NAVAL POSTGRADUATE SCHOOL

MONTEREY, CALIFORNIA

THESIS

NAVAL COMBAT SYSTEMS PRODUCT LINE ECONOMICS: EXTENDING THE CONSTRUCTIVE PRODUCT LINE INVESTMENT MODEL FOR THE AEGIS COMBAT SYSTEM

by

Kyle A. Chance

June 2019

Thesis Advisor: John M. Green
Co-Advisor: Raymond J. Madachy

THIS PAGE INTENTIONALLY LEFT BLANK

REPORT DOCUMENTATION PAGE			*Form Approved OMB No. 0704-0188*

1. AGENCY USE ONLY *(Leave blank)*	**2. REPORT DATE** June 2019	**3. REPORT TYPE AND DATES COVERED** Master's thesis

4. TITLE AND SUBTITLE
NAVAL COMBAT SYSTEMS PRODUCT LINE ECONOMICS: EXTENDING THE CONSTRUCTIVE PRODUCT LINE INVESTMENT MODEL FOR THE AEGIS COMBAT SYSTEM

5. FUNDING NUMBERS

6. AUTHOR(S) Kyle A. Chance

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Naval Postgraduate School
Monterey, CA 93943-5000

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)
N/A

10. SPONSORING / MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

12a. DISTRIBUTION / AVAILABILITY STATEMENT
Approved for public release. Distribution is unlimited.

12b. DISTRIBUTION CODE
A

13. ABSTRACT (maximum 200 words)

Navy combat systems are ship class dependent and, until recently, were acquired as stovepipes. The disaggregated stovepipe acquisition method leads to suboptimal designs and exorbitant costs throughout the system's life cycle. A product line approach could reduce costs, increase mission effectiveness, and enable more rapid deployment across the Navy and rest of the Department of Defense. Existing software product line cost models are oversimplified to model per-product characteristics and savings within a product line across the life cycle of a system. Improving these existing cost models will better support decision making for future acquisitions. By applying existing research and leveraging the Constructive Product Line Investment Model, this work estimates product line savings and return on investment for the AEGIS combat system product. The AEGIS common source library is a proven standard for an evolving product line architecture to meet Navy combat systems requirements and has proven cost savings since its inception. This research provides a methodology and cost model framework for product line decisions while extending it for the AEGIS combat system case study.

14. SUBJECT TERMS
systems engineering (SE), model-based systems engineering (MBSE), product line engineering (PLE), software product line engineering (SPLE), product line, Constructive Product Line Investment model (COPLIMO), future combat systems, surface combatant, Aegis

15. NUMBER OF PAGES
65

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	**18. SECURITY CLASSIFICATION OF THIS PAGE**	**19. SECURITY CLASSIFICATION OF ABSTRACT**	**20. LIMITATION OF ABSTRACT**
Unclassified	Unclassified	Unclassified	UU

THIS PAGE INTENTIONALLY LEFT BLANK

Approved for public release. Distribution is unlimited.

NAVAL COMBAT SYSTEMS PRODUCT LINE ECONOMICS: EXTENDING THE CONSTRUCTIVE PRODUCT LINE INVESTMENT MODEL FOR THE AEGIS COMBAT SYSTEM

Kyle A. Chance
Lieutenant, United States Navy
BS, University of Florida, 2008

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN SYSTEMS ENGINEERING

from the

NAVAL POSTGRADUATE SCHOOL
June 2019

Approved by: John M. Green
Advisor

Raymond J. Madachy
Co-Advisor

Ronald E. Giachetti
Chair, Department of Systems Engineering

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

Navy combat systems are ship class dependent and, until recently, were acquired as stovepipes. The disaggregated stovepipe acquisition method leads to suboptimal designs and exorbitant costs throughout the system's life cycle. A product line approach could reduce costs, increase mission effectiveness, and enable more rapid deployment across the Navy and rest of the Department of Defense. Existing software product line cost models are oversimplified to model per-product characteristics and savings within a product line across the life cycle of a system. Improving these existing cost models will better support decision making for future acquisitions. By applying existing research and leveraging the Constructive Product Line Investment Model, this work estimates product line savings and return on investment for the AEGIS combat system product. The AEGIS common source library is a proven standard for an evolving product line architecture to meet Navy combat systems requirements and has proven cost savings since its inception. This research provides a methodology and cost model framework for product line decisions while extending it for the AEGIS combat system case study.

THIS PAGE INTENTIONALLY LEFT BLANK

TABLE OF CONTENTS

I. INTRODUCTION....1
A. BACKGROUND....1
B. RESEARCH QUESTIONS....2
C. SPECIFIC CONTRIBUTIONS....3
D. BENEFITS....3
E. ORGANIZATION....6

II. LITERATURE REVIEW....7
A. SOFTWARE PRODUCT LINE ENGINEERING....7
B. AEGIS COMBAT SYSTEM....11
C. PARAMETRIC COST MODELING FOR PRODUCT LINE ECONOMICS USING COPLIMO....15
D. SUMMARY....16

III. METHODOLOGY AND APPROACH....17
A. BASIC CONSTRUCTIVE PRODUCT LINE MODEL (BASIC COPLIMO)....18
B. PRODUCT LINE VERSUS ONE-OFF SOFTWARE SYSTEM....20
1. Inputs....20
2. Outputs....22
C. DETAILED COPLIMO....24
1. Inputs....24
2. Outputs....26
D. MODEL VERIFICATION AND VALIDATION....30
E. THREATS TO VALIDITY....31
F. SUMMARY....32

IV. CONCLUSION....35
A. RESEARCH SUMMARY....35
B. FUTURE WORK....36

LIST OF REFERENCES....39

INITIAL DISTRIBUTION LIST....43

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF FIGURES

Figure 1. Systems Engineering "V" Model. Source: Gregg, Albert, and Clements (2017). 4

Figure 2. Costs for Developing *n* Kinds of Systems (Single versus PLE). Source: Weiss and Lai (1999). 8

Figure 3. Aegis Combat System Block Diagram. Source: Threston (2009). 11

Figure 4. Transformation from Independent Programs to a Product Line Approach. Source: Gregg, Scharadin, and Clements (2015). 13

Figure 5. AEGIS Sea Platforms. Source: Gregg et al. (2014). 14

Figure 6. Layered View of the Aegis Product Line Software Architecture. Source: Gregg et al. (2014). 14

Figure 7. Non-Product Line Aegis Baseline Basic COPLIMO Inputs 22

Figure 8. Basic COPLIMO Output Summary and Results 23

Figure 9. Detailed COPLIMO Output Graph 30

Figure 10. ROI across Aegis Baselines 37

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF TABLES

Table 1. Aegis SLOC Data 2011–2014. 18

Table 2. Productivity Benchmarks by Operating Environment. Adapted from Clark and Madachy (2015). 24

Table 3. RUSE Factors. Source Boehm et al. (2004) 25

Table 4. DOCU Factors. Source Boehm et al. (2004). 26

Table 5. RELY Factors. Source Boehm et al. (2004). 26

Table 6. Detailed COPLIMO Table of Results....27

Table 7. Detailed COPLIMO Summary of Outputs 29

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF ACRONYMS AND ABBREVIATIONS

ACS	Aegis Combat System
ADAP	adapted code
AVPROD	average software productivity
AVSIZE	average product size
AWS	Aegis Weapon System
COCOMO	Constructive Cost Model
COPLIMO	Constructive Product Line Investment Model
CSEA	Combat System Engineering Agent
CSL	common source library
DOCU	degree of documentation
ESLOC	equivalent system lines of code
LM	Lockheed Martin
LOC	lines of code
MBSE	Model Based Systems Engineering
MV	maritime vehicle
OpEnv	operating environment
PEO IWS	Program Executive Office Integrated Weapon Systems
PLE	Product Line Engineering
PM	per person month
RCR	relative cost of reuse
RCRW	relative cost of writing for reuse
R&D	research & development
RELY	required software reliability
RUSE	reused code or development for reuse
ROI	return on investment
SLOC	system lines of code
SPLE	Software Product Line Engineering
SW	software
UNIQ	unique code

THIS PAGE INTENTIONALLY LEFT BLANK

EXECUTIVE SUMMARY

For the U.S. Navy, the Aegis combat system is the system able to integrate multi-purpose radars and advanced missile and gun systems. However, with warships of varying types and mission sets, variations among different versions of the Aegis combat system present themselves. To combat variation, system complexity, and reduce cost, the U.S. Navy in partnership with Lockheed Martin has adopted a software product line approach to managing, maintaining and developing the current and future capabilities within the Aegis Combat System, which they have called the Aegis common source library.

The process to manage, maintain and update the highly complex Aegis common source library requires an intensive upfront investment and continuous software maintenance funding stream. This thesis creates a model to estimate the product line effort savings and returns on investment of the Aegis common source library. This is accomplished by extending the Constructive Product Line Investment Model (COPLIMO), utilizing Aegis baselines of varying software sizes as inputs. This extension to COPLIMO compensates for two limitations in the existing cost model. First, the extension model allows for inputs of varying sized products and is, therefore, not limited to products of only one size. Secondly, the extension models different data compositions for each individual baseline for unique, adapted and reused code.

There are two fundamental engineering challenges associated with operating and maintaining a naval ship combat system for several decades. The first is to design an affordable system capable of achieving such long life cycles. The second is designing a system capable of being relevant and lethal for many years to come. New classes of ships and variants among ship classes introduce additional engineering challenges to software design. The software product line process, adopted by Lockheed Martin for the Aegis common source library, serves to answer these challenging problems of providing affordable, multi-decade, and lethal combat systems suites. Software product lines accomplish these tasks by taking advantage of commonality across multiple variants to provide “impressive reductions in costs, faster delivery of mission capability, and improved quality” (Jones 2009, 1). Cost avoidance and return on investment numbers reported from

Lockheed Martin through the year 2014 are showing numbers that exceedingly beat expectations (Gregg, Scharadin, and Clements 2015, 310). In fact, from 2011 to 2014, Lockheed Martin reported a cumulative cost savings of $166 million for the Aegis common source library.

Parametric cost analysis performed on a pre-common source library Aegis baseline utilizing Basic COPLIMO shows promising returns and effort savings over the course of several products. The results also further justify why a company like Lockheed Martin would adopt a product line approach to developing, maintaining and delivering future Aegis baselines. The modeled results, for an Aegis baseline made up of 1.8 million system lines of code (SLOC), show a potential ROI of 3.88 after the seventh product is delivered. This is calculated in Basic COPLIMO from the outputs of product line effort savings and the initial product line investment. High future returns, like the output results in the models of this thesis, have led many industries to adopt a product line approach to software development and maintenance.

Software product line cost benefits appear to be extremely promising, and capturing these cost benefits is vital to the Department of Defense (DoD) and the acquisition process. Nolan (2009) tells us that "a good cost model is central to good decision-making and good project management" (255). Therefore, improving and validating current models like COPLIMO will better help DoD cost estimators with the challenging task of more accurately capturing the future costs of heavily software-based systems. In an attempt to provide an improved model, the author offers an extension to COPLIMO, referred to as Detailed COPLIMO. This paper utilizes the extension to model ROI and product line effort savings of five Aegis common source library baselines as well as one projected future Aegis baseline. The results of Detailed COPLIMO for the Aegis common source library are quite promising, and compare favorably to the returns Lockheed Martin has also noticed. Lockheed Martin refers to its cost avoidance specifically as a "superlinear" effect. Whereas, according to Gregg, Scharadin, and Clements (2015), this cost avoidance has exceeded traditional product line linear cost model predictions. In addition to product line effort savings and ROI, Detailed COPLIMO has been reformulated to output per product cost savings and cost avoidance. Results from Detailed COPLIMO show estimated ROI

multipliers of 3.54 for the fifth delivered Aegis baseline in the product line and 5.40 for a future Aegis baseline, with per product cost avoidance numbers varying from 21–31% after the delivery of the first product.

