# NAVAL
# POSTGRADUATE
# SCHOOL

## MONTEREY, CALIFORNIA

# THESIS

**EXPLORING THE INTEGRATION OF COSYSMO WITH A MODEL-BASED SYSTEMS ENGINEERING METHODOLOGY IN EARLY TRADE SPACE ANALYTICS AND DECISIONS**

by

Dennis J. Edwards

June 2016

| | |
|---|---|
| Thesis Advisor: | Raymond Madachy |
| Second Reader: | John Michael Green |

**Approved for public release; distribution is unlimited**

THIS PAGE INTENTIONALLY LEFT BLANK

| REPORT DOCUMENTATION PAGE | | Form Approved OMB No. 0704–0188 |
|---|---|---|

| 1. AGENCY USE ONLY (Leave Blank) | 2. REPORT DATE<br>June 2016 | 3. REPORT TYPE AND DATES COVERED<br>Master's Thesis    07-07-2014 to 06-17-2016 | |
|---|---|---|---|
| **4. TITLE AND SUBTITLE**<br>EXPLORING THE INTEGRATION OF COSYSMO WITH A MODEL-BASED SYSTEMS ENGINEERING METHODOLOGY IN EARLY TRADE SPACE ANALYTICS AND DECISIONS | | **5. FUNDING NUMBERS** | |
| **6. AUTHOR(S)**<br>Dennis J. Edwards | | | |
| **7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)**<br>Naval Postgraduate School<br>Monterey, CA 93943 | | **8. PERFORMING ORGANIZATION REPORT NUMBER** | |
| **9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)**<br>N/A | | **10. SPONSORING / MONITORING AGENCY REPORT NUMBER** | |

**11. SUPPLEMENTARY NOTES**

The views expressed in this document are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government. IRB Protocol Number: N/A.

| 12a. DISTRIBUTION / AVAILABILITY STATEMENT<br>Approved for public release; distribution is unlimited | 12b. DISTRIBUTION CODE |
|---|---|

**13. ABSTRACT** (maximum 200 words)

This research explores the integration of the Constructive Systems Engineering Cost Model (COSYSMO) into early modeling efforts for the Department of Defense (DOD). Initial acquisition decisions influence the analysis and design of systems engineers, who face an increasingly complex and dynamic environment with significant impact on system life-cycle decisions and cost. This work utilizes a model-based systems engineering (MBSE) approach and the systems modeling language (SysML) to highlight the sharing of system model data with COSYSMO to provide an estimate of systems engineering costs. The document highlights the proposed methods compliance to cost estimation techniques, adherence and support to pre-Milestone A requirements of the DOD acquisitions process, and links to the Department of Defense Architectural Framework (DODAF). Application and documentation of the author's methods include a commonly used humanitarian aid effort scenario that incorporates traditional systems functional analysis of a single water distiller. The work's results demonstrate an ability for automated and semi-automated integration with COSYSMO from the system model in a web-based tool, conclude with challenges associated with external cost model integration, and suggest future areas of continued refinement and extensions. A starting point for an ensemble of models to enhance DOD cost estimation practices results from the techniques.

| 14. SUBJECT TERMS<br>COSYSMO, DODAF, MBSE, SYSML | | | 15. NUMBER OF PAGES   143 |
|---|---|---|---|
| | | | 16. PRICE CODE |

| 17. SECURITY CLASSIFICATION OF REPORT<br>Unclassified | 18. SECURITY CLASSIFICATION OF THIS PAGE<br>Unclassified | 19. SECURITY CLASSIFICATION OF ABSTRACT<br>Unclassified | 20. LIMITATION OF ABSTRACT<br>UU |
|---|---|---|---|

NSN 7540-01-280-5500

Standard Form 298 (Rev. 2–89)
Prescribed by ANSI Std. 239–18

THIS PAGE INTENTIONALLY LEFT BLANK

**EXPLORING THE INTEGRATION OF COSYSMO WITH A MODEL-BASED
SYSTEMS ENGINEERING METHODOLOGY IN EARLY TRADE SPACE
ANALYTICS AND DECISIONS**

Dennis J. Edwards
Captain, United States Army
B.S., United States Military Academy, 2007

Submitted in partial fulfillment of the
requirements for the degree of

**MASTER OF SCIENCE IN SYSTEMS ENGINEERING**

from the

**NAVAL POSTGRADUATE SCHOOL
June 2016**

Approved by:      Raymond Madachy
                  Thesis Advisor

                  John Michael Green
                  Second Reader

                  Ronald E. Giachetti
                  Chair, Department of Systems Engineering

THIS PAGE INTENTIONALLY LEFT BLANK

# ABSTRACT

This research explores the integration of the Constructive Systems Engineering Cost Model (COSYSMO) into early modeling efforts for the Department of Defense (DOD). Initial acquisition decisions influence t he a nalysis a nd d esign o f s ystems e ngineers, w ho f ace an increasingly complex and dynamic environment with significant i m pact o n s y stem life-cycle decisions and cost. This work utilizes a model-based systems engineering (MBSE) approach and the systems modeling language (SysML) to highlight the sharing of system model data with COSYSMO to provide an estimate of systems engineering costs. The document highlights the proposed methods compliance to cost estimation techniques, adherence and support to pre-Milestone A requirements of the DOD acquisitions process, and links to the Department of Defense Architectural Framework (DODAF). Application and documentation of the author's methods include a commonly used humanitarian aid effort scenario that incorporates traditional systems functional analysis of a single water distiller. The work's results demonstrate an ability for automated and semi-automated integration with COSYSMO from the system model in a web-based tool, conclude with challenges associated with external cost model integration, and suggest future areas of continued refinement and extensions. A starting point for an ensemble of models to enhance DOD cost estimation practices results from the techniques.

THIS PAGE INTENTIONALLY LEFT BLANK

# Table of Contents

# List of Figures

THIS PAGE INTENTIONALLY LEFT BLANK

# List of Tables

THIS PAGE INTENTIONALLY LEFT BLANK

# List of Acronyms and Abbreviations

**AFB**         Air Force base

**AFIT**        Air Force Institute of Technology

**AoA**         analysis of alternatives

**AV**          All Viewpoint-DODAF

**bdd**         block definition diagram

**BPMN**        Business Process Model Notation

**C4ISR**       Command, Control, Communications, Computers, Intelligence,
                Surveillance, and Reconnaissance

**CAD**         computer aided design

**CADM**        Core Architecture Data Model

**CAPE**        Cost Assessment and Program Evaluation

**CCE**         component cost estimate

**CDM**         conceptual data module

**CDD**         Capability Development Document

**CER**         cost estimating relationship

**COCOMO**      Constructive Cost Model

**COSYSMO**     Constructive Systems Engineering Cost Model

**CV**          Capability Viewpoint-DODAF

**DAG**         Defense Acquisition Guidebook

**DAU**         Defense Acquisition University

| | |
|---|---|
| **DAS** | Defense Acquisition System |
| **DCIO** | Deputy Chief Information Officer |
| **DIV** | Data and Information Viewpoint-DODAF |
| **DM2** | DODAF Meta Model |
| **DOD** | Department of Defense |
| **DODAF** | Department of Defense Architectural Framework |
| **DOTMLPF** | Doctrine, Organization, Training, Materiel, Leadership and Education, Personnel, and Facilities |
| **DSE** | data services environment |
| **EA** | Enterprise Architecture |
| **EVM** | earned value management |
| **GAO** | Government Accountability Office |
| **HTML** | hypertext markup language |
| **ibd** | internal block diagram |
| **ICE** | initial cost estimate |
| **IDEAS** | International Defense Enterprise Architecture Specification |
| **INCOSE** | International Council on Systems Engineering |
| **JCIDS** | Joint Capabilities Integration and Development System |
| **JPSS** | Joint Polar Satellite System |
| **KPP** | Key Performance Parameter |
| **LDM** | logical data module |
| **LML** | lifecycle modeling language |

