's Context Diagram

5.3.4.2 DFD Level-0

DFD Level-0 demonstrates the data flow from the user to the major processes or sub-functions of SECET; 1. Calculate System Labor Hours, 2. Calculate Products' Labor Hours, 3. Calculate Components' Labor Hours, 4. Calculate Special Products' Labor Hours, and 5. Sum Labor Hours (Figure 7). Each process calculates labors hours and then sends the labor hours to the next process or external entity which is a Report. DFD Level-1. Calculate System Labor Hours is discussed in the next section.

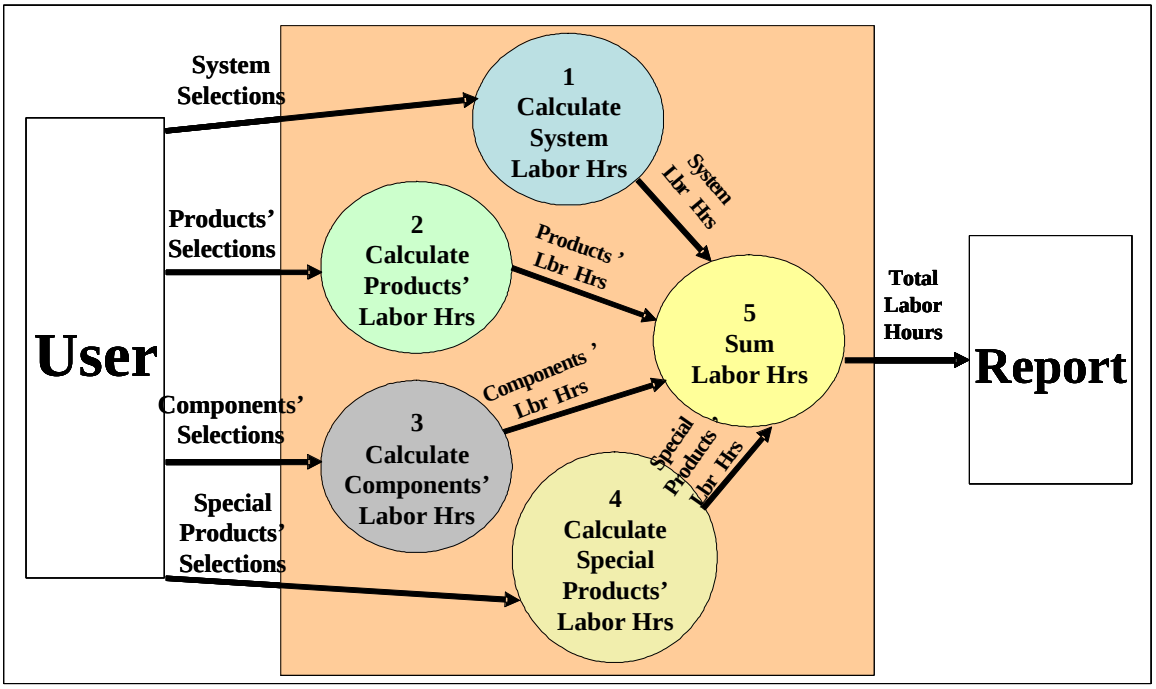

Figure 7. SECET's DFD Level-0

5.3.4.3 DFD Level-1

For simplicity and to avoid redundancy only the Calculate System Labor Hours will be discussed. DFD Level-1, representing the process Calculate System Labor Hours, shows the system parameters selected by the user are sent to the 1.1 Read User's Selections process (Figure

8). The user's selections consist of system type, parameters and IPDP Level 2 tasks appropriate for their program. After the user's selections are read the system parameters are sent to the processes 1.2 Assign Parameters' Values and the selected system type and system IPDP Level-2 tasks are sent to the process 1.4 Insert Values into Equations respectively. After values have been assigned to the selected parameters they are stored in the database for future reference and sent to the process 1.4 Insert Values into Equations for system labor hours calculation. Process 1.3 Develop System Parametric Equations takes historical data from the database and performs regression analysis to derive the system equations for each system IPDP Level-2 task. Then the system equations are sent to process 1.4 Insert Values into Equations for processing. Now that the equations have the information that is needed the output of the equations "Systems Engineering system labor hours for each IPDP Level-2 task" are sent to a report. The next section describes how the values are assigned to the selected parameters.

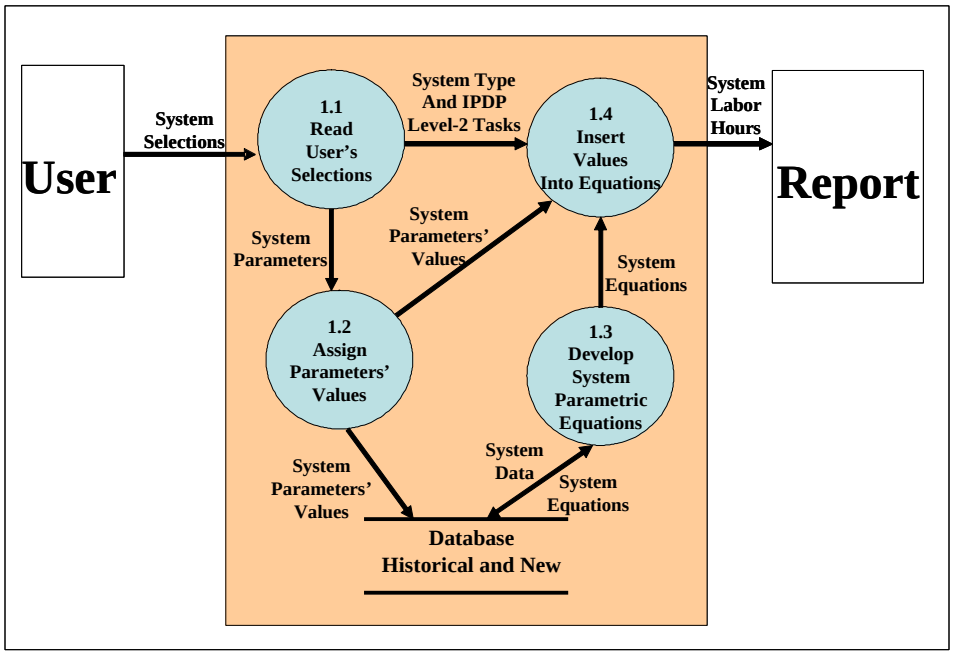

Figure 8. DFD Level-1 Calculate System Labor Hours

5.3.5 Assign Parameter Values

Recall in Section 5.3.2 where a Wideband Delphi was conducted to identify the cost drivers for systems engineering effort on a program, later we called the cost drivers attributes. Each attribute had a number of selections which we called parameters (i.e. the cost driver "Staff Experience" is an attribute and Extensive Experience, Normal Experience, and Inexperienced are the parameters of this attribute). Since these parameters are qualitative some mechanism had to be established to transform these qualitative parameters into quantitative parameters.

Parameter values were assigned by applying the principles of Design of Experiments (DOE). For those of you who are not familiar with DOE then a little background is needed. DOE is a scientific approach which allows the researcher to gain knowledge in order to better understand a process and to determine how the inputs affect the response. [Schmidt, Launsby, 1997] In our case, to really understand the drivers of systems engineering cost, one needs facts and data. We had plenty of data but very little facts about that data. It would take years to collect data through the use of one parameter at a time experimentation or a series of trail and error tests resulting in very inefficient attempts to predict systems engineering costs. In their book, Schmidt and Launsby use standardized units to build powerful models. An example of this standardization process for an experiment involving temperature would be to code the low test setting of temperature as a (-1) and the high test setting as a (+1). [Schmidt, Launsby, 1997]

Using this approach, the attribute parameters were assigned a value. Most attributes consisted of a low, medium and high parameter and received a value of (-1), (0), and (+1) respectively. Other attributes where considered *go* or *no-go*, thus the *go* parameter got a value of (+1) and the *no-go*

parameter got a value of (-1). See Tables 6 through 11 for parameter value assignments. Next, let's develop the equations.

Attribute/Parameters	Value Assignment
Program Classification	
Unclassified	-1
Classified	0
SAR	1
Customer Relationship	
Indirect Involvement	0
Direct Involvement	1
Staff Experience	
Extensive Experience	-1
Normal Experience	0
Inexperienced	1
Schedule	
Compressed	-1
Normal	0
Stretched	-1
Requirements Definition	
Well Defined	-1
Normal	0
Not Defined	1
Simulation/Model Environment	
Existing Tools	-1
Some Development of Tools	0
Extensive Dev Required	1
External Interfaces	
Easy	-1
Normal	0
Complex	1

Table 6. System Parameter Values

Attribute/Parameters	Go	No-Go
System Testing		
Formal Qual Test	1	-1
Field Test	1	-1
Flight Test	1	-1

Table 7. System Parameter Values (*Go, No-Go*)

Attribute/Parameters	Value Assignment
Requirements Definition	
Well Defined	-1
Normal	0
Not Defined	1
Design	
Simple Mod	-1
Extensive Mod	0
New Design	1
Technology Maturity	
Existing Technology	-1
Some New Technology	0
New Technology	1
Performance Requirements	
Easy	-1
Normal	0
Aggressive	1
Internal Interfaces	
Easy	-1
Normal	0
Complex	1
Operational Environment	
Benign	-1
Normal	0
Severe	1
HW Components	
under 10	-1
10 to 20	0
over 20	1
SW Components	
under 5	-1
5 to 10	0
over 10	1
Percent Vender/COTS (components)	
68 - 100	-1
31 - 67	0
0 - 30	1
WC/Tolerance Analysis Circuits	
0-2 Circuits	-1
3-5 Circuits	0
6 or Greater Circuits	1
HW Safety/Hazard Analysis	
0-1 Hazard Area	-1
2-3 Hazard Areas	0
4 or more Hazard Areas	1
RMSS Performance Requirements	
Easy	-1
Normal	0
Aggressive	1

Table 8. Product Parameter Values

Attribute/Parameters	Go	No-Go
Product Testing		
Prototype Test	1	-1
Engineering Environmental Test	1	-1
Component Test	1	-1
Integrity Test	1	-1
HALT	1	-1
RGDT	1	-1
SW Safety/Hazard Analysis		
No		-1
Yes	1	

Table 9. Product Parameter Values (*Go, No-Go*)

Attribute/Parameters	Assignment
FMEA/FMECA (Level)	
Signal level	-1
SRU level	0
LRU level	1
FMEA/FMECA (functional)	
Reliability	-1
Testability	0
Safety	1
Tech Manual (complexity)	
Basic	-1
Detailed	0
Interactive	-1
Level of Analysis	
LRU only	-1
LRU and SRU	0
LRU, SRU, SubSRU	-1

Table 10. Special Products Parameter Values

Attribute/Parameters	Go	No-Go
Special Products		
FMECA/FMEA	1	-1
RQT	1	-1
HF Demonstration/reviews	1	-1
M/T/S Demo	1	-1
LORA	1	-1
LSA	1	-1
LSAR	1	-1
Tech Manuals	1	-1
Training	1	-1
TRD	1	-1
Tech Manual (type)		
Operator	1	-1
Maintenance	1	-1
Training (type)		
Operator	1	-1
Maintenance	1	-1

Table 11. Special Products Parameter Values (*Go, No-Go*)

5.3.6 Develop Model Equations

Regression analysis was performed using historical data to develop parametric equations for each system, product, hardware component, and software component IPDP Level 2 task. As shown in the DFD Level-1, the System Labor Hours is a function of the user selections (type, IPDS Level 2 tasks, and parameters). The general form of the equations developed was of the form shown in Figure 9.