References

Boehm, Barry, A. Windsor Brown, Ray Madachy, and Ye Yang. 2004. "A Software Product Line Life Cycle Cost Estimation Model." *Proceedings of the 2004 International Symposium on Empirical Software Engineering*. 156–164. Redondo Beach, CA, USA: IEEE.

Gregg, Susan, Rick Scharadin, and Paul Clements. 2015. "The More You Do, the More You Save: The Superlinear Cost Avoidance Effect of Systems Product Line Engineering." *In Proceedings of the 19th International Conference on Software Product Line* – (SPLC '15), Nashville, TN, USA, 303–310.

Gregg, Susan, Rick Scharadin, Eric LeGore, and Paul Clements. 2014. "Lessons From AEGIS: Organizational and Governance Aspects of a Major Product Line in a Multi-Program Environment." *In Proceedings of the 18th International Software Product Line Conference* - Volume 1 (SPLC '14), New York, NY, USA, 264--273.

Jones, Lawrence. 1999. *Product Line Acquisition in the DoD: The Promise, The Challenges*. CMU/SEI-99-TN-011. Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University. https://apps.dtic.mil/dtic/tr/fulltext/u2/a373184.pdf

Nolan, Andy. 2009. "Building a Comprehensive Software Product Line Cost Model." *In Proceedings of the 14th International Software Product Line Conference*, San Francisco, CA, USA, 249—256.

THIS PAGE INTENTIONALLY LEFT BLANK

ACKNOWLEDGMENTS

I thank my advisors, John Green and Ray Madachy, for their time, patience and expertise in helping me navigate the thesis process. I will be forever grateful for their advice and course corrections helping me extend the product-line investment model. I also thank my wife for her continuous love and support throughout this process; I could not have done it without her.

THIS PAGE INTENTIONALLY LEFT BLANK

I. INTRODUCTION

A. BACKGROUND

Aegis is the U.S. Navy's most advanced shipboard anti-air warfare weapon system. Aegis plays a critical role in national defense by integrating various sensors and weapons systems to shield the fleet from inbound air and missile threats as well as providing maritime ballistic missile defense. Integrating these highly advanced modern sensors and weapon system hardware is an incredibly complex task requiring a multifaceted software support system. This software support system is perhaps the most critical portion of maintaining a competitive edge over our adversaries. The shift from the U.S. Navy having the most superior firepower and sensor suites to being more evenly matched has been well documented over the last few decades. The Chief of Naval Operation's "A Design for Maintaining Maritime Superiority version 2.0" says that in today's new security environment, "Our competitive advantage has shrunk and in some areas, is gone altogether" (Richardson 2018). The sheer amount of data being transferred within the combat systems suite only exacerbates the importance of integration and the quality of the software driving the hardware.

The U.S. Navy's answer to this challenging integration problem has resulted in the adoption of a software product line approach to managing its complex Aegis combat systems software. The new open architecture framework, more commonly known as the Aegis common source library (CSL), has laid the foundation for an agile, robust and revolutionary combat systems suite that can be rapidly integrated into multiple U.S. and foreign navy ship classes for a wide variety of mission sets. This product line supported combat system allows the U.S. Navy to modernize sensors and upgrade weapon systems more rapidly due to an ease in software and hardware integration burdens. Furthermore, there is an additional benefit of a software product line. This benefit is a more common look and feel among the Aegis platforms from a warfighter perspective. These common tactical operations decrease watch stander training time and increase their proficiency, ultimately leading to an increase in the lethality of a distributed combat systems network.

The process to manage, maintain and update this highly complex software code requires an intensive upfront investment and continuous software maintenance funding stream. This thesis aims to identify the specific returns on investment (ROI) of adopting a product line software architecture versus building a one-off software system design by analyzing the system lines of code (SLOC) of the Aegis combat system both prior to the adoption of a common source library and after. Utilizing and offering an extension to the Constructive Product Line Investment Model (COPLIMO) achieves these objectives.

B. RESEARCH QUESTIONS

1. What are the economic returns of a naval combat system software product line versus one-off naval combat system software system design?

2. How can current product line cost models be improved for software-intensive combat systems?

To address the first question, the author analyzes the product line effort savings from a combat system software suite utilizing Basic COPLIMO. The pre CSL Aegis baseline size will serve as the benchmark for the initial cost benefit analysis of a software product line. Additionally, the model outputs from Basic COPLIMO will further highlight the economic benefits of adopting a software product line over a seven-year period by calculating product line effort savings as well as ROI.

This last question is addressed by considering empirical software data from the Aegis program executive office for model calibration and ROI calculation. For better insight and detailed coverage of the Aegis product baselines, COPLIMO was extended to accept heterogeneous size inputs where the SLOC for each product varies as well as the relative portions of mission-unique, adapted, and reused software. The actual SLOC for each category are inputs. This improves upon the basic model assumption whereby each product is homogeneous in size and makeup.

C. SPECIFIC CONTRIBUTIONS

The modeling methods used in this thesis are accepted systems engineering techniques. However, utilizing empirical Aegis SLOC data, the discussion of the software product line engineering (SPLE) processes and offering a COPLIMO extension is a distinct contribution to this subject matter and will help validate and improve the COPLIMO family of product line models. Furthermore, adapting and developing a better cost model for specific DoD programs will aid decision makers in the acquisition process of future Naval combat system suites and other software intensive systems.

D. BENEFITS

Product line approaches may reduce overall program acquisition costs, increase mission effectiveness and enable more rapid capability deployment across the U.S. Navy and DoD at large. A primary contribution is the integration of parametric cost modeling within Model-based Systems Engineering (MBSE) for economic tradeoff analysis of system product lines. Improving and validating current models such as COPLIMO will greatly add to the field of systems engineering and better help future cost estimators with the challenging task of more accurately capturing the future costs of heavily software-based systems. Therefore, in order to accurately adapt cost models, the author must discuss where these savings can be obtained. The savings from adopting a product line approach traditionally grow from leveraging commonality; however, Lockheed Martin is also reporting savings across the entire spectrum of the systems engineering process.

Product line investment returns "accrue from reusing common pieces in different systems/products that share features. Furthermore, systems can be fielded faster leading to an increased overall mission effectiveness" (Boehm et al. 2013, 64) and a greater decision space for our commanders. "These benefits occur because previously built components reduce the effort and enable more rapid development" (Boehm et al. 2013, 64). Designers are then freed to shift focus and better allocate resources resulting in more rapid capability development and delivery. The added benefit of developing and delivering a needed capability at a much faster rate will give the U.S. Navy the adaptability and flexibility it so greatly needs.

Product line savings originate from multiple aspects of the systems engineering process. The systems engineering "V" in Figure 1 highlights the areas from which savings can be obtained.

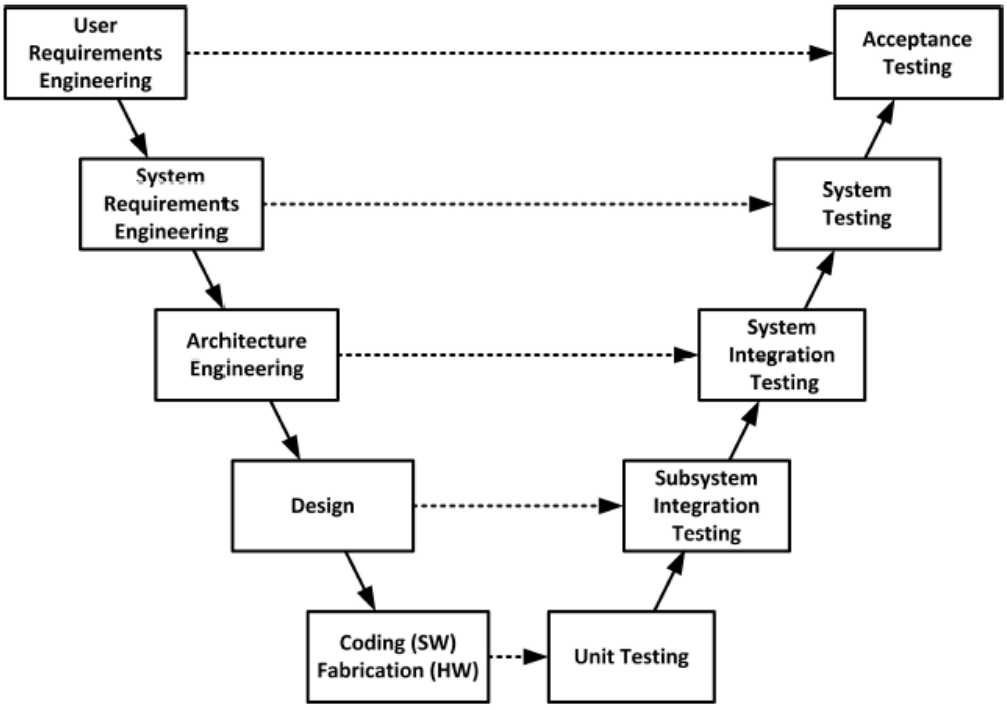


Figure 1. Systems Engineering "V" Model. Source: Gregg, Albert, and Clements (2017).

Traditional software product line engineering models capture savings from requirements engineering, architecture engineering, design and coding fabrication; however, Gregg, Albert, and Clements (2017) highlight the additional savings from the verification and validation phases of development in their article titled "Product Line Engineering on the Right Side of the 'V'." Testing for the Aegis combat system is a huge expense for the U.S. Navy, as the system is expected to work when called upon. Savings are captured by only having to test a piece of code once, because that same code is injected into many new baselines. Additionally, maintaining a high software quality can be costly. Gregg,

Scharadin, LeGore and Clements (2014) have reported excellent quantitative results of the Aegis CSL over the period of 2011–2013.

> Eliminating the need to fix a defect in multiple libraries provided substantial savings to the various government program offices (Aegis, LCS, FMS, MDA, Coast Guard). The combined savings of product line versus clone and own has totaled in excess of $80 million over the past 3 years. (271).

Their motto of "Fix it Once" has clearly paid dividends to the U.S. Navy but has also freed up resources within the Lockheed Martin organization. Requirements defects have also yielded combined government agency savings of $39 million over the past three years as well. In addition to these cost savings, Aegis CSL has removed additional integration test costs due to the fact these tests now only span one program instead of across multiple programs. They report these additional tests added 40% to the initial fix cost (Gregg, Scharadin, LeGore and Clements 2014, 271).

In 2015, Gregg, Scharadin and Clements reported even greater economic benefits as a result of the Aegis CSL product line approach. They describe the behavior as "superlinear," which they define as cost avoidance that "exceeds the cost avoidance predicted by the linear cost models" (303). Lockheed Martin's initial estimate of cost avoidance was a yearly value of 35% (309), but it reported greater than double that number. Over the span of 2011–2013, the Aegis Weapon System product line reported a cumulative cost avoidance of $119 million, or roughly $40 million per year, which matches previous reported returns. For 2014 alone, however, Lockheed Martin saw a $47 million cumulative cost avoidance or a 118% savings above the norm (Gregg, Scharadin, and Clements 2015). In total, for the years 2011–2014, that is $166 million in total cost avoidance.

Understanding the potential product line savings allows the author to better refine the cost models presented in this thesis. The cost models presented will also provide an easy-to-use framework for analyzing per product cost avoidance to allow for comparison against real-world data. Fiscally constrained environments, like the ones the U.S. Navy are currently operating in, require more scrutiny and oversight as to where its dollars are being spent. Accurate cost models can help program managers make better decisions about

about total program costs, as they will be able to improve upon forecasted expenditures from the development of a system to its inherent disposal. Additionally, other specific and realized software product line benefits will be addressed throughout this thesis.

