CYBER-PHYSICAL ACQUISITION STRATEGY FOR COTS-BASED AGILITY-DRIVEN ENGINEERING

A Dissertation
Presented to
The Academic Faculty

by

Nathan C.L. Knisely

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in the
School of Aerospace Engineering

Georgia Institute of Technology
May 2016

CYBER-PHYSICAL ACQUISITION STRATEGY FOR COTS-BASED AGILITY-DRIVEN ENGINEERING

Approved by:

Professor Dimitri Mavris, Advisor
School of Aerospace Engineering
Georgia Institute of Technology

Professor Daniel Schrage
School of Aerospace Engineering
Georgia Institute of Technology

Dr. Jean Charles Domerçant
School of Aerospace Engineering
Georgia Institute of Technology

Dr. John Salmon
School of Mechanical Engineering
Brigham Young University

Gary O'Neill
Senior Research Engineer
Georgia Tech Research Institute

Date Approved: April 1, 2016

ACKNOWLEDGEMENTS

This document took twenty-seven years to create. Without twenty-seven years of inspiration and encouragement from my friends and family, I would not have followed the path I've pursued, and would not possess the traits necessary to make this document a reality. First and foremost I would like to thank my parents, Sue and Tony Knisely. Without your commitment to me, my dreams, and my education I would never have had the tenacity to begin this process, nor the perseverance to complete it. I would also like to thank my brother, Ben, from whom I get my sense of humor and my stubbornness. Thank you for never letting any of this go to my head.

I am forever grateful to my friends an colleagues for their support and patience in life, and especially through this process. To Lani Cromwell, Nick Cochran, Daniel Hood, Damian Romney, Bridget Egan, and to all of my friends who have endured hours of my rambling, frustrated, and incoherent conversations: You inspire me, and I'm proud to call you my friends. Thank you for being there to listen. And to Eric Zellers, Alicia Sudol, and Tyler Milner, with whom I bonded over mutual anguish: Thank you for empathizing with me. Your contagious drive gave me the determination to proceed.

I would also like to thank my committee: Dr. Charles Domerçant, Dr. John Salmon, Dr. Daniel Schrage, and Mr. Gary O'Neill. Your feedback helped to guide my research, and to shape this document into what it has become. I am especially indebted to Dr. Charles Domerçant, whose mentorship helped me navigate this process from its conception through completion. Finally, I owe a tremendous debt of gratitude to my Advisor, Dr. Dimitri Mavris, who took a chance on a twenty-something from a school

he had likely never heard of, and who gave me the opportunity to pursue my dreams.

Finally, to my loving girlfriend, Rebecca Lunny: You inspire me more than you can know. I am forever in awe of your endless compassion, patience, and understanding. Thank you for giving me your love and support through this process. Thank you for sitting through countless dinner conversations about my trials and tribulations of constrained optimization methods. Thank you for reminding me to eat. Thank you for making me laugh. Thank you for being you.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Nomenclature

Δ_i Forecasted life cycle of component i

$C^B_{i,d_{red},d_{bridge}}$ Cost of performing a bridging action on component i beginning at date d_{bridge} and ending at date d_{red}

c^H_i Annual holding cost for component i

C^M_{ik} Cost of replacing component i during period k, used in the original MILP formulation

$C^N RE_{d_{red}}$ System-level non-recurring engineering cost for redesigning at date d_{red}, used in the CASCADE MILP formulation

$C^R_{i,d_{red}}$ Cost of redesigning component i at date d_{red}, used in the CASCADE MILP formulation

C^R_{ij} Cost of redesigning component i at date j, used in original MILP formulation

C^S_j Set-up cost for redesigning components at time j, used in the original MILP formulation

c^{LOT}_i Cost per unit for purchasing component i during a life-of-type buy action

c^{NRE} System-level non-recurring engineering cost to redesign a component in the system

c^{red}_i Cost to redesign component i

c^{rep}_i Cost of replacing component i using a form-fit-function alternative

C_{obs} Obsolescence cost, used in the CASCADE MILP formulation

D_{bridge} Set of all discrete d_{bridge}

d_{bridge} Discrete date during which a short-term bridging action can occur

D_{red} Set of all discrete d_{red}

d_{red} Discrete date during which a redesign can occur

i MILP component index, used in original MILP and CASCADE MILP

j MILP index for redesign date

k MILP index for design refresh period

q_i Number of annual failures for component i

r Discount rate

t_{base} Base year to which costs are discounted

$x_{i,d_{red}}$ MILP decision variable indicating whether component i was redesigned at date d_{red}, used in the CASCADE MILP formulation

x_{ij} MILP decision variable indicating if component i is replaced at time j, used in the original MILP formulation

y_j MILP decision variable indicating if a set-up cost was incurred at time j, used in the original MILP formulation

$y_{d_{red}}$ MILP decision variable indicating whether a re-certification cost was incurred at date d_{red}, used in the CASCADE MILP formulation

$z_{i,d_{red},d_{bridge}}$ MILP decision variable indicating whether a bridging action was taken on component i beginning at date d_{bridge} and ending at date d_{red}, used in the CASCADE MILP formulation

z_{ik} MILP decision variable indicating if component i was replaced during period k, used in the original MILP formulation

A-RCI Acoustic Rapid COSTS Insertion

CASCADE Cyber-physical Acquisition Strategy for COTS-based Agility-Driven Engineering

COTS Commercial Off-The-Shelf

CPS Cyber-Physical System

DAS Defense Acquisition System

DoD Department of Defense

FOC Full Operating Capability

FRP Full-Rate Production

GAO Government Accountability Office

INCOSE International Council on Systems Engineering

IOC Initial Operating Capability

JCIDS Joint Capability Integration and Development System

LOT Life-of-Type

LRIP Low-Rate Initial Production

MILP Mixed Integer Linear Programming

NRE Non-Recurring Engineering

PnP Plug and Play

PPBE Planning, Programming, Budgeting, and Execution

SUMMARY

Weapon systems rely more than ever on software to function. Consequently, embedded computing technology must keep pace with the rapid advancement of electronics so that the United States may maintain military superiority. The 1994 Perry Memo, written by then-Secretary of Defense William J. Perry, recognized this need, and prescribed Commercial Off-the-Shelf (COTS) technologies to meet the demand. Despite several proven cost and schedule benefits, COTS-based systems present numerous challenges which revolve around the issue of obsolescence. This research meets these challenges by reevaluating the Defense Acquisition System (DAS) as well as the traditional systems engineering methodology in pursuit of an agile process which can better mitigate the risks of designing a COTS-based system.

Systems engineering traditionally follows a waterfall-centric process wherein requirements are explicated and locked in as early as possible. This causes several issues for COTS-based systems: First, requirements defined too rigidly may limit the COTS components available to select from, thus limiting the benefits of COTS-based system development. Second, when requirements are locked in for COTS-based systems, it effectively means selecting specific components. Given the long development time for defense systems as well as the short life cycles of commercial electronics, doing this too early can lead to systems which are largely obsolete by the time they reach initial operating capability (IOC). Finally, given the inherent life cycle mismatch between military systems (20+ years) and COTS components (about 5 years), obsolescence is unavoidable, and the current ways that obsolescence is mitigated are costly and inefficient. Thus, there is a need for a more flexible systems engineering process which utilizes methods and tools specifically designed to quickly deal with rapidly-changing

requirements. Software engineers once dealt with similar issues, and resolved them by developing Agile Software Development practices. Thus, the development of an Agile Systems Engineering method is a promising strategy to overcoming these challenges.

The Cyber-physical Acquisition Strategy for COTS-based Agility-Driven Engineering (CASCADE) offers solutions to these problems by reevaluating the traditional systems engineering process, as well as the process of generating design refresh strategies for COTS-based systems. To understand the changes CASCADE offers, engineers must first recognize the need to maintain requirements flexibility throughout the design process. This starts by adapting the requirements definition and analysis phases to include an identification phase during which requirements are evaluated for their compatibility with COTS components. This may also require the evaluation of requirement scope to assess whether narrowly-defined requirements can be broadened to accommodate a larger set of COTS solutions. These steps help to delay the selection of COTS components in order to prevent the delivery of largely-obsolete systems upon IOC. Furthermore, to increase COTS utilization, up-front technical requirements should be balanced against the capabilities of commercially-available components.

The second element of CASCADE focuses on the sustainment-related challenges associated with COTS-based systems. Current methods for mitigating obsolescence are largely reactive in nature, which leads to excessively high sustainment costs. CASCADE utilizes forecasted obsolescence dates and component-level cost and reliability parameters as inputs into a Mixed Integer Linear Programming (MILP) formulation. This MILP formulation simultaneously considers all feasible design refresh strategies subject to constraints, and generates a single optimum strategy.

This document presents a literature review which motivates the development of the CASCADE systems engineering framework described above. An additional literature review describes several potential obsolescence forecasting methods for use

in the CASCADE methodology. This document also defines an obsolescence cost formulation, as well as an objective function for use in the Mixed Integer Linear Programming model. This verified model is then applied to a set of representative COTS-based systems in order to evaluate the underlying trends and trade-offs associated with designing and sustaining COTS-based systems.

CHAPTER I

BACKGROUND & MOTIVATION

Today's military aircraft are complex cyber-physical systems which depend upon the latest advances in embedded computing technology to provide a range of capabilities and functions. To support the affordable and timely development of such systems, the Department of Defense has prescribed acquisition policies under the Defense Acquisition System as defined in the Defense Acquisition Guidebook. Since consumers — not the military — drive advances in the electronics industry, the cyber-physical systems employed by the military must take advantage of Commercial Off-the-Shelf (COTS) technologies so that the United States may maintain military superiority[95]. Given that in a global market, everyone — including enemies — gains access to the same commercially-available technologies, the military advantage goes to the nation with the shortest development cycle[47]. The traditional systems engineering process results in the development of one-off systems which are slow to develop, costly to maintain, and which quickly fall behind the cutting edge of technology shortly after their procurement. Therefore, to maintain military superiority, it is imperative that we shift away from the traditional systems engineering process and towards one which better facilitates the development of COTS-based systems.

1.1 Today's Aircraft are Complex Cyber-Physical Systems

Today's military aircraft systems increasingly rely on software to perform key functions as illustrated in Figure 1 [37]. This software — and hence the systems, themselves — must therefore rely more and more heavily on the latest advances in embedded computing to perform these functions. The avionics of today's aircraft, in particular, rely on a tight coupling between computational processes and physical

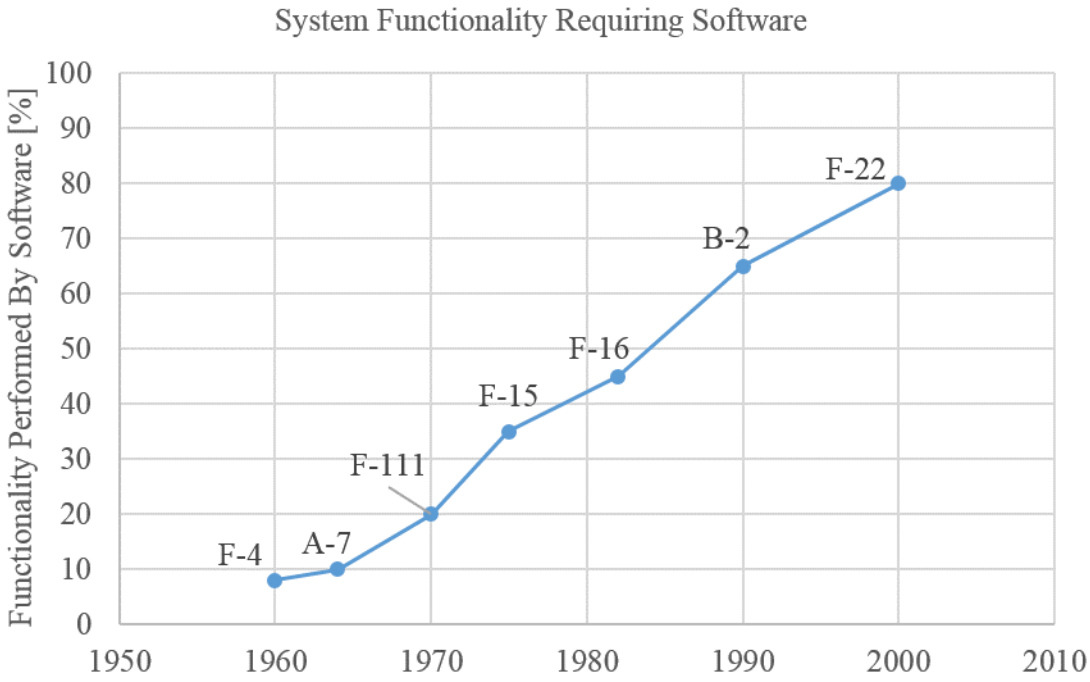

Figure 1: Aircraft systems rely increasingly on software to function [37]. Less than 10% of the functionality of the F-4 was linked to software, while as much as 80% of the functionality of the F-22 requires software. This software, in turn, requires the latest advances in embedded computing technology.

processes. Such systems are commonly referred to as *Cyber-physical systems (CPS)* [44, 63, 64, 100]. This trend towards an increased reliance on electronics in military systems was summarized nicely by an Air Force General, who noted that "in the past, the Air Force used to buy airplanes and add electronics. Today the Air Force buys computers and puts wings on them" [52].

The increased usage of electronics in aircraft has resulted in cost trends shifting away from disciplines such as structures, aerodynamics, and propulsion and towards software and electronics [108]. It is estimated that about 40% or more of the price of DoD aircraft is spent on electronics equipment, and trends indicate that this number is only growing [17, 18, 52, 108]. Meanwhile, the overall cost of new aircraft acquisitions has unwaveringly increased over time as depicted in Figure 2 [35]. Estimates for cost growth for weapons systems range from 3% annually, up to 10% annually for advanced

weapons systems [18]. Cost growth is such an issue, in fact, that Norman Augustine once quipped:

> In the year 2054, the entire defense budget will purchase just one aircraft. This aircraft will have to be shared by the Air Force and Navy 3-1/2 days each per week except for leap year, when it will be made available to the Marines for the extra day. [35]

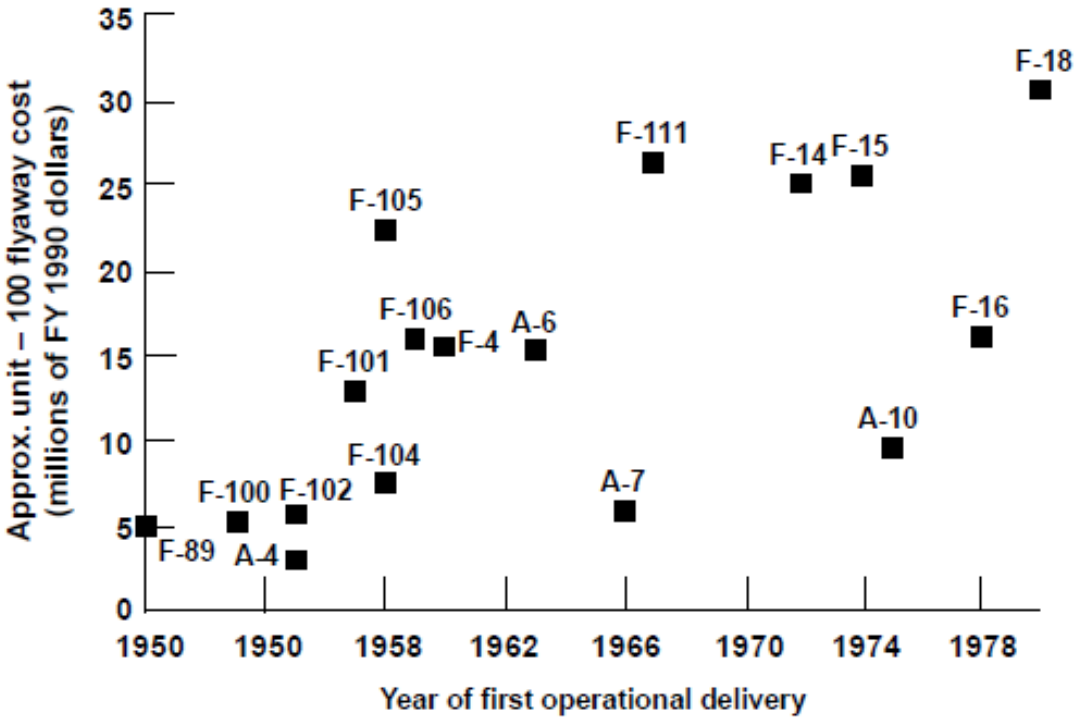

Figure 2: Estimates for the annual rate of cost growth for military aircraft range from 3% all the way to 10% for major weapons systems [35]. To maintain military superiority and ensure the needs of the military can be met, it is necessary to mitigate cost growth of weapon systems.

Thus, two trends are emerging simultaneously: Overall cost of weapon systems is increasing, while at the same time the proportion of the total cost attributable to electronics is growing. To maintain military superiority and ensure the needs of the military can be met, it is necessary to mitigate cost growth of weapon systems. Since

an increasing proportion of this cost is attributable to hardware and software, one of the best ways to mitigate cost growth is to focus on electronics.

1.2 Traditional System Acquisition

To support the development of new systems, the Department of Defense uses a strict acquisition process defined by phases and decision points that must be crossed. This consists of a three decision support systems which must be satisfied from conception through deployment [10, 22, 30, 91]. The Planning, Programming, Budgeting, and Execution (PPBE) Process allocates the resources for budgeting the system's development [30, 91]. The Joint Capability Integration and Development System (JCIDS) is responsible for identifying the requirements for a new system [30, 91]. Finally, the Defense Acquisition System (DAS) is responsible for developing and/or buying the system [30, 91]. These three systems, taken together, are known as "big-A acquisition," while the Defense Acquisition System, alone, is known as "little-a acquisition" [91]. These systems, and their relationships to one another, are illustrated in Figure 3.

The Planning, Programing, Budgeting, and Execution (PPBE) process is intended to provide the military with the best mix of forces, equipment, manpower, and support within fiscal constraints. As the name implies, it consists of four stages. The first is the planning stage, during which the national defense strategy is defined, and a plan for executing that strategy is generated. The programming and budgeting stages occur concurrently. During these stages, the proposed programs are fleshed out, and financial plans are made. Finally, during the execution stage, programs are evaluated against pre-established performance metrics [91].

The Joint Capability Integration and Development System (JCIDS) is the process through which the DoD identifies, assesses, and prioritizes required military capabilities. Not surprisingly, the JCIDS process is commonly referred to as the requirements generation process. The JCIDS process represents a capability-based approach (the

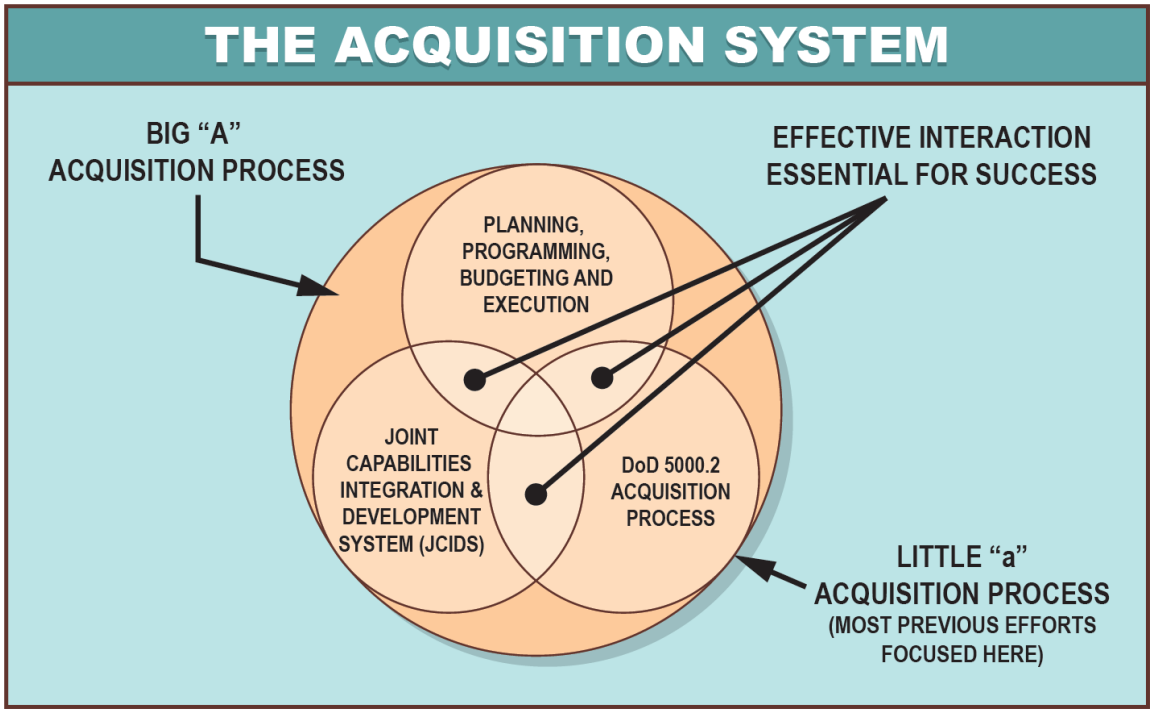

Figure 3: "Big-A" acquisition consists of three phases: The Joint Capability Integration and Development System (JCIDS), the Planning, Programming, Budgeting, and Execution (PPBE) System, and the Defense Acquisition System (DAS). The DAS is also known as "little-a" acquisition. [56]

'C' in JCIDS) to identifying warfighter needs. Before the JCIDS process was created in 2003, the DoD used a threat-based approach to identifying needs. This led to each independent branch of the military identifying its own perceived threats and developing weapon systems to address these threats. By adopting a capability-based approach, JCIDS promotes collaboration among the branches of the military. In the end, the JCIDS process is responsible for identifying if capability gaps must be met with materiel or non-materiel solutions. If the need for a materiel solution is identified and approved through the JCIDS process, then it enters the Defense Acquisition System (DAS) [91].

The objective of the DAS is "to acquire quality products that satisfy user needs with measurable improvements to mission capability at a fair and reasonable price" [30]. The DAS is sub-divided into phases, which are separated by milestones as

illustrated in Figure 4 [30, 91]. These milestones act as "gates" which check that programs meet specific statutory and regulatory requirements before the program can proceed, and their purposes are as follows [91]:

Materiel Solution Analysis Phase — This is where competing systems are analyzed using an analysis of alternatives (AoA) to determine which potential solution is best suited to meet the requirements. This phase concludes with Milestone A, at which point the Milestone Decision Authority must approve the proposed materiel solution.

Technology Maturation and Risk Reduction — During this phase the system, and any nascent technologies, are matured. A decision must be made with reasonable confidence that the proposed system can be developed further to meet the military's requirements while fitting within affordability gaps. Furthermore, requirements are refined and cost caps are finalized. The success of this phase is checked at Milestone B. Note that Milestone B marks the point at which a program becomes a program of record.

Engineering and Manufacturing Development Phase — This is when the system is actually designed and developed, and all technologies and capabilities are integrated into the system. This phase concludes with Milestone C, where it must be shown that the production design is stable, and that cost caps have not been exceeded.

Production and Development Phase — During this phase, low-rate production is authorized. It is during this phase that Initial Operational Capability (IOC) is achieved, which marks the point in time when the system can meet the minimum operational capabilities for the stated needs.

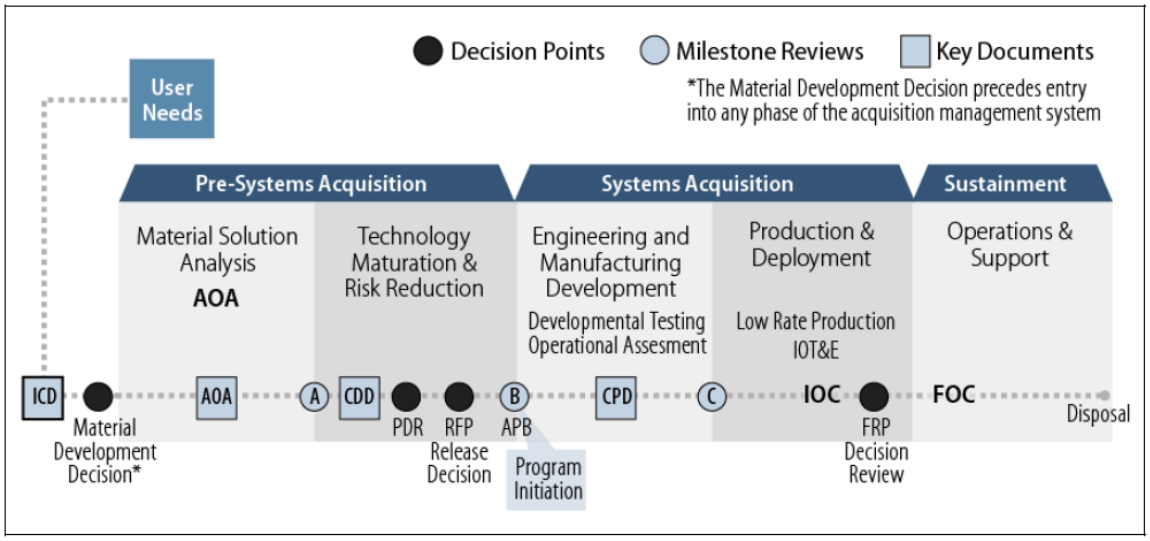

Figure 4: The Defense Acquisition System (DAS) is made up of phases, which are separated by milestones. These milestone acts as a "gate" which checks that programs meet specific statutory and regulatory requirements [91].

1.2.1 The Traditional Systems Engineering Process

If the DoD's acquisition process identifies the need for a new system, that system is developed using the systems engineering process. In this way, the acquisition process and the systems engineering process go hand-in-hand. There are many systems engineering standards, including Chapter 4 of the Defense Acquisition Guidebook, EIA-632, IEEE std. 1220-2005, and ISO/IEC 15288 [8, 58]. In practice, no single standard is used exclusively, though each one follows a similar process: Establish requirements, establish an architecture, decompose the system into subsystems, design the subsystems, build the subsystems, test the subsystems, integrate the subsystems, and then test the system [58, 103]. Figure 5 illustrates how some of these systems engineering processes overlap. These processes are often sequential, waterfall-based models that consist of a top-down requirements decomposition followed by a bottoms-up system realization. Because of this, one common way to visualize the systems engineering process is using a "V-model." Though visually distinct from the a typical waterfall model, the V-model is little more than the waterfall model visually bent in the middle

to show the transition from top-down to bottoms-up development.

To demonstrate the basic systems engineering process, this document offers a detailed explanation of one of the most widely-used processes: The International Council on Systems Engineering (INCOSE) Systems Engineering Technical Process. Note, however, that despite basing this document on the INCOSE process, the discussion and methodology developed herein can be applied to any of the other common systems engineering processes, simply by adjusting the nomenclature and timing of events. INCOSE adopts the ISO/IEC 15288 technical processes for systems engineering. These processes perform several functions like allowing requirements to be elicited and negotiated, enabling the development of a finished product, and supporting that product through its useful life and into its retirement [48]. The INCOSE Systems Engineering Technical Processes is visualized as a V-model in Figure 6, and takes place as follows [48]:

Stakeholder Requirements Definition Process — The goal of this step is to elicit and negotiate requirements from the stakeholders, as well as to define system constraints. This is done by establishing thresholds and objectives for system performance, defining measures of effectiveness, and resolving unrealizable requirements. This creates a baseline for the project's scope from production through disposal.

Requirements Analysis Process — After stakeholder requirements have been defined, they are assessed, prioritized, and balanced in the requirements analysis process. The result of this step is the generation of derived functional performance requirements as well as non-functional requirements.

Architectural Design Process — During the architectural design process, a system solution is synthesized which meets the requirements as defined in the previous steps. This step results in an architectural design baseline, as well as

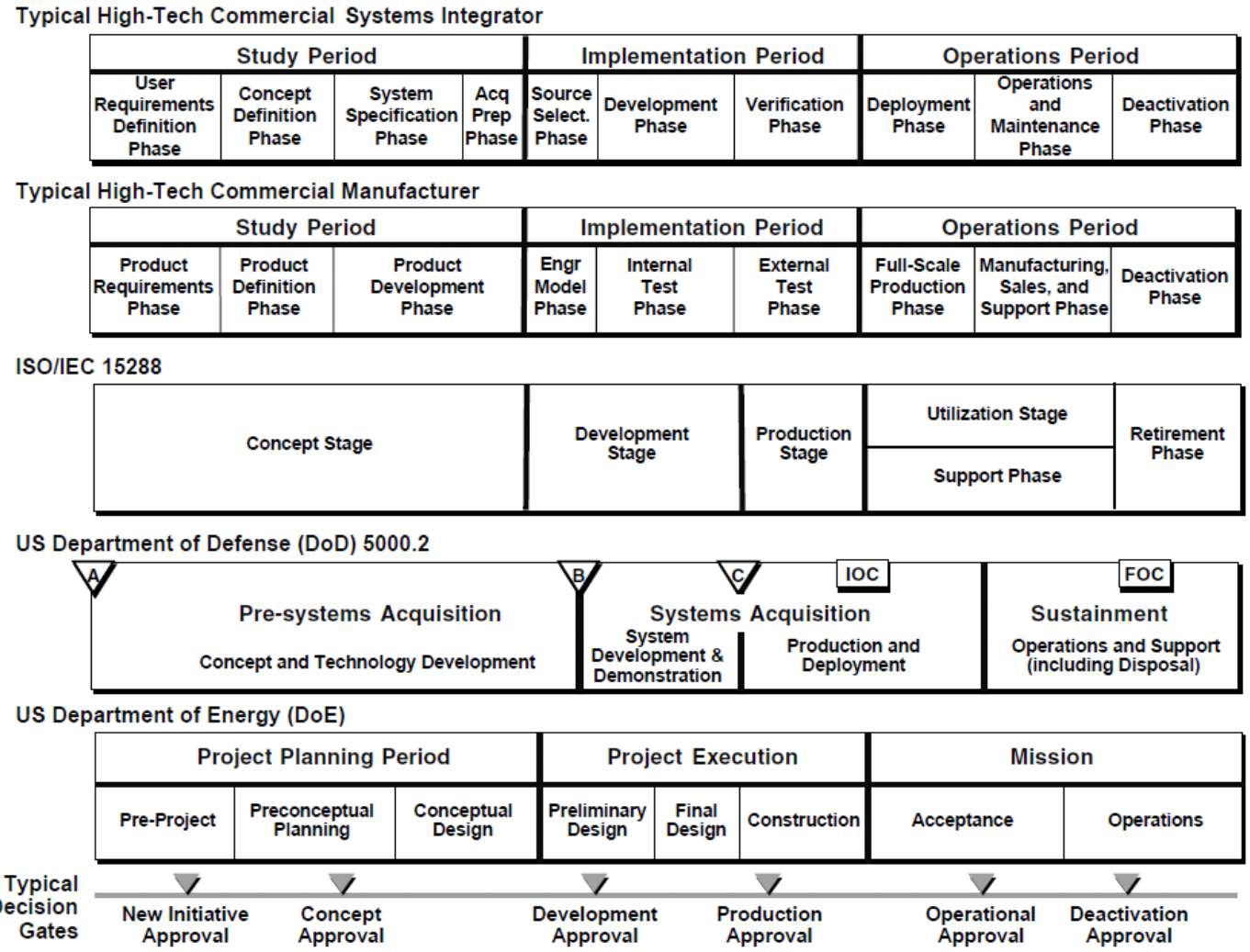

Figure 5: Several standards for systems engineering exist, and each model overlaps with the others [48]. In practice, no one standard is strictly followed — rather, a combination of standards are used [58].

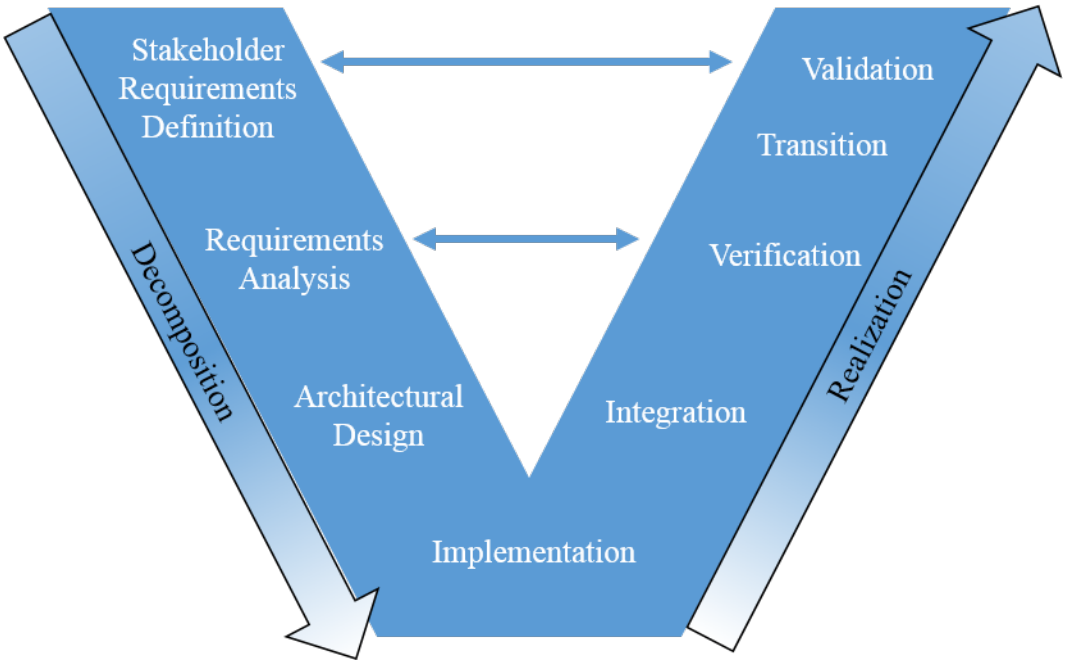

Figure 6: The traditional Systems Engineering Process consists of a top-down requirements decomposition followed by a bottoms-up system realization [106].

a detailed description of and requirements for the system elements.

Implementation Process — In this step, the requirements defined in the architectural design process are used to fabricate the system elements. Thus, this step bridges the gap between the development and production stages of the system's life cycle.

Integration Process — Once the individual components are designed and fabricated, they are combined to realize the system-of-interest during the integration process. The result is a system which meets the requirements and constraints outlined in the previous steps.

Verification Process — During the verification process, the system elements and system-of-interest are checked against the defined requirements and constraints to confirm that each has been fulfilled accordingly. In other words, this step confirms that the system has been built correctly, according the the technical

requirements. Furthermore, remedial actions are defined in the event that any requirements were not properly fulfilled.

Transition Process — This process is the point at which custody of the system-of-interest is transfered from the development team to the customer. Thus, it marks the beginning of the utilization stage of the system's life cycle.

Validation Process — The stakeholders confirm that the realized system complies with their requirements during the validation process. Thus, this step confirms whether the stakeholder requirements were correctly translated into measurable functional and non-functional requirements.

Operation Process — The operation process is simply the period during which the system is utilized to deliver its intended services.

Maintenance Process — As the name implies, the maintenance process is the process by which the system is sustained during its useful life. This process takes place alongside the operation process.

Disposal Process — The disposal process defines the way the system (or system elements) are to be removed from service once the system is depleted.

1.2.2 United States Military Standards — The Old Way of Acquiring Systems

Before commercial off-the-shelf (COTS) components were ever considered for military applications, the DoD relied on "United States Defense Standards," sometimes called MIL-STD or MIL-SPEC [90]. These standards and specifications — while intended to ensure a high level of quality, reliability, and interoperability for military products — were often applied in a one-size-fits-all fashion [62, 82]. With over 30,000 standards and specifications applied in this way, otherwise-simple components became overburdened with requirements, resulting in cost growth [82]. Anecdotal evidence

highlights how these excessive specifications for non-critical components can lead to bloated prices.

In 1984, the New York Times reported that the 10-cup coffee maker on the C-5 Galaxy aircraft was low-pressure certified and designed to withstand 50 G's, and it cost $7,622.00 as a result [25, 73]. Similarly, following the MIL-STD requirements, every computer chip was required to be hermetically sealed in a ceramic package, driving the cost of such components to be as much as 1,000% greater than their commercial counterparts which utilize plastic packaging[82]. The problem of excessive military specifications even extends to commodity products like mouse traps, plastic whistles, and chocolate syrup [52, 2]. Perhaps the most absurd example of this lies in military specification MIL-F-14499F: a 20-page set of specifications which describe how the DoD should buy fruitcakes, including that the vanilla flavoring be included "in such quantities that its presence shall be organoleptically detected, but not to a pronounced degree" [2, 34].

1.3 Acquisition Reform and the Push for COTS

By 1993, many analysts believed that the DoD's reliance on military specifications was "strangling the acquisition process" by increasing costs while preventing firms from offering innovative solutions [82]. The end of the Cold War and a desire to reduce military outlays led to a paradigm shift in acquisition practices known as Acquisition Reform [46, 52, 85]. The seeds of this reform were planted at a dinner at the Pentagon in Spring of 1993, when William J. Perry — then Deputy Secretary of Defense — told defense industry executives that with the end of the Cold War, the DoD could no longer support the military-industrial complex that had come to exist [46]. This dinner was thenceforth known as "the last supper," and represents a precursor to the acquisition reform of the 1990's colloquially known as "faster, better, cheaper" [46].

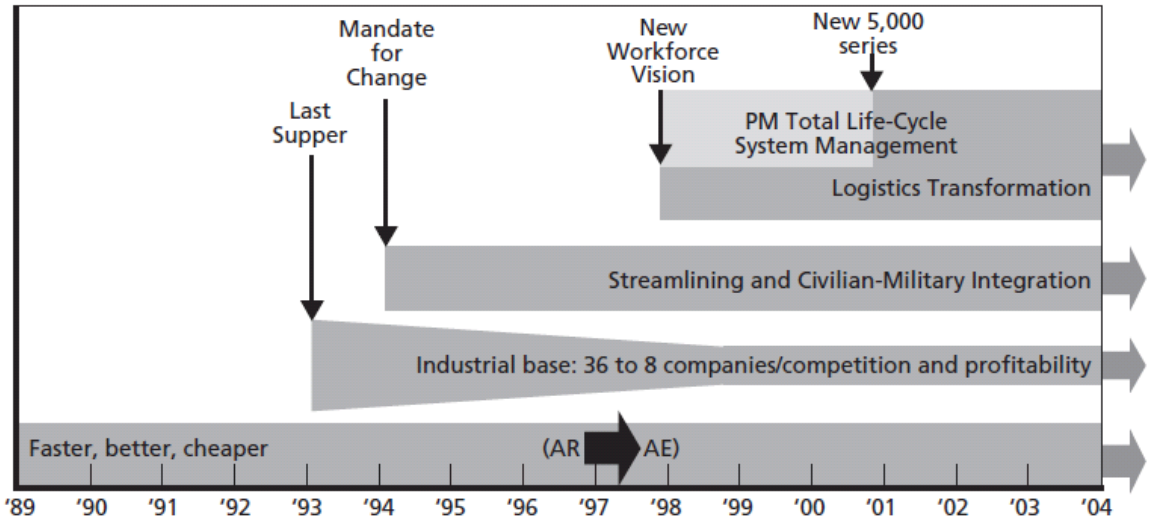

Figure 7: Many acqusition reforms were adopted through the 1980s and 1990s. William J. Perry spurred many of these reforms, beginning with the so-called "last supper" and the Mandate for Change [46].

In 1994 William J. Perry became Secretary of Defense, and he addressed the problems with the acquisition process via a speech called "A Mandate for Change". In it, he outlined many of the themes of the "faster, better, cheaper" mantra, including the need to rapidly acquire commercial products from suppliers who utilize cutting-edge techniques, the reduction or elimination of government-unique terms from contracts, and the adoption of business practices characteristic of world-class customers and suppliers. This spurred the creation and adoption of many acquisition reform policies, most of which were past in the period between 1994 and 1996 [46].

Later, in June of 1994, Perry published a memorandum now known as the Perry Memo, which is considered by many to be a seminal document in the push for wider use of commercial off-the-shelf (COTS) components [41, 52]. In it, Perry prohibited the use of military specifications (except as a last resort which would require a waiver) in favor of performance specifications [52, 85]. Unlike MIL-SPECs, performance specifications only specify *what* performance requirements the finished product must meet without dictating *how* the product should be manufactured [52]. Furthermore, Perry

called for the increase in purchases of commercial items and systems, as well as for the increased adoption of commercial practices in the development of new systems [41]. Requirements to consider COTS solutions were enacted into law in 1994 via the Federal Acquisition Streamlining Act (FASA), and in 1996 via the Clihger-Cohen Act [41, 12, 28].

1.3.1 Benefits of COTS

When it comes to COTS components and COTS-based systems, "Faster, better, cheaper" is more than just a mere mantra. Numerous case studies have shown that, indeed, increased utilization of commercial products and practices can yield systems that cost less to procure, can be identified and obtained faster than a new system could be developed, and can provide enhanced capabilities over their custom-made counterparts. This section explores these benefits by identifying exemplary case studies, as well as mechanisms that allow these benefits to be realized.

1.3.1.1 Economic Benefits

The nature of the commercial marketplace offers numerous benefits for military systems, not the least of which are the economic benefits enabled by economies of scale [4, 83, 82, 6]. The term "economies of scale" refers to the cost advantage that arises with increased output of a product [51]. The total cost of a product is divided into *fixed cost* and *variable cost*, as depicted in Figure 8. The variable cost varies proportionally to the number of united produced, while the fixed cost is insensitive changes in the production rate and remains constant. As more units are produced, the fixed cost can be amortized across all units, reducing the per-unit cost.

The electronics market segmentation depicted in Figure 9 indicates that military purchases make up a small (and dwindling) portion of the total electronics sold worldwide[38]. Consequently, if military electronic equipment is custom-made, the DoD must shoulder the entire fixed cost associated with developing and producing

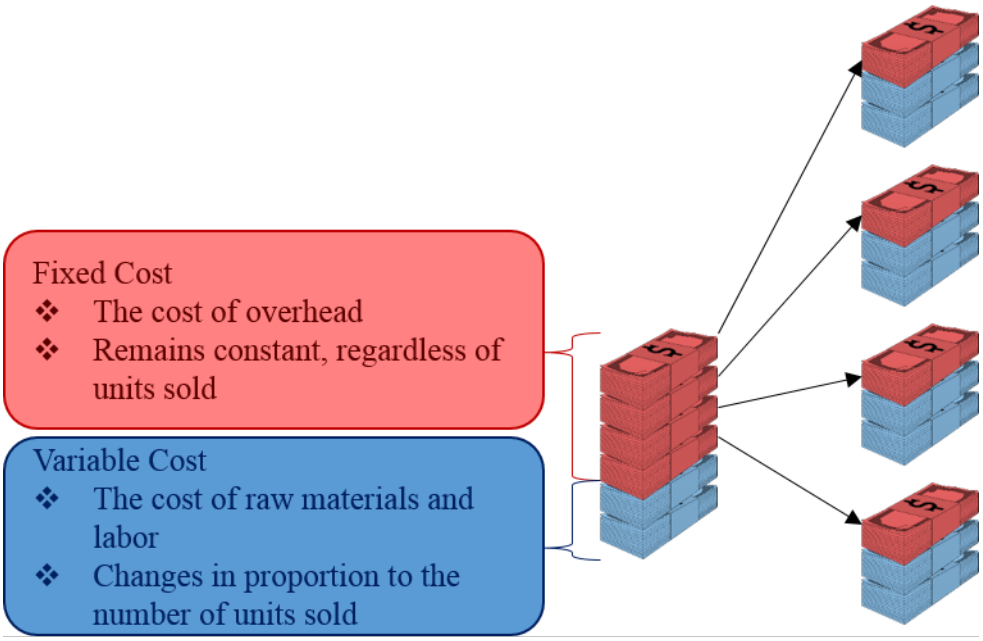

Figure 8: As the name implies, the fixed cost remains fixed regardless of the number of units produced, while the variable cost varies proportionally to the number of units produced. As more units are produced, the fixed cost can be amortized across all of the units, so the per-unit fixed cost decreases.

those products [6]. Since, by definition, COTS components are made for the larger commercial sectors, the research and development costs are built into the price of those components and are thus amortized across *all* buyers of the components [4]. Therefore, by leveraging the mass production of the COTS market, the DoD can greatly reduce the per-unit fixed cost of its weapon systems' components.

1.3.1.2 Demonstrated Benefits of COTS

Perhaps the most dramatic benefit of developing COTS-based systems is the drastically reduced development time required. A 2012 study by the Government Accountability Office showed that once contracts were awarded, off-the-shelf technologies were fielded in a median time of 4 months, compared to 8 months for modified off-the-shelf, and 9 months for newly-developed technologies [99]. The Air Force Research Laboratory also recognizes that the ability to rapidly respond to emerging needs is essential to competing in the 21st century [67, 68, 70]. By developing a library of standardized

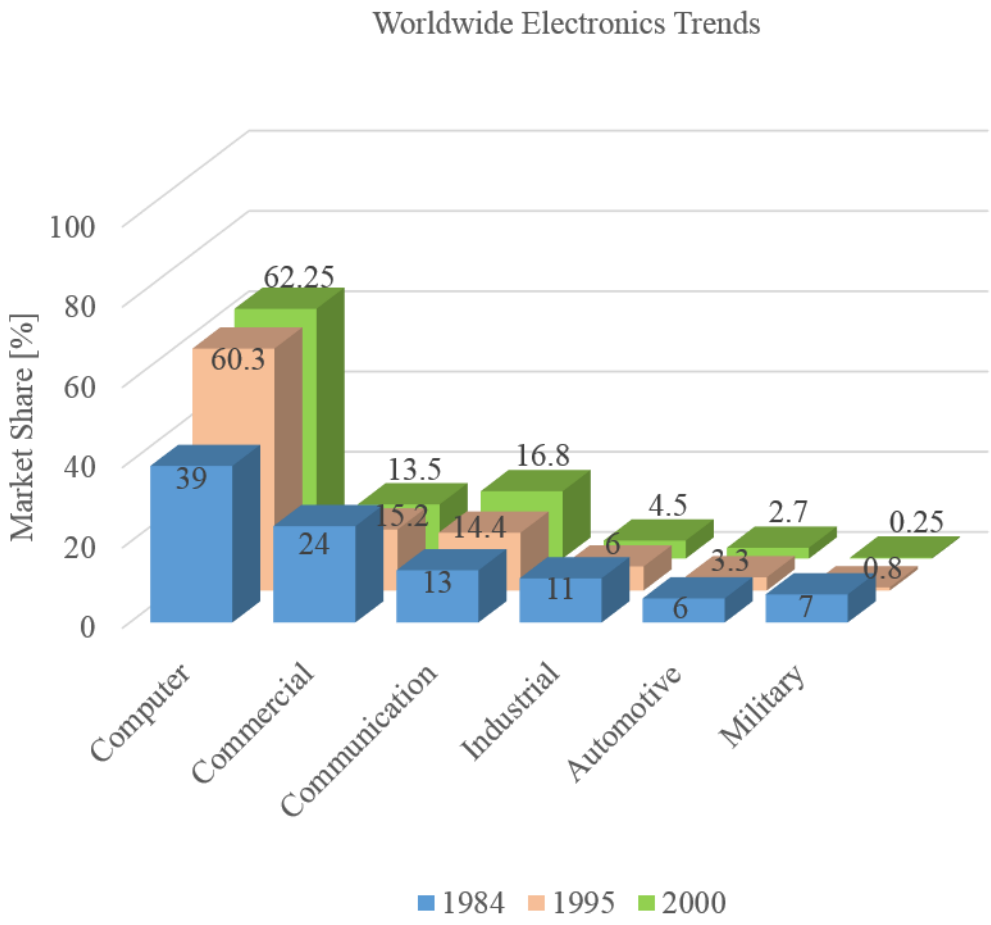

Figure 9: Worldwide electronic parts market segmentation over time [38]. Note that personal and commercial electronics dominate the market while the military comprises less of the market as time progresses. Thus, by utilizing electronic components aimed at the larger commercial sectors, the military can take advantage of the "economies of scale" benefits associated with mass production.

interfaces and relying on off-the-shelf plug-and-play (PnP) components, they aimed to bring the development time of spacecraft down from years to weeks, or even days [76]. They were able to demonstrate the merits of this approach by designing and building a working satellite in just six days, with only four hours spent on assembly [67, 68].

As discussed in § 1.3.1.1, another major benefit to utilizing COTS components is a marked reduction in procurement cost. By leveraging commercially-available technologies, military agencies can take advantages of the economies of scope/scale associated with mass production [47]. By doing this, the fixed costs associated with the research and development of those technologies is shared across all military and non-military customers. The Navy saw this effect first-hand in the early 90's via their Acoustic Rapid COTS Insertion (A-RCI) program. Under this program, when tasked with upgrading the United States Submarine Force's SONAR systems, the Navy saved about $6 billion in development costs and about $80 million in ship-set costs by utilizing COTS-based SONAR systems [59].

Certainly none of these benefits would be meaningful if off-the-shelf components didn't also meet the system-level requirements. Numerous case studies show that COTS-based systems can, in fact, meet or exceed the capabilities of traditional systems. A-RCI, for instance, not only met the requirements of the United States Submarine Force, it also reduced the mean-time-to-repair from 20 minutes down to just 2 minutes [41]. The Navy's 1992 Mission Computer Upgrade (MCU) of its E-2 Hawkeye aircraft demonstrated even more impressive improvements. By utilizing a COTS-based mission computer, the program benefited from 50% weight savings, a 33% reduction in volume, and a tenfold increase in performance [41].

1.4 COTS Challenges

Despite the demonstrated benefits presented by COTS-based systems, there are some obstacles preventing these benefits from being widely realized. These challenges exist for both existing systems as well as for systems under development, though the sources of these challenges are unique. Challenges associated with *designing* a system take place prior to that system's initial operating capability (IOC) and are therefore refereed to in this document as "pre-IOC challenges." These pre-IOC challenges are discussed in § 1.4.1. After a system has been delivered, new challenges arise associated with sustaining that system throughout its life cycle. Such challenges are referred to as "post-IOC challenges" in this document, and are discussed in § 1.4.2. Finally, § 1.4.3 presents several case studies that highlight both pre- and post-IOC challenges.

Both pre-IOC and post-IOC challenges revolve heavily around the issue of component obsolescence. What most people call "obsolescence" is actually more in-line with what the literature calls "discontinuance," therefore several clarifying definitions follow: *Obsolescence* is the phenomenon that occurs when the technology that defines a part or product is no longer implemented [93]. *Discontinuance*, on the other hand, occurs when a manufacturer stops producing a particular part or product, even if that product is still available through third-party means [93]. Note, then, that obsolescence occurs at a technology level, while discontinuance occurs at a manufacturer-specific level [79, 93]. Peter Sandborn adds three additional clarifying definitions of obsolescence shown in Table 1 [87].

1.4.1 Challenges During Development (Pre-IOC)

The acquisition system described in §1.2 is long and cumbersome, and it therefore often results in the slow delivery of systems which exceed budgets and/or fail to meet performance expectations [58, 91, 103, 107]. This, coupled with the exponential progression of technology captured in Moore's Law, means that often times large

Table 1: Peter Sandborn suggests three sub-categories of obsolescence which indicate the severity and effect of the obsolescence event [87].

Type	Description
Weak	Refers to an obsolescence event which requires no change to existing or new systems. As long as the obsolete item is available (either through aftermarket sources, or in inventory) then new systems can be built using it, and existing systems can be repaired using it.
Strong A	Refers to an obsolescence event which does not require the removal of the obsolete part from existing systems, but for which new systems cannot be built with the obsolete part.
Strong B	Refers to an obsolescence event for which existing systems cannot continue to operate with the obsolete part, and for which new systems cannot be built with the obsolete part.

percentages of the commercial components employed on COTS-based systems are obsolete by the time the first product is delivered [33, 65, 80, 107, 109]. For instance, in an analysis of 32 information system acquisitions, the office of the Assistant Secretary of Defense for Network Information Integration showed that the average time between operational requirements definition and IOC was 91 months [58]. Meanwhile, the 12-18 month turn-over for commercial technology means there is a high likelihood that the delivered systems will not be up-to-date [54, 58]. This phenomenon is captured in Figure 10, which indicates that during the lengthy development of a surface ship's SONAR system, over 70% of the COTS components used were obsolete before IOC was reached. A House Armed Services Committee report for the FY2007 defense authorization bill captured this problem nicely, stating:

> Simply put, the Department of Defense (DOD) acquisition process is broken. The ability of the Department to conduct the large scale acquisitions required to ensure our future national security is a concern of the committee. The rising costs and lengthening schedules of major defense acquisition programs lead to more expensive platforms fielded in fewer

numbers. The committee's concerns extend to all three key components of the Acquisition process including requirements generation, acquisition and contracting, and financial management. [91]

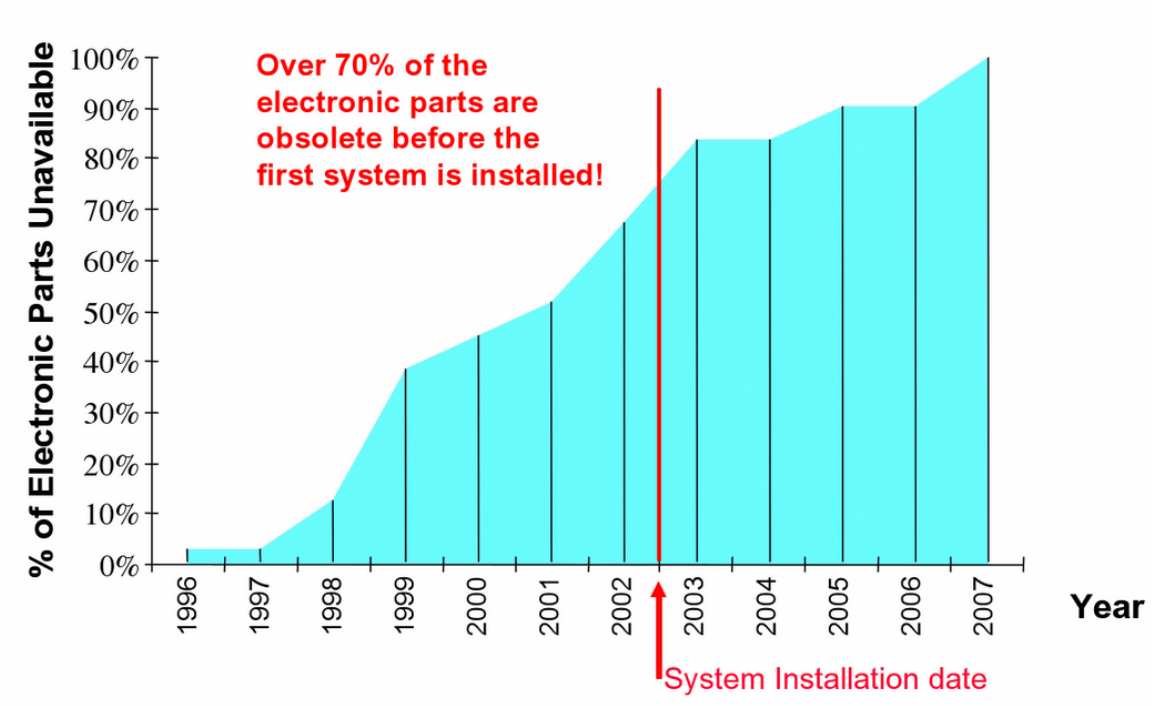

Figure 10: Pre-IOC obsolescence is exemplified in the life cycle of a surface ship's sonar system. It shows that over 70% of the commercial off-the-shelf (COTS) parts were obsolete or otherwise un-procurable before the initial operational capability (IOC) [80].

The issues described above affect all DoD acquisitions, but they are particularly problematic for COTS-based systems due to the issue of obsolescence. Despite this, the goal cannot be to try to eliminate obsolescence altogether. The worldwide electronic trends shown in Figure 9 are a double-edged sword. They not only mean that the military can benefit from economies of scale afforded by commercial practices, but also that the military does not hold enough market share to effect change in the industry. Instead of focusing on eliminating obsolescence, the goal must be to mitigate the *cost of* obsolescence. Thus, to gain the cost (and other) advantages of COTS components, it is necessary to fix the DoD acquisition system to better facilitate the

procurement of COTS-based systems.

1.4.2 Challenges During Operations and Support (Post-IOC)

Once a system passes IOC, obsolescence continues to be an issue, and the high cost of mitigating obsolescence contributes to high sustainment costs which dominate the total life cycle cost [21, 53, 4, 91]. The high cost of post-IOC obsolescence is exemplified by numerous cases. For instance, $500 million was spent to redesign an obsolete RADAR on the F-16, and $264,000 was spent stockpiling an obsolete logic device for the KC-130 to ensure the systems could be supported[69]. The high cost of mitigating obsolescence means that the cost to sustain COTS-based systems often exceeds the original procurement cost [53]. In fact, up to 70% of the life cycle cost of a system can accrue in the operations and support phase of the system's life [91]. Accordingly, such systems are referred to in the literature as "sustainment-dominated systems[1]"[92].

1.4.3 Demonstrated Challenges of COTS

The Littoral Combat Ship (LCS) was designed to replace slower and larger specialized ships like minesweepers, and it featured a flexible mission module space and a shallow draft. The original requirements set featured 15,261 technical requirements, and the ship was expected to be constructed using commercial practices at an estimated cost of $220 million. The final requirements set was provided four months later, and the original set of technical requirements was nearly doubled. Neither the DoD nor the shipbuilders fully understood the consequence of this requirements growth. As a result, the development process was reverted to a more cumbersome, military-centric environment. Thus, the commercial advantages were lost, and the estimated cost grew to around $500 million [90].

A similar example is found in the presidential helicopter replacement (VH-71). A

[1]The concept of "sustainment-dominated systems" is discussed in detail in § 2.3.1.

$1.7 billion contract was awarded to a Lockheed Martin-led team, who proposed a design based on the commercial AugustaWestland EH101 aircraft. New requirements inserted after the original contract was awarded plainly exceeded the limits of available technology. By applying requirements akin to MIL-SPECs to a COTS-based system, the subsystems needed to be reengineered, and the estimated cost of the program grew to $11 billion [90].

The Armed Reconnaissance Helicopter (ARH) was proposed as a replacement for the OH-58 Kiowa Warrior fleet. The design was commercially-derived, beginning with a commercial helicopter and adding intelligence, surveillance, reconnaissance (ISR) and engagement capabilities. The winning contract stated that the helicopter would be certified using commercial standards. After the contract was awarded, however, the US Army's Aviation and Missile Research Development and Engineering Center mandated that the helicopter would go through military certification instead. This resulted in increased development time while producing lower capability. Furthermore, the cost for development grew from $350 million to $900 million, while the cost per aircraft grew from $8.5 million to $14 million. In the end, the program was cancelled, citing cost increases as the primary reason [90].

These case studies demonstrate a lack of forethought regarding changing requirements in a COTS environment. COTS components are designed to meet the needs of a broad commercial market. When COTS-based systems are burdened with requirements that fall outside the norms for commercial components, considerable time, effort, and money is needed to redesign the system. The logical result of this is cost and schedule slippage.

1.5 Gaps in Current Methods

The previous section identifies the signs and symptoms of a broken system, but it does not identify the root cause of those symptoms. In this section, the "gaps" in

the current approach to designing and acquiring COTS-based systems are identified. These gaps represent the mechanisms which can prevent the benefits discussed in §1.3.1 from being realized, and in some cases cause the drawbacks shown in §1.4.3. Since the challenges for COTS-based systems can be broken into pre-IOC and post-IOC challenges, the gaps identified in the following section are similarly broken into two groups. First, gaps in the standard systems engineering process are identified in §1.5.1. These gaps primarily cause the pre-IOC challenges, but may also lead to post-IOC challenges. Next, gaps in the acquisition process are presented in §1.5.2. These primarily cause post-IOC challenges.

1.5.1 Gaps in Standard Systems Engineering Process

The Defense Science Board studied the development of six COTS-based systems (some successful, and some not successful), and concluded that programs do not adequately integrate systems engineering analysis early enough to influence decisions and trade-offs [90]. Due to the rapid rate of technology advancement, the overly-rigid structure of the traditional systems engineering process is insufficient when applied to the development of COTS-based systems [23, 52, 54, 58, 103, 90, 105]. Traditionally, systems engineering is a top-down and process-bound process which is shortsightedly focused on early correctness [58, 103]. As such, its goal is to explicate requirements as early as possible, and to postpone modifications to the system until the maintenance phase [58, 74]. While this approach has worked for countless ground-up system designs, it presents new challenges when applied to the development of COTS-based systems.

William J. Perry noted that "the problem of unique military systems [does] not begin with the standards. The problem [is] rooted in the requirements determination phase of the cycle" [52]. For custom development under the traditional systems engineering process, the development team identifies and fixes requirements early on, defines an architecture, and then undertakes the custom implementation [23,

105]. When this process is applied to the development of COTS-based systems, these rigidly-defined requirements tend to preclude the use of COTS altogether [23, 24, 105]. In other words, it is unlikely that the COTS marketplace will yield any products that fit the *a priori* requirements imposed on the system [23].

If the *a priori* requirements do manage to yield a set of suitable COTS components, a new set of challenges presents itself. The traditional systems engineering process's focus on locking in requirements early and postponing design modifications until the maintenance phase creates an inability to adapt to a changing environment — both in terms of changing user requirements, and a changing technological landscape [58]. Given this inflexibility, in conjunction with long development times and the rapid advancement of the COTS marketplace, it is easy to see why obsolescence is often encountered before production begins [54, 58, 65, 107]. Current systems engineering practices to not provide a framework to rapidly respond to change, and this inability is an underlying theme in the development of many cyber-physical systems [58].

1.5.2 Gaps in Current Acquisition Strategies

A March 2009 report by the Defense Science Board concluded that "the conventional DOD acquisition process is too long and too cumbersome to fit the needs of the many IT systems that require continuous changes and upgrades" [58]. Furthermore, participants in a RAND study on acquisition reform expressed concerns that even the most reasonable-sounding acquisition reforms — like the elimination of MIL-SPECs — could backfire when the DoD finds itself having to support those systems over their long life cycles [46, 52]. Indeed, it is the continuous need for changes and upgrades that leads to high support costs, and hence the phenomenon of sustainment-dominated systems [53, 4, 84]. The rapid and constant change in the technological and operational environment for weapon systems has pushed the traditional acquisition

process to the limits of its capabilities, and thus it sometimes fails to produce systems that meet estimated costs and/or performance expectations [91, 103]. Fundamentally, DoD's increased reliance on COTS-based systems requires a paradigm shift in the way acquisitions are done [74].

The fast pace of technology development, coupled with the long life cycles of military systems, means that obsolescence is an unavoidable consequence of COTS-based military systems. To deal with this obsolescence during the operations and support phase of such systems, many acquisition activities must be repeated throughout the life of the program, and in some sense, development and sustainment activities must merge [74]. For instance, given the ever-present need to manage obsolescence, it becomes necessary to repeat cycles of requirements definition, commercial item evaluation, and systems engineering throughout the entire acquisition process — including the operations and support phase [74]. Numerous acquisitions have stumbled for not adopting this mindset [74]. As was noted by another Defense Science Board study, the DoD must adopt alternative acquisition strategies specifically aimed at COTS [57, 90].

1.6 Research Goals

Section 1.3.1 identified the potential benefits of COTS-based systems, including lower procurement costs due to economies of scale, lower cycle times due to commercial practices, and a potential for lower total life-cycle cost. Still, several gaps in the ways such systems are developed and sustained prevent these benefits from being fully realized. These gaps were identified in § 1.4, and were broken into two categories: The Pre-IOC challenges introduced in §1.4.1 include a lack of systems engineering tools to accommodate COTS-based system development, as well as a fundamental flaw in the requirements definition phase of the systems engineering process which tends to preclude the use of COTS. The Post-IOC challenges introduced in §1.4.2

include the high cost of mitigating obsolescence during a system's life. These gaps motivate the primary goal of this research:

Research Objective
To develop a methodology for the development and sustainment of long life cycle COTS-based systems which addresses the obsolescence concerns associated with COTS components throughout the system's life cycle.

Since COTS-based systems currently exist and new ones are being developed, achieving this objective requires that both pre-IOC obsolescence issues and post-IOC challenges be overcome. Therefore, this research objective is achieved by answering two primary research questions:

Research Question: 1
How should the requirements definition phase of the engineering process be modified with obsolescence in mind to enable the development of a new COTS-based design methodology?

The goal of Research Question 1 is to address the pre-IOC challenges associated with the systems engineering process. Thus, answering Research Question 1 will help to close the gaps associated with designing new COTS-based systems. Research Questions 2 is aimed at addressing the post-IOC obsolescence challenges associated with both new *and* old systems:

Research Question: 2
How should obsolescence mitigation strategies be incorporated into a new COTS-based design methodology?

To answer each of these primary research questions, several sub-questions will need to be addressed, as described in Chapter 2.