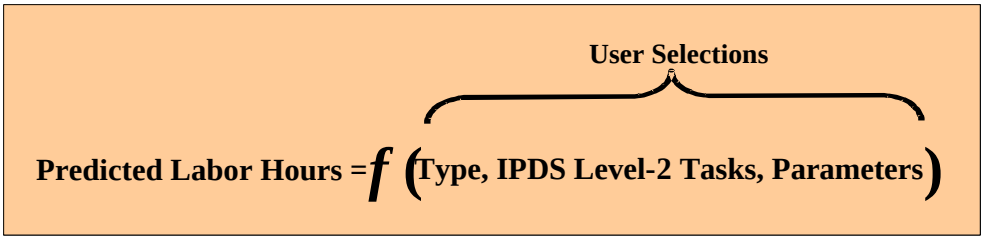

Figure 9. General Equation Form

What is a parametric equation? What is regression analysis? See the next two sections for a brief overview.

5.3.6.1 Parametric Equation Overview

In his book, Richard Hunt explains parametric equations this way. "If x and y are continuous functions of t for t in an interval I, we can describe a curve C in the xy-plane by writing

C: $x = x(t)$, $y = y(t)$, t in I.

The equations $x = x(t)$, $y = y(t)$, are called *parametric equations* and the real variable t is called a *parameter* of the curve. The graph of the *parametric curve* C is the set of all points $(x(t), y(t))$, t in I." [Hunt, 1988] (Figure 10)

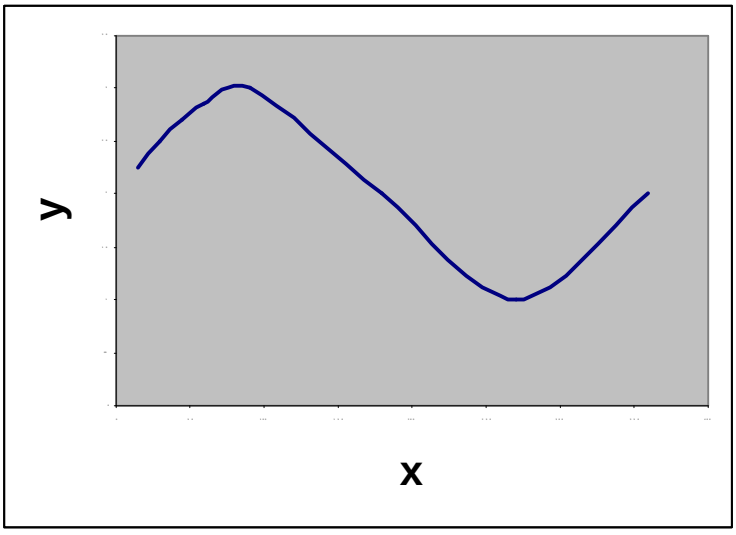

Figure 10. Parametric Curve Example

5.3.6.2 Regression Analysis Overview

Regression analysis is one of the most widely used statistical procedures for developing equations. Regression analysis provides the best-fit curve between two or more independent variables and the dependent variable. The Least Squares method is commonly used to find the best linear relationship to estimate the dependent variable for any measured independent variable. [Ertas, Jones, 1996] The general form of a *simple linear regression* equation is

$$y = a_0 + a_1 x_1.$$

Where *y* is the dependent variable, x_1 is the independent variable, a_0 is a constant, and a_1 is the regression coefficient. In the case where there are *n* number of independent variables the regression analysis is referred to as *Multiple Linear Regression*. The general form of a m*ultiple linear regression* equation with n number of independent variables is

$$y = a_0 + a_1x_1 + a_2x_2 + a_3x_3 + \ldots + a_nx_n.$$

Where *y* is the dependent variable, x_1, x_2, x_3,… x_n are the independent variables, a_0 is a constant, and a_1, a_2, a_3,…a_n are the regression coefficients. Now let's apply regression analysis to our systems engineering cost estimating problem.

5.3.6.3 Regression Analysis Applied

Recall we need to develop equations for each system, product, hardware component, and software component IPDP Level 2 task. Let's start with just one equation at the system level (radar) and the system IPDP Level 2 task, System Planning and Management. The first step in regression analysis is to go to the database and find "radar" data for the IPDP Level 2 task, System Planning and Management, this step can be automated or done manually. Suppose there are twelve radars in the database, all with labor hours for the IPDP Level 2 task System Planning and Management and all have the attribute information needed. Now construct a matrix with each row representing a different radar, each column representing the parameter values and the last column representing the observed (actual) labor hours for each radar's IPDP Level 2 task System Planning and Management (Table 12). Recall from Tables 6 and 7, there are ten system attributes with two or three possible selections (parameters) each.

	Program Classification	Customer Relationship	Staff Experience	Schedule	Requirements Definition	Simulation/Model Environment	External Interfaces	Formal Qual Test	Field Test	Flight Test	observed
Radar	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}	y
1	-1	0	-1	0	-1	-1	-1	-1	-1	-1	500
2	-1	0	-1	0	-1	-1	0	-1	-1	-1	590
3	0	0	-1	0	-1	0	0	-1	-1	-1	680
4	0	0	0	0	-1	0	0	-1	-1	-1	720
5	-1	0	0	0	-1	0	-1	1	-1	-1	750
6	0	0	0	0	-1	1	0	1	-1	1	1000
7	0	0	1	0	0	1	0	1	1	-1	1225
8	0	0	1	1	0	1	0	1	1	-1	1280
9	0	1	1	1	0	1	0	1	1	-1	1310
10	1	1	1	1	0	1	0	1	1	-1	1350
11	1	1	1	1	1	1	0	1	1	-1	1500
12	1	1	1	1	1	1	1	1	1	1	1800

Table 12. System Parameter Matrix (historical data)

Row one of Table 12. represents the parameter values for each attribute and the observed labor hours of System Planning and Management for Radar 1 and so on. Notice, Radar 1 and Radar 12 had 500 hours and 1800 hours for System Planning and Management in the database respectively. Why such a big difference in labor hours for two different radars? We expected Radar 12 to have more hours because of the parameter selections made by the lead systems engineer. Now that we have the data compiled in a matrix, let's apply regression analysis to derive the equation.

Remember we want to derive an equation of the form $y = a_0 + a_1x_1 + a_2x_2 + a_3x_3 + \ldots a_{10}x_{10}$. Where y is the dependent variable (labor hours), $x_1, x_2, x_3, \ldots x_{10}$ are the independent variables, a_0 is a constant, and $a_1, a_2, a_3, \ldots a_{10}$ are the regression coefficients. To derive this equation we want to solve for $a_0, a_1, a_2, a_3, \ldots a_{10}$. How do we take our data from Table 12 and derive an equation similar to the one above with real numbers for $a_0, a_1, a_2, a_3, \ldots a_{10}$? Let's rewrite Table 12 into an equation using the parameter values as matrix A, the constant and coefficients as matrix x and the actual labor hours as matrix B. In order to calculate an a_0, a new first column consisting of all ones is included in the parameter matrix (A).

$$\mathbf{A} = \begin{pmatrix} 1 & -1 & 0 & -1 & 0 & -1 & -1 & -1 & -1 & -1 & -1 \\ 1 & -1 & 0 & -1 & 0 & -1 & -1 & 0 & -1 & -1 & -1 \\ 1 & 0 & 0 & -1 & 0 & -1 & 0 & 0 & -1 & -1 & -1 \\ 1 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & -1 & -1 & -1 \\ 1 & -1 & 0 & 0 & 0 & -1 & 0 & -1 & 1 & -1 & -1 \\ 1 & 0 & 0 & 0 & 0 & -1 & 1 & 0 & 1 & -1 & 1 \\ 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 1 & -1 \\ 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & -1 \\ 1 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & -1 \\ 1 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & -1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & -1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{pmatrix} \quad \mathbf{x} = \begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \\ a_4 \\ a_5 \\ a_6 \\ a_7 \\ a_8 \\ a_9 \\ a_{10} \end{pmatrix} \quad \mathbf{B} = \begin{pmatrix} 500 \\ 590 \\ 680 \\ 720 \\ 750 \\ 1000 \\ 1225 \\ 1280 \\ 1310 \\ 1350 \\ 1500 \\ 1800 \end{pmatrix}$$

Isn't the following equation true?

$$\underset{\mathbf{A}}{\begin{pmatrix} 1 & -1 & 0 & -1 & 0 & -1 & -1 & -1 & -1 & -1 & -1 \\ 1 & -1 & 0 & -1 & 0 & -1 & -1 & 0 & -1 & -1 & -1 \\ 1 & 0 & 0 & -1 & 0 & -1 & 0 & 0 & -1 & -1 & -1 \\ 1 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & -1 & -1 & -1 \\ 1 & -1 & 0 & 0 & 0 & -1 & 0 & -1 & 1 & -1 & -1 \\ 1 & 0 & 0 & 0 & 0 & -1 & 1 & 0 & 1 & -1 & 1 \\ 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 1 & -1 \\ 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & -1 \\ 1 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & -1 \\ 1 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & -1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & -1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{pmatrix}} * \underset{\mathbf{x}}{\begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \\ a_4 \\ a_5 \\ a_6 \\ a_7 \\ a_8 \\ a_9 \\ a_{10} \end{pmatrix}} = \underset{\mathbf{B}}{\begin{pmatrix} 500 \\ 590 \\ 680 \\ 720 \\ 750 \\ 1000 \\ 1225 \\ 1280 \\ 1310 \\ 1350 \\ 1500 \\ 1800 \end{pmatrix}}$$

Doesn't this look familiar? **A * x = B**

Isn't this just a system of linear equations? Now, let's solve this system of linear equations to derive the constant and coefficients.

5.3.6.3.1 Equation Coefficients

Since we know A and B let's rewrite the equation as x = B / A. The software MATLAB is a powerful tool to find the "best-fit" solution, a vector representing the coefficients of the equation:

$y = a_0 + a_1x_1 + a_2x_2 + a_3x_3 + a_4x_4 + a_5x_5 + a_6x_6 + a_7x_7 + a_8x_8 + a_9x_9 + a_{10}x_{10}$.

Figure 11 gives the coefficients and equation derived from the equation "x = B / A". The largest coefficients are a_5 and a_7 indicating that the attributes "Requirements Definition" and "External

Interfaces" are the drivers of systems engineering effort for the IPDP Level 2 task, Systems Engineering Planning and Management.