| | |
|---|---|
| **MBSE** | model-based systems engineering |
| **MDA** | milestone decision authority |
| **MDAP** | major defense acquisition programs |
| **MDD** | materiel development decision |
| **MOE** | Measure of Effectiveness |
| **MOP** | Measure of Performance |
| **MSA** | materiel solution analysis |
| **NATO** | North Atlantic Treaty Organization |
| **NPS** | Naval Postgraduate School |
| **NR** | Net Ready |
| **OMB** | Office of Management and Budget |
| **OMG** | Object Management Group |
| **OMG SysML** | Object Management Group Systems Modeling Language |
| **OSD** | Office of the Secretary of Defense |
| **OV** | Operational Viewpoint-DODAF |
| **PES** | physical exchange schema |
| **PPBE** | Planning, Programming, Budgeting, and Execution |
| **RAM-C** | reliability, availability, maintainability, cost |
| **SE** | Systems Engineering |
| **SEP** | Systems Engineering Plan |
| **SoS** | system of system |
| **SPEC** | Systems and Proposal Engineering Company |

| | |
|---|---|
| **StdV** | Standards Viewpoint-DODAF |
| **SV** | Systems Viewpoint-DODAF |
| **SvcV** | Services Viewpoint-DODAF |
| **SysML** | Systems Modeling Language |
| **TEMP** | Test and Evaluation Master Plan |
| **UAV** | Unmanned Aerial Vehicle |
| **UML** | Universal Modeling Language |
| **UPDM** | Universal Profile for DODAF and MODAF |
| **U.S.** | United States |
| **WBS** | work breakdown structure |
| **XML** | Extensible markup language |
| **XSD** | XML schema definition |

# Executive Summary

Growing complexity is a continuous challenge for many large organizations. For major defense acquisitions in the Department of Defense (DOD), this challenge has additional regulatory and budgetary constraints and potentially catastrophic impacts to poorly designed, underperforming, or delayed delivery to the warfighter. The increasing demand for interoperability and rate of change in digital information is fueling the complexity growth. This thesis discusses the integration of existing methods, models, and tools to support cost understanding for early trade decisions.

Fielding able, agile, and affordable defense systems requires efficient use of available resources including data, but also require integration into the event-driven process of defense acquisitions to impact early life-cycle decisions and analysis. One current data-driven process with clear consequences to defense acquisitions and national attention is affordability. One growing trend in systems engineering, which can leverage this data for systems analysts, systems architects, and systems engineers, is model-based systems engineering (MBSE). Using the system model as the foundational element for decision maker communication, system analysis, and model documentation poses both benefits and c oncerns. Benefits for cost estimation include increased speed, reduced bias, and built-in sensitivity for cost understanding of systems. Concerns involve information and model security, building robust models for both communication and analysis, and lastly, organizational alignment to support the MBSE trend that fits practical DOD policy.

This work provides three key discussion points and presents them in themes of compliance, data focus, and effective integration. Initially, a formulation describing the relationships and mappings among current acquisition and cost estimate practices to the systems engineering activities and the current MBSE trends is given. Next, a humanitarian relief scenario and distiller model illustrate and enhance the MBSE approach. An introduction of the constructive systems engineering cost model (COSYSMO) follows, which utilizes approved DOD cost estimation techniques and requires a relatively simple model input to estimate systems engineering costs. The system model, the modeling language used (SysML), and COSYSMO are brought together to discuss the identification of the necessary cost estimate parameters, and the result of this integration generates a cost estimate for the humanitarian

relief scenario by only using the system model data. Finally, a method to improve the parameter extraction is proposed and demonstrated as an initial proof of concept with a web-based application. Recommendations for future work and extensions include using the discussed methodology to enhance integration of systems modeling and cost estimation for the DOD. Figure 1 provides a conceptual overview of the work and highlights a foundational aspect of this thesis; communication from a model.
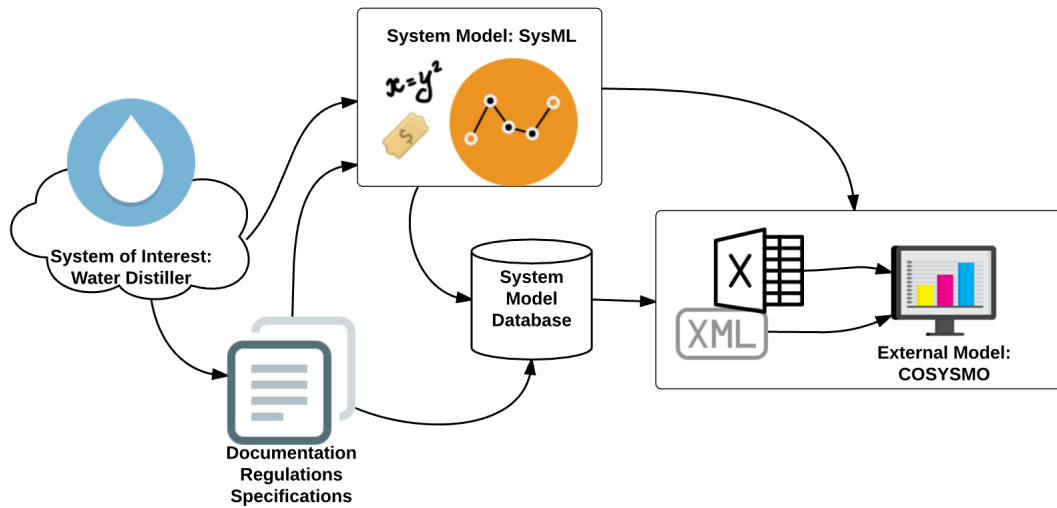


Figure 1: Conceptual Overview of External Cost Model Integration.

# Acknowledgments

This effort is indebted to the continued love and support of my wife, Tiffiny, and my daughter Camille. I would also like to extend my appreciation and gratitude to my thesis advisor Ray Madachy and second reader Mike Green, for their expertise and support for this work. Finally, I must also extend my sincerest appreciation to the staff, faculty, and students of the Systems Engineering and Systems Engineering Analysis curriculum during my Naval Postgraduate School experience. Their constant encouragement and sound advice helped support critical thinking and continuous consideration of perspectives.

A special thank you to original authors Sanford Friedenthal, Alan Moore, Rick Steiner and the publisher Elsevier for permission to use copyrighted materials. The figures illustrating the original water distiller SysML diagrams in this work are:

"Reprinted from *A Practical Guide to SysML: The Systems Modeling Language*, 2nd Edition, Sanford Friedenthal, Alan Moore, Rick Steiner, Water Distiller Example Using Functional Analysis, 393-429, Copyright (2012), with permission from Elsevier."