1.7 Document Organization

This document is organized as follows: Chapter 2 presents a literature review which expands on the problems identified above, as well as potential elements of a solution. Chapter 2 also identifies additional research sub-questions which drive this research, and it culminates in a framework for the Cyber-physical Acquisition Strategy for COTS-based Agility-Driven Engineering (CASCADE) methodology. Chapter 3 builds upon this framework through the creation of a Mixed Integer Linear Programming (MILP) formulation, as well as an accompanying obsolescence cost model. These two models are then verified in Chapter 4. Chapter 5 applies the verified MILP formulation to a set of representative systems in order to uncover underlying cost trends which drive the development of COTS-based systems. Finally, Chapter 6 summarizes the findings of this research, and identifies areas of future work.

CHAPTER II

LITERATURE REVIEW

The traditional systems engineering process was introduced in §1.2.1, and the top-down requirements decomposition portion of the this process (i.e. the left side of the systems-engineering V) was identified in § 1.5.1 as being a primary challenge in the development of COTS-based systems. This is because the system implementation phase — where the top-down requirements decomposition transitions to the bottoms-up system integration — is fundamentally different for COTS-based systems. COTS-based systems represent a paradigm shift in system development from a "make" mentality to a "buy" mentality [106]. Therefore, instead of designing, creating, and fabricating system elements, systems engineers are selecting and purchasing these system elements. This requires the upstream elements to be adjusted to match this fundamental change in implementation. In particular, this requires redefining and re-prioritizing the activities associated with requirements definition and requirements analysis. This is the core focus of Research Question 1:

Research Question: 1
How should the requirements definition phase of the engineering process be modified with obsolescence in mind to enable the development of a new COTS-based design methodology?

2.1 Needed Changes to Requirements Decomposition

The fundamental shift from a "make" to "buy" mentality requires a similar shift in the way requirements are handled throughout the systems engineering process. Section 1.5.1 highlights two primary issues in the the traditional systems engineering

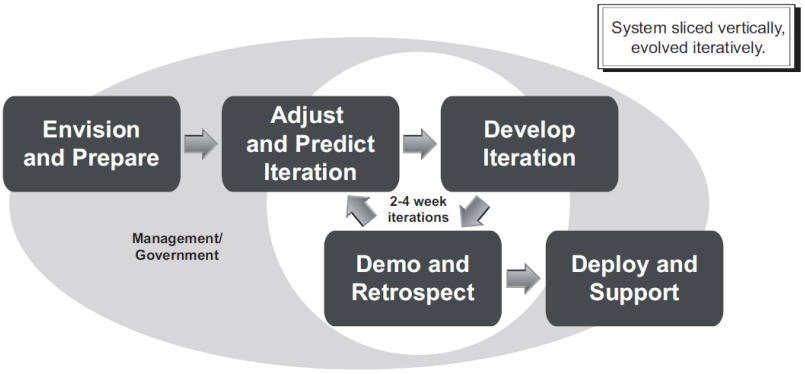

Figure 11: The disciplined agile process involves iteration between development, and reevaluation of requirements based on environmental trends [103].

process: The first is a tendency to define requirements too rigidly, which results in the preclusion of COTS component usage. The second is the desire to identify and lock in those rigid requirements as early as possible, which results in the selected COTS components being obsolete (or nearly obsolete) by the time the first units are delivered. The common factor between these two issues is inflexibility. Inflexibility in how requirements are initially defined results in requirement sets which are overburdening, and which do not match the offerings of the COTS marketplace. Inflexibility throughout design means that changes in the technological landscape cannot be accounted for after requirements have been locked in. The need for increased flexibility is not unique to the systems engineering process. Software developers once faced similar challenges which they attributed to their overly-rigid process-bound development models. This eventually led to a new paradigm in software development known as *agile software development* [19, 45, 58, 94, 103].

Agile software development has been defined in the literature in a number of ways. One such definition describes agile software development as:

> An iterative and incremental (evolutionary) approach to software development which is performed in a highly collaborative manner by self-organizing teams within an effective governance framework with just enough

ceremony that produces high quality software in a cost effective and timely manner which meets the changing needs of its stakeholders [16, 19].

An alternate definition calls it:

> A persistent behavior or ability of a sensitive entity that exhibits flexibility to accommodate expected or unexpected changes rapidly, follows the shortest time span, uses economical, simple and quality instruments in a dynamic environment and applies updated prior knowledge and experience to learn from the internal and external environment [94].

Regardless of the preferred definition, agility refers to "the speed of operations within an organization and speed in response to customers" [58]. This school of thought represents a departure from process-bound methods, and towards a system that embraces (or at least accommodates) change, as captured in Figure 11 [45, 58, 94]. This allows software developers to respond swiftly to both changing requirements *and* changing technology [58]. In doing so, agile software development offers the short-term benefits of reduced cost and higher quality, as well as the long-term benefits of increased responsiveness [19]. Ultimately, Agile is more of a state of mind and less of a set of rules that must be blindly followed [103]. Adopting a similar state of mind in the domain of systems engineering can help address the issues associated with inflexibility during the design of COTS-based systems. Research Question 1.1 reflects this notion by asking what, specifically, must change to move towards a more agile and COTS-friendly systems engineering process:

Research Question: 1.1
What changes need to be made to the SE Requirements Decomposition to better facilitate the use of COTS components?

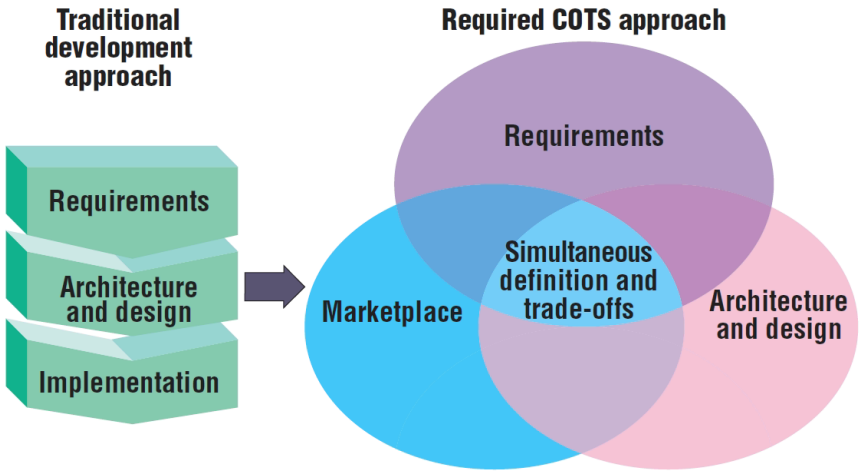

Figure 12: The traditional systems engineering process is rigid, waterfall-centric, and focused on explicating requirements as early as possible. This creates issues wherein the COTS marketplace cannot meet the needs outlined by the requirements, effectively precluding any available COTS components. To address this, the system requirements, COTS marketplace, and system architecture and design must be considered simultaneously. Furthermore, the bottoms-up requirements imposed by COTS components must be balanced against stated customer requirements [23].

Both of the aforementioned pre-IOC challenges can be addressed by adjusting various aspects of the requirements definition/analysis phases of the systems engineering process. The first issue is the specific and overly-rigid requirements that can preclude the use of COTS components. This issue ultimately stems from a lack of knowledge about the COTS marketplace.

David Walden uses the analogy of a "creator" versus a "composer" to describe the changing role of systems engineers [106]. The traditional systems engineer acts like a "creator." That is, she is responsible for defining and refining requirements through a top-down requirements decomposition, and then using these requirements to perform a bottoms-up system integration and test. The system is ultimately designed down to the lowest-level elements, and the inner-workings of each element is known. A COTS-based systems engineer is more akin to a "composer." A composer does not *make* each instrument, but must instead select appropriate instruments for her needs.

Furthermore, she must be able to take advantage of the features of each instrument, and understand the emergent properties when multiple instruments are used together. By analogy, a COTS-based systems engineer does not make the low-level system elements, but must be aware of the available components in the COTS marketplace, and must have the ability to select an appropriate set of components to achieve the desired goals[106].

Under the traditional systems engineering paradigm, the customer defines the requirements, which ultimately determine the system's capabilities. For the agile COTS-based paradigm, this emphasis on locking in requirements early must be relaxed. This is because the components available in the COTS marketplace effectively impose additional (or modified) requirements on the system. Therefore, the top-down requirements definition must be adjusted for these bottoms-up requirements imposed by the COTS marketplace. This can be done by adopting from the Agile methodology described above. Figure 12 captures this notion, depicting the simultaneous consideration of requirements, the COTS marketplace, and system architecture and design.

The first step involves assessing the original requirements set for COTS applications. This may be done in an informal manner with the help of subject-matter-experts. The goal is to separate the requirements that COTS components *cannot* satisfy from those which COTS components *can* or *may* satisfy. Ideally, the set of requirements for which COTS components are not applicable should be minimized as much as possible.

The next step involves prioritizing the COTS-applicable customer requirements. This can be done either categorically or numerically, as long as essential (wholly inflexible) requirements can be separated from more-flexible requirements. The goal here must be to minimize, as much as possible, the set of wholly-inflexible requirements, as these inflexible requirements will result in a narrower set of applicable

COTS components. Note that the intention is not to eliminate the original set of requirements — merely to relax it where possible to aid in an initial assessment of the COTS marketplace.

The resulting "relaxed" requirements set can then be used to assess the COTS marketplace for acceptable components. This is done by filtering the set of components by the "relaxed" requirements, starting first with the highest-priority requirements and imposing additional requirements starting with the next-highest requirements. If, after this filtering process, no COTS components are shown to satisfy the relaxed requirements, then either requirements were not relaxed enough, or some subset of the requirements may not have applicable COTS solutions. In either case, the previous steps should be repeated until a COTS solution space opens up.

The filtered results will yield a subset of the COTS marketplace which meets the relaxed requirements. The next step is to assess the down-selected set of COTS components, and ultimately strike a balance between the ideal requirements set (i.e. the original requirements set) and the performance provided by available COTS components. This can be done in a variety of ways. For instance, using a multi-attribute decision-making (MADM) technique, one could rank-order the down-selected COTS components in terms of closeness to the desired solution (i.e. closeness to the original requirements). Either via a predefined threshold, or using the expert opinions of subject-matter experts, components deemed "good enough" can then be selected, and the associated requirements officially relaxed. As Navy Admiral Jonathan Greenert once said: "We can no longer afford, strategically or fiscally, to let the perfect be the enemy of the good — or the good enough — when it comes to critical war fighting capability. [105]" In that vein, the ultimate goal of this step is not to end up with a set of COTS components that meets the original set of requirements, but rather to end up with a set of components that satisfies the essential requirements, and which deviates as little as possible from the non-essential requirements. These concepts are

summarized in Conjecture 1.1a:

Conjecture: 1.1a

There is a need for an identification phase at the beginning of the systems engineering process, during which customer requirements are evaluated and prioritized for COTS applicability. Also during this phase, the COTS marketplace is evaluated, and the bottoms-up requirements imposed by available COTS components are balanced against stated customer requirements to encourage higher COTS utilization.

Conjecture 1.1a addresses the first pre-IOC issue: the preclusion of COTS component usage due to overburdening and rigid requirements. It does not, however, consider the life cycle of the selected COTS components. Consequently, given the relatively short life cycles of commercial electronics and the long development times associated with military systems, there is a high likelihood that the down-selected COTS components will be at or near obsolescence by the time production is reached. This warrants additional changes to the systems engineering process.

Throughout the development of the system, the technology landscape will continue to change. New, possibly-revolutionary, items will enter the market, and old items will become obsolete. Meanwhile, the availability of other items may decrease and/or their cost may increase, resulting in initially-selected components becoming obsolete or undesirable. Likewise, options that were previously eliminated from the original down-selection (possibly due to high cost, low availability, or low reliability) may become desirable due to changes in production, price, or other market forces. These potential changes in the COTS marketplace necessitate the periodic reassessment of the COTS marketplace, and the possible re-selection of COTS components. This complement to Conjecture 1.1a is summarized in Conjecture 1.1b as follows:

Conjecture: 1.1b

There is a need for the periodic reassessment of the COTS marketplace throughout the systems engineering process. The goal of these assessments is to determine how the COTS technology landscape has changed, and to reevaluate selected COTS components. Components which are at or near obsolescence *must* be replaced, and new components which better meet the original requirements set should be considered.

The required changes identified in this section are illustrated in Figure 13.

2.2 Modified COTS

Conjectures 1.1a and 1.1b assume that systems engineers have the ability to relax requirements in order to increase the number of COTS components available to select from. However, this will often not be a feasible option since stakeholders are unlikely to adapt their needs to match the commercial market. That is, the requirements are requirements for a reason. Fortunately, systems engineers are not limited to modifying requirements to match commercially-available components. Instead, systems engineers can modify the *components* to meet their requirements [102]. COTS components which have been modified for this purpose are simply referred to as *modified off-the-shelf (MOTS)* [41, 102].

The Federal Aviation Administration (FAA) defines modified off-shelf components as those which are "employed outside the manufacturers environmental specifications"[102]. Other sources refer to such components as *ruggedized off-the-shelf (ROTS)* or *government off-the-shelf (GOTS)*[41, 61]. Krinke and Pai, for instance, define *ruggedized off-the-shelf* products as "products purchased as COTS, and subsequently modified, burned-in, tested, used outside the supplier's specification limits, and/or identified differently to perform under adverse environmental conditions" [61]. This inconsistent terminology can lead to a lack of understanding about the costs and benefits

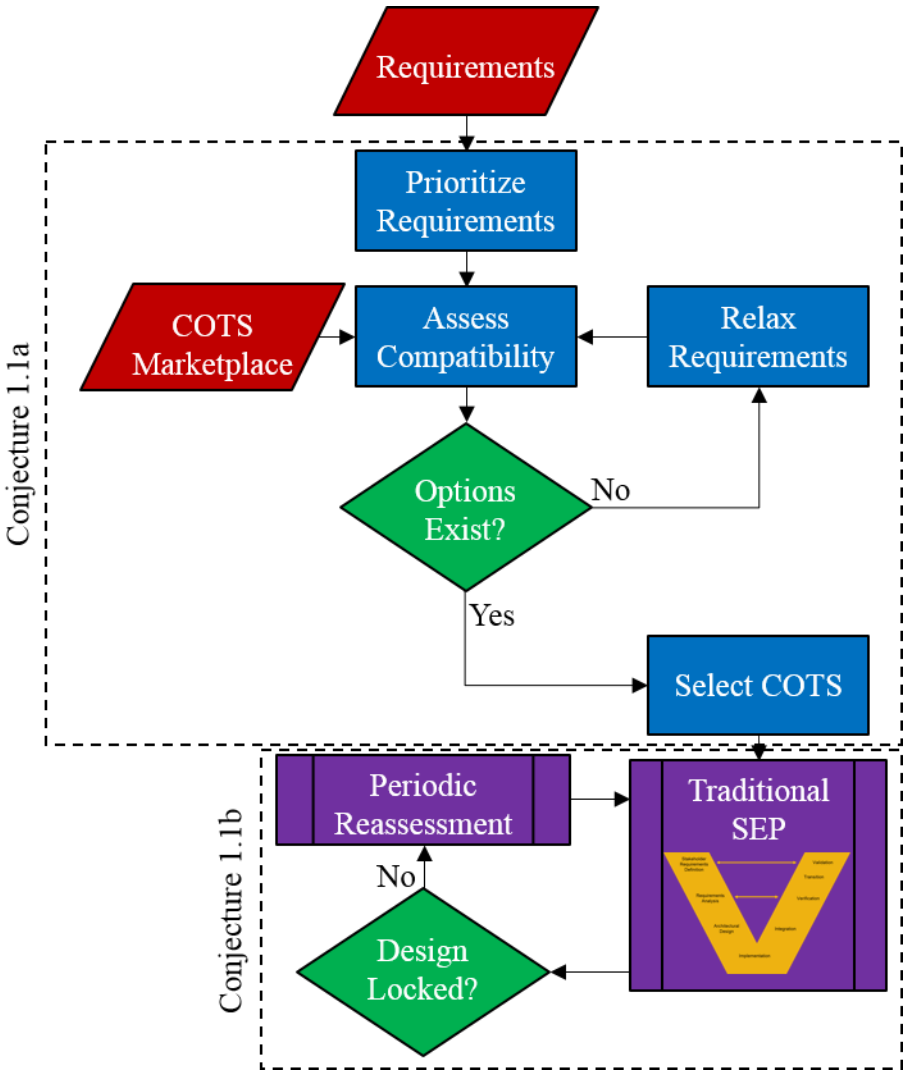

Figure 13: Conjecture 1.1a indicates the need for an "identification phase" at the beginning of the systems engineering process, during which requirements are prioritized and compared against the COTS marketplace. After the initial selection of components, Conjecture 1.1b indicates the need for a periodic reassessment of the COTS marketplace to account for evolving requirements, as well as an evolving technology landscape.

associated with COTS modification.

A Defense Science Board Task Force found that when organizations claim to use "off-the-shelf" systems, they are often using components that have been modified to some degree. They therefore identified eight "levels" of "off-the-shelf" components, wherein lower levels indicate fewer modifications. For instance, a Level-1 COTS component is purchased and used "as is," whereas a Level-5 COTS component requires an extensive redesign with military-specific parts. Thus, Level-1 COTS can be thought of as "truly off-the-shelf," and all other levels represent some degree of modification. A full description of each of these levels is provided in Table 2[90].

Although COTS modification represents an alternative to relaxing requirements as shown in Figure 14, modification requires a basic tradeoff. On the one hand, modifying COTS expands the pool of components available to select from without requiring the technical requirements to be altered. On the other hand, by modifying the original parts, they are no longer truly "off the shelf," and any technical problems encountered throughout the modified components' lives become the sole responsibility of the modifiers [20]. In addition to the time and effort needed to make the modifications, modified COTS components may need to be re-certified by the military at a high cost[32, 90]. Furthermore, modified COTS often encounter numerous technical problems throughout their lives, causing the true cost of modifying those components to rise [74]. In essence, the act of modifying COTS components requires that some of the benefits identified in §1.3.1 be sacrificed.

The tradeoffs between the increased capability afforded by modified versions of COTS and the increased cost required to develop, certify, and sustain those components is not well understood. Without an upfront understanding of these trades, it is impossible to anticipate the expected costs or benefits of modifying COTS components [90]. This motivates Research Question 1.2:

Table 2: A Defense Science Board Task Force identified a lack of consensus about what the term "commercial systems" refers to in the literature. They created a spectrum of COTS definitions ranging from level 1 to level 8 — where lower numbers are closer to "truly" off-the-shelf [90].

Level	Description
1	DOD buys a component or system from an original equipment manufacturer — either domestic or foreign — and uses it as is.
2	DOD buys a component or system from an original equipment manufacturer and then makes minor modifications. This describes changes that do not affect functionality, such as painting it green.
3	DOD buys a component or system from an original equipment manufacturer and then makes significant modifications that affect functionality. This many include adding armored doors, weapons systems, communications systems, or a ballistically tolerant fuel system.
4	DOD buys a component or system from an original equipment manufacturer, but specifies significant modifications in the purchase agreement that are made prior to delivery.
5	DOD buys a component or system based on an existing product. System requirements drive the replacement of many subsystems with other military-specified components.
6	DOD directs a manufacturer or system integrator to modify a prototype product to meet requirements.
7	DOD directs a manufacturer or system integrator to assemble a collection of components independently qualified on different existing systems into a new system.
8	DOD specifies and purchases a product that does not yet exist, but requires commercial development and utilizes commercial plants or processes.

Research Question: 1.2
What trends emerge between reliability and cost when switching from low-level to high-level COTS?

Answering this research question requires further investigation in the form of experimentation. It is reasonable, however, to postulate that increasing modification (i.e. tending toward a higher COTS "level" as given in Table 2) will increase the up-front cost associated with modifying and/or the per-unit cost for modified components. Furthermore, since COTS component modification is often done to increase component survivability within the military environment, it is also reasonable to postulate that the failure rate of modified COTS components will decrease. Since anecdotal evidence suggests that even small reliability improvements can come at a high cost, it can be expected that the advantages of reduced failures will be overshadowed by the increased cost associated with modified COTS components. This notion is captured in Hypothesis 1.2:

Hypothesis: 1.2
The increased cost of modification will outweigh the benefits of reduced failures, thus driving up the total obsolescence cost.

2.3 Obsolescence

Even with the ability to design and deliver up-to-date COTS-based systems, obsolescence is simply unavoidable — especially in the case of long life-cycle safety-critical systems. The fast pace of technology development for commercial components, coupled with the long life cycles of military systems, means that obsolescence is inevitable. Since the military cannot opt to *not* support their weapon systems, they are forced to pay the high price of mitigating this obsolescence. The following sections aim to expand upon Research Question 2, which is repeated below:

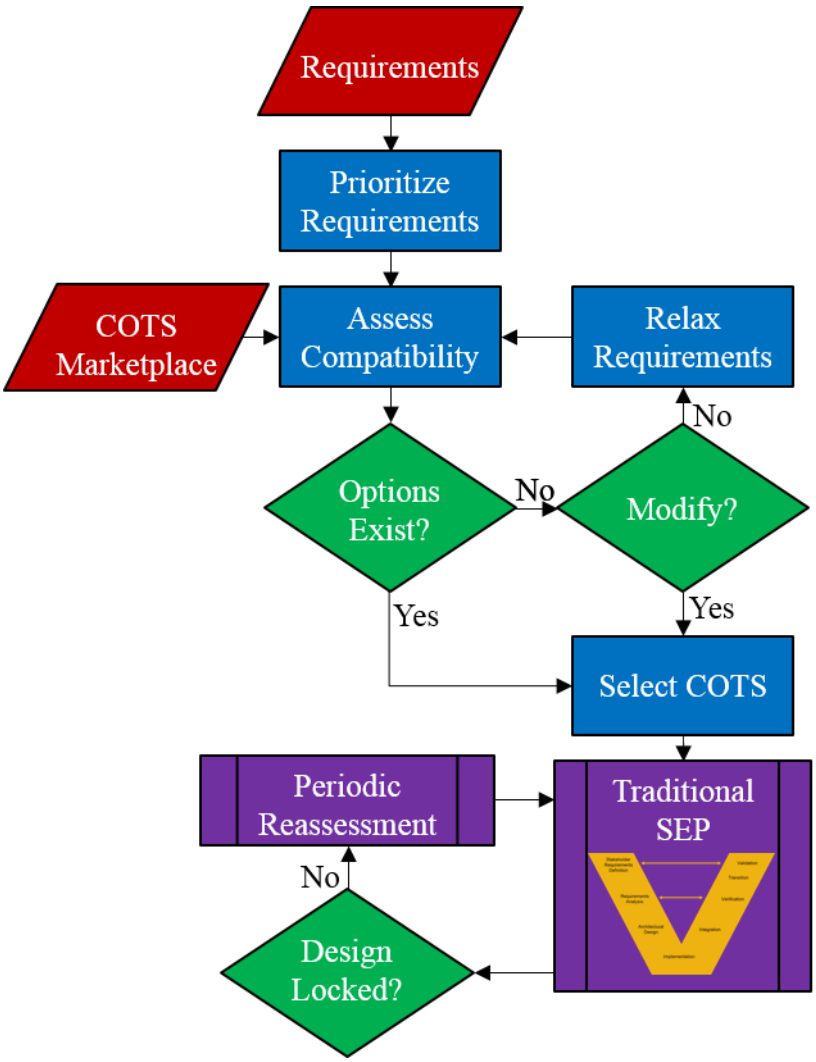

Figure 14: The CASCADE design framework describes the elements needed to design COTS-based systems from an agile systems engineering perspective. These include the need to prioritize and compare requirements against the components available in the COTS marketplace, and the option to modify COTS components to better meet requirements.

Research Question: 2

How should obsolescence mitigation strategies be incorporated into a new COTS-based design methodology?

Section 2.3.1 expands upon the concept of sustainment-dominated systems and introduces product life cycles. Next, §2.4 discusses means of forecasting when obsolescence may occur. Finally, in §2.5, common means of mitigating obsolescence are presented, and ways to optimally select obsolescence mitigation plans are introduced.

2.3.1 Sustainment-Dominated Systems

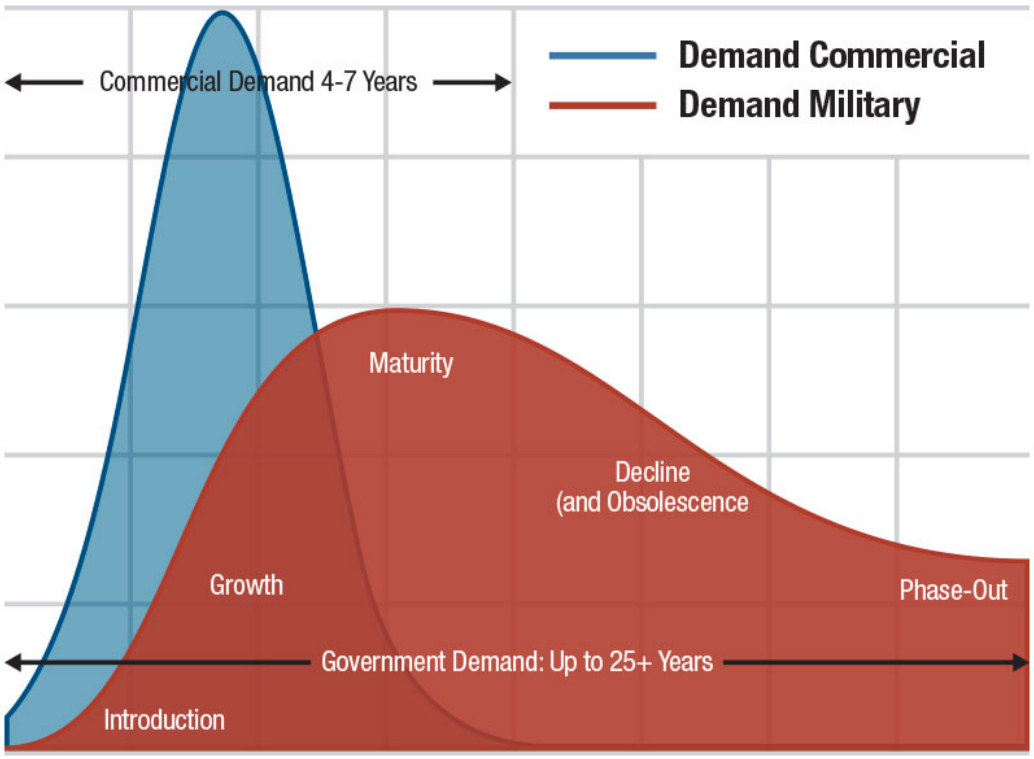

Figure 15: Typical military systems have life cycles of over 20 years, while the life cycles of COTS components can range from 4 to 7 years, or down to months [58, 66, 69].

It is important to consider that obsolescence is not an issue in and of itself. That is to say, obsolescence only *becomes* an issue when an obsolete component is in need

of repair. Therein lies the problem for COTS-based military systems: Military systems are typically designed for life cycles of greater than 25 years, while commercial components are only designed for approximately five year life cycles, as illustrated in Figure 15[65, 81, 92]. By designing these systems with life cycles that far exceed the life cycles of their constituent parts, obsolescence becomes inevitable. This life cycle mismatch is exacerbated by the extended service lives that many military systems receive. It is not uncommon for weapon systems to see service lives well over 40 years[65]. The B-52, for instance, has had its service life extended to over 94 years[65]. Figure 16 illustrates the extended lives of several military aircraft.

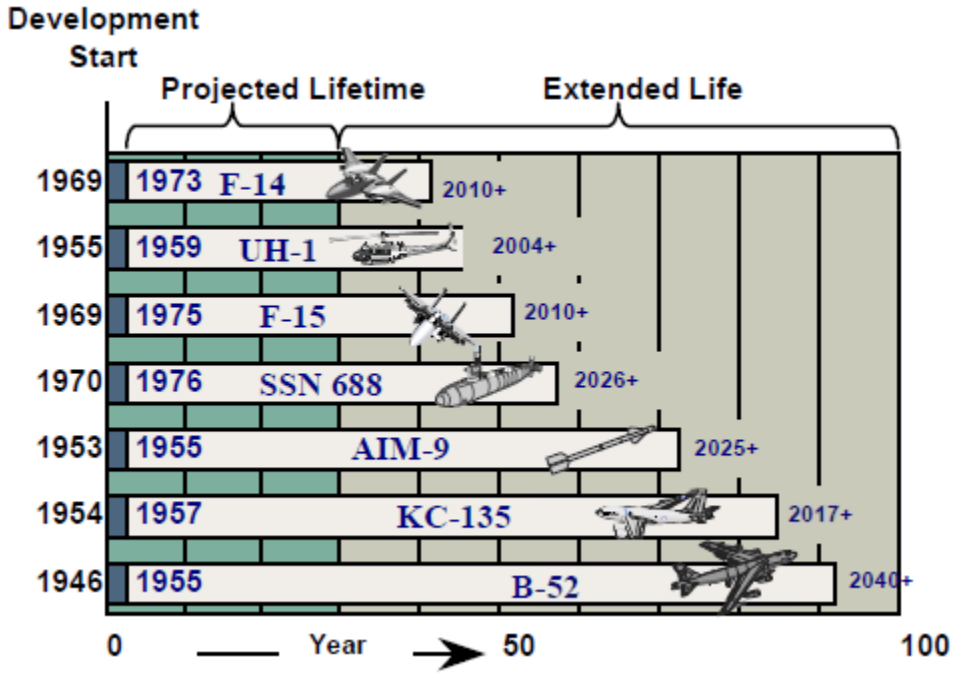

Figure 16: With fewer new weapon systems being developed, existing systems are being expected to be operational for far longer than originally planned. It is not uncommon for weapon systems to see service lives of 40+ years — or as many as 94 years as is the case for the B-52 [65]. These life cycle extensions exacerbate the problems of life-cycle mismatch associated with COTS-based systems.

The life cycle mismatch described above becomes a particularly threatening issue when the system in question is safety-critical. This means that the failure or unavailability of the system poses a safety risk to those who depend on it. Such is the case

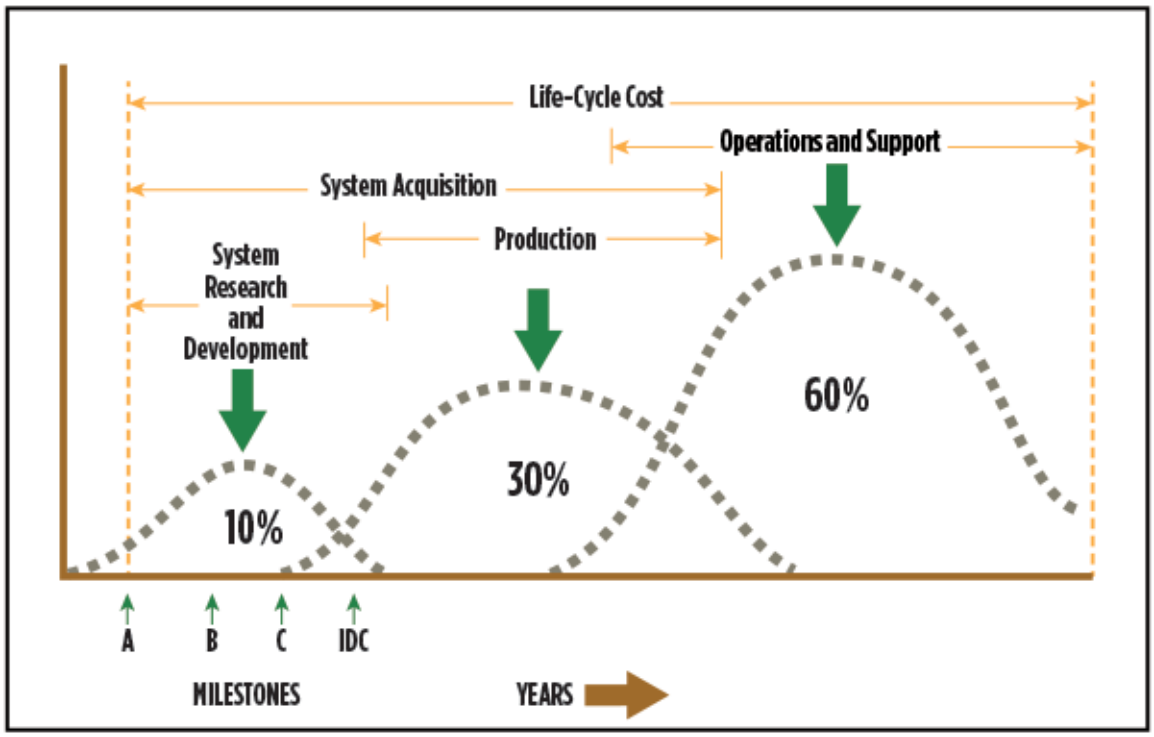

Figure 17: While only about 10% of a system's total life-cycle cost is invested during the research and development phase, decisions made *during* this phase commit about 70% of the remaining life-cycle cost. Due to the need to support long-life cycle systems, a large proportion of the total life-cycle cost is spent for operations and support [31].

for weapon systems. The inability to properly use a weapon system due to an unrepairable component poses a threat to the operator of that weapon system, as well as potentially to the nation. The unavoidable obsolescence seen by long-life cycle COTS-based military systems, coupled with their safety-critical nature, results in sustainment costs which often far exceed the initial procurement costs[21, 44, 65, 69, 84, 109]. In fact, up around 60–70% of the total life cycle cost of such systems can come from the operations and support phase [91]. Figure 17 notionally depicts a typical spending profile, illustrating how O&S costs dominate the total life cycle cost.

2.3.2 Component Life Cycle

Products tend to follow a predictable life cycle described by six common life cycle stages [79]. These stages include introduction, growth, maturity, decline, phase-out,

and discontinuance or obsolescence as illustrated in Figure 18 [79, 93]. A full component life cycle can be mathematically represented by a component life cycle curve (sometimes called a "sales curve") which takes the form of a Gaussian distribution defined by

$$f(x) = k \exp^{(-(x-\mu)^2/2\sigma^2)}$$

where k is the peak sales, μ is the mean, and σ is the standard deviation [93]. Each of the six life cycle stages represents one standard deviation of the system's life cycle, with the exception of the maturity stage which represents the period $\pm 1\sigma$ around the mean [93].

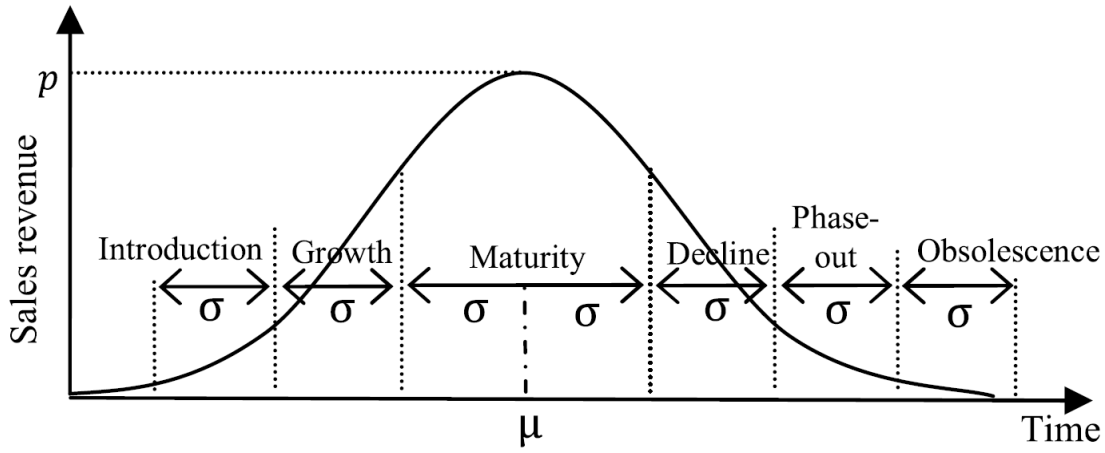

Figure 18: The product life cycle is characterized by 6 life cycle stages which correspond to changes in part sales [71].

Descriptions of the six life cycle stages are as follows [79]:

Introduction takes place from $(\mu-3\sigma, \mu-2\sigma)$ and represents debut of a new product, whether it's evolutionary or revolutionary.

Growth takes place from $(\mu-2\sigma, \mu-1\sigma)$ and is characterized by market acceptance. During this phase, the increase in sales justifies the use of specialized production

equipment, which, in turn, allows for mass production. It is here that economies of scale begin to be realized.

Maturity takes places from $(\mu - 1\sigma, \mu + 1\sigma)$ and is characterized by high-volume sales. During this phase, competitors with economic advantage may enter the market.

Decline takes place from $(\mu + 1\sigma, \mu + 2\sigma)$ and is characterized by decreasing demand and decreasing profit margins. Typically only a few specialized manufacturers remain.

Phase-out is initiated when the manufacturer decides on a date to stop production, and takes place from $(\mu + 2\sigma, \mu + 3\sigma)$. It is during this phase that manufacturers generally issue a discontinuance notice which provides a last time buy date, suggests alternative parts, and/or suggests aftermarket sources for the part.

Discontinuance begins at $\mu + 3\sigma$ and represents when the manufacturer stops production of the part.

2.4 Obsolescence Forecasting

The description of the life cycle curve in §2.3.2 describes the *general* trend of a product's life cycle, but does not specify exactly when particular events will occur. Of particular importance for this body of work is the timing of obsolescence. A key enabler in the ability to estimate life cycle costs for long-life cycle systems is the ability to forecast component obsolescence. This leads to the following sub-research question:

Research Question: 2.1
What is a suitable method of forecasting obsolescence?

To be considered "suitable," a method must match the following criteria:

1. The method must provide a numeric estimate for the obsolescence date.

2. The method must utilize data that is available to industry outsiders.

3. The method should provide confidence intervals where possible.

4. The method must not be proprietary.

There are two basic types of obsolescence forecasts: Long-term model-based forecasts, and short-term data-driven forecasts [88]. Long-term forecasts are used when obsolescence is more than a year away, and are effective tools to enable pro-active or strategic obsolescence management[1] [88]. By contrast, short-term forecasts monitor for imminent obsolescence events by searching the supply chain for precursors to obsolescence[2] [88]. On the one hand, the closer to the *actual* obsolescence date one gets, the better the forecasts become [89]. On the other hand, the closer to the actual obsolescence date one gets, the less *useful* those forecasts are [89]. Since even proactive/strategic methods will occasionally require short-term reactive mitigation, both types of forecasts will be considered for this methodology.

2.4.1 Manufacturer Inquiry

Perhaps the most obvious way to estimate the date of obsolescence is simply to ask the manufacturer. Indeed, this method has been shown to be accurate in most instances. A survey of such inquiries was conducted, and the error in each estimate was calculated via

$$\text{Error} = 100\left[\frac{D_O - D_{\text{inquiry}}}{D_{\text{MQO}} - D_{\text{inquiry}}} - 1\right]$$

[1]The terms "pro-active" and "strategic" obsolescence are fully defined later in § 2.5. For the purposes of this discussion, however, pro-active and strategic obsolescence management simply indicates that forecasted obsolescence dates are known, and future solutions have been generated.

[2]Such precursors include: reductions in number of sources for a part, reduction in distributor inventories, and announcements of upcoming obsolescence made by the manufacturer [88].

where D_O is the actual date of obsolescence, D_{inquiry} is the date the inquiry was made to the manufacturer, and D_{MQO} is the manufacturer's quoted obsolescence date [84].

Results from the survey show that 58.7% of manufacturer's obsolescence quotes are accurate. Furthermore, results show that manufacturers tend to give conservative estimates, with 25.6% of actual obsolescence events occurring *after* the quoted date, and only 15.7% occuring before. On average, the remaining production duration is 35% longer than the date quoted by the manufacturer. These results are depicted in Figure 19.

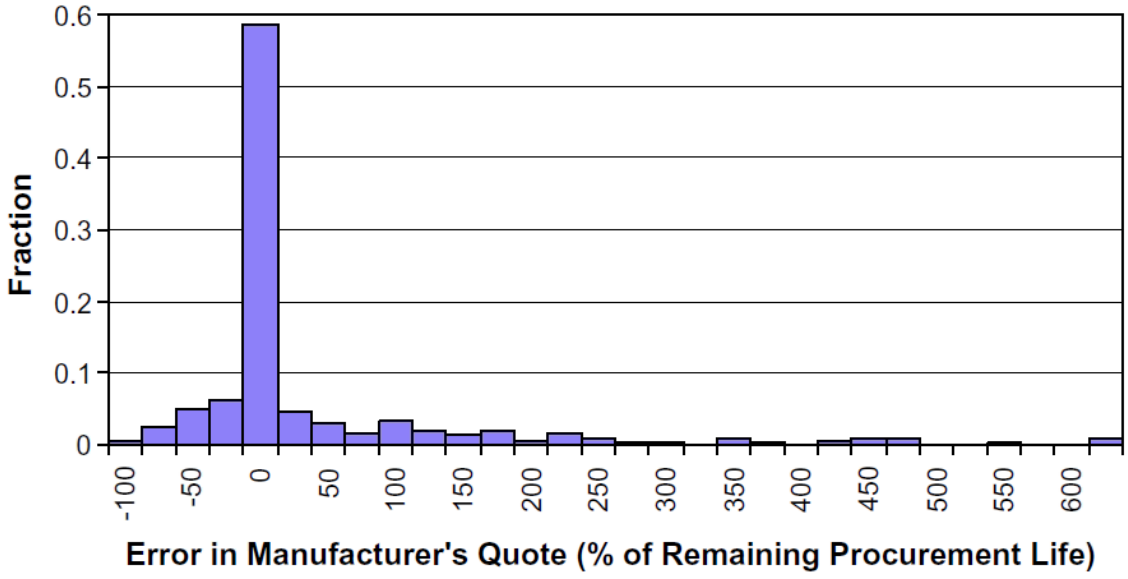

Figure 19: Error in manufacturer's obsolescence forecast[84]

While these results are promising for existing obsolescence issues, they are not of much use for dealing with the obsolescence of parts that do not yet exist. Such is the problem for long-life cycle systems — eventually the initial selection of parts will be replaced, and it is the obsolescence dates of the *new* components that becomes important. Hence, except for predicting the obsolescence of the initial set of components, this method has limited utility.

2.4.2 Sales Curve Forecasts

Solomon *et al.* present a methodology for forecasting the years to obsolescence based on sales data [93]. Their method accommodates varying fidelities of sales data, and can even forecast obsolescence for parts that have not yet been produced [93, 89]. The methodology works as follows [93]:

Step 1: Identify the Device or Technology Group — A device or technology group is a family of devices that share common technological and functional characteristics, although they may be produced by different manufacturers [93].

Step 2: Identify Primary and Secondary Attributes — The primary attribute of a device or technology group is a characteristic which defines that device or technology group [93]. For instance, memory density would be a primary attribute for memory modules. A secondary attribute is a characteristic which further classifies a product within its primary attribute [93]. For instance, a secondary attribute for memory modules might be voltage.

Step 3: Determine the Number of Sources — It is necessary to determine as many vendors of the device or technology as possible, since more data will lead to better results.

Step 4: Obtain Sales Data — Sales data can take the form of number of units sold, revenue, or percentage of market share of a given technology over time [93].

Step 5: Curve-fit Sales Data if Available — Fit a Gaussian distribution of the form

$$f(x) = k \exp^{(-(x-\mu)^2/2\sigma^2)}$$

to the sales data for the primary attribute, where k is the peak sales [93]. From this, one can extract the mean, μ, and standard deviation, σ. Alternatively, if

insufficient data exists to accurately fit a distribution to, one can estimate μ by taking the year of peak sales, and estimate σ by defining it as on-third the time interval between the product's introduction and the year of peak sales [93].

Step 6: Determine the Zone of Obsolescence — The zone of obsolescence represents the window in time during which obsolescence is predicted to occur. It is given by $(\mu + 2.5\sigma - p, \mu + 3.5\sigma - p)$, where p is the present date [93].

Step 7: Modify the Zone of Obsolescence — The above steps are repeated for the secondary attribute. If the years of obsolescence for the secondary attribute fall within $\pm 3\sigma$ years of that of the main attribute, then the zone of obsolescence will be defined by the interval (a, b) listed in Table 3.

Table 3: The "zone of obsolescence" used in Sales Curve forecasting can be found using the relationship between the parameters of the sales curve, including the mean of the primary attribute (μ), the mean of the secondary attribute (μ_j), the standard deviation of the primary attribute (σ), the standard deviation of the secondary attribute (σ_j), and the present date (p) [93].

Condition	(a, b)
$\mu_j + 3.5\sigma_j - p < \mu + 2.5\sigma - p$	$a = \mu_j + 2.5\sigma_j - p$ $b = \mu_j + 3.5\sigma_j - p$
$\mu + 2.5\sigma - p \leq \mu_j + 3.5\sigma_j - p$ AND $\mu_j + 3.5\sigma_j - p \leq \mu + 3.5\sigma - p$	$a = \min(\mu + 2.5\sigma - p, \mu_j + 2.5\sigma_j - p)$ $b = \min(\mu + 3.5\sigma - p, \mu_j + 3.5\sigma_j - p)$
$\mu_j + 3.5\sigma_j - p > \mu + 3.5\sigma - p$	$a = \min(\mu_j + 2.5\sigma_j - p, \mu + 2.5\sigma - p)$ $b = \mu + 3.5\sigma - p$

By computing the zone of obsolescence for similar components with different primary attribute values, the time-varying technology trends can be determined [89]. Using memory modules, for instance, the zone of obsolescence can be plotted as a function of memory density to show how the technology is expected to evolve over time [89]. Thus, Sales Curve Forecasting enables obsolescence forecasting for existing components *and* components that have yet to enter the market.

This method of obsolescence forecasting is useful for a wide range of applications, since it can be performed using varying types and quantities of sales data. It has also been shown to provide obsolescence forecasts with the same accuracy as commercial forecasting approaches [89]. Despite this, one of its main drawbacks is its inability to provide a quantitative confidence interval for its forecasts beyond a discrete upper and lower bound.

2.4.3 Data Mining Approach

Sandborn recognized that better forecasts with quantifiable uncertainty estimates could lead to better obsolescence mitigation, and thus lead to better cost avoidance [89]. Therefore Sandborn extends Solomon's sales curve approach described in §2.4.2 to increase its accuracy and incorporate quantifiable confidence intervals [89, 88]. He does this by considering that the "window of obsolescence" is dependent on manufacturer-specific and part-specific business practices [89]. Similar to the sales curve forecasting approach, Sandborn's model starts with historical market data — namely historical last-order dates [89]. These data are collected and sorted by manufacturer, and then for each part instance the peak sales (μ_p) and standard deviation (σ_p) are computed as in the sales curve approach [89]. Next, the last-order dates for each part instance and manufacturer are expressed as a number of standard deviations past the peak sales year for that part [89]. These results are plotted in a histogram, which represents the probability distribution of when, relative to peak sales year, the specific manufacturer issues a last-order date [89]. A Gaussian distribution is then fitted to this histogram to field parameters μ_{LO} and σ_{LO}[89]. Finally, the window of obsolescence is given by

$$\mu_p + (\mu_{LO} \pm x\sigma_{LO})\sigma_p \tag{1}$$

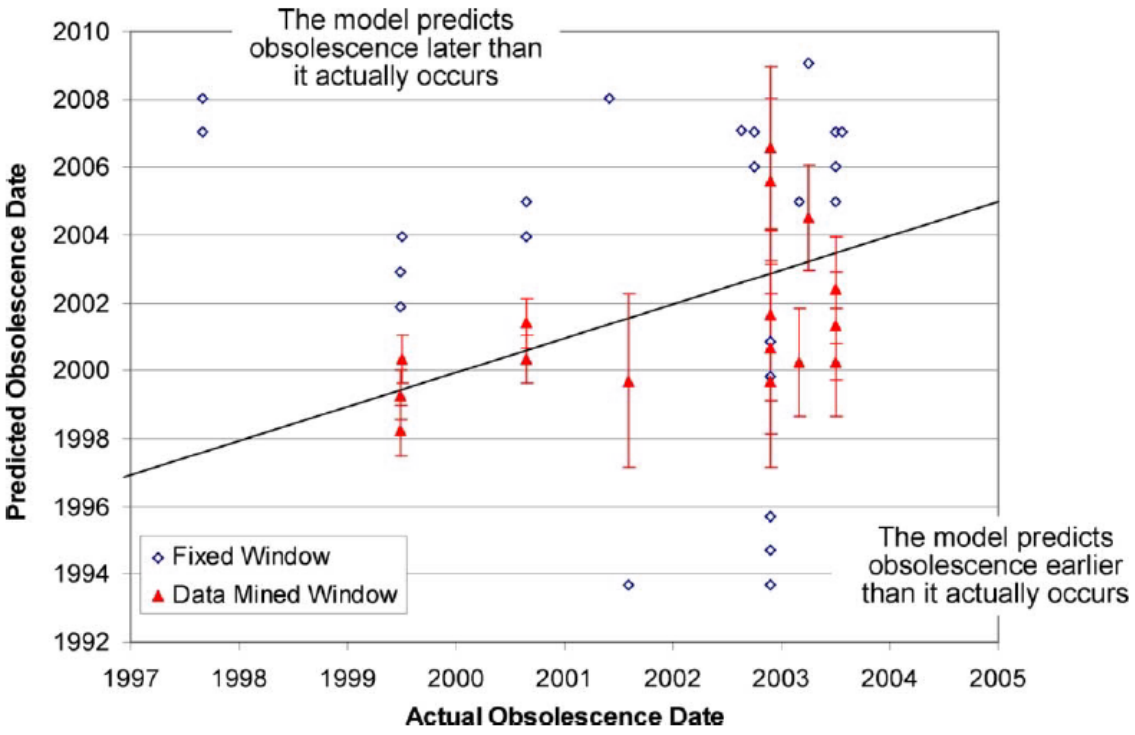

Figure 20: Actual-by-predicted plot showing the results of the sales curve forecasting method versus the data mining forecasting method [89]. The blue diamonds represent the upper and lower bounds of the obsolescence dates predicted by the Sales Curve Forecasting method. The red triangles and their error bars represent the expected obsolescence date predicted by the Data Mining approach. These results indicate that the Data Mining approach is capable of producing obsolescence predictions which are more accurate than the Sales Curve Forecasting approach.

where x depends on the desired confidence level [89]. For example, $x = 1$ represents 68% confidence, while $x = 2$ represents 94% confidence. Similar to the Sales Curve Forecasting approach described in § 2.4.2, obsolescence dates can be plotted as a function of the primary attribute to reveal the expected obsolescence of components that haven't even entered the market [89].

The data mining approach to obsolescence forecasting yields a substantial improvement in accuracy over the sales curve forecasting method presented in §2.4.2, as shown in Figure 20 [89]. Thusly, this method provides accuracy, but it does so at the cost of additional data needs in the form of last-order dates.

2.4.4 Procurement Life Modeling

The Sales Curve Forecasting method and the Data Mining approach to obsolescence forecasting both rely on the "primary" and/or "secondary" attributes of a given technology. As such, they require an abundance of data in order to be effective. Redding *et al.* call these attributes "evolutionary parametric drivers," and suggest that no such data exists for the majority of electronics [88, 84]. Procurement Life Modeling offers an indirect method for forecasting obsolescence *without* such data. Instead of using data to forecast the date of obsolescence directly, Procurement Life Modeling uses the relationship between the procurement life of a product (L_P) and that product's introduction date (D_I) to predict the date of obsolescence (D_O) [88]. Hence, if data for L_P and D_I can be obtained, then $L_P = D_O - D_I$ can be used to predict the date of obsolescence.

Fundamentally, Procurement Life Modeling seeks a relationship between L_P and D_I for a set of products of a particular type (e.g. memory modules). The required data can be obtained from product databases such as SiliconExpert™, which provides data for thousands of electronics across a range of manufacturers. From these data, a probability density function (PDF) can be generated for the procurement life, L_P, by fitting a 2-parameter Wiebull distribution given by

$$f(t) = \frac{\beta}{\eta}\left(\frac{t}{\eta}\right)^{\beta-1} e^{-\left(\frac{t}{\eta}\right)^{\beta}}$$

where parameters β and η are estimated using maximum likelihood estimation. Using this PDF, the hazard rate, $h(t)$ can be computed via

$$h(t) = \frac{f(t)}{1 - F(t)} \tag{2}$$

where $F(t)$ is the CDF [84]. The hazard rate represents the probability that a part will become non-procurable (obsolete) at time t assuming it was procurable in the

Table 4: No single obsolescence forecasting method stands out as the "best." For the proposed methodology, the selected method will depend upon the required accuracy of the forecasts. ✓= "good" ✗= "bad" ○= "neutral."

	Manufacturer Inquiry	Sales Curve Forecasting	Data Mining	Procurement Life Modeling
Captures Uncertainty	✗	✗	✓	✗
Predicts Obsolescence of Future Components	✗	✓	✓	✗
Utilizes Accessible Data	✓	✓	✗	○

period $(0, t)$. Additionally, using the parameters of the Wiebull distribution, the mean procurement life, $\overline{L_P}$ for a given type of part can be calculated via

$$\overline{L_P} = \eta\Gamma\left(1 + \frac{t}{\beta}\right) \tag{3}$$

While useful in instances where insufficient data exists, Procurement Life Modeling does not provide the same level of accuracy as sales curve forecasting or data mining. Furthermore, despite requiring *less* data than either of the other two methods, a substantial amount of data is still required. Thus, this method should only be used in the extreme case where insufficient data exists for the other two methods while still satisfying the needs of this method.

2.4.5 Summary of Proposed Methods

The obsolescence forecasting methods are all very similar in nature, though they require varying quantities and fidelities of data. As is to be expected, models requiring more data — such as Sales Curve Forecasting or Data Mining — produce results with greater accuracy. On the other hand, methods like Manufacturer Inquiry and Procurement Life Modeling may prove useful in cases where less data is available.

Research Question 2.1 seeks a "suitable" method for forecasting obsolescence. It is reasonable to postulate that an enhanced ability to forecast obsolescence will lead to more effective obsolescence mitigation plans, and hence a lower obsolescence mitigation cost. Therefore a "suitable" method for forecasting obsolescence is the one which produces the highest achievable accuracy with the available data. However, whether or not extra effort should be spent on highly-accurate obsolescence forecasts depends on how sensitive the obsolescence cost is to forecast accuracy relative to other cost and schedule parameters. Anecdotal evidence suggests that the cost of mitigating obsolescence is sensitive to obsolescence forecast dates, so it is not unreasonable to assume that obsolescence forecast accuracy will be a main driver of obsolescence cost. This is summarized in Hypothesis 2.1a below:

Hypothesis: 2.1a
The accuracy of obsolescence forecasts will be a main driver of obsolescence cost for COTS-based systems.

This is irrelevant, however, if obsolescence cost *increases* with increasing forecast accuracy. It is therefore necessary to experimentally demonstrate the non-increasing relationship between obsolescence forecast accuracy and obsolescence cost by testing Hypothesis 2.1b:

Hypothesis: 2.1b
The obsolescence cost will be non-increasing with increasing accuracy of obsolescence forecasts.

2.5 Mitigating Obsolescence

Obsolescence is virtually unavoidable for long life-cycle COTS-based systems due to the problem of life-cycle mismatch as discussed in § 2.3. In the case of safety-critical

systems, it is imperative that obsolescence be remedied as quickly as possible. This document defines the means of mitigating obsolesce on three different levels — each with its own scope. *Obsolescence mitigation actions* describe the specific solution applied to each individual instance of obsolescence. That is, obsolescence mitigation actions are applied on the component level each time obsolescence is encountered. As Section 2.5.1 describes, obsolescence mitigation actions can involve short-term solutions or more-permanent component replacements.

The collection of all obsolescence mitigation actions is called the *obsolescence mitigation plan*. While obsolescence mitigation *actions* are applied at specific instances in time to specific components, obsolescence mitigation *plans* span all COTS components in the system throughout the entire life-cycle. Each obsolescence mitigation plan is created via a specific *obsolescence mitigation strategy*, each of which falls into one of three categories as described in Section 2.5.2.

2.5.1 Obsolescence Mitigation Actions

Each instance of obsolescence must be remedied by one of several *obsolescence mitigation actions*. Each obsolescence mitigation action can be broadly grouped into one of two categories: Redesigns (or replacements), and "Bridging Actions." As the name suggests, redesign involves designing an obsolete component out of the system [65]. This option is often not desirable since it is typically costly, and it tends to cannibalize design resources that could otherwise be used for designing new products [21]. Although redesigns represents long-term or permanent solutions to obsolescence, they may be infeasible due to scheduling constraints, or inviable due to their high costs. When this is the case, obsolescence must be remedied via a bridging actions.

The purpose of bridging actions is to bridge the gap between when obsolescence is encountered and when obsolescence can be handled more permanently. Therefore, unlike redesigns which take place at discrete dates, bridging actions take place over a

range of dates. The term "Bridging Action" is not a specific resolution in and of itself, but rather a term which describes several possible actions. Several specific bridging action commonly found in the literature are described below:

Reclamation — Salvaging the part from equipment that is beyond economical repair, from decommissioned systems, or otherwise unused systems [69].

Substitution — This solution involves analyzing the characteristics of the obsolete item, and attempting to locate a similar part [65]. This option requires that the substitute component have the same form, fit, and function as the original [21, 53].

Aftermarket — Aftermarket sources are firms that purchase the stock of obsolete components, or who buy obsolete production lines to maintain the ability to reproduce obsolete products [65, 69]. These aftermarket products can be purchased and used to fulfill the demand for obsolete components.

Emulation/Cloning — This involves developing replacements for the obsolete parts using state-of-the-art techniques [65, 69]. With this option, there is a risk that the new part will fail to meet unspecified performance characteristics of the original item [65].

Life-of-Type/Lifetime Buy — This alternative involves purchasing a final order of the obsolete component(s) to sustain the system over a period of time — possibly for the remainder of the system's life[7, 65]. This is a high-risk option since a poorly-planned purchase quantity can have devastating consequences [53]. If too many components are purchased, those components must be stored and maintained in inventory, costing both money and resources. If too few components are purchased, then the system will not be supportable over its full life cycle.

Table 5: Costs for various obsolescence mitigation actions [65, 69]. Note that Life-of-Type (LOT) Buy cost depends on the timing of obsolescence and the duration of the LOT buy, so no value is listed.

Resolution	Low	Average	High
Existing Stock	$0	$0	$0
Reclamation	$629	$1,884	$3,249
Substitute	$5,000	$18,111	$16,500
Aftermarket	$15,390	$47,360	$114,882
Emulation	$17,000	$68,012	$150,000
Redesign (Minor)	$22,400	$111,034	$250,000
Redesign (Major)	$200,000	$410,152	$770,000
Life of Type Buy	N/A	N/A	N/A

Typical values for the non-recurring engineering cost for each of these actions are listed in Table 5. Typical costs and durations for these actions are also shown in Figure 21.

2.5.2 Obsolescence Mitigation Strategies

Obsolescence mitigation strategies fit into three broad categories: Reactive, Pro-active, or Strategic [86, 53, 88, 87, 84, 109, 21, 65]. As the name suggests, reactive strategies work by reacting to obsolescence *after* it has occurred. Pro-active solutions focus on predicting when obsolescence will occur and pre-planning the optimal solution to deal with each obsolescence event. Strategic solutions combine the two by utilizing reactive strategies as short-term solutions until better long-term solutions can be applied. Figure 22 depicts the relationship between these three obsolescence mitigation strategies. These three methods are discussed in detail below.

2.5.2.1 Reactive Obsolescence Management

Reactive strategies focus on dealing with obsolescence *after* it has appeared [65, 53]. Such strategies focus more on quick solutions with short-term gains, rather than long-term solutions [65]. Peter Sandborn suggests that reactive strategies can lead to "death by a thousand cuts" as managers spend their time and resources making

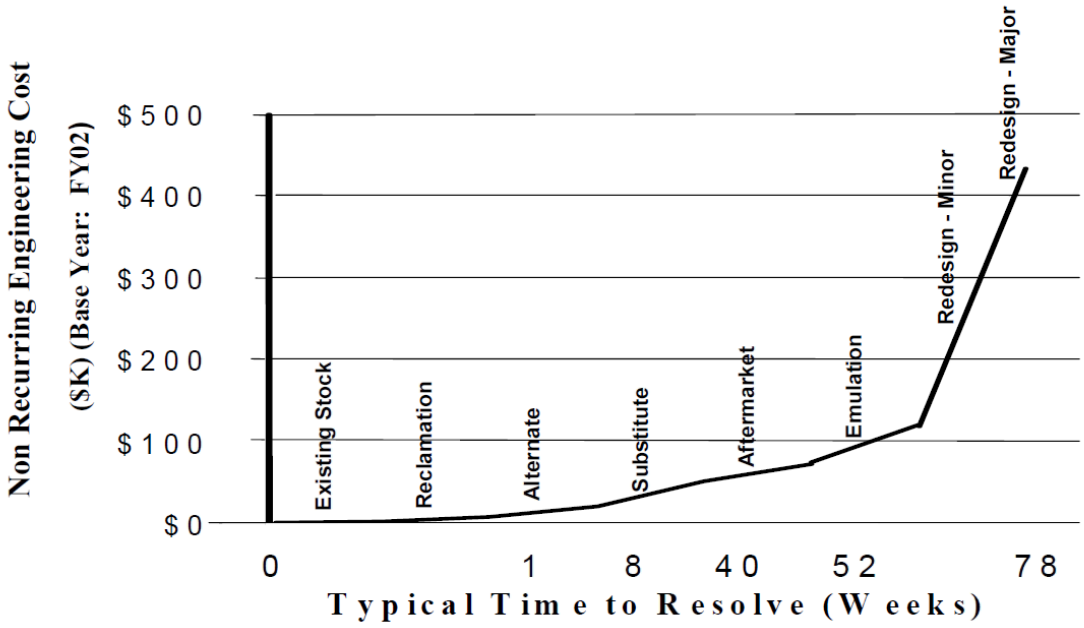

Figure 21: Non-recurring engineering costs and durations for various reactive obsolescence management strategies [7]

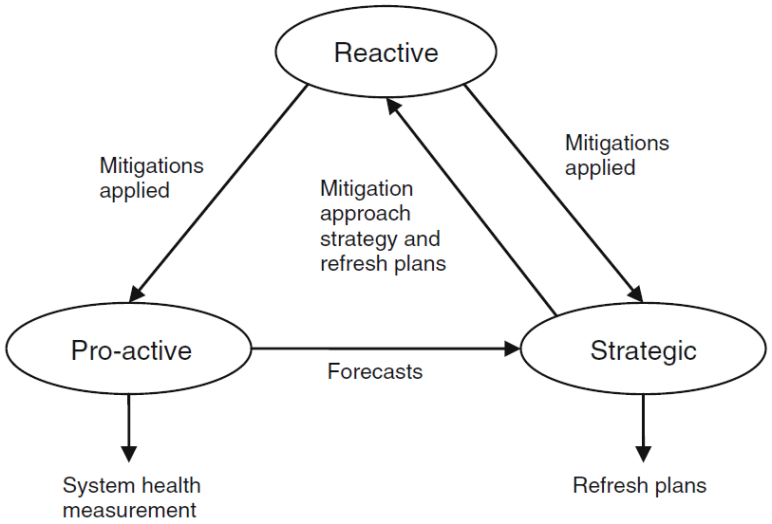

Figure 22: Obsolescence mitigation strategies can take place on three levels: Reactive, pro-active, and strategic [88]. Although strategic obsolescence mitigation offers the most cost savings, there will always be a need for reactive strategies.

innumerable independent obsolescence mitigation decisions [87]. Despite this, most obsolescence mitigation measures are reactive in nature [65, 86, 84, 21]. Reactive measures will always be necessary in the event of unexpected obsolescence, however more savings is possible via pro-active or strategic solutions [86, 87, 65].

2.5.2.2 Pro-active/Strategic Obsolescence Management

Unlike reactive strategies, which deal with obsolescence after it has become an issue, pro-active strategies focus on predicting obsolescence and finding optimal solutions *before* it becomes an issue [65, 87, 84, 109]. To do this, pro-active strategies require the determination of obsolescence risk for each component of the system, knowledge of current inventory levels, and status of spares [87]. Ultimately, however, pro-active strategies simply entail forecasting obsolescence events and pre-planning which specific obsolescence mitigation actions will be used to handle each of them. Thus, a key enabler for pro-active management of obsolescence is the ability to forecast the future obsolescence of parts [87, 109].

Strategic solutions combine elements of reactive and pro-active methods to enable greater cost avoidance [87]. The most common way of doing this is through Design Refresh Planning [109]. It works by coordinating multiple individual obsolescence mitigation actions in a single event called a "design refresh" [87, 109]. Design refreshes take place at pre-determined times, and act as long-term obsolescence management solutions [65]. Since obsolescence events cannot be expected to fall in line with pre-defined design refreshes, it is still necessary to carry out short-term obsolescence solutions until long-term solutions can be applied at the next refresh [65, 87]. Design refresh planning relies on the optimal mix of reactive and pro-active solutions, so key enablers include the ability to optimally plan design refreshes and to accurately forecast obsolescence [87, 84, 109].

2.6 Methods for Optimizing Obsolescence Management

While using pro-active or strategic obsolescence mitigation strategies reduces the cost of obsolescence events by forecasting their occurrence, reactive actions are still necessary to deal with unanticipated obsolescence.[86, 87, 65]. Therefore, in support of Research Question 2, it becomes necessary to find a method, or set of methods, which is capable of optimizing the obsolescence mitigation process. This leads to the following sub research question:

Research Question: 2.2
What is a suitable method to generate optimal obsolescence mitigation plans that minimize obsolescence cost?