Coefficients derived from regression analysis										
a_0	a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8	a_9	a_{10}
1095.8	40.0	30.0	63.3	55.0	173.3	3.3	113.3	80.0	52.5	70.0
Plug coefficients into the equation										
Lbr Hrs = $a_0 + a_1x_1 + a_2x_2 + a_3x_3 + a_4x_4 + a_5x_5 + a_6x_6 + a_7x_7 + a_8x_8 + a_9x_9 + a_{10}x_{10}$										
Solution Equation										
Lbr Hrs = $1095.8 + 40.0x_1 + 30.0x_2 + 63.3x_3 + 55.0x_4 + 173.3x_5 + 3.3x_6 + 113.3x_7 + 80.0x_8 + 52.5x_9 + 70.0x_{10}$										

Figure 11. Equation Coefficients

5.3.6.3.2 R^2 Goodness of Fit Measure

Now that we have derived an equation to predict labor hours we need to determine the "Goodness of Fit" or strength of the model. The Coefficient of Determination (R^2) represents the strength of the model and is calculated by the equation:

$R^2 = 1 – (SSResid ÷ SSTo)$. [Devore, Peck, 1986]

Where,

SSResid = ? (y – y)2 = Residual Sum of Squares

SSTo = ? (y –?)2 = Total Sum of Squares

y = Observed Labor Hours

y = Predicted Labor Hours

? = Mean of Observed Labor Hours

In the first column of Table 13 are the observed (actual) labor hours. Column two gives the predicted values using the equation from the regression analysis. The difference (residuals) between the observed values and the predicted values are found in column three with the squares

of the differences in column four. Column five gives the squares of the differences between the observed values and the mean of the observed values. The totals at the bottom of columns four and five give the sum of the respective squares. As shown in Table 13 the coefficient of determination was calculated to be (R^2) = 0.998, indicating the strength of the model with 1.0 being the strongest.

	1	2	3	4	5
	observed	predicted values	residuals		
	y	$\hat{y}$	y - $\hat{y}$	(y - $\hat{y}$)2	(y -?)2
	500	500	-0.1	0.0	312201.6
	590	613	-23.4	547.6	219726.6
	680	657	23.3	542.9	143451.6
	720	720	0.0	0.0	114751.6
	750	727	23.3	542.9	95326.6
	1000	1023	-23.3	542.9	3451.6
	1225	1225	0.1	0.0	27639.1
	1280	1280	0.1	0.0	48951.6
	1310	1310	0.1	0.0	63126.6
	1350	1350	0.1	0.0	84826.6
	1500	1523	-23.2	538.2	194701.6
	1800	1777	23.5	552.3	549451.6
sum =	12705	12705	0.5	**3266.8**	**1857606.3**
? =	1059	1059			

SSResid =Σ (y -$\hat{y}$)^2 **SSTo =Σ (y -?)^2**

R^2 = 1-(SSResid/SSTo) = 0.998

Table 13. Coefficient of Determination (R^2)

Now that we have the equation for the Systems Engineering labor hours for the System Level IPDP Level 2 task Systems Engineering Planning and Management, let's derive the equations for all System Level IPDP Level 2 tasks using regression analysis. Remember the same system attributes and parameters apply for all system IPDP Level 2 tasks. Table 14 gives the observed values for all IPDP Level 2 tasks for each radar and Figure 12 is a summary of all system level

equations along with their associated coefficient of determination (R^2). All IPDP Level 2 tasks' coefficients indicate that the attributes "Requirements Definition" and "External Interfaces" are the drivers of Systems Engineering effort, however, the data is notional and may not represent actual cost drivers.

	Observed Values			
Radar	SE Pln & MGMT	System Req Def	System Prelm Dsgn	System IV&V
1	500	800	1100	1800
2	590	1180	1600	2667
3	680	1360	2200	3667
4	720	1440	2335	3892
5	750	1500	2432	4053
6	1000	2000	3600	6000
7	1225	2800	4900	8250
8	1280	2910	5200	9000
9	1310	2970	5248	9080
10	1350	3050	5378	9297
11	1500	3350	5864	10307
12	1800	3850	7000	12000
sum =	12705	27210	46857	80012
? =	1059	2268	3905	6668

Table 14. System IPDP Level-2 Tasks Observed Values

Systems Engineering Planning and Management Equation $R^2 = 0.998$
$\text{Lbr Hrs} = 1095.8 + 40.0x_1 + 30.0x_2 + 63.3x_3 + 55.0x_4 + 173.3x_5 + 3.3x_6 + 113.3x_7 + 80.0\,x_8 + 52.5x_9 + 70.0x_{10}$
System Requirements Definition Equation $R^2 = 0.999$
$\text{Lbr Hrs} = 2225.0 + 80.0x_1 + 60.0x_2 + 110.0x_3 + 110.0x_4 + 330.0x_5 + 40.0x_6 + 410.0x_7 + 260.0\,x_8 + 180.0x_9 + 15.0x_{10}$
System Preliminary Design Equation $R^2 = 0.999$
$\text{Lbr Hrs} = 3828.7 + 130.0x_1 + 48.0x_2 + 229.7x_3 + 300.0x_4 + 580.7x_5 + 280.7x_6 + 594.7x_7 + 363.5x_8 + 373.5x_9 + 176.0x_{10}$
System IV&V Equation $R^2 = 0.999$
$\text{Lbr Hrs} = 6422.8 + 217.0x_1 + 80.0x_2 + 349.3x_3 + 750.0x_4 + 1134.3x_5 + 534.3x_6 + 991.3x_7 + 622.5\,x_8 + 547.5x_9 + 226.5x_{10}$

Figure 12. System IPDP Level-2 Tasks Derived Equations

5.3.6.4 Equations for Products, Components, and Special Products

Recall, the three Product IPDP Level 2 tasks from Figure 5 were: 1) Product Requirements Definition, 2) Product Preliminary Design, and 3) Product IV&V. Hardware and software components were a rollup of all hardware and software components respectively and had the following four IPDP Level 2 tasks each: 1) Components Requirements Definition, 2) Components Preliminary Design, 3) Detail Design Support, and 4) Components Integration and Test. Since Special Products did not have IPDP Level 2 tasks, each special product had a unique equation derived from the special products' attributes and historical data. To avoid redundancy, the development of these equations will not be discussed in this report, however, the same development process was used to derive and validate the equations. In the end, four System level, three Product level, four Hardware Component level, four Software Component level, and ten Special Products' equations were developed for a total of twenty-five equations.

5.3.7 Establish Model Software Structure

During the development process, a software structure was established to meet the requirements listed on Table 2. A prototype model was developed in Excel using the IPDP Level 2 tasks, special products, attributes, parameter selections, and equations to give the software engineer an example of the functionality of the parametric model. This prototype was converted by the software engineer to a more dynamic software package to allow for ease of use, flexibility, linkage to databases, and additional functionality.

5.3.8 Calibration/Validation

Recall Eleanor Spector's comment, "I fully support the use of properly calibrated and validated parametric cost estimating techniques…'. What does it mean to calibrate and validate a parametric model?

5.3.8.1 Calibration

According to the "Joint Industry/Government Parametric Estimating Handbook", Calibration is defined as "a process of adjusting a commercial parametric model to a specific contractor's historical cost experience and business culture. Calibration is accomplished by calculating adjustment factor(s) to compensate for differences between an organization's historical costs and the costs predicted by the cost model that are based on default values." [DoD, 1999] Since our parametric model (coefficients of equations) was developed using our own historical data then by definition it is calibrated.

5.3.8.2 Validation

Model validation is a continuous process used to ensure that the model is a good predictor. Validation can be accomplished by several means such as, 1) comparing model results with results from another model or method, 2) using historical program data used in the building of the model to determine if the output of the model predicts the costs incurred by that program (i.e. comparing the observed values with the predicted values), and 3) historical program data can be excluded from the building process and then used to determine if the model is a good predictor but most companies don't have enough program data to start with, so this may not be a good indicator. Finally, validation of the model can include comparing model results with that of new programs but this would involve waiting for months or even years before the programs are

completed and the data used to validate. Which ever validation method is used the validation process is continuous in that the model should be updated periodically with current data and the validation process repeated. [DoD, 1999]

SECET was validated by comparing actual labor hours with predicted labor hours using the derived equations. Table 13 shows the actual and predicted labor hours along with the computed R^2 (0.998) for the Systems Engineering Planning and Management equation indicating this equation was a good predictor of labor hours.

5.4 Implementation

So far we have derived equations from historical data to predict Systems Engineering labor hours as well as calculated the strength of the model. Now let's use our parametric model to predict labor hours for a new program. Since we have a high confidence in the strength of the model then we should have high confidence in our prediction for a new program. Not so fast. Having a high confidence in the model as a good predictor is not enough. What about the confidence in the new program's parameter selections for each attribute? Certainly, parameter selections play a big part in accurately predicting the systems engineering labor for a new program. How do we assure ourselves we have selected the right parameters?

5.4.1 Prediction And Confidence Factors

To avoid risk in our prediction, a Wideband Delphi was conducted to assign parameters for each attribute of our new system. During this process, confidence factors were assigned to these parameters to determine our risk in our prediction based on the uncertainty in the parameter selections. Nominal, minimum, and maximum values were determined for each attribute to

calculate the nominal, minimum, and maximum systems engineering labors hours which were calculated to be 15,387 hours, 9,159 hours, and 16,997 hours respectively (Tables 15 & 16). The nominal value of 15,387 hours was our expected value. (See Appendix B for an example SECET WBS output with the associated predicted labor hours.) Confidence factors were assigned to each parameter based on expert opinion of meeting the projected parameter (Table 16). For example, the participants of the Wideband Delphi selected the parameter "Normal" for the attribute, Requirements Definition, and agreed they were 90% confident in their selection with a 10% probability that this attribute would have the parameter "Well Defined". After each parameter was assigned a confidence factor, then a Monte-Carlo simulation, using the software application Crystal Ball, was performed to assess the risk of our nominal (predicted) labor hours based on the confidence in the parameter selections.

System Parameters Selected by Wideband Delphi											Total SE Effort at System Level Predicted Values		
	Program Classification	Customer Relationship	Staff Experience	Schedule	Requirements Definition	Simulation/Model Environment	External Interfaces	Formal Qual Test	Field Test	Flight Test	Nominal	Min	Max
	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}	y	y	y
New program	0	1	-1	1	0	-1	0	1	1	-1	15387	9159	16997

Table 15. System Parameters Selected by Wideband Delphi

	Parameter Value			Confidence		
	nominal	Min	Max	Nominal	Min	Max
x1	0	-1	0	0.90	0.10	0.90
x2	1	0	1	0.80	0.20	0.80
x3	-1	-1	0	0.90	0.90	0.10
x4	1	0	1	0.90	0.10	0.90
x5	0	-1	0	0.90	0.10	0.90
x6	-1	-1	0	0.75	0.75	0.25
x7	0	-1	0	0.80	0.20	0.80
x8	1	1	1	1.00	0.00	0.00
x9	1	1	1	1.00	0.00	0.00
x10	-1	-1	-1	1.00	0.00	0.00

Table 16. Parameter Confidence Factors

5.4.2 Monte-Carlo Simulation Results

Discrete distributions were applied to the confidence factors in the simulation to assess the risk of the predicted value of 15,387 hours. From this simulation, conducting 10,000 trials, it was determined that 21.7% of the output values were greater than the predicted value of 15,387 hours, giving a 21.7% risk of the actual labor hours being greater than the predicted hours. This gives a 78.3% probability that the actual hours will be less than the predicted hours. The mean value and standard deviation of the simulation was calculated to be 14,830 hours and 1,224 hours respectively. Figure 13 gives the reverse cumulative chart and statistical results of the Monte-Carlo simulation. Because of the parameter selections and the associated discrete distributions there is a big spike in the graph close to the nominal point of 15,387 increasing the risk to over 50%. Iterations of the Wideband Delphi may changes this effect, however, if this is the final outcome then management needs to be made aware of this spike in risk and may elect to increase the hours to mitigate this risk.