License number: 3854261199824

License date: April 15, 2016

THIS PAGE INTENTIONALLY LEFT BLANK

# CHAPTER 1:
## Introduction

The Department of Defense (DOD), among many major organizations, is seeing an increase in the demand for systems engineering and systems integration [1]–[3]. Teams of engineers, program managers, and organizational decision makers adapt into geographically and domain diverse teams to meet this growing demand. Each team member possesses a unique perspective, a preferred practice, and inherent limitations. Providing practical and affordable solutions require appropriate identification, understanding, and management of these constraints. The complexities of problems are also increasing [3]–[5]. Some assert that the complexity is growing at an unsustainable rate and level [3]. The global economy and speed of information have placed much of the focus on early design aspects due to their long-term life-cycle effects and made initial design choices increasingly critical. Often the system design and the ability to untangle massive amounts of information, inputs, and constraints of a proposed system is complex, dynamic, and time-consuming. These descriptors, along with the primary mission of sovereign defense, place the DOD in a challenging position. The warfighter demand requires a unique tempering of innovative technology, a continued prioritization of resources and funding, and the flexibility to meet a dynamic and determined enemy.

This work provides a means to enhance current external model integration and systems engineering cost estimation techniques using a model-based systems engineering (MBSE) methodology. This MBSE approach brings together the interrelated ideas of system analysis and cost estimations through the system architecture and model. Specifically, the work will highlight how the Constructive Systems Engineering Cost Model (COSYSMO) and external models like it, can leverage existing DOD systems engineering techniques and practices to support the organization's ability to develop, improve, and sustain able, agile, and affordable defense systems through system model integration. This chapter will serve as the introduction to the overall problem, highlight the current role of systems engineering in the DOD, provide the research objective and focus of the work, and finally, conclude with the thesis organization and logical development of the research.

## 1.1  Overview

Early aspects of major defense acquisition programs (MDAP) for the DOD are critical [6]–[8]. The early identification and risk mitigation of current and proposed complex systems are a constant challenge for key contributors to the core processes associated with the Joint Capabilities Integration and Development System (JCIDS) process [9]–[11]. Key among those core processes and the foundation of this work is systems engineering. The current model-based approach to systems engineering provides an efficient means to evaluate the impact of competing designs, compare proposed alternatives, and assess varying technological maturity. The approach also provides a means to incorporate available data and information about a system of interest into various organizations and validated tools across the DOD [9]. Despite the standardization of data structure, storage, and exchange mandated by the Department of Defense Architectural Framework (DODAF) a need still exists in developing efficient means to integrate accurate, useful, and validated data into early trade space decisions. Also needed is an understanding of the impacts of cost over a system lifecycle. The COSYSMO is one of many documented, available, and validated tools available to the DOD in regards to major defense systems that demonstrate this gap, has shown previous utility in major defense acquisitions, and the model is an effective means to estimate the systems engineering cost of an effort.

## 1.2  Research Objective

The objective of this thesis is to explore COSYSMO systems engineering cost estimates and its integration into early decisions for the DOD. This research will explore COSYSMO and propose a method to locate useful COSYSMO parameters from an MBSE approach applied to a water distiller, count those representative model entities, and pass them to COSYSMO. The overall value of the study is to highlight challenges with COSYSMO integration with Systems Modeling Language (SysML), understand the impact of the DODAF requirements for the DOD, and highlight future areas for COSYSMO integration for major defense acquisitions programs.

## 1.3 Research Focus

The nature of the research topics presented and the tool under consideration suggest a DOD focus with a heavy concentration on the practice, tools, and integration of the research results to the progression of systems engineering and MBSE. Towards this end, the following research questions are:

**Research Question 1:** How does the integration of COSYSMO fit into the current DOD acquisitions process?

**Research Question 2:** How does COSYSMO map to DODAF?

**Research Question 3:** How does COSYSMO map to SysML?

**Research Question 4:** What challenges exist with integrating COSYSMO systems engineering cost estimates into early Department of Defense (DOD) decisions?

## 1.4 Thesis Methodology

The following organization applies as an overarching approach to the exploration of COSYSMO and its integration for MDAP in the DOD. The first section uses a historical lens to introduce the topics of MDAP, cost estimation, MBSE, and their natural links to DODAF. The discussion includes the formulation of each and their use in the DOD. Next, an in-depth discussion of COSYSMO from its initial formulation to its current state will provide sufficient detail to serve as an understanding of the model, its intended purpose, its necessary input, and output. One presented example of a COSYSMO tool highlights the ease of use and data entry methods. An overview of SysML follows with a focus on primary origins, the diagram types, and the model purposes, which serve to highlight the specific modeling language, used in this work. The work will then transition to a proposed MBSE methodology using a well-documented, simple, and validated model of a water distiller from the works of [12]–[14] and stored by Object Management Group (OMG) and Elsevier [15]. This section provides context for finding, counting, and passing necessary variables from the distiller model as inputs to COSYSMO while highlighting challenges and benefits of such efforts in both written and visual methods. Finally, the work will conclude with the output of the research results and suggest future areas for refinement and improvement.

## 1.5   Thesis Assumptions

The following assumptions applied throughout the entire execution of this exploratory research of COSYSMO integration. These assumptions highlight overarching and broad aspects in order to discuss the proposed problem statement and associated boundaries of the problem. The order does not suggest prioritization or importance.

- Documentation of processes and procedures represent an ideal case and what is expected (JCIDS, DAS, SE, or cost estimation).
- Models presented or created represent the best available understanding of the true state of nature for a system as a snapshot in time.
- A representative and sufficient set of system models are achievable through MBSE.
- A relationship exists between at least one or many SysML model entities and at least one of the COSYSMO parameters.

## 1.6   Thesis Organization

The research discussed will focus on the specifics of the integration of COSYSMO for DOD applications; however, the concept of identifying model entities, performing analysis on the information, and passing that information into accurate, useful, and validated tools is expected to have related techniques for other external models utilized in the DOD. Refinement and validation techniques may be required to complete applying this approach to other more complex models, but the concept of finding usable entities that maintain the necessary relationships and rigor should apply. Model validation does not occur in this work. Instead, the proposed methods provide a possible solution until either validation of this approach or a refinement occurs. The use of a well-known and readily available model will focus future efforts on integration refinement and serve as a basis for improvement.

The following is a brief description of each chapter of this research effort and provides both an overview of the contents of each chapter and allows for an overarching view of the thesis logical development. This chapter is an introduction to the discussed problem, the overarching thesis assumptions, and methodology. Chapter 2 is a literature review to provide ample background on the essential topics of MDAP, cost estimation, MBSE, DODAF, and SysML. This chapter also provides a detailed overview of COSYSMO from its original form to its current state by discussing the available literature on the cost model

and follow-on iterations, assessments, and applications of the tool. Chapter 3 serves as the formulation and documentation of the author's proposed means for integration. This chapter applies an abbreviated MBSE approach to a water distiller scenario for a humanitarian aid organization. This scenario is adapted and expanded from [12]–[14]. The majority of this chapter focuses on the proposed method of identifying the links between the SysML models and the COSYSMO input variables using the water distiller as context. Additionally, discussions include the source of the system model, the modeling tool, and challenges when applicable. Chapter 4 contains the analysis and results of the proposed methodology as applied to the water distiller previously discussed and provides results of the cost model integration. Finally, Chapter 5 concludes the work with recommendations to explore other models, offers a suggestion for a means of validation of the results, and ends with suggestions for future refinements and work.

THIS PAGE INTENTIONALLY LEFT BLANK

# CHAPTER 2:
# Background: Themes of Compliance, Data Focus, and Modeling Trends

This chapter serves as the necessary and sufficient background information to understand and follow the underlying themes of event-driven compliance, design trades, and data focus found among the concepts of MDAP, cost estimation, DODAF, and MBSE. The aggregation of these ideas highlights an exciting and challenging integration concept for the DOD. Most of the discussion that follows gives historical and regulatory information to show the progression and development of the ideas as applicable to the DOD. For those familiar with all of the topics mentioned, the reader is encouraged to transition to Sections 2.4 and 2.6 to discuss the aspects of the specific external model COSYSMO and the modeling language presented in this work, SysML. A clear understanding of those two sections will provide sufficient context for the methodology, analysis, and recommendations that appear in Chapters 3-5.