To be considered, a method must:

1. Consider multiple obsolescence mitigation actions such as life-of-type buys, re-designs, emulation, etc.
2. Capture temporal cost elements, such as the time-value of money or time-related penalties for storing and maintaining inventory
3. Consider the obsolescence of multiple components simultaneously
4. Adopt the end-user/buyer's perspective
5. Capture the relationship between component reliability and cost to address the tradeoffs associated with component modification

This section describes a number of known methods for selecting and/or optimizing obsolescence mitigation plans.

2.6.1 Newsvendor Model

A classic problem known as the "newsvendor problem" can be used to model one-time business decisions such as life-of-type buy purchases at the end of a production run [49, 80]. The newsvendor problem is the problem of deciding the size of a single order to make under uncertain demand, when there is a cost associated with both overbuying and underbuying [49, 80, 40]. For instance, purchasing too much stock (overbuying) leads to unused units, which must then be sold or recycled at a loss. Buying too few units (underbuying), on the other hand, means that costly corrective measures must be taken to ensure the system can be supported. The object of the newsvendor problem is to minimize the total cost of overbuying or underbuying stock.

Let c denote the cost of a product, let $p > c$ denote the selling price of the product, and let $s < c$ denote the salvage value of the product [40]. The cost of overbuying one product is $c_o = c - s$, where c is the unit cost and s is the salvage cost. Similarly, the cost of underbuying one product is $c_u = p - c$, which represents the profit forfeited when one too few products are available to sell. Let the demand, D, follow a probability distribution for which $E[D] = \mu$ and $V[D] = \sigma^2$ [40]. If the decision variable, Q, represents the quantity of products ordered, then the expected loss, L, due to overstock and understock costs is

$$L(Q) = \overbrace{c_o \int_{-\infty}^{Q} (Q - x) f(x) dx}^{\text{Expected Cost of Overbuying}} + \overbrace{c_u \int_{Q}^{\infty} (x - Q) f(x) dx}^{\text{Expected Cost of Underbuying}} \tag{4}$$

By setting the derivative of Equation 4 equal to zero, the optimal purchase quantity, Q^* is found to be

$$Q^* = F^{-1}\left(\frac{c_u}{c_o + c_u}\right) \tag{5}$$

where $F(Q)$ is the cumulative distribution function of the demand[3] [80]. Note that no assumption was made as to the exact nature of the demand distribution, $f(x)$, thus the above result holds for any distribution.

Equation 5 provides reasonable results in some business contexts such as retail, but presents a few gaps when applied to a lifetime buy decision [80, 49]. The classic newsvendor problem, as described above, assumes that the time between purchasing and selling/using products is relatively short. However, for life-of-type (LOT) buys components are purchased, placed into inventory, and used over the course of years [80]. Thus, when applied to LOT buys, the newsvendor problem fails to account for the cost of storing and maintaining goods in inventory, and does not take into account the time-value of money [80]. Peter Sandborn adjusts for this by redefining the total understock cost as $C_{u_i} = \sum_{j=1}^{y} q_{i,j} \frac{c_u}{(1+r)^{j-1}}$, where r is the discount rate, and the quantity for the i^{th} discrete demand in the j^{th} year is given by

$$q_{i,j} = \begin{cases} \lceil \frac{D_i - Q}{y} \rceil & : (D_i - Q) \geq (y - j + 1)\lceil \frac{D_i - Q}{y} \rceil \\ (D_i - Q) - (y - j)\lceil \frac{D_i - Q}{y} \rceil & : (y - j)\lceil \frac{D_i - Q}{y} \rceil < (D_i - Q) < (y - j + 1)\lceil \frac{D_i - Q}{y} \rceil \\ 0 & : 0 < (D_i - Q) \leq (y - j)\lceil \frac{D_i - Q}{y} \rceil \text{ or } 0 > (D_i - Q) \end{cases} \tag{6}$$

Figure 23 depicts the results of a newsvendor optimization under varying discount rates when $c_o = \$2$ (year 1 dollars), $c_u = \$28$ (year 1 dollars), $\mu = 1000$ parts, and $\sigma = 50$ parts. The optimum lifetime buy quantity with a discount rate of $r = 5\%$ is found to be $Q^* = 1075$ parts, compared to $Q^* = 1038$ parts when and $r = 0\%$ [80].

Despite Sandborn's adjustment for the time-value of money, the newsvendor problem leaves gaps when compared to the criteria listed at the beginning of this section. Firstly, the classic (and modified) newsvendor problem only explicitly optimize the

[3]For a full derivation of this result, see sources [40] or [80]

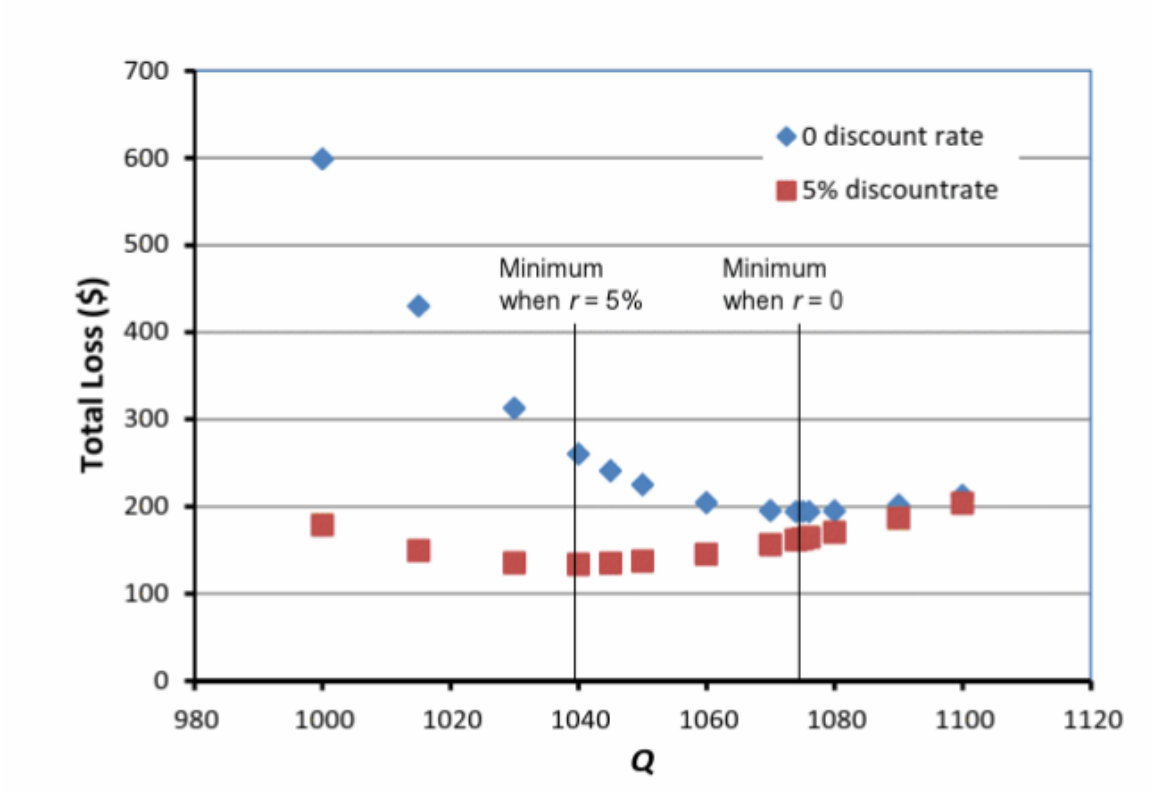

Figure 23: Results of newsvendor optimization with $c_o = \$2$ (year 1 dollars), $c_u = \$28$ (year 1 dollars), $\mu = 1000$ parts, and $\sigma = 50$ parts. Results are shown for discount rates of $r = 5\%$ and $r = 0\%$ [80].

lifetime buy strategy[4]. This also means that the newsvendor problem only acts reactively. Secondly, the newsvendor problem only considers the obsolescence of one part at a time, whereas the ideal method should consider the optimal strategy for a collection of components at once. Finally, the newsvendor problem only optimizes cost, and does not include any parameters which address the components' reliability.

2.6.2 Teunter and Fortuin Model

Ruud Teunter and Leonard Fortuin developed a model which seeks final orders which minimize accumulated costs through the end of life for complex machines [101]. They use marginal analysis to develop an explicit formula which yields a close-to-optimal

[4]Note that the one *could* adjust c_u to account for penalty associated with using one of the other strategies in the event of insufficient inventory, but the newsvendor problem fundamentally still only produces the optimal purchase quantity.

final-order quantity (LOT buy quantity), $\hat{n}$ [101]. They also use stochastic dynamic programming to yield the actual optimal buy quantity, n^* [101].

To find the actual optimal solution, their model iterates through the product life cycle and accumulates the production, holding, removal, and shortage costs which contribute to sustainment at each step [101, 36]. For each part's final-order quantity, n_i, the objective is to minimize the accumulated cost given by the equation below, which is reproduced from [36]

$$a^{j_i-1}c_i n_i + \mathrm{E}_{s_j,D_j}[(\sum_{j=j_i}^{T} a^{j-1}(\frac{h_i}{12}S_{j(i)} + ap_i(S_{j(i)} + (s_j - d_j)q_i)^- + a^T r_i(S_{T(i)} + (s_T - d_T)q_i)^+))] \tag{7}$$

where,

a Function of the discount factor ($e^{-R/12}$), R = time in years from the start date

c_i Initial purchase cost of the part i (present when $t = t_i$)

n_i Final order purchase quantity for part i at the beggining of time step 1

s_j Supply of system parts (quantity distribution), in j^{th} time step

d_j Demand of system parts (quantity distribution), in j^{th} time step

D_j Date corresponding to the current time step j

h_i Holding cost for part i (present when $t > t_i$)

$S_{j(i)}$ Stock at the beginning of interval j for part i; $S_1 = n_i$

p_i Penalty cost of part i if it is obsolete but available from alternative sources

p_{su} Penalty cost of system if any of its parts are unavailable from all possible sources

r_i Removal/residual cost of part i (parts removed at the end of life)

j Index of the current time step

T Time

t_s Time step (in years)

q_i Instances of part i in a single system.

Teunter and Fortuin's model closes some of the gaps of the newsvendor problem, but it remains inadequate for strategic obsolescence mitigation. For instance, their model only explicitly captures the LOT buy action — although the penalties associated with other bridging actions could be roughly captured by varying p_{su}. Furthermore, unlike in the newsvendor problem, Teunter and Fortuin's model adopts the seller's perspective, which does not resemble the DoD's perspective closely enough [101]. Finally, the systems engineering methodology proposed herein requires considerations for component modification, and must therefore capture both component cost and reliability. Thus, Teunter and Fortuin's model is insufficient as presented.

2.6.3 Porter's Design Refresh Model

Perhaps the simplest model for life-cycle planning is provided by Porter's Design Refresh model [80]. Porter's model addresses the decision of when to initiate a redesign given an obsolescence event by providing rules-of-thumb for engineering teams [80, 39]. The model considers two facets of the problem: First, it uses net present value (NPV) analysis to consider the "actual" cost of redesigning a system [39]. Second, the model considers the cost of holding life-of-type buy inventory over long periods of time [39]. The sum of these two values yields the total cost of the redesign and LOT buy.

Porter's analysis shows the trade-off between redesign cost and life-of-type buy cost when deferring a redesign to a later date [80, 39]. As the timing for a redesign is delayed, the net present value of the non-recurring redesign cost decreases, making it a more economically viable option [80, 39]. At the same time, as the timing for a redesign is delayed, the number of components required to sustain the system *until* the

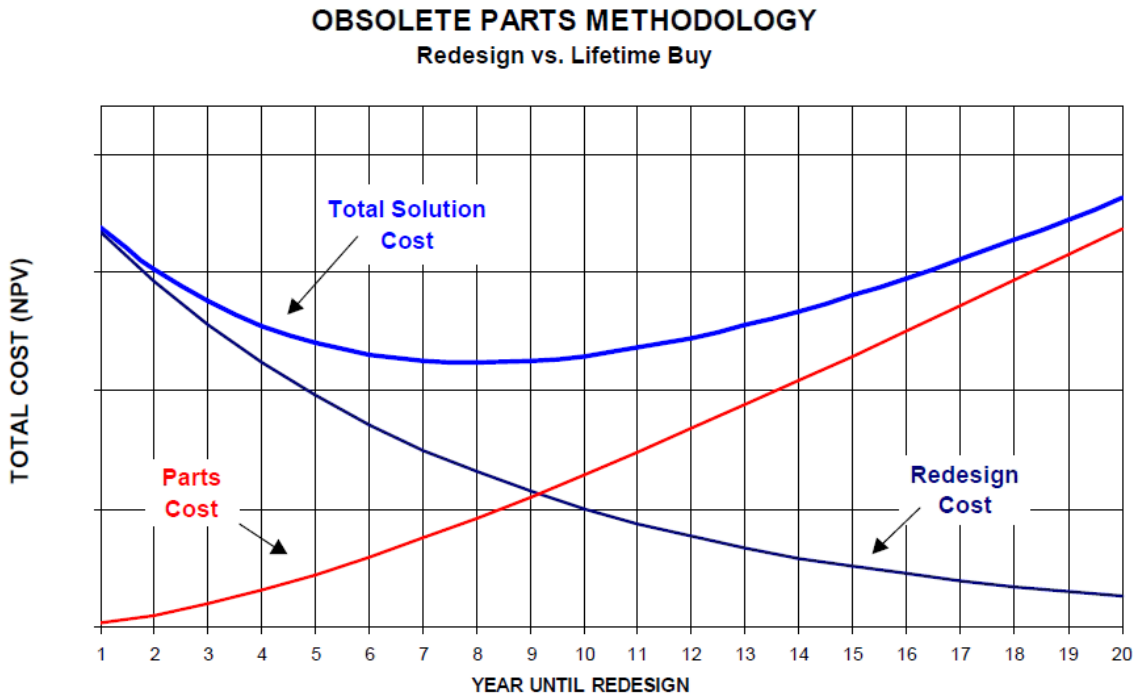

Figure 24: As the timing of a redesign is deferred to a later date, the net-present-value of the redesign decreases. At the same time, the number of components required to sustain the system through the next redesign increases, as does the holding cost of those components. The total cost of the redesign/bridge-buy is the sum of the two curves [39].

redesign increases, as does the cost of storing those components. [80, 39]. Thus, for a given non-recurring redesign cost and a component unit cost, one can find the optimal timing for a redesign as illustrated in Figure 24. Sandborn generalized Porter's model to show the optimum refresh year, Y_R to be

$$Y_R = \frac{1}{-r} \ln\left(\frac{P_0 Q}{r C_{DR_0}}\right) \tag{8}$$

where r is the weighted average cost of capital, P_0 is the purchase price of the part in year 0, Q is the number of parts needed each year, and C_{DR_0} is the design refresh cost in year 0 [88].

To provide some basic guidelines for engineering teams faced with a redesign decision, Porter also plots the locus of points which minimize the total cost for varying

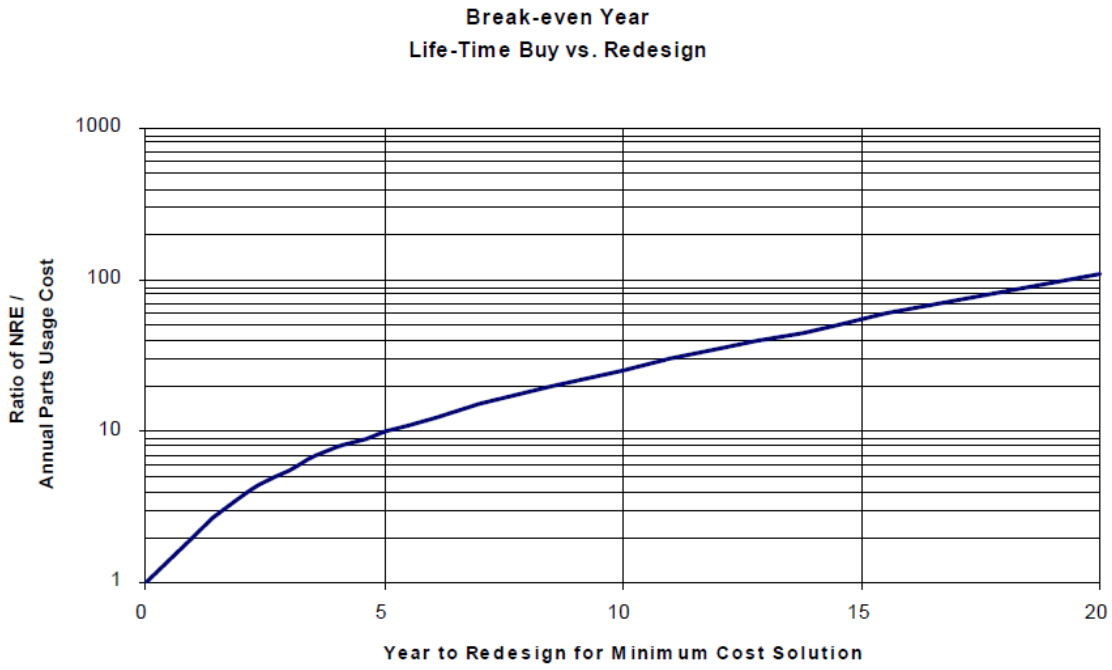

Figure 25: Locus of points representing the optimal redesign time for varying ratios of non-recurring engineering cost (for redesign) to annual parts usage costs [39].

levels of non-recurring engineering cost and annual parts usage cost as shown in Figure 25 [39]. From these results, Porter shows that if the non-recurring engineering (NRE) cost for the redesign is less than or equal to the annual parts usage cost, then a redesign should be initiated immediately [39]. Furthermore, Porter concludes that if the ratio of NRE redesign cost to annual parts usage cost exceeds 200, then a LOT buy should be used [39].

Porter's model provides useful insight and rules-of-thumb associated with redesign decisions, but it is still insufficient for the methodology developed herein. The original form of the model only deals with the obsolescence of one part at a time, though it can be extended to treat multiple parts by reapplying the model after each redesign to predict the next redesign [39, 80]. Still, even this extended model cannot capture the coupled effects associated with multiple redesigns [80]. Furthermore, Porter's model only considers the trades between two obsolescence mitigation actions: LOT buys and redesigns. Despite this, its straightforward presentation and transparent

calculations make it a reasonable starting point in the creation of a new method.

2.6.4 Cattani and Souza Model

Cattani and Souza use a version of Porter's design refresh model, in conjunction with elements of the newsvendor problem, to show the benefits of delaying an end-of-life purchase [26, 80]. Their model consists of two parts: First, they show the expected profit for an end-of-life build[5], and use that result to derive the optimal build quantity. Second, they use the previous result to show the benefit of delaying an end-of-life build decision. They show that if the price of components is non-increasing, then the optimal end-of-life build quantity, Q^* is given when

$$\frac{d\Pi_{(1,N)}(Q^*)}{dQ} = \sum_{i=1}^{N} \alpha^{i-1}[(p_i - p_{i-1})(1 - F_{(1,i)}(Q)) - hF_{(1,i)}(Q)] - c = 0 \qquad (9)$$

where $\Pi_{1,N}$ represents the profit between periods 1 and N, p_i is the price at period i, $F_{(1,i)}$ represents the CDF of demand in periods 1 through i, h represents the holding cost, and α represents the discount factor [26].

To determine the benefit of delaying the end-of-life purchase, Cattani and Souza compare $\Pi_{(1,N)(Q_1^*)}$ to $\Pi_{(d+1,N)(Q_{d+1}^*)}$, where d represents the delay [26]. They further show that the delay benefit, $B(d)$, is nondecreasing in d as shown in Figures 26 and 27[26]. Note that this result agrees with the results offered by Porter's design refresh model (Figure 25).

As with Porter's model, Cattani and Souza's model provides useful insights, but it still fails to capture the necessary aspects for the methodology proposed herein. For instance, it fails to capture multiple obsolescence mitigation options, and instead focuses on LOT buy optimization. It also does not capture the obsolescence of multiple parts at once, making it insufficient for multi-component systems. Finally, although

[5]The nature of this formulation is not important to the discussion that follows. To view this formulation, see [26]

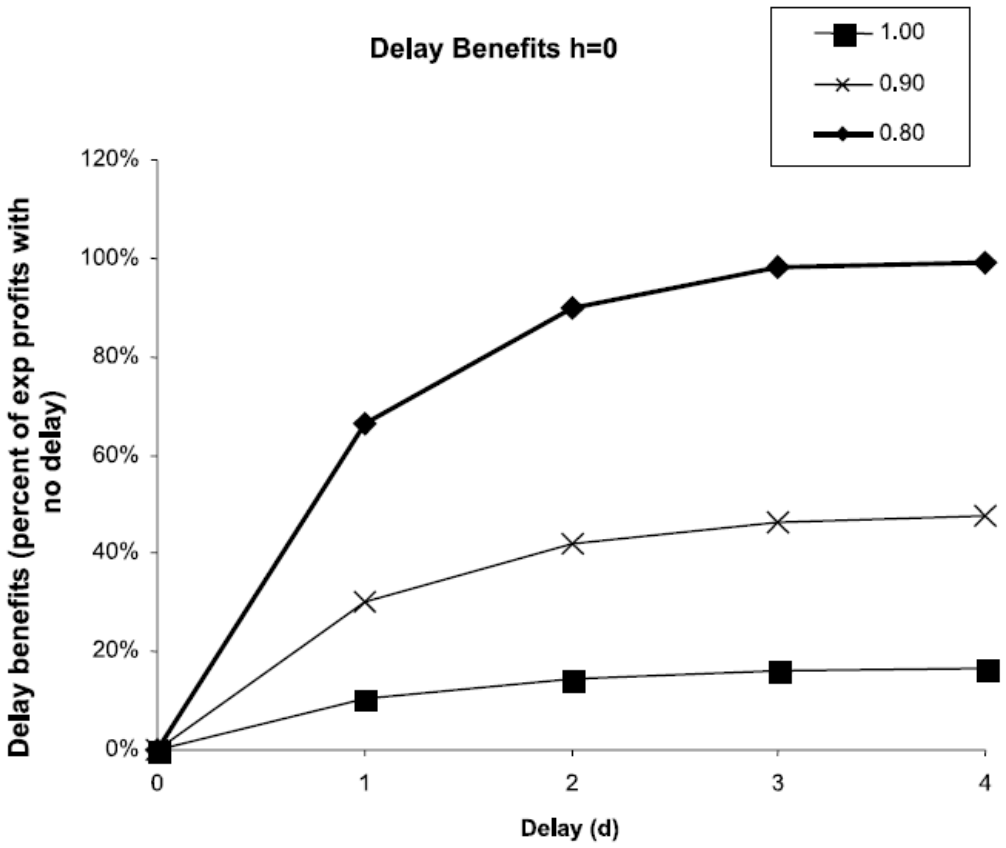

Figure 26: Results of Cattani and Souza's model for showing the benefits of delaying an end-of-life purchase. Results are shown for no holding cost ($h = 0$). Varying discount factors are shown for each, as noted in the legend [26].

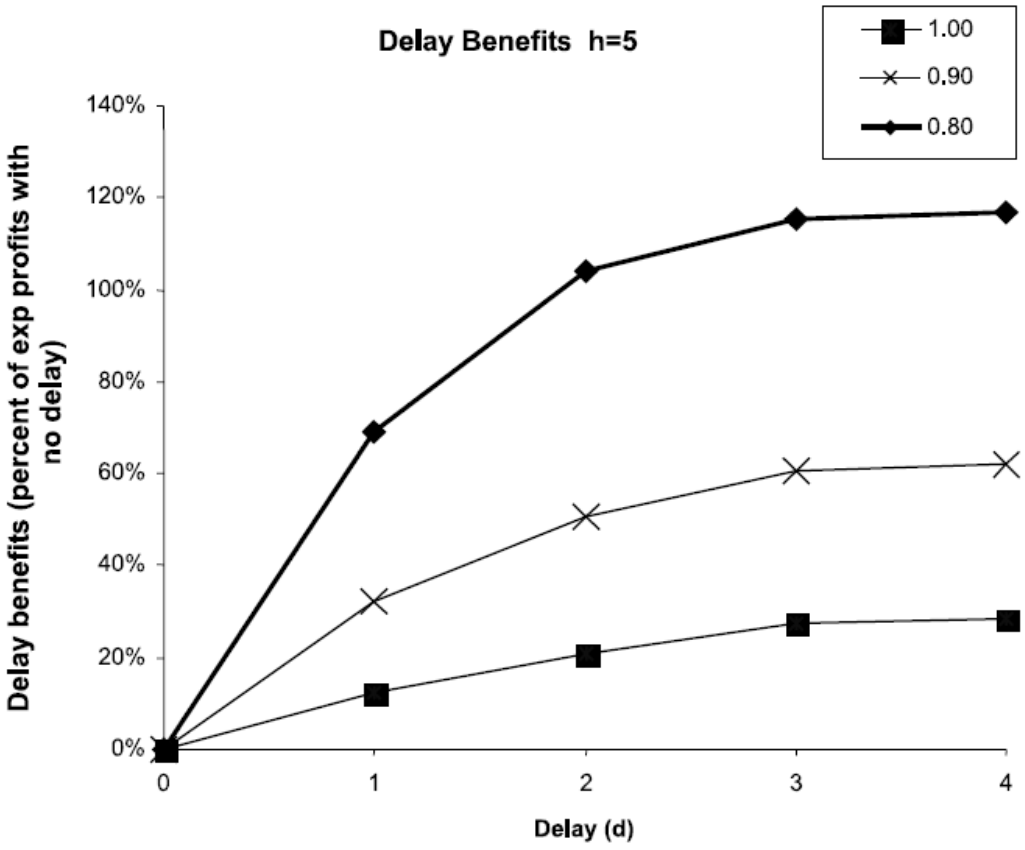

Figure 27: Results of Cattani and Souza's model for showing the benefits of delaying an end-of-life purchase. Results are shown for a holding cost of \$5 ($h = 5$). Varying discount factors are shown for each, as noted in the legend [26].

their model captures the trades associated with delaying a LOT buy decision, it does not capture the trades between component reliability and cost which must be considered for component modification.

2.6.5 Bradley and Guerrero Model

While other models tackle obsolescence mitigation for existing designs, Bradley and Guerrero consider the obsolescence-related life-cycle cost trades associated with *designing* a system [21]. They suggest that the optimal mitigation strategy is irrelevant if the initial design is highly subject to obsolescence to begin with [21]. For instance, a design which requires a large number of obsolescence mitigation actions will see reduced profit compared to a more durable design, even if those mitigation actions are optimally selected[21]. They do this by modeling proactive obsolescence management while considering technology evolution and the availability of electronic parts throughout a system's life [21, 65].

Bradley and Guerrero's model revolves heavily around the notion of product durability. They define durability, τ, to be "the duration of time over which a product design remains viable, such that all parts required by that design are readily available from original manufacturers"[21]. Thus, a product is considered *durable* if none of its parts are obsolete by the end of its life cycle[6], or *non-durable* if at least one part is obsolete by the end of its life cycle [21]. In general, choosing a product which is early in its life cycle results in greater durability, but typically at a higher per-unit cost [21]. The general form for the profit, Π_N, of a non-durable design is given by

$$\Pi_N = \int_0^L \left\{ \int_0^\tau (\pi - c_N^1) q_t e^{-rt} dt + \int_\tau^L (\pi - c_N^2) q_t e^{-rt} dt - C_O e^{-r\tau} \right\} f(\tau) d\tau - C_N \quad (10)$$

where L is the life-cycle duration of the system, $t \in [0, L)$ is time, τ is the durability

[6]The authors note that a truly "durable" product in this sense is virtually unachievable and that the notion of a fully-durable design simply acts as a baseline for comparing non-durable designs [21].

(time until part obsolescence), π is the unit revenue, c_N^1 is the unit manufacturing cost of the product before obsolescence, $c_N^2 > c_N^1$ is the unit manufacturing cost after obsolescence, q_t is the instantaneous demand rate at time t, r is the discount factor, C_O is the fixed cost of obsolescence mitigation, and C_N is the fixed cost of the original non-durable design [21]. For comparison, the profit, Π_D, for a fully durable design is given by

$$\Pi_D = \int_0^L (\pi - c_D) q_t e^{-rt} dt - C_D \tag{11}$$

where c_D is the unit manufacturing cost of a durable product, and C_D is the fixed cost of the initial durable design [21]. Using Equations 10 and 11, the difference in cost is given by $\delta(\mathbb{E}\tau, T_g) = \Pi_D(T_g) - \Pi_N(\mathbb{E}, T_g)$, where T_g is the duration of growth for the demand of the product [21]. Finally, they define the level set, $\mathscr{L}(\mathbb{E}\tau, T_g)$ as

$$\mathscr{L}(\mathbb{E}\tau, T_g) = \{\mathbb{E}\tau, T_g | \delta(\mathbb{E}\tau, T_g) = 0\} \tag{12}$$

as noted in [21].

Bradley and Guerrero use the level set defined by Equation 12 to show the boundary between optimal durable and optimal non-durable designs, as illustrated in Figure 28[21]. They show how this boundary shifts in response to changes in design parameters such as pre-obsolescence unit cost (c_N^1), post-obsolescence unit cost (c_N^2), fixed cost of obsolescence mitigation (C_O), and expected durability ($\mathbb{E}\tau$) [21]. Furthermore, they show how this boundary changes in response to various business scenarios, such as changes in demand rate over time [21].

Bradley and Guerrero's model offers two main beneficial elements for the purposes of the methodology developed herein. For instance, it considers the tradeoffs between component cost and component durability which, in part, captures the costs and benefits associated with moving from "low level" COTS to "high level" COTS [90]. Secondly, it considers the time-value of money, which is important for the long

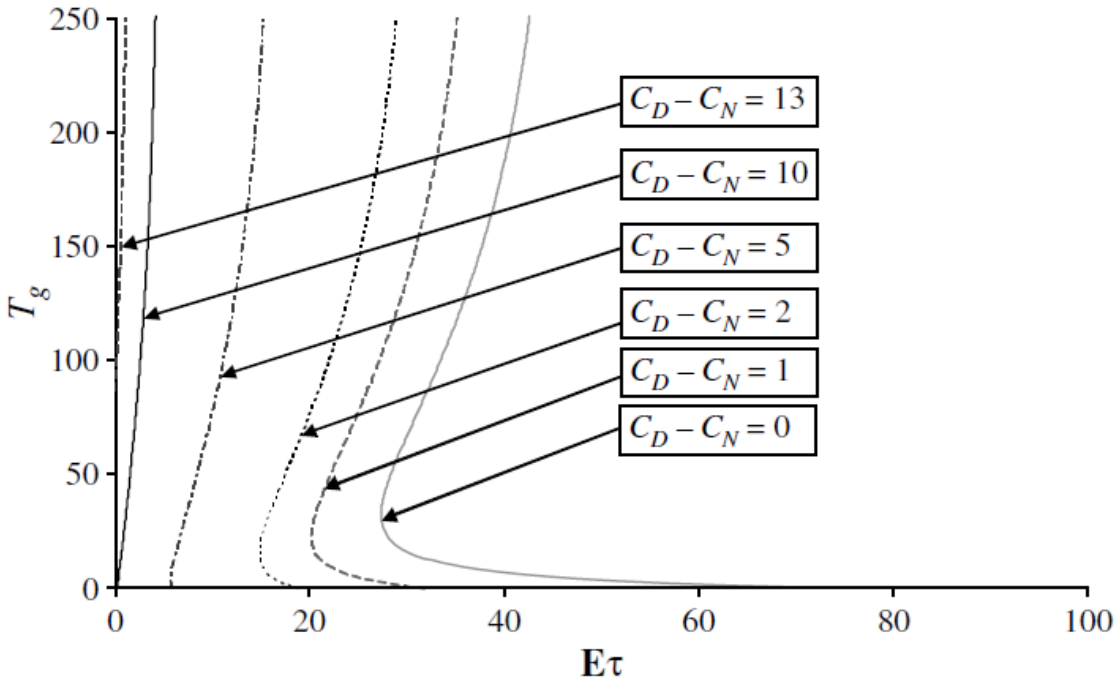

Figure 28: Sample results from Bradley and Guerrero's model showing the shifting boundary as a function of non-recurring engineering cost differential [21].

life cycles of military systems. There are, however, a few drawbacks to this model. The model does not, for instance, explicitly consider the cost of various obsolescence mitigation actions. This could be adjusted for, however, by considering a relationship between the selected action, the fixed cost of obsolescence (C_O), and the post-obsolescence per-unit cost (c_N^2). Also, the model considers maximizing profit (Π_N) and thus adopts a seller's perspective. The methodology developed herein requires a model which adopts a buyer's perspective. This can be adjusted for by adapting the equations to reflect unit cost instead of profit. The most significant drawback to this model is its consideration of only the "first obsolescence cycle," meaning it only considers the trades associated with the *first* component to go obsolete [21].

2.6.6 MILP Refresh Model

Zheng *et al.* develop a model which utilizes mixed integer linear programming (MILP) to determine design refresh plans which miminize life-cycle cost. Their model considers both *when* to perform design refreshes and *which components* to include in the design refreshes. It does this by making a few simplifying assumptions: First, their model assumes that the production plan is known and fixed. Second, it assumes that design refreshes can only take place during these pre-determined production events. This second assumption has the benefit of reducing the set of potential design refreshes to a finite and discrete set. Finally, it assumes that "short-term approaches [are] applied on a component-specific basis until the next design refresh" [109]. In other words, when short-term obsolescence mitigation actions are taken, they span the period between the obsolescence date and the date of the next redesign — never longer [109].

Two methods of optimizing design refreshes are considered: deterministic and probabilistic. The deterministic model uses several binary integer variables to determine whether a design refresh happens during a particular production event. y_i is a binary integer variable which represents whether or not a design refresh has occurred ($y = 0$ represents no refresh, and $y = 1$ represents a refresh) [109]. The binary integer variable x_{ij} represents whether component i is replaced at time point t_j [109]. Finally, z_{ik} is a binary integer variable which indicates whether component i becomes obsolete and is replaced using some short-term mitigation strategy during time period T_k [109]. Thus, x_{ij}, y_j, and z_{ik} represent the decision variables of the model. Parameters of the model include the set-up cost for each design refresh at time t_j (C_j^S), design refresh cost for component i at time t_j (C_{ij}^R), and short-term mitigation cost for component i during time period T_k (C_j^S) [109]. The objective function using these values, is then:

$$\sum_{j=1}^{n}(\sum_{i=1}^{m} C_{ij}^{R} x_{ij} + C_{j}^{S} y_{i}) + \sum_{k=1}^{n+1}\sum_{i=1}^{m} C_{ik}^{M} z_{ik}$$

Note that this objective function — while visually complex — is quite straightforward. It is simply the sum of the costs of each individual obsolescence mitigation action.

Not all combinations of $\{x_{ij}, y_j, z_{ik}\}$ are valid, however. For instance, setup cost is only incurred when components are replaced in the system, and this setup cost is only counted once for each redesign date — regardless of the number of components that were replaced. Mathematically, this is represented by the following inequality constraint:

$$\sum_{i=1}^{m} x_{ij} \leq M y_j (M \gg m)$$

Since y_j is restricted to 0 or 1, $y_j = 1$ if and only if at least one $x_{ij} = 1$. Additionally, whenever a component is replaced using a redesign, its obsolescence date is updated to a future date. Zheng *et al.* represent this mathematically via:

$$d_i = (1 - x_{ij}) d_i + x_{ij} D_{ij}^{R}$$

where d_i is the date in which the redesign occurred, and D_{ij}^{R} is the updated obsolescence date. Similarly, the constraint

$$d_i = (1 - z_{ik}) d_i + z_{ik} D_{ik}^{M}$$

indicates that the obsolescence date must also be updated when short-term mitigation actions are taken. Combining these constraints with the objective function leaves the full form of the mixed integer linear programming formulation as follows:

$$\begin{aligned} \text{Minimize} \sum_{j=1}^{n}(\sum_{i=1}^{m} C_{ij}^{R} x_{ij} + C_{j}^{S} y_i) + \sum_{k=1}^{n+1}\sum_{i=1}^{m} C_{ik}^{M} z_{ik} \\ \text{S.T.} \sum_{i=1}^{m} x_{ij} \leq M y_j (M \gg m) \\ d_i = (1 - x_{ij}) d_i + x_{ij} D_{ij}^{R} \\ d_i = (1 - z_{ik}) d_i + z_{ik} D_{ik}^{M} \end{aligned} \tag{13}$$

Note that the two date constraints must first be converted into linear inequality constraints before this formulation is usable[7].

The probabilistic version of their model works by considering a range of potential obsolescence dates for each component, and then applying Equation 13 for each combination of component obsolescence dates. Thus, the design refresh optimization problem becomes a combinatorial problem wherein each combination of components and design refresh dates must be computed. From there, the combination which yields the lowest expected life cycle cost is selected [109].

This mixed integer linear programming method of optimizing design refreshes offers some useful elements, but is ultimately inadequate for the methodology proposed herein. Its consideration of multiple parts simultaneously is a necessary aspect of the overall methodology being developed, but its inability to account for short-term mitigation actions of all durations means that it fails to capture the true complexity of the refresh optimization problem. This could potentially be remedied by adjusting the short-term mitigation cost for each component, C_{ik}^{M}, such that it is a function of the timing of obsolescence and the selected mitigation strategy. Finally, the model, as presented, does not consider the time-value of money, which is a necessary aspect of the proposed methodology.

Table 6: The Bradley and Guerrero Model was created to investigate the trades considered during the system design phase. It is therefore best suited for the proposed methodology. ✓= "good" ✗= "bad" ○= "neutral."

	Newsvendor	Porter's Design Refresh	Bradley & Guerrero	Cattani & Souza	Teunter & Fortuin	MILP Refresh Model
Multiple Actions Considered	✗	✓	○	✗	✗	○
Time Value of Money	○	✓	✓	✓	✓	✗
Treats Multiple Parts	✗	○	✗	✗	✓	✓
Adopts Buyer's Perspective	✓	✓	○	✓	✗	✓
Captures Reliability and Cost	✗	✗	✓	✗	✗	✗

2.6.7 Summary of Proposed Methods

This section presented a wide variety of methods for selecting and optimizing obsolescence mitigation strategies. Ultimately, no single method achieves all of the requirements for the proposed methodology, as shown in Table 6. Therefore it becomes necessary to select the model that provides the most useful "backbone," and to adapt that model to better suit the needs of this body of work. The Mixed Integer Linear Programming model presented in § 2.6.6 fits this need by providing a framework in the form of an objective function, as well as an initial set of linear inequality constraints. It falls short, however, since it does not explore an exhaustive set of feasible solutions — specifically when considering short-term bridging strategies. Furthermore, the cost efficients used in the objective function are found using subject-matter expert opinions. Since SMEs are subject to biases and uncertainty, a more robust and transparent means of computing cost is desirable. For this, the Bradley and Guerrero model presented in § 2.6.5 provides a strong starting-point. Unlike the other options, it offers the ability to visualize the underlying trends associated with the systems engineering trades. Its biggest drawback is its inability to analyze multiple obsolescence events at once, making it a perfect compliment to the Mixed Integer Linear Programming formulation.

Conjecture: 2.2
The Mixed Integer Linear Programming Design Refresh model provides a robust foundation for optimizing obsolescence mitigation plans. By relaxing its assumptions and incorporating the desirable elements of the Bradley and Guerrero model and Porter's Design Refresh Model, a new MILP formulation can be created that explores a larger feasible space.

[7]These constraints are converted into linear inequality constraints later in § 4.2.2.

2.7 Research Question Review and Methodology Summary

Chapter 1 motivated the primary research objective, repeated below:

> **Research Objective**
> To develop a methodology for long life cycle COTS-based systems which addresses the obsolescence concerns associated with COTS components throughout the system's life cycle.

This research objective was addressed via two driving research questions:

> **Research Question: 1**
> How should the requirements definition phase of the engineering process be modified with obsolescence in mind to enable the development of a new COTS-based design methodology?

and

> **Research Question: 2**
> How should obsolescence mitigation strategies be incorporated into a new COTS-based design methodology?

These two primary research questions were further decomposed, and a formalized systems engineering framework called the Cyber-physical Acquisition Strategy for COTS-based Agility-Driven Engineering (CASCADE) was created, depicted in Figure 29. To address the sustainment-related challenges of COTS-based systems, a new hybrid design refresh model was proposed which combines a Mixed Integer Linear Programming formulation with an obsolescence cost model. In the next chapter, this new CASCADE design refresh model is developed, and is later tested in Chapter 4.

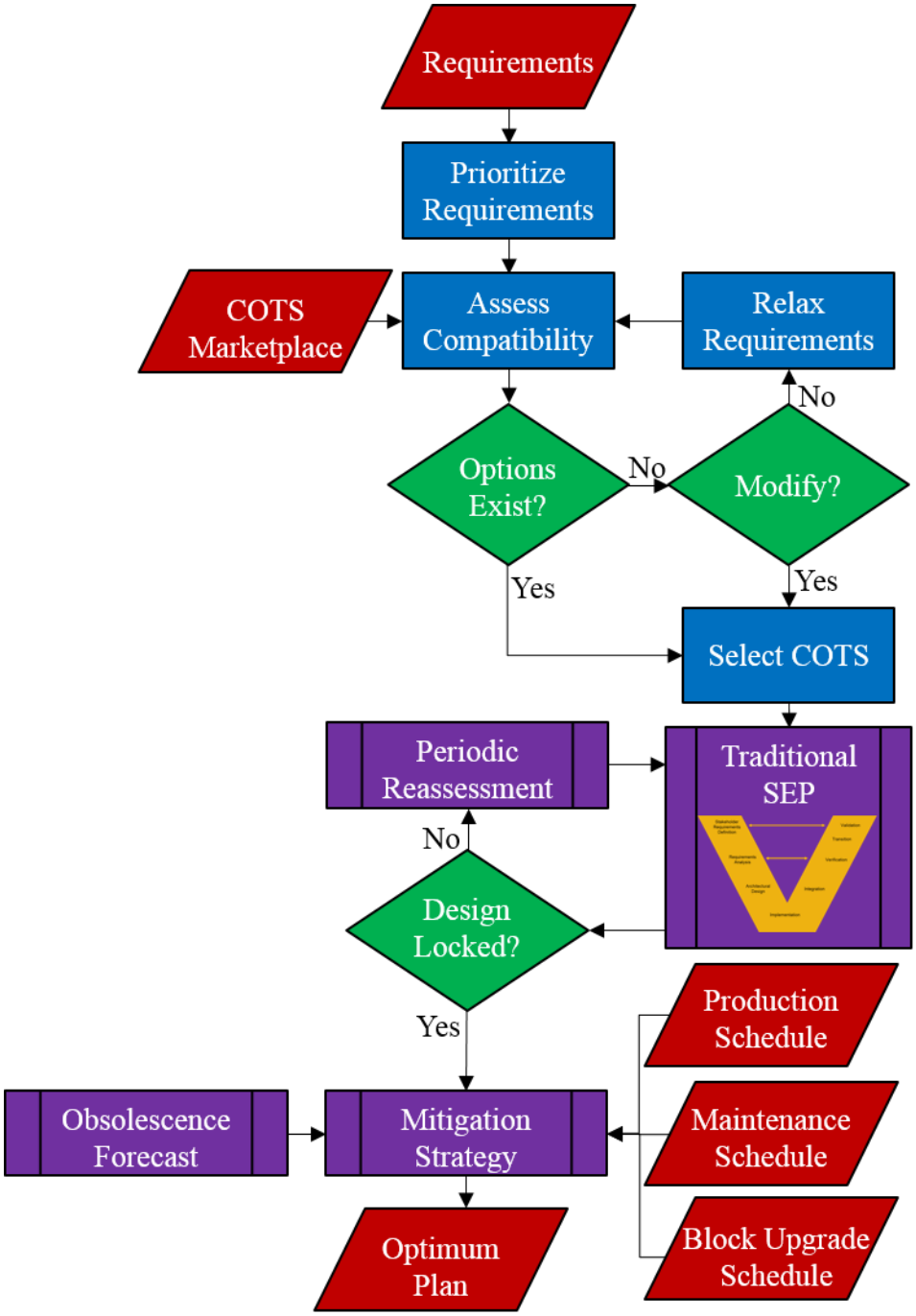

Figure 29: Flow diagram depicting the CASCADE methodology. The CASCADE methodology adds several new elements to the existing systems engineering and DoD acquisition processes. First, the systems engineering process starts with an "identification phase," which works by matching the requirements definition with the components available in the COTS marketplace. The COTS marketplace must be reassessed throughout the design process to ensure that the final system is up to date. Next, during the sustainment phase, obsolescence forecasts must be iterated with obsolescence mitigation strategies to ensure an optimum strategy is followed.

CHAPTER III

METHODOLOGY FORMULATION

Chapter 2 discussed the need for a new method for optimizing design refresh plans, and it outlined the desired elements of such a strategy. These desired elements include a transparent means of computing the economic impact of each potential obsolescence mitigation action (including considerations for the time-value of money), and a means of simultaneously balancing multiple obsolescence mitigation actions against one another. Although it was shown that no one method captures all of these desired elements, three existing methods were chosen as starting points for the development of a new method: The Mixed Integer Linear Programming (MILP) formulation used by Zheng *et al.* (presented in § 2.6.6) was selected as a means of generating optimal obsolescence mitigation plans. Elements of the Bradley and Guerrero model (presented in § 2.6.5) and Porter's Design Refresh Model (presented in § 2.6.3) will be incorporated into the MILP formulation as a means of computing the cost of each potential obsolescence mitigation action.

This chapter uses the MILP Design Refresh Model, Bradley and Guerrero's model, and Porter's Design Refresh model as starting points for the development of a new method for generating optimal design refresh plans. Section 3.1 discusses the limitations of the current MILP design refresh model, and presents an adapted formulation that addresses these limitations. This new CASCADE MILP formulation requires a quick and accurate means of computing the costs of individual obsolescence mitigation actions, so § 3.3.2 adapts the Bradley and Guerrero cost model to fill this need.

3.1 Requirements for Optimal Obsolescence Mitigation

One of the key elements selected for the CASCADE methodology is the use of Mixed Integer Linear Programming. Zheng *et al.*'s MILP model provides a sound starting point, but as § 2.6.6 notes, it falls short of some of the necessary elements of the CASCADE methodology. This section elaborates on these shortcomings, and presents a "path forward" that aims to relax some of the limiting assumptions made in the original formulation. To better understand these assumptions and their limitations, § 3.1.2 discusses the basic events throughout a system's life cycle that must be considered for an exhaustive formulation. Next, § 3.1.1 discusses the standard form of any Mixed Integer Linear Programming formulation. With a better understanding of a system's life cycle and standard MILPs, § 3.1.3 re-visits the original MILP formulation, and elaborates on its shortcomings.

3.1.1 Mixed Integer Linear Programming

Before discussing how to *change* the original MILP Design Refresh Model, it is first necessary to have a basic understanding of Mixed Integer Linear Programming, in general. This section first describes this broad category of optimization problems known as Mixed Integer Linear Programming problems. Then, a sub-category of this class of problem known as *Mixed 0/1 Integer Linear Programming* is described. Finally, two broad methods for solving such problems are presented.

Mixed Integer Linear Programming (MILP) is a constrained optimization in which the objective function and constraints are linear[43]. The term *mixed integer* is used because some (or all) of the decision variables are restricted to being integers. Many types of problems can be solved using a MILP formulation, including the so-called "knapsack problem," warehouse location problems, "traveling salesman" problem, and many others [27, 43]. Mixed Integer Linear Programming problems take the

form

$$\begin{aligned} &\max cx + hy \\ &\text{subject to } Ax + Gy \leq b \\ &x \geq 0 \text{ integral} \\ &y \geq 0 \end{aligned} \tag{14}$$

where the objective function coefficients are in row vectors $c = (c_1, c_2, \ldots, c_n)$ and $h = (h_1, h_2, \ldots, h_p)$, and the constraint data are in the $m \times n$ matrix $A = (a_{ij})$, the $m \times p$ matrix $G = (g_{ij})$, and the column vector $b = \begin{pmatrix} b_1 \\ \vdots \\ b_m \end{pmatrix}$. Column vectors $x = \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix}$ and $y = \begin{pmatrix} y_1 \\ \vdots \\ y_p \end{pmatrix}$ are the decision variables [27]. As the term "mixed" implies, this general form allows for both integral (x) and real-valued (y) decision variables.

A special form of MILP is a "pure integer linear program," which takes a similar form, but only allows for integral decision variables. Thus, a pure integer linear program takes the form

$$\begin{aligned} &\max cx \\ &\text{subject to } Ax \leq b \\ &x \geq 0 \text{ integral} \end{aligned} \tag{15}$$

Pure Integer Linear Programming problems represent a subset of standard MILP problems.

An even more-specific class of Mixed Integer Linear Programming problem is known as "Mixed 0,1 Linear Program." As the name implies, this class of problem limits the decision variables to values of either 0 or 1 (representing "on/off," or

"yes/no"), only [27]. These problems take the form

$$\begin{aligned} &\max cx \\ &\text{subject to } Ax \leq b \\ &x \in \{0,1\} \end{aligned} \tag{16}$$

Since the problem of optimizing design refresh strategies requires that a "yes or no" decision be made for each action in a set of potential obsolescence mitigation actions, this "Mixed 0,1 Linear Program" formulation provides the right level of flexibility while limiting the solution space to only meaningful values. Note that since "Mixed 0,1 Linear Programming" is a sub-set of standard MILP problems (and for the sake of readability), the term "MILP" will continue to be used throughout this document when referring to the original and new MILP design refresh formulations.

3.1.1.1 Solving MILP Problems

For any MILP problem, the set of feasible solutions, S, is known as the *mixed integer linear set* [27]. For the Mixed 0, 1 Linear Program subclass of problem, this set is

$$S := \{x \in \{0,1\}^n : Ax \leq b\}$$

It is tempting (and, indeed, correct) to assume that every $x \in S$ can be checked individually in order to obtain the optimum. This method (known as *e*xplicit enumeration) is impractical, however, for all but the simplest of applications [27, 43] . Consider, for instance, a system with five components, and 10 possible redesign dates throughout the system's life cycle. The full enumeration of obsolescence mitigation plans for such a system would include $2^{50} \approx 10^{15}$ redesign schedules, alone — not including bridging actions. If a single cost calculation takes a mere 1 microsecond, calculating the cost for each redesign schedule would take over 3,500 years. This method is impractical for all but the simplest of applications.

Fortunately, many established algorithms exist to solve these kinds of problems more efficiently. Two general approaches approaches are discussed herein: *Cutting Plane* methods, and *Branch and Bound* methods[1]. Both of these methods begin by first relaxing the integrality restrictions on the decision variables. That is, it is assumed that $x_i \in \mathbb{R}$, without requiring $x_i \in \{0, 1\}$ [27, 43]. As the name implies, the Branch and Bound method relies on two main steps: Branching (splitting the optimization problem into smaller sub-problems and then solving), and Bounding (keeping track of the known lower bound of the optimum so that effort is not focused on sub-optimal subsections of the search space). The Cutting Plane method works by "slicing" planes through the relaxed space to carve away infeasible or sub-optimal portions of the space, eventually creating a polyhedral. Both methods terminate once the optimum solution of the relaxed space also lies in the original set of feasible solutions [27].

3.1.2 Life Cycle Events

As the previous section discussed, the problem of optimizing obsolescence mitigation plans requires the selection of discrete obsolescence mitigation actions (e.g. redesign or LOT buy) at discrete points in time. To fully understand the shortcomings of the original MILP formulation, it is therefore important to understand not only *which* actions can be taken, but also *when* it is possible to apply them. The question of *when* requires an understanding of the events that take place in a typical system's life cycle. These include production events, block upgrades, and scheduled maintenance, as discussed in Sections 3.1.2.1 through 3.1.2.3, below.

[1]Note that many versions of these approaches exist, including hybrid approaches. The discussion herein is merely intended to provide a general overview of such algorithms.

3.1.2.1 Production

To avoid replacing parts twice — once for existing systems and once for newly-produced systems — it is often desirable to schedule replacements in-line with production events. Production can take the form of Low-Rate Initial Production (LRIP) or Full-Rate Production (FRP) [1].The primary difference between the two is the quantity of systems being produced. Early buyers use LRIP as an opportunity to test the initial set of systems to ensure commitment to new programs[3]. During FRP, proven systems are produced more efficiently and at a higher rate, and often at a lower cost[3]. For the purposes of the Mixed Integer Linear Programming formulation developed in this chapter, both LRIP and FRP can be viewed as opportunities to apply obsolescence mitigation actions. Each production date can be represented mathematically by d_{prod}, and the set of all production dates for the system is denoted D_{prod}.

3.1.2.2 Block Upgrades

The Department of Defense's preferred means of acquisition is known as *evolutionary acquisition*[104]. By this approach, an operationally useful and supportable capability is delivered as quickly as possible, followed by one or more "blocks" (or "increments"), which incrementally increase the capability of the system[97, 104]. A recent example of this acquisition strategy is the F-35 Joint Strike Fighter, whose acquisition was broken into three main software blocks[97, 98]. Block 1 provided initial training capabilities, and was largely completed in 2012[97]. Blocks 2 and 3 were broken into smaller blocks: 2A, 2B, 3i, and 3F[98]. Blocks 2B and 3i will provide initial warfighting capabilities, while block 3F provides the full suite of warfighting capabilities[98]. Figures 30 and 31 show the development and hierarchy of block upgrades, respectively, for the Joint Strike Fighter.

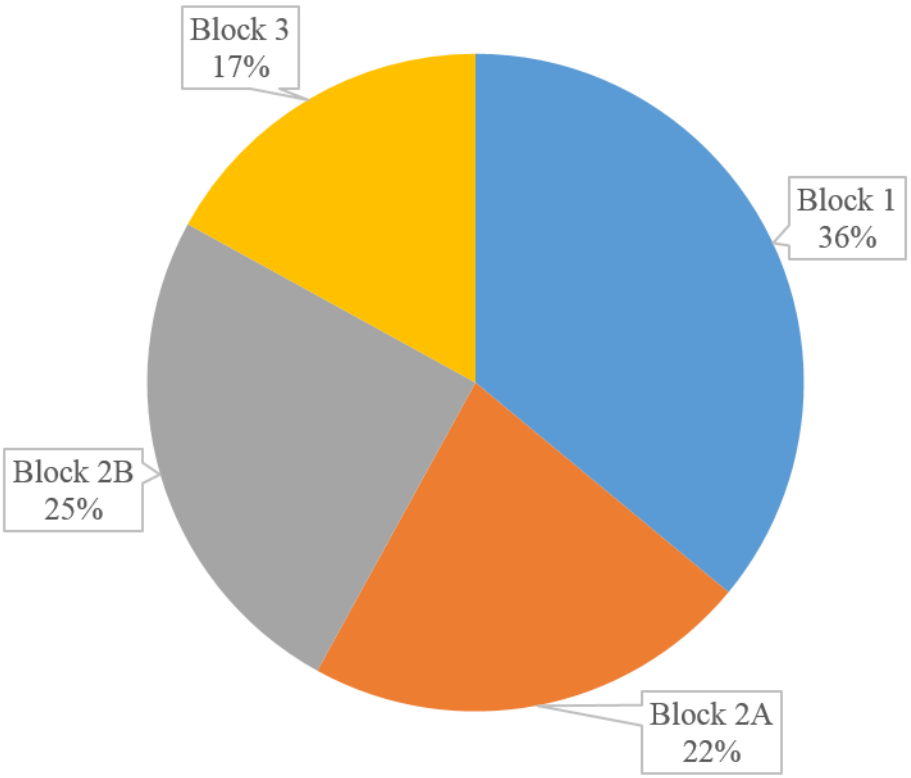

Figure 30: Percent of sensor fusion development in the F-35 Joint Strike Fighter's software blocks[97].

As with production events, Block Upgrades represent discrete opportunities throughout a system's life cycle during which obsolete parts can be replaced. In fact, replacing obsolete components during Block Upgrades may be desirable since doing so has the added benefit of reducing (or eliminating) the system-level non-recurring engineering (NRE) cost during those periods. DoD Instruction 7041.3 defines "common costs" to be any costs whose magnitudes and timings are identical between alternatives[11]. If a system undergoes a block upgrade at a given date d_{block}, it will incur some degree of system-level NRE cost, including the cost to re-certify the system. Therefore, system-level NRE costs needn't be included in the cost of obsolescence management at time d_{block} so as to avoid double-counting this common cost. As before, the set of all Block Upgrade dates (d_{block}) throughout the system's life cycle is denoted D_{block}.

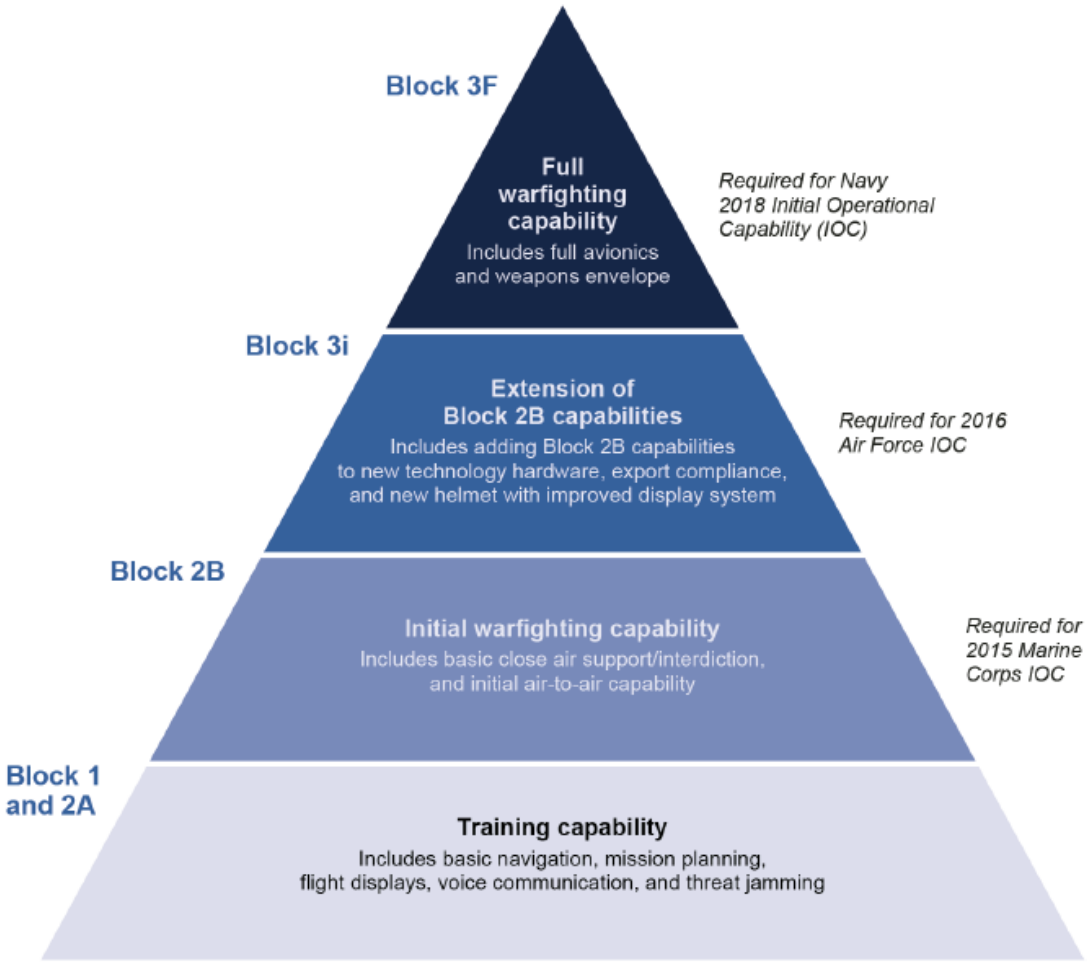

Figure 31: Hierarchy of block upgrades for the F-35 Joint Strike Fighter[98].

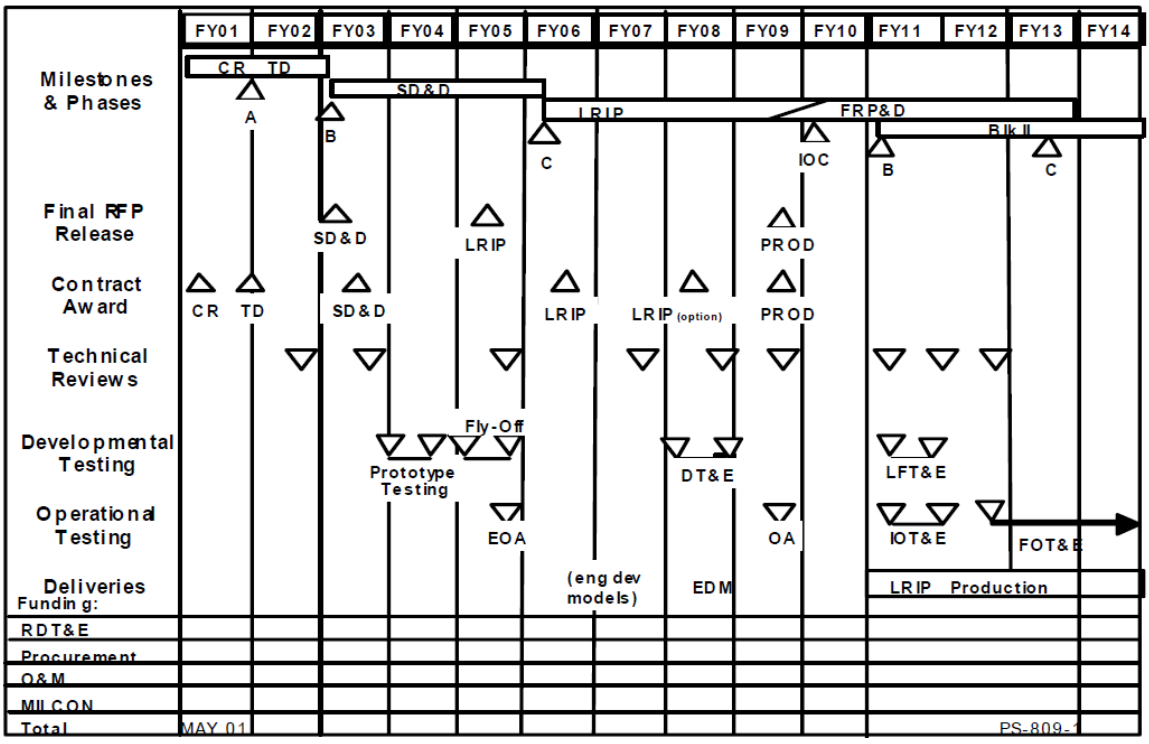

Figure 32: Example top-level system life cycle for defense/aerospace systems[104].

3.1.2.3 Scheduled Maintenance

The final time during which obsolete parts can be replaced is during scheduled maintenance periods. In the military, maintenance is accomplished in one of two primary ways: Field-level maintenance, and Depot-level maintenance[77]. Field-level maintenance represents day-to-day repairs, and is typically comprised of shop-type work such as repairing circuit boards or maintaining software[77]. Depot-level maintenance is a compliment to field-level maintenance, and can range from major system repairs to complete rebuilding of systems. The set of all maintenance dates (d_{maint}) throughout the system's life cycle is denoted D_{maint}.

3.1.2.4 Impact on Design Refresh Planning

Each of the discrete events discussed above impacts when each obsolescence mitigation action may take place, as well as the impact of those actions. For instance,

the replacement of a component via redesign can only take place at one of the discrete dates listed above. Namely, redesigns can only take place during a discrete $d_{red} \in D_{red}$, where $D_{red} = D_{prod} \cup D_{block} \cup D_{maint}$. Life-of-Type (LOT) Buys, on the other hand, are more flexible since the original components continue to be used, even after obsolescence. Therefore, LOT Buys can begin whenever a component first goes obsolete, regardless of when this occurs during the system's life cycle. LOT Buys *are* limited, however, in their duration. When the stock of LOT Buy components depletes to zero, a new obsolescence mitigation action must take place. Therefore, unless the LOT Buy terminates at the end of the system's life, its termination must coincide with the date of a scheduled redesign. Given the restriction on the timing of redesigns above, this means that LOT Buys can *begin* at the date of obsolescence (regardless of where this falls during the system's life), but must *end* at a production, maintenance, or block upgrade event.

3.1.3 Original MILP Formulation

With an understanding of what Mixed Integer Linear Programs are and how they work, as well as an understanding of the necessary system life cycle events that must be captured for an exhaustive design refresh formulation, it is now possible to discuss some of the limitations of the original MILP design refresh formulation. The original MILP model was presented in Equation 13, and is reproduced below [109]:

$$\text{Minimize} \sum_{j=1}^{n}(\sum_{i=1}^{m} C_{ij}^{R} x_{ij} + C_{j}^{S} y_i) + \sum_{k=1}^{n+1}\sum_{i=1}^{m} C_{ik}^{M} z_{ik}$$

$$\text{S.T.} \sum_{i=1}^{m} x_{ij} \leq M y_j (M \gg m)$$

$$d_i = (1 - x_{ij}) d_i + x_{ij} D_{ij}^{R}$$

$$d_i = (1 - z_{ik}) d_i + z_{ik} D_{ik}^{M}$$

As noted, this formulation presents several limitations. Firstly, the decision variables (x_{ij}, y_j, and z_{ik}) limit the feasible space to only a subset of the possible obsolescence mitigation options. The decision variables x_{ij} and y_j allow each component to be replaced at any one of the discrete redesign times. However, z_{ik}, which indicates whether component i was replaced using a short-term strategy during period k, limits the possible short-term solutions for each component. The authors note that "short-term approaches [are] applied on a component-specific basis until the next design refresh" [109]. In other words, when short-term bridging solutions are applied, only one such solution is considered for each component: beginning from the date of obsolescence until the next available redesign date. In reality, as the Porter Design Refresh model shows, the optimal date of a redesign may not be "as soon as possible," but may instead take place several redesigns in the future[39]. Thus, the original MILP formulation does not accurately reflect the reality of obsolescence cost trends by overly-constraining the feasible space and thus eliminating feasible solutions.