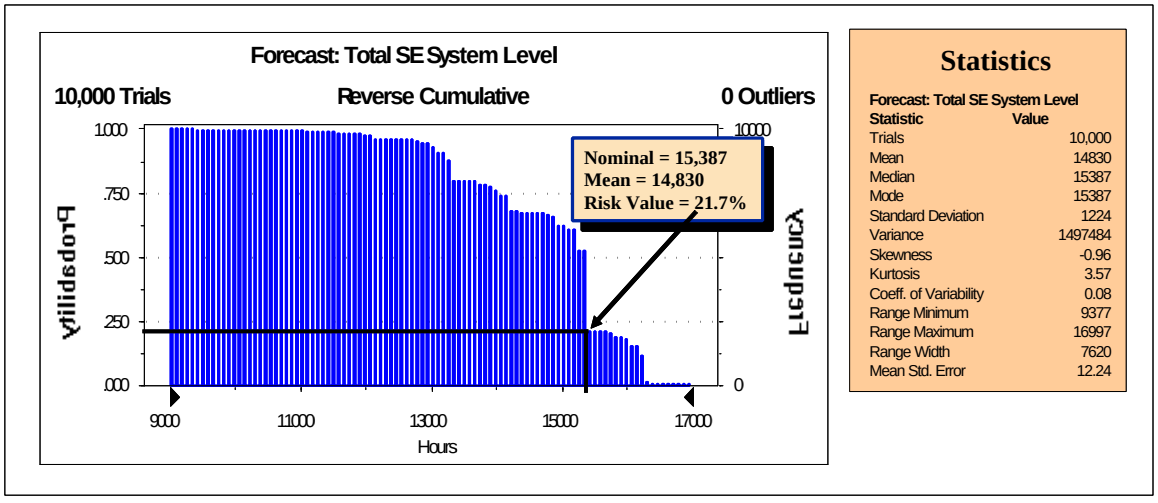

Figure 13. Monte-Carlo Simulation Results

5.5 Conclusions

The government encourages the use of properly calibrated and validated parametric models because they produce better estimates and reduce the cycle time to produce the estimates. Building a parametric model will take upfront time and resources, however, once it is developed it can be a good predictor of costs as long as the database is updated with current data and the model is validated periodically.

The process steps to building Raytheon's parametric Systems Engineering Cost Estimating Tool (SECET) included 1) Defining the requirements, 2) Identifying the cost drivers, 3) Establishing a data collection methodology, 4) Developing the DFDs, 5) Assigning parameter values, 6) Developing the model equations using regression analysis, 7) Establishing the model software structure, and 8) Validating the model. Cost was collected and predicted at the system, product, and component IPDP level-2 tasks and for special products. Identified cost drivers became the attributes for the system, product, component, and special products. Attributes consisted of parameters that were assigned values of (-1), (0), and (1). Some attributes had three parameters while others were determined to be "go" or "no-go" and received a (1) and (-1) respectively. Regression analysis was used to develop the model equations. Validation included comparing the observed values with the predicted values and R^2 Goodness of Fit measure.

Even though the computed R^2 indicated that SECET was a good predictor of labor hours there was concern about the confidence in the parameter selections for new programs. A Wideband Delphi was conducted to assign the parameters and associated confidence factors and a Monte-Carlo simulation was performed to assess and mitigate the risk of the prediction.

CHAPTER-VI
COMPARISON TO OTHER MODELING TECHNIQUES

6.1 Introduction

Chapter-II of this report discussed a few commonly used commercial hardware and software parametric models and Chapter-V of this report gave an example of building a parametric model by applying regression analysis to develop equations for systems engineering cost estimating. This chapter will briefly compare the techniques used to develop the equations in the common commercial parametric models to the techniques used in developing SECET.

As in any cost estimating model, there are *inputs* made by the user reflecting the "system" they are estimating and there is a model *engine* containing equations developed from using various techniques to calculate the *outputs* such as effort, cost, schedule, risk, etc. (Figure 14).

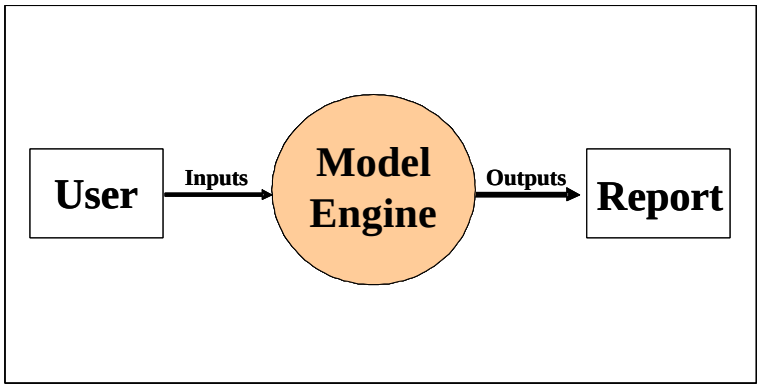

Figure 14. Generic Cost Estimating Model

Why compare cost estimating models? Clearly, as a program manager, I would want the most accurate model used to estimate the cost of my program. What makes cost estimating models more or less accurate? This question could take an entire report to address, so to save time let's

look at the obvious. First, the methods used to derive the model equations are critical to the accuracy. Secondly, if historical data is used to derive the equations then the quality of the data is imperative to the accuracy of the equations. Thirdly, the inputs by the user need to accurately reflect the system that is being estimated; but, this doesn't indicate the accuracy of the model itself, instead of the process of using the model. This third point is mentioned to bring attention to the fact that when developing a model, extreme attention needs to be made to the user's needs and point of reference when using the model. Also, as explained in Chapter-V, aren't the inputs by the user the identified cost drivers? Let's take a look at a few common commercial cost estimating models and compare the techniques used in developing the equations for systems engineering cost estimating.

When conducting research for writing this report the author noticed many similarities in the techniques used. All common models used a parametric approach using historical data, cost drivers, attributes, complexity, size, etc.. Sure, the "system" attributes may have had different names but one could map most of them from model to model. Most models were focused on estimating hardware production cost or software development cost. What about systems engineering cost? Isn't most of systems engineering effort conducted during development? Doesn't systems engineering develop requirements for both hardware and software design engineers?

How is the systems engineering effort calculated in these models? Without access to the equations, the author asked commercial model users and researched model developer's articles, papers etc. and determined in most cases, if not all, systems engineering effort is calculated by

applying a factor to the overall engineering effort (i.e. systems engineering effort is 10% of all engineering effort). Some models include a table with default values for effort allocated to different resources which can be modified by the user to reflect their experience. One could argue that is a sound way to estimate systems engineering effort, after all, don't we have access to data that will give us the proportion of systems engineering to all of engineering? Maybe…maybe not. One fallacy in this approach is that in some cases when all of engineering effort decreases systems engineering may actually increase or vice-versa. Maybe the pie might be the same but the pieces can look very different. Can the attribute selections have an inverse affect on different engineering processes? If the requirements are poorly defined by the customer does this have an impact on systems engineering and the other engineering processes? If systems engineering does their job right then the other engineering processes wouldn't be affected by poorly defined requirements because the system engineer is responsible for defining and refining the requirements. Another factor to consider when the effort isn't allocated correctly is that for some companies the systems engineering labor rates are higher than electrical, software, and mechanical engineering labor rates. It is the author's opinion that separate equations be developed to accurately estimate systems engineering effort.

One model that is currently under development that focuses on systems engineering effort is the COSYSMO project led by Dr. Barry Boehm, USC. Chapter-II Section 2.3 provides an overview of the project. The identified cost drivers, attributes, etc. are similar to those used in developing SECET, however, COSYSMO is concentrated on software intensive systems. [Valerdi, 2002] Yes, systems are becoming more software intensive but many companies still develop "systems" with very little software components.

One interesting approach that COCOMO, COSYSMO, Price-S, SEER-SEM, and others have in common is “Size Drivers”. In software cost estimating models, size drivers are Source Lines of Code (SLOC) and Function Points (FP). Can systems engineering effort be calculated using a “Size Driver”? What would it be? First, let’s examine how size drivers are used.

6.2 Size Drivers

What is a size driver? In the real estate industry “square feet” represents size and is certainly a driver of the cost, with attributes such as wood or brick, type of flooring, type of kitchen cabinets, you get the picture. In the oil industry “barrels of oil” is a standard measure. In the software industry there are only two widely used software measures, 1) Source Lines of Code (SLOC) and 2) Function Points (FP). [MOSAIC, 2002] Using the above sizing measures, the real estate, oil, and software industries can define productivity as “square feet”, “barrels of oil”, and “SLOC” or “FP” per unit of labor and expense respectively. In other words, the software industry standard might be 10 SLOC per hour of labor. Then adjustment factors can be applied to the baseline standard based on complexity of the software to derive the total software estimate. Let’s take a look at what some software cost estimating experts have to say about using SLOC and FP as a means to measure software size and then examine if there is a potential application to sizing systems engineering.

6.2.1 SLOC

First of all, the top commercial software cost estimating models used today use some form of SLOC to produce software cost estimates, based on that alone, there must be some accuracy to this method. According to Price Systems, their software cost estimating model Price S “delivers estimates to within 5% of actual cost after calibration”. [PRICE, 2002] One might question this

method because software systems are coded in many different languages thus efficient code is penalized by having a smaller size. "An entire generation of software researchers assumed incorrectly that improving productivity meant increasing the number of lines of code that could be developed per year, hence lower the cost per source line." [Dreger, 1989] There-in lies the dilemma, some experts believe that SLOC is a viable, accurate means to estimate software cost, while others such as Caper Jones stated at a talk to the Chicago Quality Assurance Association on November 22, 1996 that anyone using Lines of Code (LOC) is "committing profession malpractice". [MOSAIC, 2002] What about Function Points used for estimating software costs?

6.2.2 Function Point Analysis (FPA)

Originally developed by Allan J. Albrecht in 1979 and since evolved substantially, FPA is by far the most accurate and effective software metric ever developed. [Dreger, 1989] In his book, Caper Jones states; "since 1991…, the function point metric has now become the dominant software metric in the United States and in at least 20 other countries". [Jones, 1996] He goes on to state that the uses of FP are expanding as a general business metric such as Business Process Reengineering, Outsource Analysis, Taxation, Tool Capacities, Make Verses Buy Analysis, Software Quality, and others. "Using function point analysis, the Internal Revenue Service (IRS) has significantly improved the quality of its software project management as measured by the Software Engineering Institute (SEI) Capability Maturity Model (CMM)." [Tichenor, 1997] Their adapted method called the "fast count" gives them a function point sizing accuracy of r = 0.9985 when compared to the standard International Function Point Users Group (IFPUG) method and allows them to size software about four times faster than the IFUG method allows. So enough about how great it is…what is a Function Point?

Many books and papers have been written describing function point analysis in great detail, which will not be attempted here, however, a brief overview is warranted. A function point as defined by Brian Dreger is "one end-user business function". [Dreger, 1989] For example, a program rated as having 100 function points delivers 100 business functions to the user. What are business functions? Software business functions are commonly categorized into five groups; 1) *Outputs* (bill of materials, customer invoices, payroll checks, reports, etc.), 2) *Inquiries* (on-line input and on-line output, menu screen, help screen, etc.), 3) *Inputs* (screen data entry, mouse, Automatic Teller Machine, bar code readers, etc.), 4) *Files* (databases, master files, logical internal files, etc.), and 5) *Interfaces* (file of records from/to another application, databases shared with/from other applications, transaction files received/sent to another application, etc.). [Dreger, 1989] Boy, didn't we just open up a can of worms. As you might expect without detailed definitions and descriptions of the five groupings of software business functions one could easily double dip or miscount the function points, however, there are standards placed on these definitions to ease confusion and Brian Dreger does an excellent job in his book of providing easy to understand descriptions and examples. How are functions points derived and translated into effort?