## 2.1   Major Defense Acquisition Programs: The Big Ticket Items

Major defense acquisitions programs, those in excess of $480 million for Research, Development, Test, and Evaluation (RDT&E) or, $2.79 billion in fiscal year 2014 constant dollars or, designated as such by the Under Secretary of Defense for Acquisition, Technology, and Logistics (USD(AT&L)), have a very special role in the DOD [7], [8], [10], [11], [16]. These programs must meet government accountability measures and audits to ensure that the policies and principles in place by the organization are being followed and comply with federal regulations for use of taxpayer dollars. This report will explore early trade space decisions and therefore will focus on those activities that are up to and including the Milestone A decision point. These activities have a notable impact on the lifecycle of the system and depend heavily on systems engineering decisions [3], [17]. Highlighted in Figure 2.1 adapted from the *Defense Acquisition Guidebook (DAG)*, these activities include a mix of enabling science and technology for incorporation into early prototyping

and concept development for the materiel development decision (MDD), to the disposal of that system [16] often termed "cradle to grave" [18]. There are several variants of Figure 2.1, which include the various major reviews, audits, and best practices events, but this one captures the collective system understanding in early decisions before Milestone A.
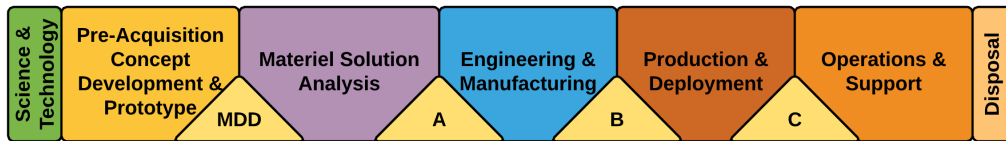


Figure 2.1: Overview of MDAP Milestones for a Defense System. Adapted from [16, 4.2.1.F1].

Capability-based assessments generated from the warfighter and their appropriate service components serve as a catalyst for the science and technology industries to meet DOD requirements and build pre-acquisition concepts. In this portion of the assessment, Doctrine, Organization, Training, Materiel, Leadership and Education, Personnel, and Facilities (DOTMLPF) considerations are addressed to highlight changes to current policies and guidelines that are affected by the proposed capability but also weigh the option of no materiel solution at all. After incorporating these DOTMLPF considerations and the output of the developmental system planning, technical feasibility, and cost considerations, the milestone decision authority (MDA) for the program determines the materiel development path [11]. This decision maker approves either continued development of the program to the materiel solution analysis (MSA) Phase or at an entry point beyond the MSA Phase or stops development. An unfavorable MDD means the program does not continue. An entry point beyond the MSA is an acceptance of program risk due to an assessment of the technological maturity, program complexity, and criticality of the capability for the warfighter.

### 2.1.1   Pre-Milestone A

After a materiel solution approval, the systems engineering activities during the MSA Phase result in critical products and analysis for the program. Chief among these activities includes a system model and system architecture [16], [17], [19] that focuses on the operational concept, system boundaries and interfaces, operational and functional requirements, and finally, life-cycle system performance and costs of a preferred system. Concurrent activities, which occur during this phase, influence the system model. These activities include an

analysis of alternatives (AoA), systems performance, technical, and operational analysis. The outputs of this analysis in the form of documents and various reports help shape the developing technical and programmatic planning tools and strategies for the system. These items become the source data for the system model. Figure 2.2 adapted from the *DAG*, helps summarize the intricacies of the early trade decisions and analysis of the MSA Phase [16]. Typical plans include the Systems Engineering Plan (SEP), Test and Evaluation Master Plan (TEMP), Program Protection Plan (PPP), or Life-Cycle Sustainment Plan (LCSP). Strategies include the Technology Development Strategy (TDS), Test and Evaluation Strategy (TES), and prototyping strategy among many others [16], [17], [20]. These plans have lasting impacts on a system.
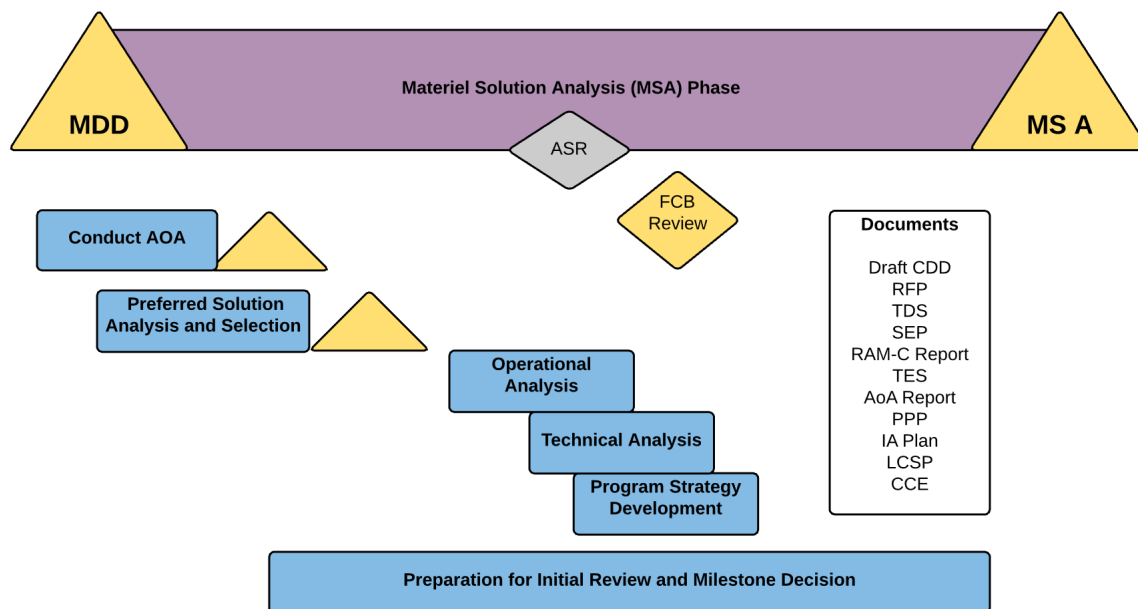


Figure 2.2: Milestone Overview- Early System Life-Cycle Analysis and Decisions. Adapted from [16, Figure 4.2.3.F1].

The yellow triangles present the entry and exit decisions for the MSA Phase. The blue items represent a series of concurrent analysis, planning, and strategy development that follow a favorable MDD. These actions are the source of the various documents and data sources, which focus on an understanding of the required system capability to meet user need, system performance, system life-cycle planning, and cost. The process and refinement are iterative and highly linked despite the presented sequential diagram. Information on the reliability, availability, maintainability, cost (RAM-C) and the component cost estimate (CCE), for

example, will be immature and always changing as multiple agencies and organization refine their understanding and system considerations [7]. This early phase of the system design is the focus of this work. The potential benefits are increasing communication, data efficiency, and early cost understanding by fusing the information and data used to develop the system model and an available external cost model COSYSMO.

### 2.1.2 DOD Systems Engineering Tasks Through Milestone A

The DOD has specific expectations for each of the milestone decision points and technical reviews [21]. Studies have linked successful programs to knowledge-based practices and data-driven decisions [8], [22], [23]. The DOD milestones specify a process and the associated metrics for evaluating a given system throughout its lifecycle in a well- organized and reproducible way. Table 2.1 provides those early decision points related to systems engineering through Milestone A [16, Table 4.2.1.T1]. These early decisions highlight the impact of systems engineering for major defense acquisitions systems and show themes of able, agile, and affordable defense systems. There is a definite DOD focus on cost, schedule, and performance.