The second limitation of the original MILP formulation is the lack of transparency and traceability associated with calculating the cost coefficients C_{ij}, C_j, and C_{ik}. Zheng *et al.* suggest using $C_{ij}^R = Q_i \times C_i \times M_{ij}^R$ and $C_{ik}^M = Q_i \times C_i \times M_{ik}^M$ where Q_i is the demand rate for component i, C_i is the original cost of component i, and M_{ij}^R and M_{ik}^M are modifiers provided by the user or defaulted using cost multiplier data [109]. While nothing is inherently wrong with these calculations, the use of modifiers M_{ij}^R and M_{ik}^M requires the input of subject-matter experts. These modifiers thus mask any assumptions and biases made by the SMEs, thus limiting the transparency and traceability of the cost coefficients. While these cost coefficients do not interfere with the mathematical validity of the Mixed Integer Linear Programming formulation, a more-explicit means of calculating cost is desirable.

The final, and most cumbersome, limitation of this formulation is its lack of conformance to the standard form of a Mixed Integer Linear Programming problem, as given

in Equation 14. The constraints $d_i = (1 - x_{ij})d_i + x_{ij}D_{ij}^R$ and $d_i = (1 - z_{ik})d_i + z_{ik}D_{ik}^M$ mandate that if a redesign or short-term mitigation occurs at time i or k, respectively, then the date of obsolescence must be updated to reflect that action. These constraints must be re-stated in the form of linear inequalities in order to conform to the standard form of a Mixed Integer Linear Programming problem, as given in Equation 14[2]. Furthermore, Equation 13 does not include any constraints requiring that any obsolescence mitigation take place at all. That is to say, $x_{ij} = 0$, $y_j = 0$, and $z_{ik} = 0$ for all i, j, k is a perfectly valid solution to the MILP formulation. Thus, the optimal strategy (given the constraints, as defined) will always be the trivial answer of "do nothing."

3.2 CASCADE MILP Formulation

With a clear understanding of the limitations of the original MILP design refresh model, as well as an understanding of the necessary elements of the desired formulation, an improved CASCADE MILP can be formulated. This is done in two parts: First, § 3.2.0.1 addresses the exclusion of feasible bridging actions by adjusting the decision variables and objective function. Next, § 3.2.0.2 generates the necessary linear inequality constraints to reflect the changes to the objective function. While these changes provide a mathematically rigorous way to optimize design refresh strategies, the coefficients of this new CASCADE formulation still require cost estimates for each possible obsolescence mitigation action. The means for computing these cost coefficients is addressed later in § 3.3.

3.2.0.1 CASCADE Objective Function

The first major change to the original MILP formulation is the adaptation of the decision variable subscripts. While this is not a mathematical necessity, doing so

[2]For the sake of clarity, this exercise comes later in Chapter 4. For now, suffice it to say that the constraints must be re-stated in the form of linear inequalities for this formulation to be useful.

will make the later changes easier to understand. The original formulation uses the decision variables x_{ij}, y_j, and z_{ik} to denote when components are replaced, when the system is re-certified, and when bridging actions are employed, respectively. The subscripts are indexes that indicate the component number (i), the index of the redesign date (j), and the index denoting the period in which a bridging strategy was employed (k). For clarity, the new subscripts are the component index (i), the date of redesign (d_{red}), and the date at which a bridging strategy *begins* (d_{bridge}). Thus, the decision variables retain their meaning, but their subscripts appear as $x_{i,d_{red}}$, $y_{d_{red}}$, and $z_{i,d_{red},d_{bridge}}$.

Note that an additional subscript has been added to the decision variable z. Where the original formulation only indicated the period in which a bridging strategy *began*, this new formulation also indicates when a bridging strategy *ends*. This addresses the first limitation (the exclusion of feasible bridging strategies) of the original MILP formulation by allowing bridging solutions in a range of durations where the original MILP formulation only accounted for one. Thus, the term $z_{3,2025,2020}$ represents a decision variable that indicates whether component 3 underwent a short-term mitigation starting in 2020 and ending in 2025.

With this new nomenclature, the original objective function must be re-stated as follows:

$$\begin{aligned} C_{obs} = &\sum_{i=1}^{m} \left(\sum_{d_{red} \in D_{red}} C^{R}_{i,d_{red}} x_{i,d_{red}} \right) + \sum_{d_{red} \in D_{red}} C^{NRE}_{d_{red}} y_{d_{red}} \\ &+ \sum_{i=1}^{m} \left(\sum_{d_{red} \in D_{red}} \left(\sum_{d_{bridge} \in D_{bridge}} C^{B}_{i,d_{red},d_{bridge}} z_{i,d_{red},d_{bridge}} \right) \right) \end{aligned} \tag{17}$$

Note that there is very little structural difference between this objective function and the original objective function, with the exception of an additional summation. This extra summation, which does not appear in the original objective function, serves to expand the feasible space to include bridging actions of *all* durations.

The subscript, d_{red} represents the date that a part *may* be redesigned. Similarly, d_{bridge} represents the date that a part *may* go obsolete. Since redesigns are most economical when done "just in time," the dates of redesigns are discrete, and they align with pre-planned production, maintenance, or upgrade dates [92, 109]. Thus, the set of potential redesign dates for each component is

$$D_{red} = D_{prod} \cup D_{block} \cup D_{maint}$$

where D_{Prod}, D_{Block}, and D_{Maint} are the discrete sets of production dates, block-upgrade dates, and maintenance dates, respectively, as described in § 3.1.2. Furthermore, for each component, i, the set of potential obsolescence dates is

$$D_{bridge,i} = \{d_{bridge} \in \mathbb{R} | d_{IOC} \leq d_{bridge} \leq d_{EOL} \wedge (d_{bridge} - \Delta_i) \in D_{red}\}$$

where d_{IOC} represents the date of IOC, d_{EOL} represents the end of life date of the system, and $\Delta_i > 0$ represents the life cycle of component i. Put more plainly, this says that component i may require a bridging solution at any date which is Δ_i after a redesign, and within the life of the system. Thus, if component i has a life cycle of $\Delta_i = 10$ years and is replaced in year $d_{red} = 2025$, it will go obsolete in year $d_{bridge} = d_{red} + \Delta_i = 2035$ and require a bridging solution unless it is replaced again prior to its obsolescence.

3.2.0.2 CASCADE Constraints

The sets of redesign dates and bridge dates must next be formulated into linear equality and inequality constraints in order to be usable in the standard MILP form. The mathematical representations of the constraints are as follows:

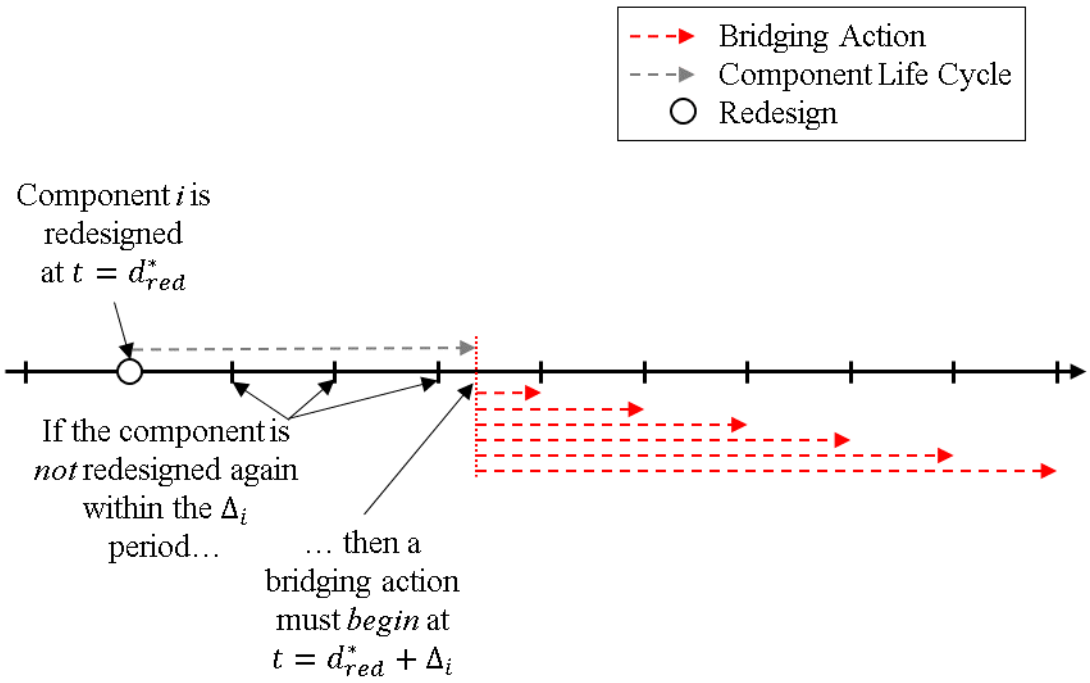

Figure 33: Visualization of MILP Constraint 1

$$\text{For each } i,\ d^*_{red} \in D_{red}\text{:}$$
$$x_{i,d^*_{red}} \geq \sum_{d_{red} > d^*_{red} + \Delta_i} z_{i,d_{red},d^*_{red}+\Delta_i} \tag{18a}$$

This constraint says that if a redesign takes place for component i at a particular time $t = d^*_{red}$, then *at most* one bridging action can be taken for component i which *starts* at $t = d^*_{red} + \Delta_i$. This constraint can be visualized in Figure 33.

$$\text{For each } i,\ d^*_{red} \in D_{red}\text{:}$$
$$x_{i,d^*_{red}} \geq \sum_{d_{bridge} < d^*_{red}} z_{i,d^*_{red},d_{bridge}} \tag{18b}$$

Constraint 18b says that if a bridging action for component i ends at a particular time $t = d^*_{red}$ then a redesign must take place at time $t = d^*_{red}$. Another way to say this is that whenever a redesign for component i takes place at $t = d^*_{red}$, then at most one bridging action may take place for component i which terminates at $t = d^*_{red}$.

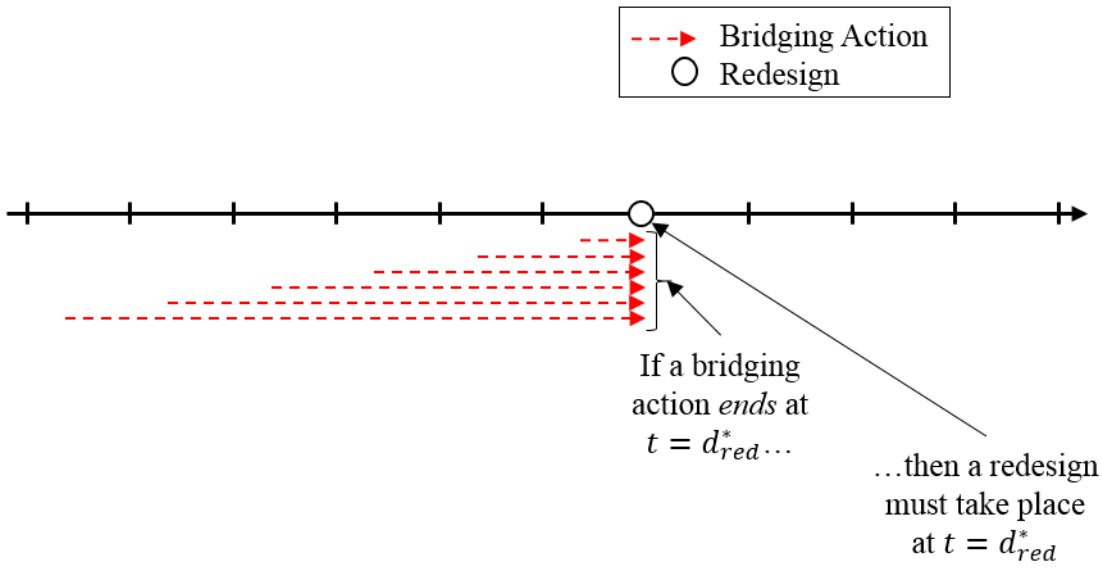

Figure 34: Visualization of MILP Constraint 2

While this constraint may seem self-evident, it is necessary to include it so as to be exhaustive. This constraint can be visualized in Figure 34.

For each i, $d^*_{red} \in D_{red}$:

$$\sum_{d^*_{red}<d_{red}<d^*_{red}+\Delta_i} x_{i,d_{red}} + \sum_{d_{bridge}\leq d^*_{red}+\Delta_i} \left(\sum_{d_{red}>d^*_{red}+\Delta_i} z_{i,d_{red},d_{bridge}} \right) \geq 1 \tag{18c}$$

Constraint 18c says that for each component, i, and within every time period Δ_i, one of the following must be true:

- The component must be redesigned
- A bridging solution must take place
- A previous bridging strategy must be underway which spans the Δ_i period

Put more plainly, constraint 18c says that for each component that faces obsolescence, *at least one* obsolescence mitigation action must be taken. Without this explicit constraint, the optimum strategy would simply be to let each part go obsolete without any obsolescence resolution. This constraint can be visualized in Figure 35.

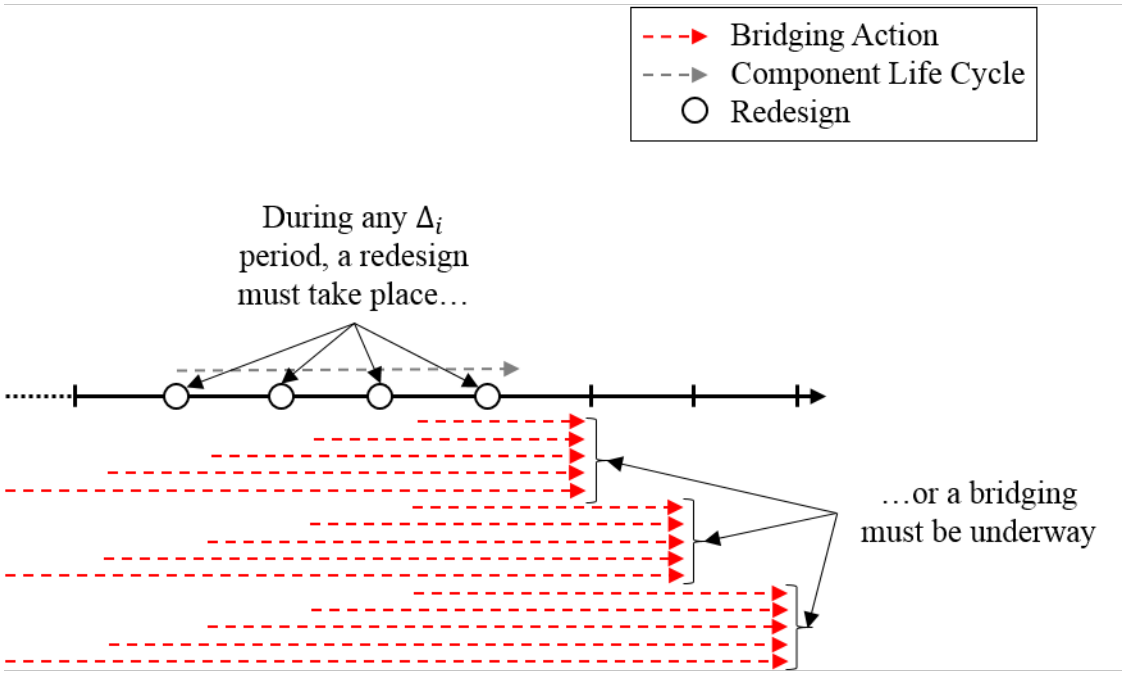

Figure 35: Visualization of MILP Constraint 3

$$\text{For each } d_{red} \in D_{red}: \\ \sum_{i=1}^{m} x_{i,d_{red}} \leq M y_{d_{red}} \text{ For } M \gg m \tag{18d}$$

Constraint 18d says that if *any* redesign takes place at $t = d_{red}$, then the program must incur some system-level non-recurring engineering (NRE) cost. Note that this constraint is identical in form to the third constraint from the original formulation (Equation 13).

$$x_{i,d_{red}} \in \{0, 1\} \tag{18e}$$

$$y_{d_{red}} \in \{0, 1\} \tag{18f}$$

$$z_{i,d_{red},d_{bridge}} \in \{0, 1\} \tag{18g}$$

Finally, Constraints 18e through 18g simply require that each decision variable either equal 0 or 1.

It is important to recognize that the summations above are not merely individual constants. Rather, they represent the rules necessary to build an exhaustive set of constraints. Combining constraints 18a through 18g with the objective function given in Equation 17 gives the final form of the Mixed Integer Linear Programming problem:

Minimize

$$C_{obs} = \sum_{i=1}^{m} \left(\sum_{d_{red} \in D_{red}} C^{R}_{i,d_{red}} x_{i,d_{red}} \right) + \sum_{d_{red} \in D_{red}} C^{NRE}_{d_{red}} y_{d_{red}} + \sum_{i=1}^{m} \left(\sum_{d_{red} \in D_{red}} \left(\sum_{d_{bridge} \in D_{bridge}} C^{B}_{i,d_{red},d_{bridge}} z_{i,d_{red},d_{bridge}} \right) \right)$$

S.T.

$$\begin{aligned}
&\text{For each } i,\ d^{*}_{red} \in D_{red}\text{: } x_{i,d^{*}_{red}} \geq \sum_{d_{red} > d^{*}_{red} + \Delta_i} z_{i,d_{red},d^{*}_{red}+\Delta_i} \\
&\text{For each } i,\ d^{*}_{red} \in D_{red}\text{: } x_{i,d^{*}_{red}} \geq \sum_{d_{bridge} < d^{*}_{red}} z_{i,d^{*}_{red},d_{bridge}} \\
&\text{For each } i,\ d^{*}_{red} \in D_{red}\text{: } \sum_{d^{*}_{red} < d_{red} < d^{*}_{red} + \Delta_i} x_{i,d_{red}} \\
&+ \sum_{d_{bridge} \leq d^{*}_{red} + \Delta_i} \left(\sum_{d_{red} > d^{*}_{red} + \Delta_i} z_{i,d_{red},d_{bridge}} \right) \geq 1 \\
&\text{For each } d_{red} \in D_{red}\text{: } \sum_{i=1}^{m} x_{i,d_{red}} \leq M y_{d_{red}} \text{ For } M \gg m \\
&\text{For each } i,\ d_{red} \in D_{red}\text{: } x_{i,d_{red}} \in \{0,1\} \\
&\text{For each } d_{red} \in D_{red}\text{: } y_{d_{red}} \in \{0,1\} \\
&\text{For each } i,\ d_{red} \in D_{red},\ d_{bridge} \in D_{bridge} : z_{i,d_{red},d_{bridge}} \in \{0,1\}
\end{aligned} \tag{19}$$

Equation 19 addresses two of the limitation of the original MILP formulation presented in §3.1.3. Namely, it presents the problem in the standard form of a mixed

integer linear programming problem, and it addresses all feasible bridging options[3]. It does not, however, address how the cost coefficients are calculated. While there is nothing inherently wrong with utilizing subject-matter experts, as used in the original formulation, doing so does not provide transparency and traceability. A suggested method for the calculation of these cost coefficients is presented in §3.3.2.

3.3 Cost Analysis

When considering the costs and benefits of various alternative systems, economic factors often play a large role. This is particularly true in the acquisition of military systems, since they are often costly and the DoD must work around a well-defined budget [29]. Furthermore, the long life cycles of many military systems means that costs are accrued over many disparate years, so exogenous factors such as inflation must be considered[78]. This section describes the important elements for any engineering cost analysis, and defines the tools necessary for the cost function used herein.

3.3.1 Time Value of Money

People often take for granted that money has a constant value over time when, in fact, this is not the case. Over short periods, the value of money can be assumed to be constant. The promise of $100 today is, for all intents and purposes, equivalent to the promise of $100 tomorrow. This is not true, however, over longer periods of time. Over periods of years, one must consider the *opportunity cost* of money. Opportunity cost is "the alternative value foregone when an asset is used for other

[3]Note that additional constraints could be defined which may further help to scope out infeasible/inviable mitigation plans. For instance, the objective function and constraints, as defined, guarantee a minimum-total-cost obsolescence mitigation plan. This optimal plan, however, ignores potential annual spending limits. Thus, it may be necessary to add an additional constraint that ensures that annual expenditures do not exceed a maximum threshold. Furthermore, there may be scheduling limits on the frequency with which redesigns can take place. It may therefore be necessary to add a constraint which limits the proximity of redesign actions to one-another.

purposes" [11]. Therefore, ignoring inflation, \$100 today has the opportunity to earn interest if invested, whereas \$100 one year (or any appreciable period) from now does not[29, 80]. This variable value of money over time is aptly refereed to as *time value of money* [11, 29, 78, 80].

To account for the time value of money, it is necessary to *discount* future expenditures to a baseline year. Discounting is effectively the inverse of interest being accrued over time[29]. To understand discounting better, first consider the "forward" process. An investment of C dollars made today at an interest rate of $r\%$ per year would earn $\$C \cdot r$ in interest after one year, and would therefore be worth a total of $\$C \cdot (1 + r)$. After year two, the investment would earn an additional $r\%$ on the $\$C \cdot (1 + r)$, and would therefore be worth $\$C \cdot (1 + r)^2$. Extending this example further, an investment of $\$C$ today at an interest rate of $r\%$ will be worth

$$C \cdot (1 + r)^n$$

in n years[29].

By discounting future values to a base year, one is effectively asking "how much money would need to be invested in the base year to be worth the value of the expenditure in its future year." To answer this question, the math simply needs to be reversed. So, a future payment of $\$P_1$ one year from today is worth $\$P_1/(1 + r)$, assuming a discount rate of $r\%$[4]. Extending this to an arbitrary number of years, a future payment of $\$P_n$ is equivalent to

$$P_n/(1 + r)^n$$

dollars today, assuming P_n occurs n years in the future[29]. Therefore, continuing the example from earlier, \$100 one year from now is worth $\$100/(1 + .05) = 95.24$ today, assuming it could have earned %5 interest.

[4]Note that the discount rate is related but not identical to the interest rate in the earlier example. The distinction between the two will be made clear shortly.

Table 7: 2014 Nominal and Real discount rates as published in Appendix C of OMB Circula A-94[9].

	3-Year	5-Year	7-Year	10-Year	20-Year	30-Year
Nominal	1.7	2.2	2.5	2.8	3.1	3.4
Real	0.1	0.4	0.7	0.9	1.2	1.4

As noted earlier, the discount rate and the interest rate are related but not identical. The discount rate reflects the cost of borrowing money over a specific period of time, and its value depends on the national and international economies, as well as market sector[80]. Typical ranges for the US government, for instance, are between 0.9–2.7%, whereas public companies may use rates between 10-12% or higher [80]. Discount rates are expressed in one of two forms: *real* and *nominal*[11, 29, 80]. The real rate assumes a constant dollar. That is, inflation is not considered in the real discount rate. The nominal rate, on the other hand, implicitly includes inflation[11, 29]. Note that neither of these is "more correct" than the other — the use of one over the other depends on whether or not inflation is accounted for in the future costs. It is important, however, to maintain consistency in a given calculation and to not mix the use of these two forms[11].

The discount rates for the DoD is based on the U.S. Treasury Department's cost of borrowing funds, and new rates are published annually in the Office of Management and Business (OBM) Circular A-94, appendix C[9, 11]. Table 7 gives the most recent discount rates as of the time of this publication. Note that the rate changes over time. Rates are higher for long-duration projects because long-term treasury bonds carry higher interest rates[29].

3.3.2 Cost Coefficients

This section describes how to compute the cost coefficients used in the objective function of Equation 19. These coefficients capture the time-value of money by incorporating elements of Bradley and Guerrero's profit model (originally presented in

§ 2.6.5) and Porter's Design Refresh model (originally presented in § 2.6.3). This new cost function will serve as a transparent means of computing the cost coefficients of Equation 19 by reducing the dependence on subject-matter experts.

To guide the development of the cost coefficients, it is helpful to consider each type of obsolescence mitigation action separately and then combine these separate cost functions into one. The cost of redesign is

$$c_{obs_i}^{red} = c_i^{red} e^{-r(t_{start} - t_{base})} \tag{20a}$$

where c_i^{red} is the cost of redesign component i in year t_{start} dollars, t_{base} is the base year, and r is the discount rate. Note that this form is borrowed directly from Porter's Design Refresh model[39, 80]. Thus, the cost of obsolescence due to redesign is simply the discounted cost of redesigning the component.

Next, consider the cost of a life-of-type buy which begins at time t_{start} and ends at time t_{end}. The original form of Porter's Design Refresh Model, described in § 2.6.3, shows that this is simply the one-time cost to purchase the necessary components discounted to some base year. Thus

$$c_{obs_i}^{LOT} = c_i^{LOT}(t_{end} - t_{start}) q_i e^{-r(t_{start} - t_{base})}$$

where c_i^{LOT} is the cost of component i at the time when the life-of-type buy is made (t_{start}), t_{base} is the base year, q_i is the demand rate for the component, and r is the discount rate. As §2.6.3 indicated, however, this formulation ignores the cost to store the components throughout the life-of-type buy. To correct for this, it is necessary to include the holding cost for each component throughout the life-of-type buy. Considering that the number of components being maintained reduces over time as those components are used, the holding cost takes the form of an integral. Thus, the cost of a life-of-type buy is given by

$$c_{obs_i}^{LOT} = c_i^{LOT}(t_{end} - t_{start})q_i e^{-r(t_{start}-t_{base})} + \int_{t_{start}}^{t_{end}} c_i^H (t_{end} - s)q_i e^{-r(s-t_{base})} ds \quad (20b)$$

where c_i^H is the annual cost per unit of holding component i[5].

Some obsolescence mitigation strategies may result in a change in the per-unit cost of a component/system. For instance, when a component is replaced with an alternative component (e.g. via emulation or the use of a third-party vendor) , the price of that replacement component often exceeds the price of the original component. The cost of such replacement strategies is given by

$$c_{obs_i}^{rep} = \int_{t_{start}}^{t_{end}} c_i^{rep} q_i e^{-r(s-t_{base})} ds \quad (20c)$$

where c_i^{rep} is the change in cost of component i due to replacing the component[6], t_{start} is the first year that the replacement part is used, t_{end} is the final year the replacement part is used, t_{base} is the base year, and r is the discount rate.

Finally, many obsolescence mitigation actions may require that some system-level non-recurring engineering (NRE) cost be incurred. This includes the cost of re-certification, training, documentation, set-up, etc. The non-recurring engineering cost is given by

$$c_{obs}^{NRE} = c^{NRE} e^{-r(t_{start}-t_{base})} \quad (20d)$$

where c^{NRE} is the system-level non-recurring engineering cost at time t_{start}, t_{base} is

[5]As in the Porter Design Refresh Model, the holding cost addresses all costs associated with holding and managing LOT buy inventory. These costs include the cost of warehousing, insurance, security, testing, etc. As Porter notes, these costs are typically covered via an "inventory holding cost factor," which can range from 10% to 30% of the cost of the inventory[39]. Thus, a straightforward way of factoring in the annual holding cost for this methodology is simply to use a percent of the unit cost for each component.

[6]It is important to note that c_i^{rep}, unlike the other cost terms used, represents the *change* in cost due to replacement — not the nominal cost of the replacement part. Thus, if a \$5,000 component is replaced using a component costing \$6,000, then $c_i^{rep} = \$1,000$.

the base year, and r is the discount rate[7].

Equations 20a through 20d can be combined to give the total cost of any set of obsolescence mitigation actions between time t_{start} and t_{end} discounted to a base year t_{base}, as shown in Equation 21.

$$c_{obs_i} = \int_{t_{start}}^{t_{end}} \left(c_i^{rep} + c_i^H \left(t_{end} - s\right)\right) q_i e^{-r(s-t_{base})} ds + c_i^{red} e^{-r(t_{start}-t_{base})} + c_i^{NRE} e^{-r(t_{start}-t_{base})} + c_i^{LOT} (t_{end} - t_{start}) q_i e^{-r(t_{start}-t_{base})} \tag{21}$$

Note that Equation 21 represents an exhaustive cost formulation for a single component i across any span of time (t_{start} – t_{end}) while undergoing any obsolescence mitigation action or set of actions. For the purposes of this document it serves as a transparent and exhaustive means to calculate each of the cost coefficients in Equation 19. That is not to suggest, however, that its utility is limited to this purpose. When used outside the context of the specific MILP problem presented herein, Equation 21 enables the comparison of competing designs through the computation of the level set, $\mathscr{L}$, as was done in the original Bradley and Guerrero model presented in §2.6.5.

3.4 CASCADE MILP Assumptions

Throughout the development of the CASCADE MILP formulation, several explicit and implicit assumptions were made. To better understand how and when this formulation is applicable, it is important to understand each of these assumptions:

Assumption 1: Discrete Redesign Dates — As §3.1.2 discussed, it is assumed that obsolete components can only be replaced via redesign during discrete

[7]The base non-recurring engineering cost (c^{NRE}) can be obtained in a variety of ways, including via subject-matter experts. A straightforward means of obtaining c^{NRE} is simply the summation of each known non-recurring cost (e.g. $c^{NRE} = c_{certification} + c_{training} + c_{documentation} + \ldots$).

dates. These dates coincide with pre-planned production, maintenance, and block upgrade dates.

Assumption 2: Fixed Component Life Cycles — It is assumed that the life cycle of replacement parts is fixed, and is identical to the life cycle of the original component (namely, Δ_i). This assumption is consistent with the original MILP formulation[109].

Assumption 3: Discrete Obsolescence Dates — It is assumed that the set of dates during which components *may* go obsolete is discrete. In other words, if a component goes obsolete, its date of obsolescence is known and fixed (although it may be updated as specific obsolescence mitigation actions are taken). Specifically, the obsolescence date for each component, i, occurs Δ_i years after the last date of redesign[8].

Assumption 4: Component Redesigns Require NRE cost — Just as in the original MILP formulation, it is assumed that a system-level non-recurring engineering (NRE) cost must be incurred each time at least one redesign occurs[109]. This cost is fixed, regardless of the number of redesigned components (unless no components are redesigned).

Beyond these assumptions, the CASCADE MILP formulation is quite accommodating. For instance, the cost coefficients formulated in §3.3.2 includes the discount rate, r. This enables the consideration of the time-value of money by discounting future costs (i.e. when $r > 0$). If desired, however, all costs can be computed in then-year dollars simply by setting $r = 0$. Furthermore, when computing the values of the cost coefficients, one needn't assume that each parameter is fixed in time (although

[8]While less obvious, Assumption 3 is a consequence of Assumptions 1 and 2. Since redesigns can only be scheduled during discrete dates (Assumption 1), and since each replacement component has a fixed Δ_i life cycle (Assumption 2), the set of potential obsolescence dates is Δ_i years after each of the discrete redesign dates.

doing so simplifies the calculations). For instance, if enough information is known to predict how the cost and reliability of each component will evolve over time (that is, if $c_i^{LOT}(t)$ and/or $q_i(t)$ are known), then the temporal cost/reliability effects can be included when calculating each cost coefficient. For simplicity, though, it is sufficient to assume these values to be fixed.

Finally, although the CASCADE MILP was developed to address the sustainment of COTS-based hardware systems, the mathematical formulation does not limit its scope to *just* hardware. That is, nothing inherent to the objective function or constraints limits the CASCADE MILP from being applied to software systems. Applying the CASCADE MILP to a software system will only affect the values of the parameters used to calculate the cost coefficients — *not* the underlying process. Furthermore, the CASCADE MILP makes no assumptions about which specific level(s) within the system hierarchy (e.g. sub-system, component, part, etc.) the method is being applied to. Again, applying the CASCADE MILP to varying levels withing the system hierarchy will merely change the values of the cost coefficients. The underlying theory remains intact.

3.5 Procedure

Section 3.2 defined the objective function and constraints of the CASCADE Mixed Integer Linear Programming (MILP) formulation (Equation 19) and § 3.3 developed the equation that defines the obsolescence cost for a single component given a single obsolescence mitigation action (Equation 21). In this section, these elements are combined together into a single continuous procedure for generating optimum design refresh strategies:

Step 1: Defining System Parameters — The first step is to define the component and system-level parameters that will drive the selection of an optimum design refresh strategy. As alluded to earlier, these parameters include: Δ_i, the

life cycle of each component of type i; c_i^{red}, the cost to replacing component i in then-year dollars; c_i^{LOT}, the per-unit cost for component i during a life-of-type buy; c_i^{rep}, the change in cost associated with replacing a component via emulation/substitution; and c^{NRE}, the system-level non-recurring engineering.

Step 2: Establishing a Timeline — The second step is to define the system's timeline. That is, what pre-defined dates exist (e.g. maintenance dates, block upgrade dates, production dates) throughout the system's life cycle, during which system or component redesigns may take place? These dates, taken together, define the set D_{red} as described earlier. Also during this step, the start dates of all potential bridging actions are defined for each component, i, using the component life cycles (Δ_i) defined in Step 1. To do so, define the set of dates

$$D_{bridge,i} = \{d_{bridge} \in \mathbb{R} | d_{IOC} \leq d_{bridge} \leq d_{EOL} \wedge (d_{bridge} - \Delta_i) \in D_{red}\}$$

where d_{IOC} and d_{EOL} are the date of initial operational capability (IOC) and end of life (EOL), respectively.

Step 3: Calculating Cost Coefficients — The next step is to calculate the cost coefficients ($C^R_{i,d_{red}}$, $C^{NRE}_{d_{red}}$, and $C^B_{i,d_{red},d_{bridge}}$) in Equation 19 using Equation 21. The coefficient $C^R_{i,d_{red}}$ is computed for all i and d_{red} by setting $\left\{c_i^{rep}, c_i^H, c_i^{NRE}, c_i^{LOT}\right\} = 0$ and $t_{start} = d_{red}$. Thus

$$C^R_{i,d_{red}} = c_i^{red} e^{-r(d_{red} - t_{base})}$$

The non-recurring engineering cost, $C^{NRE}_{d_{red}}$, is a system-level cost, and therefore only needs to be computed once for each redesign date, d_{red}. Thus, for all d_{red}, $C^{NRE}_{d_{red}}$ is computed by setting $\left\{c_i^{rep}, c_i^H, c_i^{red}, c_i^{LOT}\right\} = 0$ and $t_{start} = d_{red}$ in Equation 21:

$$C^{NRE}_{d_{red}} = c^{NRE} e^{-r(d_{red} - t_{base})}$$

Finally, the cost of bridging actions ($C^B_{i,d_{red},d_{bridge}}$) is calculated using Equation 21 by setting $\left\{c^{NRE}, c_i^{red}\right\} = 0$, $t_{start} = d_{red}$, and $t_{end} = d_{bridge}$. The values of c_i^{rep}, c_i^H, and c_i^{LOT} will depend on the particular bridging actions being considered. Also, since the term "bridging action" represents several possible obsolescence mitigation options, the minimum-cost bridging action for each range of times should always be selected as follows[9]:

$$C^B_{i,d_{red},d_{bridge}} = \min \left(\int_{d_{bridge}}^{d_{red}} c_i^{rep} q_i e^{-r(s-t_{base})} ds, \right.$$
$$c_i^{LOT}(d_{red} - d_{bridge}) q_i e^{-r(d_{bridge}-t_{base})}$$
$$\left. + \int_{d_{bridge}}^{d_{red}} c_i^H (d_{red} - s) q_i e^{-r(s-t_{base})} ds \right)$$

Step 4: Generating MILP Constraints — With the cost coefficients computed, the next step is to define the constraints of the mixed integer linear programming problem as defined in Equation 19. Since most real-world applications of Equation 19 will have hundreds of constraints, the simplest way to generate these constraints is with the help of a computing environment such as ®.

Step 5: Running MILP — The final step is simply to optimize the Mixed Integer Linear Programming problem using an appropriate optimization method, such as branch-and-bound [43]. As before, this is carried out using a computing environment such as MATLAB®.

[9]Note that the option to replace a component using a third-party vendor or emulation may not always exist for each component. In such cases, the cost of replacement should be ignored, entirely, in favor of the LOT buy option.

Table 8: Table of component-level and system-level parameters for the example problem.

	Component 1	Component 2	Component 3
Δ_i	6 yrs.	9 yrs.	12 yrs.
c_i^{LOT}	\$8	\$10	\$12
c_i^H	\$10	\$10	\$10
c_i^{red}	\$80,000	\$100,000	\$120,000
q_i	1,000	500	250

	System
c^{NRE}	\$10,000
d_{IOC}	2015
Production Dates	$\{2015, 2020, 2025, 2030, 2035\}$
d_{EOL}	2040
d_{base}	2015

3.6 Example Problem

Section 3.5 described the step-by-step procedure for optimizing design refresh strategies. In this section, those steps are applied to a representative problem to demonstrate the procedure. The three components in this hypothetical system represent varying levels of ruggedization, with the least-rugged component (component 1) having the lowest cost and the highest demand, and the most-rugged component (component 3) having the highest cost and lowest demand. Normally, *Step 1: defining system parameters* would be done via market research and/or the help of subject-matter experts, but in this example representative values are chosen to reflect varying levels of ruggedization. Similarly for *Step 2*, it is assumed that $d_{IOC} = 2015$ and $d_{EOL} = 2040$ (i.e. a 25-year life cycle), and production events take place every five years. These component and system-level cost and schedule parameters are shown in Table 8.

In this example there are a total of 50 decision variables: 18 $x_{i,d_{red}}$'s (six for each of the three components), 26 $z_{i,d_{red},d_{bridge}}$'s (one for each combination or component, redesign date, and bridge date), and 6 $y_{d_{red}}$'s (one for each redesign date). *Step 3*

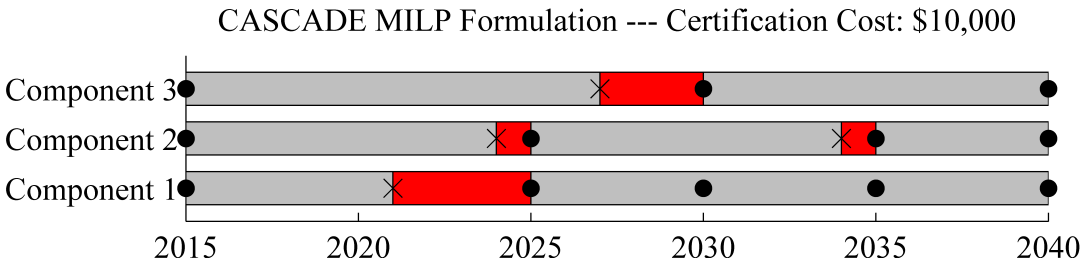

Figure 36: Timeline representing a system with three components. Black X's represent dates when components have gone obsolete and a bridging strategy was employed. Black dots represent dates when the component was replaced with a redesigned component. Red bands represent times when a bridging strategy was in use.

is to calculate the cost coefficients. This is done using the MATLAB® script in Appendix A, resulting in 50 cost values — one for each decision variable. In an unconstrained environment, 50 decision variables would result in over 10^{15} possible combinations of obsolescence mitigation actions. Fortunately, this number is significantly reduced in *Step 4* when Constraints 18a through 18g are generated using the MATLAB® code in Appendix B. This results in a total of 54 constraints, each of which is a linear equation with 50 terms (one for each decision variable).

In *Step 5*, the cost coefficients, decision variables, and constraints are all inputted into a MILP optimizer[10] to yield the optimum obsolescence mitigation plan. In this example, the optimum plan comes at a cost of $165,919. In this plan, component 1 is replaced in 2025, 2030, and 2035; component 2 is replaced in 2035 and 2040; and component 3 is only replaced in 2040. Also, 4 total short-term bridging actions are used: component 1 is allowed to go obsolete in 2021 and a life-of-type buy sustains it through 2025; component 2 goes obsolete in 2024 and is bridged through 2025; component 2 goes obsolete again in 2034 and is bridged until 2035; and component 3 goes obsolete in 2027 and is bridged until 2030. This timeline is illustrated in Figure 36.

Suppose, now, that an identical system is being designed, but two pre-planned

[10]In this case, MATLAB®'s "intlinprog" function is used to perform the optimization.

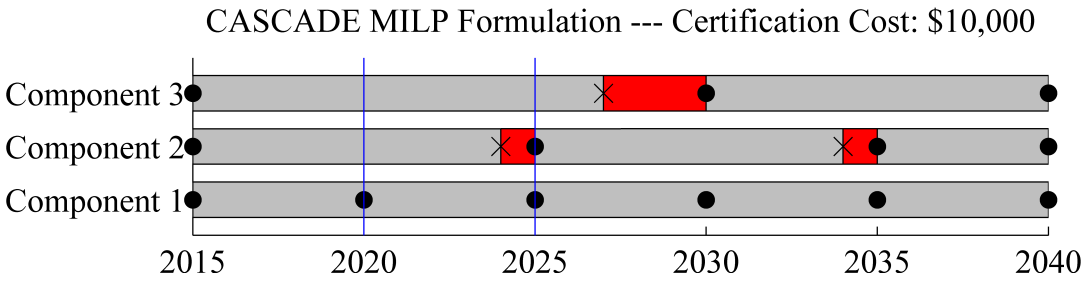

Figure 37: Timeline representing a system with three components, including two pre-planned block upgrades in 2020 and 2025. Black X's represent dates when components have gone obsolete and a bridging strategy was employed. Black dots represent dates when the component was replaced with a redesigned component. Red bands represent times when a bridging strategy was in use. The blue vertical lines represents the dates of the block upgrades. Note that unlike in Figure 36, component 1 is replaced in 2020 to take advantage of the free re-certification in this year.

block upgrades are scheduled for the years 2020 and 2025. In this scenario, $c_{2020}^{NRE} = 0$ and $c_{2025}^{NRE} = 0$ since no additional certification cost needs to be spent during the years of the block upgrades. That is, the system is already being re-certified in 2020 and 2025 because of the block upgrade, so no *additional* re-certification cost is incurred as a result of redesigning an obsolete part. In this example, the optimum design refresh strategy comes at a cost of $157,845 — a savings of $8,074 over the previous strategy. This is because component 1 is redesigned in 2020, thus taking advantage of the free re-certification in that year, and furthermore eliminating the need for the original bridging strategy between 2021 and 2025. This scenario is captured in the timeline in Figure 37. Note that while the savings in this alternative strategy may seem trivial, the alignment of redesigns with pre-planned upgrade/maintenance events can result in appreciable savings in systems with many components and long life cycles.

CHAPTER IV

VERIFICATION EXPERIMENTS

Chapter 3 discussed the limitations of the original MILP Design Refresh model, as well as the desired elements of a more comprehensive model. From these observations, a new CASCADE MILP formulation was developed in § 3.2, which includes an adjusted objective function and the necessary linear inequality constraints. Next, in § 3.3, the same process was followed for the development of a new cost model which enables the cost of individual obsolescence mitigation actions to be quickly and transparently computed. These two elements, together, make up the CASCADE design refresh model. This new design refresh model is intended to act as a decision support tool when designing new COTS-based systems, and it allows for the generation of an optimum design refresh strategy. Before the results of this new CASCADE design refresh model can be trusted, however, it is necessary to verify *its* outputs against the outputs of existing models.

This chapter demonstrates the validity of the new CASCADE design refresh model by verifying the two main elements of the model: First, the CASCADE cost model is verified in § 4.1 by showing that the CASCADE cost model converges to known results under the same set of assumptions. Next, § 4.2 proves that the new CASCADE formulation produces design refresh plans that are always as good as, but often better than, the original MILP formulation. Finally, § 4.3 explores some of the general trends that are observed when using the CASCADE formulation. These observations will aid in the analysis of results in Chapter 5.

4.1 Cost Model Verification

Before Equation 21 (The CASCADE cost furmulation) should be used to examine new trends, it is important to first verify it against existing accepted results in the literature. For example, as §2.6.3 notes, the Porter Design Refresh model is one of the most well-understood and widely-cited design refresh models in the literature, despite its simplicity. Thus, to be trusted, Equation 21 must agree with Porter's Design Refresh model under its assumptions.

Porter's model assumes that only one component has gone obsolete, and that the only costs to consider are the cost of redesigning the system/component, and the cost of a life-of-type buy. In the context of Equation 21, this means that $c_i^{rep} = 0$, $c_i^H = 0$, and $c^{NRE} = 0$. Applying these assumptions, as well as parameters $c_i^{LOT} = 10$ and $c_i^{red} = 100,000$ yields the result depicted in Figure 38. Note that Figure 38 is virtually identical to Figure 24 (the original formulation), suggesting that Equation 21 can be trusted under the assumptions of the original Porter Design Refresh model.

Since Equation 21 agrees with the Porter Design Refresh Model, it can then be used to extend the analysis by relaxing the assumptions of the original model. For instance, the original formulation assumes that there is no cost to maintain components that are being held for a life-of-type buy (i.e. $c_i^H = 0$). While this simplifies the math and provides an easily-understandable result, it does not fully capture the cost of storing components. Equation 21 allows for this assumption to be relaxed by setting $c_i^H > 0$. As Figure 39 shows, the cost of a single obsolescence event increases monotonically as c_i^H increases, as expected[1].

Beyond serving as a mere extension of the Porter Design Refresh model, Equation 21 enables the calculation of obsolescence cost under uncertainty. Figure 40 depicts the cost of obsolescence for a single-component system relative to the date of redesign,

[1] A mathematical proof of this is given in Appendix C.

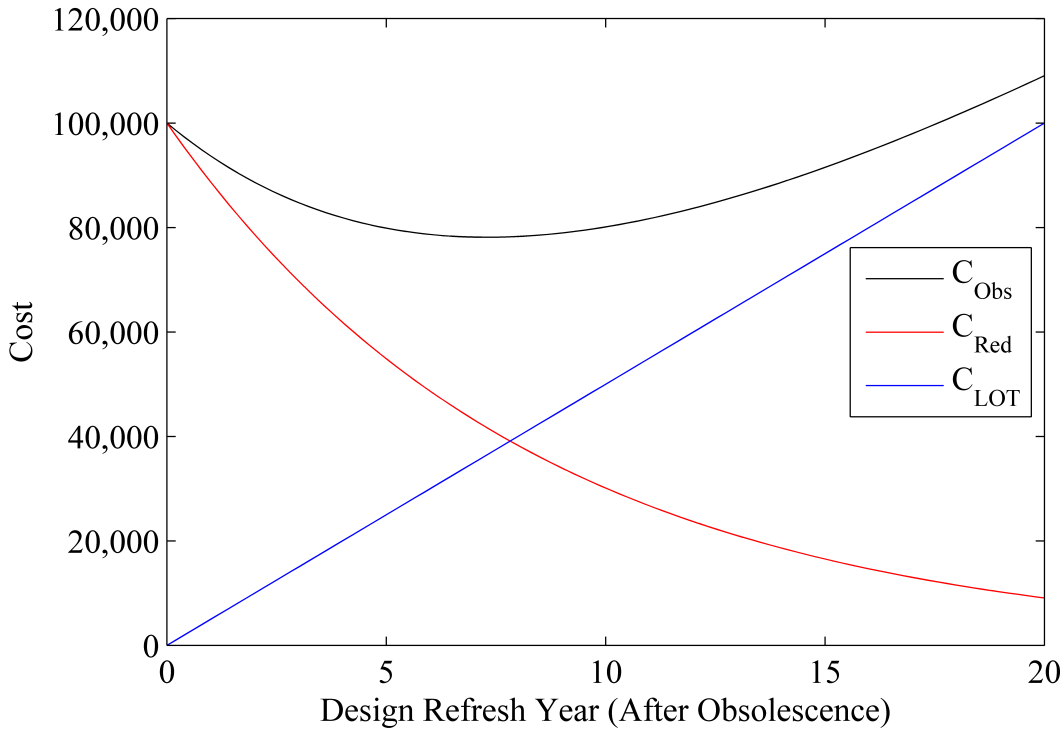

Figure 38: Verification test of Equation 21 using the assumptions of the Porter Design Refresh model. Note that these results mirror those produced by the original formulation of the Porter Design Refresh model, suggesting that Equation 21 can be trusted under these conditions.

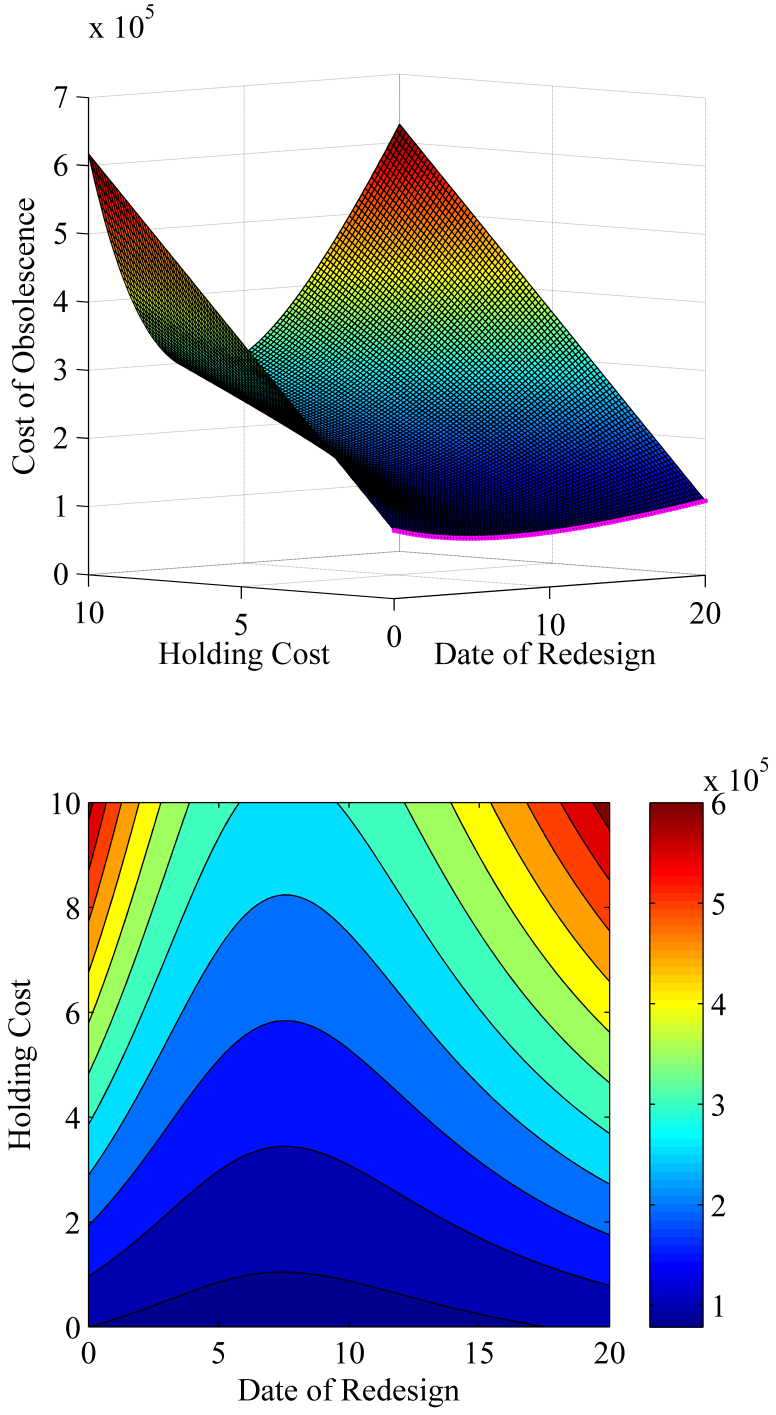

Figure 39: Cost of obsolescence for a single component ($c_i^{LOT} = 10$, $c_i^{Red} = 100,000$) at varying redesign dates ($0 \leq d_{red} \leq 20$) with varying annual holding cost ($0 \leq c_i^H \leq 10$). Note that the red curve at $c_i^H = 0$ in the surface plot (left) represents the result, which assumes that there is no cost of holding/maintaining components.

assuming the date of obsolescence is unknown. For all valid redesign dates (i.e. $d_{red} \geq d_{obs}$), it is assumed that a LOT buy is performed between d_{obs} and d_{red}. Consequently, for every fixed redesign date, the cost of obsolescence decreases monotonically as the date of obsolescence increases up to $d_{red} = d_{obs}$, representing a "just in time" replacement. Every vertical slice through Figure 40 is essentially a version of the curve shown in Figure 38 — just displaced in time.

4.2 CASCADE MILP Verification

Section 3.1.3 described the limitations of the original Mixed Integer Linear Programming (MILP) Design Refresh formulation, and Section 3.2 offered a new CASCADE MILP formulation which addressed those limitations. This new formulation is intended as a decision support tool that allows for the rapid generation of optimal design refresh plans. Before it should be used, however, it is necessary to verify the results of this new formulation. Beyond just conforming to the standard form of a MILP problem, it is also necessary to show that the CASCADE formulation produces feasible design refresh plans that the original formulation could not. This requires two main components: First it must be shown that all feasible solutions in the original MILP formulation are also feasible in the CASCADE formulation. This guarantees that if the optimum strategy is within the constraints of the original formulation, it will also be found by the CASCADE formulation. Second, it is necessary to show that under some set of conditions, the true optimum strategy falls outside of the constraints of the original MILP while still falling within the constraints of the CASCADE formulation.

This section is organized into three parts. Section 4.2.1 describes two reactive baseline strategies — namely the reactive Last-Time Buy and reactive Bridge-Buy strategies. These represent the status quo for dealing with obsolescence, and they serve as a "lower bar" against which to compare the CASCADE formulation. Next,

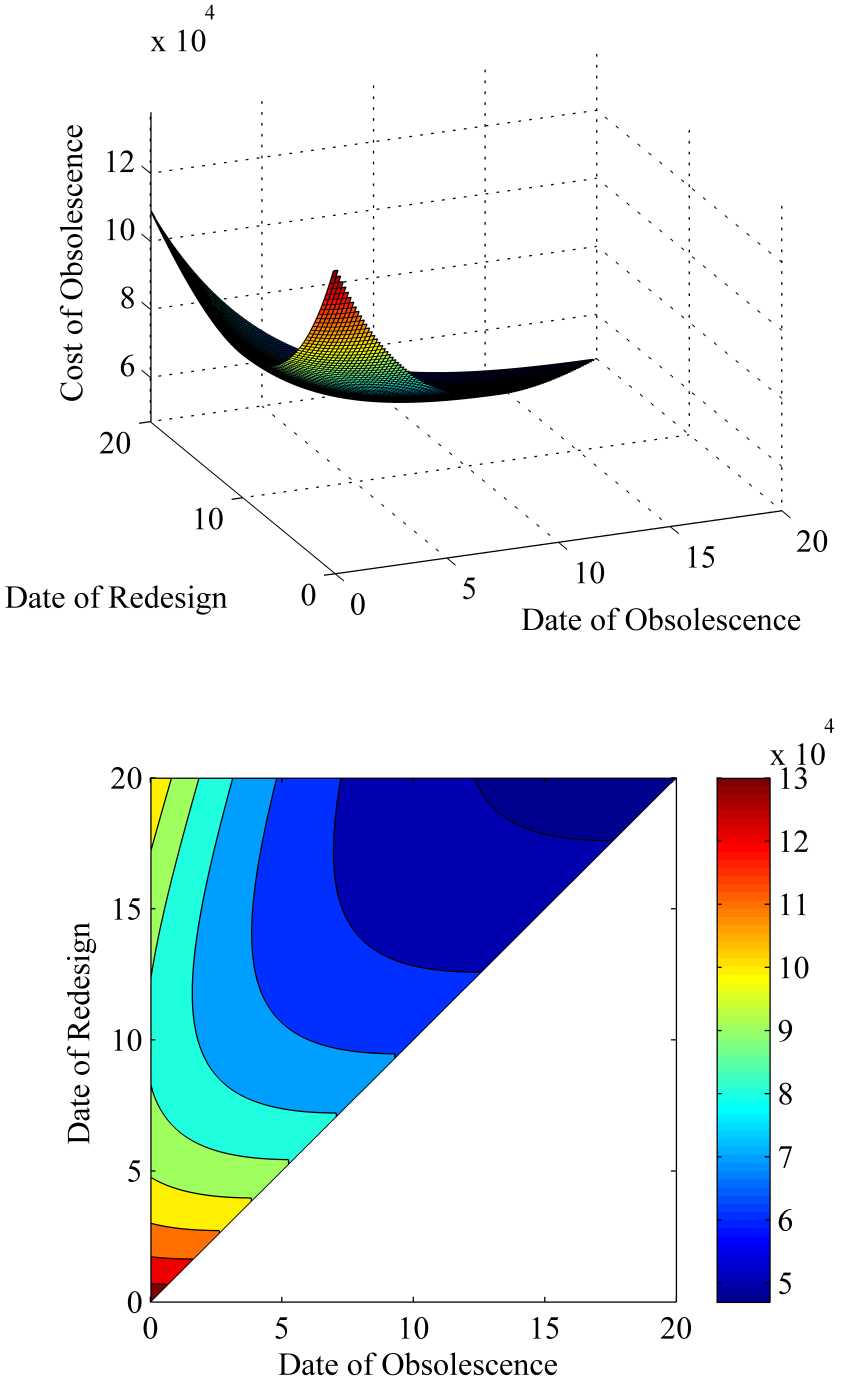

Figure 40: Cost of obsolescence for a single component given an uncertain obsolescence date. Note that all dates $d_{red} < d_{obs}$ are invalid, and are therefore omitted.

Section 4.2.2 provides a proof that the set of feasible solutions in the CASCADE MILP formulation are a proper superset of the set of feasible solutions of the original MILP. Finally, Section 4.2.3 provides evidence that high redesign or system-level NRE costs will cause the true optimum plan to fall outside the constraints of the original MILP.

4.2.1 Reactive Baselines

As noted in § 2.6, reactive obsolescence mitigation strategies are the most basic ways to deal with obsolescence. As the name implies, reactive strategies do not "look ahead" at predicted obsolescence events, and instead deal with obsolescence reactively — *after* it has occurred. Thus, reactive strategies represent the "lower bar" for obsolescence management. In other words, any optimized strategy must necessarily be better than a purely reactive strategy. To verify that the CASCADE formulation meets this requirement, is tested against two reactive baselines: the Reactive Bridge-Buy, and the Reactive Last-Time Buy.

In the *Reactive Bridge-Buy* strategy, every instance of obsolescence is addressed with a short-term LOT buy which bridges the period between the date of obsolescence and the next available redesign date. As a result, many redesigns will take place over the course of the system's life cycle under this reactive strategy. The compliment to this strategy is the *Reactive Last-Time-Buy* strategy. Under this strategy, no redesigns are ever performed. Instead, whenever obsolescence is encountered, a single LOT buy is performed which spans the period between the date of obsolescence and the system's end of life. Figure 41 and 42 show example system timelines for the Reactive Bridge-Buy and Reactive Last-Time Buy strategies, respectively.

4.2.2 Proof of CASCADE Optimality

Proving that the CASCADE MILP formulation produces a better optimum than the original formulation requires two elements: First, it must be shown that the Mixed

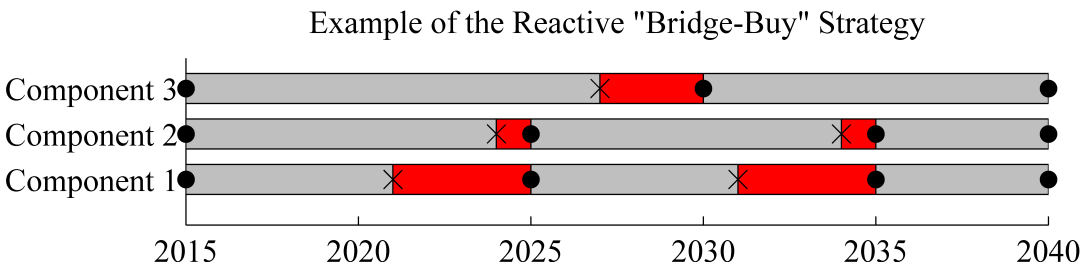

Figure 41: Example of a Reactive Bridge-Buy timeline. At each instance of obsolescence, a short-term LOT buy is made which spans until the next available redesign date.

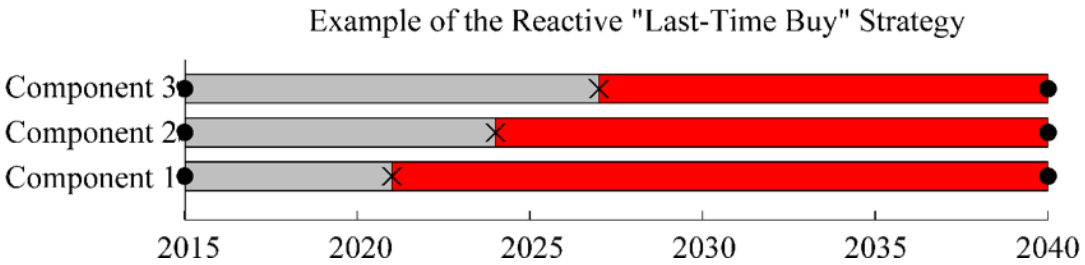

Figure 42: Example of a Reactive Last-Time Buy timeline. At the first instance of obsolescence for each component, a single LOT buy is made which spans until the system's end of life.

Integer Linear Set (the set of all feasible solutions) of the original MILP formulation is a proper subset of the Mixed Integer Linear Set of the CASCADE formulation. In other words, it must be proven that every feasible solution in the original formulation is also feasible in the new formulation. Second, it must be shown that some set of parameters will cause the original MILP formulation to produce a design refresh strategy that is less optimal than the CASCADE formulation.

To demonstrate that CASCADE's Mixed Integer Linear Set is a proper superset of the original MILP's, it is helpful to first re-write the objective function and constraints in a similar form. To do this, the first step is to re-write the decision variables of the original MILP to match the CASCADE MILP. Thus, the decision variable x_{ij}

becomes $x_{i,d_{red}}$, y_j becomes $y_{d_{red}}$, and z_{ik} becomes $z_{i,d_{red},d_{block}}$[2]. With this change, the objective function for the original MILP formulation is mathematically identical to that of the CASCADE MILP — namely:

$$C_{obs} = \sum_{i=1}^{m} \left(\sum_{d_{red} \in D_{red}} C^{R}_{i,d_{red}} x_{i,d_{red}} \right) + \sum_{d_{red} \in D_{red}} C^{NRE}_{d_{red}} y_{d_{red}} + \sum_{i=1}^{m} \left(\sum_{d_{red} \in D_{red}} \left(\sum_{d_{bridge} \in D_{bridge}} C^{B}_{i,d_{red},d_{bridge}} z_{i,d_{red},d_{bridge}} \right) \right)$$

The next step is to re-write the constraints of the original MILP to match the form of the constraints of the CASCADE MILP. In the CASCADE MILP, Constraint 18a says that whenever a redesign takes place for a given component, then *at most* one bridging strategy can be taken which starts at Δ_i years beyond that redesign date. Constraint 18a is repeated below:

$$\text{For each } i,\ d^*_{red} \in D_{red}\text{:}$$
$$x_{i,d^*_{red}} \geq \sum_{d_{red} > d^*_{red} + \Delta_i} z_{i,d_{red},d^*_{red}+\Delta_i}$$

In the original MILP formulation, this constraint still applies, but only one $z_{i,d_{red},d_{bridge}}$ is ever considered — namely, the $z_{i,d_{red},d_{bridge}}$ that terminates at the next available redesign date. Therefore, this constraint, as applied to the original MILP, is written as:

$$\text{For each } i,\ d^*_{red} \in D_{red}\text{:} \qquad (22a)$$
$$x_{i,d^*_{red}} \geq z_{i,d_{next},d^*_{red}+\Delta_i}, \text{ where } d_{next} = \min(\{d_{red} > d^*_{red} + \Delta_i\})$$

[2]Note that although the decision variable z has an additional subscript (namely d_{red}), it is merely there so that a more direct comparison can be made to the CASCADE MILP formulation. That is, the subscript d_{red} is superfluous for the decision variable z since the original MILP formulation stipulates that bridging strategies only be employed until the next available redesign, so d_{red} can only ever take on one value in the original MILP formulation. This will be made clear when the constraints are defined.

Thus, any $\left\{x_{i,d_{red}}, y_{d_{red}}, z_{i,d_{red},d_{bridge}}\right\}$ that satisfies Constraint 22a of the original MILP also satisfies Constraint 18a of the CASCADE MILP.