First, each function point is given a rating of simple, average, or complex with factors applied (e.g. a simple *output* function point receives a factor of 4 totaling four "unadjusted function points). Then each grouping is totaled to provide a group total "unadjusted function points". Then all groups are totaled providing "total unadjusted function points". After the "total unadjusted function points" are calculated then an adjustment factor is applied based on 14 production environment factors;

1. data communications
2. distributed data or processing
3. performance objectives
4. heavily-used configuration
5. transaction rate
6. on-line data entry
7. end user efficiency
8. on-line update
9. complex processing
10. reusability
11. conversion and installation ease
12. operational ease
13. multiple-site use
14. facilitate change. [Dreger, 1989]

The total “unadjusted function points” is then multiplied by this adjustment factor to derive the total adjusted function point count. This total function point count is then multiplied by a productivity rate (e.g. 100 function points X 1.5 hours per function point = 150 hours). Is “sizing” a feasible method for estimating systems engineering effort?

6.3 Application to Sizing Systems Engineering Effort?

Assuming that “sizing” by counting SLOC or FP are viable and accurate methods for estimating software engineering cost, are there “products” such as lines of code or function points that

systems engineers produce that could be used to estimate the size of systems engineering effort? The COSYSMO project might be on the right track. They have identified seven size drivers of systems engineering effort; 1) number of system requirements, 2) number of major interfaces, 3) number of technical performance measures, 4) number of operational scenarios, 5) number of modes of operation, 6) number of different platforms, and 7) number of unique algorithms. [Valerdi, 2002]

The question is…will counting these size drivers encounter the same problem that counting SLOC has encountered? For example, suppose the systems engineering productivity rate for defining system requirements was one requirement per labor hour and there are ten system requirements, then the total labor hours for defining systems requirements would be ten hours. Just as in lines of code there are more or less efficient system requirements. One system requirement can be ambiguous requiring sub-requirements for a total of five requirements or one requirement can be efficiently written such that no sub-requirements are needed. Both achieve the same results but the efficient requirement is penalized. So let's say both scenarios were accomplished in the same amount of time, say one hour, then one person's rate is 5 requirements per hour and the other person's rate is 1 requirement per hour. If productivity is evaluated on the number of requirements one can write then we may see inefficient results. Remember people will perform to how they are measured.

Does systems engineering cost estimating need to have a size measurement? Maybe…maybe not. Are the COSYSMO size drivers important? Yes. As demonstrated in Chapter-V, SECET was developed without using a "size driver" but it does use some form of the COSYSMO size

drivers as attributes in the development of the parametric equations. The COSYSMO project could be using "size drivers" because the purpose of the project is to enhance the current capability of the COCOMO II model, which uses size drivers, by introducing systems engineering activities. [Valerdi, 2002] It will be interesting to see how this project evolves.

It is the opinion of the author of this report that using function points as a means to size systems engineering effort on a program needs to be examined in more detail and just might prove to be a viable and accurate method.

How does the accuracy of the systems engineering cost estimating method affect follow-on contracts such as Performance Based Logistic (PBL) or support? Chapter-VII will address the "Death Spiral".

CHAPTER-VII
ADDRESSING THE DEATH SPIRAL

7.1 Pay Me Now or Pay Me Later

How many times have we heard that statement? It really should say "Pay Me Now or Pay Me a Lot More Later". As discussed in the introduction of this report, the Honorable Jacques S. Gansler stated, "Unfortunately, we are trapped in a "death spiral." The requirement to maintain our aging equipment is costing us much more each year: in repair costs, down time, and maintenance tempo. But we must keep this equipment in repair to maintain readiness. It drains our resources—resources we should be applying to the modernization of the traditional systems and development and deployment of the new systems." [U.S. Air Force, 2000] What drives costs in the operations and support phase of a program?

In the past, the program manager was only responsible for the acquisition costs and was not held accountable for the Life Cycle Cost (LCC). Naturally, if one receives promotions and raises based on acquisition cost and not LCC then we know where the focus will be. LCC reduction methods such as reliability growth tests, High Accelerated Life Tests (HALT), supportability studies, etc. may be cut during the early stages of design and development because these efforts increase the program manager's design and development costs. Thus, design alternatives reducing LCC may not be considered. One can argue that many decisions influence LCC, however, the greatest opportunity for influencing LCC is realized in the design and development phase of a program. [Blanchard, 1992] Benjamin Blanchard states in his book that experience has shown that a major portion of the projected LCC (approx. 85%) for a program is influenced by the decisions made during the concept design and planning phase and the system preliminary

design phase. (Figure 15) What are the customer and contractor doing during these phases of a program? Put simply, Systems Engineering, understanding the customer's needs and transforming those needs into requirements. The customer needs include not just system performance such as range or velocity, but user's needs as well such as the operational environment and availability requirements. Is it reliable? Will it work when I'm out in the battle field? Will there be one available when I need it? These are just a few of the warfighter's concerns.

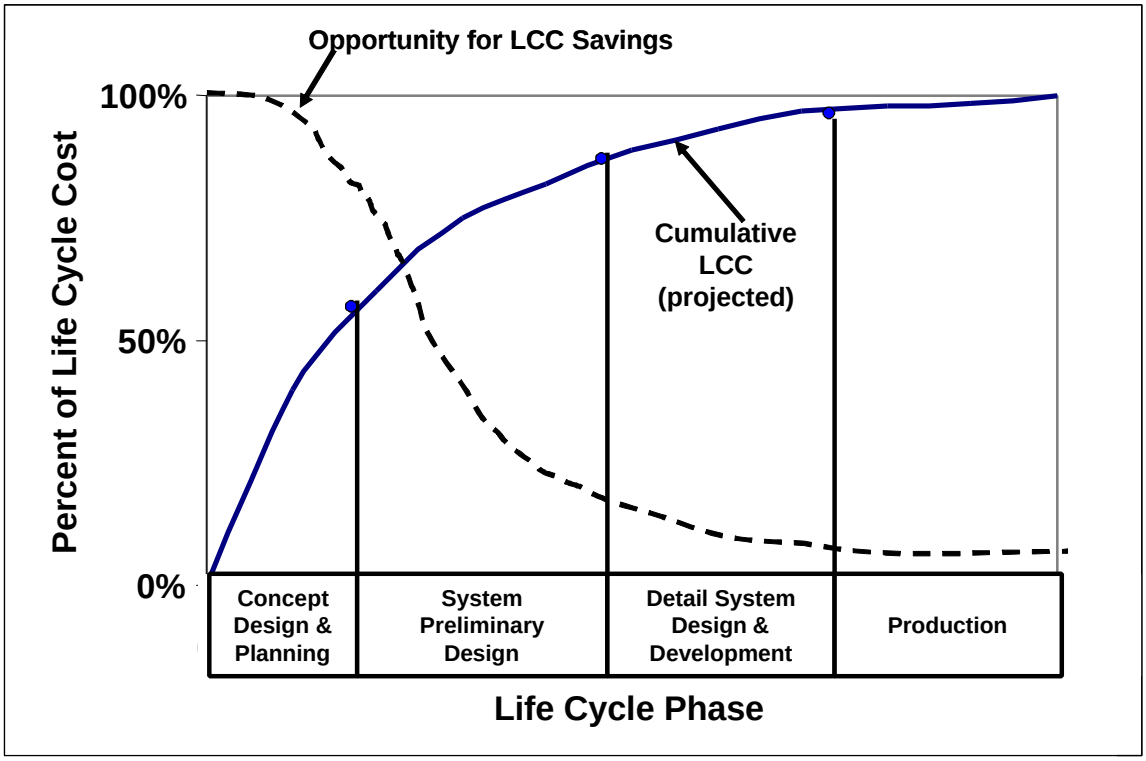

Figure 15. Systems Engineering Opportunity to Influence LCC [Blanchard, 1992]

Clearly, properly implemented systems engineering processes can influence the success of the warfighter and at the same time influence the LCC. What influences properly implemented systems engineering processes? Certainly, adequate funding. How does a program receive adequate funding to implement sound systems engineering processes? The customer's funding

profile is a driver, however, understanding what it cost to implement these processes is a driver as well. A good start to understanding the costs of implementing systems engineering processes is a quality database. A cost estimating methodology and process that is repeatable and defendable could alleviate program managers cutting systems engineering budgets. What happens to systems engineering processes when there isn't enough funding to do it right the first time? Sub-processes, procedures, and tests such as Reliability Growth Tests and Consolidated Systems Engineering for Test are eliminated from the Integrated Master Plan (IMP) and Integrated Master Schedule (IMS).

The next two sections give case studies of how systems engineering in the disciplines of reliability and testability impact a PBL and support costs respectively.

7.2 Reliability Case Study

The following case study shows the impact that a reliability growth test had on projected PBL costs. Reliability growth tests are typically conducted during the early stages of a product development program with the objective of identifying failure modes and incorporating design changes before the program enters the production phase. [Bieda, 1991] Obviously, if failure modes are not identified, corrected, and implemented before the system is fielded then operating and support costs can increase significantly because of increased failures, repair costs, and spares' costs. Not to mention a decrease in availability to the warfighter.

In this case study, the failure modes identified and corrected during the design phase increased the system Mean Time Between Failure (MTBF) by 3X. This reliability growth resulted in a

$300M cost reduction for the PBL contract (Figure 16). Minimal cost associated with the reliability growth test is absorbed by the development contract, however, the payback in the PBL cost is significant.

Cause and effect…accurate systems engineering cost estimating caused appropriate systems engineering funding which caused the reliability growth test to be conducted which caused the MTBF to increase by 3X which caused the PBL cost to decrease by $300M. Now, accurate systems engineering cost estimating isn't the primary reason for this effect. Clearly, proper implementation of systems engineering processes played a key role, however, without accurate cost estimating and funding it makes it very difficult to implement processes that cost a little more upfront but with high payback later on. Just think, the $300M savings can be spent on developing new technology instead of repairing failing equipment…Addressing the Death Spiral.