Figure 2.3 shows the systems engineering emphasis applied to the DOD eight technical processes and eight technical management process associated with MDAP and over the system lifecycle [16]. These customized actions represent the DOD's development of a process specifically addressing defense weapon systems, but the technical and technical management process are very similar to foundational systems engineering guidebooks and textbooks [1], [17], [20], [24]. Different types of systems and organizations will use slight variations of these actions, but this is how the DOD conducts systems engineering. The DOD systems engineering process is the department's means to design, synthesize, analyze, and evaluate [17] various competing considerations for the DOD within the regulatory and procedural constraints of the organization.

The MDAP early trade analysis and decisions present a unique blend of interlinked analysis across multiple domains and perspectives. Decisions and products are the result of data-driven and focused analysis for systematic evaluation of the programs over time. Currently, a system model is the focal point of this analysis and depending on the type and breadth of the model may present a source for external model integration. This work proposes the use

Table 2.1: Systems Engineering Tasks Through Milestone A. Early Trade Space Decisions. Adapted from [16, Table 4.2.1.T1].

| DOD Acquisition Event | Objective | Technical Maturity | Goal |
|---|---|---|---|
| **Materiel Development Decision (MDD)** | Assess potential materiel solutions and appropriate entry phase into DOD system life cycle | Capability gap met with materiel solution acquisition | Identify technically feasible solutions which may meet a validated DOD capability need and understand system technical risk |
| **Alternative Systems Review (ASR)** | Identify a preferred materiel solution which can affordably meet user needs with acceptable risk | Determine system parameters; balance system cost, schedule, and risk | Establish initial system performance and plan for additional Milestone A criteria |
| **Milestone A** | Investment decision about technology maturation and preliminary system design | An affordable solution identified for the warfighter need with acceptable technology risk, scope, and system complexity | Synchronization of resources, to meet warfighter needs and systems performance objectives within affordability constraints. Mitigate technical risk |

of a systems engineering cost model as one external model that may bring benefit to the MSA Phase by supporting efficiency and analysis through data reuse and cost estimation in line with the early decisions of the DOD. This section highlights the possible entry point for early trade space analysis in the DOD MDAP as the MSA Phase of the system lifecycle. While not the only available model, this selected model will serve as proof of concept, for external models like it. The next section will discuss the management and regulation of the

| Legend | SE Technical Management and Technical Processes -- Focus Areas in Acquisition Phases | | | | | |
|---|---|---|---|---|---|---|
| ● = Major Use<br>◉ = Moderate Use<br>○ = Minor Use | **Pre-MDD** | **MSA** | **TD** | **EMD** | **P&D** | **O&S** |
| **TECH MGMT PROCESSES** | | | | | | |
| Decision Analysis | ● | ● | ● | ● | ● | ● |
| Technical Planning | ● | ● | ● | ● | ● | ● |
| Technical Assessment | ◉ | ● | ● | ● | ● | ● |
| Requirements Management | ◉ | ● | ● | ● | ● | ● |
| Risk Management | ◉ | ● | ● | ● | ● | ● |
| Configuration Management | ○ | ◉ | ● | ● | ● | ● |
| Technical Data Management | ○ | ● | ● | ● | ● | ● |
| Interface Management | ◉ | ● | ● | ● | ● | ● |
| **TECHNICAL PROCESSES** | | | | | | |
| Stakeholder Requirements Definition | ◉ | ● | ● | ◉ | ○ | ○ |
| Requirements Analysis | ◉ | ● | ● | ● | ○ | ○ |
| Architecture Design | ◉ | ● | ● | ● | ○ | ○ |
| Implementation | ○ | ◉ | ◉ | ● | ◉ | ○ |
| Integration | ○ | ◉ | ◉ | ● | ● | ○ |
| Verification | ○ | ◉ | ◉ | ● | ● | ◉ |
| Validation | ○ | ◉ | ◉ | ● | ● | ● |
| Transition | ○ | ○ | ◉ | ● | ● | ● |

Figure 2.3: Focus During DOD Systems Engineering Tasks- Weapon System Life-Cycle Example. Source: [16, Table 4.3.1.T2].

system model across the DOD enterprise and its architectural framework.

## 2.2 DODAF: A Historical Response to Complexity

### 2.2.1 Origin of DODAF

DODAF version 2.02, has its beginning in the Command, Control, Communications, Computers, Intelligence, Surveillance, and Reconnaissance (C4ISR) Architecture Framework version 1.00 originally developed in 1996 [25]. The focus then was the growing complexity of defense systems and increasing speed and expectation of these capabilities from the joint warfighter. This growth of complexity and ideas raised Congressional and Secretary of Defense level concern and resulted in specific administrative requirements, DOD directives, and policy. Chief among these documents was the Clinger-Cohen Act and the Office of Management and Budget (OMB) Circular A-130, which mandated the development of architectures to understand and better align information and technology mission area for

the DOD to meet national objectives.

Efforts continued with the information and technical mission area, producing C4ISR Architecture Framework Version 2.00 in 1997 and mandating the use of the framework in 1998. Five years later the C4ISR Architecture Framework expanded to include other DOD mission areas, and DODAF version 1.0 emerged as the architectural framework to adhere to further regulatory guidance, but also an evolving DOD [25], [26]. The initial DODAF implementation served as a systematic way to guide architecture development and support decision making for the DODs current day core processes. These core processes include JCIDS, Defense Acquisition System (DAS), Planning, Programming, Budgeting, and Execution (PPBE), Systems Engineering (SE), net-centric integration/operational planning, and portfolio management [10], [11], [25]–[28]. The change in scope and scale to the foundational C4ISR Architecture Framework had a similar effect on the documentation and number of products developed to support the new framework. Version 1.0 of DODAF, created three volumes of documentation and over 100 products. Many termed the original versions of DODAF as "product centric" and in Volume I of DODAF version 1.0; each diagram used the Universal Modeling Language (UML) [27], [25, Vol. I, pp. 1-2], [26, Vol. I, p. 5].

Continued technological maturity and architectural development across the DOD provided the justification for the first fundamental change of DODAF. The need for data overshadowed the physical practice of building products. The need included not only the information or data itself, but also, efficient means to store, maintain, and reuse the information across core mission areas [25]–[27]. This fundamental change of DODAF introduced Core Architecture Data Model (CADM) as an integral component of DODAF and included the new utility of integrated and federated architectures [25], [28]. Similarly, key relationships emerged across the enterprise architecture of the DOD as data was now usable across larger portions of the DOD and the architectural framework became a better means to inform decision makers through a logically consistent, data supported, and organizationally standardized discussion [25].

In its current form, DODAF version 2.02 is a continued refinement of the often called "data centric" approach of DODAF version 1.5 to include a clear delineation of data, model, view, and viewpoint [27], [25, Vol. I, pp. 1-2], [26, Vol. I, p. 5]. DODAF Meta Model

13

(DM2) replaced CADM as the foundational component for data concepts, relationships, and attributes [26], [28]. In addition to the switch in the logic, a model refinement occurred. A reduction took place from the original hundreds of models in version 1.5 to only 52 models in version 2.0. Additionally, eight viewpoints exist: all view (AV), capability view (CV), data and information view (DIV), operational view (OV), project view (PV), service view (SvcV), standards view (StdV), and finally the systems view (SV). This reduction and grouping represented a focus shift and the system architecture development became second to data collection, storage, and maintenance necessary for efficient and effective decisions in the six core processes previously mentioned [25]. The reduction in the total number of models occurred with the introduction of the "fit for purpose model" which allowed for the development of models outside of the traditional DODAF views as long as the newly defined DODAF compliance was achieved [26, Vol. I, pp. 1]. DODAF compliance requires two elements: adherence to the DM2 concepts, relationships, and attributes and transfer of the system architectural data compliant to the physical exchange schema (PES) for DODAF [27]. That current PES is an XML schema definition (XSD).