Constraint 18b in the CASCADE MILP formulation says that whenever *any* bridging action ends at date $t = d^*_{red}$, then a redesign must take place at $t = d^*_{red}$[3]:

$$\text{For each } i,\ d^*_{red} \in D_{red}:$$
$$x_{i,d^*_{red}} \geq \sum_{d_{bridge} < d^*_{red}} z_{i,d^*_{red},d_{bridge}}$$

As before, this constraint still applies to the original MILP formulation, but only one $z_{i,d_{red},d_{bridge}}$ is ever considered — namely, the $z_{i,d_{red},d_{bridge}}$ that starts at the previous obsolescence date. Therefore, this constraint, as applied to the original MILP, is written:

$$\text{For each } i,\ d^*_{red} \in D_{red}: \qquad (22b)$$
$$x_{i,d^*_{red}} \geq z_{i,d_{next},d^*_{red},d_{prev}}, \text{ where } d_{prev} = \max(\{d_{bridge} < d^*_{red}\})$$

Thus, any $\left\{x_{i,d_{red}}, y_{d_{red}}, z_{i,d_{red},d_{bridge}}\right\}$ that satisfies Constraint 22b of the original MILP will always also satisfy Constraint 18b of the CASCADE MILP.

Constraint 18c in the CASCADE MILP formulation says that for each component, i, and within every time period Δ_i, one of the following must be true:

- The component must be redesigned
- A bridging action must take place
- A previous bridging action must be underway which spans the Δ_i period

Effectively, Constraint 18c says that "do nothing" is not an acceptable solution. This same constraint holds true for the original MILP formulation, so Constraint 22c for the original MILP is:

[3]Another way to say this is that whenever a redesign takes place at $t = d^*_{red}$, then at most one bridging action may take place which terminates at $t = d^*_{red}$.

$$\text{For each } i,\ d^*_{red} \in D_{red}\text{:} \sum_{d^*_{red}<d_{red}<d^*_{red}+\Delta_i} x_{i,d_{red}} + \sum_{d_{bridge}\leq d^*_{red}+\Delta_i} \left(\sum_{d_{red}>d^*_{red}+\Delta_i} z_{i,d_{red},d_{bridge}} \right) \geq 1 \tag{22c}$$

Since Constraints 18c and 22c are identical, any $\left\{x_{i,d_{red}}, y_{d_{red}}, z_{i,d_{red},d_{bridge}}\right\}$ that satisfies Constraint 22c of the original MILP will always also satisfy Constraint 18c of the CASCADE MILP.

Finally, Constraint 18d in the CASCADE MILP formulation says that whenever *any* component is redesigned at date $t = d_{red}$, then the program must incur a system-level non-recurring engineering (NRE) cost. This same constraint holds true for the original MILP formulation, so Constraint 22d for the original MILP is:

$$\text{For each } d_{red} \in D_{red}\text{:} \quad \sum_{i=1}^{m} x_{i,d_{red}} \leq M y_{d_{red}} \text{ For } M \gg m \tag{22d}$$

Since Constraints 18d and 22d are identical, any $\left\{x_{i,d_{red}}, y_{d_{red}}, z_{i,d_{red},d_{bridge}}\right\}$ that satisfies Constraint 22d of the original MILP will always also satisfy Constraint 18d of the CASCADE MILP.

Combining the adapted objective function with Constraints 22a through 22d yields:

Minimize

$$C_{obs} = \sum_{i=1}^{m} \left(\sum_{d_{red} \in D_{red}} C^{R}_{i,d_{red}} x_{i,d_{red}} \right) + \sum_{d_{red} \in D_{red}} C^{NRE}_{d_{red}} y_{d_{red}} + \sum_{i=1}^{m} \left(\sum_{d_{red} \in D_{red}} \left(\sum_{d_{bridge} \in D_{bridge}} C^{B}_{i,d_{red},d_{bridge}} z_{i,d_{red},d_{bridge}} \right) \right)$$

S.T.

$$\begin{aligned}
&\text{For each } i,\ d^{*}_{red} \in D_{red}\text{: } x_{i,d^{*}_{red}} \geq z_{i,d_{next},d^{*}_{red}+\Delta_i}, \\
&\quad \text{where } d_{next} = \min(\{d_{red} > d^{*}_{red} + \Delta_i\}) \\
&\text{For each } i,\ d^{*}_{red} \in D_{red}\text{: } x_{i,d^{*}_{red}} \geq z_{i,d_{next},d^{*}_{red},d_{prev}}, \\
&\quad \text{where } d_{prev} = \max(\{d_{bridge} < d^{*}_{red}\}) \\
&\text{For each } i,\ d^{*}_{red} \in D_{red}\text{: } \sum_{d^{*}_{red} < d_{red} < d^{*}_{red}+\Delta_i} x_{i,d_{red}} \\
&+ \sum_{d_{bridge} \leq d^{*}_{red}+\Delta_i} \left(\sum_{d_{red} > d^{*}_{red}+\Delta_i} z_{i,d_{red},d_{bridge}} \right) \geq 1 \\
&\text{For each } d_{red} \in D_{red}\text{: } \sum_{i=1}^{m} x_{i,d_{red}} \leq M y_{d_{red}} \text{ For } M \gg m \\
&\text{For each } i,\ d_{red} \in D_{red}\text{: } x_{i,d_{red}} \in \{0,1\} \\
&\text{For each } i,\ d_{red} \in D_{red}\text{: } y_{d_{red}} \in \{0,1\} \\
&\text{For each } i,\ d_{red} \in D_{red},\ d_{bridge} \in D_{bridge} : z_{i,d_{red},d_{bridge}} \in \{0,1\}
\end{aligned} \tag{23}$$

Since the objective functions are identical, and since all $\left\{x_{i,d_{red}}, y_{d_{red}}, z_{i,d_{red},d_{bridge}}\right\}$ which satisfy the original MILP (Equation 23) also satisfy the CASCADE MILP (Equation 19) — but not the other way around — it must be true that the set of feasible solutions in the original MILP represent a proper subset of those of the CASCADE MILP. Therefore, the CASCADE MILP will always produce a design refresh strategy which is as good as, or better than, the original MILP. A similar process can be followed to show that the CASCADE formulation produces strategies

as good as, or better than, the reactive baselines. This is done in Appendix E.

4.2.3 Comparison Against Baseline Strategies

The previous section proves that the CASCADE MILP formulation explores a feasible space that fully contains the feasible space of the original MILP formulation. By definition, then, the CASCADE formulation will always produce an optimum obsolescence mitigation plan that is as good as or better than the optimum plan produced by the original formulation. The CASCADE formulation's larger feasible space is moot, however, if the optimum obsolescence mitigation plan still falls inside the constraints of the original formulation. Therefore, to show that the original formulation should be abandoned in favor of the CASCADE formulation, it is necessary to show that there are parameters which cause the optimum plan to fall outside the constraints of the original formulation.

To provide context for this comparison, a 3-component example system was created which represents a generic consumer-grade electronics system. The system and component-level parameters used are provided in Table 9. Note that these values are not meant to represent a *particular* system, but are instead used to highlight the primary differences between the CASCADE formulation and the baseline strategies.

Figures 43 and 44 depict the Reactive Last-Time Buy and the Reactive Bridge-Buy baseline strategies, respectively. As discussed in § 4.2.1, the Reactive Last-Time Buy baseline strategy allows each component to become obsolete exactly once, at which point a single long-term LOT buy is made which lasts until the end of the system's life. Under the Reactive Last-Time Buy baseline obsolescence is encountered only three times, but the resulting obsolescence mitigation cost is $735,085. Under the Reactive Bridge-Buy baseline, each time obsolescence is encountered for each component, a short-term LOT buy is made which lasts until the next-available redesign date, at which point the component is replaced. Using the parameters listed in Table 9,

Table 9: Table of component-level and system-level parameters for use in the verification of the CASCADE MILP formulation.

	Component 1	Component 2	Component 3
Δ_i	6 yrs.	9 yrs.	12 yrs.
c_i^{LOT}	\$8	\$10	\$12
c_i^{H}	\$10	\$10	\$10
c_i^{red}	\$80,000	\$100,000	\$120,000

	System
c^{NRE}	\$1,000,000
d_{IOC}	2015
Production Dates	$\{2015, 2020, 2025, 2030, 2035\}$
d_{EOL}	2040
d_{base}	2015

obsolescence is encountered a total of five times using this strategy, resulting in the need for five short-term LOT buys and five redesigns. Because of this, under the Reactive Bridge-Buy baseline \$719,083 must be spent to mitigate obsolescence.

Figures 45 and 46 depict the optimum design refresh plans generated by the original MILP formulation and the CASCADE MILP formulation, respectively. Both of these pro-active obsolescence mitigation approaches allow for future obsolescence events to be intelligently planned for throughout the system's life cycle. Because of this, both MILP formulations produce obsolescence mitigation plans which cost significantly less than either of the reactive strategies. Using the original MILP formulation, five instance of obsolescence are encountered, requiring five short-term LOT buys and five redesigns. However, unlike the Reactive Bridge-Buy baseline shown in Figure 44, many of the redesigns are scheduled concurrently using the original MILP formulation, therefore avoiding the need to pay the high system-level NRE cost multiple times. Because of this, the original MILP formulation produces an obsolescence mitigation plan which costs \$566,943.

Using the CASCADE MILP formulation, obsolescence is encountered five times,

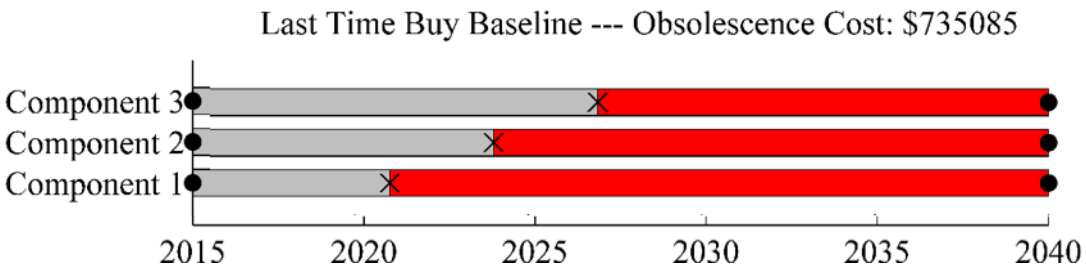

Figure 43: Obsolesce mitigation plan using the Reactive Last-Time Buy baseline. Following this strategy, the cost to mitigate obsolescence is $735,085.

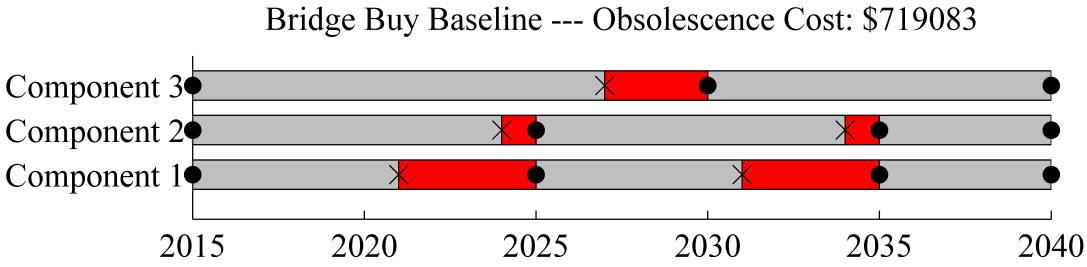

Figure 44: Obsolescence mitigation plan using the Reactive Bridge Buy baseline. Following this strategy, the cost to mitigate obsolescence is $719,083.

requiring a total of five short-term LOT buys. However, unlike the Original MILP formulation, the CASCADE formulation only requires three redesigns, all of which are scheduled concurrently. This further avoids the need to pay the high system-level NRE cost, resulting in a plan which costs only $439,349. The cost and schedule requirements for each of the obsolescence mitigation strategies compared in this section are presented in Table 10.

Table 10: Table of results showing the cost and schedule for mitigating obsolescence under varying design refresh strategies. The results clearly show that the CASCADE MILP formulation produces the lowest-cost obsolescence mitigation strategy given the inputs provided.

Strategy	Bridge Buy	Lifetime Buy	Original MILP	CASCADE MILP
Obsolescence Cost	$719,083	$735,085	$566,943	$439,349
# Re-Qualifications	3	0	2	1
# Redesigns	5	0	5	3
# LOT Buys	5	3	5	5

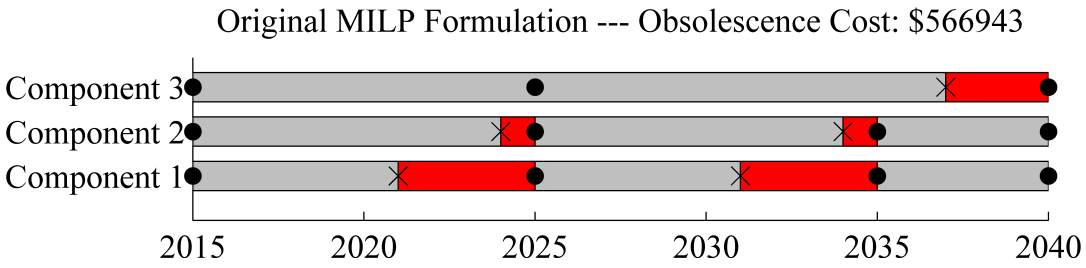

Figure 45: Obsolescence mitigation plan as optimized using the original MILP formulation. Following this strategy, the cost to mitigate obsolescence is $566,943.

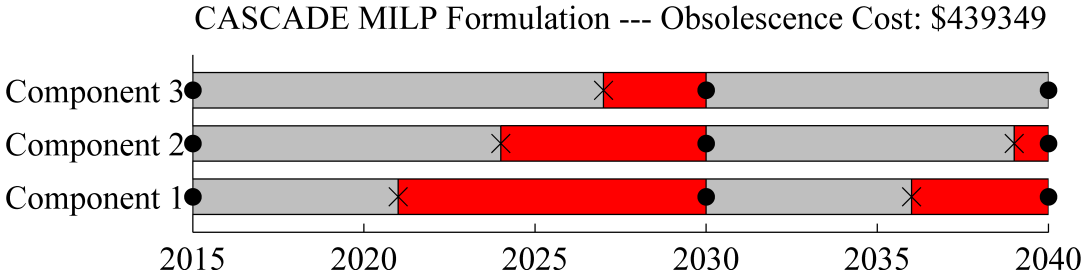

Figure 46: Obsolescence mitigation strategy as optimized using the new MILP formulation developed herein. Following this strategy, the cost to mitigate obsolescence is $439,349.

4.3 Observed Trends

As Section 4.2 showed, the CASCADE MILP produces optimal obsolescence mitigation plans which out-perform the status quo. CASCADE's larger feasible space enables the exploration of obsolescence mitigation plans which extend beyond the feasible space of the original MILP. This was demonstrated mathematically via the proof given in § 4.2.2, but further consideration is warranted to help understand the impact of this difference. This section compares the two MILP formulations, and explores how the mathematical differences between the two manifest themselves when the two methods are applied.

4.3.1 Parameter Effect on Optimum

Using either MILP formulation varying the cost parameters will push optimum strategy in one of two basic directions: towards a redesign-heavy strategy, or towards a zero-redesign strategy. Generally speaking, increasing the cost parameters associated

with redesigning components (e.g. c_i^{red} and c^{NRE}) will make redesigns less financially viable as compared to other obsolescence mitigation actions, thus pushing the optimum strategy towards a zero-redesign solution. Likewise, increasing cost parameters associated with bridging actions (e.g. c_i^{LOT}, c_i^H, and q_i) will make redesigns *more* financially viable relative to bridging actions, thus pushing the optimum strategy towards a redesign-heavy strategy. To demonstrate this, Figures 47 and 48 illustrate the effect of varying the system-level NRE cost using the original MILP and CASCADE MILP formulations, respectively. For simplicity, it is assumed that the only system-level NRE cost is the cost associated with certifying the system.

In both Figures 47 and 48, two basic phenomena are observed: First, as the cost to re-certify the system increases redesigns are planned concurrently in order to avoid paying the certification cost multiple times. Second, as the cost to re-certify increases the duration of bridging actions increases. This allows for redesigns to be skipped entirely to prevent the need for paying certification cost altogether. Both formulations produce the same optimum result when the certification cost is low (e.g. \$0–\$10,000), but as the cost to re-certify increases past \$100,000, the two formulations begin to diverge. The original MILP produces the same obsolescence mitigation plan for all $c^{NRE} \geq$ \$100,000 while the optimal plan continues to change under the CASCADE formulation. This is because the original MILP (unlike the CASCADE MILP) only allows for short-term bridging actions. Consequently, when the "true optimum" plan calls for long-term bridging actions, the original MILP reaches the limit of its feasible space. Since the CASCADE MILP explores a larger feasible space, it continues to produce optimal strategies past this limit.

4.3.2 Block Upgrade Effect on Optimum

The trends observed in § 4.3.1 assume that no block upgrades are scheduled. However, a profound change can be expected when the scheduling of block upgrades is

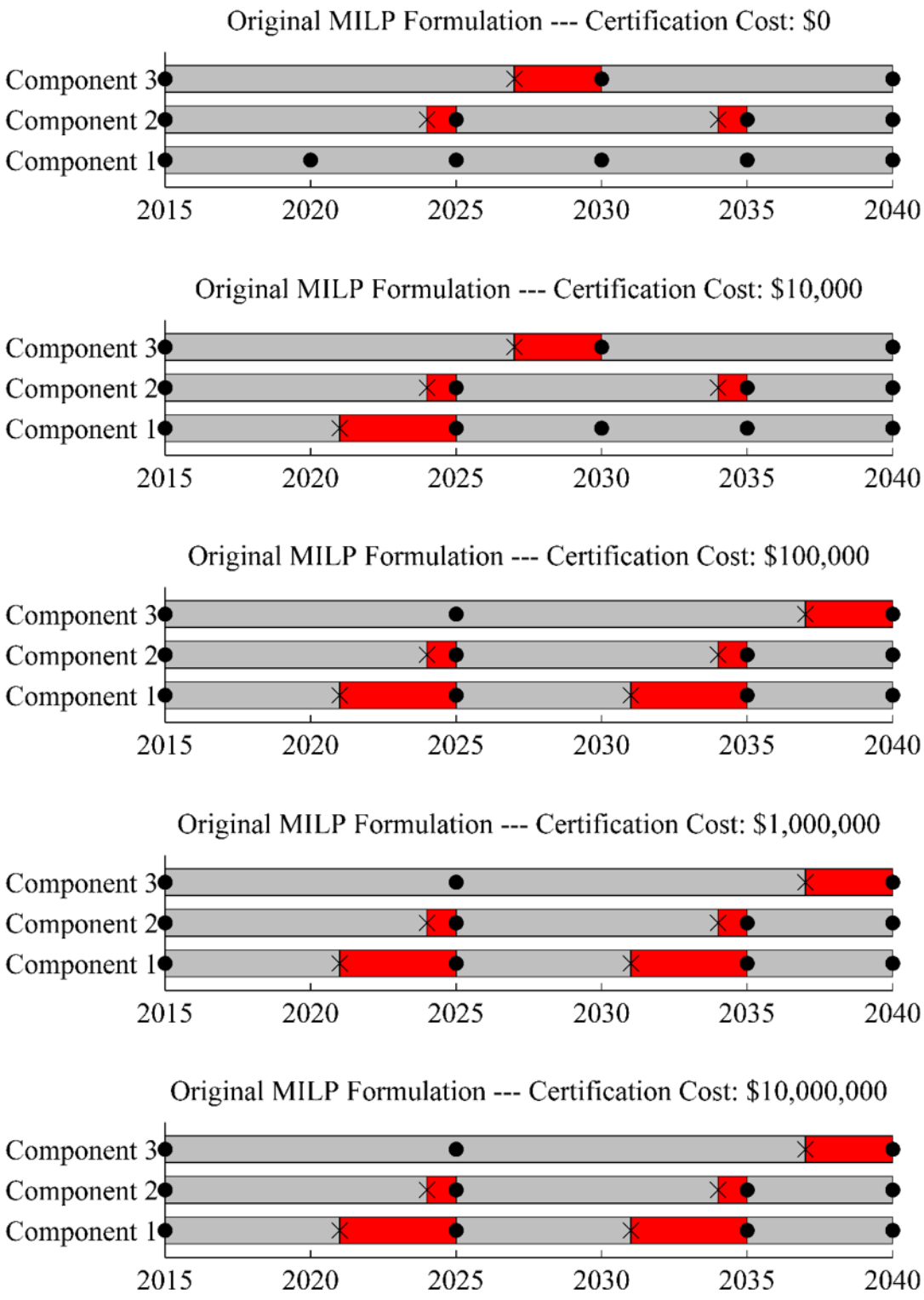

Figure 47: Optimized obsolescecne mitigation plans using the original MILP formulation with varying certification cost. As certification cost increases, two phenomena emerge: First, redesigns are planned concurrently in order to avoid paying certification cost multiple times. Second, the duration of bridging actions increases as the certification cost increases. This allows for redesigns to be skipped entirely to prevent the need for paying certification cost altogether.

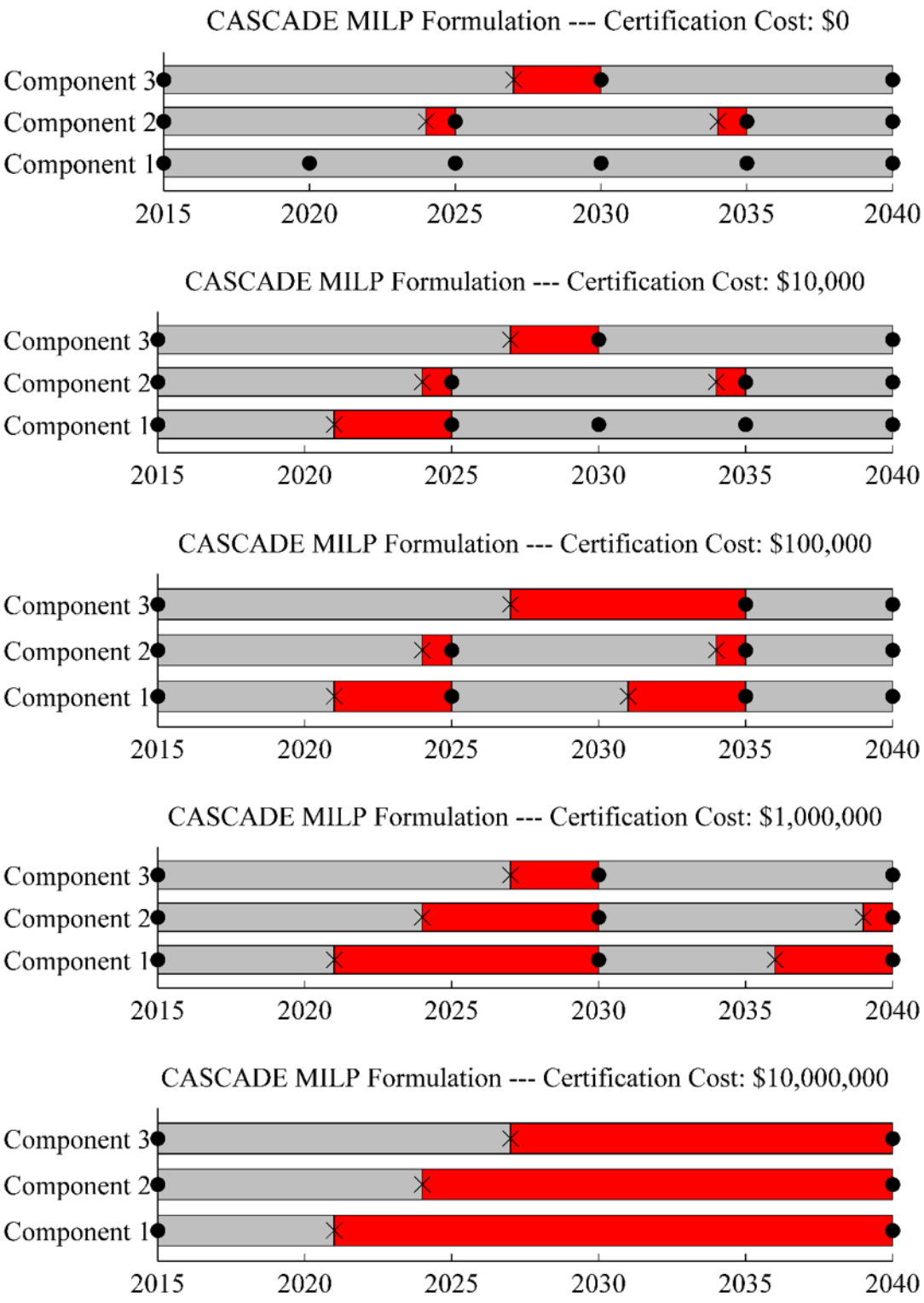

Figure 48: Optimized obsolescecne mitigation plans using the CASCADE MILP formulation with varying certification cost. As certification cost increases, two phenomena emerge: First, redesigns are planned concurrently in order to avoid paying certification cost multiple times. Second, the duration of bridging actions increases as the certification cost increases. This allows for redesigns to be skipped entirely to prevent the need for paying certification cost altogether.

considered when generating obsolescence mitigation plans. As § 3.1.2.2 indicated, the system-level certification cost can be considered "free" when replacing components at the same time as block upgrades. This is because re-certification of the system is already required by the block upgrade, and is thus already budgeted. This phenomenon can be observed in Figure 49, which shows the optimal obsolescence mitigation plan using the CASCADE MILP formulation, and assuming the block upgrades are scheduled for 2020 and 2025.

4.4 Verification Summary

The experiments and proofs presented in this chapter serve as verification tests for the CASCADE design refresh method formulated in Chapter 3. First, the cost model that was developed in § 3.3 was verified against the results of the Porter Design Refresh model. It was shown that the CASCADE cost model reduces to the same form as the Porter Design Refresh model when the appropriate simplifying assumptions are made. Next, Section 4.2 verified the CASCADE MILP formulation by first providing a proof that the CASCADE formulation fully captures the decision space explored by the original MILP formulation. Section 4.2.3 then demonstrated that the CASCADE MILP finds optimal solutions which fall outside of the feasible space of the original MILP formulation. Finally, a demonstration of the differences between the CASCADE formulation and the original MILP formulation was given in Section 4.3. With both elements of the CASCADE Design Refresh Model verified, the model can now be confidently used in further analyses. Chapter 5 applies the CASCADE MILP formulation in experiments devised to address the hypotheses developed in Chapter 2

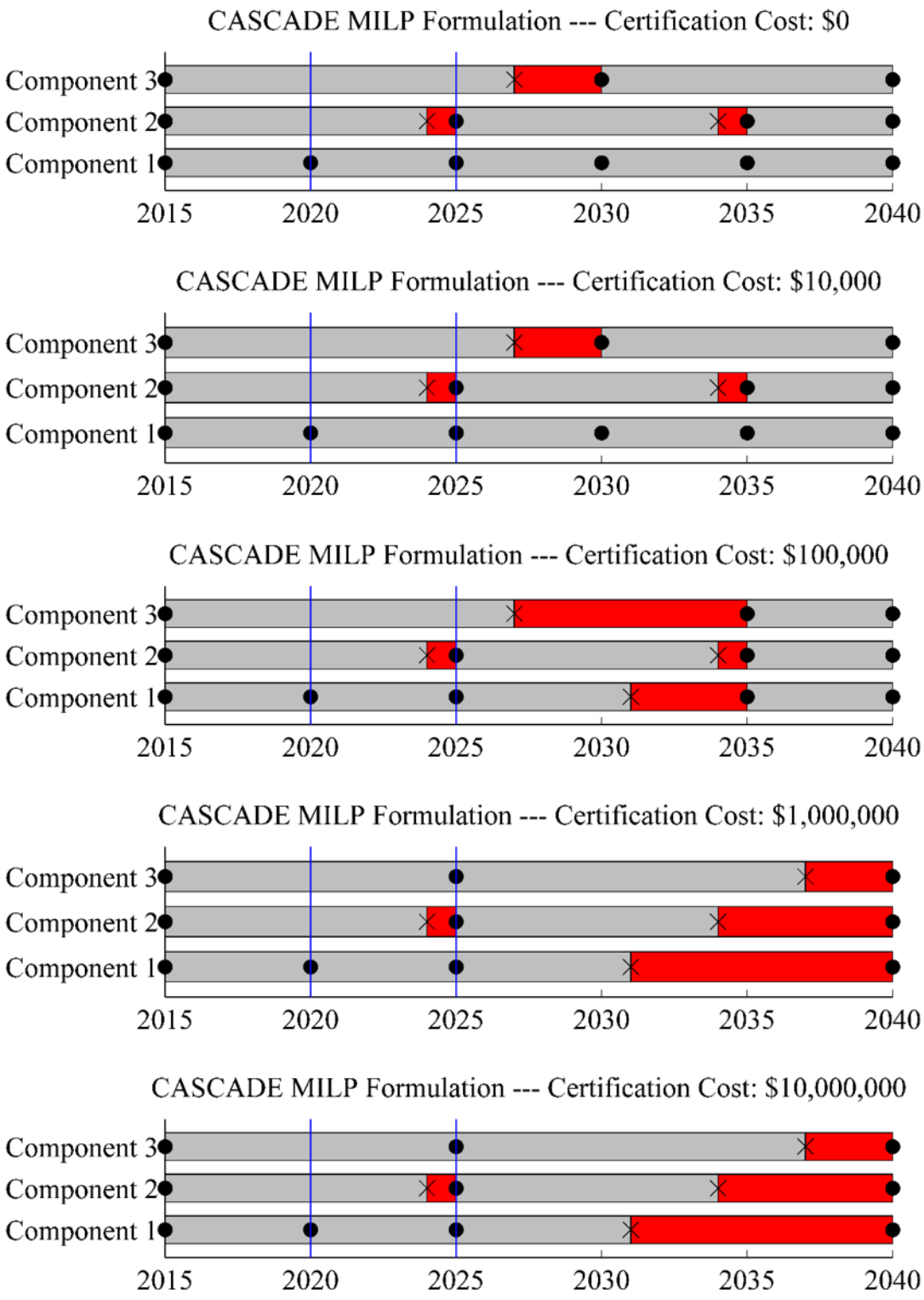

Figure 49: Optimized obsolescecne mitigation plans using the CASCADE MILP formulation with varying certification cost, assuming block upgrades are scheduled for 2020 and 2025. Note that the optimal strategy takes advantage of the "free" certification associated with block upgrades. Consequently, the optimal plans differ greatly from those depicted in Figure 48.

CHAPTER V

USE CASE EXPERIMENTS

Chapters 1 presented the background and literature review which motivated the pursuit of a new design refresh planning methodology. Potential candidates for this methodology were presented in Chapter 2, along with a set of guiding research questions and hypotheses. A brand new methodology, called CASCADE, was developed in Chapter 3. Having verified this methodology in Chapter 4 it can now be applied to a set of use cases to uncover new trends. In this chapter, the CASCADE methodology is applied to a set of representative systems in the form of use case experiments which are aimed at addressing the hypotheses generated in Chapter 2.

This chapter is organized into two sections: Section 5.1 presents Experiment 1, which tests the most suitable forecasting method for the overall CASCADE methodology. This experiment is further broken into two parts, which are introduced in §§ 5.1.1 and 5.1.2. Next, § 5.2 presents Experiment 2, which tests the effects of component modification on obsolescence cost. This experiment is also broken into two parts, which are presented in §§ 5.2.1 and 5.2.2.

5.1 Experiment 1: Testing Obsolescence Forecasting

The methodology developed in Chapter 2 requires the ability to forecast the date of obsolescence for existing and future components, as illustrated in Figure 50. Section 2.4 reviewed several different methods which can fill this need — including manufacturer inquiry, sales curve forecasting and data mining — in an attempt to answer Research Question 2.1, repeated below:

Research Question: 2.1

What is a suitable method of forecasting obsolescence?

The criteria for a "suitable method" are:

1. The method must provide a numeric estimate for the obsolescence date.
2. The method must utilize data that is available to industry outsiders
3. The method should provide confidence intervals where possible
4. The method must not be proprietary

Each of the options presented in § 2.4 meets some of the criteria outlined above, but none of the options meets all of the criteria. Therefore, further investigation is required to determine what can be deemed "most suitable."

The "most suitable" method for forecasting obsolescence depends on the relationship between forecast accuracy and obsolescence cost. If the optimum strategy is highly sensitive to forecast accuracy, then the most accurate forecasting method should be chosen. This notion is captured in Hypothesis 2.1a, repeated below:

Hypothesis: 2.1a

The accuracy of obsolescence forecasts will be a main driver of obsolescence cost for COTS-based systems.

This is not enough, however, to down-select the "most suitable" method. Beyond being a main driver, it is reasonable to hypothesize that the obsolescence cost, itself, decreases as accuracy increases. However, if obsolescence cost does not decrease (or, in fact, increases) with increasing forecast accuracy, then the choice of forecasting method is irrelevant. This is captured in Hypothesis 2.1b, repeated below:

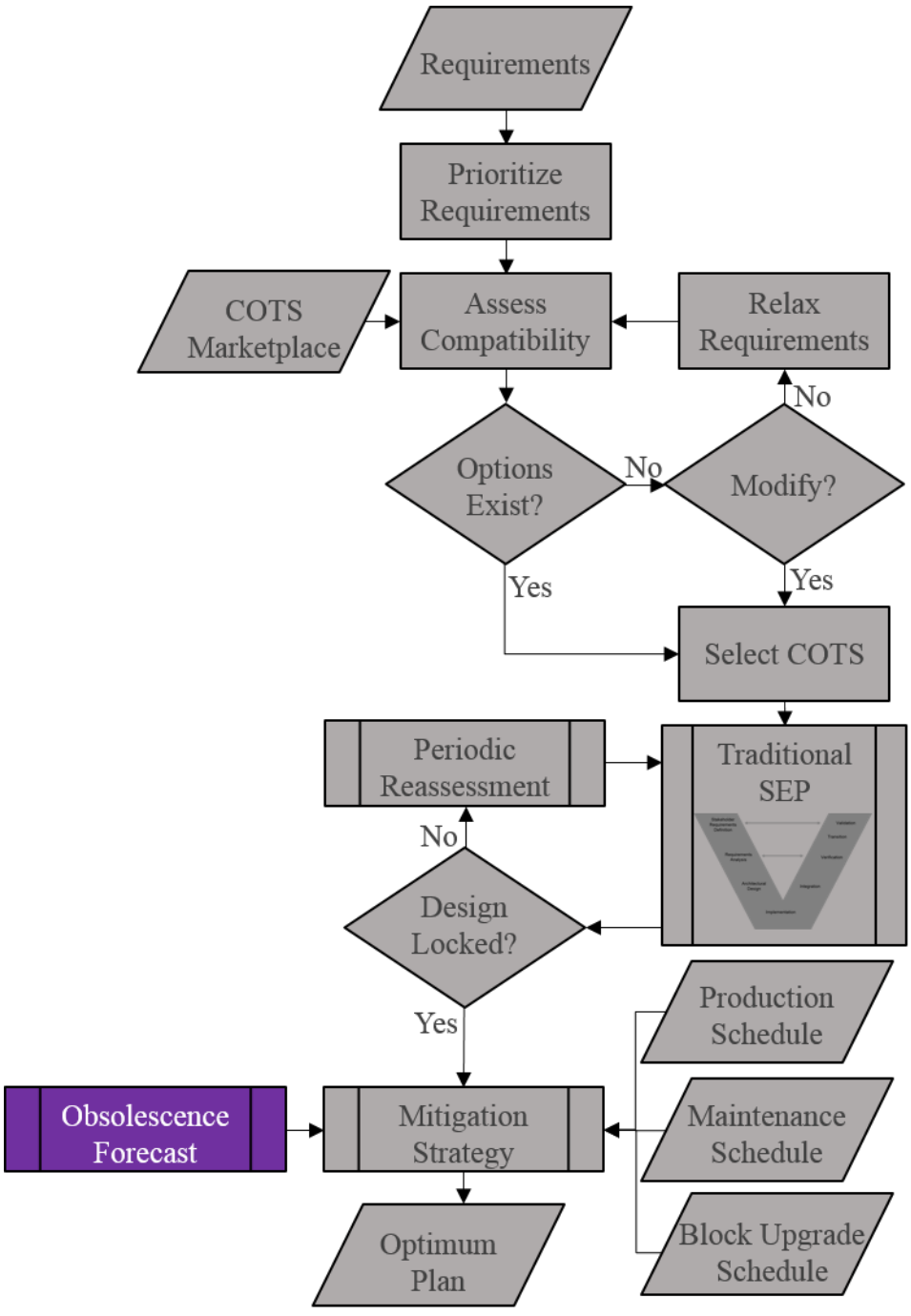

Figure 50: Experiment 1 focuses on the forecasting module of the CASCADE framework.

Hypothesis: 2.1b
The obsolescence cost will be non-increasing with increasing accuracy of obsolescence forecasts.

Thus, the selection of an obsolescence forecasting method is dependent on the support or rejection of these hypotheses. If either, or both, of these hypotheses is refuted, then the "most suitable" method is simply the method that requires the least data (e.g. Sales Curve Forecasting). If both of these hypotheses are supported, then the "most suitable" method is the method which provides the highest accuracy, despite any increase in data required (e.g. Data Mining). Since the selection of a "most-suitable" forecasting method depends on the rejection or support of two hypotheses, Experiment 1 is broken into two sub-experiments: Experiment 1a, presented in § 5.1.1, tests Hypothesis 2.1a. Next, Experiment 1b, presented in § 5.1.2, tests Hypothesis 2.1b.

5.1.1 Experiment 1a

Experiment 1a tests Hypothesis 2.1a, which postulates that obsolescence forecast accuracy is a main-driver of obsolescence cost for COTS-based systems. While it is tempting to simply run a Monte Carlo on the input parameters of the Mixed Integer Linear Programming model, this approach will not suffice. Each time the parameters are changed, a new optimum obsolescence mitigation plan is created. Therefore, running a Monte Carlo on the input parameters is liable to change the nature of the obsolescence mitigation plan, entirely. To avoid this, the following procedure is used:

Step 1: Define a Representative System — Firstly, a representative system is needed, about which an obsolescence mitigation plan is generated. For each component, cost and schedule parameters are defined, including annual failures

(q_i), unit cost (c_i^{LOT}), redesign cost (c_i^{red}), and holding cost (c_i^H)[1].

Step 2: Create "Forecast Data" — In lieu of forecasting obsolescence for actual components, it is assumed that forecast data already exists for each component. To represent these forecasted dates, a component life cycle, Δ_i, for each component is defined[2]. These Δ_i values represent the predicted dates of obsolescence for each component and are subject to uncertainty. This uncertainty is accounted for in Step 4, below.

Step 3: Build an Optimized Plan — The parameters of the representative system from Steps 1 and 2 are entered into the CASCADE Mixed Integer Linear Programming formulation developed in Chapter 3. The result of this step is an optimized obsolescence mitigation plan that specifies which obsolescence mitigation actions should be taken, and when. These actions include when redesigns are to be performed, when Life-of-Type buys are to be performed and their duration, and when the system incurs a system-level non-recurring engineering (NRE) cost.

Step 4: Define "Truth" Data — The cost and forecast parameters defined in Steps 1 and 2 represent predictions, but the actual values cannot be known with certainty *a priori*. To account for this uncertainty, it is assumed that the true cost and schedule parameters follow normal distributions, each with an independent mean centered about the forecasted obsolescence date, and an independent standard deviation.

[1]Note that the replacement cost (c_i^{rep}) associated with component substitution/emulation is ignored for this experiment. This is because the option to replace a component with a form-fit-function alternative does not always exist for all systems, and must be considered on a system-by-system basis. Thus, eliminating c_i^{rep} from the analysis generalizes the solution such that the resulting trends will be applicable to *all* systems.

[2]Recall that Δ_i represents the duration of the component life, and *not* a specific date of obsolescence. Despite this, defining Δ_i for each component is functionally equivalent to specifying an obsolescence date since, by definition, obsolescence can always be predicted to occur Δ_i years after a component's introduction.

Step 5: Compute the "True" Obsolescence Cost — Using the "truth" parameters from Step 4, and explicitly following the optimized strategy developed in Step 3, the true cost of obsolescence is computed. This value is presented as a relative uncertainty (i.e. a percent deviation from the expected value) so as to make the result system-agnostic.

Step 6: Repeat — Steps 4 and 5 are repeated using a Monte Carlo simulation to generate a histogram of output values.

The representative system used in this experiment represents a three-component avionics system. The cost values come from representative systems found via market research, and the reliability is based on military avionics systems [5, 75, 97, 98]. To protect proprietary information, the cost and reliability inputs have been normalized and are presented as unitless, and generic component names are used. The redesign cost for each component is 1, but the component cost (c_i^{LOT}) varies for each component. The system-level non-recurring engineering cost is half of the cost of redesign, or 0.5. As recommended by Porter, the annual holding cost (c_i^H) for each component is 25% of the cost of each component[39]. These values are given in Table 11. For the purposes of determining the sensitivity of the output on the variability of each input, it is assumed that each input has an uncertainty of ±10% of the expected value.

5.1.1.1 Experiment 1a Results and Analysis

Figure 51 shows the optimized design refresh plan for the example system used in Experiment 1a. The optimum plan allows Component 1 to go obsolete in 2021, Component 2 to go obsolete in 2024, and Component 3 to go obsolete in 2027. At these dates, a LOT buy is made for each component. Components 1 and 2 are replaced together in 2055. The expected obsolescence cost using this strategy is 4.760526 in normalized 2015 dollars.

Table 11: Table of component-level and system-level parameters for Experiment 1a. These cost values come from representative systems found via market research, and reliability is based off of the following sources: [5, 75, 97, 98]

	Component 1	Component 2	Component 3
Δ_i	6 yrs.	9 yrs.	12 yrs.
c_i^{LOT}	.0285	.021	.0255
c_i^{H}	.007125	.00525	.006375
c_i^{red}	1	1	1
q_i	0.1	0.1	0.1

	System
c^{NRE}	.5
d_{IOC}	2015
Production Dates	$\{2015, 2020, 2025, 2030, 2035, 2040, 2045, 2050, 2055\}$
d_{EOL}	2065
d_{base}	2015

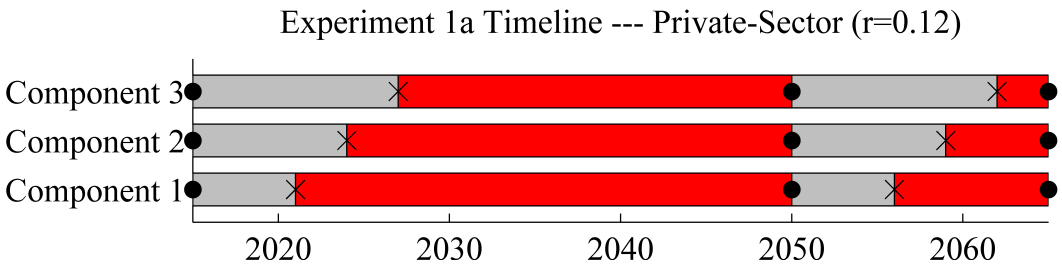

Figure 51: Optimized design refresh plan for Experiment 1a, using parameters defined in Table 11 and assuming a private sector discount rate (r=0.12).

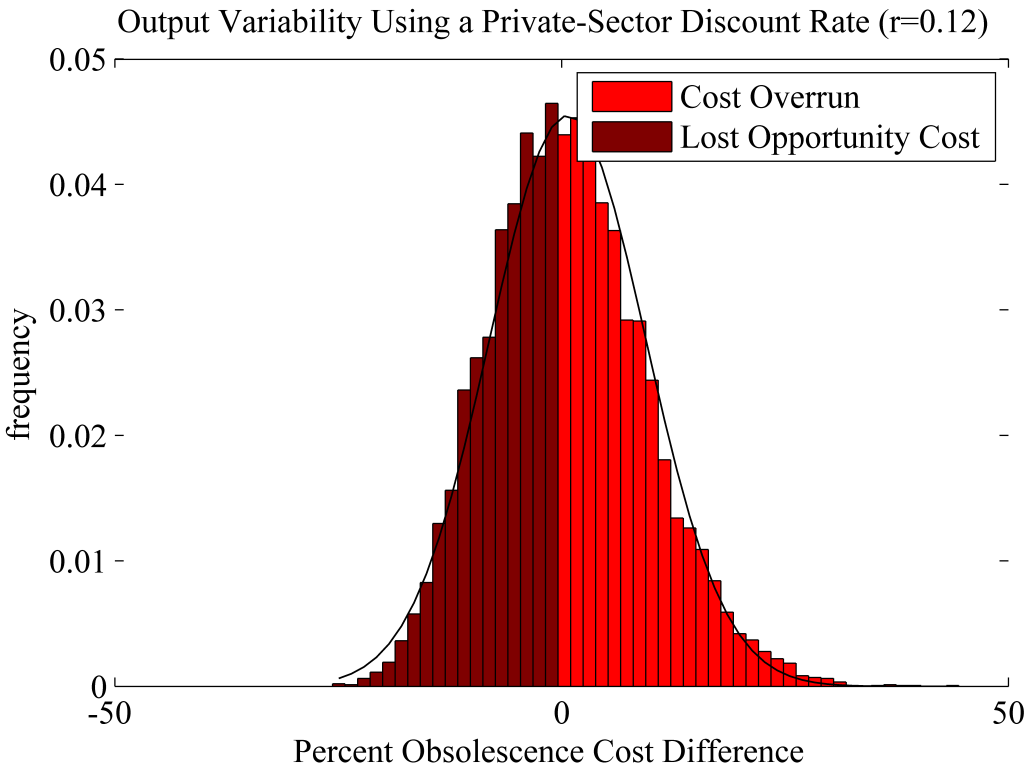

Figure 52: Histogram of % change in obsolescence cost, assuming 10% error in each input. Here, all input are allowed to vary simultaneously, and the discount rate reflects that of the private sector (r=0.12).

Using the design refresh plan shown in Figure 51, the "actual" obsolescence cost was computed using a 10,000-point Monte-Carlo simulation, applying a normal distribution to each input with a standard deviation of ±10% on each parameter. This was done in two ways: The first method varied each parameter one-at-a-time by ±10%. Thus, for each of the five parameters (c_i^H, c_i^{LOT}, c_i^{red}, q_i, and Δ_i), a 10,000-run Monte Carlo was run, resulting in five unique output distributions. Second, each parameter was allowed to vary simultaneously, resulting in a single output distribution.

Table 12 shows the results for each Monte Carlo run, including both one-at-a-time (OAT) inputs, and simultaneous input distributions when a private-sector discount rate (r=0.12) is used. As the results indicate, a forecast uncertainty of 10% (10% uncertainty in Δ_i) results in 6.05% uncertainty in the obsolescence cost. This represents a sensitivity of 0.4839, indicating that the variability of Δ_i accounts for about 48.4%

Table 12: Experiment 1a Monte Carlo results assuming a private-sector discount rate (r=0.12). Results are displayed for inputs varied one-at-a-time (OAT) and simultaneously.

Type	Input % Uncertainty	# Runs	Output % Uncertainty
OAT	$\Delta_i \pm 10\%$	10,000	$c_{obs} \pm 6.1\%$
OAT	$c_i^{LOT} \pm 10\%$	10,000	$c_{obs} \pm 2.0\%$
OAT	$c_i^{red} \pm 10\%$	10,000	$c_{obs} \pm 1.1\%$
OAT	$c_i^{H} \pm 10\%$	10,000	$c_{obs} \pm 3.0\%$
OAT	$q_i \pm 10\%$	10,000	$c_{obs} \pm 5.0\%$
Simult.	$\left\{\Delta_i, c_i^{LOT}, c_i^{red}, c_i^{H}, q_i\right\} \pm 10\%$	10,000	$c_{obs} \pm 8.7\%$

of the variability in c_{obs}. By comparison, variability in annual failures (q_i) accounts for about 32.8% of the variability of c_{obs}, and variability in holding cost (c_i^H) accounts for 11.8% of the variability of c_{obs}. Component cost (c_i^{LOT}) and redesign cost (c_i^{red}) collectively account for about 7% of the variability in c_{obs}. These sensitivities are shown in a Pareto Plot in Figure 53.

Interestingly, the impact of the input parameters on the output changes when switching from a private-sector discount rate (r=0.12) to a government discount rate (r=0.034). At the lower discount rate, the sensitivity of c_{obs} to variations in Δ_i drops dramatically from 48.4% to 17.4%. At the same time, the sensitivity of c_{obs} to variations in q_i grows from 32.8% to 49.8%, while the sensitivity to variations in c_i^H grows from 11.8% to 30.3%. Meanwhile, the sensitivity of c_{obs} to variations in c_i^{LOT} drops from 5.4% to 2.6%, and the sensitivity to variations in c_i^{red} drops from 1.6% to 0.0%. These sensitivities are depicted in a Pareto Plot in Figure 55, and the values for both government and private discount rates captured in Table 14.

This observed shift in parameter sensitivities can be explained by the nature of the optimum obsolescence mitigation plan generated for each discount rate. At the higher private-sector rate (r=0.12), the optimum obsolescence mitigation plan calls for three redesigns, and obsolescence is encountered a total of six times, as illustrated in Figure 51. When the discount rate is lowered to r=0.034, the optimum

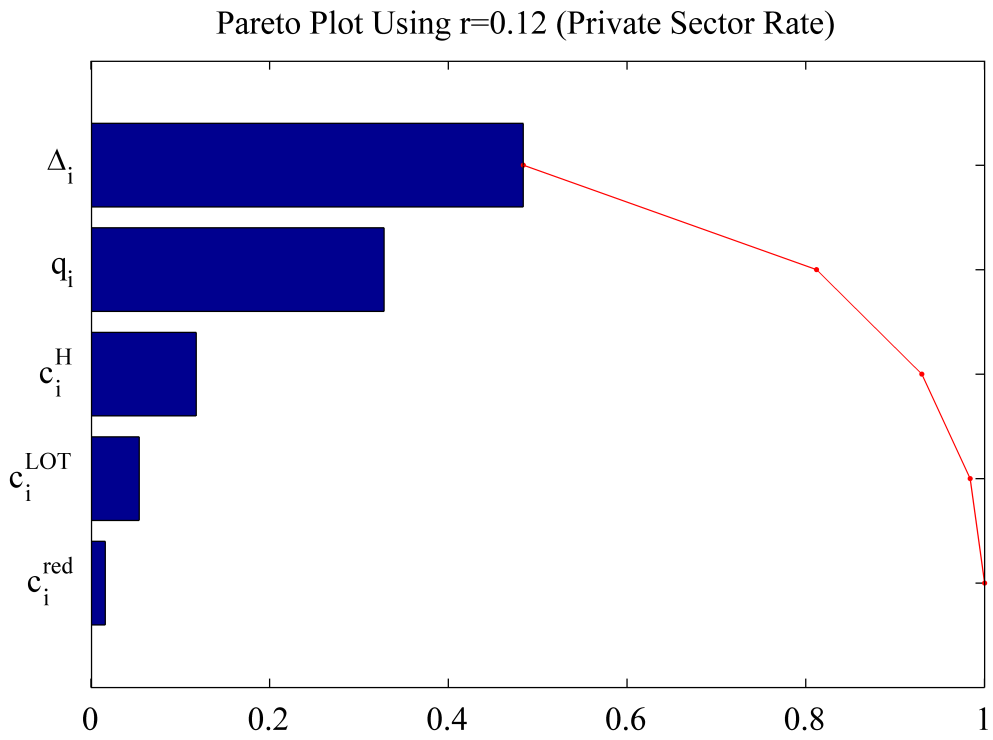

Figure 53: Pareto plot depicting the sensitivity of obsolescence mitigation cost (c_{obs}) to variations in each input parameter, assuming a private-sector discount rate (r=0.12). The blue bars represent the sensitivity of c_{obs} to each input, and the red line represents the cumulative effect of the inputs.

Table 13: Experiment 1a Monte Carlo results assuming a government discount rate (r=0.034). Results are displayed for inputs varied one-at-a-time (OAT) and simultaneously.

Type	Input % Uncertainty	# Runs	Output % Uncertainty
OAT	$\Delta_i \pm 10\%$	10,000	$c_{obs} \pm 1.4\%$
OAT	$c_i^{LOT} \pm 10\%$	10,000	$c_{obs} \pm 2.6\%$
OAT	$c_i^{red} \pm 10\%$	10,000	$c_{obs} \pm 0.0\%$
OAT	$c_i^{H} \pm 10\%$	10,000	$c_{obs} \pm 4.6\%$
OAT	$q_i \pm 10\%$	10,000	$c_{obs} \pm 5.9\%$
Simult.	$\{\Delta_i, c_i^{LOT}, c_i^{red}, c_i^{H}, q_i\} \pm 10\%$	10,000	$c_{obs} \pm 8.0\%$

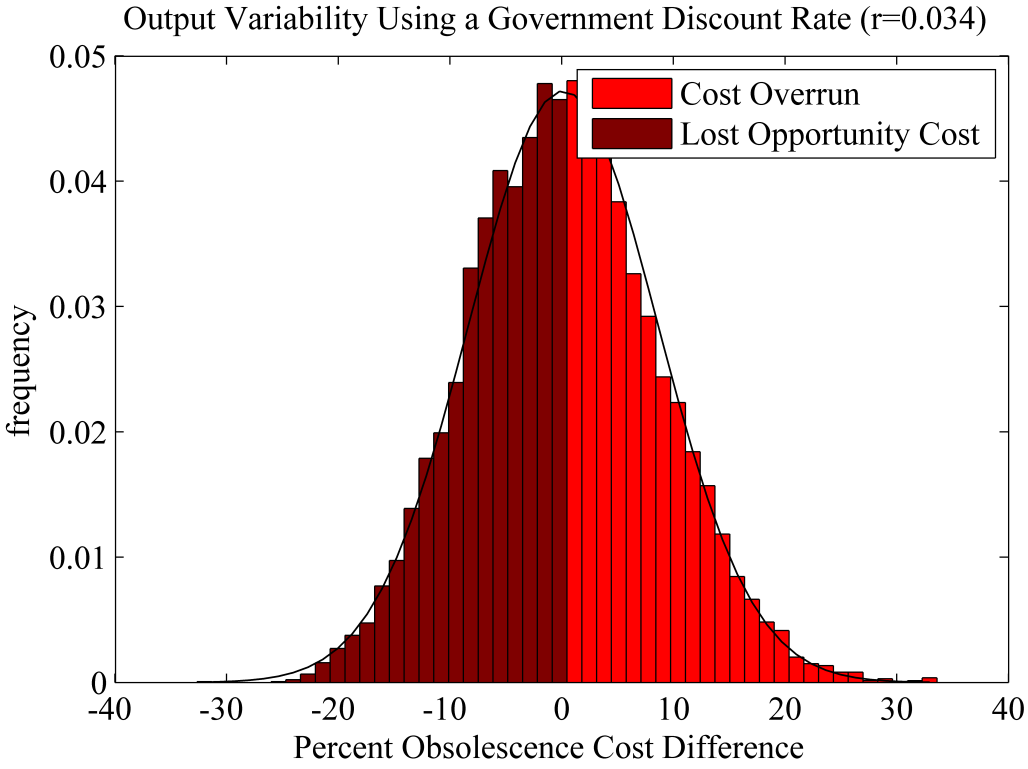

Figure 54: Histogram of % change in obsolescence cost, assuming 10% error in each input. Here, all input are allowed to vary simultaneously, and the discount rate reflects that of the government (r=0.034).

Table 14: Experiment 1a comparison of parameter sensitivities using a government discount rate (r=0.034) vs private-sector discount rate (r=0.12).

Parameter	Sensitivity Coefficient	
	Gov't Rate (r=0.034)	Private Rate (r=0.12)
Δ_i	17.4%	48.4%
c_i^{LOT}	2.6%	5.4%
c_i^{red}	0.0%	1.6%
c_i^{H}	30.3%	11.8%
q_i	49.8%	32.8%

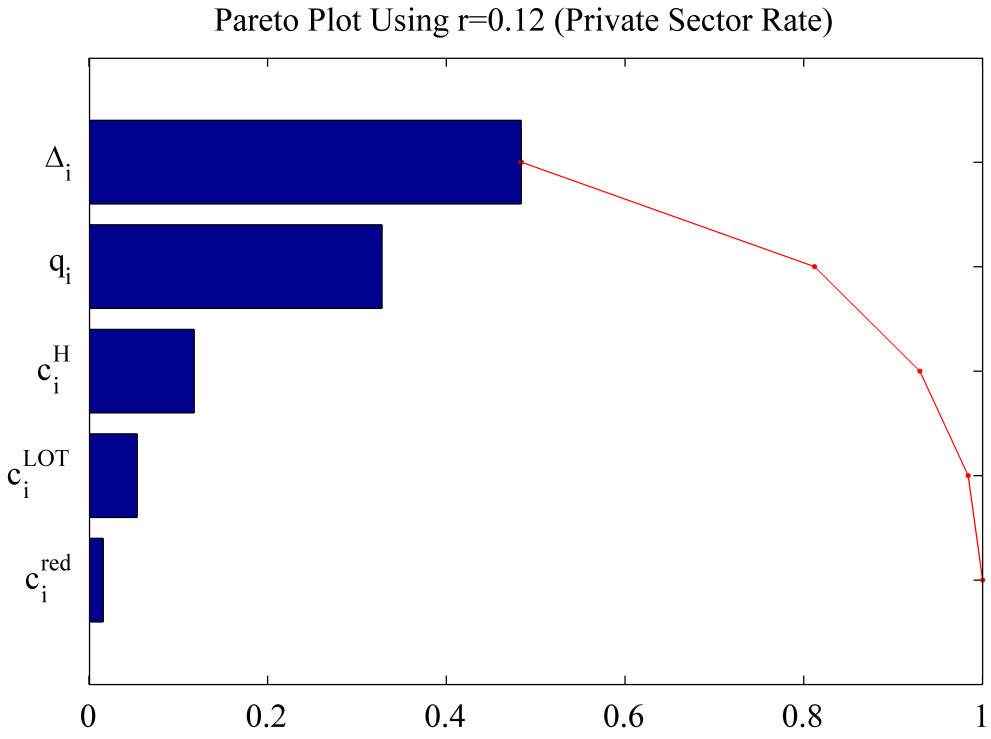

Figure 55: Pareto plot depicting the sensitivity of obsolescence mitigation cost (c_{obs}) to variations in each input parameter, assuming a government discount rate (r=0.034). The blue bars represent the sensitivity of c_{obs} to each input, and the red line represents the cumulative effect of the inputs.

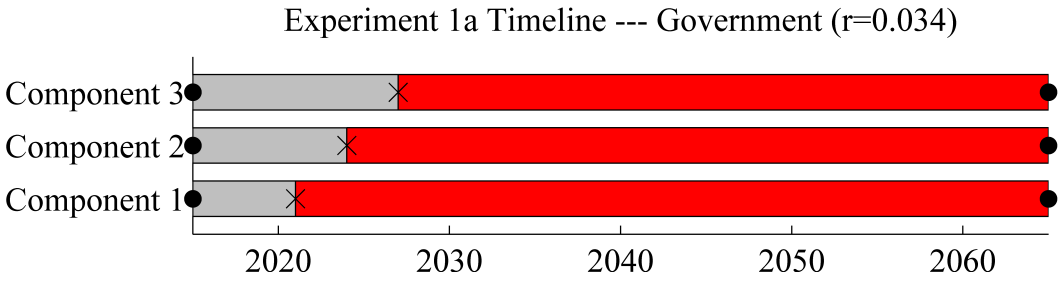

Figure 56: Optimized design refresh strategy for Experiment 1a, using parameters defined in Table 11, and assuming a government discount rate (r=0.034).

obsolescence mitigation plan features no redesigns, and obsolescence is encountered only three times, as illustrated in Figure 56. Since the private-sector plan experiences obsolescence more frequently than the government plan, forecast uncertainty has a greater impact for the private sector. Simply by encountering fewer individual obsolescence events, the government plan is less sensitive to uncertainty in obsolescence forecasts.

It is important to consider that the sensitivities depicted in Figures 53 and 55 are elements of a continuum. That is, the sensitivity of the output to each of the input parameters will vary continuously as the discount rate is varied. This is illustrated in Figures 57 and 58, which show the sensitivity of c_{obs} to variations in each parameter as a function of discount rate. Figure 57 depicts these variations as a line graph, which helps to illustrate when each parameter begins to dominate over others. For instance, Figure 57 shows that the sensitivity of c_{obs} to variations in forecast uncertainty (Δ_i) starts low when the discount rate is low, but gradually climes as the discount rate increases. When the discount rate increases to approximately r=0.058 the forecast uncertainty (Δ_i) begins to dominate over component cost (c_i^{LOT}). When the discount rate increases to approximately r=0.081 the forecast uncertainty (Δ_i) begins to dominate over annual failures (q_i). Finally, when the discount rate is larger than approximately r=0.13, variations in forecast uncertainty (Δ_i) drive more than 50% of the response. The proportional impact of each parameter is illustrated via an area plot in Figure 58.

5.1.1.2 Experiment 1a Conclusion

Experiment 1a tested Hypothesis 2.1a, which postulates that obsolescence forecast accuracy is a main-driver of obsolescence cost for COTS-based systems. An example system was tested under the assumptions of two different financial environments: Private-Sector (discount rate = 0.12) and government (discount rate = 0.034). It

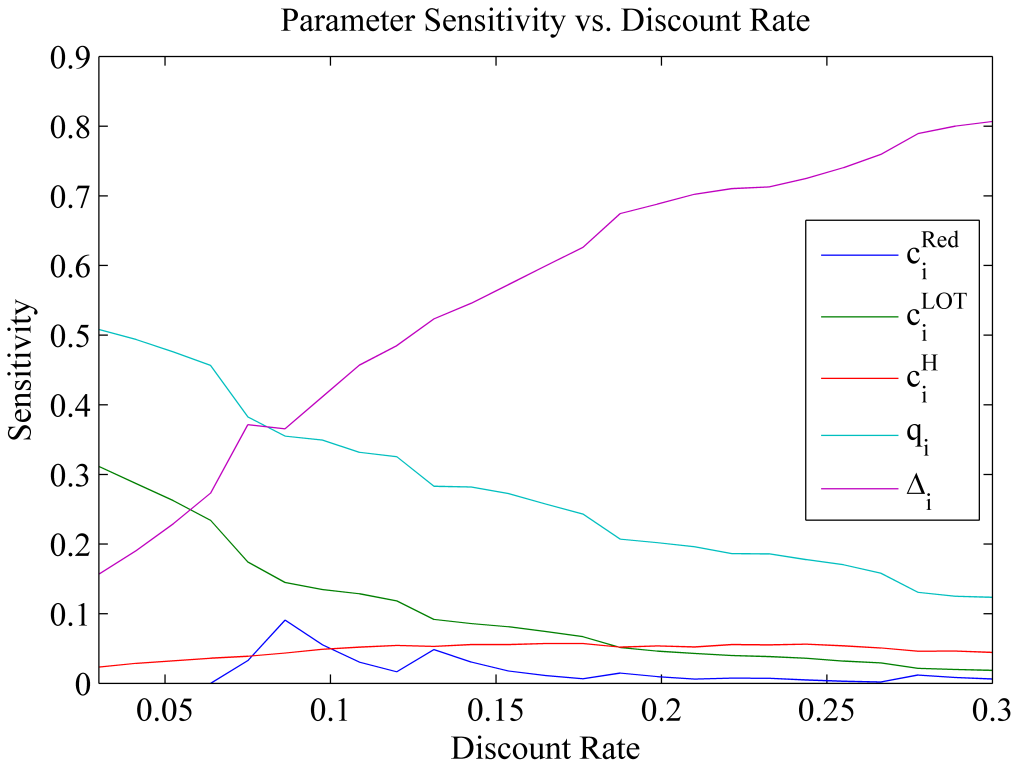

Figure 57: Line graph depicting the sensitivity of c_{obs} to each parameter as a function of discount rate. Note that at very low discount rates, c_{obs} is primarily driven by variability in annual failures (q_i), component cost (c_i^{LOT}), and obsolescence forecast (Δ_i). When the discount rate increases to approximately r=0.058 the obsolescence forecast (Δ_i) begins to dominate over component cost (c_i^{LOT}). When the discount rate increases to approximately r=0.081 the obsolescence forecast (Δ_i) begins to dominate over annual failures (q_i). Finally, when the discount rate is larger than approximately r=0.13, variations in the obsolescence forecast (Δ_i) drive more than 50% of the response.