Cost models also explain that not all attempts "at product line reuse will generate large savings" (Boehm et al. 2004, 163). Boehm, Brown, Madachy and Yang (2004) argue that "a good deal of domain engineering needs to be done well in order to identify software code most likely to be product-specific, fully reusable, or reusable with adaptation" (2004, 163). This means that there will be added cost in the development of a software product line and subsequent architectures must be thought of and developed prior to specific code writing. Cost models like COPLIMO will also help evaluate the tradeoffs of different architectural options and determine when product line approaches are justified. Ultimately, according to Nolan, "a good cost model is central to good decision-making and good project management" (Nolan 2009, 255).

E. ORGANIZATION

This thesis is organized into three main segments that address a literature review, methodology and approach, and conclusions, including future work. Chapters I and II give the background and benefits of adopting a product line software architecture. A review of the evolution of the Aegis combat system software introduces the reader to a proven software product line model that delivers rapid and robust capability to the U.S. Navy as well as select foreign navies. This review also sets the stage for Chapter III, the parametric cost modeling of a software product line as well as the introduction to Detailed COPLIMO.

Chapter IV concludes the material presented by offering a summary of the work and offering suggestions for future research topics. Current cost estimation models must continue to be refined in an effort to provide more accurate cost information to stakeholders, decision-makers and acquisition professionals. The thesis is organized to present the reader with a logical transition between SPLE, the proven software product line Aegis CSL, and the robust product line model, COPLIMO.

II. LITERATURE REVIEW

In this chapter, a comprehensive literature review is conducted describing the concepts of SPLE, Return on Investment (ROI) and the COPLIMO product line model. The literature review will also provide a brief history of the Aegis combat system for context and introduce the reader to a successful DoD software product line titled the Aegis common source library (CSL).

A. SOFTWARE PRODUCT LINE ENGINEERING

Product line engineering is a proven industry process that, when applied to software development, has revolutionized the way companies and families of products are developed and architected. Northrop, of Software Engineering Institute, defines software product lines as "a set of software-intensive systems that share a common, managed feature set satisfying a particular market segment's specific needs or mission and that are developed from a common set of core assets in a prescribed way" (Northrop 2002, 32). Therefore, the discipline and technical rigor devoted to maintaining and developing software product lines would be SPLE.

Understanding SPLE requires an understanding of how society influences industry. Bosch highlights the consumer's "need for speed" in regards to product development and states, "customer adoption of new products and new functionality in existing products is increasingly rapid" (Bosch 2017, 26). He further explains the ever-increasing size of software and uses automotive software as an example. History has told us that a "1 million lines of code (LOC) system in 2005 will be a 10 million LOC system in 2010 and a 100 million LOC system in 2015" (Bosch 2017, 25). SPLE is the domain that attempts to solve the challenging problem of delivering robust and high-quality software systems at a pace that matches customer desire. In addition to attempting to solve this conundrum, companies that have implemented software product lines reported "impressive reductions in costs, faster delivery of mission capability, and improved quality" (Jones 2009, 1). Cost reductions are not immediate and typically, initial investments take time before offering any return on investment. Historical data indicates that families of systems do require a

higher up-front investment when compared to single system design but do over time reach a break-even point as modeled in Figure 2 from Weiss and Lai.

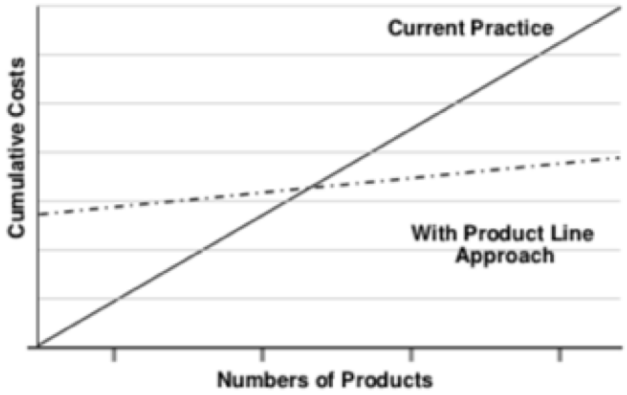


Figure 2. Costs for Developing *n* Kinds of Systems (Single versus PLE). Source: Weiss and Lai (1999).

SPLE also promises additional organizational benefits as highlighted in Donohoe's 2014 presentation on the "Introduction to Software Product Lines." These benefits allow organizations to:

- Achieve large-scale productivity gains
- Improve the time to market
- Achieve greater market agility
- Improve product quality
- Increase predictability of cost, schedule, and quality (Donohoe, 2014, Slide 8)

Accurately estimating the break-even point is an important piece of information for companies to know. Evaluating and being able to predict when the company sees a profit can drastically impact future product development and the overall decision to make a certain product or system or not. DoD programs are also subject to similar scrutiny as they are all measured by their overall cost, schedule of delivery and performance of the system. Specific modeling tools that make better and more accurate predictions on future costs like

those explored in this thesis will be shown in Chapter III, the methodology and approach chapter.

Shifting the focus back to product lines and supporting software architecture begs one fundamental question. If the goal of product lines is to produce "a family of products designed to take advantage of (their) common aspects and predicted variabilities" (Weiss and Lai 1999, 5), then how do companies also adopt this mindset to software development in support of these common product lines? The answer to this question is not as simple as one might think and, according to Pohl, Böckle, and van der Linden (2005), requires, at a minimum, three main prerequisites:

1. The enabling technologies: these consist of implementation technologies, object-oriented programming, component technology, binding techniques, middleware, and configuration management

2. The process maturity: these include process models, requirements engineering and modeling techniques

3. The domain characteristics and expertise: these consist of domain knowledge and domain stability. (16-18)

These prerequisites set the foundation on which to build and implement a software product line and require the systematic and disciplined approach of SPLE. The SPLE discipline is the "paradigm to develop software applications (software-intensive systems and software products) using platforms and mass customization" (Pohl, Böckle, and van der Linden 2005, 14). SPLE offers some distinct advantages over single system software designs, which include propagation of error corrections, organized evolution and easing the burden of complexity as SLOC increases. More importantly though, SPLE simplifies cost estimation because "calculating prices for products realised within the product line is relatively straightforward and does not include much risk" (Pohl, Böckle, and van der Linden 2005, 12).

There are three specific types of software code utilized in the cost estimation models within this thesis. Those are unique, adapted and reused code. Unique code is

simply newly developed code and requires the highest amount of effort to develop. Adapted code is code that only requires slight modification from one product line variant to another and therefore requires less effort than developing unique code. Lastly, reused code is code that can be transposed from one product to another. Although reused code by definition is not modified, there is still effort associated with the transfer of code from one product to another. The author ties effort-to-software code type because the models examined in this thesis calculate effort savings based upon the specific code type. Chapter III of this thesis will discuss the effort to code type relationship in greater detail.

The specific code analyzed in this thesis is the SLOC size of different Aegis baselines of both pre- and post-common source library adoption. Lockheed Martin is the company contracted to maintain the common source library for the U.S. Navy's Aegis combat systems suite. Utilizing the SPLE discipline, Lockheed Martin has revolutionized the way the Navy maintains and injects new capability into a warship's combat system.

Ultimately, if the goal for the Navy is to become more agile and flexible to the world's ever-changing threat environment, it is then obvious that SPLE is a worthwhile domain to explore and develop. Pohl, Böckle and van der Linden prove that over time there is improved quality through the reuse of software code. In fact, they argue, "The artefacts in the platform are reviewed and tested in many products. They have to prove their proper functioning in more than one kind of product. The extensive quality assurance implies a significantly higher chance of detecting faults and correcting them, thereby increasing the quality of all products" (Pohl, Böckle, and van der Linden 2005, 10). The byproduct of many rounds of testing and review is a higher quality software product. This is a critical requirement for any combat system suite that is expected to perform without fail. Pohl, Böckle and van der Linden go on to discuss how many of today's systems of even minimal complexity contain software. Much like with the U.S. Navy's combat suite, these systems, "are becoming software-intensive systems, not only because variability can be implemented more flexibly than in pure hardware, but also because of the fact that software allows the introduction of new functionality that could not easily be achieved without it" (Pohl, Böckle, and van der Linden 2005, 14).

B. AEGIS COMBAT SYSTEM

Designed to be built around the Aegis Weapon System (AWS), where the entire ship becomes a "single fully integrated weapon of war" (Threston 2009, 109), the Aegis combat system was first deployed on the cruiser USS Ticonderoga (CG-47) in 1981 and was "comprised of over 850 individual equipment elements and weighed over 600 tons. It also included all of the officers and sailors needed to operate the system" (Threston 2009, 115). Critical components of the Aegis Weapon System are shown in the Figure 3 block diagram, which highlights critical systems and their linked architecture.

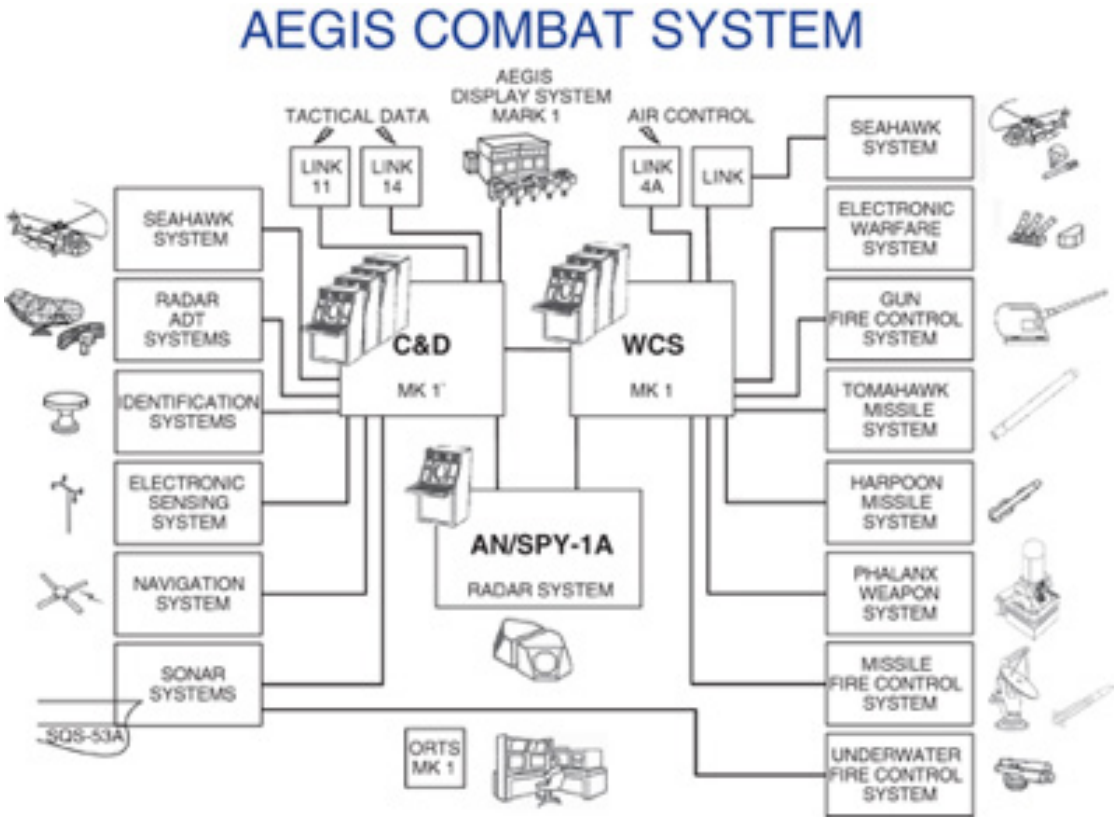


Figure 3. Aegis Combat System Block Diagram.
Source: Threston (2009).