This brief but detailed history of DODAF is to reinforce key ideas. The first construct is the anchoring regulatory and historical aspects of DODAF in both compliance and business process improvements. These improvements are directly related to cost and financial planning decisions due to growing complexity, which when reviewing the regulatory information of the Clinger-Cohen Act, OMB Circular A-130, and DOD Enterprise construct, link directly to cost and financial aspects of decision support. The second construct is the data emphasis, both as a means to support decision-making and fundamental to aspects of core DOD processes. These core DOD processes include specific DOD systems engineering tasks to develop new technologies, create and improve new systems, and add value through system life-cycle decision support and analysis despite growing complexity. Finally, the last construct highlights both the trend towards models as a source for data, a way to communicate, and an adaptable means to express different perspectives. From the standpoint of the Deputy Chief Information Officer (DCIO), the DODAF models measured utility is in the models ability to communicate across multiple platforms and levels of an organization and adhere to the governing logic constructs for use across the DOD to meet organizational needs and objectives. The communication of initial cost information proposed in this work seems pertinent and in line with current DOD guidance.

## 2.2.2 DODAF Meta-Model (DM2)

Terms, definitions, and structure allow for clear, concise, and efficient communication and as discussed in later sections highlight means for analysis when used in a structured form. Any introductory survey of a new topic or concept usually includes a familiarization with new terms, what the terms mean, and how they relate to an overarching concept. Over time, conventions and accepted norms become accepted standards for a domain. DODAF is no exception and as discussed in its short history highlights both the acceptance and rejections from its users. In fact, the DM2 is a discriminating aspect of DODAF. A finite and configuration managed [27] set of terms or entities are used to describe the various dimensions of the 52 DODAF models and eight viewpoints and how they support the DOD core processes [28]. This meta-model provides the approved syntax (terms) to specify the semantics (relationships) and format of all data, data storage, and exchange across architectures and tools in the DOD enterprise. The DM2 model provides a traceable and verifiable standard for process owners, architecture developers, data users, engineers, and analysts alike. It allows for the specific needs, methods, and tools of many various and unique organizations to communicate to a written standard. To ensure compliance with the DM2 conventions profiles or plug-ins have emerged for some languages and tools [29], [30] but the over 400 active terms in the DM2 is a challenge to utilize.

DM2 is also not without flaws [31]. NPS's Ronald Giachetti noted the lack of consistency and completeness as the DM2 term "performer" did not appear in any of four widely used systems engineering handbooks or desk guides. Also noted was the lack of clarity in the utilization of the term "activity." In practice, systems engineers distinguish between operational activities and functions, but DODAF and the DM2 only specify the single "activity" term [31]. Most concerning, is that the terms requirements and risk also did not appear despite their commonplace occurrence in systems engineering [31] and its focal point in DOD system acquisitions [1], [4], [16], [20], [32]. The lack of risk management and inconsistency occurred in [22] and [33] for both government and non-government modeling efforts. When conflict arrives, the DM2 becomes the basis for resolution and similar to a mathematical proof, the DM2 separates the point of contention to its fundamental terms and relationships to ensure consistency, completeness, and clarity. This user feedback is part of the evolution and maturation of DODAF and its use in the DOD, but these flaws and the 417 current terms associated with DM2 can negate some of the utility gained for effective

communication among diverse and disperse teams.

Despite these shortcomings, DM2 follows the current multi-layered data conventions in its design. Three data layers exist in the DM2 and the DODAF DIV 1-3 models. Pictured in Figure 2.4, the base or foundational data layer of DM2 is the PES. This layer consists primary of the actual source code information or data required to develop the structure of the next higher data levels. The DODAF PES is expressed as an XSD file and ensures that the logical data module (LDM) and conceptual data module (CDM) conveyed is in the required format and consistent with the Extensible markup language (XML). This check on data form and consistency are often termed ensuring a well-formed and valid set of data exists [34]. The PES is the DM2 standard that all of the information in the data layers must meet; this rule set creates the XSD or schema for short [27]. The CDM is the context and guidance; the LDM is the physical system model, and the PES is the physical system model data in a structured form. Entities that do not comply with the schema do not meet DODAF compliance.
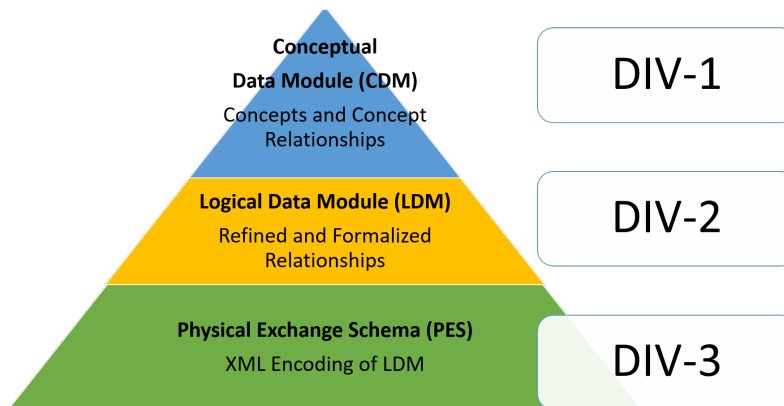


Figure 2.4: Modeling Convention DM2 Data Layers. Adapted from [27].

The next data layer, the LDM is the physical information portrayed in a system model and is typically the layer where a system architecture takes its form [19], [31], [35]. In its entirety, there are 510 terms, 417 of those names are active in the DODAF dictionary. The DM2 term groupings are by foundational ontological properties and semantic concepts into sub models or DM2 data groups, which make the translation to the CDM understandable and simplify the data layer [26]. Volume II of the DODAF version 2.0 highlights the concepts, attributes, and relationships of each DM2 term and the thirteen groups or clusters [27]. The next two lists provide the definition and categorization of the DM2 groups for the reader

16

directly from [27] as either a primary or supporting architectural constructs. The irregular capitalization and awkward grammar in the definitions denotes a DM2 term or specific DM2 phrase.

## Principle Architectural Constructs

| | |
|---|---|
| **Performers:** | Any entity - human, automated, or any aggregation of human and/or automated - that performs an activity and provides a capability. |
| **Resource Flows:** | The behavioral and structural representation of the interactions between Activities (which are performed by Performers) that is both temporal and results in the flow or exchange of objects such as information, data, materiel, and performers. |
| **Information and Data:** | Representations (descriptions) of things of interest and necessary for the conduct of activities. Information is the state of a something of interest that is materialized – in any medium or form – and communicated or received. |
| **Rules:** | How rules, standards, agreements, constraints, and regulations and are relevant to architectures. A principle or condition that governs behavior; a prescribed guide for conduct or action |
| **Goals:** | How goals, visions, objectives, and effects relate and bear on architectures. A desired state of a Resource |
| **Capability:** | The ability to achieve a Desired Effect under specified [performance] standards and conditions through combinations of ways and means [activities and resources] to perform a set of activities. |
| **Services:** | A mechanism to enable access to a set of one or more capabilities , where the access is provided using a prescribed interface and is exercised consistent with constraints and policies as specified by the service description. The mechanism is a Performer. The "capabilities" accessed are Resources – Information, Data, Materiel, Performers, and Geo-political Extents. |
| **Project:** | All forms of planned activities that are responsive to visions, |

goals, and objectives that aim to change the state of some situation. A temporary endeavor undertaken to create Resources or Desired Effects.