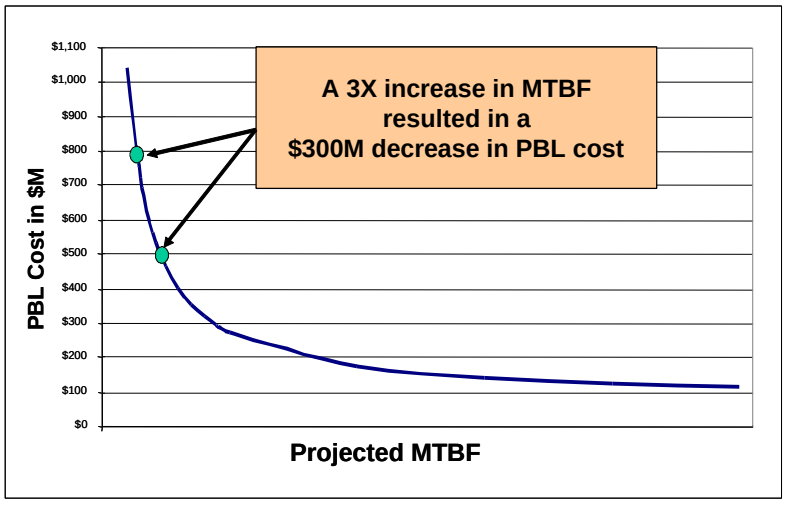

Figure 16. Reliability Growth Test Impact on PBL Cost

7.3 Testability Case Study

The following case study shows the potential support cost reduction by implementing Consolidated Systems Engineering for Test (CSET) on a program. CSET is similar to Integrated Diagnostics (ID) concept that was fostered by the DoD in the '80s. Both are focused on coordinating test/diagnostic capabilities to meet customer needs, with CSET expanding to accommodate the contractor's internal needs and requirements with the objective being to minimize cost, schedule, and risk while maximizing customer satisfaction. [Sallade, Brown, 1999]

Failing to properly fund, coordinate, and consolidate test needs and requirements can contribute to the death spiral. This is true for development, production, and support programs. Design Verification Test (DVT), Hardware Integration, Hardware-Software Integration, System Integration, System Verification/Validation, and, Factory/Production Test requirements can be consolidated, coordinated, flowed down and managed to minimize cost, schedule, and risk.

For example, "by utilizing Built-In-Test (BIT) in Integration Verificaton &Validation (IV&V) we simplify the IV&V process and minimize down time while we collect BIT performance data that can be fed back to mature and stabilize the BIT tests and parameters. This in turn reduces the initial Can Not Duplicate (CND) and False alarm rate of the fielded systems and avoids the associated unnecessary field returns." [Sallade, Brown, 1999]

In this case study, Program X had a history of added costs due to a high number of CND returns, creating increased down time, reduced operational readiness and availability, and increased

repair costs and the need for additional spares. Analyses conducted demonstrated the CNDs had a significant impact to the return rate. Further evaluation disclosed that CSET methods were not implemented during the development phase of Program X. It was believed that the CND rate would have been significantly reduced if CSET methods had been implemented, resulting in a potential 40% decrease in support costs (Figure 17).

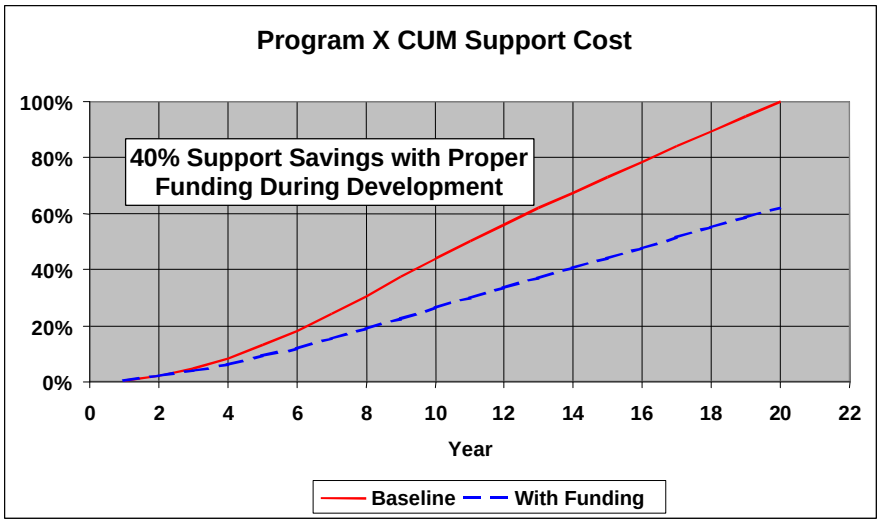

Figure 17. Testability Impact on Support Cost

What about the increased development costs for CSET implementation? It has been shown that the added non-recurring costs for BIT, driven by the increased requirements and capability placed on BIT to support the other test environments, are actually offset by decreased schedule and development test costs. BIT/testability hardware increases recurring cost but are offset by the savings in factory and depot test equipment. Overall, CSET implementation reduces other development costs causing a marginal increase in total development cost, if any. Bottom line, this is another example of how proper funding and implementation of systems engineering processes can impact the Death Spiral.

CHAPTER-VIII
SUMMARY and CONCLUSIONS

Systems Engineering shapes the success of all phases of a program by transforming customer needs into requirements that drive the design, ensuring structured integration and verification against the requirements, ensuring the technical integrity of a system, and integrating the efforts of all engineering disciplines. Systems engineering success is driven by properly implemented processes which are in part driven by adequate funding. Along with the customers funding profile, adequate systems engineering funding is driven by the ability to accurately estimate the funding required to implement sound systems engineering processes. Estimating systems engineering effort accurately is critical to the success of a program.

The advantages and disadvantages of four common cost estimating approaches were given with the conclusion that system maturity, program phase, data, and time will determine the best approach to use and that two or more approaches should be used when providing an estimate. One approach, parametric cost estimating was discussed in detail with an example of how Raytheon built the Systems Engineering Cost Estimating Tool (SECET). The process steps to building SECET were discussed at length and included 1) Defining the requirements, 2) Identifying the cost drivers, 3) Establishing a data collection methodology, 4) Developing the Data Flow Diagrams, 5) Assigning parameter values, 6) Developing the model equations using regression analysis, 7) Establishing the model software structure, and 8) Validating the model. The coefficient of determination (R^2) was calculated to determine the strength of the model and a risk analysis was performed to assess the risk in the prediction based on the parameter selections by the Wideband Delphi.

Commonly used commercial hardware and software modeling techniques were discussed along with a comparison to SECET. It was concluded that most commercial cost estimating models apply a factor to the overall engineering effort to derive the systems engineering effort, which may not be the right approach because some system attributes may affect engineering processes differently. "Sizing" is a common approach for estimating software with Source Lines Of Code (SLOC) and Function Points being the most widely used sizing methods. A comparison of these two sizing methods was given along with comments from experts. Commercial models using SLOC for sizing are commonly used in many industries with a "claimed" high degree of accuracy. According to some experts using function points as a means to size software is far more accurate than lines of code. The function point metric has become the dominant software metric in the United States and has expanded to other business areas. The COSYSMO project is using size drivers to estimate systems engineering effort. Function Point Analysis may prove to be a viable means to estimating accurate systems engineering effort.

Experience has shown that the greatest opportunity for influencing Life Cycle Cost (LCC) is realized during the design and develop phase of a program, the phase when most of systems engineering effort is performed. Too often during the design and development phase, not enough funding is allocated to systems engineering processes because of inadequate estimates or because program managers cut important systems engineering processes such as reliability, testability, and supportability to save short term costs without considering the impact to follow on contracts such as Performance Based Logistics (PBL) and support. Two case studies were given demonstrating the impact systems engineering processes such as Reliability Growth Test (RGT)

and Consolidated Systems Engineering for Test (CSET) can have on support costs, contributing to the “Death Spiral”.

Bottom line, Systems Engineering cost estimating methodologies need to improve because it makes good business sense not just short term but long term, addressing the death spiral. With CMMI knocking at our door, estimates need to be repeatable and defendable, and companies must implement standard processes and tools such as those discussed in this report.

REFERENCES

1. [U.S. Air Force, 2000] "Cost as an Independent Variable and Total Ownership Cost GUIDEBOOK", *U.S. Air Force RTOC Reduction in Total Ownership Cost*, 30 October 2000, VERSION 2.0.

2. [PRICE, 2002] PRICE Systems Website: http://pricesystems.com.

3. [DoD, 1999] *DoD Joint Industry/Government Parametric Estimating Handbook*, Second Edition (Spring 1999).

4. [SEER, 2002] Galorath Website: http://galorath.com.

5. [NASA/AF, 2002] NASA NASA / Air Force Cost Model (NAFCOM) Website: http://nafcom.saic.com.

6. [ParaModel, 2002] Mainstay Software corporation Website: http://mainstay.com.

7. [USC, 2002] USC Center for Software Engineering Ongoing Research, Website: http://sunset.usc.edu/research/COCOMOII/index.html.

8. [Kelberlau, 2001] "SE Cost Estimating What Others are Doing", **Author**: Bill Kelberlau, Principle Systems Engineer, Raytheon, Plano, Texas, (Presentation Slides: October, 2001).

9. [Reifer, 2002] "COSYSMO Constructive Systems Engineering Cost Model Status Briefing: GSAW 2002", **Author**: Donald Reifer, University of Southern California Center for Software Engineering, (Presentation Slides: February, 2002). Website: http://valerdi.com/cosysmo.

10. [Blanchard, 1998] SYSTEMS ENGINEERING MANAGEMENT, Second Edition, **Author**: Benjamin S. Blanchard, **Publisher**: John Wiley & Sons, Inc. New York, 1998, (page 12).

11. [Raytheon, 2002] Raytheon Intranet: Systems Engineering Homepage. Website: http://raytheon.com.

12. [Kollman, Norby, 2001] "Systems Engineering Principles", **Author**: Tom Kollman, Raytheon Systems Engineering, Greg Norby, Raytheon Systems Engineering, (Presentation Slides for Texas Tech ENG 5000 class, December, 2001).

13. [Raytheon IPDP, 2002] Raytheon Integrated Product Development Process (IPDP).

14. [Michaels, Wood, 1989] DESIGN TO COST, **Author**: Jack V. Michaels. PE, CVS; William P. Wood, CVS, **Publisher**: John Wiley & Sons, Inc. New York, 1989.

15. [Boehm, 1981] SOFTWARE ENGINEERING ECONOMICS, **Author**: Barry W. Boehm, **Publisher**: Prentice-Hall, Inc., Englewood Cliffs, N.J., 1981.

16. [Dennedy, 1998] "The FAR and Parametric Estimating: What is a Properly Calibrated and Validated Model?", *The International Society of Parametric Analysts and The Society of Cost Estimating and Analysis Joint International Conference Proceedings, June 1998*, (page 944), **Author**: Jim Dennedy, MCR Federal, Inc..

17. [Spector, 1995] An August 28, 1995 memorandum from the Office of the Under Secretary of Defense for Directors of Defense Agencies, signed by Eleanor A. Spector, Director, Defense Procurement, *DoD Joint Industry/Government Parametric Estimating Handbook*, Second Edition (Spring 1999).

18. [Hertling, 1998] "An Overview of the Parametric Cost Estimating Initiative", *The International Society of Parametric Analysts and The Society of Cost Estimating and Analysis Joint International Conference Proceedings, June 1998*, (page 940), **Author**: Virgil Hertling, Contract Cost/Price Analyst at Headquarters, Air Force Materiel Command, Directorate of Contracting, Pricing and Finance Branch at Wright Patterson AFB, Ohio.

19. [Gershenfeld, 1999] THE NATURE OF MATHEMATICAL MODELING, **Author**: Neil Gershenfeld, **Publisher**: Cambridge University Press, New York, N.Y., 1999.

20. [Minkiewiez, 2002] "What is parametric cost estimating", e-mail correspondence (2002) from Arlene Minkiewiez, Chief Scientist, Price Systems to Belinda Brown (author of this report), Raytheon.

21. [Raytheon CMMI, 2002] "System and Product Attribute Definitions", Raytheon CMMI Systems Engineering Cost Estimating Website, 2002.