Arranged from left to right, Figure 3 follows the three traditional naval warfare functions of detection, control and engage. Threston defines each as

- *Detection*—The presence of potential threats must be detected and their location, course, and speed determined.

- *Control*—The detected threat must be identified, and if it is determined to be hostile, a command decision to engage it (and with what weapon) must be made.

- *Engage*—The assigned weapon must engage the enemy, the outcome of which will determine the next course of action. (Threston, 2009, p. 111)

Initial development of the combat system resulted in obvious interface issues with other sensors and weapons systems where each interface had to be managed individually. The obvious complexity of integrating all of the required sensors and weapon systems led to the ship-specific or combat system-specific stovepipe approach. Hall's thesis in 2018 highlighted the benefits for the U.S. Navy to better architect its combat system by utilizing the fundamentals of product line engineering. This thesis attempts to further emphasize the benefits of such an approach by capturing the economic benefit associated with adopting a product line approach to software coding, development and integration. As discussed previously, many of the benefits associated with a product line architecture are also captured in a software product line. The U.S. Navy, alongside Lockheed Martin, realized these benefits and began to adopt the software product line mindset for the continued refinement and modernization of the Aegis combat system, and called it the Aegis common source library. The transformation process from the one-off combat system design to a product line approach for Aegis can be seen in Figure 4.

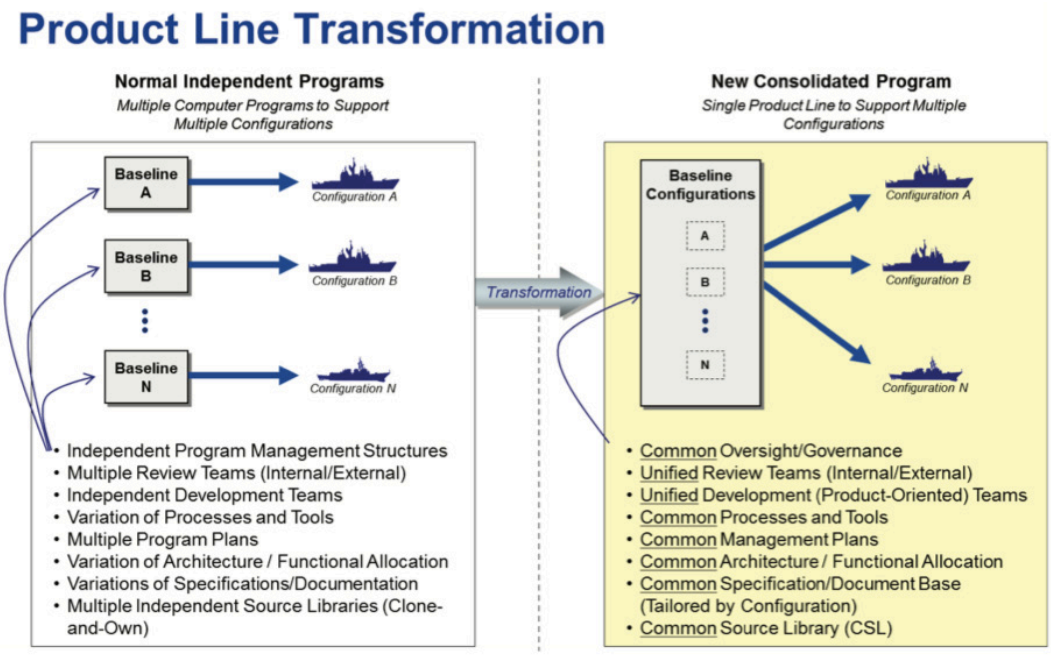


Figure 4. Transformation from Independent Programs to a Product Line Approach. Source: Gregg, Scharadin, and Clements (2015).

The term baseline is given to different versions of the Aegis combat system much like a software company would assign a number to different versions of their operating systems. Aegis baseline 9 was the first baseline of its kind to evolve from the common source library where "the primary objective of the CSL approach is to develop once and build and deploy many times from one set of common source code" (Gregg, Albert, and Clements 2017, 166). The idea of building and developing a baseline only once, having to only maintain and fix one library of code, and being able to then use this code on many different platforms and ship types revolutionized the way the U.S. Navy built and maintained its combat system suite software. The CSL now enables the U.S. Navy to develop common mission capabilities from single sets of specifications and apply those capabilities across a multitude of U.S. Navy, U.S. Coast Guard and international ship classes, as illustrated in Figure 5.

Figure 5. AEGIS Sea Platforms. Source: Gregg et al. (2014).

However, in order for the CSL model to be successful, an open architecture must be adopted where hardware and software could be updated independently by componentizing the systems and building software that was able to integrate system capability rather than specific systems themselves. This layered approach can be seen in Figure 6, which highlights the importance of architecture in software product lines.

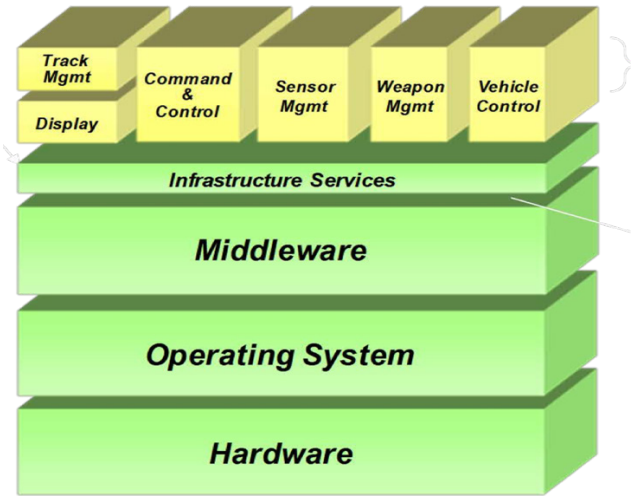


Figure 6. Layered View of the Aegis Product Line Software Architecture. Source: Gregg et al. (2014).

This paper will examine multiple versions of Aegis baseline 9 that span deliveries from the years 2011 through 2014. The goal will be to produce a model that will help decision makers better estimate the costs associated with large and high visibility software systems. This effort is extremely important to the Navy and the DoD. As the Navy "has an active effort under way to expand the product line approach to the entire surface combat fleet" (Gregg et al. 2014, 273).

C. PARAMETRIC COST MODELING FOR PRODUCT LINE ECONOMICS USING COPLIMO

This paper will utilize the Constructive Product Line Investment Model or COPLIMO to calculate the expected return on investment of the U.S. Navy's Aegis combat systems software product line. But first, it is important to understand the importance of cost models to software product lines. According to Nolan (2009), "A model is a formal and objective representation of a project or business and is by definition a simplification of reality," moreover, "a model is there to help you make a decision but not make the decision for you" (250). How we model and in what way can be specific to the application of a system. In the case of this thesis, the author attempts to offer and extension to a proven software product line cost model. The factors that make up the model are extremely important and must be objective and measurable. As Nolan (2009) explains, "if you can't measure it you can't control it, but if you can't estimate it then you don't understand what is happening" (255).

COPLIMO is a model based upon the well-calibrated, multi-parameter Constructive Cost Model (COCOMO) II for software development and is designed to help calculate costs, savings and ROI of software product lines (Boehm, Brown, Madachy, and Yang 2004). To expand upon COCOMO II, Boehm, Brown, Madachy and Yang show that software product line investment models should also include the relative cost of writing for reuse (RCWR) as well as the relative cost of reuse (RCR). Where RCWR is the cost associated for writing code with the intent purpose and knowledge the code will be reused for future product line variants and, therefore, will require a greater degree of effort and higher cost in development. RCR on the other hand, not only factors in the "amount of effort required to modify existing software" (Boehm, Brown, Madachy,

and Yang 2004, 158), but must also include the effort associated with the coder understanding and becoming familiar with the software code.

The primary indicator required for analysis and comparison of the product line economics in this thesis is the return on investment or ROI. COPLIMO calculates ROI by dividing the product line effort net savings by the original product line reuse investment. ROI is the ratio of the cumulative net savings associated with the reduction of effort due to a product line and the initial investment to architect a product line.

D. SUMMARY

The product line approach to developing and distributing software is perhaps the greatest way to maintain and modernize the U.S. Navy's combat system suites. Lockheed Martin has proven this model through the Aegis common source library and the DoD at large should continue to look for ways to integrate SPLE and the product line approach into many other defense programs. The cost and effort savings have been critical in recent budgetary constrained environments and those savings will allow other resources to pour back into research and development (R&D) and fleet maintenance.

Refining and validating cost models to maintain simplicity at the highest level but also include many cost variables that can often over complicate estimates is of vital national importance. The goal of this paper is to use the data collected from one of the DoD's largest software product line success stories and refine, validate and extend the proven software product line COPLIMO.

III. METHODOLOGY AND APPROACH

The ROI of a properly architected product line has been proven many times over (Brownsword and Clements 1996). For this thesis, specific data from the Aegis combat system program office or Program Executive Office for Integrated Warfare Systems (PEO IWS) are modeled offering an extension to Basic COPLIMO. Distribution standards levied by the PEO IWS limit the public sharing of the actual SLOC data; however, in an effort to allow the reader an opportunity to examine the types of data entered into the model, Table 1 serves as a generalized view of the type of data that is used for extensive analysis in this thesis. Additional data collection methods were attempted. These attempts include requests to access Aegis software resources data reports (SRDR) and attempts to access to the Cost Assessment Data Enterprise (CADE) databases. However, due to time constraints, the author was forced to abandon these data collection methods and utilize SLOC data from the Aegis program office.

For analysis, the actual SLOC data is utilized and appropriate cost factors for the Aegis combat system are selected to run in the model. The results of the models are compared to reported cost avoidance numbers of the Aegis CSL from the years 2011 through 2014. These cost avoidance numbers from Lockheed Martin show extremely promising returns of the product line approach. According to a 2013 Naval Sea Systems Command release, Lockheed Martin was assigned as the prime contractor for Aegis Combat System Engineering Agent (CSEA) efforts and it "will evolve and maintain the Aegis Weapon System (AWS) and Aegis Combat System (ACS) for CG 47 class cruisers, DDG 51 class destroyers and possible future surface combatant ship classes" (NAVSEA 2013, para. 3). In fact, in 2013, Lockheed Martin was awarded $100,685,094 for a contract for their Aegis CSEA efforts (NAVSEA 2013).

In this Chapter, the author presents a model of a non-product line Aegis baseline to show potential product line effort savings and return on investment. This serves to enforce the economic benefits of a product line approach to a software system of a similar size. Following the single non-product line baseline model, the author offers an extension to Basic COPLIMO in an effort to model and capture the product line savings and ROI of the

existing Aegis CSL product line as well as predict future product line effort savings and ROI for a future CSL baseline. The results of extension model, Detailed COPLIMO, are then compared to Lockheed Martin's reported cost avoidance numbers from 2011–2014.

Table 1. Aegis SLOC Data 2011–2014.

Baseline	ESLOC	Unique	UNIQ %	Adapted	ADAP %	Reused	RUSE %
0	1,800,000	N/A	N/A	N/A	N/A	N/A	N/A
A	1,900,000	100,000	5.56%	540,000	30.00%	1,260,000	66.32%
B	2,100,000	200,000	10.53%	570,000	30.00%	1,330,000	63.33%
C	2,500,000	400,000	19.05%	630,000	30.00%	1,470,000	58.80%
D	2,700,000	200,000	8.00%	750,000	30.00%	1,750,000	64.81%
E	3,100,000	400,000	14.81%	810,000	30.00%	1,890,000	60.97%
F*	3,300,000	200,000	6.45%	930,000	30.00%	2,170,000	65.76%
Average	2,350,000						

Adapted from PEO IWS, unpublished data, March 01, 2018.