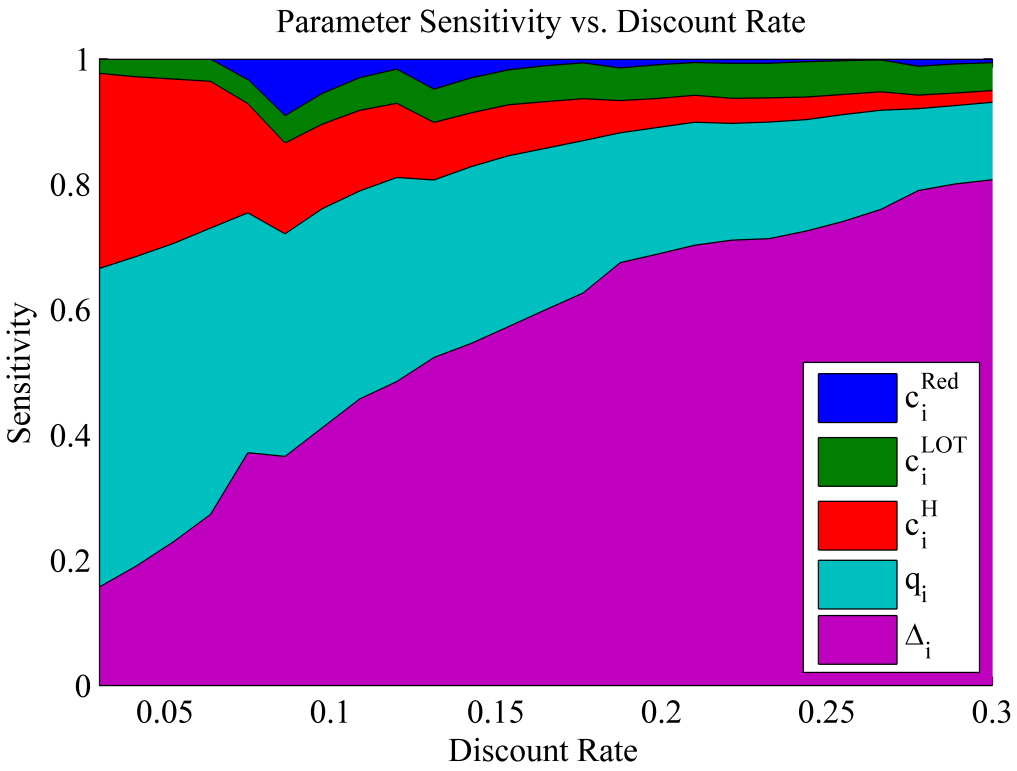

Figure 58: Area plot depicting the cumulative sensitivity of c_{obs} to each parameter as a function of discount rate. Note that at very low discount rates, c_{obs} is primarily driven by variability in annual failures (q_i), component cost (c_i^{LOT}), and obsolescence forecast (Δ_i). As the discount rate increases, the sensitivity of c_{obs} to variability in Δ_i begins to dominate. Also note that sensitivity to variations in annual holding cost (c_i^H) and redesign cost (c_i^{red}) are minimal throughout the entire domain.

was shown that the sensitivity of the optimal c_{obs} depends on which of these two environments the system is being designed for. Within a private-sector environment, the optimal design refresh strategy includes multiple LOT buys for each component, thus making the obsolescence forecast (Δ_i) a main-driver of obsolescence cost. Under a government financial environment, the optimal strategy shifts to a zero-redesign strategy akin to the Reactive Last-Time Buy baseline. Following this strategy, only one LOT buy is performed for each component, so the effect of obsolescence forecast uncertainty on obsolescence cost decreases, and is overshadowed by the number of annual failures (q_i) and component cost (c_i^{LOT}). In conclusion, Hypothesis 1a is supported when a high discount rate is imposed, such as a private-sector financial environment, but refuted when subjected to lower discount rates similar to a government financial environment.

5.1.2 Experiment 1b

Experiment 1a shows that the impact of forecast accuracy on the variability of obsolescence cost depends on the nature of the financial environment for which the system is being designed. Despite this, answering Research Question 2.1 still requires the testing of Hypothesis 2.1b, which postulates that obsolescence cost is non-increasing with increasing forecast accuracy. To test this, the following procedure is used:

Step 1: Define a Representative System — As in Experiment 1a, a representative system is needed, about which an optimal obsolescence mitigation plan is generated. For each component, cost and schedule parameters are defined, including annual failures (q_i), unit cost (c_i^{LOT}), redesign cost (c_i^{red}), and holding cost (c_i^H)

Step 2: Create "Forecast Data" — In lieu of forecasting obsolescence for actual components, it is assumed that forecast data already exists for each component. To represent these forecasted dates, a component life cycle, Δ_i, for each component is defined[3]. These Δ_i values represent the predicted dates of obsolescence for each component and are subject to uncertainty. This uncertainty is accounted for in Step 4, below.

Step 3: Build an Optimized Strategy — The parameters of the representative system from Steps 1 and 2 are entered into the CASCADE Mixed Integer Linear Programming formulation developed in Chapter 3. The result of this step is an optimized obsolescence mitigation plan that specifies which obsolescence mitigation actions should be taken, and when. These actions include when redesigns are to be performed, when Life-of-Type buys are to be performed

[3]Recall that Δ_i represents the duration of the component life, and *not* a specific date of obsolescence. Despite this, defining Δ_i for each component is functionally equivalent to specifying an obsolescence date since, by definition, obsolescence can always be predicted to occur Δ_i years after a component's introduction.

and their duration, and when the system incurs a system-level non-recurring engineering (NRE) cost.

Step 4: Define "Truth" Data — The cost and forecast parameters defined in Steps 1 and 2 represent predictions, but the actual values cannot be known with certainty *a priori*. To account for this uncertainty, it is assumed that the true cost and schedule parameters follow normal distributions, each with an independent mean centered about the forecasted obsolescence date, and an independent standard deviation.

Step 5: Compute the "True" Obsolescence Cost — Using the "truth" parameters from Step 4, and explicitly following the optimized strategy developed in Step 3, the true cost of obsolescence is computed. This value is presented as a relative uncertainty (i.e. a percent deviation from the expected value) so as to make the result system-agnostic.

Step 6: Repeat via Monte Carlo — Steps 4 and 5 are repeated using a Monte Carlo simulation to generate a histogram of output values.

Step 7: Repeat with Varying Forecast Accuracy — The output of Step 6 is a histogram of "true" obsolescence costs for each repetition of Steps 4 and 5. Steps 4 through 6 are then repeated, changing the forecast accuracy (standard deviation) each time. The result of these repetitions is a set of histograms representing varying forecast accuracy.

Since it is possible that the results will depend on the complexity of the system, the procedure described above is repeated for two different systems: A 1-component system, and a 3-component system. The parameters used for the 1 and 3-component systems are shown in Tables 15 and 16, respectively. Furthermore, since the results of Experiment 1a indicate vastly different strategies under a private-sector financial

Table 15: Table of component-level and system-level parameters for Experiment 1b, using a 1-component system.

	Component 1
Δ_i	6 yrs.
c_i^{LOT}	.0285
c_i^H	.007125
c_i^{red}	1
q_i	0.1

	System
c^{NRE}	.5
d_{IOC}	2015
Production Dates	$\{2015, 2020, 2025, 2030, 2035, 2040, 2045, 2050, 2055\}$
d_{EOL}	2065
d_{base}	2015

environment versus a government financial environment, the procedure above is repeated under both sets of conditions. Each case is repeated, varying the forecast uncertainty between 0.01% and 10% in Step 7. The output distributions of c_{obs} will be captured using a 10,000-run Monte Carlo for each setting of forecast uncertainty.

5.1.2.1 Experiment 1b Results and Analysis

Figure 60 depicts the variation in output uncertainty versus uncertainty in obsolescence forecasts for the 1-component system described in Table 15. There is a clear trend indicating a direct correlation between forecast uncertainty and uncertainty in c_{obs}. This means that there is both an increase in potential cost overruns, and an increase in potential lost opportunity cost as forecast uncertainty grows. Since cost overruns may lead to the inability to adequately support the system throughout its life cycle, decision makers would need to increase the budget as a buffer.

Similarly, Figure 61 shows the variation in output uncertainty versus uncertainty in obsolescence forecasts for the 3-component system described in Table 16. The

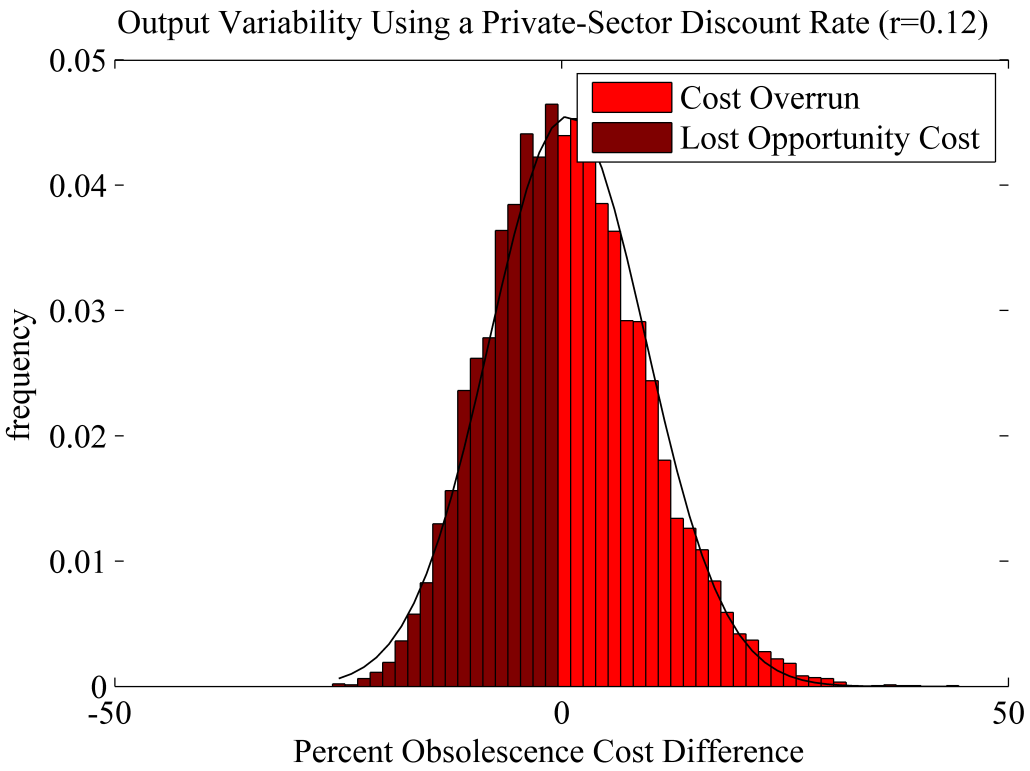

Figure 59: Histogram of obsolescence cost given an uncertain obsolescence forecast. Note that two negative outcomes are possible when the true obsolescence cost deviates from the predicted cost: When $c_{obs}^{actual} > c_{obs}^{predicted}$, a cost overrun occurs. This may impact the ability to adequately support the system through its full life cycle. A case where $c_{obs}^{actual} < c_{obs}^{predicted}$ represents a lost opportunity cost. This means that financial resources were allocated where they were not used.

Table 16: Table of component-level and system-level parameters for Experiment 1b, using a 3-component system.

	Component 1	Component 2	Component 3
Δ_i	6 yrs.	9 yrs.	12 yrs.
c_i^{LOT}	.0285	.021	.0255
c_i^{H}	.007125	.00525	.006375
c_i^{red}	1	1	1
q_i	0.1	0.1	0.1

	System
c^{NRE}	.5
d_{IOC}	2015
Production Dates	$\{2015, 2020, 2025, 2030, 2035, 2040, 2045, 2050, 2055\}$
d_{EOL}	2065
d_{base}	2015

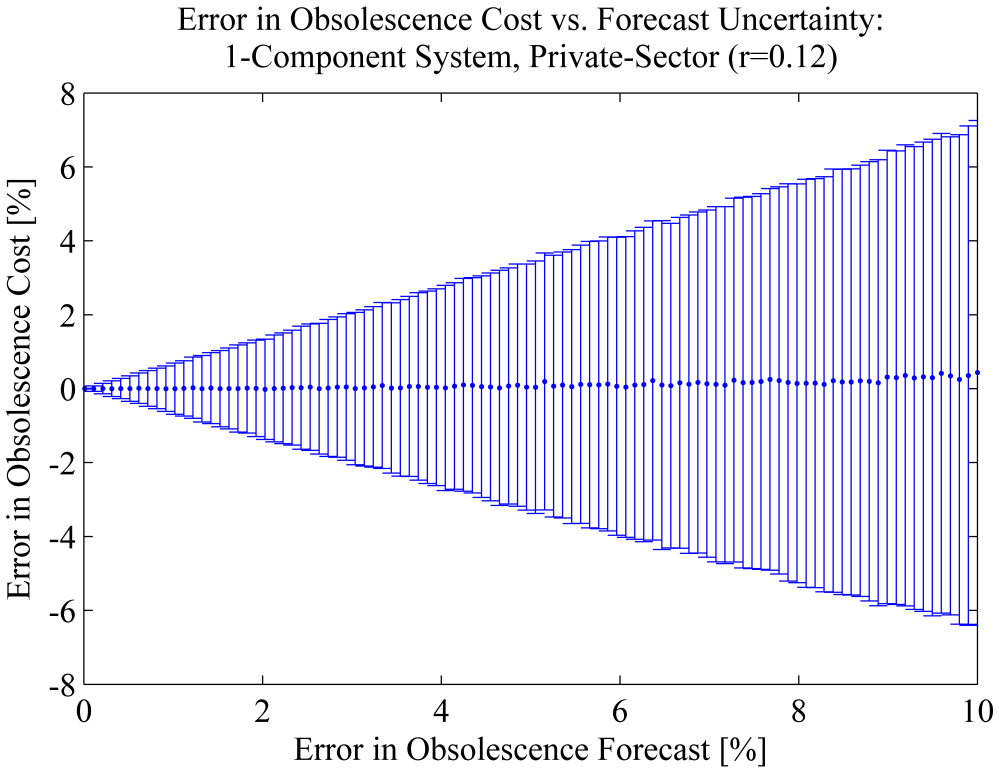

Figure 60: Uncertainty in c_{obs} vs obsolescence forecast uncertainty for a 1-component system using a private-sector discount rate (r=0.12). Uncertainties in obsolescence forecasts were varied between 0.01% to 10%. Error bars are generated using a 10,000-run Monte Carlo, and represent a 1σ percent deviation from the mean.

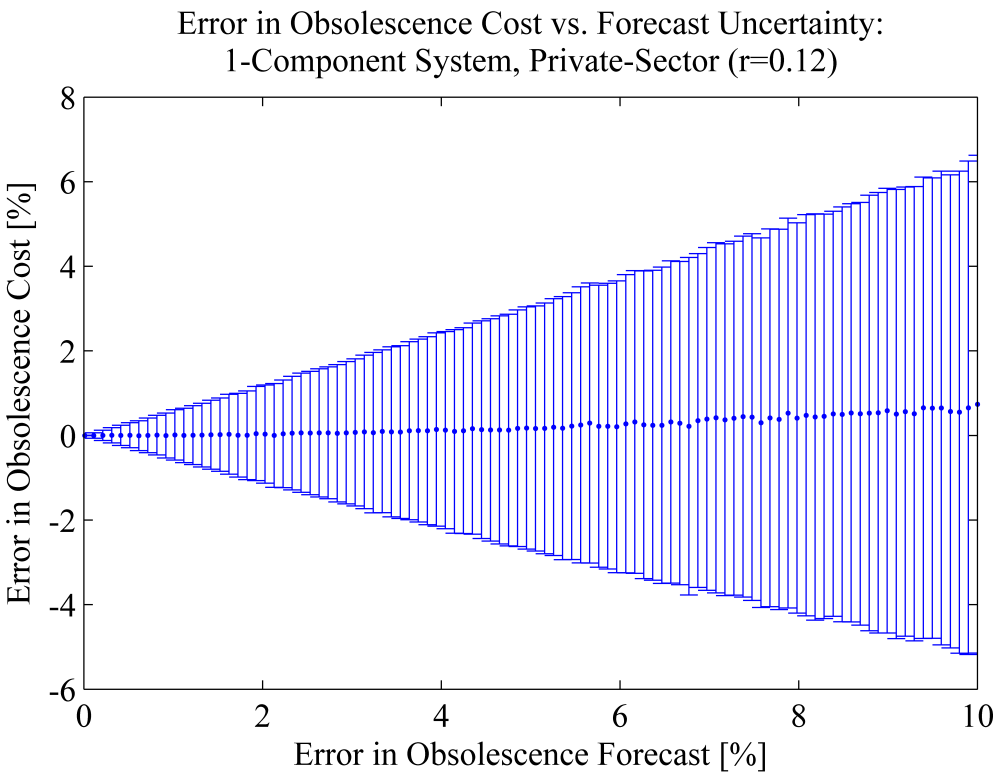

Figure 61: Uncertainty in c_{obs} vs obsolescence forecast uncertainty for a 3-component system using a private-sector discount rate (r=0.12). Uncertainties in obsolescence forecasts were varied between 0.01% to 10%. Error bars are generated using a 10,000-run Monte Carlo, and represent a 1σ percent deviation from the mean. Note that the results are virtually indistinguishable from those of the 1-component system depicted in Figure 60, suggesting that system complexity has little bearing on the effects of obsolescence forecast uncertainty.

results for the 3-component system are virtually identical to those of the 1-component system, suggesting that system complexity plays little role in the impact of forecast uncertainty.

Figure 62 shows the impact of varying forecast uncertainty for a 3-component system using a private-sector discount rate (r=0.12) and a government discount rate (r=0.034). The general trend applies using either discount rate: uncertainty in obsolescence cost (c_{obs}) is directly correlated with uncertainty in obsolescence forecasts. Note, however, that the magnitude of the error in obsolescence cost under the government discount rate is significantly lower than that of the private-sector. In fact,

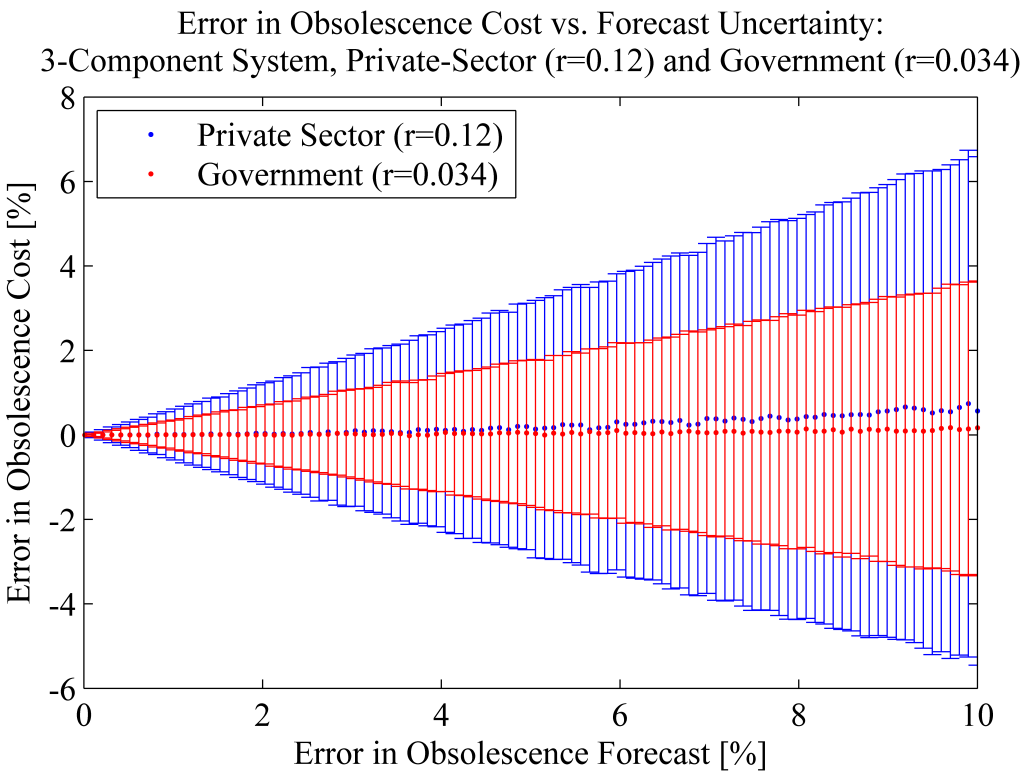

Figure 62: Uncertainty in c_{obs} vs obsolescence forecast uncertainty for a 3-component system using a private-sector discount rate (r=0.12) and government discount rate (r=0.034). Uncertainties in obsolescence forecasts were varied between 0.01% to 10%. Error bars are generated using a 10,000-run Monte Carlo, and represent a 1σ percent deviation from the mean. Note that the error using the private-sector rate is consistently over 90% greater than that of the government rate, regardless of forecast error. This result is consistent with the results of Experiment 1a.

uncertainty in obsolescence cost in the private sector is approximately 93% higher than for the government, regardless of forecast uncertainty. This may be due to the fact that forecast uncertainty is a main-driver under in a private-sector environment, but not in a government environment, as Experiment 1a showed. Alternatively, this may be a function of the expected cost of obsolescence, which is significantly higher for the government than for the private sector ($354,392 versus $135,675).

As mentioned above, for safety-critical long-life-cycle systems, the inability to support the system should be avoided at all costs, so cost overruns are unacceptable. To avoid these cost overruns, decision makers should budget an additional "buffer"

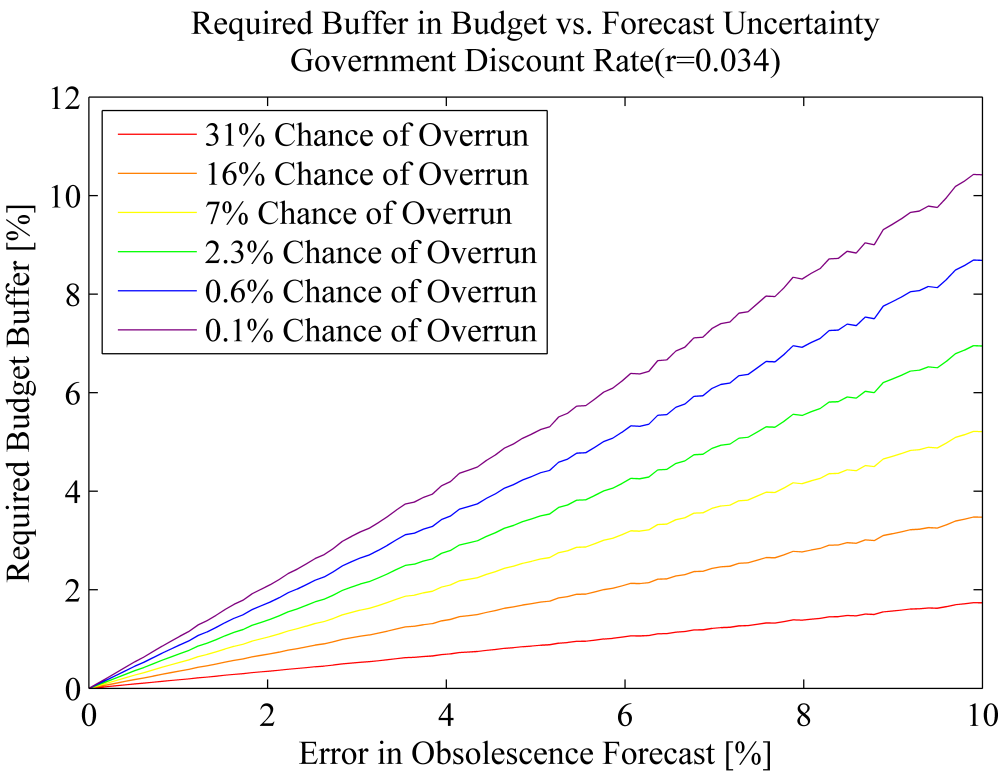

Figure 63: Required buffer (given as a percent of expected obsolescence cost) in the budget to protect against cost overruns. Note that the magnitude of this buffer is a function of the obsolescence forecast accuracy, as well as the desired confidence level.

beyond the expected obsolescence cost. The size of this buffer is dependent on the accuracy of the obsolescence cost prediction (which is, in turn, dependent on the accuracy of the obsolescence forecasts), as well as the confidence with which the decision maker wishes to avoid a cost overrun. Figure 63 shows the obsolescence cost buffer required as a function of both forecast accuracy and desired confidence level. The magnitude of this buffer was calculated using a right-tailed test on the output distributions of the Monte Carlo. As Figure 63 shows, the required buffer increases monotonically with increasing obsolescence forecast error. Thus, obsolescence cost is shown to be non-increasing with increasing forecast accuracy, and so Hypothesis 2.1b is supported.

5.1.2.2 Experiment 1b Conclusion

Experiment 1b tested Hypothesis 2.1b, which postulates that obsolescence cost is non-decreasing with increasing forecast uncertainty. A 1-component system and a 3-component system were tested, and it was shown that the variability of the resulting obsolescence mitigation cost was virtually identical for both systems. This suggests that system complexity has little bearing on the impact of forecast accuracy on cost. Next, the 3-component system was tested using a government discount rate (r=0.034) and private-sector discount rate (r=0.12). Both environments exhibited a direct correlation between uncertainty in obsolescence cost (c_{obs}) and forecast uncertainty. That is, as forecast uncertainty grows, uncertainty in obsolescence cost grows, regardless of financial environment.

Two scenarios were considered for variations in cost: actual cost may be *higher* than predicted cost (cost overrun), or actual cost may be *lower* than predicted cost (lost opportunity cost). Both scenarios were shown to be equally likely, but the literature review indicates that the consequences of cost overruns far outweigh the consequences of lost opportunity cost. To account for this, decision makers should budget a "buffer" into the expected cost of obsolescence. Since this buffer grows as forecast uncertainty grows, the cost of obsolescence is non-decreasing with increasing forecast uncertainty. Thus, Hypothesis 2.1b is supported.

5.1.3 Experiment 1 Conclusion

Experiment 1 aimed at answering Research Question 2.1, repeated below:

> **Research Question: 2.1**
> What is a suitable method of forecasting obsolescence?

Several different options exist, but there is a tradeoff between obsolescence forecast accuracy and the amount of data required. Ultimately it was argued that the "most

suitable" method depends on the relationship between forecast accuracy and the expected cost of obsolescence. If the expected cost of obsolescence is highly sensitive to forecast uncertainty, then the most-accurate method should be selected. However, if obsolescence cost decreases or remains constant with increasing obsolescence forecast uncertainty, then a highly-accurate forecast may be unnecessary. These two factors were addressed using two hypotheses, repeated below:

Hypothesis: 2.1a
The accuracy of obsolescence forecasts will be a main driver of obsolescence cost for COTS-based systems.

Hypothesis: 2.1b
The obsolescence cost will be non-increasing with increasing accuracy of obsolescence forecasts.

Experiment 1a tested Hypothesis 2.1a using a 10,000-run Monte Carlo, which varied several cost and schedule parameters. This result supported Hypothesis 2.1a for high discount rates such as in the private sector, but failed to support it for lower discount rates such as in a government environment. This suggests that highly-accurate forecasting methods, such as Data Mining, may be the best suited for high discount-rate environments, but less-precise methods may be acceptable in the case of lower discount rates. In fact, the example problem used in Experiment 1a showed that a zero-redesign strategy was optimal when operating at a low discount rate. In this particular instance, the simple-but-accurate Manufacture Inquiry method would suffice, since only one forecasted obsolescence date is needed for each component.

Experiment 1b tested Hypothesis 2.1b using several 10,000-run Monte Carlo simulations to observe the variability of c_{obs} under various forecast uncertainties. The impact of forecast uncertainty was shown to be more pronounced for the private sector

than for the government, although both environments resulted in increased obsolescence cost with increasing forecast uncertainty. Thus, Hypothesis 2.1b is supported.

The exact trends observed in Experiment 1a and 1b will vary depending on the nature of the system being observed, but recommendations can still be made based on their results. Systems in both a government and a private-sector environment will benefit from increased forecast accuracy, but the effects of forecast error are only a main-driver in the private sector. Thus, Research Question 2.1 can be conditionally answered as follows: *The most-accurate obsolescence forecasting method available (e.g. Data Mining) is strongly advised when the discount rate is high, but a simpler obsolescence forecasting method (e.g. Sales Curve Forecasting) may be substituted when the discount rate is low.*

5.2 Experiment 2: Testing COTS Modification

The CASCADE framework includes the option to modify COTS components as a means to increase COTS utilization, as depicted in Figure 64. A literature review reveals conflicting accounts of the costs or benefits of modifying COTS components, suggesting a lack of understanding about the underlying tradeoffs. Ignoring anecdotal evidence, virtually no published data exists which quantifies the benefits or drawbacks of COTS component modification for COTS-based systems. Research question 1.2 aims to address this gap, and is repeated below:

> **Research Question: 1.2**
> What trends emerge between reliability and cost when switching from low-level to high-level COTS?

The goal of Research Question 1.2 is to uncover the system-level obsolescence cost trends that emerge when modification is applied at the component level. However, since COTS component modification can take many forms, Experiment 2 primarily focuses on a sub-set of COTS modification known as *ruggedization*. Through ruggedization, COTS components can become more survivable when applied in harsh military environments, and can therefore be expected to fail less frequently. Thus, critical to Experiment 2 is the component-level relationship between reliability and cost. That is, it is necessary to understand the cost implications of increasing reliability.

Very little published data exists that empirically addresses component-level cost-vs-reliability relationships for military components. Despite this, component cost can be expected to follow a few basic trends. Aggarwal and Gupta define four requirements for any cost-vs-reliability relationship as follows[13, 14]:

1. Cost is a monotonically increasing function of component reliability

2. The cost of a high reliability component is very high

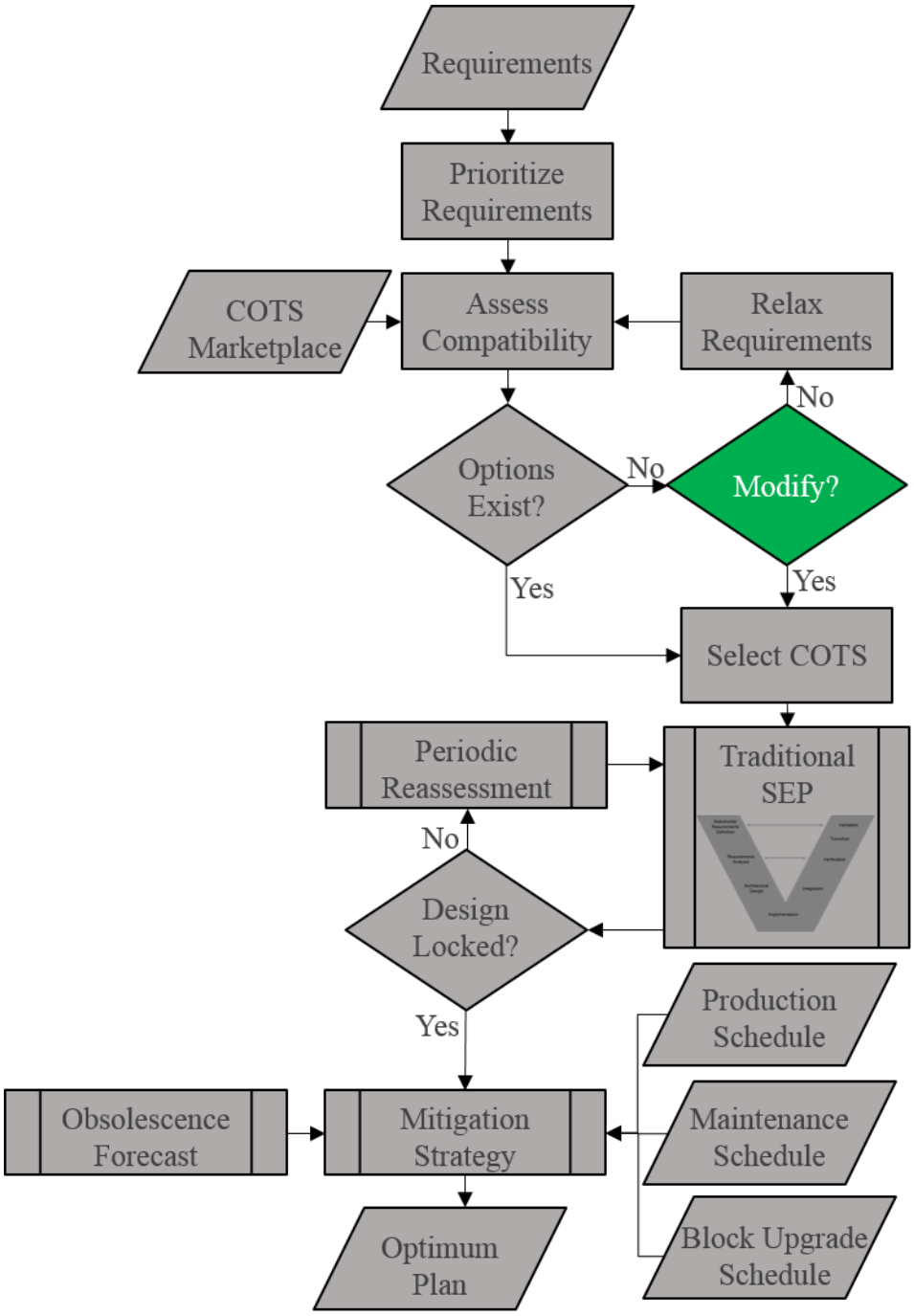

Figure 64: Experiment 2 applies the CASCADE MILP developed in Chapter 3 to uncover the underlying tradeoffs between COTS component modification.

3. The cost of a low reliability component is very low

4. The derivative of cost with respect to reliability is a monotonically increasing function of reliability

Appendix F expands on these requirements, and highlights specific cost-vs-reliability relationships found in the literature.

Furthermore, it is not clear whether component modification will impact the fixed cost, the variable cost, or some combination of the two. In other words, it is unknown whether modifying COTS components will impact the one-time redesign cost, or whether it will increase the cost of each modified component. For instance, most of the systems in a RAND study saw no increase in unit cost after a reliability improvement program, despite a sizable increase in R&D costs[15]. In contrast, market research indicates that the unit cost for ruggedized tablets and laptops is considerably higher than their non-ruggedized counterparts[50, 60, 72]. These discrepancies suggest that the nature of the cost impact of ruggedization depends on *who* is doing the ruggedizing: If the military purchases COTS components and modifies/ruggedizes them then the unit cost remains relatively stagnant despite an increase in redesign cost. If, however, the military purchases components that have been modified/ruggedized by the manufacturer, then the redesign cost remains relatively stagnant despite an increased unit cost.

To address both possibilities, Experiment 2 is broken into two sub-experiments: First, Experiment 2a (§ 5.2.1) explores the obsolescence cost trends assuming ruggedization impacts the cost of redesigning components. Next, Experiment 2b (§ 5.2.2) explores the obsolescence cost trends assuming ruggedization impacts the per-unit cost of components. Each sub-experiment compares the results of the CASCADE MILP formulation in two ways: First, obsolescence cost-vs-reliability trends are explored for both a government and a private-sector financial environment. Next, the

results of the CASCADE MILP formulation are compared against the results of several baseline obsolescence mitigation strategies, including the Reactive Last-Time Buy, Reactive Bridge-Buy, and the Original MILP formulation.

5.2.1 Experiment 2a — Ruggedization vs Redesign Cost

Experiment 2a tests the system-level obsolescence cost impact of component ruggedization assuming ruggedization impacts both component reliability (q_i) and component redesign cost (c_i^{red}). To do this, the CASCADE MILP formulation developed in Chapter 3 is applied to a representative system while the parameters of that system are varied. The procedure for Experiment 2a is as follows:

Step 1: Define a Representative System — Firstly, a representative system is needed, about which an obsolescence mitigation strategy is defined. Also during this step, cost and schedule parameters will be defined, including system-level NRE cost (c^{NRE}), component life cycles (Δ_i), discount rate (r), and holding cost (c_H). Component cost and reliability parameters are not defined in this step, since these values are varied during experimentation.

Step 2: Compute Obsolescence Cost Under Varying c_i^{red} and q_i — Next, for each combination of redesign cost (c_i^{red}) and annual failures (q_i), an optimum obsolescence mitigation plan is generated using the CASCADE MILP formulation developed in Chapter 3.

Step 3: Visualize Cost Impact — By plotting the obsolescence cost (c_{obs}) as a function of component redesign cost (c_i^{red}) and annual failures (q_i), the impact of ruggedization on obsolescence cost is determined[4].

[4]If specific component-level cost-vs-reliability relationships are known, these can be superimposed on the surface to determine exact trends. Several such trends are explored in Appendix F.

Table 17: Component-level and system-level parameters for Experiment 2a.

	Component 1
Δ_i	6 yrs.
c_i^{LOT}	.0285
c_i^{H}	.007125
c_i^{red}	1–10
q_i	.025–0.1

	System
c^{NRE}	.5
d_{IOC}	2015
Production Dates	$\{2015, 2020, 2025, 2030, 2035, 2040, 2045, 2050, 2055\}$
d_{EOL}	2065
d_{base}	2015

To reduce the dimensionality of the problem, a simple 1-component system described in Table 17 is used in this analysis[5]. To protect proprietary information, the cost and reliability inputs have been normalized and are presented as unitless, and a generic component name has been used. The results of Experiment 2a are split into two sections: Section 5.2.1.1 compares the CASCADE MILP result under a private-sector discount rate and a government discount rate. Second, Section 5.2.1.2 compares the CASCADE MILP result against three baseline strategies.

5.2.1.1 Experiment 2a Results: Private vs Government

Figure 65 depicts the obsolescence cost trends associated with ruggedization assuming a private-sector discount rate (r=0.12). The results indicate that obsolescence cost (c_{obs}) varies linearly with changing annual failures (q_i) and component redesign cost (c_i^{red}) throughout the domain, except when redesign cost is low and annual failures are high. That is because the nature of the optimum obsolescence mitigation plan

[5]Note that using a 1-component system has the added benefit of "magnifying" the cost trends associated with ruggedization, making them easier to distinguish.

changes throughout the domain. As the annual failures (q_i) increase and component redesign cost (c_i^{red}) decreases, the optimum strategy switches from zero redesigns to one redesign, as illustrated by Figure 66. The same behavior is not exhibited when the discount rate is lowered, however.

At the lower government discount rate (r=0.034), the optimum obsolescence mitigation plan requires zero redesigns throughout the entire domain. As a result, the obsolescence cost (c_{obs}) varies linearly with changing annual failures (q_i) and component redesign cost (c_i^{red}) throughout the *entire* domain, as illustrated in Figure 67. Note that this planar behavior matches the Last-Time Buy baseline strategy depicted in Figure 69.

The trends illustrated in Figures 65 through 67 lend insight into the basic trade-offs associated with component ruggedization. As components are ruggedized the redesign cost (c_i^{red}) can be expected to increase while the number of annual failures (q_i) can be expected to decrease[6], regardless of what the specific component-level cost-vs-reliability trend is. At the government discount rate, Figure 67 indicates that the benefits of fewer annual failures outweigh the increase in redesign cost, so obsolescence cost decreases monotonically as ruggedization increases[7] The same basic trend is observed under the private-sector discount rate throughout most of the domain, except when annual failures are high and redesign cost is low, as indicated by Figure 65.

[6]This assumes that redesign cost (and not the component cost) is being affected by ruggedization.

[7]Note that this trend can be expected under the government discount rate, since the optimum obsolescence mitigation plan calls for zero redesigns throughout the entire domain. Hence, no penalty is observed when the cost of redesigning increases.

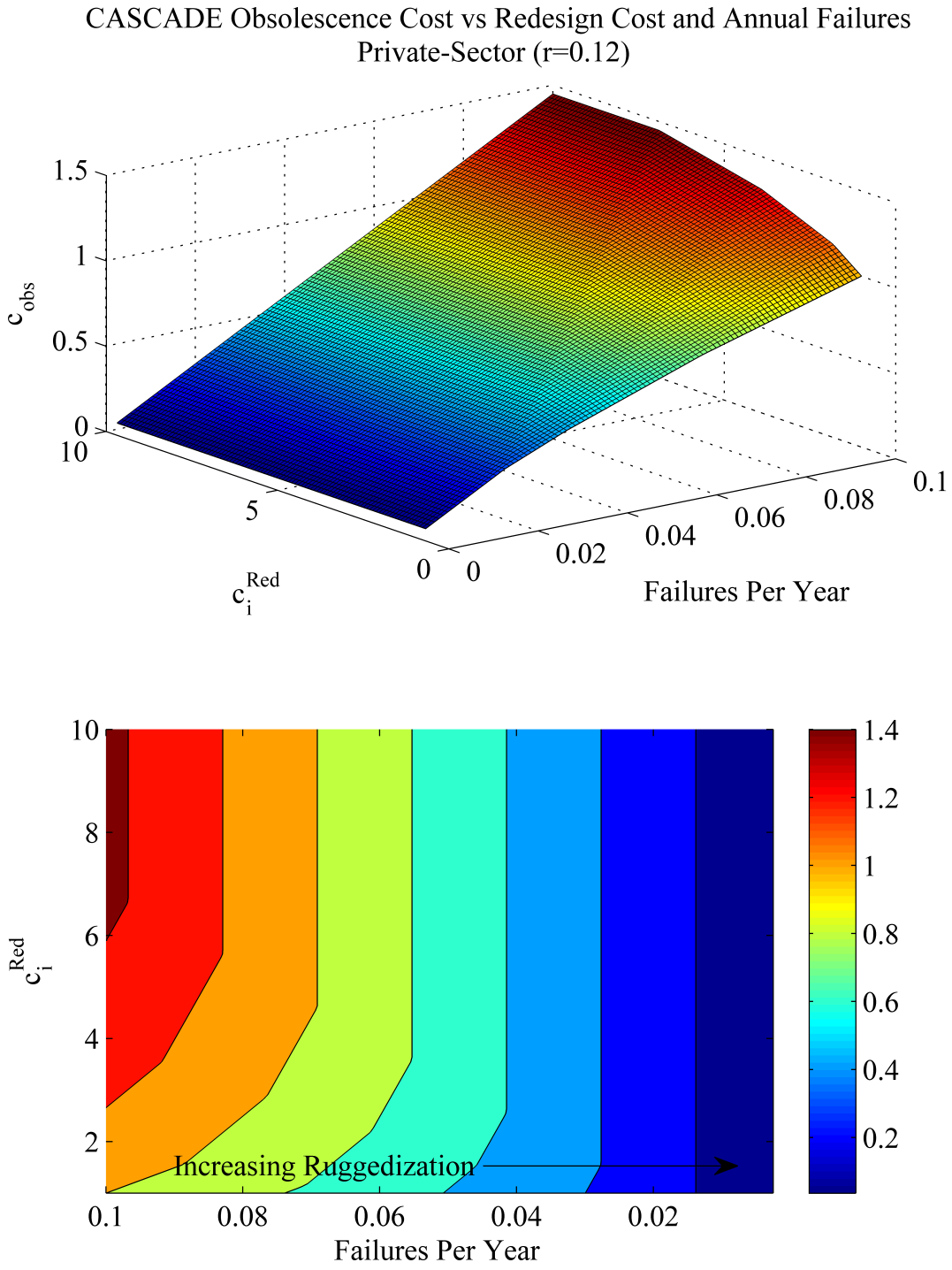

Figure 65: Optimum obsolescence cost (c_{obs}) as a function of component redesign cost (c_i^{red}) and annual failures (q_i) assuming a private-sector discount rate (r=0.12). Note that the obsolescence cost has been normalized such that the obsolescence cost using un-ruggedized components equals 1.

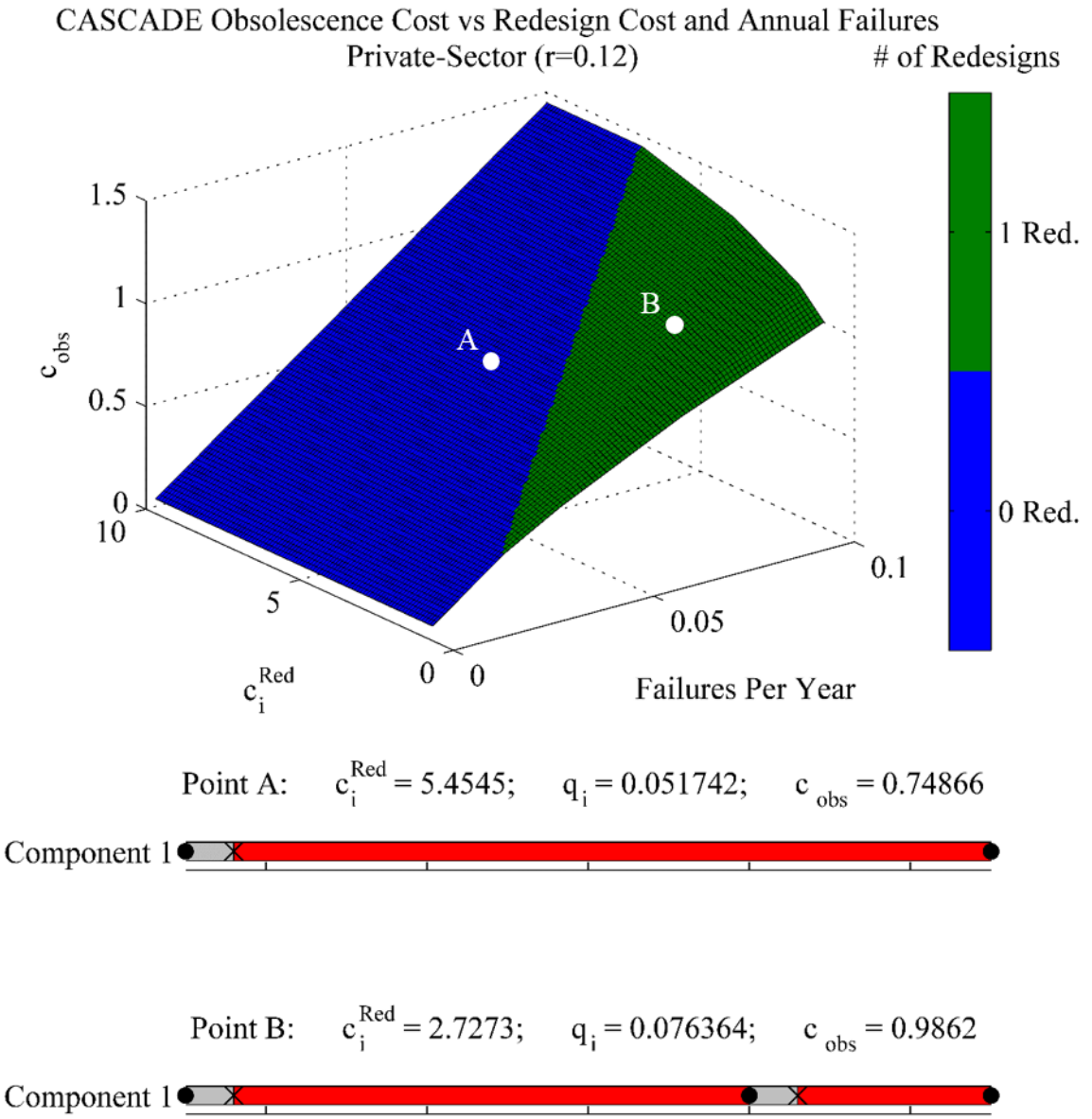

Figure 66: The nature of the optimum obsolescence mitigation plan changes throughout the domain. As the annual failures (q_i) increase and component redesign cost (c_i^{red}) decreases, the optimum plan switches from zero redesigns to one redesign, as indicated by the coloration in the surface plot. When this occurs, the obsolescence cost (c_{obs}) on the z axis varies to reflect this change. Note that the obsolescence cost has been normalized such that the obsolescence cost using un-ruggedized components equals 1.

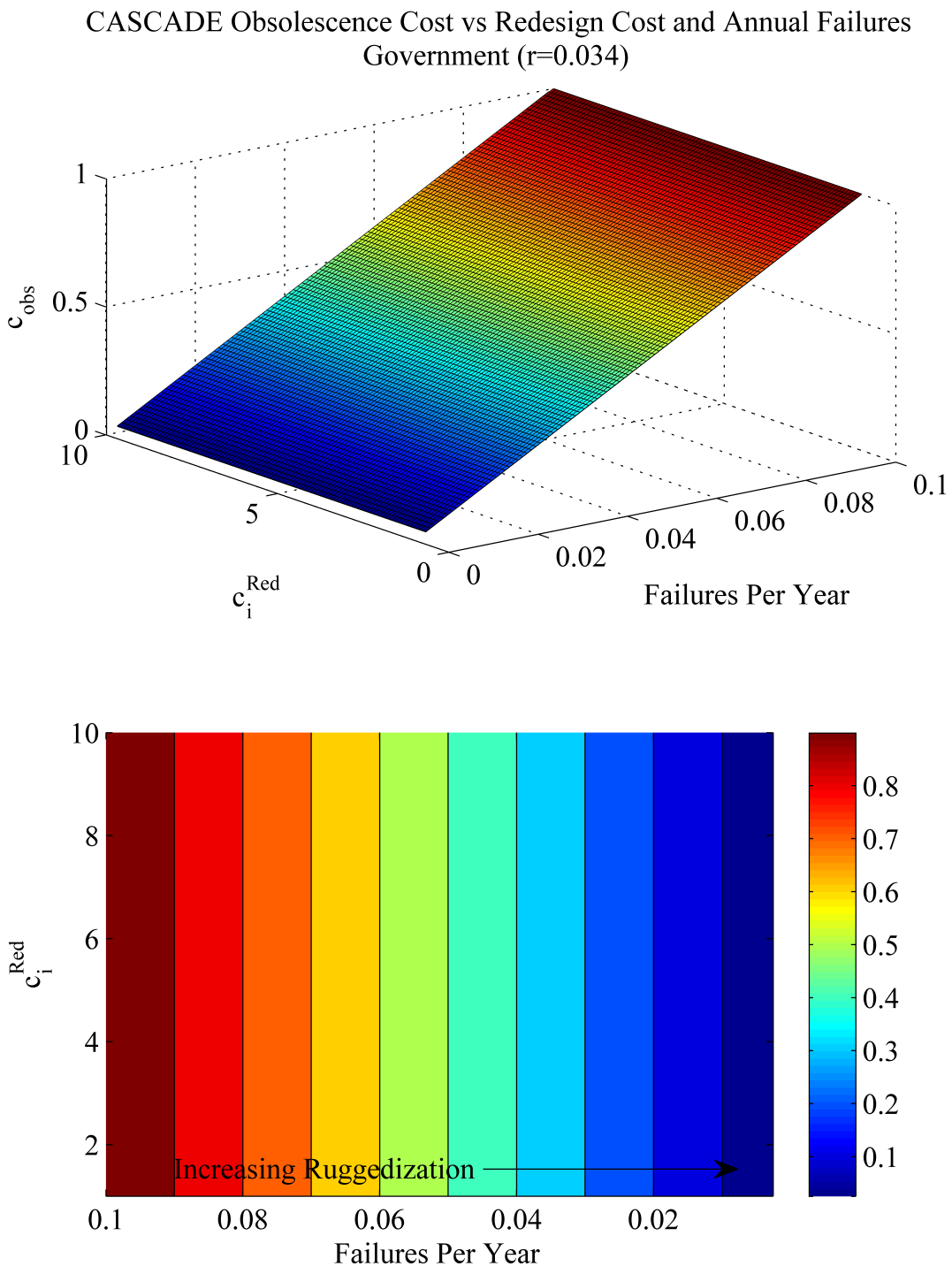

Figure 67: Optimum obsolescence cost (c_{obs}) as a function of component redesign cost (c_i^{red})and annual failures (q_i) assuming a government discount rate (r=0.034). Note that the trend matches that of the Reactive Last-Time Buy baseline depicted in Figure 69. Also note that the obsolescence cost has been normalized such that the obsolescence cost using un-ruggedized components equals 1.

5.2.1.2 Experiment 2a Results: Comparison Against Baselines

Figure 68 illustrates the obsolescence cost (c_{obs}) trend under varying annual failures (q_i) and component redesign cost (c_i^{red}) using the Reactive Bridge-Buy baseline strategy described in § 4.2.1. Using this strategy, each time obsolescence is encountered it is first mitigated using a short-term bridging action that lasts until the next-possible redesign date, at which point the obsolete component is replaced via a redesign. As Figure 68 shows, obsolescence cost varies linearly with both annual failures and redesign cost. Furthermore, there is a large obsolescence cost penalty as redesign cost increases, and a negligible obsolescence cost impact as annual failures decrease. Thus, the penalty for increased redesign cost outweighs the benefit of decreased annual failures using the Reactive Bridge-Buy baseline strategy. This would lead decision makers to believe that ruggedization should be avoided to prevent an increase in obsolescence cost. This is the opposite conclusion drawn from the CASCADE MILP shown in Figures 65 and 67.

Figure 69 depicts the obsolescence cost (c_{obs}) trend under varying annual failures (q_i) and component redesign cost (c_i^{red}) using the Reactive Last-Time Buy baseline strategy described in § 4.2.1. Using this strategy, each instance of obsolescence is mitigated using a single long-term bridging action that lasts until the end of the system's life. As Figure 69 shows, obsolescence cost varies linearly with both annual failures and redesign cost. Furthermore, there is a sizable obsolescence cost benefit observed as annual failures decrease, and there is no obsolescence cost impact when redesign cost increases. This would lead decision makers to believe that increasing ruggedization results in a decrease of obsolescence cost. This is the same conclusion drawn using the CASCADE MILP, except that the CASCADE MILP affords greater obsolescence cost savings when annual failures are high and redesign cost is low, as illustrated by Figure 65.

Finally, Figure 70 depicts the obsolescence cost (c_{obs}) trend under varying annual

failures (q_i) and component redesign cost (c_i^{red}) using the Original MILP formulation as a baseline. The results indicate that the Original MILP produces an obsolescence mitigation plan identical to the Reactive Bridge-Buy baseline. This suggests that the parameters of the system described in Table 17 have driven the Original MILP to the limits of its feasible space. In this case, that limit coincides with the Reactive Bridge-Buy baseline. This same behavior was captured earlier in § 4.2.3. Thus, as with the Reactive Bridge-Buy baseline, the Original MILP formulation would lead decision makers to believe that ruggedization should be avoided to prevent an increase in obsolescence cost. This is the opposite conclusion drawn from the CASCADE MILP shown in Figures 65 and 67.

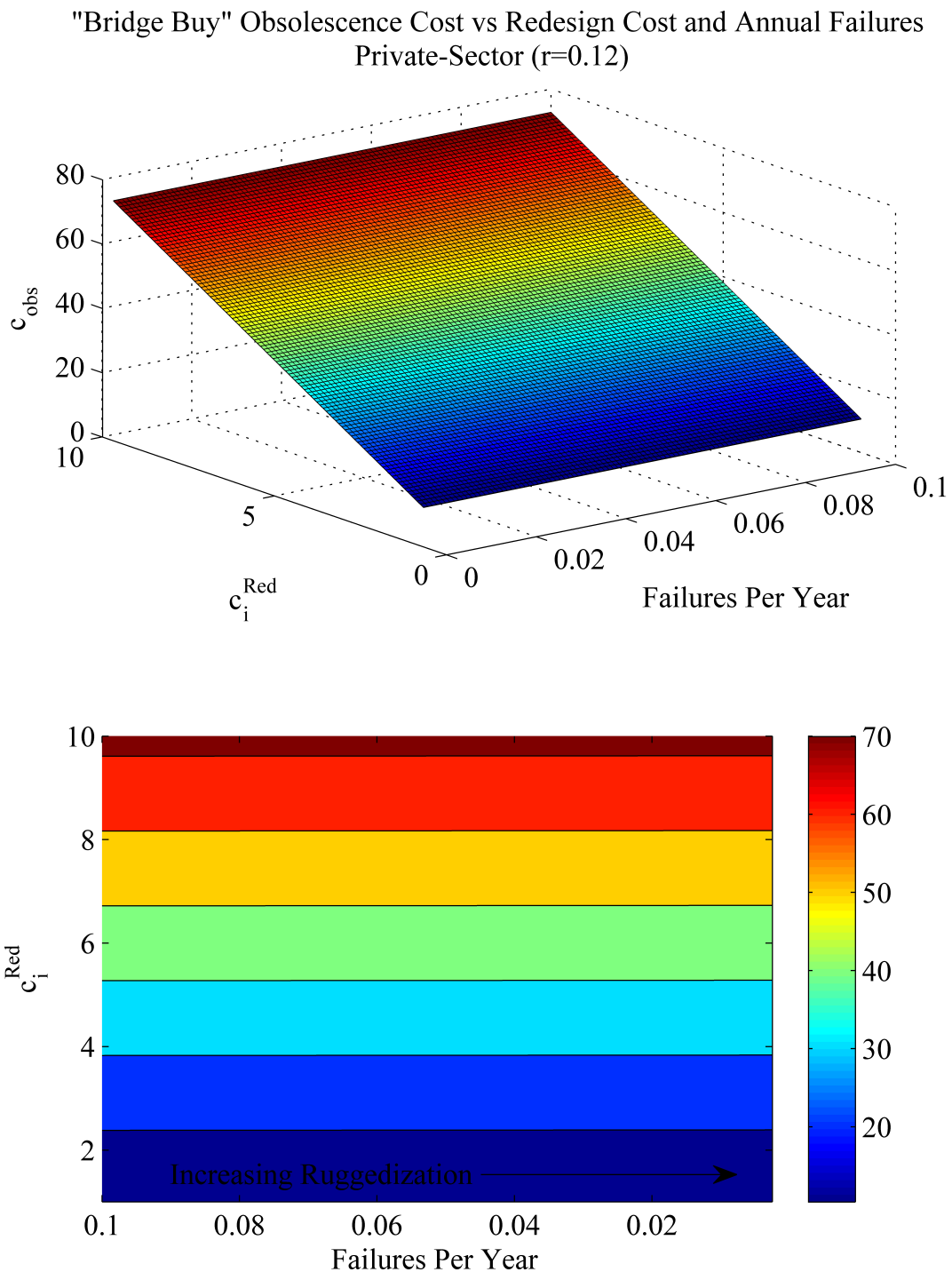

Figure 68: Obsolescence cost (c_{obs}) as a function of component redesign cost (c_i^{red}) and annual failures (q_i) using the Reactive Bridge-Buy baseline strategy, and assuming a private-sector discount rate (r=0.12). Note that this strategy results in an obsolescence cost that is several orders of magnitude larger than the CASCADE strategy depicted in Figure 65, indicating that the Bridge-Buy baseline is highly sub-optimal within this domain.

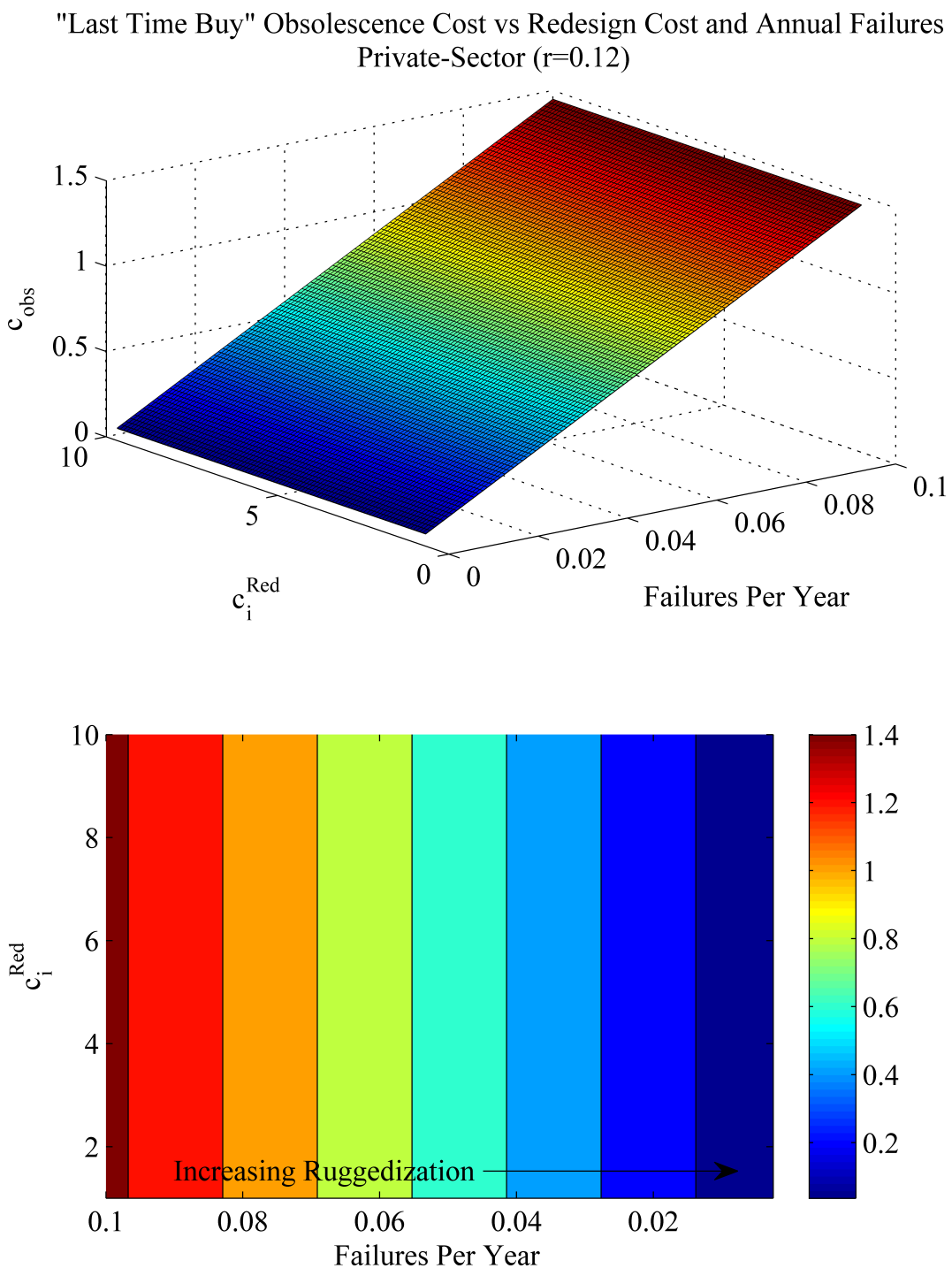

Figure 69: Obsolescence cost (c_{obs}) as a function of component redesign cost (c_i^{red}) and annual failures (q_i) using the Reactive Last-Time Buy baseline, and assuming a private-sector discount rate (r=0.12). Note that the result is nearly identical to the CASCADE result depicted in Figure 65, except when c_i^{Red} is low and q_i is high.

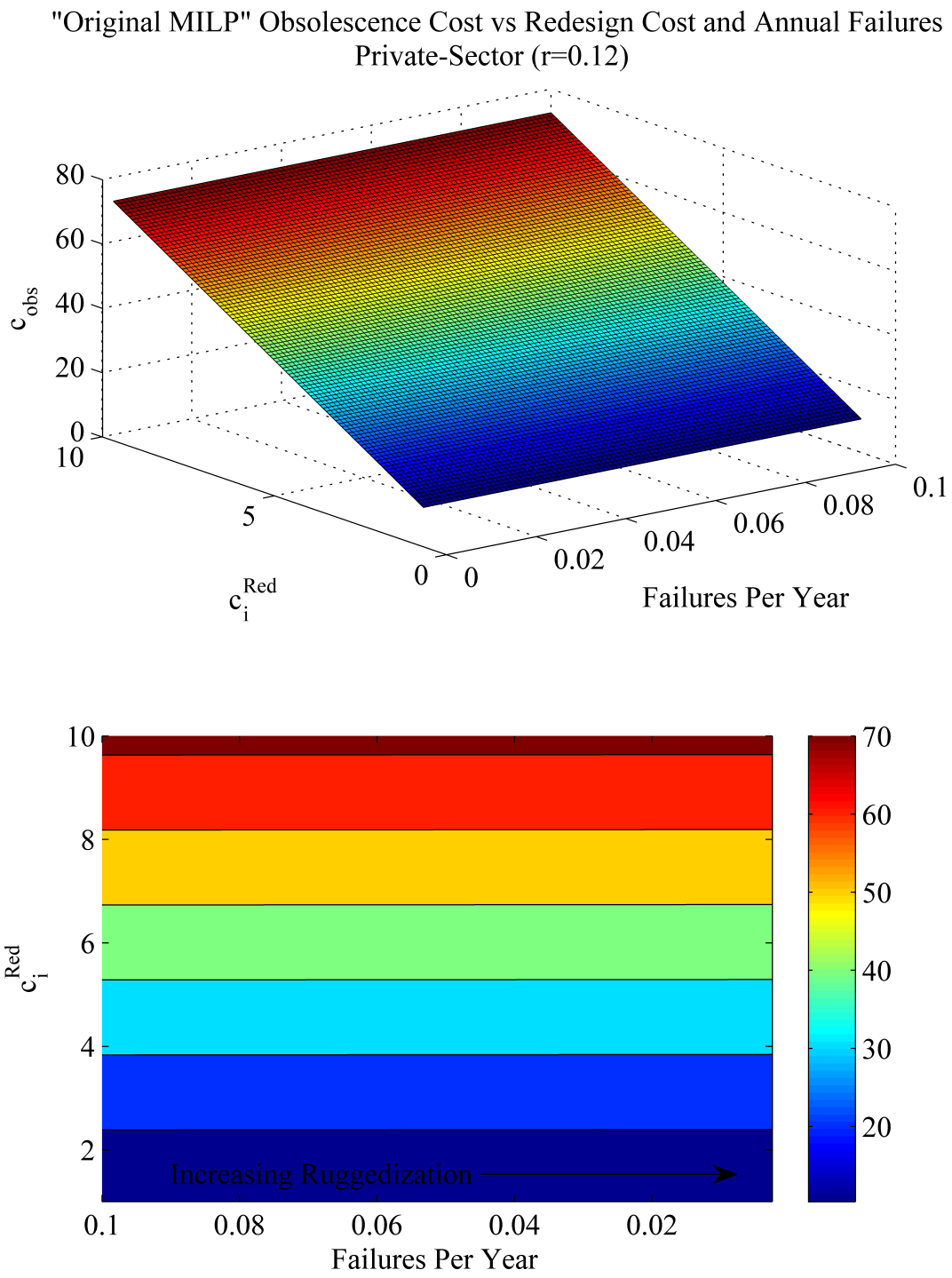

Figure 70: Obsolescence cost (c_{obs}) as a function of component redesign cost (c_i^{red}) and annual failures (q_i) using the Original MILP formulation, and assuming a private-sector discount rate (r=0.12). Note that this strategy results in an obsolescence cost that is several orders of magnitude larger than the CASCADE strategy depicted in Figure 65, indicating that the Bridge-Buy baseline is highly sub-optimal within this domain.