22. [Delaney, Vaccari, 1989] DYNAMIC MODELS AND DISCRETE EVENT SIMULATION, **Author**: William Delaney and Erminia Vaccari, **Publisher**: Marcel Dekker, Inc., New York, N.Y., 1989 (page 3).

23. [The American Heritage Dictionary, 1976] THE AMERICAN HERITAGE DICTIONARY OF THE ENGLISH LANGUAGE, New College Edition, **Editor**: William Morris, **Publisher**: Houghton Mifflin Company, Boston, Massachusetts, 1976 (pages 85 & 951).

24. [Davis, 1993] SOFTWARE REQUIREMENTS REVISION OBJECTS, FUNCTIONS, & STATES, **Author**: Alan M. Davis, **Publisher**: PTR Prentice Hall, Englewood Cliffs, N.J., 1993 (page 69).

25. [Schmidt, Launsby, 1997] UNDERSTANDING INDUSTRIAL DESIGNED EXPERIMENTS, Fourth Edition, **Author**: Stephen R. Schmidt, Ph.D., Robert G.

Launsby, M.S., **Publisher**: Air Academy Press & Associates, Colorado Springs, CO 1997.

26. [Hunt, 1988] CALCULUS WITH ANALYTIC GEOMETRY, **Author**: Richard A. Hunt, Purdue University, **Publisher**: Harper & Row Publishers, Inc., 1988. (page 766)

27. [Ertas, Jones, 1996] THE ENGINEERING DESIGN PROCESS, Second Edition, **Author**: Atila Ertas, Jesse Jones, Department of Mechanical Engineering Texas Tech University, **Publisher**: John Wiley & Sons, Inc., New York, 1996.

28. [Devore, Peck, 1986] STATISTICS THE EXPLORATION AND ANALYSIS OF DATA, **Author**: Jay Devore, Roxy Peck, California polytechnic State University, **Publisher**: West Publishing Company, St. Paul, MN 1986.

29. [Valerdi, 2002] "COSYSMO Constructive Systems Engineering Cost Model: USC Annual Research Review", **Author**: Ricardo Valerdi, University of Southern California Center for Software Engineering, (Presentation Slides: March, 2002). Website: http://valerdi.com/cosysmo.

30. [MOSAIC, 2002] "Software Sizing Measures", **Author**: Unknown. Website: http://testablerequirements.com/soft_size_meas_m.htm.

31. [Dreger, 1989] FUNCTION POINT ANALYSIS, **Author**: J. Brian Dreger, Boeing, **Publisher**: Prentice Hall, Englewood Cliffs, New Jersey, 1989.

32. [Jones, 1996] APPLIED SOFTWARE MEASUREMENT ASSURING PRODUCTIVITY AND QUALITY, Second Edition, **Author**: Caper Jones, **Publisher**: McGraw-Hill, New York, New York, 1996.

33. [Tichenor, 1997] "The Internal Revenue Service Function Point Analysis Program: A Brief", **Author**: Charles B. Tichenor, 1997.

34. [Blanchard, 1992] LOGISTICS ENGINEERING AND MANAGEMENT, **Author**: Benjamin S. Blanchard, Virginia Polytechnic Institute and State University, **Publisher**: Prentice Hall, Englewood Cliffs, New Jersey, 1992.

35. [Bieda, 1991] "Reliability Growth Test Management in a Product Assurance Environment Within the Automotive Component Industry", *Annual Reliability and Maintainability Symposium Proceedings, January 1991,* (page 317), **Author**: John Bieda, AC Rochester Division GMS, Flint Michigan.

36. [Sallade, Brown, 1999] "Consolidated Systems Engineering for Test", **Author**: Rex Sallade, Belinda Brown, Raytheon, 1999.

APPENDIX A
ATTRIBUTE DEFINITIONS

System Level Attributes

System Testing – this attribute defines the types of system testing required by the program. Note that the user may select one or more of the options.

- Formal Qual Test – formal testing of the system is required to ensure the system meets all requirements. This typically would imply formal test plans, procedures/results requiring customer approval and in some cases customer witnessing.
- Field Test – this entails testing of the system in a remote location under 'real world conditions'. This may require coordination with other agencies regarding test ranges, supporting systems, etc...
- Flight-Testing – system will be tested in flight conditions. May require that the system meet certain environmental conditions, form/fit/function or other conditions that are unique to a flight environment. Generally implies customer test range and flight system.

Program Classification – this attribute is used to define the security classification of the program. The higher the classification, the more effort will be required to comply with security requirements.

- Unclassified – no security requirements associated with the program
- Classified – program, or some aspects of the program, are at the confidential or secret level
- SAR – program, or some aspects of the program, are at the top secret or Special Access Required level

Customer Relationship – this attribute is used to define the level of customer involvement on the program. If the customer is directly involved in all aspects of the program, i.e. participates in all design reviews, has on-site representatives, etc., then the effort can be expected to be slightly higher.

- Indirect Involvement – the customer has delegated the day-to-day management and technical direction of the program to Raytheon. The customer participates and provides direction at major program milestones.
- Direct Involvement – the customer, or his representative, actively participates in the day-to-day execution of the program. Customer approval or concurrence is required for all significant technical decisions.

Staff Experience – this defines the experience level of the systems engineering staff expected to be assigned to the program. A more experienced staff would be expected to accomplish the same set of tasks with less effort than an inexperienced staff.

- Extensive Experience – the entire staff has previous experience with this type of system/technology. For example, this might be an add-on program where the staff is the same group of people who did the original engineering effort. Typically, the average experience level for all SE/SSE's assigned to the program would be >15 years.
- Normal Experience – the staff is comprised of a mix of engineers, which represents the North Texas engineering pool. This would include both inexperienced and experienced systems engineers. Typically, the average experience level for all SE/SSE's assigned to the program would be 5-15 years.
- Inexperienced – the systems engineers assigned to this program have little or no prior experience with this system/technology. Effort will be required to 'get up the learning curve'. Typically, the average experience level for all SE/SSE's assigned to the program would be <5 years.

Type of Schedule – this attribute defines the type of schedule required by the program. Schedules that have been shortened may require additional resources to execute. Schedules that have been lengthened because of funding profiles or program constraints may also incur additional cost because of the 'overhead' that must be carried along.

- Compressed – the schedule is accelerated (shortened) by at least 20% over what would be considered a normal development schedule.
- Normal – there is no requirement to shorten or lengthen the normal development cycle.
- Stretched – the schedule has been lengthened by at least 20% over what would be considered a normal development schedule.

External Interfaces – the type of external interfaces can have a significant impact on the systems engineering effort. The definition of easy and complex is provided below.

- Easy – the external interfaces are simple and straightforward. While there may be many interfaces, the technology, bandwidth, data, etc. are easily understood and require no extraordinary effort to define.
- Normal – the external interfaces are a standard mix of technologies, only a few of the interfaces are not fully understood. The new interfaces are not a major risk to the program.
- Complex – the external interfaces require the use of new technology, stress the physical limits of the interface, or impose severe constraints on bandwidth/latency, etc. Other examples might include interfacing to a system/subsystem that is also in development necessitating some volatility in the interface definition.

Simulation/Model Environment – this attribute will define the development environment available to the system engineer in developing system requirements. Almost all programs use some sort of model (or combination of models) to define system requirements, be it a 6DOF, radar model, EO model, etc. When these models are pre-existing and have been validated on

prior efforts, the system engineer does not have to expend effort to develop these resulting in less overall effort on the program.

- Existing Tools – models/simulations that are directly applicable to the program exist and have been validated as part of previous efforts. These can be used directly with minimal changes to support requirements definition on the program.
- Some Development of Tools – existing models/simulations must undergo modification to reflect the requirements of the program. An example might be an existing radar model that does not support one or more modes required by the program. New mode(s) must be added to the model and validated in order to support requirement definition for the program.
- Extensive Development Required – models/simulations that accurately reflect the system requirements do not exist and must be developed and validated prior to defining system performance/requirements.

Requirements Definition – this defines the degree to which the requirements are known. This has a direct correlation to the level of effort necessary to define the subsystem; number of trade studies required, hardware/software tradeoffs, etc.

- Well Defined – subsystem requirements are well defined as a result of customer specifications, previous programs, etc. Minimal tradeoffs are necessary to finalize the requirements at the subsystem level.
- Normal – the top-level system requirements are defined but must be allocated to various subsystems. Nominal trade studies will be required to properly allocate/define the subsystem requirements. The technologies/domains required for the subsystem are understood.
- Not Defined – little or no definition exists for subsystem requirements. Significant trade studies will be required to properly allocate system level requirements to subsystems. New technologies/domains may be required to be explored/understood to properly define the subsystem requirements.

Product/Component Attributes

Testing – this attribute defines the level of testing that will be performed at the subsystem level. Note that one or more options may be selected.

- Prototype – testing will be performed under engineering conditions in a laboratory environment.
- Environmental – subsystem will be subjected to full environmental testing (‘shake and bake’)
- Component – formal test, i.e. approved test plans/procedure/results, at the component level
- Integrity – formal test at the subsystem / product level
- HALT – High Accelerated Life Test

- RD/GT – Reliability Demonstration / Growth Test

Requirements Definition – this defines the degree to which the requirements are known. This has a direct correlation to the level of effort necessary to define the subsystem; number of trade studies required, hardware/software tradeoffs, etc.

- Well Defined – subsystem requirements are well defined as a result of customer specifications, previous programs, etc.. Minimal tradeoffs are necessary to finalize the requirements at the subsystem level.
- Normal – the top-level system requirements are defined but must be allocated to various subsystems. Nominal trade studies will be required to properly allocate/define the subsystem requirements. The technologies/domains required for the subsystem are understood.
- Not Defined – little or no definition exists for subsystem requirements. Significant trade studies will be required to properly allocate system level requirements to subsystems. New technologies/domains may be required to be explored/understood to properly define the subsystem requirements.

Design – this attribute defines the degree of system design required. This is the effort that is typically performed to write the subsystem sections in a SSDD.

- Simple Modification – this is a simple modification to an existing subsystem. The problem definition is well understood and no new major technologies/capabilities are required.
- Extensive Modification – major modifications are required to an existing subsystem. This might include addition of a new mode/capabilities, technology upgrades, etc.
- New Design – this requires a complete design of the subsystem. New technologies may be required, complete definition of all subsystem capabilities is required, extensive trade studies may be necessary to complete the subsystem design.

Technology Maturity – this defines the level of technology maturity necessary to meet the subsystem requirements. Having to develop a new technology typically requires additional systems engineering effort to perform the appropriate trade studies and industry surveys. Technology might include such things as clock speed, bandwidth, processing capability, power density, etc.

- Existing Technology – existing technology is sufficient to meet subsystem requirements.
- Some new technology – a mix of technologies where the new technology is not a major risk to the program.
- New Technology – new technology will need to be developed to achieve the desired level of subsystem performance.

Performance Requirements – performance requirements refers to the level of performance necessary to meet subsystem requirements.

- Easy – the performance requirements are well understood and should be easily attainable with current methods/technologies/strategies. Trade studies should be minimal.
- Normal – performance requirements are not beyond State of the Art. While there may be some trade studies required, it is not expected that major technical obstacles will be encountered.
- Aggressive – the performance requirements of the subsystem are extremely aggressive. No existing system is capable of achieving these requirements and new methods/technologies/strategies will be required in order to meet requirements. Significant trade studies may be required to determine how these requirements will be met.