Table 1 data represents a generalized view of the distribution limited Aegis SLOC data obtained from PEO IWS. Baselines A through E are representative of versions within Aegis baseline 9 in the Aegis CSL and span the timeframes of 2011 through 2014. These baselines are in no way specifically tied to actual delivered Aegis builds or ship classes. They merely serve as representations of product line variants of the Aegis CSL. Baseline F is representative of a future baseline, Aegis baseline 10, and the projected size is based upon PEO IWS's projected SLOC size for this baseline.

A. BASIC CONSTRUCTIVE PRODUCT LINE MODEL (BASIC COPLIMO)

As discussed previously, Basic COPLIMO is the cost model manipulated in this thesis. The author then utilized Excel as the spreadsheet tool for implementation of the Constructive Product Line Model. Inputs for Basic COPLIMO are then entered directly into the spreadsheet tool. The first input value is average software productivity called AVPROD. AVPROD is simply the SLOC written in a per person month (SLOC/PM). The next input is titled average product size (AVSIZE). One basic assumption for the Basic COPLIMO is that all variants analyzed are of similar product size. Proceeding after AVSIZE are the expected reuse category percentages values. The first code type percentage

utilized in the model is the unique code percentage (UNIQ%). The next code type utilized is the percentage of adapted code (ADAP%). Finally, the percentage reused or RUSE% is auto calculated in the spreadsheet tool for the user to ensure all three code types are fully accounted for in the total SLOC.

The next inputs for the model are for the expected relative costs of reuse (RCR). According to Boehm, Brown, Madachy and Yang, RCR represents the percentage of cost to reuse "the software in a new product line family application relative to developing newly-built software" (Boehm et al. 2004, 156). The logical cost of unique code or RCR-UNIQ in the model spreadsheet tool is set at 100% due to the nature of newly created and architected code; however, in reality this percentage can in fact be over 100%. This is due to the cost of developing new code in a software product line can exceed non-product line software development (Clark and Madachy 2015). For the sake of simplicity, however, the analysis completed in this thesis utilizes 100% for the value for RCR-UNIQ. The model creators list the two definitions for the RCR-ADAP and RCR-RUSE. For RCR-ADAP, they define this as the "percentage of cost to reuse the software with modifications in a new product line family application relative to developing newly-built software for the application" (Boehm et al. 2004). For RCR-RUSE, the creators define this as "the percentage of cost to reuse the software without modifications in a new product line family application relative to developing newly-built software for the application" (Boehm et al. 2004).

The last input value is titled the expected cost of writing for reuse or RCWR. Boehm, Brown, Madachy, and Yang provide an excellent definition of RCWR as "the added cost of writing software to be most cost-effectively reused across a product line family of applications, relative to the cost of writing the software to be used in a single application (Boehm et al. 2004, 156).

To begin using COPLIMO and to propose an extension of the model, the author of this paper recommends conducting the following steps prior to using or modifying the model:

1. Collect and organize SLOC data

2. Calculate unique, adapted & reused SLOC and respective percentages

3. Determine relative costs of reuse for each code type

4. Determine and calculate relative costs of writing for reuse

Upon entering the required inputs into COPLIMO spreadsheet tool, the user can then navigate to the output tab to view the model results. The outputs consist of three main areas of focus: the summary of the inputs, a graph on the seven-year product line effort savings, and a table of results. The summary of inputs offers the user a snapshot of the data used for output calculations. The seven-year product line effort savings graphically allows the reader to visualize the breakeven point of the modeled product line. Finally, the table of results holds the data calculated by the model and is organized in a logical and mathematical fashion.

B. PRODUCT LINE VERSUS ONE-OFF SOFTWARE SYSTEM

The purpose of this section is to identify the economic benefits of a software product line versus a one-off software system design. The specific software system examined in this model is a known Aegis combat system of non-product line decent, which is referred to in Table 1 as Baseline 0. Baseline 0 is an approximate size of a pre-common source library Aegis baseline. The modeling software used for analysis in this paper is a spreadsheet tool adapted from Basic COPLIMO. To show the reader exact inputs and outputs of the COPLIMO model, the generalized size of baseline 0 is used. A summary of inputs is explained next, followed by a summary of outputs.

1. Inputs

The recommended values for Basic COPLIMO are utilized as inputs for comparing a product line software system versus a one-off software system design. The following list shows the inputs utilized for this model as well as the definition of each and the justification in selecting these values. These inputs in actuality represent a range of values. However, for simplicity, the calculations are completed with specific values.

1. AVPROD: estimated number of SLOC produced by the developer in per person-month (PM). 150 was selected and is recommended by the model creators for critical real-time control applications.

2. AVSIZE: estimated number of SLOC across the product line. 1,800,000 SLOC selected, as it is the generalized size of a non-product line combat system suite from the Aegis SLOC data.

3. UNIQ%: estimated percentage of unique code. 10% selected, as it represents the average UNIQ% across the Aegis CSL for this thesis.

4. ADAP%: estimated percentage of adapted code. 30% selected and is applied to all Aegis baselines in this thesis.

5. RUSE%: estimated percentage of reused code. 60% selected and represents the average RUSE% across the Aegis CSL for this thesis.

6. RCR-UNIQ: the percentage of "the cost of reusing the software in a new product line family application relative to developing newly-built software" (Boehm et al. 2004). 100% selected, as it is the COPLIMO recommended value.

7. RCR-ADAP: "percentage of cost to reuse the software with modifications in a new product line family application relative to developing newly-built software for the application" (Boehm et al. 2004). 40% selected, as it is the COPLIMO recommended value.

8. RCR-RUSE: "the percentage of cost to reuse the software without modifications in a new product line family application relative to developing newly-built software for the application" (Boehm et al. 2004). 5% selected, as it is the COPLIMO recommended value.

9. RCRW: "the added cost of writing software to be most cost-effectively reused across a product line family of applications, relative to the cost of

writing the software to be used in a single application (Boehm et al. 2004). 1.85 selected, as it is the COPLIMO recommended value.

Figure 7 shows the input values from the list above in the Basic COPLIMO spreadsheet tool.

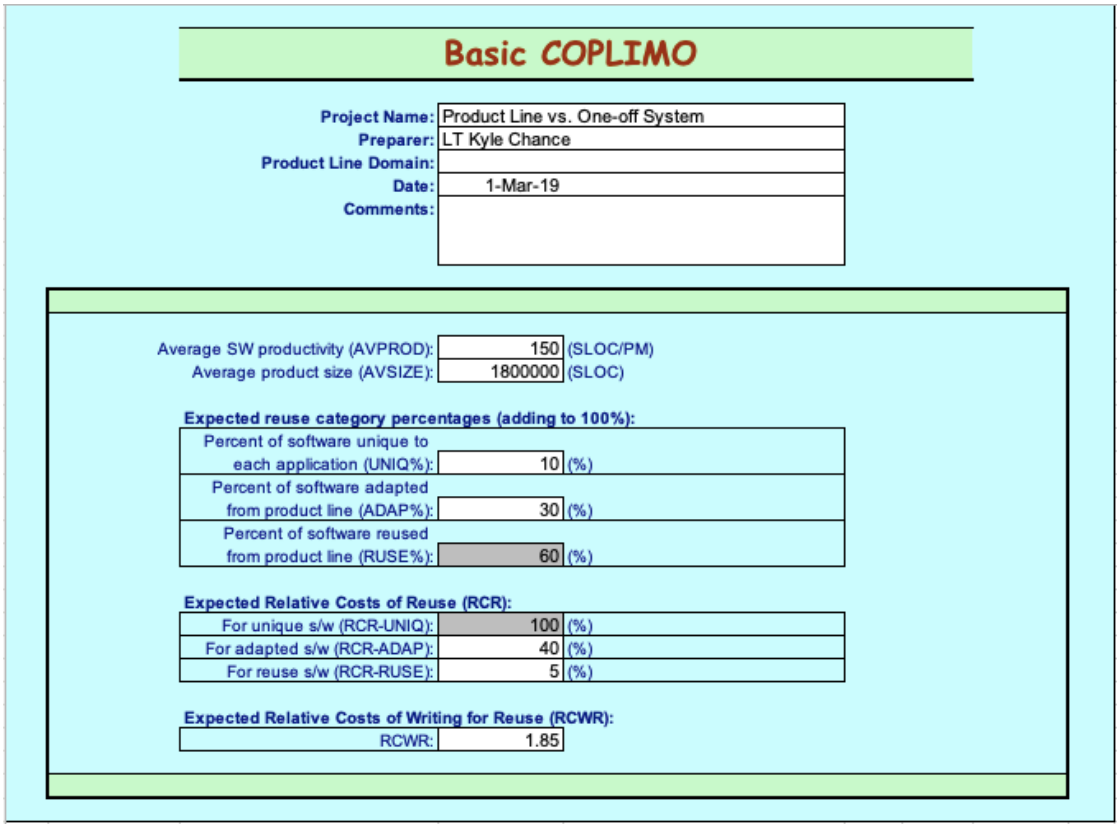
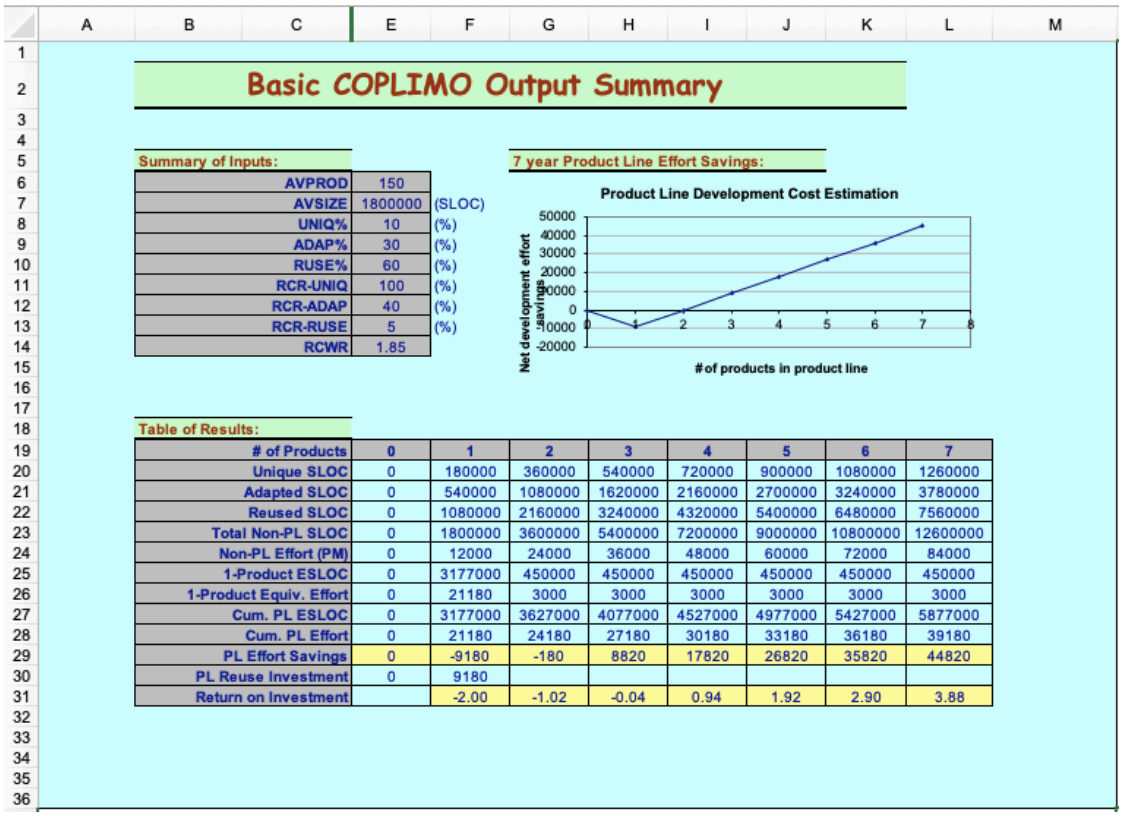


Figure 7. Non-Product Line Aegis Baseline Basic COPLIMO Inputs

2. Outputs

Figure 8 shows the outputs of the Basic COPLIMO spreadsheet tool, as well as the modeled results.