5.2.1.3 Experiment 2a Conclusion

Experiment 2a investigates the impact of component modification on system obsolescence cost assuming component cost remains constant, but redesign cost does not. An optimized design refresh plan was created using the CASCADE MILP formulation developed in Chapter 3, assuming a private-sector discount rate ($r = 0.12$) as well as a government discount rate ($r = 0.034$). The results show that under both financial environments, the trends between obsolescence cost (c_{obs}), component redesign cost (c_i^{red}), and annual failures (q_i) look nearly identical. Component modifications which improve reliability almost always improve obsolescence cost, regardless of the impact on the cost of redesigning. The only exception to this rule is for low-reliability and low-redesign-cost components assuming a private-sector discount rate.

Furthermore, it was shown that the CASCADE MILP formulation out-performs both the Reactive Bridge-Buy Baseline and the Original MILP baseline throughout the entirety of the domain used. Furthermore, the CASCADE MILP formulation out-performs the Reactive Last-Time Buy baseline when annual failures (q_i) are high and component redesign cost (c_i^{red}) is low. It is important to consider, however, that the relationships observed herein are specific to the parameters used in this experiment. Other systems may result in different trends, but the process of determining the optimum strategy remains the same. For instance, if the system-level NRE cost were sufficiently low, or if the cost of storing components were sufficiently high, then variations in redesign cost may have a greater impact on obsolescence cost. Such scenarios should be investigated on a case-by-case basis using the procedure described above.

5.2.2 Experiment 2b — Ruggedization vs Component Cost

Experiment 2b tests the system-level obsolescence cost impact of component ruggedization assuming ruggedization impacts both component reliability (q_i) and component unit cost (c_i^{LOT}). To do this, the CASCADE MILP formulation developed in Chapter 3 is applied to a representative system while the parameters of that system are varied. The procedure for Experiment 2b is as follows:

Step 1: Define a Representative System — Firstly, a representative system is needed, about which an obsolescence mitigation strategy is defined. Also during this step, cost and schedule parameters are defined, including system-level NRE cost (c^{NRE}), component life cycles (Δ_i), discount rate (r), and holding cost (c_i^H). Component cost and reliability parameters are not defined in this step, since these values are varied during experimentation.

Step 2: Compute Obsolescence Cost Under Varying c_i^{LOT} and q_i — Next, for each combination of component cost (c_i^{LOT}) and annual failures (q_i), an optimum obsolescence mitigation plan is generated using the Mixed Integer Linear Programming formulation developed in Chapter 3.

Step 3: Visualize Cost Impact — By plotting the obsolescence cost (c_{obs}) as a function of component cost (c_i^{LOT}) and annual failures (q_i), the impact of ruggedization on obsolescence cost is determined[8].

To reduce the dimensionality of the problem, a simple 1-component system described in Table 18 is used in this analysis[9]. To protect proprietary information, the cost and reliability inputs are normalized and are presented as unitless, and a

[8] If specific component-level cost-vs-reliability relationships are known, these can be superimposed on the surface to determine exact trends. Several such trends are explored in Appendix F.

[9] Note that using a 1-component system has the added benefit of "magnifying" the cost trends associated with ruggedization, making them easier to distinguish.

Table 18: Table of component-level and system-level parameters for Experiment 2b.

	Component 1
Δ_i	6 yrs.
c_i^{LOT}	.0285–.285
c_i^{H}	.007125
c_i^{red}	1
q_i	.025–0.1

	System
c^{NRE}	.5
d_{IOC}	2015
Production Dates	$\{2015, 2020, 2025, 2030, 2035, 2040, 2045, 2050, 2055\}$
d_{EOL}	2065
d_{base}	2015

generic component name is used. The results of Experiment 2b are split into two sections: First, § 5.2.2.1 compares the CASCADE MILP result under a private-sector discount rate and a government discount rate. Second, Section 5.2.2.2 compares the CASCADE MILP result against three baseline strategies, including the Reactive Last-Time Buy, Reactive Bridge-Buy, and the Original MILP formulation.

5.2.2.1 Experiment 2b Results: Private vs Government

Figure 71 depicts the obsolescence cost trends associated with ruggedization assuming a private-sector discount rate (r=0.12). The results indicate that the optimum obsolescence cost (c_{obs}) varies nonlinearly with both annual failures (q_i) and component cost (c_i^{LOT}). Furthermore, Figure 72 illustrates that the optimum obsolescence mitigation plan changes throughout the domain. At low annual failures and low component cost, the optimum obsolescence mitigation plan requires zero redesigns. As annual failures and component cost both increase, the number of required redesigns increases. This provides further evidence that obsolescence mitigation plans must be generated on a case-by-case basis, and that no single plan will be optimal for all

systems.

Figure 73 depicts the obsolescence cost trends associated with ruggedization assuming a government discount rate (r=0.034). The results indicate that the optimum obsolescence cost (c_{obs}) varies nonlinearly with both annual failures (q_i) and component cost (c_i^{LOT}), following a similar trend to Figure 71. The two trends are not identical, however, as can be seen when comparing the number of required redesigns indicated by Figures 72 and 74. Figure 74 illustrates that a much larger portion of the domain calls for a zero-redesign plan as compared to the private-sector (shown in Figure 72). Despite this, the CASCADE MILP requires more redesigns as annual failures and component cost both increase, regardless of the discount rate.

The trends illustrated in Figures 71 through 74 lend insight into the basic tradeoffs associated with component ruggedization. Using either discount rate, there is always a cost benefit to reducing annual failures at a fixed component cost, and there is always a cost penalty to increasing component cost at a fixed number of failures — as expected. Whether or not there is a system-level obsolescence cost benefit to component ruggedization, however, depends on the component cost-vs-reliability trend that is expected when each component is ruggedized. If a small increase in reliability requires a large increase in unit cost, then it is reasonable to expect an overall cost penalty when ruggedizing. If, however, increasing component reliability through ruggedization is relatively easy (and thus requires little cost), then it is reasonable to expect an overall obsolescence cost benefit when ruggedizing.

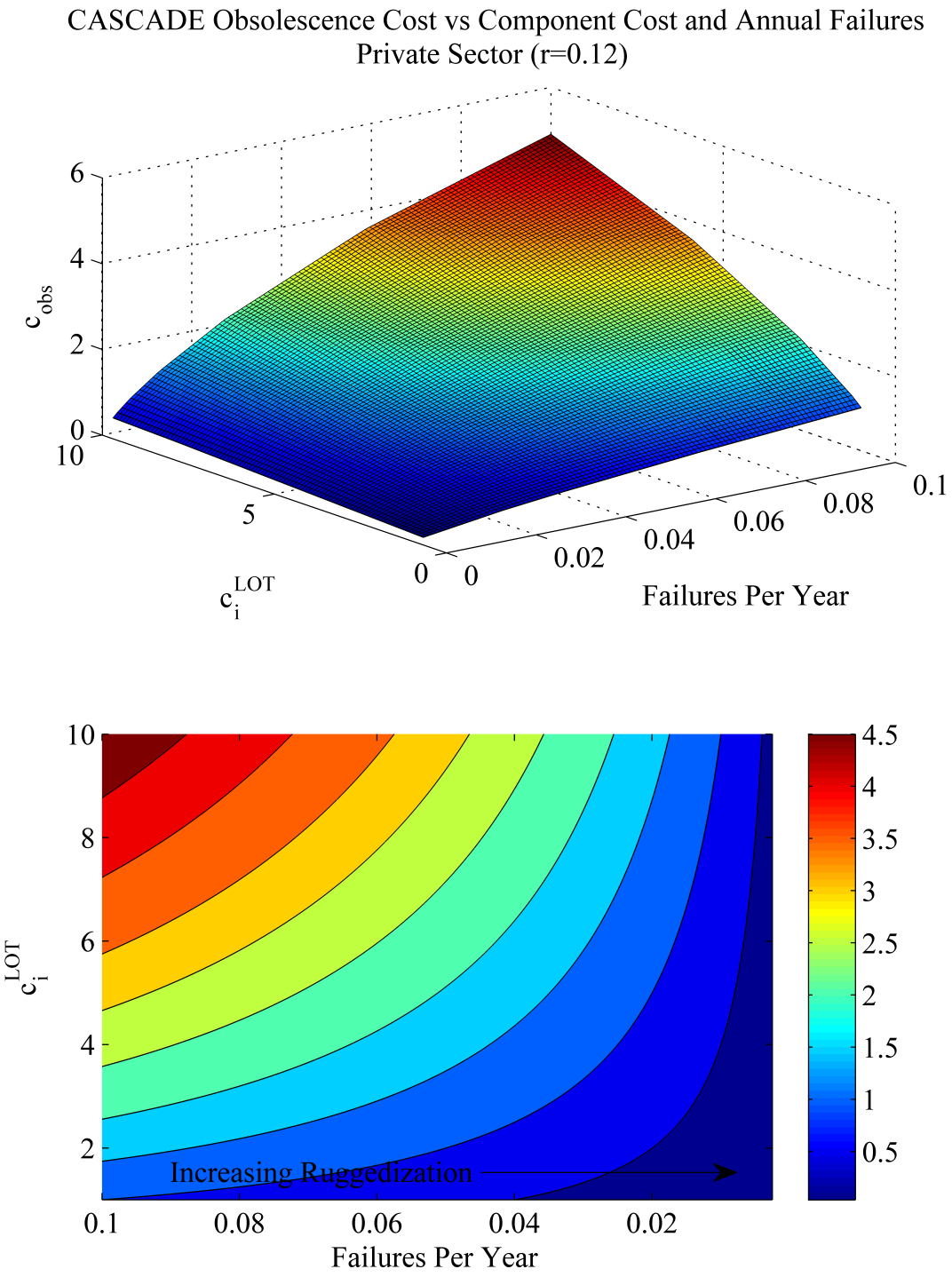

Figure 71: Optimum obsolescence cost (c_{obs}) as a function of component cost (c_i^{LOT}) and annual failures (q_i) assuming a private-sector discount rate (r=0.12). Note that the obsolescence cost has been normalized such that the obsolescence cost using unruggedized components equals 1.

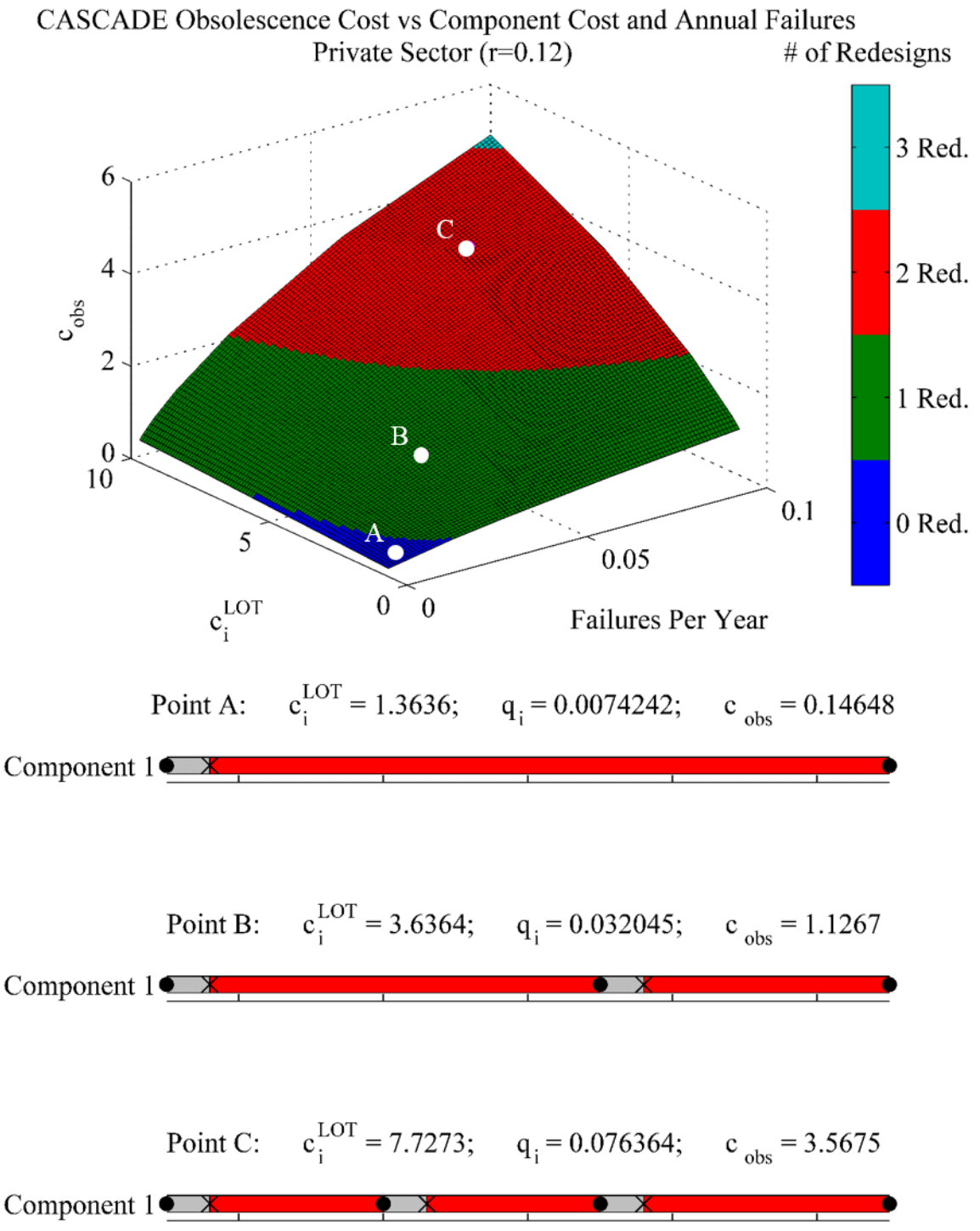

Figure 72: Using a private-sector discount rate (r=0.12), the nature of the optimum obsolescence mitigation plan changes throughout the domain. As the annual failures (q_i) and component cost (c_i^{LOT}) increase, the number of redesigns required by the optimum plan also increases as indicated by the coloration in the surface plot. When this occurs, the obsolescence cost (c_{obs}) on the z axis varies to reflect this change. Note that the obsolescence cost has been normalized such that the obsolescence cost using un-ruggedized components equals 1.

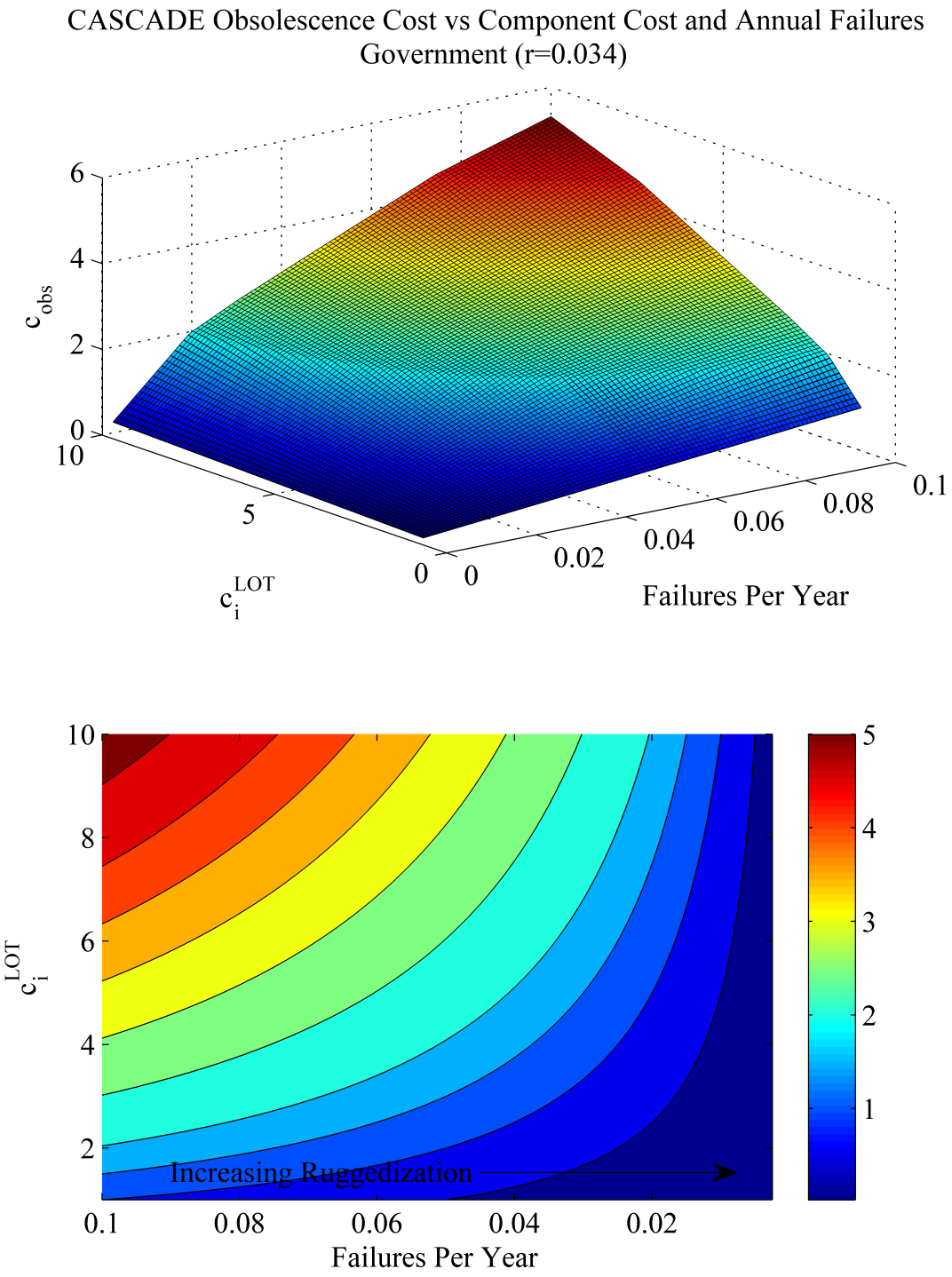

Figure 73: Optimum obsolescence cost (c_{obs}) as a function of component cost (c_i^{LOT}) and annual failures (q_i) assuming a government discount rate (r=0.034). Note that the obsolescence cost has been normalized such that the obsolescence cost using unruggedized components equals 1.

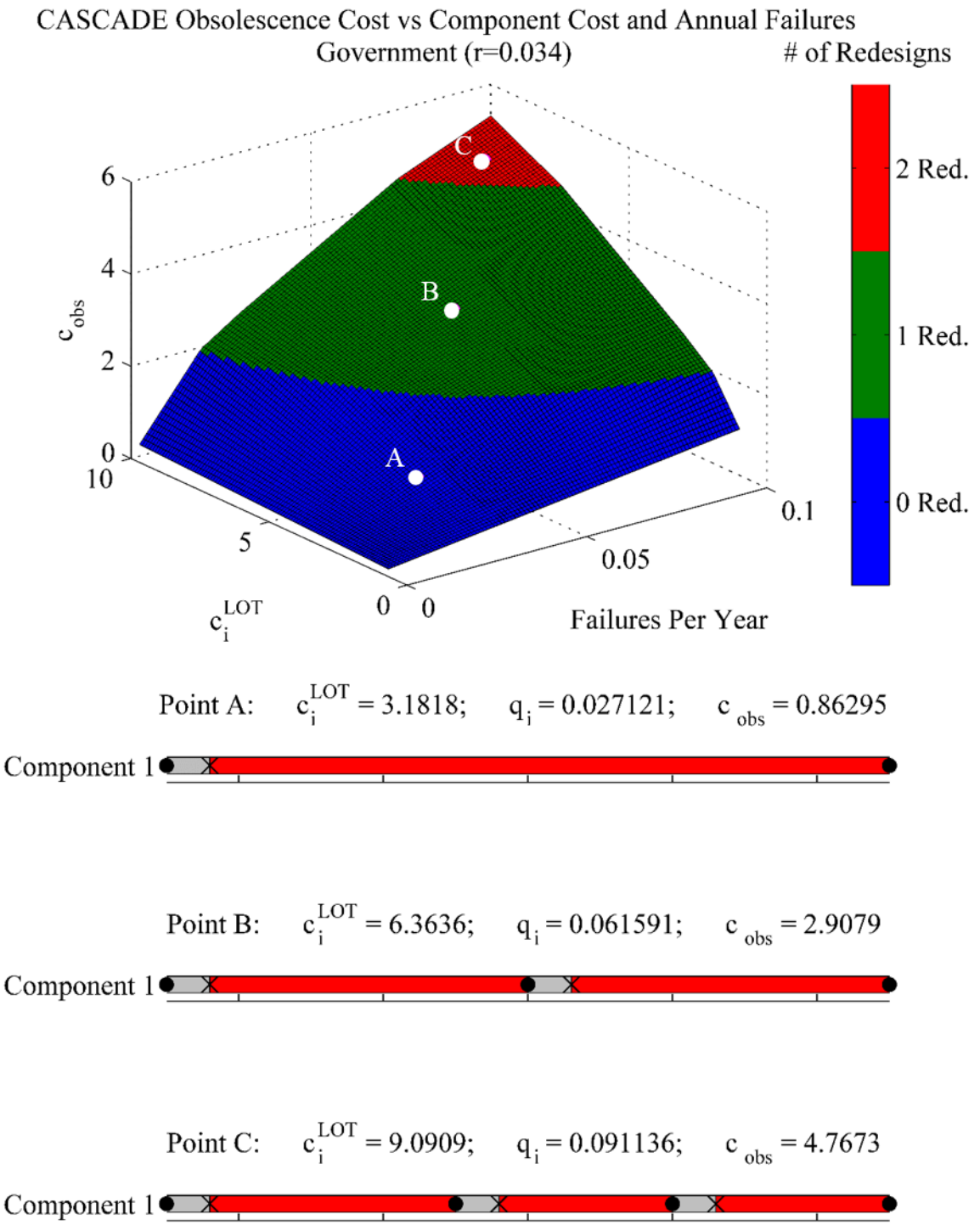

Figure 74: Using a government discount rate (r=0.034), the nature of the optimum obsolescence mitigation plan changes throughout the domain. As the annual failures (q_i) and component cost (c_i^{LOT}) increase, the number of redesigns required by the optimum plan also increases as indicated by the coloration in the surface plot. When this occurs, the obsolescence cost (c_{obs}) on the z axis varies to reflect this change. This result is similar to the private-sector result depicted in Figure 72. Note that the obsolescence cost has been normalized such that the obsolescence cost using unruggedized components equals 1.

5.2.2.2 Experiment 2b Results: Comparison Against Baselines

Figure 75 illustrates the obsolescence cost (c_{obs}) trend under varying annual failures (q_i) and component unit cost (c_i^{LOT}) using the Reactive Bridge-Buy baseline strategy described in § 4.2.1. Using this strategy, each time obsolescence is encountered it is first mitigated using a short-term bridging action that lasts until the next-possible redesign date, at which point the obsolete component is replaced via a redesign. As Figure 75 shows, obsolescence cost varies nonlinearly with both annual failures and redesign cost, similar to the CASCADE MILP trend shown in Figure 71. However, the Reactive Bridge-Buy strategy shown in Figure 75 produces obsolescence mitigation plans costing an order of magnitude more than the CASCADE MILP, highlighting the superiority of the CASCADE MILP.

Figure 76 illustrates the obsolescence cost (c_{obs}) trend under varying annual failures (q_i) and component unit cost (c_i^{LOT}) using the Reactive Last-Time Buy baseline strategy described in § 4.2.1. Using this strategy, each instance of obsolescence is mitigated using a single long-term bridging action that lasts until the end of the system's life. As Figure 76 shows, obsolescence cost varies nonlinearly with both annual failures and redesign cost, similar to the CASCADE MILP trend shown in Figure 71. In fact, as illustrated in Figures 72 and 74, the CASCADE MILP converges to the same result as the Reactive Last-Time Buy baseline when annual failures and component costs are low. As the number of annual failures and component cost both grow, the CASCADE MILP diverges from the Reactive Last-Time Buy baseline, as can be seen when comparing Figures 71 and 76.

Finally, Figure 77 illustrates the obsolescence cost (c_{obs}) trend under varying annual failures (q_i) and component unit cost (c_i^{LOT}) using the Original MILP formulation as a baseline. As with the previous two baselines, the Original MILP baseline produces obsolescence cost trends which vary nonlinearly as component cost and annual

failures vary. In fact, the Original MILP formulation produces results that are identical to the Reactive Bridge-Buy baseline. This suggests that the parameters of the system described in Table 18 have driven the Original MILP to the limits of its feasible space. In this case, that limit coincides with the Reactive Bridge-Buy baseline. This same behavior was captured earlier in § 4.2.3.

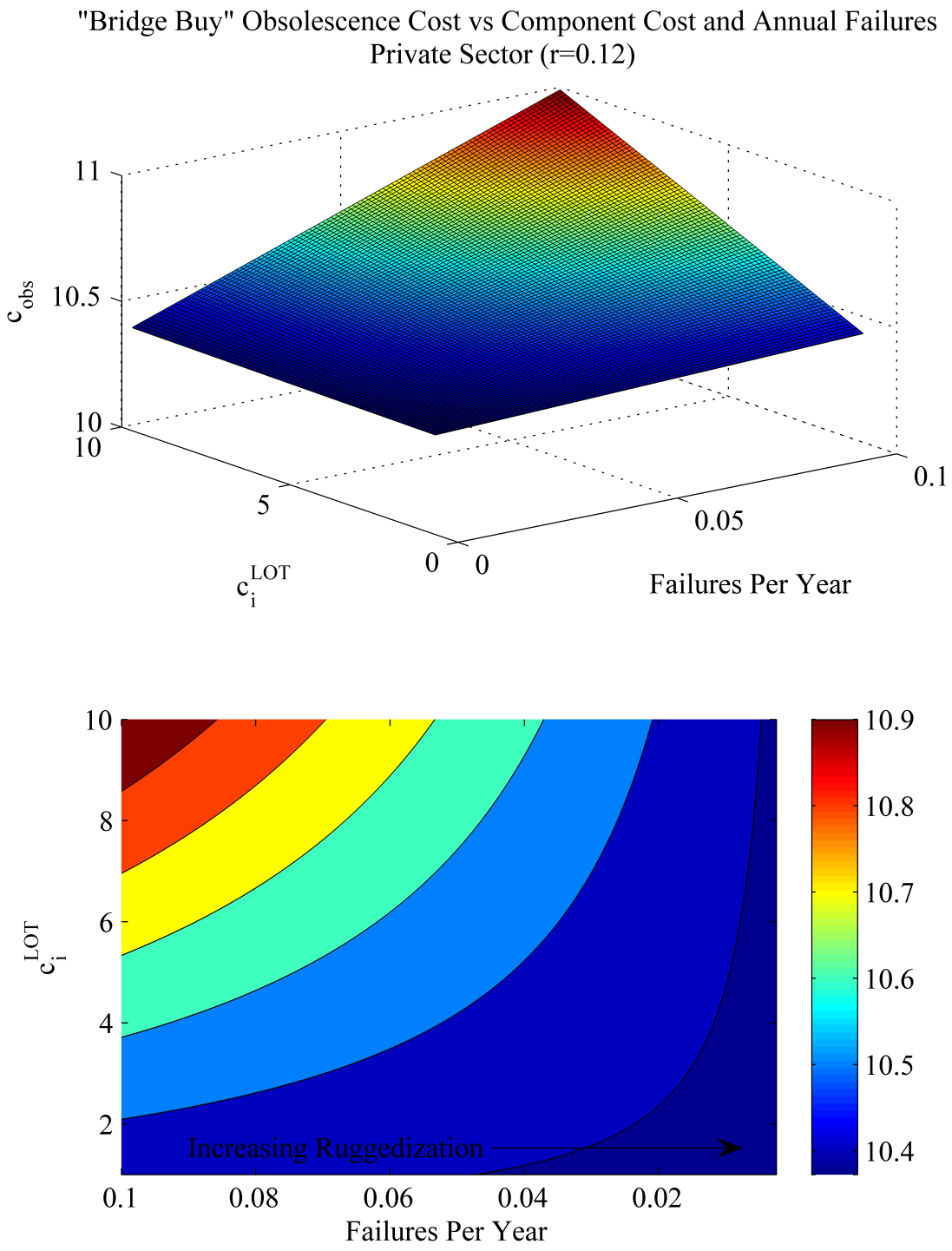

Figure 75: Obsolescence cost (c_{obs}) as a function of component cost (c_i^{LOT}) and annual failures (q_i) using the Reactive Bridge-Buy baseline strategy, and assuming a private-sector discount rate (r=0.12). Note that despite a similar trend, this strategy results in an obsolescence cost that is an order of magnitude larger than the CASCADE strategy depicted in Figure 71, indicating that the Bridge-Buy baseline is highly sub-optimal within this domain.

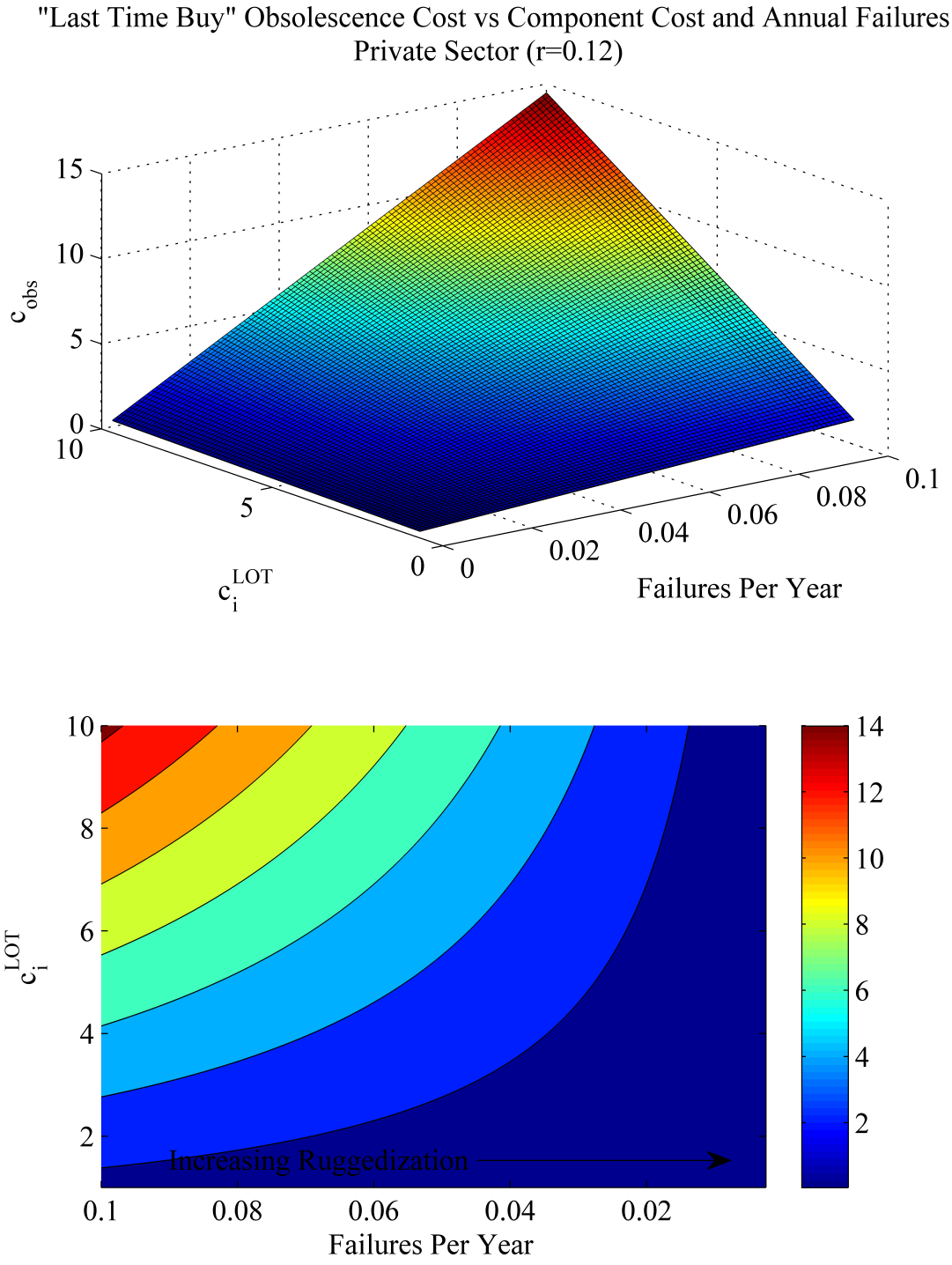

Figure 76: Obsolescence cost (c_{obs}) as a function of component cost (c_i^{LOT}) and annual failures (q_i) using the Reactive Last-Time Buy baseline strategy, and assuming a private-sector discount rate (r=0.12). Comparing this result to the CASCADE result depicted in Figure 72 indicates that the Last-Time Buy baseline diverges from CASCADE when annual failures and component cost are high, indicating that the Last-Time Buy baseline is sub-optimal within this region of the domain..

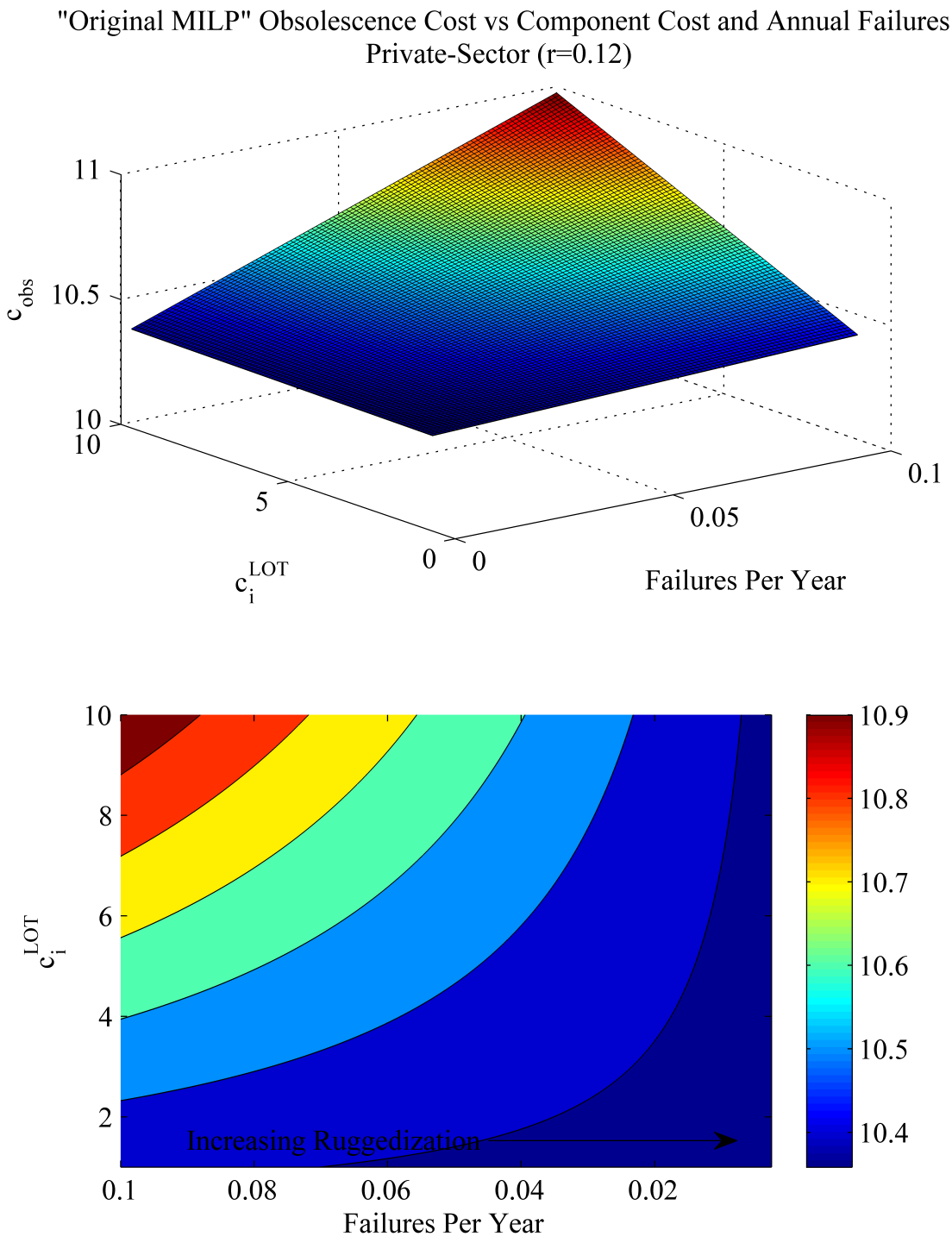

Figure 77: Obsolescence cost (c_{obs}) as a function of component cost (c_i^{LOT}) and annual failures (q_i) using the Original MILP formulation, and assuming a private-sector discount rate (r=0.12). Note that despite a similar trend, this strategy results in an obsolescence cost that is an order of magnitude larger than the CASCADE strategy depicted in Figure 71, indicating that the Original MILP formulation produces highly sub-optimal results within this domain.

5.2.2.3 Experiment 2b Conclusion

Experiment 2b investigates the impact of component modification on system obsolescence cost (c_{obs}) assuming redesign cost (c_i^{red}) remains constant but component cost (c_i^{LOT}) does not. An optimized design refresh plan was created using the CASCADE MILP formulation developed in Chapter 3, assuming a private-sector discount rate ($r = 0.12$) as well as a government discount rate ($r = 0.034$). The results show that under both financial environments, the trends between obsolescence cost (c_{obs}), component cost (c_i^{LOT}), and annual failures (q_i) look nearly identical. Modifications which improve reliability with no component cost increase always reduce obsolescence cost. Likewise, modifications that increase component cost with no impact on reliability always increase obsolescence cost. Otherwise, the impact on obsolescence cost depends on the component cost-vs-reliability curve. For shallow curves, increasing reliability will always improve obsolescence cost. For more dramatic curves (like exponential or reciprocal), however, increasing component reliability may actually increase obsolescence cost.

Furthermore, it was shown that the CASCADE MILP formulation out-performs both the Reactive Bridge-Buy Baseline and the Original MILP baseline throughout the entirety of the domain used. Furthermore, the CASCADE MILP formulation outperforms the Reactive Last-Time Buy baseline when annual failures (q_i) are high and component redesign cost (c_i^{red}) is low. It is important to consider, however, that the relationships observed herein are specific to the parameters used in this experiment. Other systems may result in different trends, but the process of determining the optimum strategy remains the same. For instance, if the system-level NRE cost were sufficiently high, or if the cost of storing components were sufficiently low, then variations in component cost may have a greater impact on obsolescence cost. Such scenarios should be investigated on a case-by-case basis using the procedure described above.

5.2.3 Experiment 2 Conclusion

Experiment 2 sought to answer Research Question 1.2, repeated below:

> **Research Question: 1.2**
> What trends emerge between reliability and cost when switching from low-level to high-level COTS?

The literature indicates that ruggedization can affect system and component cost in a variety of ways, including changes to per-unit component cost, and/or changes to non-recurring engineering cost. To address both possibilities, Research Question 1.2 was addressed using two experiments. Experiment 2a assumes that ruggedization has no effect on per-unit component cost, but that increasing ruggedization increases the cost of future redesigns. Experiment 2b assumes that the cost of redesigning remains constant, but that the per-unit component cost of ruggedization changes. For completeness, both of these scenarios were analyzed under a private-sector financial environment and a government financial environment.

Given the parameters of the example problem used, Experiment 2a show that the optimum strategy converges to the "Last Time Buy" strategy over nearly all of the domain for both the private-sector and government financial environments. As such, modifications which improve component reliability always reduce the cost of obsolescence. It is, however, possible that modifications may reduce reliability (such as changes to a component's form factor), in which case modifications should be avoided so as to limit increases in obsolescence cost.

Experiment 2b, which considers component cost changes as a function of reliability, yields more nuanced results. In this scenario, unlike in Experiment 2a, the results are not planar. Thus, the obsolescence cost trend is strongly tied to the relationship of component cost and component reliability. When reliability can be "bought" at

a low price (i.e. a low slope), then obsolescence cost decreases monotonically with increasing modification. If, however, increasing component reliability comes at a high cost, then the optimum strategy may be to limit the degree of ruggedization.

Since many factors influence the relationship between component modification and system obsolescence cost, no one trend stands out as "correct." Instead, Research Question 1.2 must be answered as follows: *The impact of component modification on system obsolescence cost depends on several factors, including the financial environment, and the relationship between component reliability and cost. An analysis should be performed on a system-by-system basis to determine the optimum degree of component modification.*

CHAPTER VI

CONCLUSION

Chapter 1 identified the need for a new methodology for designing and planning for long life cycle COTS-based systems. An extensive literature review revealed Agile Systems Engineering as a means to design such systems, but a gap was revealed in how such systems are sustained. Addressing this gap required a means of forecasting when obsolescence would occur, computing the cost of competing obsolescence mitigation actions, and optimizing the set of all feasible obsolescence mitigation actions to create a single design refresh strategy. The elements which fit each of these needs were selected in Chapter 2, and these selected elements were combined and verified in Chapters 3 and 4 respectively. Finally, the verified methodology was applied to representative systems in a variety of different scenarios in Chapter5. This chapter summarizes the findings of this work, and offers guidance on future opportunities to extend this research.

6.1 Summary of Findings

This research was motivated by the rising cost of complex cyber-physical military systems, as well as a growing need to field such systems sooner. As Chapter 1 noted, one promising solution to both of these problems is the increased utilization of commercial off-the-shelf (COTS) components. Despite this, a literature search revealed that designing long-life cycle COTS-based systems using traditional systems engineering practices leads to long development times, and the delivery of out-of-date systems which are costly to sustain — thus eliminating the purported benefits of increased COTS utilization. This, coupled with a persistent need to mitigate obsolescence throughout the life cycle of COTS-based systems, leads to high sustainment costs.

Thus, this research aimed to develop a new COTS-based system design and sustainment methodology which addresses these challenges, as captured in the primary research objective, repeated below:

Research Objective
To develop a methodology for long life cycle COTS-based systems which addresses the obsolescence concerns associated with COTS components throughout the system's life cycle.

The research objective was decomposed into two primary research questions, the first of which addresses the challenges of designing COTS-based systems. A literature review in Chapter 2 traced these challenges primarily back to the requirements definition phase of the traditional Systems Engineering process. Thus, the first primary research question focuses on this phase of the Systems Engineering process, as repeated below:

Research Question: 1
How should the requirements definition phase of the engineering process be modified with obsolescence in mind to enable the development of a new COTS-based design methodology?

While answering Research Question 1 is useful for newly-designed COTS-based systems, it fails to address the aforementioned sustainment challenges present in both new and old COTS-Based systems. Thus, Research Question 2 serves as a compliment to Research Question 1, and is repeated below:

Research Question: 2
How should obsolescence mitigation strategies be incorporated into a new COTS-based design methodology?

Research Question 1 was addressed primarily through a literature review, starting with way COTS-based systems are designed. The literature review revealed that applying the traditional process-bound Systems Engineering process to the design of COTS-based systems leads to a mismatch in requirements. In short, through the traditional Systems Engineering process, stakeholders lock in their requirements early, and those requirements are decomposed and allocated to low-level system elements. While this works fine for the design of new systems, the design of COTS-based systems is more limited due to the "bottom-up" requirements imposed on the system by the COTS marketplace. Especially in complex systems which may take years to develop, it is not uncommon for the up-front requirements to conflict with the commercially-available components. Furthermore, the volatility of the commercial sector means that components that were available to procure at the *beginning* of the design process may be unprocurable by the *end* of design. To address these specific challenges, Research Question 1.1 was posed, as repeated below:

Research Question: 1.1
What changes need to be made to the SE Requirements Decomposition to better facilitate the use of COTS components?

Further literature review revealed that many of the challenges of designing COTS-based systems are not unique. In fact, the software development community addressed the challenges of volatile requirements and a volatile technological landscape through the use of Agile Software Development. Agile development allows software developers to embrace changes *during* development — whether those changes come from the customer or from changing technology. Thus, answering Research Question 1.1 began with the pursuit of an "Agile Systems Engineering" process. Although this notion exists in the literature, and indeed many elements of such a process have been proposed,

there was a lack of a unified framework that captured these elements. Therefore, Research Question 1.1 was answered via two conjectures based on an extensive literature review. Conjecture 1.1a addresses the need to consider the bottom-up requirements imposed on the system by the COTS marketplace. It is repeated below:

Conjecture: 1.1a
There is a need for an identification phase at the beginning of the systems engineering process, during which customer requirements are evaluated and prioritized for COTS applicability. Also during this phase, the COTS marketplace is evaluated, and the bottoms-up requirements imposed by available COTS components are balanced against stated customer requirements to encourage higher COTS utilization.

While Conjecture 1.1a addresses the challenge of decomposing an initial set of COTS-compatible requirements, it does not address the volatility of requirements *throughout* the design process. Conjecture 1.1b addresses this second element by proposing a periodic reassessment of the COTS marketplace throughout the design process, and it is repeated below:

Conjecture: 1.1b
There is a need for the periodic reassessment of the COTS marketplace throughout the systems engineering process. The goal of these assessments is to determine how the COTS technology landscape has changed, and to reevaluate selected COTS components. Components which are at or near obsolescence *must* be replaced, and new components which better meet the original requirements set should be considered.

Taken together, Conjectures 1.1a and 1.1b represent the backbone of the Cyber-physical Acquisition Strategy for COTS-Based Agility-Driven Engineering (CASCADE)

methodology. A flow diagram of the proposed CASCADE methodology is presented in Figure 78.

Conjectures 1.1a and 1.1b emphasize relaxing requirements to increase COTS utilization, which requires that some desired capabilities be sacrificed. In some cases, however, relaxing requirements may be undesirable, or even impossible. In such cases, the literature suggests *modifying* or *ruggedizing* existing COTS components to better suit the needs of the customer. While doing so may increase COTS utilization without sacrificing requirements, there is a lack of understanding in the literature of the impact that ruggedization has on cost. Some sources suggest that ruggedizing COTS components will reduce failures and thus reduce sustainment costs, while other sources suggest that the failure rate may *increase* and/or the cost of repairs may become financially inviable. Research Question 1.2 aimed to find a quantifiable relationship between the degree of ruggedization and its impact on obsolescence cost, and it is repeated below:

Research Question: 1.2
What trends emerge between reliability and cost when switching from low-level to high-level COTS?

Since there is little agreement in the literature about the cost impact of ruggedization, and virtually no quantitative evidence to support the qualitative claims, Research Question 1.2 was addressed via experimentation. Anecdotal evidence suggested that ruggedization would lead to reduced Operations and Support (O&S) cost, but would require increased non-recurring engineering cost to account for the modifications. Thus, Hypothesis 1.2 was proposed, and is repeated below:

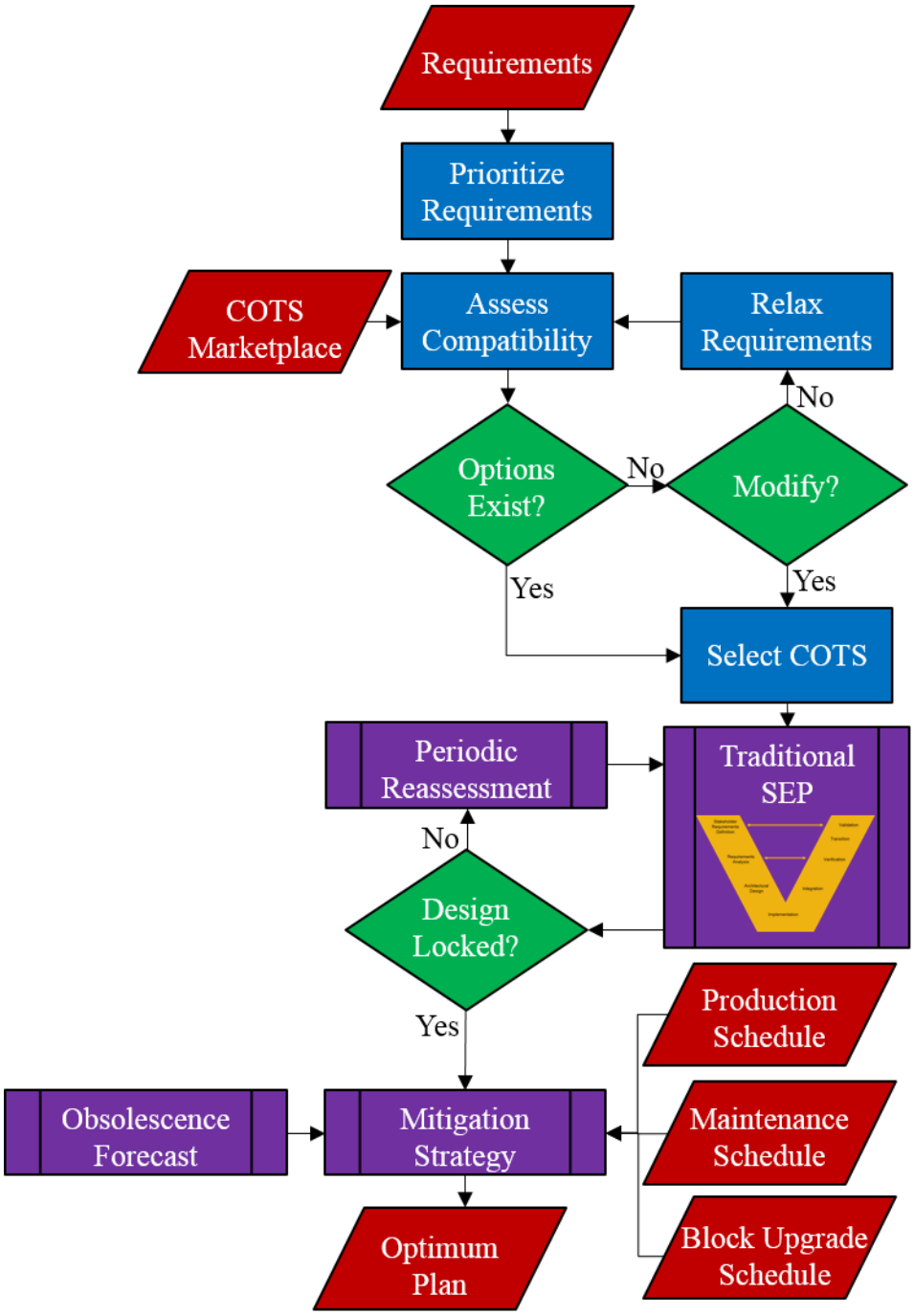

Figure 78: Flow Diagram of CASCADE methodology.

Hypothesis: 1.2
The increased cost of modification will outweigh the benefits of reduced failures, thus driving up the total obsolescence cost.

Hypothesis 1.2 was tested via two experiments presented in Chapter 5. The first experiment considers the impact of ruggedization on obsolescence cost assuming only non-recurring engineering cost is impacted. The second experiment assumes that non-recurring engineering cost is unaffected, but rather the unit cost of the ruggedized COTS components varies. Both of these experiments yielded three similar conclusions: First, the system-level cost impact of ruggedization depends heavily on the nature of the design refresh strategy employed. It was shown that the CASCADE MILP formulation invariably produces a strategy with the lowest obsolescence cost at all levels of ruggedization. Second, the system-level cost impact of ruggedization depends heavily on the nature of the financial sector for which the system is designed. Finally, the system-level cost impact of ruggedization depends on the component-level cost-vs-reliability trends associated with ruggedization. Generally speaking, when the cost of reliability is sufficiently low, increasing ruggedization monotonically reduces obsolescence cost. At higher rates of cost growth, a more detailed analysis is required.

The CASCADE methodology depicted in Figure 78 shows that the creation of a design refresh strategy requires an ability to forecast when obsolescence will occur. This naturally led to Research Question 2.1, repeated below:

Research Question: 2.1
What is a suitable method of forecasting obsolescence?

A literature review in Chapter 2 identified several options to fit this need, but no one method stood out as being "most suitable." In fact, there was a clear trade-off between the forecast accuracy and the amount of data required to perform the forecast.

Thus, it was concluded that the "most suitable" method depended on the needs of the overall CASCADE methodology. If, for instance, the overall CASCADE methodology was insensitive to forecast accuracy, then the simplest forecasting method could be employed. If instead, however, the CASCADE methodology was highly sensitive to forecast accuracy, then the most-accurate forecasting method should be selected, despite the increased data requirement. With this in mind, two experiments were conducted in Chapter 5: The first was designed to test Hypothesis 2.1a, repeated below:

Hypothesis: 2.1a

The accuracy of obsolescence forecasts will be a main driver of obsolescence cost for COTS-based systems.

The results of this experiment revealed that the impact of forecast accuracy was strongly driven by the financial sector for which the system was designed. At low discount rates (e.g. r=0.034, as in the Government), the optimum design refresh strategy employed few redesigns throughout they system's life cycle. Consequently, forecast accuracy had little impact on the obsolescence cost of the optimum strategy, and thus a less-accurate forecasting method could be used[1]. In a more aggressive financial environment (e.g. r=0.12, as in the Private Sector), the optimum design refresh strategy employed many obsolescence mitigation actions throughout the system's life cycle, and so the accuracy of obsolescence forecasts was a main driver. In such cases, a highly-accurate forecasting approach such as Data Mining (or a proprietary method) should be considered.

The impact of forecast accuracy is moot, however, unless obsolescence cost is shown to be non-increasing with increasing forecast accuracy. Though it may seem

[1]Furthermore, as Chapter 5 notes, in the unique case in which the optimum strategy is a zero-redesign strategy, then the Manufacturer Inquiry forecasting approach is likely the best option.

trivial, the second and final necessary element to answering Research Question 2.1 requires the support of Hypothesis 2.1b, repeated belo:

Hypothesis: 2.1b
The obsolescence cost will be non-increasing with increasing accuracy of obsolescence forecasts.

The associated experiment, presented in Chapter 5, showed that there are two potential consequences to an inaccurate obsolescence cost estimate: If obsolescence cost is *over*-estimated, then resources are allocated which are never utilized, resulting in lost opportunity cost. If obsolescence cost is *under*-estimated, then too few resources are allocated, and the system cannot be adequately supported through its entire life cycle. To avoid the latter, an obsolescence cost buffer should be budgeted, the size of which depends on the forecast accuracy, as well as the desired confidence in avoiding a cost overrun. The results of this experiment showed that the size of this buffer increases monotonically as forecast uncertainty increases. Thus, Hypothesis 2.1b was supported.

Given forecasted obsolescence dates for each component, the final element of the CASCADE methodology is the generation of an optimal design refresh strategy. The literature in Chapter 2 revealed that most systems are supported via a purely reactive means. That is, obsolescence mitigation actions are only considered once obsolescence has already occurred. The CASCADE methodology seeks a more pro-active approach to obsolescence mitigation through the creation of design refresh strategies. A literature review of non-proprietary design refresh planning methods revealed a sizable gap in existing capabilities. Many existing non-proprietary methods fail to consider the time-value of money, and most only consider one component at a time. This naturally led to Research Question 2.2, repeated below:

Research Question: 2.2

What is a suitable method to generate optimal obsolescence mitigation plans that minimize obsolescence cost?

While no method was found that met the needs of the CASCADE methodology, three methods stood out as being complimentary components of a new method: The first piece is the Mixed Integer Linear Programming (MILP) Design Refresh model presented by Zheng *et al.*, which allows optimum design refresh strategies to be generated for multi-component systems. Despite serving as a sound starting point, the original MILP formulation makes simplifying assumptions which do not adequately capture the needs of the CASCADE methodology. The last two pieces are the obsolescence cost model presented by Bradley and Guerrero and the Porter Design Refresh model. These cost formulations allow for the cost of individual obsolescence mitigation actions to be computed in a transparent and mathematically-concise manner. Thus, Research Question 2.2 was answered via Conjecture 2.2, repeated below:

Conjecture: 2.2

The Mixed Integer Linear Programming Design Refresh model provides a robust foundation for optimizing obsolescence mitigation plans. By relaxing its assumptions and incorporating the desirable elements of the Bradley and Guerrero model and Porter's Design Refresh Model, a new MILP formulation can be created that explores a larger feasible space.

By relaxing the constraints of the original MILP Design Refresh model, and incorporating elements of the Bradley and Guerrero cost model, a new CASCADE MILP formulation was created in Chapter 3. Before this new formulation could be applied to representative systems, however, it was necessary to verify it against expected results. This was done in Chapter 4. Through this verification process, it was proven

that the CASCADE MILP formulation produces optimized design refresh strategies that are as good as, and often better than, the original MILP formulation. It was also shown that the new obsolescence cost formulation agrees with existing results such as the Porter Design Refresh Model.

6.2 Contributions

This research provides several contributions to the fields of systems engineering and design refresh planning. These contributions relate to the primary research objective, the techniques developed to answer the research questions, as well as the answers to those research questions, themselves.

The first contribution is the development of a single, continuous, agile systems engineering framework specifically aimed at COTS-based systems. One of the primary concerns with designing COTS-based systems — especially complex COTS-based systems — is the volatility of both requirements and technologies. When requirements are explicated too early in the development of COTS-based systems (as is done in the traditional systems engineering process), there is little to no leeway for dealing with changing requirements, or adapting to a changing technological landscape. While many sources point towards the notion of "Agile Systems Engineering" to address this problem, virtually no sources offered a concise and step-by-step framework with which to carry out this process. CASCADE provides this by combining proven elements of Agile processes into a single concise framework.

The second contribution is the development of a single, exhaustive cost formulation which allows the impact of individual obsolescence mitigation actions to be assessed. Some design refresh planning methods rely on subject-matter experts to give the cost of individual obsolescence mitigation actions. This means of cost estimation lacks traceability, and subject-matter experts are prone to biases. The CASCADE cost model combines the most-desirable elements of existing cost models, and it allows for

the rapid and transparent computation of obsolescence cost impacts. Thus, unlike subject-matter experts, this new CASCADE cost model allows for a more exhaustive exploration of potential design refresh strategies.

The third (and arguably most-significant) contribution is the creation of a new CASCADE Mixed Integer Linear Programming Design Refresh model. This new model allows an optimum design refresh strategy to be selected from among thousands of feasible options. Furthermore, the proofs provided in Chapter 4 show that the set of feasible solutions in the CASCADE MILP model represent a superset of the alternative design refresh strategies — including the original MILP formulation. Thus, it has been proven that the CASCADE MILP produces design refresh strategies which are always as good as or better than existing baselines.

Finally, by applying the CASCADE MILP formulation generated in this document, a new means of evaluating the impact of ruggedization was created. This allows for the straightforward evaluation of the impact of ruggedization on obsolescence cost.

6.3 Future Work

While the CASCADE methodology provides many contributions to the field of obsolescence management, the scope of this research leaves room for future work. For instance, the CASCADE systems engineering framework presented in Chapter 2 provides general guidelines for how to carry out the agile development of a COTS-based system. While a literature review gives merit to the structure and elements of this framework, a true test of this CASCADE framework will come from its application in a real-world problem.

Secondly, the application of the CASCADE systems engineering framework will also help to address some unanswered elements of this methodology. For instance, this research proposes that up-front technical requirements be prioritized and balanced

against the components available in the COTS marketplace. While this proposal solves the problem of requirements mismatch at a high level, the method for *how* to prioritize and relax requirements is yet untested. Furthermore, while the CASCADE framework proposes iteratively reassessing the COTS marketplace throughout the systems engineering process, it is unclear how often this should be done. For instance, is there a discrete point in the design process past which components should not be swapped? Is there a systematic way to determine the best set of components to consider swapping? What are the underlying trades and trends associated with requirements rigidity and the timing of COTS component selection?

Thirdly, the CASCADE MILP formulation takes pre-planned block upgrades as an input, but the source of this input is yet undetermined. This begs the question: What is a systematic process for determining the optimal timing of design refreshes? How can the CASCADE MILP formulation be altered to allow for the simultaneous optimization of block upgrade scheduling? Addressing these questions will provide an integrated environment which enables the simultaneous optimization of multiple objectives.

Finally, the CASCADE methodology assumes that decision makers have down-selected to a COTS-based architecture. This implicitly assumes that an open systems architecture is the best solution to managing obsolescence and delivering state-of-the-art systems. An extension of this research is needed that addresses the high-level decision to develop an open system or a monolithic system.