**Reification:** The process of reifying or to regard (something abstract) as a material or concrete thing. Reification, in DODAF 2, is used to introduce the concept of the varying levels of architectural descriptions or refinement and traceability between the levels.

**Organizational Structure:** Representations of the organization types, organizations and individuals that is present in the architecture.

**Supporting Architectural Constructs**

**Measures:** All form of measures (metrics) applicable to architectures including needs satisfaction measures, performance measures, interoperability measures, organizational measures, and resource physical measures (e.g., mass.). The magnitude of some attribute of an individual.

**Locations:** A point or extent in space that may be referred to physically or logically.

**Pedigree:** The origin and the history of something; broadly: background, history [27].

Note that each of these groups represents several DM2 terms, and that each term has a very specific usage in each of the DM2 groups. Adherence to this mapping must govern the system model see Figure 2.5. This mapping is what expresses a model's completeness and rigor. The *performer* group is provided as an example to highlight the level of detail and rigor of the DM2 groups and highlight the notes on the model that provide the common language to help explain the terms. It is apparent the model is evolving and improving from user feedback. To express a unique entity entirely requires its expansion from its DM2 group down to the necessary level of detail and consideration. In the given *performer* example, this would include a specificity from the *performer* as a resource, organization, activity or person (purple entities) down to the specific, and measurable level of skill, agreement, or rule (blue objects) associated with the originally specified performer type. Also, note at the bottom left of Figure 2.6 that everything has parts, and subparts and any DM2 term can have
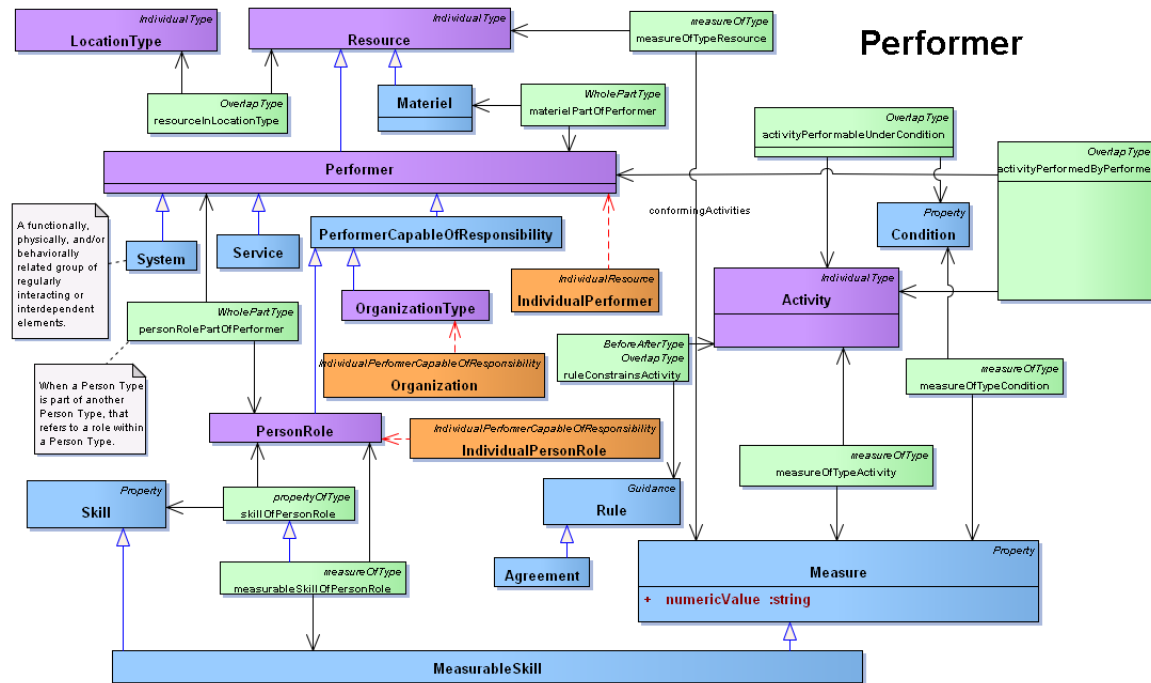
Figure 2.5: DM2 Performer Group. Source: [36].

measures. The full set of unique combinations presents a significant amount of information to specify an entity and meet the defined DM2 schema, but a unique identifier exists to describe each model entity. This existence is critical to the system model integration efforts of this thesis.

Finally, the conceptual data module CDM provides the highest level of abstraction for a particular system written for executives and decision makers. Similar to the DM2 group definitions and Figure 2.6, the CDM provides a general expression to explain the relationship between DM2 concepts, but the terminology is slightly different. A single simple phrase expresses the overarching relationship of the concept for each of the 12 concepts described. The CDM is meant to convey the information in a non-technical aspect and addresses the standard interrogatives of who, what, when, where, why of similar frameworks such as Zachman [37] or Universal Core [27]. Note that the foundational underpinning for DM2 is the International Defense Enterprise Architecture Specification (IDEAS) construct shown in Figure 2.6. This ontology, developed by the North Atlantic Treaty Organization (NATO) nations of the United States, United Kingdom, Canada, Australia, and Sweden provides structure and clarity for defense-related collaboration and data exchange [36] DODAF
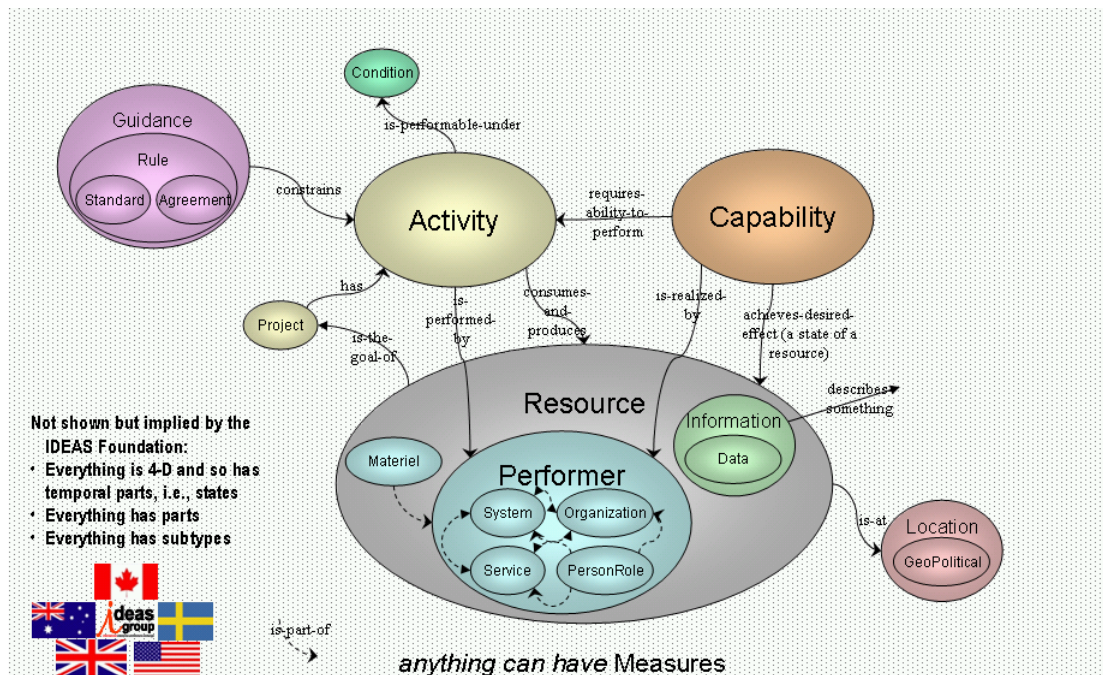
Figure 2.6: IDEAS Logic That Influences DM2 CDM. Source: [36].

inherited many of the rules and constructs used from IDEAS and modified the terminology to fit DOD users [36].