Basic COPLIMO Output Summary

Summary of Inputs:

AVPROD	150	
AVSIZE	1800000	(SLOC)
UNIQ%	10	(%)
ADAP%	30	(%)
RUSE%	60	(%)
RCR-UNIQ	100	(%)
RCR-ADAP	40	(%)
RCR-RUSE	5	(%)
RCWR	1.85	

7 year Product Line Effort Savings:

Table of Results:

# of Products	0	1	2	3	4	5	6	7
Unique SLOC	0	180000	360000	540000	720000	900000	1080000	1260000
Adapted SLOC	0	540000	1080000	1620000	2160000	2700000	3240000	3780000
Reused SLOC	0	1080000	2160000	3240000	4320000	5400000	6480000	7560000
Total Non-PL SLOC	0	1800000	3600000	5400000	7200000	9000000	10800000	12600000
Non-PL Effort (PM)	0	12000	24000	36000	48000	60000	72000	84000
1-Product ESLOC	0	3177000	450000	450000	450000	450000	450000	450000
1-Product Equiv. Effort	0	21180	3000	3000	3000	3000	3000	3000
Cum. PL ESLOC	0	3177000	3627000	4077000	4527000	4977000	5427000	5877000
Cum. PL Effort	0	21180	24180	27180	30180	33180	36180	39180
PL Effort Savings	0	-9180	-180	8820	17820	26820	35820	44820
PL Reuse Investment	0	9180						
Return on Investment		-2.00	-1.02	-0.04	0.94	1.92	2.90	3.88

Figure 8. Basic COPLIMO Output Summary and Results

Basic COPLIMO estimates the product line effort savings over the course of seven future products in a product line. The breakeven point in this model is just slightly after year two, when the effort line crosses the x-axis. After the three-year mark, the model shows significant ROI multiplier values of 0.94, 1.92, 2.90 and 3.88 for years four through seven. These large ROI values alone help make the case for why a defense contractor like Lockheed Martin would pursue a product line approach to managing the U.S. Navy's Aegis combat system software. Basic cost models like the one explored in this section serve to provide rough order of magnitude initial savings estimates. These initial estimates can then be used by decision-makers early in the acquisitions process to make more informed decisions. In the following section, the author examines empirical data and adapts the Basic COPLIMO model to fit the varying sized multi-baseline Aegis SLOC numbers.

C. DETAILED COPLIMO

Basic COPLIMO is designed to model products of the same size and relative composition of mission-unique, adapted and reused software. This would unrealistically model a total SLOC size that does not vary among the different Aegis baselines, and each would show identical economic returns. Because this thesis is utilizing empirical data for which the assumption does not apply, the author offers an extension to Basic COPLIMO to capture product line effort savings, as well as ROI across all of the Aegis baselines that vary in size and composition. The Detailed COPLIMO extension provides per product cost savings and avoidance in addition to cumulative product line effort savings and ROI.

1. Inputs

Some inputs from Basic COPLIMO remain the same; however, the three input values for specific percentages for UNIQ, ADAP and RUSE code will be left in respective SLOC sizes for each of the Aegis baselines. The rest of the input values are calculated specifically for Detailed COPLIMO and are derived as follows: First, the average software productivity or AVPROD recommended for critical real-time control applications is 150 SLOC/PM. However, for a better representation of the software system analyzed in this thesis, productivity statistics from Clark and Madachy (2015) are used. The Aegis combat system operating environment (OpEnv) is classified as a maritime vehicle (MV) and from Table 2, the mean productivity of an MV is 163 ESLOC/PM.

Table 2. Productivity Benchmarks by Operating Environment. Adapted from Clark and Madachy (2015).

		Productivity (ESLOC/PM)							
OpEnv	N	LCI	Mean	UCI	SE	SD	Q1	Mdn.	Q3
AV	80	118.9	137.6	156.3	9.4	84.1	77.7	117.6	172.0
GF	133	186.5	208.2	230.0	11.0	126.8	118.2	192.0	280.0
GV	26	110.2	140.6	171.0	14.8	75.3	89.9	121.0	169.5
MV	42	138.9	163.0	187.2	12.0	77.5	88.1	178.8	217.3
OV	25	82.8	122.5	162.1	19.2	96.2	59.1	95.4	161.4

The next assumption is for AVSIZE. COPLIMO requires a constant AVSIZE for analysis, and this value is calculated by taking the average of all SLOC data for each of the six baselines. This value found in Table 1 is 2,339,780 SLOC. For the extension offered in this paper, however, AVSIZE is not directly applicable to the output calculations. Nevertheless, AVSIZE is maintained in the extension model output to give the reader context on the average size of the Aegis baselines used for analysis.

Boehm, Brown, Madachy, and Yang outline the process for determining the weighted value for RCWR. RCWR is determined by calculating the product of the development for reuse (RUSE) times the degree of documentation (DOCU) times the required software reliability (RELY). Tables 3 through 5 from Boehm, Brown, Madachy and Yang show the descriptors, rating levels, and effort multipliers for each of the RCRW factors.

Table 3. RUSE Factors. Source Boehm et al. (2004).

RUSE Descriptors		None	Across project	Across program	Across product line	Across multiple product lines
Rating Levels	Very Low	Low	Nominal	High	Very High	Extra High
Effort Multipliers	N/A	0.95	1.00	1.07	1.15	1.24

Table 4. DOCU Factors. Source Boehm et al. (2004).

DOCU Descriptors:	Many life cycle needs unco-vered	Some life cycle needs unco-vered.	Right-sized to life cycle needs	Exce-ssive for life cycle needs	Very excessive for life cycle needs	
Rating Levels	Very Low	Low	Nominal	High	Very High	Extra High
Effort Multipliers	0.81	0.91	1.00	1.11	1.23	N/A

Table 5. RELY Factors. Source Boehm et al. (2004).

RELY Descriptors:	Slight incon-venie-nce	Low, easily recove-rable losses	Moderate, easily recover-able losses	High fina-ncial loss	Risk to hum-an life	
Rating Levels	Very Low	Low	Nominal	High	Very High	Extra High
Effort Multipliers	0.82	0.92	1.00	1.10	1.26	N/A

The RCWR value used in Detailed COPLIMO for this analysis is calculated by multiplying the "very high" value for RUSE of 1.15, the "very high" value for DOCU of 1.23, and the "very high" value for RELY of 1.26. This results in an RCRW value of 1.78.

2. Outputs

After the inputs are calculated and selected, Detailed COPLIMO is run for each Aegis baseline of varying software size. Again, distribution limits on actual baseline size are limited by PEO IWS; therefore, the actual SLOC size is hidden from the reader. For context, Table 6 shows the Detailed COPLIMO output table with generalized SLOC data from Table 1.

Table 6. Detailed COPLIMO Table of Results

# of Products	0	1	2	3	4	5	6
Aegis Baseline		A	B	C	D	E	F*
Unique SLOC	0	100000	200000	400000	200000	400000	200000
Adapted SLOC	0	540000	570000	630000	750000	810000	930000
Reused SLOC	0	1260000	1330000	1470000	1750000	1890000	2170000
Total Non-PL SLOC	0	1900000	4000000	6500000	9200000	12300000	15600000
Total Non-PL Effort (PM)	0	11656	24540	39877	56442	75460	95706
1-Product ESLOC	0	3304000	379000	379000	379000	379000	379000
1-Product Equiv. Effort (PM)	0	20270	2325	2325	2325	2325	2325
Cum. ESLOC	0	3304000	3683000	4062000	4441000	4820000	5199000
Cum. PL Effort (PM)	0	20270	22595	24920	27245	29571	31896
PL Effort Savings (PM)	0	-8613	1945	14957	29196	45890	63810
PL Reuse Investment	0	8613	0	0	0	0	0
Per Product Non PL Effort (PM)	0	11656	12883	15337	16564	19018	20245
Per Product PL Effort (PM)	0	2325	3034	4451	3604	5021	4175
Per Product Cost Savings (PM)	0	-8613	9850	10887	12960	13997	16071
Per Product Cost Avoidance	0	-173.89%	23.55%	29.02%	21.76%	26.40%	20.62%
Cum. ROI	0	-2.00	-0.86	0.41	1.91	3.54	5.40

Utilizing Figure 12 as a script, the author will explain the values and calculations for each of the baselines beginning with SLOC size. Understanding how the total SLOC size is broken down for each type of software code is critical for understanding the model. Equations 1 through 3 show the formulas used specifically in Detailed COPLIMO for deriving the presumed UNIQ, ADAP and RUSE SLOC sizes:

$$UNIQ = \left(New\ Baseline\ SLOC\ size\right) - \left(Old\ Baseline\ SLOC\ size\right) \quad (1)$$

$$ADAP = \left(Old\ Baseline\ SLOC\ Size\right) * 30\% \quad (2)$$

$$RUSE = \left(Total\ New\ Baseline\ SLOC\ size\right) - UNIQ - ADAP \quad (3)$$

The next value the model utilizes for capturing product line savings is the total non-product line SLOC size. This critical value represents the required size of an Aegis baseline that is developed outside of a product line approach or, as referred to in this research, a one-off software system design. The model then uses this value to calculate non-product line effort which ultimately is required in calculating the overall effort savings of a product line. The total non-product line SLOC size is calculated by summing the UNIQ, ADAP and RUSE code across each of the baselines. Equation 4 represents the total non-product line SLOC size.

$$\text{Total Non-PL Size} = \sum (\text{UNIQ} + \text{ADAP} + \text{RUSE}) \tag{4}$$

The next extension model calculation is for Non-product line effort. Basic COPLIMO uses Equation 5 to calculate Non-product line effort; however, the Detailed COPLIMO spreadsheet tool utilizes Equation 6 as AVSIZE is not directly applicable to the extension model:

$$Non-PL\ Effort = \left(Number\ of\ \text{Products} * AVSIZE\right) \div AVPROD \tag{5}$$

$$\text{Non-PL Effort} = \frac{\text{Total Non-PL SLOC}}{\text{AVPROD}} \tag{6}$$

The subsequent equations for the extension model are the same as in Basic COPLIMO and are shown in Equations 7–10:

$$\text{Cumulative Product Line ESLOC} = \frac{(UNIQ * RCR - UNIQ) + (ADAP * RCR - ADAP) + (RUSE * RCR - RUSE)}{100} \tag{7}$$

$$\text{Cumulative Product Line Effort} = \frac{\text{Cumulative Product Line ESLOC}}{\text{AVPROD}} \tag{8}$$

$$\text{Product Line Effort Savings} = \text{Non-PL Effort} - \text{Cum. PL Effort} \tag{9}$$

$$\text{ROI} = \frac{\text{PL Effort Savings} - \text{PL Reuse Investment}}{\text{PL Reuse Investment}} \tag{10}$$

The next set of equations for the extension model aid in calculating a per product cost savings percentage for each of the Aegis baselines. Per product cost savings is calculated from non-product line effort and product line effort for each baseline which are calculated as shown in Equations 11–13.

$$\text{Per Product Non PL Effort} = \frac{\text{Total Non PL SLOC}}{\text{AVPROD}} \quad (11)$$

$$\text{Per Product PL Effort} = \frac{\frac{(\text{UNIQ}*\text{RCR-UNIQ})+(\text{ADAP}*\text{RCR-ADAP})+(\text{RUSE}*\text{RCR-RUSE})}{100}}{\text{AVPROD}} \quad (12)$$

$$\text{Per Product Cost Savings} = 1 - \frac{(\text{Per Product Non PL Effort} - \text{Per Product PL Effort})}{\text{Per Product Non PL Effort}} \quad (13)$$

Finally, per product cost avoidance is calculated utilizing Equation 14.

$$\text{Per Product Cost Avoidance} = \frac{\text{Non PL ESLOC} - \text{PL ESLOC}}{\text{Non PL ESLOC}} \quad (14)$$

The results for the Detailed COPLIMO software model have been scrubbed of all actual SLOC data, and only the cumulative product line effort, product line effort savings, product line reuse investment, per product cost savings, per product cost avoidance, and cumulative ROI are shared. The output data is shown in Table 7, and product line effort savings is graphed in Figure 9.