Internal interfaces – this defines the type of internal interfaces. While these selections will not map one to one with the external interfaces of a program, they will provide an approximation of the effort required (it is recognized that selection of this attribute will be a matter of 'engineering judgement'). The definition of easy and complex is provided below.

- Easy – the internal interfaces are simple and straightforward. While there may be many interfaces, the technology, bandwidth, data, etc. are easily understood and require no extraordinary effort to define.
- Normal – the internal interfaces are a standard mix of technologies, only a few of the interfaces are not fully understood. The new interfaces are not a major risk to the program.
- Complex – the internal interfaces require the use of new technology, stress the physical limits of the interface, or impose severe constraints on bandwidth/latency, etc.. Other examples might include interfacing to a system/subsystem that is also in development necessitating some volatility in the interface definition.

Operational Environment – this attribute defines the environment the subsystem will be expected to operate in. This typically refers to the environmental requirements and can significantly impact the design and test of the subsystem.

- Benign – the operating environment is not expected to stress the product with respect to the normal military product of this type.
- Normal – the operating environment is expected to stress the product similarly to the general military product of this type.
- Severe – one or more aspect of the operating environment is expected to be severe and beyond our normal design expectations.

HW Components – this defines the number of hardware subassemblies (generally CCAs) that might be required by the subsystem. The number of subassemblies is self-explanatory.

SW Components – this defines the number of software CSCI that might be required by the subsystem. The number of CSCI is self-explanatory.

Percent Vendor / Commercial-off-the-shelf (COTS) (subassemblies) – this defines what percent of the subassemblies are expected to be purchased and / or COTS. The selections are self-explanatory.

WC/Tolerance Analysis Circuits – Worst Case/Tolerance Analysis is an analytical validation of long-term electrical design performance margin accounting for part and circuit variability over externally induced environmental and electrical extremes. Select the estimated number of circuits for which detailed analysis will be required.

- 0-2 circuits
- 3-5 circuits
- 6 or greater circuits

HW Safety/Hazard Analysis - HW Safety/Hazard Analysis determines the specific hazards, causes, mitigations, and verification for the hardware and facilities. These concerns are those that result in damage to the system or facility, or injury or death to personnel via a physical interaction. Typical hardware concerns are strength, weight, sharp edges and corners, vibration, chemical reaction, or energy exchange. The goal of this analysis is to provide feedback early in the design so that changes can be made in a cost-effective manner. Early identification of changes necessary to maintain or increase the safety of facilities is also desired. The number of safety critical areas (such as RF, high voltage, Laser, explosives) defines the amount of detailed hazard analysis that will be required for the product. Select the quantity of areas that will require this analysis.

- 0-1 hazard area
- 2-3 hazard areas
- 4 or greater hazard areas

SW Safety/Hazard Analysis - SW Safety Analysis provides safety requirements to the software development team early in the development process. Select if analysis of software for safety determination will be required.

- No
- Yes

RMSS Performance Requirements - Reliability/Maintainability/Supportability/System Safety and Human Factors **(RMSS)** performance requirements refers to the level of RMSS performance necessary to meet requirements.

- Easy – the performance requirements are well understood and should be easily attainable with current methods/technologies/strategies. Trade studies should be minimal.

- Normal – performance requirements are not beyond State of the Art. While there may be some trade studies required, it is not expected that major technical obstacles will be encountered.
- Aggressive – the performance requirements are extremely aggressive. No existing system is capable of achieving these requirements and new methods/technologies/strategies will be required in order to meet requirements.

Special Products Attributes

Special Products

FMECA/FMEA - A Failure Mode, Effects and Criticality Analysis (FMECA) is an evaluation/design technique that examines the potential failure modes within a system in order to determine the end effects of those failure modes. Severity and probability of occurrence are used to rank those failure modes and prioritize corrective action or mitigation. The FMECA is used in planning the system maintenance concept and activities, defining system architecture, fault recovery, fault tolerance, and system failure detection and isolation provisions. A Failure Mode, Effects Analysis (FMEA) is a subset of FMECA.

RQT - The Reliability Qualification Test (RQT) or Reliability Demonstration Test (RDT) is conducted to measure equipment MTBF while exposed to stresses expected during its actual service life. MTBF is calculated based on the total accumulated test time and the total number of chargeable failures. Compliance is determined by comparing the test results with the accept/reject criteria of the RDT Plan.

HF Demonstrations/Reviews - The Human Factors (HF) Demonstration/Review is a formal process conducted by the product developer to determine whether specific HF requirements have been achieved in the design. This test will be conducted in accordance with the HF Test Plan. This test plan defines the test objectives, schedule, specific procedures, test method, pass/fail criteria, test equipment, test team, etc. Details of the HF verification requirements and methods for compliance are covered in the HF Test Plan.

M/T/S Demonstration - The Maintainability/Testability/Safety demonstration test is a formal process conducted by the product developer to determine whether specific M/T/S requirements have been achieved in the design. This test will be conducted in accordance with the M/T/S Test Plans. These test plans define the test objectives, schedule, specific procedures, test method, pass/fail criteria, test equipment, test team, etc. Details of the M/T/S verification requirements and methods for compliance are covered in the M/T/S Test Plans.

LORA - The Level of Repair Analysis (LORA) or Repair Level Analysis (RLA) evaluates each alternative to determine the optimum utilization of support resources over the life of the system. The decision can often be made on purely economic considerations. The LORA/RLA model is designed to determine the most cost-effective alternative to performing a maintenance task and to determine where the task can be accomplished most cost effectively.

LSA - The Logistics Support Analysis (LSA) is an iterative analytical process by which the logistics support necessary for a new system is identified and evaluated. LSA is a design tool employed throughout the early phases of system development and often includes the maintenance analysis, reliability-centered maintenance requirements, Level-Of-Repair Analysis (LORA), Life Cycle Cost (LCC) analysis, spares analysis, resource analysis, warranty analysis, and logistics modeling.

LSAR - The Logistics Support Analysis Record (LSAR) formally documents the results of the LSA. The LSAR is the single database and source for all logistic support data and information for the new system. The results of the maintenance task analysis process are recorded in narrative form on the LSA data table D. A companion data table D1 is also prepared for each maintenance task that contains a tabular list of all the support and test equipment, tools, and spare and repair parts required to perform the task.

Tech Manuals - Technical Manual Development is the process used to prepare the requisite Operation & Maintenance Manuals and Illustrated Parts Breakdown Manuals or Repair Parts and Special Tools List to support the new system. The purpose of these manuals is to describe the system, provide procedures necessary to install and operate the system, perform maintenance, and provide a means for parts identification.

Training - Technical Training material development and presentation is the process used to prepare the customer to operate and maintain the system. Instructor Guides, Student Lessons Guides, and Course Audiovisual materials are used to support the training. The instructor that prepares the material also presents this material to the customer Instructor and Key Personnel (IKP). In-progress reviews are conducted at key periods during the development process. The customer is able, using the training materials provided during the training class, to establish follow-on training for system operators and maintainers.

TRD - The Test Requirements Document (TRD) specifies, in detail, how the unit is to be tested. It defines the method the design engineer communicates detailed test requirements to the test engineer.

Special Products' Attributes

FMECA/FMEA (Level) - Indicate if the FMECA or FMEA is required at the signal, SRU or LRU level. The effort will be modified by the number of Product and/or Components in the system (20 signals per SRU are assumed for signal level).

- Signal level
- SRU level
- LRU level

FMECA/FMEA (Functional) - FMECA/FMEA can be a combined effort to support Reliability, Testability and/or Safety. Select the appropriate functions.

- Reliability
- Testability

- Safety

Tech Manual (Type) - Technical manual generations will greatly vary depending on the type required. This attribute provides selection of types that will require different efforts.
- Operator
- Maintenance

Tech Manual (complexity) - Technical manual generations will greatly vary depending on the complexity. This attribute provides selection of complexities that will require different efforts.
- Basic
- Detailed
- Interactive

Training (Type) - Technical Training material development and presentation will greatly vary depending on the type required. This attribute provides selection of types that will require different efforts.
- Operator
- Maintenance

Level of Analysis - RMSS analyses will greatly vary depending on the level required. This attribute provides selection of levels that will require different efforts.
- LRU only
- LRU and SRU
- LRU, SRU, SubSRU

APPENDIX B
SECET WBS OUTPUT EXAMPLE

SECET WBS Ouput				Predicted Hours
Total Systems Engineering Labor Hours				**186853**
Total System Labor Hours				**182463**
System 1	**Radar**	**System IPDP Level-2 Tasks**		**15387**
		Systems Engineering Planning & Mgmt		1177
		System Requirements Definition		2670
		System Preliminary Design		4227
		System Integration, Verification & Validation		7313
	Product 1	**Antenna**	**Active Element**	**41496**
		Product IPDP Level-2 Tasks		**9576**
		Product Requirements Definition		1596
		Product Preliminary Design		3192
		Product Integration, Verification & Validation		4788
		Hardware Components' IPDP Level-2 Tasks		**15960**
		Component Requirements Definition		1596
		Component Preliminary Design Support		3192
		Detail Design Support		4788
		Component Integration and Test		6384
		Software Components' IPDP Level-2 Tasks		**15960**
		Component Requirements Definition		1596
		Component Preliminary Design Support		3192
		Detail Design Support		4788
		Component Integration and Test		6384
	Product 2	**Receiver-Exciter**	**Hybrid**	**42224**
		Product IPDP Level-2 Tasks		**9744**
		Product Requirements Definition		1624
		Product Preliminary Design		3248
		Product Integration, Verification & Validation		4872
		Hardware Components' IPDP Level-2 Tasks		**16240**
		Component Requirements Definition		1624
		Component Preliminary Design Support		3248
		Detail Design Support		4872
		Component Integration and Test		6496
		Software Components' IPDP Level-2 Tasks		**16240**
		Component Requirements Definition		1624
		Component Preliminary Design Support		3248
		Detail Design Support		4872
		Component Integration and Test		6496
	Product 3	**Processor**	**FPGA Based**	**41860**
		Product IPDP Level-2 Tasks		**9660**
		Product Requirements Definition		1610
		Product Preliminary Design		3220
		Product Integration, Verification & Validation		4830
		Hardware Components' IPDP Level-2 Tasks		**16100**
		Component Requirements Definition		1610
		Component Preliminary Design Support		3220
		Detail Design Support		4830
		Component Integration and Test		6440
		Software Components' IPDP Level-2 Tasks		**16100**
		Component Requirements Definition		1610
		Component Preliminary Design Support		3220
		Detail Design Support		4830
		Component Integration and Test		6440
	Product 4	**Transmitter**	**AESA Driver Module**	**41496**
		Product IPDP Level-2 Tasks		**9576**
		Product Requirements Definition		1596
		Product Preliminary Design		3192
		Product Integration, Verification & Validation		4788
		Hardware Components' IPDP Level-2 Tasks		**15960**
		Component Requirements Definition		1596
		Component Preliminary Design Support		3192
		Detail Design Support		4788
		Component Integration and Test		6384
		Software Components' IPDP Level-2 Tasks		**15960**
		Component Requirements Definition		1596
		Component Preliminary Design Support		3192
		Detail Design Support		4788
		Component Integration and Test		6384
	Special Products			**4390**
		FMECA/FMEA		190
		RQT		240
		Tech Manuals		1200
		Training		1260
		TRD		1500

Table 17. SECET WBS Output Example (notional data)