APPENDIX A

MATLAB CODE — COST FUNCTION

```
function [ costObs ] = CASCADE_CostFunction(t1, t2, tBase, demandRate,...
discountRate, costReplace, costHolding, costRedesign, costNRE, costLOT)
%CASCADE_CostFunction computes the cost of specific obsolescence mitigation
%   actions for COTS-based systems. A description of this cost model is
%   given in the dissertation entitled ``Cyber-physical Acquisition
%   Strategy for COTS-based Agility-Driven Engineering'' written by
%   Nathan C.L. Knisely


%INPUTS:
%   t1 = starting year
%   t2 = ending year
%   tBase = base year (the year to which costs will be discounted)
%   demandRate = demand rate. Assumed to be constant over time
%   discountRate = discount rate. Assumed to be constant over time
%   costReplace = Difference in unit cost for components being replaced
%   costHolding = Annual holding cost per unit of stored items for LOT Buy
%   costRedesign = NRE cost of redesigning a component/system
%   costNRE = System-level NRE cost associated with redesigning a component
%               or components in the system.
%   costLOT = Unit cost of components purchased for the life-of-type(LOT)
%               Buy obsolescence mitigation action


if discountRate ~= 0
```

```
costRecurring = demandRate.*(exp(-t1.*discountRate).*...
(costHolding.*(-t1.*discountRate+t2.*discountRate-1) + ...
costReplace.*discountRate)+exp(-discountRate.*t2).*...
(costHolding-costReplace.*discountRate))./discountRate.^2;
else
costRecurring = costReplace .* (t2 - t1) .* demandRate...
+ costHolding .* t2 .* demandRate .* (t2 - t1)...
- costHolding .* demandRate ./2 .* (t2.^2 + t1 .^2);
end

costNonrecurring = costRedesign.*exp(-discountRate.*t1)...
+ costNRE.*exp(-discountRate.*t1)...
+ costLOT.*(t2-t1) .* demandRate .* exp(-discountRate.*t1);

costObs = exp(discountRate*tBase).*(costRecurring + costNonrecurring);
end
```

APPENDIX B

MATLAB CODE — OPTIMIZER

```
function [c_obs, numRed] = CASCADE_MILP(componentMatrix,...
NRE_Cost, IOC, endOfLife, baseYear, redesignDates,...
freeCertDates, discountRate, shouldPlot)
%CASCADE_MILP generates optimum obsolescence mitigation plans using a
%   Mixed Integer Linear Programming (MILP) formulation presented in the
%   dissertation entitled ``Cyber-physical Acquisition Strategy for
%   COTS-based Agility-Driven Engineering'' written by Nathan C.L. Knisely


%OUTPUTS:
%   c_obs = Obsolescence cost for the optimum obsolescence mitigation plan
%   numRed = Number of redesigns required by the optimum obsolescence
%               mitigation plan

%INPUTS:
%   componentMatrix = A 6-by-n matrix of parameters for each component in
%       the COTS-based system of interest, where n is the number
%       of components. Row 1 gives the component index. Row 2 gives the
%       expected component life cycle (Delta_i). Row 3 gives the unit cost
%       for each component when a Life-of-Type Buy action is taken.
%       Row 4 gives the cost to redesign the component. Row 5 gives the
%       annual holding cost per units being stored for a Life-of-Type Buy.
%       Row 6 gives the demand rate (annual failures) of each component
%   NRE_Cost = System-level non-recurring engineering (NRE) cost incurred
```

```
%       each time a component is redesigned
%   IOC = Year of initial operatinc capability
%   endOfLife = Year of the system's end of life
%   baseYear = Base year to which costs are discounted
%   redesignDates = Vector of dates during redesigns may occur
%   freeCertDates = Vector of dates for which redesigns do not incur
%a system-level non-recurring engineering cost
%   discountRate = Discount rate used for computing the net-present value
%   shouldPlot = Binary (0/1) value which indicates whether or not the
%       optimum obsolescence mitigation plan should be plotted. 1=Plot,
%       0=Don't Plot


numberOfComponents = size(componentMatrix,2);%# of components in system

%% Array of dates when redesigns can occur:
redesignDateMatrix = [];

%% Array of Dates when a Bridge Buy can start (row 1) and end (row 2)
% and component index (row 3):
bridgeBuyDates = [0;0;0];
for i = 1:numberOfComponents
delta = componentMatrix(2,i);
for j = 1:size(redesignDates,2);
tempBBDate = redesignDates(1,j) + delta;
tempREDDate = redesignDates(1,redesignDates(1,:) > tempBBDate);

bridgeBuyDates = ...
[bridgeBuyDates(1,:) ones(1,size(tempREDDate,2))*tempBBDate;...
bridgeBuyDates(2,:) tempREDDate;...
bridgeBuyDates(3,:) ones(1,size(tempREDDate,2))*i];
end
```

```
redesignDateMatrix = [redesignDateMatrix [redesignDates; ...
NaN(2,length(redesignDates)); ones(1,length(redesignDates))*i]];
end
bridgeBuyDates = bridgeBuyDates(:,2:end); %removes zeros from first column



for j = 1:size(redesignDateMatrix,2)
i=redesignDateMatrix(4,j);
jStar = redesignDateMatrix(1,j);
delta = componentMatrix(2,i);

%% Constraint 1
xTemp = zeros(1,size(redesignDateMatrix,2));
xTemp(j) = -1;

yTemp = zeros(1,size(redesignDates,2));

zTemp = zeros(1,length(bridgeBuyDates));
zTemp(bridgeBuyDates(2,:) > (jStar + delta) &...
bridgeBuyDates(1,:) == jStar + delta & bridgeBuyDates(3,:)==i) = 1;

const1(j,:) = [xTemp yTemp zTemp];

%% Constraint 2
xTemp = zeros(1,size(redesignDateMatrix,2));
xTemp(j) = -1;

yTemp = zeros(1,size(redesignDates,2));

zTemp = zeros(1,size(bridgeBuyDates,2));
zTemp(bridgeBuyDates(2,:)==jStar & bridgeBuyDates(1,:)...
```

```
< jStar & bridgeBuyDates(3,:)==i) = 1;

const2(j,:) = [xTemp yTemp zTemp];

%% Constraint 3
xTemp = zeros(1,length(redesignDateMatrix));
xTemp(jStar<redesignDateMatrix(1,:) & redesignDateMatrix(1,:)...
<=(jStar+delta) & redesignDateMatrix(4,:)==i) = -1;

yTemp = zeros(1,size(redesignDates,2));

zTemp = zeros(1,length(bridgeBuyDates));
zTemp(bridgeBuyDates(1,:)<=(jStar+delta) & bridgeBuyDates(2,:)...
>(jStar+delta) & bridgeBuyDates(3,:)==i) = -1;

const3(j,:)= [xTemp yTemp zTemp];




%% Constraint 4
xTemp = zeros(1,length(redesignDateMatrix));
xTemp(redesignDateMatrix(1,:) == jStar) = 1;

yTemp = zeros(1,size(redesignDates,2));
yTemp(redesignDates==jStar) = -length(xTemp)*2;

zTemp = zeros(1,length(bridgeBuyDates));

const4(j,:)= [xTemp yTemp zTemp];

end
```

```
%Removes all rows that are all zero
const1(sum(abs(const1),2)'==0,:)=[];
const2(sum(abs(const2),2)'==0,:)=[];
const3(sum(abs(const3),2)'==0,:)=[];
const4(sum(abs(const4),2)'==0,:)=[];
const4=unique(const4,'rows');

%concatinate constraints into one matrix
inEqualityConstraintsLEFT = [const1; const2; const3; const4];
inEqualityConstraintsRIGHT = [zeros(size(const1,1),1);...
zeros(size(const2,1),1);...
-1 * ones(size(const3,1),1);...
zeros(size(const4,1),1)];

%remove rows with identical constraints
[inEqualityConstraintsLEFT ia ic] = ...
unique(inEqualityConstraintsLEFT,'rows');
inEqualityConstraintsRIGHT = inEqualityConstraintsRIGHT(ia);



%% Calculate Cost Coefficients

% Redesign Costs (C^R)
costX = zeros(1,length(redesignDateMatrix));
for i = 1:length(redesignDateMatrix)

componentIndex = redesignDateMatrix(4,i);
t1 = redesignDateMatrix(1,i);
t2 = 0; %Not applicable for redesigns. Defaulted to zero.
```

```
demandRate = 0; %Not applicable for redesigns. Defaulted to zero.

costItem = 0; %Not applicable for redesigns. Defaulted to zero.
costHolding = 0; %Not applicable for redesigns. Defaulted to zero.
costRedesign = componentMatrix(4,componentIndex);
c_cert_temp = 0; %Not applicable for redesigns. Defaulted to zero.
costLOT = 0; %Not applicable for redesigns. Defaulted to zero.
costX(i) = CASCADE_CostFunction(t1,t2,baseYear,demandRate,...
discountRate,costItem,costHolding,...
costRedesign,c_cert_temp,costLOT);
end

%Cost of first and last redesigns = 0. This insures that the optimum
%   plan is anchored to the start and end of the system life cycle
for i = 1:numberOfComponents
first = find(redesignDateMatrix(4,:)==i,1,'first');
last = find(redesignDateMatrix(4,:)==i,1,'last');

costX(first) = 0;
costX(last)=0;
end



%System-level NRE cost (C^{NRE})
costY = ones(1, size(redesignDates,2)) * NRE_Cost;
for i = 1:size(redesignDates,2)

t1 = redesignDates(i);
t2 = 0; %Not applicable for redesigns. Defaulted to zero.
demandRate = 0; %Not applicable for redesigns. Defaulted to zero.

costItem = 0; %Not applicable for redesigns. Defaulted to zero.
```

```
costHolding = 0; %Not applicable for redesigns. Defaulted to zero.
costRedesign = 0;
c_cert_temp = NRE_Cost; %Not applicable for redesigns. Defaulted to zero.
costLOT = 0; %Not applicable for redesigns. Defaulted to zero.
costY(i) = CASCADE_CostFunction(t1,t2,baseYear,demandRate,...
discountRate,costItem,costHolding,costRedesign,c_cert_temp,costLOT);
end

%sets the NRE cost for IOC and endOfLife to zero. This helps insure that
%   the optimum plan is anchored at the start and end of the system's life
%   cycle
costY(1)=0;
costY(end)=0;


%Set any pre-planned maintainance/block-upgrade dates to have zero
%   certification cost
for i = 1:length(freeCertDates)
costY(redesignDates==freeCertDates(i))=0;
end


%Cost of bridging actions using the LOT buy obsolescence mitigation action
%   (C^B)
costZ = zeros(1,length(bridgeBuyDates));
for i = 1:length(bridgeBuyDates)
componentIndex = bridgeBuyDates(3,i);
t1 = bridgeBuyDates(1,i);
t2 = bridgeBuyDates(2,i);
demandRate = componentMatrix(6,componentIndex);
costItem = 0;
costHolding = componentMatrix(5,componentIndex);
```

```
costRedesign = 0;
c_cert_temp = 0;
costLOT = componentMatrix(3,componentIndex);
costZ(i) = CASCADE_CostFunction(t1,t2,baseYear,demandRate,...
discountRate,costItem,costHolding,costRedesign,c_cert_temp,costLOT);
end

%all cost vectors are combined into one for use in the intlinprog function
f = [costX costY costZ]';
intcon = 1:length(f);
intcon = intcon';
A = inEqualityConstraintsLEFT;
b = inEqualityConstraintsRIGHT;
Aeq = [];%there are no equality constraints
beq = [];%there are no equality constraints
lb = zeros(length(f),1);%lower bound = 0
ub = ones(length(f),1);% upper bound = 1


options = optimoptions('intlinprog','display','off');
[optimum c_obs temp temp2]=intlinprog(f,intcon,A,b,Aeq,beq,lb,ub,options);
optimalRedesign = optimum(1:length(costX));
optimalBridgeBuy = optimum(end-length(costZ)+1:end);

numRed = sum(sum(optimalRedesign)-numberOfComponents*2);


%if the optimum obsolescence mitigation plan must be plotted, set
%   shouldPlot = 1
if shouldPlot == 1;
fig=figure(1);
set(gcf,'units','inches','position',...
```

```
[1 1 6 ((numberOfComponents + 2/3) * .25 + .4792)])
set(gcf,'units','inches','position',...
[1 1 6 ((numberOfComponents + 1/3) * .25 + 0.4792)])
ax=gca;
clf

for i = 1:numberOfComponents
BBdates = bridgeBuyDates([1 2],and(bridgeBuyDates(3,:)==i,...
logical(optimalBridgeBuy)'));
RedDates = redesignDateMatrix(1,and(redesignDateMatrix(4,:)==i,...
logical(optimalRedesign)'));

rectangle('Position',[IOC,i-1/3,endOfLife-IOC,2/3],...
'FaceColor',[.75 .75 .75]);
hold on
for j=1:size(BBdates,2)
rectangle('Position',[BBdates(1,j),i-1/3,...
BBdates(2,j)-BBdates(1,j),2/3],'FaceColor','red');
end
hold on
plot(BBdates(1,:),ones(1,length(BBdates))* i, 'kx','markersize',10)
plot(RedDates(1,:),ones(1,length(RedDates))* i,'k.','markersize',20)
end
axis([IOC endOfLife 0 numberOfComponents+1])
axis([IOC endOfLife 1/3 numberOfComponents+2/3])
set(gca,'YTick',1:numberOfComponents)

labelList = {};
for j = 1:numberOfComponents
labelList{j} = ['Component ' num2str(j)'];
end
labelList{j+1} = '';
```

```
set(gca,'YTickLabel',labelList);
title(['Certification Cost = $' num2str(NRE_Cost)])

for i=1:length(freeCertDates)
date = freeCertDates(i);
plot([date date], [0 numberOfComponents + 1], 'b-')
end

set(findall(gcf,'-property','FontSize'),...
'FontName','Times New Roman','FontSize',12)

end


end
```

APPENDIX C

PARTIAL DERIVATIVES

To better understand the relationship between the obsolescence cost (c_{obs}) and its input parameters, this appendix presents the partial derivatives of the cost formulation presented in Equation 21 — namely $\frac{\partial c_{obs}}{\partial c_i^{rep}}$, $\frac{\partial c_{obs}}{\partial c_i^{H}}$, $\frac{\partial c_{obs}}{\partial q_i}$, $\frac{\partial c_{obs}}{\partial c_i^{red}}$, $\frac{\partial c_{obs}}{\partial c^{NRE}}$, and $\frac{\partial c_{obs}}{\partial c_i^{LOT}}$. These derivatives are derived below using Equation 24 found in Appendix D.

C.1 Computing $\frac{\partial c_{obs}}{\partial c_i^{rep}}$

The derivative of c_{obs} with respect to c_i^{rep} is computed straightforwardly from Equation 24 as follows:

$$\frac{\partial c_{obs}}{c_i^{rep}} = \left[\frac{q_i}{r}\left(e^{-rt_{start}} - e^{-rt_{end}}\right)\right] e^{rt_{base}}$$

Note that since $t_{start} < t_{end}$ and $r \geq 0$, then by definition $(e^{-rt_{start}} - e^{-rt_{end}}) \geq 0$. Furthermore, since $t_{base} \geq 0$, it must always be true that $e^{rt_{base}} \geq 1$. Finally, since $q_i \geq 0$, it follows that $\frac{\partial c_{obs}}{\partial c_i^{rep}} \geq 0$. Thus, for all q_i, r, t_{start}, t_{end}, and t_{base} in their domains, c_{obs} is non-decreasing with increasing c_i^{rep}.

C.2 Computing $\frac{\partial c_{obs}}{\partial c_i^{H}}$

Starting with equation 24, the derivative of c_{obs} with respect to c_i^{H} is computed as follows:

$$\frac{\partial c_{obs}}{\partial c_i^{H}} = \frac{-q_i e^{-r(t_{start}-t_{base})} t_{start}}{r} + \frac{q_i e^{-r(t_{start}-t_{base})} t_{end}}{r} - \frac{q_i e^{-r(t_{start}-t_{base})}}{r^2} + \frac{q_i e^{-r(t_{end}-t_{base})}}{r^2}$$

$$= \left[\frac{q_i e^{-rt_{start}}}{r}\left(t_{end} - t_{start}\right) + \frac{q_i}{r^2}\left(e^{-rt_{end}} - e^{-rt_{start}}\right)\right] e^{rt_{base}}$$

For simplicity, let $\Delta t = t_{end} - t_{start}$. Then

$$\frac{\partial c_{obs}}{\partial c_i^H} = \left[\frac{q_i e^{-rt_{start}}}{r} \Delta t + \frac{q_i e^{-rt_{start}}}{r^2} \left(e^{-r\Delta t} - 1 \right) \right] e^{rt_{base}}$$

$$= \left\{ \frac{q_i e^{-rt_{start}}}{r} \left[\Delta t + \frac{1}{r} \left(e^{-r\Delta t} - 1 \right) \right] \right\} e^{rt_{base}}$$

Note that since $q_i \geq 0$, $t_{start} \geq 0$, $r \geq 0$, and $t_{base} \geq 0$, it must always be true that $\left[\frac{q_i e^{-rt_{start}}}{r} \right] e^{rt_{base}} \geq 0$. Thus, if it can be shown that $\left[\Delta t + \frac{1}{r} \left(e^{-r\Delta t} - 1 \right) \right] \geq 0$ in its domain, then it must be true that $\frac{\partial c_{obs}}{\partial c_i^H} \geq 0$ in its domain. To show this, we assume it to be true and proceed as follows:

$$\begin{aligned} \Delta t + \frac{1}{r} \left(e^{-r\Delta t} - 1 \right) &\geq 0 \\ \frac{1}{r} \left(e^{-r\Delta t} - 1 \right) &\geq -\Delta t \\ e^{-r\Delta t} - 1 &\geq -r\Delta t \\ e^{-r\Delta t} &\geq -r\Delta t + 1 \end{aligned}$$

This is true for all $r\Delta t \geq 0$, thus, c_{obs} is non-decreasing for increasing c_i^H.

C.3 Computing $\frac{\partial c_{obs}}{\partial q_i}$

$\frac{\partial c_{obs}}{\partial q_i}$ is computed straightforwardly from Equation 24 in Appendix D as follows:

$$\begin{aligned} \frac{\partial c_{obs}}{\partial q_i} = \Bigg[& \frac{-c_i^{rep}}{r} \left(e^{-rt_{end}} - e^{-rt_{start}} \right) - \frac{c_i^H t_{end}}{r} \left(e^{-rt_{end}} - e^{-rt_{start}} \right) \\ & + \frac{c_i^H}{r^2} \left(e^{-rt_{end}} \left(rt_{end} + 1 \right) - e^{-rt_{start}} \left(rt_{start} + 1 \right) \right) \\ & + c_i^{LOT} \left(t_{end} - t_{start} \right) e^{-rt_{start}} \Bigg] e^{rt_{base}} \end{aligned}$$

Note that since $c_i^{rep} \geq 0$, $r \geq 0$, and $t_{end} > t_{start}$, it must be true that

$$\frac{-c_i^{rep}}{r} \left(e^{-rt_{end}} - e^{-rt_{start}} \right) \geq 0$$

Similarly, since c_i^{LOT}, it must be true that

$$c_i^{LOT}\left(t_{end}-t_{start}\right)e^{-rt_{start}} \geq 0$$

Finally, since $t_{base} > 0$, it must be true that $e^{rt_{base}} \geq 1$. Therefore, if it can be shown that

$$\frac{c_i^H t_{end}}{r}\left(e^{-rt_{end}}-e^{-rt_{start}}\right)+\frac{c_i^H}{r^2}\left(e^{-rt_{end}}\left(rt_{end}+1\right)-e^{-rt_{start}}\left(rt_{start}+1\right)\right) \geq 0$$

for all parameter values in their domains, then it must be true that $\frac{\partial c_{obs}}{\partial q_i} \geq 0$ and therefore c_{obs} is non-decreasing with increasing q_i. This is accomplished using the following steps:

Start by multiplying both sides by $\frac{r}{c_i^H}$ to yield

$$t_{end}\left(e^{-rt_{end}}-e^{-rt_{start}}\right)+\frac{1}{r}\left(e^{-rt_{end}}\left(rt_{end}+1\right)-e^{-rt_{start}}\left(rt_{start}+1\right)\right) \geq 0$$

$$-t_{end}e^{-rt_{end}}+t_{end}e^{-rt_{start}}+\frac{1}{r}\left(e^{-rt_{end}}\left(rt_{end}+1\right)-e^{-rt_{start}}\left(rt_{start}+1\right)\right) \geq 0$$

$$-t_{end}e^{-rt_{end}}+t_{end}e^{-rt_{start}}+t_{end}e^{-rt_{end}}+\frac{1}{r}e^{-rt_{end}}-t_{start}e^{-rt_{start}}-\frac{1}{r}e^{-rt_{start}} \geq 0$$

Next, cancel terms and continue as follows:

$$\cancel{-t_{end}e^{-rt_{end}}}+t_{end}e^{-rt_{start}}+\cancel{t_{end}e^{-rt_{end}}}+\frac{1}{r}e^{-rt_{end}}-t_{start}e^{-rt_{start}}-\frac{1}{r}e^{-rt_{start}} \geq 0$$

$$t_{end}e^{-rt_{start}}+\frac{1}{r}e^{-rt_{end}}-t_{start}e^{-rt_{start}}-\frac{1}{r}e^{-rt_{start}} \geq 0$$

$$e^{-rt_{start}}\left(t_{end}-t_{start}-\frac{1}{r}\right)+\frac{1}{r}e^{-rt_{start}} \geq 0$$

Next, multiply both sides by $e^{rt_{start}}$ to yield:

$$\left(t_{end} - t_{start} - \frac{1}{r}\right) + \frac{1}{r}e^{-rt_{end}+rt_{start}} \geq 0$$

$$t_{end} - t_{start} - \frac{1}{r} + \frac{1}{r}e^{-r(t_{end}-t_{start})} \geq 0$$

As before, let $\Delta t = t_{end} - t_{start}$:

$$\Delta t - \frac{1}{r} + \frac{1}{r}e^{-r\Delta t} \geq 0$$

$$\Delta t + \frac{1}{r}\left(-1 + e^{-r\Delta t}\right) \geq 0$$

$$\frac{1}{r}\left(-1 + e^{-r\Delta t}\right) \geq -\Delta t$$

$$e^{-r\Delta t} \geq -r\Delta t + 1$$

Note that $e^{-r\Delta t} \geq -r\Delta t + 1$ is true for all r and Δt in their domains, thus $\frac{\partial c_{obs}}{\partial q_i} \geq 0$ and c_{obs} is non-decreasing with increasing q_i for all parameters in their domains.

C.4 Computing $\frac{\partial c_{obs}}{\partial c_{red}}$

The partial of c_{obs} with respect to c_i^{red} can be computed straightforwardly from Equation 24 as follows:

$$\frac{\partial c_{obs}}{\partial c_i^{red}} = \left[e^{-rt_{start}}\right] e^{rt_{base}}$$

Note that since $r \geq 0$, $t_{start} > 0$, and $t_{base} > 0$, it must be true that $\frac{\partial c_{obs}}{\partial c_i^{red}} \geq 0$ for all parameters in their domains, and therefore c_{obs} is non-decreasing with increasing c_i^{red}.

C.5 Computing $\frac{\partial c_{obs}}{\partial c_i^{LOT}}$

The partial of c_{obs} with respect to c_i^{LOT} can be computed straightforwardly from Equation 24 as follows:

$$\frac{\partial c_{obs}}{\partial c_i^{LOT}} = \left[\left(t_{end} - t_{start} \right) q_i e^{-rt_{start}} \right] e^{rt_{base}}$$

Note that since $t_{end} > t_{start} > 0$, $q_i \geq 0$, $r \geq 0$, and $t_{base} > 0$, it must be true that $\frac{\partial c_{obs}}{\partial c_i^{LOT}} \geq 0$ for all parameters in their domains, and therefore c_{obs} is non-decreasing with increasing c_i^{LOT}.

C.6 Computing $\frac{\partial c_{obs}}{\partial c^{NRE}}$

The partial of c_{obs} with respect to c_i^{NRE} can be computed straightforwardly from Equation 24 as follows:

$$\frac{\partial c_{obs}}{\partial c^{NRE}} = \left[e^{-rt_{start}} \right] e^{rt_{base}}$$

Note that since $r \geq 0$, $t_{start} > 0$, and $t_{base} > 0$, it must be true that $\frac{\partial c_{obs}}{\partial c^{NRE}} \geq 0$ for all parameters in their domains, and therefore c_{obs} is non-decreasing with increasing c^{NRE}.

APPENDIX D

EXPANDED FORM OF COST FUNCTION

Equation 21 is repeated below:

$$c_i^{obs} = \overbrace{\int_{t_{start}}^{t_{end}} \left(c_i^{rep} + c_i^H \left(t_{end} - s\right)\right) q_i e^{-r(s-t_{base})} ds}^{c_i^A} + \underbrace{c_i^{red} e^{-r(t_{start}-t_{base})} + c_i^{NRE} e^{-r(t_{start}-t_{base})} + c_i^{LOT}(t_{end} - t_{start}) q_i e^{-r(t_{start}-t_{base})}}_{c_i^B}$$

Part A of the cost function, c^A can be expanded as follows:

$$c_i^A = \int_{t_{start}}^{t_{end}} c_i^{rep} q_i e^{-r(s-t_{base})} ds + \int_{t_{start}}^{t_{end}} c_i^H q_i (t_{end} - s) e^{-r(s-t_{base})} ds$$

$$= c_i^{rep} q_i e^{rt_{base}} \int_{t_{start}}^{t_{end}} e^{-rs} ds + c_i^H q_i t_{end} e^{rt_{base}} \int_{t_{start}}^{t_{end}} e^{-rs} ds - c_i^H q_i e^{rt_{base}} \int_{t_{start}}^{t_{end}} s e^{-rs} ds$$

$$= c_i^{rep} q_i e^{rt_{base}} \left(\frac{1}{-r} e^{-rt_{end}} - \frac{1}{-r} e^{-rt_{start}}\right) + c_i^H q_i e^{rt_{base}} t_{end} \left(\frac{1}{-r} e^{-rt_{end}} - \frac{1}{-r} e^{-rt_{start}}\right) - c_i^H q_i e^{rt_{base}} \left(\frac{1}{(-r)^2} e^{-rt_{end}} (-rt_{end} - 1) - \frac{1}{(-r)^2} e^{-rt_{start}} (-rt_{start} - 1)\right)$$

Distributing terms gives:

$$c_i^A = \frac{-c_i^{rep} q_i e^{rt_{base}}}{r} e^{-rt_{end}} + \frac{c_i^{rep} q_i e^{rt_{base}}}{r} e^{-rt_{start}} - \frac{c_i^H q_i t_{end} e^{rt_{base}}}{r} e^{-rt_{end}} + \frac{c_i^H q_i t_{end} e^{rt_{base}}}{r} e^{-rt_{start}} + \frac{c_i^H q_i t_{end} e^{rt_{base}}}{r} e^{-rt_{end}} + \frac{c_i^H q_i e^{rt_{base}}}{r^2} e^{-rt_{end}} - \frac{c_i^H q_i t_{start} e^{rt_{base}}}{r} e^{-rt_{start}} - \frac{c_i^H q_i e^{rt_{base}}}{r^2} e^{-rt_{start}}$$

Canceling terms gives:

$$
\begin{aligned}
c_i^A =& \frac{-c_i^{rep} q_i e^{rt_{base}}}{r} e^{-rt_{end}} + \frac{c_i^{rep} q_i e^{rt_{base}}}{r} e^{-rt_{start}} - \cancel{\frac{c_i^H q_i t_{end} e^{rt_{base}}}{r} e^{-rt_{end}}} \\
&+ \frac{c_i^H q_i t_{end} e^{rt_{base}}}{r} e^{-rt_{start}} + \cancel{\frac{c_i^H q_i t_{end} e^{rt_{base}}}{r} e^{-rt_{end}}} + \frac{c_i^H q_i e^{rt_{base}}}{r^2} e^{-rt_{end}} \\
&- \frac{c_i^H q_i t_{start} e^{rt_{base}}}{r} e^{-rt_{start}} - \frac{c_i^H q_i e^{rt_{base}}}{r^2} e^{-rt_{start}}
\end{aligned}
$$

$$
\begin{aligned}
=& \frac{-c_i^{rep} q_i e^{rt_{base}}}{r} e^{-rt_{end}} + \frac{c_i^{rep} q_i e^{rt_{base}}}{r} e^{-rt_{start}} + \frac{c_i^H q_i t_{end} e^{rt_{base}}}{r} e^{-rt_{start}} \\
&+ \frac{c_i^H q_i e^{rt_{base}}}{r^2} e^{-rt_{end}} - \frac{c_i^H q_i t_{start} e^{rt_{base}}}{r} e^{-rt_{start}} - \frac{c_i^H q_i e^{rt_{base}}}{r^2} e^{-rt_{start}}
\end{aligned}
$$

Thus, the final expanded form of c_i^{obs} is:

$$
\begin{aligned}
c_i^{obs} =& c_i^A + c_i^B \\
=& \frac{-c_i^{rep} q_i e^{rt_{base}}}{r} e^{-rt_{end}} + \frac{c_i^{rep} q_i e^{rt_{base}}}{r} e^{-rt_{start}} + \frac{c_i^H q_i t_{end} e^{rt_{base}}}{r} e^{-rt_{start}} \\
&+ \frac{c_i^H q_i e^{rt_{base}}}{r^2} e^{-rt_{end}} - \frac{c_i^H q_i t_{start} e^{rt_{base}}}{r} e^{-rt_{start}} - \frac{c_i^H q_i e^{rt_{base}}}{r^2} e^{-rt_{start}} \\
&+ c_i^{red} e^{rt_{base}} e^{-rt_{start}} + c_i^{NRE} e^{rt_{base}} e^{-rt_{start}} + c_i^{LOT} (t_{end} - t_{start}) q_i e^{rt_{base}} e^{-rt_{start}}
\end{aligned}
\tag{24}
$$

APPENDIX E

PROOF OF MILP OPTIMALITY

A proof was presented in Chapter 4 that shows that the CASCADE MILP formulation yields optimal obsolescence mitigation plans which are as-good-as or better-than the original MILP formulation presented in § 2.6.6. This appendix provides similar proofs which show that the CASCADE MILP produces optimal obsolescence mitigation plans which are always as-good-as or better-than two common reactive strategies.

To prove that the new MILP formulation yields an optimal strategy equal to or better-than the three baseline strategies ("Last Time Buy," "Bridge Buy," and the Original MILP), it is helpful to first present each of these strategies in the standard form of a Mixed Integer Linear Programming problem. Section E.1 proves that the CASCADE MILP out-performs the Reactive "Last-Time Buy" strategy, while Section E.2 proves that the CASCADE MILP out-performs the Reactive "Bridge-Buy" strategy. Both proofs start by describing the respective reactive strategy using the same mathematical form as the CASCADE MILP, and then showing that the CASCADE MILP produces a larger feasible space.

E.1 "Last Time Buy" Formulation

Under the "Last Time Buy" strategy, each component is allowed to go obsolete exactly once, at which point a single life-of-type buy is made which stretches to the end of life of the system. No redesigns are performed, and the system never incurs a re-qualification cost penalty. Using the same naming conventions as the new MILP formulation developed in § 3.2, the constraints for the "Last Time Buy" strategy would be as follows:

$$\text{for each } i\text{: } z_{i,d_{EOL},d_{first}} = 1, \text{ where } d_{first} = d_{IOC} + \Delta_i \tag{25a}$$

Constraint 25a stipulates that for each component, i, a LOT buy occurs at Δ_i years after d_{IOC}, and lasts until the end of life of the system, d_{EOL}. Thus, any $\left\{x_{i,d_{red}}, y_{d_{red}}, z_{i,d_{red},d_{bridge}}\right\}$ that satisfies Constraint 25a of the Last-Time Buy formulation will also satisfy Constraint 18a of the CASCADE MILP.

$$\text{For each } i,\ d_{red} \in D_{red}\text{: } x_{i,d_{red}} = 0 \tag{25b}$$

Constraint 25b says that no redesigns can occur for any component. Consequently, the system never needs to be re-qualified. Thus, any $\left\{x_{i,d_{red}}, y_{d_{red}}, z_{i,d_{red},d_{bridge}}\right\}$ that satisfies Constraint 25b of the Last-Time Buy formulation will always also satisfy Constraint 18b of the CASCADE MILP.

$$\text{For each } d_{red} \in D_{red}\text{: } y_{d_{red}} = 0 \tag{25c}$$

Constraint 25c simply says that no re-qualification cost is ever incurred at any redesign date. This is valid under the CASCADE MILP in the unique instance where no redesigns are selected. Thus, any $\left\{x_{i,d_{red}}, y_{d_{red}}, z_{i,d_{red},d_{bridge}}\right\}$ that satisfies Constraint 25c of the Last-Time Buy formulation will always also satisfy Constraint 18d of the CASCADE MILP.

Since obsolescence cost is computed the same way (the objective function is the same as the CASCADE MILP), regardless of which strategy is used, Constraints 25a through 25c can be combined to form the MILP version of the "Last Time Buy" baseline as follows:

Minimize

$$C_{obs} = \sum_{i=1}^{m} \left(\sum_{d_{red} \in D_{red}} C^{R}_{i,d_{red}} x_{i,d_{red}} \right) + \sum_{d_{red} \in D_{red}} C^{NRE}_{d_{red}} y_{d_{red}} + \sum_{i=1}^{m} \left(\sum_{d_{red} \in D_{red}} \left(\sum_{d_{bridge} \in D_{bridge}} C^{B}_{i,d_{red},d_{bridge}} z_{i,d_{red},d_{bridge}} \right) \right)$$

S.T.

$$\begin{aligned}
&\text{for each } i\text{: } z_{i,d_{EOL},d_{first}} = 1, \text{ where } d_{first} = d_{IOC} + \Delta_i \\
&\text{For each } i,\ d_{red} \in D_{red}\text{: } x_{i,d_{red}} = 0 \\
&\text{For each } d_{red} \in D_{red}\text{: } y_{d_{red}} = 0 \\
&\text{For each } i,\ d_{red} \in D_{red}\text{: } x_{i,d_{red}} \in \{0,1\} \\
&\text{For each } i,\ d_{red} \in D_{red}\text{: } y_{d_{red}} \in \{0,1\} \\
&\text{For each } i,\ d_{red} \in D_{red},\ d_{bridge} \in D_{bridge} : z_{i,d_{red},d_{bridge}} \in \{0,1\}
\end{aligned} \tag{26}$$

Since the objective functions are identical, and since all $\left\{x_{i,d_{red}}, y_{d_{red}}, z_{i,d_{red},d_{bridge}}\right\}$ which satisfy the Last-Time Buy formulation (Equation 26) also satisfy the CASCADE MILP (Equation 19) — but not the other way around — it must be true that the set of feasible solutions in the Reactive Last-Time Buy baseline represent a proper subset of those of the CASCADE MILP. Therefore, the CASCADE MILP will always produce a design refresh strategy which is as good as, or better than, the Reactive Last-Time Buy baseline.

E.2 "Bridge Buy" Formulation

Under the "Bridge Buy" baseline, each component is allowed to become obsolete, at which point a LOT buy is made which stretches to the date of the next redesign. Using the same naming conventions as the CASCADE MILP formulation developed in § 3.2, the constraints for the "Bridge Buy" strategy would be as follows:

$$\text{For each } i,\ d^*_{red} \in D_{red}\text{: } x_{i,d^*_{red}} = z_{i,d_{next},d^*_{red}+\Delta_i},\ \text{where } d_{next} = \min(\{d_{red} > d^*_{red} + \Delta_i\}) \tag{27a}$$

Constraint 27a says that whenever component i is redesigned at data d^*_{red}, a new LOT buy must be initiated Δ_i years later, and that LOT buy must stretch until the next available redesign date. In short, components are always allowed to become obsolete after each redesign, at which point the shortest-possible LOT buy will occur until the *next* possible redesign date. Thus, any $\left\{x_{i,d_{red}}, y_{d_{red}}, z_{i,d_{red},d_{bridge}}\right\}$ that satisfies Constraint 27a of the Bridge-Buy formulation will also satisfy Constraint 18a of the CASCADE MILP.

$$\text{For each } i,\ d^*_{red} \in D_{red}\text{: } x_{i,d^*_{red}} = z_{i,d_{next},d^*_{red},d_{prev}},\ \text{where } d_{prev} = \max(\{d_{bridge} < d^*_{red}\}) \tag{27b}$$

Constraint 27b compliments Constraint 27a by saying that whenever a redesign occurs at date d^*_{red}, it must be paired with a LOT buy which was initiated at the latest possible date prior to d^*_{red}. Again, put more plainly, redesigns cannot occur unless paired with a LOT buy which commenced *just before* the redesign. Thus, any $\left\{x_{i,d_{red}}, y_{d_{red}}, z_{i,d_{red},d_{bridge}}\right\}$ that satisfies Constraint 27b of the Bridge-Buy formulation will always also satisfy Constraint 18b of the CASCADE MILP.

$$\text{For each } i,\ d^*_{red} \in D_{red}\text{:}$$
$$\sum_{d^*_{red} < d_{red} < d^*_{red}+\Delta_i} x_{i,d_{red}} + \sum_{d_{bridge} \le d^*_{red}+\Delta_i} \left(\sum_{d_{red} > d^*_{red}+\Delta_i} z_{i,d_{red},d_{bridge}} \right) \ge 1 \tag{27c}$$

Constraint 27c says that for each component, i, and within every time period Δ_i, one of the following must be true:

- The component must be redesigned

- A bridging solution must take place
- A previous bridging strategy must be underway which spans the Δ_i period

Put more plainly, constraint 27c says that for each component that faces obsolescence, *at least one* obsolescence mitigation action must be taken. Since constraint 27c is identical to constraint 18c or the CASCADE MILP, any $\left\{x_{i,d_{red}}, y_{d_{red}}, z_{i,d_{red},d_{bridge}}\right\}$ that satisfies Constraint 27c of the Bridge-Buy formulation will always also satisfy Constraint 18c of the CASCADE MILP.

$$\text{For each } d_{red} \in D_{red}\text{:} \quad \sum_{i=1}^{m} x_{i,d_{red}} \leq M y_{d_{red}} \text{ For } M \gg m \tag{27d}$$

Constraint 27d is identical to Constraint 18d of the CASCADE MILP formulation. It says that if *any* redesign takes place at $t = d_{red}$, then the program must incur a recertification cost. Thus, any $\left\{x_{i,d_{red}}, y_{d_{red}}, z_{i,d_{red},d_{bridge}}\right\}$ that satisfies Constraint 27d of the Bridge-Buy formulation will always also satisfy Constraint 18d of the CASCADE MILP.

As before, the cost of obsolescence is computed in the same way as before (i.e. the objective function remains the same), so the MILP formulation of the "Bridge Buy" baseline strategy can be written by combining Constraints 27a through 27c as follows:

Minimize

$$C_{obs} = \sum_{i=1}^{m} \left(\sum_{d_{red} \in D_{red}} C^{R}_{i,d_{red}} x_{i,d_{red}} \right) + \sum_{d_{red} \in D_{red}} C^{NRE}_{d_{red}} y_{d_{red}} + \sum_{i=1}^{m} \left(\sum_{d_{red} \in D_{red}} \left(\sum_{d_{bridge} \in D_{bridge}} C^{B}_{i,d_{red},d_{bridge}} z_{i,d_{red},d_{bridge}} \right) \right)$$

S.T.

$$\begin{aligned}
&\text{For each } i,\ d^{*}_{red} \in D_{red}\text{: } x_{i,d^{*}_{red}} = z_{i,d_{next},d^{*}_{red}+\Delta_i}, \\
&\quad \text{where } d_{next} = \min(\{d_{red} > d^{*}_{red} + \Delta_i\}) \\
&\text{For each } i,\ d^{*}_{red} \in D_{red}\text{: } x_{i,d^{*}_{red}} = z_{i,d_{next},d^{*}_{red},d_{prev}}, \\
&\quad \text{where } d_{prev} = \max(\{d_{bridge} < d^{*}_{red}\}) \\
&\text{For each } i,\ d^{*}_{red} \in D_{red}\text{: } \sum_{d^{*}_{red} < d_{red} < d^{*}_{red}+\Delta_i} x_{i,d_{red}} \\
&+ \sum_{d_{bridge} \leq d^{*}_{red}+\Delta_i} \left(\sum_{d_{red} > d^{*}_{red}+\Delta_i} z_{i,d_{red},d_{bridge}} \right) \geq 1 \\
&\text{For each } d_{red} \in D_{red}\text{: } \sum_{i=1}^{m} x_{i,d_{red}} \leq M y_{d_{red}} \text{ For } M \gg m \\
&\text{For each } i,\ d_{red} \in D_{red}\text{: } x_{i,d_{red}} \in \{0,1\} \\
&\text{For each } i,\ d_{red} \in D_{red}\text{: } y_{d_{red}} \in \{0,1\} \\
&\text{For each } i,\ d_{red} \in D_{red},\ d_{bridge} \in D_{bridge} : z_{i,d_{red},d_{bridge}} \in \{0,1\}
\end{aligned} \tag{27e}$$

Since the objective functions are identical, and since all $\left\{x_{i,d_{red}}, y_{d_{red}}, z_{i,d_{red},d_{bridge}}\right\}$ which satisfy the Bridge-Buy formulation (Equation 27e) also satisfy the CASCADE MILP (Equation 19) — but not the other way around — it must be true that the set of feasible solutions in the Reactive Bridge-Buy baseline represent a proper subset of those of the CASCADE MILP. Therefore, the CASCADE MILP will always produce a design refresh strategy which is as good as, or better than, the Reactive Bridge-Buy baseline.

APPENDIX F

COST OF RUGGEDIZATION

As §2.2 alluded to, COTS components are not generally designed to meet the demands of a military environment, and may therefore not be suitable for use by the military [32, 42]. One way to deal with this (apart from developing custom components) is to modify or ruggedize existing COTS components. On the other hand, modifying COTS components brings its own set of challenges. Some programs fail, in fact, because of the assumption that commercial components could simply be modified to meet requirement[20, 74, 102]. In order to demonstrate the system-level trade-off between ruggedization, reliability, and cost, it is important to first understand the impact of ruggedization at the component level.

This appendix presents various relationships between reliability and cost which will be useful in a system-level analysis. There is, however, a virtual absence of reliable data in the literature that relates ruggedization to cost. To account for this absence of data, this appendix focuses on the relationship between system *reliability* and cost — a subject for which data does exist (albeit sparsely) in the literature [15]. Generally speaking, an increase in reliability (by way of ruggedization) can be expected to increase RDT&E, as discussed in §F.2. Meanwhile, investing in more-durable components means fewer repairs (although those repairs may be more costly), resulting in lower O&S costs. These notional trends are shown in Figure 79[5, 55, 96].

Component modification or ruggedization can occur in two basic ways. Components can be purchased which have been ruggedized by the manufacturer, or un-ruggedized components can be purchased and ruggedized by the end-user. To account for the differences, this appendix is split into two sections: Section F.1 explores the

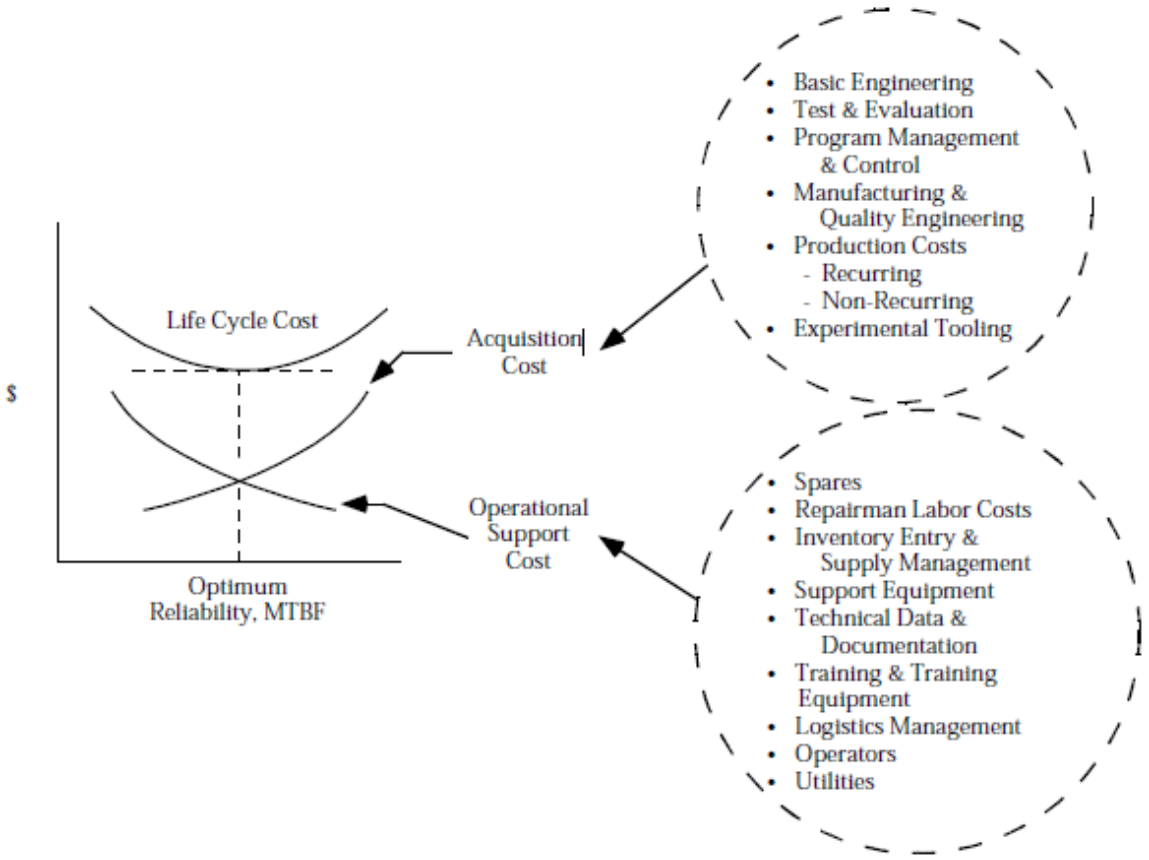

Figure 79: Notional life cycle cost vs. reliability. Note that acquisition cost is monotonically increasing with increasing reliability[5].

relationship between component cost and reliability. Section F.2 explores the relationship between component *redesign* cost and reliability.

F.1 Component Cost vs. Reliability

While little published data exists that explicitly relates component cost to reliability, general trends can be taken from anecdotal cost data. For instance, a RAND study found that tires rated for 20,000 miles cost only a few percent more than those rated for 15,000 miles, while tires rated for 40,000 miles cost only 15% more than those[15]. Despite this, a 75,000 mile tire would cost multiple times that of a 40,000 mile tire, suggesting a point of diminishing returns. They found that this trend relating cost and reliability is consistent with most systems: a relatively-flat portion followed by a

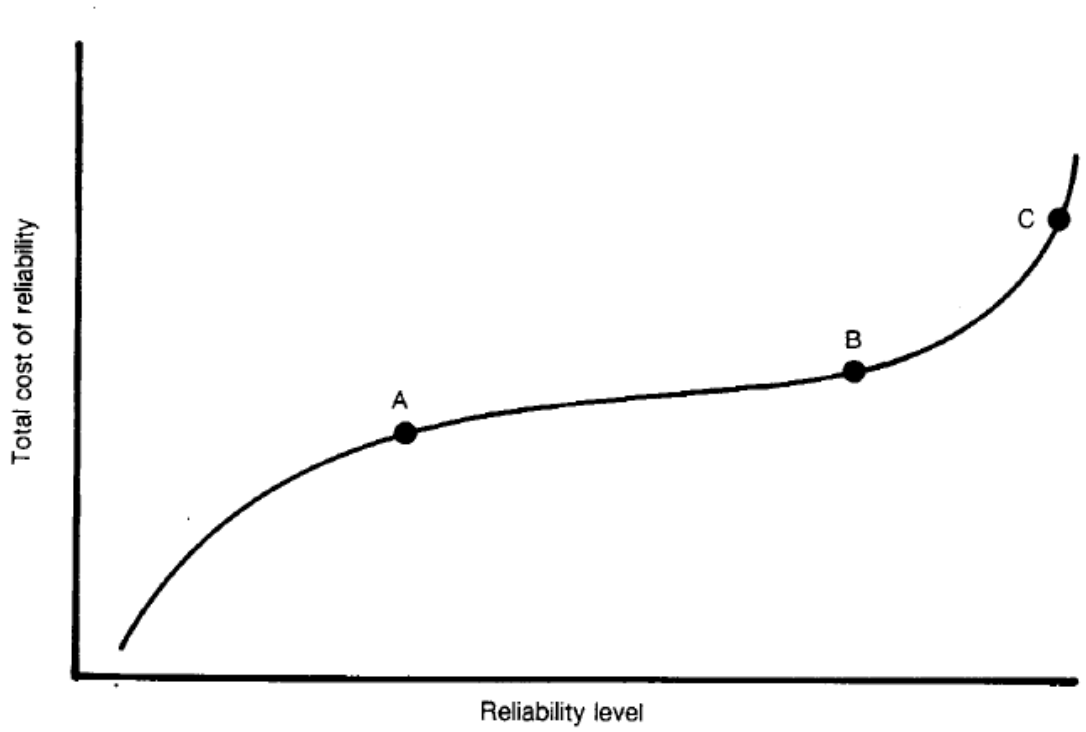

Figure 80: Notional cost of reliability. Point A represents most military systems, for which reliability has historically been a relatively low priority. Point C, on the other hand, represents highly-reliable systems such as spacecraft. Note that between A and B, large changes in reliability can be had for little increase in cost, whereas a large investment is needed to move between B and C[15].

steeply rising sector, as shown in Figure 80 [15].

It is important to consider, however, that while this trend may be representative of whole systems, it does not necessarily represent the cost vs reliability of individual components. For this, anecdotal cost data can provide guidance. A consumer tablet producer reports that its consumer-grade model sells for $2,200 with a failure rate of 25% per year. Meanwhile, its ruggedized tablets sell for $3,300, with an improved failure rate of 4% per year[60, 72]. Thus, at only a 36% cost increase per component, reliability is increased more than six-fold. Similarly, another company reports that the costs for handheld computers range from $472 for consumer-grade, to $879 for ruggedized[50]. While these data provide general guidance in terms of orders-of-magnitude, it is important to consider that they are presented for marketing purposes,

so caution is advised when using these values.

Despite ambiguity as to the exact shape of the cost vs reliability function, Aggarwal and Gupta define four requirements for such a relationship as follows[13, 14]:

1. Cost is a monotonically increasing function of component reliability
2. The cost of a high reliability component is very high
3. The cost of a low reliability component is very low
4. The derivative of cost with respect to reliability is a monotonically increasing function of reliability

F.2 Redesign Cost vs. Reliability

If un-ruggedized components are purchased and later modified by the end-user, then the unit cost of the components does not increase. Instead, the cost of modification manifests itself as an additional non-recurring engineering (NRE) or redesign cost. Little information exists in the literature that directly links redesign cost to ruggidization levels for COTS components. Therefore, as before, reliability improvement programs for large systems are used in place of component-level reliability improvements, with the expectation that the general trends will apply at both levels.

Perhaps the most compelling evidence relating NRE cost and reliability comes from the RAND corporation, which compares the reliability-related RDT&E expenditures for various systems to their corresponding changes in reliability. These data, presented in Table 19, suggest that relatively small investments in RDT&E expenditures can have a great impact on system reliability. It should be noted, however, that these data do not consider the full RDT&E expenditures — simply those for which the authors attribute to reliability growth.

Table 19: Percent change in reliability for a given percent increase in reliability-focused RDT&E expenditures[15]. Note that reliability is measured in MTBF.

Case	Change in Reliability (%)	Increase in RDT&E Expenditure (%)
CH-47D	114	0-70
Phalanx	191	5.4
Minuteman	1400-1500	50
Carousel IV	400-500	180
Carousel IV	67-100	80
F-18	100	12
Naval Electronics	228	76

REFERENCES

[1] "Glossary of Defense Acquisition Acronyms and Terms."

[2] "Fruitycake: A Yummy MIL-F-14499f," *Time*, vol. 127, p. 73, Jan. 1986.

[3] "Defense Aircraft Investments: Major Program Commitments Based on Optimistic Budget Projections," Testimony GAO/T-NSIAD-97-103, Government Accounting Office, Mar. 1997.

[4] "Naval Sea Systems Command Commercial Off-The-Shelf and Non-Developmental Items Handbook," June 2000.

[5] "Electronic Reliability Design Handbook," handbook, Department of Defense, 2003.

[6] "Defense Agencies Meet Readiness Challenges with Commercial Off the Shelf (COTS)-Based Systems," white Paper, Cisco Systems, 2005.

[7] "Diminishing Manufacturing Sources and Material Shortages (DMSMS) Guidebook," Apr. 2005.

[8] "Survey of Model-Based Systems Engineering (MBSE) Methodologies," June 2008. INCOSE-TD-2007-003-01.

[9] "Circular A-94 Appendix C," Dec. 2014.

[10] "DoD Directive 5000.02 - The Defense Acquisition Guidebook," Jan. 2015.

[11] "Economic Analysis for Decision-Making," Sept. 2015.

[12] Act, Federal Acquisition Streamlining, "Public Law 103-355," *Washington, DC*, vol. 5, 1994.

[13] Adamantios, M., "Reliability Allocation and Optimization for Complex Systems," *Annual Reliability and Maintainability Symposium*, 2000.

[14] Aggarwal, K. and Gupta, J., "On Minimizing the Cost of Reliable Systems," *Reliability, IEEE Transactions on*, vol. 24, no. 3, p. 205, 1975.

[15] Alexander, A., "The Cost and Benefits of Reliability in Military Equipment," tech. rep., Dec. 1988.

[16] Ambler, S. W., "Disciplined Agile Software Development: Definition," 2012.

[17] Arena, M. V., ed., *Why has the cost of Navy ships risen?: a macroscopic examination of the trends in U.S. Naval ship costs over the past several decades*. Santa Monica, CA: RAND, 2006.

[18] Arena, M. V., *Why has the cost of fixed-wing aircraft risen?: a macroscopic examination of the trends in U.S. military aircraft costs over the past several decades*. Santa Monica, CA: Rand, National Defense Research Institute and Project Air Force, 2008.

[19] Barbacci, M. R., Ellison, R. J., Lattanze, A., Stafford, J., Weinstock, C. B., and Wood, W., "Quality attribute workshops," 2002.

[20] Boehm, B. and Abts, C., "COTS integration: Plug and Pray?," *Computer*, vol. 32, no. 1, pp. 135–138, 1999.

[21] Bradley, J. R. and Guerrero, H. H., "Product Design for Life-Cycle Mismatch," *Production and Operations Management*, vol. 17, pp. 497–512, Sept. 2008.

[22] Brown, B., *Introduction to defense acquisition management*. Government Printing Office, 2010.

[23] Brownsword, L., Oberndorf, T., and Sledge, C. A., "Developing new processes for COTS-based systems," *IEEE Software*, vol. 17, no. 4, pp. 48–55, 2000.

[24] Burge, D. S., "A Functional Approach to Quality Function Deployment," Jan. 2007.

[25] Calloni, D. B., "MIL-SPEC vs. COTS Standards: Necessary Harmony for Affordable Multilevel Secure Architectures," Feb. 2006.

[26] Cattani, K. D. and Souza, G. C., "Good buy? Delaying end-of-life purchases," *European Journal of Operational Research*, vol. 146, no. 1, pp. 216–228, 2003.

[27] Conforti, M., Cornujols, G., and Zambelli, G., *Integer Programming*, vol. 271 of *Graduate Texts in Mathematics*. Cham: Springer International Publishing, 2014.

[28] CONGRESS, UN, "Clinger-Cohen Act of 1996," *Public Law*, 1996.

[29] Connor, K. and Dryden, J., *New approaches to defense inflation and discounting*. Santa Monica, California: RAND, 2013.

[30] Defense Acquisition University, *Defense Acquisition Gudebook*. Defense Acquisition University, Sept. 2013.

[31] Deitz, D., Eveleigh, T. J., Holzer, T. H., and Sarkani, S., "Improving Program Success Through Systems Engineering Tools in Pre-Milestone B Acquisition Phase," tech. rep., DTIC Document, 2013.

[32] Demko, E., "Commercial-Off-The Shelf (COTS): A Challenge To Military Equipment Reliability," in *Annual Reliability and Maintainability Symposium*, pp. 7–12, IEEE, 1996.

[33] Dr. Carlo Kopp, "COTS - Revolution, Evoltion or Devolution?," *Defence Today*, June 2011.

[34] Engelberg, S., "Pentagon Plans $2.3 Billion Saving in '91 by Cutting 16,000 Jobs," *The New York Times*, Jan. 1990.

[35] Eskew, H. L., "Aircraft cost growth and development program length: Some augustinian propositions revisited," tech. rep., DTIC Document, 2000.

[36] Feng, D., Singh, P., and Sandborn, P., "Lifetime buy optimization to minimize lifecycle cost," in *Proceedings of the 2007 aging aircraft conference*, 2007.

[37] Ferguson, J., "Crouching dragon, hidden software: software in DoD weapon systems," *IEEE Software*, vol. 18, no. 4, pp. 105–107, 2001.

[38] Foucher, B., Kennedy, R., Kelkar, N., Ranade, Y., Govind, A., Blake, W., Mathur, A., and Solomon, R., "Why a new parts selection and management program?," *Components, Packaging, and Manufacturing Technology, Part A, IEEE Transactions on*, vol. 21, no. 2, pp. 375–382, 1998.

[39] G. Zell Porter, "An Economic Method for Evaluating Electronic Component Obsolescense Solutions."

[40] Gallego, G., "1 The Newsvendor Problem," 1995.

[41] Gansler, J. S. and Lucyshyn, W., "Commercial-Off-The-Shelf (COTS): Doing it Right,"

[42] Gansler, J. S. and Lucyshyn, W., "Commercial-Off-The-Shelf (COTS): Doing It Right," tech. rep., University of Maryland,School of Public Policy,Center for Public Policy and Private Enterprise,College Park,MD,20742, Sept. 2008.

[43] Genova, K. and Guliashki, V., "Linear integer programming methods and approachesa survey," *Journal of Cybernetics and Information Technologies*, vol. 11, no. 1, 2011.

[44] Gill, C., Cytron, R., and Schmidt, D., "Multiparadigm scheduling for distributed real-time embedded computing," *Proceedings of the IEEE*, vol. 91, pp. 183–197, Jan. 2003.

[45] Haberfellner, R. and Weck, O., "Agile SYSTEMS ENGINEERING versus AGILE SYSTEMS engineering," in *INCOSE International Symposium*, vol. 15, pp. 1449–1465, Wiley Online Library, 2005.

[46] Hanks, C. H., *Reexamining military acquisition reform: are we there yet?* Santa Monica, CA: Rand, 2005.

[47] Hanratty, M., Lightsey, R., and Larson, A., "Open Systems and the Systems Engineering Process," Jan. 1999.

[48] Haskins, C., Forsberg, K., Krueger, M., Walden, D., and Hamelin, D., "Systems engineering handbook," in *INCOSE,*, 2006.

[49] Hill, A. V., "The newsvendor problem," *White Paper*, pp. 57–23, 2011.

[50] Intermec Technologies Corporation, "How Ruggedness Reduces TCO for Mobile COmputers," white Paper, 2009.

[51] Investopedia, LCC, "Economies of Scale," 2015.

[52] J. Ronald Fox, "Defense Acquisition Reform, 19602009 An Elusive Goal," 2011.

[53] Jenab, K., Noori, K., D. Weinsier, P., and Khoury, S., "A dynamic model for hardware/software obsolescence," *International Journal of Quality & Reliability Management*, vol. 31, pp. 588–600, Apr. 2014.

[54] Joseph-Malherbe, S. M., "Agile and lean principles and systems engineering: a synergy?," 2011.

[55] Juran, J. M. and Godfrey, A. B., eds., *Juran's quality handbook*. New York: McGraw Hill, 5th ed ed., 1999.

[56] Kadish, R. T., Abbott, G., Cappuccio, F., Hawley, R., Kern, P., and Kozlowski, D., "Defense acquisition performance assessment report," tech. rep., DTIC Document, 2006.

[57] Kendall, F., "Better Buying Power 3.0," Sept. 2014.

[58] Kennedy, M. R. and Umphress, D. A., "An Agile Systems Engineering Process: The Missing Link?," tech. rep., DTIC Document, 2011.

[59] Kerr, C. G. and Miller, R. W., "A Revolutionary Use of COTS in a Submarine Sonar System," *Journal of Defense Software Engineering*, Nov. 2004.

[60] Krebs, D., "Total Cost of Ownership (TCO) Models for Mobile Computing and Communications Platforms," tech. rep., July 2007.

[61] Krinke, T. and Pai, D., "COTS/ROTS For Mission-Critical Systems."

[62] Kujawski, E., "Unintended consequences of performance specifications for the reliability of military weapon systems," *Systems Engineering*, pp. n/a–n/a, 2009.

[63] Lee, E. A., "Cyber-physical systems-are computing foundations adequate," in *Position Paper for NSF Workshop On Cyber-Physical Systems: Research Motivation, Techniques and Roadmap*, vol. 2, Citeseer, 2006.

[64] Lee, E. A., "Cyber Physical Systems: Design Challenges," pp. 363–369, IEEE, May 2008.

[65] Livingston, H., "GEB1: Diminishing manufacturing sources and material shortages (DMSMS) management practices," in *Proceedings of the DMSMS Conference*, pp. 21–24, 2000.

[66] Lowdermilk, R. and Carey, D. D., "Synthetic Instrumentation Eases ATE Obsolescence Woes."

[67] Lyke, J., "Plug-and-play as an Enabler for Future Systems," in *Proceedings of the AIAA Space Conference*, 2010.

[68] Lyke, J., "Space-Plug-and-Play Avionics (SPA): A Three-Year Progress Report," American Institute of Aeronautics and Astronautics, May 2007.

[69] McDermott, J., Shearer, J., and Tomczykowski, W., "Resolution Cost Factors for Diminishing Manufacturering Sources and Material Shortages," tech. rep., ARINC Incorporated, Feb. 1999.

[70] McNutt, C. J., Vick, R., Whiting, H., and Lyke, J., "Modular Nanosatellites(Plug-and-Play) PnP CubeSat," in *7th Responsive Space Conference*, 2009.

[71] Meng, X., Thornberg, B., and Olsson, L., "Strategic Proactive Obsolescence Management Model," *IEEE Transactions on Components, Packaging and Manufacturing Technology*, vol. 4, pp. 1099–1108, June 2014.

[72] MobileDemand, "Total Cost of Ownership: Rugged Tablets vs. Consumer Tablets," tech. rep.

[73] Mohr, C., "MILITARY PRICE ON COFFEE CITED AS $7,622," *The New York Times*, Sept. 1984.

[74] Money, A. L. and Gansler, J. S., "Commercial Item Acquisition_: Considerations and Lessons Learned," July 2000.

[75] Myers, M., "Navy Races to Keep Aging F/A-18 Hornets Flying Longer," Dec. 2015.

[76] Neff, J. M., Some, R., and Lyke, J., "Lessons Learned in Building a Spacecraft XML Taxonomy and Ontology," *AIAA Infotech@ Aerospace*, 2007.

[77] Office of the Assistant Secretary of Defense, "Maintenance Overview," Feb. 2016.

[78] Ostwald, P. and McLaren, T., *Cost Analysis and Estimating for Engineering and Management*. Upper Saddle River, NJ: Pearson Prentice Hall, 2004.

[79] Pecht, M. G. and Das, D., "Electronic part life cycle," *Components and Packaging Technologies, IEEE Transactions on*, vol. 23, no. 1, pp. 190–192, 2000.

[80] Peter Sandborn, *Cost Analysis of Electronic Systems*, vol. 1. Hackensack, NJ: World Scientific, 2013.

[81] Prabhakar, V. J. and Sandborn, P., "A part total cost of ownership model for long life cycle electronic systems," *International Journal of Computer Integrated Manufacturing*, vol. 25, no. 4-5, pp. 384–397, 2012.

[82] RadiSys, "Is COTS On Track to Meet the Needs of the Military?," May 2011.

[83] Reliability Analysis Center, "Commercial Off-the-Shelf Equipment and Non-Developmental Items."

[84] Sandborn, P., Prabhakar, V., and Ahmad, O., "Forecasting electronic part procurement lifetimes to enable the management of DMSMS obsolescence," *Microelectronics Reliability*, vol. 51, pp. 392–399, Feb. 2011.

[85] Sandborn, P. and Singh, P., "Electronic part obsolescence driven product redesign optimization," in *Proc. FAA/DoD/NASA Aging Aircraft Conference*, 2002.

[86] Sandborn, P., "Beyond reactive thinkingWe should be developing pro-active approaches to obsolescence management too!," *DMSMS Center of Excellence Newsletter*, vol. 2, no. 3, p. 4, 2004.

[87] Sandborn, P., "Strategic management of DMSMS in systems," *DSP Journal*, pp. 24–30, 2008.

[88] Sandborn, P., "Managing Obsolescence Risk," in *Through-life Engineering Services* (Redding, L. and Roy, R., eds.), pp. 341–357, Cham: Springer International Publishing, 2015.

[89] Sandborn, P. A., Mauro, F., and Knox, R., "A Data Mining Based Approach to Electronic Part Obsolescence Forecasting," *IEEE Transactions on Components and Packaging Technologies*, vol. 30, pp. 397–401, Sept. 2007.

[90] Schneider, W., *Buying Commercial: Gaining the Cost/Schedule Benefits for Defense Systems*. DIANE Publishing, 2009.

[91] Schwartz, M., "Defense Acquisitions: How DOD Acquires Weapon Systems and Recent Efforts to Reform the Process," tech. rep., Congressional Research Service, May 2014.

[92] Singh, P. and Sandborn, P., "Obsolescence Driven Design Refresh Planning for Sustainment-Dominated Systems," *The Engineering Economist*, vol. 51, pp. 115–139, July 2006.

[93] Solomon, R., Sandborn, P. A., and Pecht, M. G., "Electronic part life cycle concepts and obsolescence forecasting," *Components and Packaging Technologies, IEEE Transactions on*, vol. 23, no. 4, pp. 707–717, 2000.

[94] Stelzmann, E., "Contextualizing agile systems engineering," *Aerospace and Electronic Systems Magazine, IEEE*, vol. 27, no. 5, pp. 17–22, 2012.

[95] SUBCOMMITTEE, P., "DEFENSE ACQUISITION REFORM: WHERE DO WE GO FROM HERE?," 2014.

[96] Sudol, A., *A Methodology for Modeling the Verification, Validation, and Testing Process for Launch Vehicles*. PhD thesis, Georgia Institute of Technology, Dec. 2015.

[97] Sullivan, M., Fairbairn, B., Bonner, M., Roberts, W. K., Stockdale, E., Park, J., Porter, M., and Lack, J., "F-35 Joint Strike Fighter: Current Outlook Is Improved, but Long-Term Affordability Is a Major Concern," tech. rep., DTIC Document, 2013.

[98] Sullivan, M., Masters, T., Anderson, P., Bennett, J., Bonner, M., Hassinger, K., Porter, M., Suding, M., and Volk, A., "F-35 Joint Strike Fighter: Assessment Needed to Address Affordability Challenges," tech. rep., DTIC Document, 2015.

[99] Sullivan, M., Zuckerstein, K., Ashley, J., Chanley, J., Greifner, L., Hemes, M., Jezewski, L., Ortiz, J., Schatz, S., and Stott, R., "Urgent Warfighter Needs: Opportunities Exist to Expedite Development and Fielding of Joint Capabilities," tech. rep., DTIC Document, 2012.

[100] Sztipanovits, J., Ying, S., Cohen, I., Corman, D., Davis, J., Khurana, H., Mosterman, P. J., Prasad, V., and Stormo, L., "STEERING COMMITTEE FOR FOUNDATIONS FOR INNOVATION IN CYBER-PHYSICAL SYSTEMS,"

[101] Teunter, R. H. and Fortuin, L., "End-of-life service," *International Journal of Production Economics*, vol. 59, no. 1, pp. 487–497, 1999.

[102] Thornton, R. K., "Review of Pending Guidance and Industry Findings on Commercial Off-the-Shelf (COTS) Electronics in Airborne Systems," tech. rep., DTIC Document, 2001.

[103] Turner, R., "Toward Agile systems engineering processes," *Crosstalk. The Journal of Defense Software Engineering*, pp. 11–15, 2007.

[104] US Department of Defense, *A Guide to the Project Management Body of Knowledge*. Fort Belvoir, VA: Defense Acquisition University Press, first edition ed., June 2003.

[105] VanLeer, M. D. and Jain, R., "A framework to address the impact of system of systems integration using commercially off the shelf (COTS) technology," *International Journal of System of Systems Engineering*, vol. 4, no. 1, pp. 23–43, 2013.

[106] Walden, D. D., "The Changing Role of the Systems Engineer in a System of Systems (SOS) Environment," in *Systems Conference, 2007 1st Annual IEEE*, pp. 1–6, IEEE, 2007.

[107] Williams, S., Drezner, J. A., McKernan, M., Shontz, D., Sollinger, J. M., Rand Corporation, Arroyo Center, United States, and Army, *Rapid acquisition of army command and control systems*. Santa Monica, CA: RAND Corporation, 2014.

[108] Winter, D. and Works, B. P., *Cyber Physical SystemsAn Aerospace Industry Perspective*. Boeing, 2008.

[109] Zheng, L., Terpenny, J., Sandborn, P., and Nelson, R., "Design refresh planning models for managing obsolescence," in *ASME 2012 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, pp. 579–590, American Society of Mechanical Engineers, 2012.