Table 7. Detailed COPLIMO Summary of Outputs

Table of Results:						
Aegis Baseline	**A**	**B**	**C**	**D**	**E**	**F***
Cumulative PL Effort	20288	22264	24240	26217	28193	30169
PL Effort Savings	-8800	1954	15234	29727	46652	64748
PL Reuse Investment	8800	0	0	0	0	0
Per Product Cost Savings	-8800	9577	10492	12765	13661	15849
Per Product Cost Avoidance	-176.60%	24.77%	31.23%	22.49%	27.73%	21.04%
Cumulative ROI	-2.00	-0.91	0.28	1.73	3.28	5.08

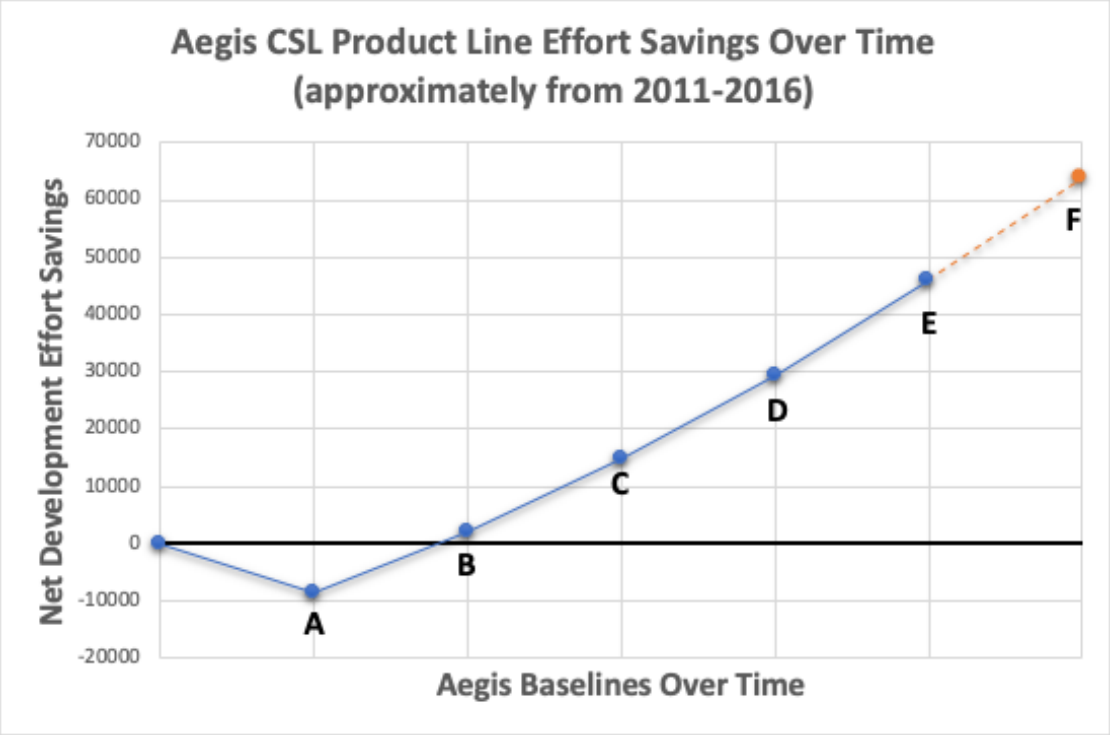


Figure 9. Detailed COPLIMO Output Graph

D. MODEL VERIFICATION AND VALIDATION

The purpose of this section is to both verify and validate the model and results. While verification and validation on their own are distinctly different, together they serve to provide confidence in the results of the model. Schlesinger defines model verification as the "substantiation that a computerized model represents a conceptual model within specified limits of accuracy," and model validation as the "substantiation that a computerized model within its domain of applicability possesses a satisfactory range of accuracy consistent with the intended application of the model" (Schlesinger et al. 1979, 104). More simply, verification ensures that the implementation of the model's formulas and calculations were accurate while validation ensures results accurately describe real world behavior.

Verification of the results is completed by running the same set of formulas through a Python script written by Professor Ray Madachy of the Naval Postgraduate School. These independent calculations made in Python yielded the same results as the aforementioned

spreadsheet tool and serve to verify the initial results. Additionally, it is important to highlight the origins of the formulas used within Detailed COPLIMO. The formulas used were derived from existing product line cost models. These existing product line cost models are peer reviewed and are foundational to product line economics.

For the purposes of this thesis, model validation is done by comparing the model results to real world values reported by Lockheed Martin. The specific comparison made in this paper centers around the output for cost avoidance. Detailed COPLIMO results for cost avoidance compare favorably to what Lockheed Martin was estimating in terms of initial cost avoidance for the Aegis CSL. Lockheed Martin's initial yearly cost avoidance estimates were calculated to be 35% (Gregg, Scharadin, and Clements 2015, 309). Detailed COPLIMO yielded cost avoidance numbers of 21–31% after the initial baseline. Again, these modeled results compare favorably to the predicted values from Lockheed Martin and serve to indirectly correlate the validity of Detailed COPLIMO.

E. THREATS TO VALIDITY

This section identifies potential reasons that the author has reached the wrong conclusions, which are thus invalid. Furthermore, some of the potential issues are flagged as areas for future research to eliminate the threats to validity.

Misinterpreting the data is a possible threat to the validity of the Detailed COPLIMO results. While the Aegis SLOC data obtained from the program office contained actual SLOC sizes of program builds, the individual baselines calculated for this thesis do not exactly match delivered Aegis baseline sizes. These inaccuracies in actual baseline SLOC size values stem from two main areas and are related to how total baseline size was calculated in this paper. First, unique code was calculated by determining the difference in size between the baselines. In actuality, unique code would simply be the newly developed code for the baseline. Secondly, adapted code was calculated by applying an author chosen 30% factor on the baseline total size. This 30% factor was chosen to help reduce the distribution restrictions the program office placed on the public sharing of the data. Therefore, if the two calculations for unique and adapted code yield inaccurate SLOC

sizes, it can be noted, the modeled results may not fully capture product line economic benefits.

Another threat to the validity of the results lies in the cost model input factors for AVPROD, RCR values and RCWR. The threat to validity is the cost model input factors might not accurately describe actual Aegis CSL behavior. Knowing this, the author attempted to select cost factors that best reflect real world behavior. Specifically, the values for average productivity and relative cost of writing for reuse were selected from articles written by the COPLIMO developers, as well as the *Software Cost Estimation Metrics Manual for Defense Systems*.

The last threat to validity addressed in this paper serves to answer the question, how might the author have come to the wrong conclusions? The author stated previously, that Detailed COPLIMO cost avoidance results compare favorably to the initial estimates reported by Lockheed Martin. However, it is important to note, the author does not know how Lockheed Martin is measuring or calculating their cost avoidance values. More specifically, the author does not know which activities or phases of the software development life cycle are being factored into their calculations. For example, the author does not know if Lockheed Martin factors in configuration management or requirements analysis costs into its cost avoidance calculations. Assumptions could be made that that a company would try to include all possible savings from the activities to make cost avoidance numbers look more promising, however, it is not known to the author which specific activities are included in reported cost avoidance numbers.

F. SUMMARY

The methodology and analysis presented in this chapter introduced the reader to the Basic version of COPLIMO, as well as the inputs and outputs of the model. This introduction led to the modeling of a single Aegis baseline to observe the benefits of a product line approach. Rough order of magnitude benefits were seen in this model and provided further justification for adopting a product line approach in the development of software-intensive naval combat systems.

Further adaptation of the model led to Detailed COPLIMO. Definitions and justifications for each of the selected input values, as well as the formulas from which the outputs were derived were shown. The outputs of Detailed COPLIMO include cumulative product line effort, product line effort savings, per product cost savings, per product cost avoidance, and cumulative ROI. Next, the author attempted to verify and validate the model by utilizing Python for independent implementation of the model, discussing model formulas and comparing modeled cost avoidance with reported numbers from Lockheed Martin's initial cost avoidance numbers for the Aegis common source library product line. Furthermore, the author attempted to validate the model by emphasizing the origins of the cost factors selected for the model. The cost factors selected originate from the literature of experts in the field of software cost estimation, as well as empirical data for software productivity. Lastly, the author presented potential threats to the model's validity. These threats included the inaccuracy of input data, cost factors that did not explain real world behavior, and the unknown of which activities of the software development life cycle were included or excluded from Lockheed Martin's cost avoidance calculations.

THIS PAGE INTENTIONALLY LEFT BLANK

IV. CONCLUSION

A. RESEARCH SUMMARY

This research set out to answer two specific questions. The first being, what are the economic returns of a naval combat system software product line? This question is addressed in this thesis by modeling a pre-common source library Aegis baseline. The basic cost model presented the potential economic benefits of adopting a product line approach to an Aegis baseline of that size. More specifically, the model captured the increased effort savings and higher return on investment multipliers of a software product line verses a one-off software system design. This basic cost model could also be applied to a wide variety of other shipboard systems. Navy ships are becoming more and more complex with regard to software reliance, and therefore, product line opportunities abound. Product line approaches have proven to yield great returns, and the U.S. Navy and the DoD should heavily invest in future opportunities to implement the product line approach to ultimately reduce future acquisition costs.

The second question addressed in this thesis was, how can current product line cost models be improved for software-intensive combat systems? This question is addressed by developing an extension to a current product line cost model, COPLIMO. The extension model was modified to account for varying SLOC sized products, as well as baselines with different compositions for unique, adapted, reused code. Additionally, the extension model offered per product cost savings and per product cost avoidance. Cost avoidance values allowed for a like product comparison between real world cost avoidance numbers seen by Lockheed Martin and those calculated in this thesis. One byproduct of the extension model is the ability to use this model for other systems. The extension model in this thesis focused solely on the Aegis combat system or anti-air warfare shipboard function, however, this extension model could be applied to other functional areas like anti-submarine warfare or even bridge functions like navigation.

This research also shows that Basic COPLIMO is a useful model for gaining rough order of magnitude insight into a product line investment. More specifically, Basic

COPLIMO offers upfront initial ROI values across a product line. This simple model could be used very early on in the acquisition phase, well before specific systems are developed or even selected. Outputs of Basic COPLIMO can also be used during an analysis of alternatives to provide objective estimates to aid decision-makers.

The Detailed COPLIMO extension model allows for the inclusion of more data to capture the individual characteristics of specific programs. The goal was to utilize the model to capture the real-world product line cost benefits of the Aegis combat system product line by better predicting past and future Aegis baseline savings and ROI. This was accomplished by modeling five different Aegis baselines and one future Aegis baseline to capture cumulative product line effort, product line effort savings, per product cost savings, per product cost avoidance and cumulative ROI. The added granularity of Detailed COPLIMO yielded higher returns on investment than in the basic cost model. In fact, after the 6th product, ROI from the basic model was only 2.90, while Detailed COPLIMO yielded an ROI of 5.40 for the 6th product. In addition to greater fidelity, Detailed COPLIMO also offers value for project managers who could utilize the model for trade off analysis in the management of individual projects.

Cost avoidance numbers from Lockheed Martin for the Aegis common source library were also compared with the output results of Detailed COPLIMO, more specifically, the total cost avoidance percentages from the years 2011 through 2014. Detailed COPLIMO provided similar results to Lockheed Martin's initial cost avoidance estimates, showing avoidance numbers varying from 21–31% after the delivery of the first product.

B. FUTURE WORK

This thesis set out to offer a modeling framework to capture product line benefits of the Aegis combat system product line. Further extensions of Detailed COPLIMO should be explored to better model other naval systems outside of Aegis, as well as other software intensive systems within the DoD. Additionally, this thesis only considered point estimates for input values, when in reality, these input values represent a range of values. Two areas of focus could then be applied to adapting the model to accept ranges of values as inputs.

First, sensitivity analysis could be done on the input factors to gauge their impact on the outputs such as in a tradeoff analysis. For example, a sensitivity analysis could be completed on the input factor, RCWR. Figure 10 shows the sensitivity the relative cost of writing for reuse has on ROI.

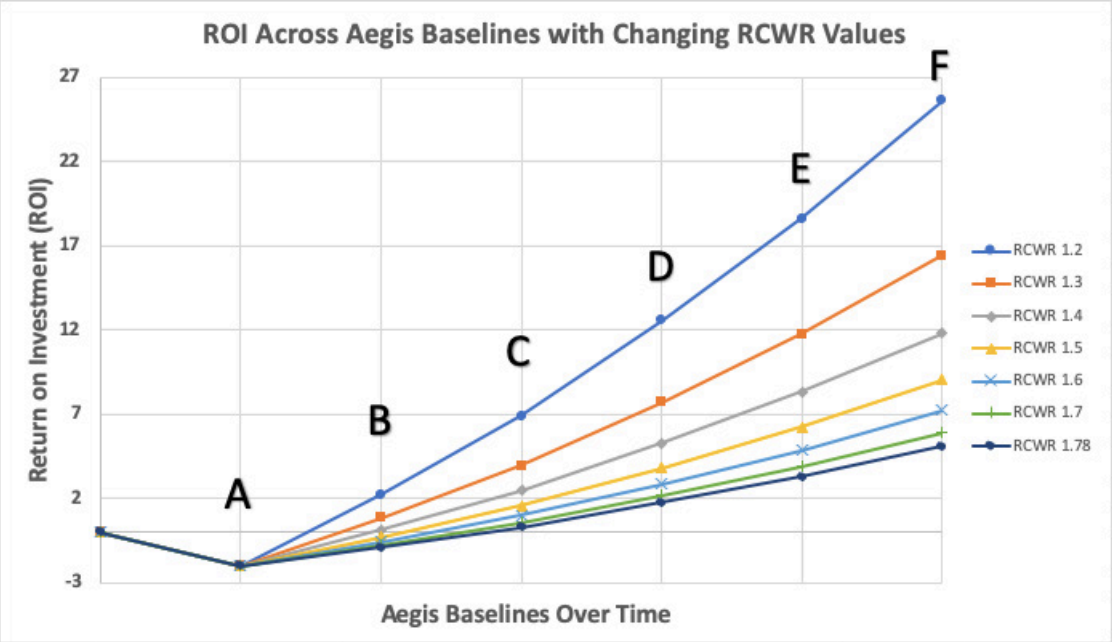


Figure 10. ROI across Aegis Baselines

Figure 10 highlights the increasing ROI over time with the reduction in the relative cost of writing for reuse. This insight provides justification for investing in process improvements which could reduce writing for reuse costs to ultimately bring down the RCWR value and increase modeled ROI.

The second area of focus for model refinement would involve adapting the model for Monte Carlo simulations. Point estimate input factors would be replaced with values from randomly sampled distribution types and ranges. The model would then be run many hundreds or thousands of times to yield a distribution of output results and corresponding probabilities. Both sensitivity analysis and Monte Carlo simulations capture uncertainty within the model, and can help determine the likelihood of obtaining target results.

Data distribution rights, set by the program office, limited the display of actual SLOC values for the particular Aegis baselines analyzed in this thesis. Future work should be conducted to reduce these distribution restrictions and also to obtain additional SLOC data for future baselines, programmatic cost information on the Aegis combat system, and cost savings formulas from Lockheed Martin.

Access to software resources data reports (SRDRs) from Lockheed Martin to the Government contained in the Cost Assessment Data Enterprise (CADE) database would allow for the gathering of actual baseline sizes and compositions. The SRDRs should also contain the reuse weights Lockheed Martin utilizes. These additional data points would give additional verification and validation to the model and allow for the better refinement of Detailed COPLIMO.

Future work should also include the integration of additional models from within the COPLIMO family of cost models. The full COCOMO II model could be integrated to refine Detailed COPLIMO by completely replacing the AVPROD cost factor. COCOMO II accomplishes this by computing effort with size and cost factors. It is a nonlinear model for the software diseconomy of scale and has product, personnel, platform and project factors (Boehm et al. 2000).

Additionally, focus could be directed to integrating with System COPLIMO. This integration would allow capturing both the software and hardware costs of a program including maintenance. System COPLIMO incorporates system costs such as annual change costs, ownership time, and annual interest rates into product line effort savings and returns on investment calculations (Boehm et al. 2011).

Ultimately, the U.S. Navy is tasked to achieve more with fewer resources. Product line approaches to software development and the development of higher fidelity cost models will give the U.S. Navy a competitive edge to be more lethal and agile moving forward.

LIST OF REFERENCES

Boehm, Barry, Chris Abts, A. Winsor Brown, Sunita Chulani, Bradford Clark, Ellis Horowitz, Ray Madachy, Donald Reifer, and Bert Steece. 2000. *Software Cost Estimation with COCOMO II*. Upper Saddle River, NJ: Prentice-Hall.

Boehm, Barry, A. Windsor Brown, Ray Madachy, and Ye Yang. 2004. "A Software Product Line Life Cycle Cost Estimation Model." *Proceedings of the 2004 International Symposium on Empirical Software Engineering*: 156–164.

Boehm, Barry, Jo Ann Lane, and Raymond Madachy. 2011. "Total ownership cost models for valuing system flexibility." *In Proceedings of the 2011 Conference on Systems Engineering Research,* Los Angeles, CA, USA. https://calhoun.nps.edu/handle/10945/60657

Boehm, Barry, Ray Madachy, and Ye Yang. 2004. "Basic COPLIMO." CSSE USC. https://csse.usc.edu/csse/research/COPLIMO/

Boehm, Barry, Tommer Ender, Jo Ann Lane, Raymond Madachy, Adam Ross, Kevin Sullivan, and Gary Witus. 2013. *Tradespace and Affordability – Phase 1*. Report Number SERC-2013-TR-039-1. https://web.sercuarc.org/documents/technical_reports/1525444564-SERC-2013-TR-39-1-Tradespace-and-Affordability-Phase-1-RT-46.pdf

Bosch, Jan. 2017. *Speed, Data, and Ecosystems: Excelling in a Software Driven World*. Boca Raton, Florida: CRC Press.

Brownsword, Lisa, and Paul Clements. 1996. "A Case Study in Successful Product Line Development (CMU/SEI-96-TR-016)." Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University. http://resources.sei.cmu.edu/library/asset-view.cfm?AssetID=12587

Clark, Brad, and Ray Madachy. 2015. *Software Cost Estimation Metrics Manual for Defense Systems*. Haymarket, VA: Software Metrics Inc.

Donohoe, Patrick. "Introduction to Software Product Lines." Accessed April 1, 2019, https://resources.sei.cmu.edu/asset_files/Presentation/2014_017_101_423722.pdf

Gregg, Susan, Denise Albert, and Paul Clements. 2017. "Product Line Engineering on the Right Side of the 'V' ". *In Proceedings of the 21st International Systems and Software Product Line Conference* – Volume A (SPLC '17), Seville, Spain, 165–174.

Gregg, Susan, Rick Scharadin, and Paul Clements. 2015. “The More You Do, the More You Save: The Superlinear Cost Avoidance Effect of Systems Product Line Engineering.” *In Proceedings of the 19th International Conference on Software Product Line* – (SPLC ‘15), Nashville, TN, USA, 303–310.

Gregg, Susan, Rick Scharadin, Eric LeGore, and Paul Clements. 2014. “Lessons from AEGIS: Organizational and Governance Aspects of a Major Product Line in a Multi-Program Environment.” *In Proceedings of the 18th International Software Product Line Conference* - Volume 1 (SPLC ‘14), New York, NY, USA, 264—273.

Hall, Robert. 2018. “Utilizing a Model Based Systems Engineering Approach to Develop a Combat System Product Line.” Master’s thesis, Naval Postgraduate School. https://calhoun.nps.edu/handle/10945/59675

Jones, Lawrence. 1999. *Product Line Acquisition in the DoD: The Promise, The Challenges*. CMU/SEI-99-TN-011. Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University. https://apps.dtic.mil/dtic/tr/fulltext/u2/a373184.pdf

Madachy, Ray. 2018. “System COPLIMO.” https://csse.usc.edu/tools/System_COPLIMO

Madachy, Ray. 2019.”System Cost Model Suite.” https://csse.usc.edu/tools/cost_model_suite.php

Naval Sea Systems Command Public Affairs. 2013. “Aegis Combat System Engineering Agent Contract Award Announced.” U.S. Navy. Last Modified: March 4, 2013. https://www.navy.mil/submit/display.asp?story_id=72500

Nolan, Andy. 2009. “Building a Comprehensive Software Product Line Cost Model.” *In Proceedings of the 14th International Software Product Line Conference*, San Francisco, CA, USA, 249—256.

Northrop, Lina. 2002. “SEI’s software product line tenets.” *IEEE Software* Volume 19, no. 4 (August): 32–40. http://dx.doi.org/10.1109/MS.2002.1020285

Pohl, Klaus, Günter Böckle, and Frank van der Linden. 2005. Software Product Line Engineering, Foundations, Principles, and Techniques. Berlin Heidelberg, Germany: Springer-Verlag.

Program Executive Office for Integrated Warfare Systems, U.S. Navy, unpublished data, March 10, 2019.

Richardson, John. 2018. “A Design for Maintaining Maritime Superiority Version 2.0.” Accessed April 1, 2019. https://www.navy.mil/navydata/people/cno/Richardson/Resource/Design_2.0.pdf

Schlesinger, Stewart, Roy Crosbie, Roland Gagne, George Innis, C.S. Lalwani, Joseph Loch, Richard Sylvester, Richard Wright, Naim Kheir, and Dale Bartos. 1979. "Terminology for Model Credibility." *SIMULATION* 32, no. 3 (March 1979): 103–104. Accessed May 30, 2019. http://dx.doi.org/10.1177/003754977903200304

Threston, Joseph T. 2009. "The Story of AEGIS: The AEGIS Combat System." *American Society of Naval Engineers* 121, no. 3(October): 109–132.

Weiss, David M., and Chi Tau R. Lai. 1999. *Software Product-Line Engineering: A Family Based Software Development Process*. Reading, MA: Addison-Wesley.

THIS PAGE INTENTIONALLY LEFT BLANK

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center
   Ft. Belvoir, Virginia

2. Dudley Knox Library
   Naval Postgraduate School
   Monterey, California