The discussion and examples of DM2 show the breadth and depth of the framework. This structure highlights the ontological construct that allows DM2 to serve as a means to standardize data, regulate data exchange, and improve communication efforts across a large organization without specifying a methodology or tool. Defined in [38], but adapted from Gruber, an ontology is "an explicit formal specification of a conceptualization that consists of a set of concepts in a domain and relations among them." [38, p. 43] The utility of the ontological approach allows for the rigor of the PES and LDM levels of the DM2, but also provides a means to communicate with a sufficient degree of abstraction both in terms and in practice. Inconsistencies and ambiguities emerge from a lack of discipline in a project effort when the usage of standard terms, their definitions, and ontological structure is absent. Any ambiguity, whether verbal or written, must be mitigated through the consistent usage of terminology to account for inherent deficiencies of trying to encapsulate complex and dynamic systems in only 417 terms [35].

This concept helps explain why DODAF compliance is only data described using the DM2 terms (concepts, associations, and attributes) and that the entire set of architectural data is transferable among users, tools, and storage repositories using the DM2 PES. It represents the current ability to express the DOD core processes in a logically rigorous and consistent manner. DODAF compliance does not ensure model utility, only model conformance, and provides data exchange standards for DOD system interoperability. This interoperability has extended from the C4ISR joint concerns of the mid to late nineties to the current multinational aspects of a technologically advancing and globally linked threat. This need for interoperability provides one factor contributing to defense system complexity.

### 2.2.3  DODAF Milestone A Requirements

A subset of DODAF models follow a similar submission and approval process to the document, system analysis, and Milestone A approvals discussed in Section 2.1 covering MDAP and the various Milestone decisions. These models correspond to the level of analysis and understanding necessary to proceed through the acquisitions process and are mandated by [10] and [11]. In total, 25 models are listed as part of the overall system engineering and analysis effort during the MSA Phase and include models from seven of the eight viewpoints of DODAF [11], [39]. Figure 2.7 highlights the specific models required for system Milestone A decisions. Specifically, 23 particular models make up the Capability Development Document (CDD) associated with the early decisions in the MSA Phase. Two models, the OV-2, and SV-7 occur in both the DODAF and Net Ready (NR) Key Performance Parameter (KPP) sections that address required models. In prior versions of the JCIDS, these descriptions were included as one table. The NR-KPP section was a recent change, to focus efforts on the need to facilitate decision-making based on performance attributes and system analysis [11]. Again, we see themes of compliance and data-driven decisions supporting the DOD, with potential impact to cost estimation when considering affordability measures and trades.

## 2.3  Cost Estimation: Its Role in DOD and MDAP

Cost estimation is a necessity in almost any organization, and when considered on a practical scale, it occurs in most everyone's life. The question of how much something costs can be as simple as the cost of purchasing a single item at a retail store or it can be as detailed as how

User Need

Validation Tasks

Verification Tasks

Requirements Development and Analysis

DIV-1 Problem Statement OV-4 CV-2 DIV-2

Assumptions, Constraints, Limitations OV-1

Frame Operational Context

OV-2 Requirements Development OV-5A

CV-3 Functional Analysis and Allocation

OV-5B SV-6 SV-5A

Derive Functions-Physical Functional Behavioral

SV-1 Determine Asset/Component

Functional Allocation SV-8

SV-4 DIV-3

Interface Understanding PV-2

SV-2

System Limitations CV-6 Design Synthesis

StdV-1 StdV-2 System Analysis OV-6C SV-7 AV-2

System Lifecycle Planning

Trades and Affordability Understanding

System Solution

Figure 2.7: DODAF Models at Early Decision Points and Milestones. Adapted from [28, pp. 89,92] and [11, Table D1 and D-E-4].

much will the next generation defense system cost over its lifecycle? While the small retail store purchase may seem trivial, the purpose and perspective of who is making the estimate are necessary, what included aspects and considerations of the item occur, and finally how long is the product intended to meet the need of the user. This observation points to the notion that cost estimation requires information or data to illustrate purpose, scope, and context before any expectation of monetary value. If the data is not available yet or requires considerable normalization to use, then documented and communicated assumptions must provide a reasonable placeholder until better information or data present itself. In their 2015 work on cost estimation methods and tools, Daniel Nussbaum and Greg Mislick point out that in cost estimation, "We are looking for an approximate answer to help us plan the expense while we sort through the various options that we may be considering" [40, p. 2]. This comment begs the question: What makes a good cost estimate?

### 2.3.1 Overview of Good Cost Estimation Techniques

The government accountability office (GAO) reports that there are nine characteristics of a good cost estimate [7], [41]. This assessment comes with over 40 years of historical information and analysis and sources from both government and public industries. The

United States federal government developed a 12-step process for accurate, reliable, and credible cost estimate outlined in [7]. The intent of the 12-step process is to help business and program decisions, support useful trade studies, and support performance baselines [7] within the requirements of applicable United States regulations, policy, and guidance [40].

As the complexity of critical defense acquisitions continues to grow, the challenge presented to cost estimating professionals is also increasing. Figure 2.8 from a GAO 2009 report highlights the balancing act of the cost estimating team [7, p. 17]. Changing the relative size or importance of any one of the boxes on the scale due to uncertainty or error has the potential to shift the apparatus. The goal, however, is accurate cost estimates that help create realistic budgets and affordable funding profiles for the United States federal government. Mislick and Nussbaum propose that in rapidly changing conditions and immature technologies commonly seen in major defense acquisitions, "a complete, reasonable, creditable, and analytically defensible cost estimate may be all that is achievable" [40, p. 2]. Any means to improve the cost data available and information efficiency among DOD agencies seems useful.
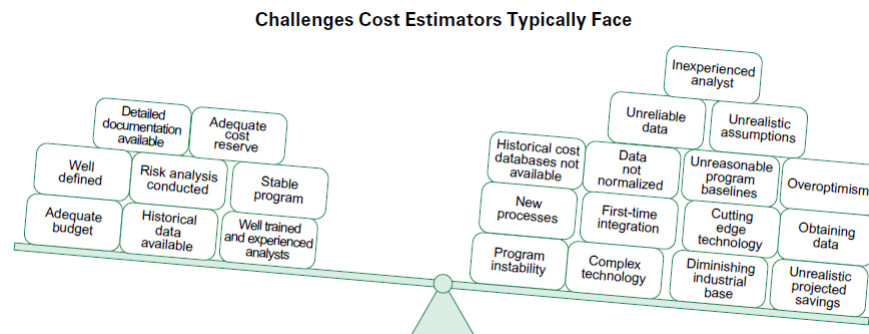


Figure 2.8: Scale of Challenges for Cost Estimators. Source: [7, p. 17].

So what contributes to poor estimates? Over-optimism, lack of data, and limited resources are most frequently cited and identified in case study reviews of cost estimation efforts [7]. Similarly, a lack of requirement understanding, haste, mixed interpretations of written or verbal communications, and indigent cost estimating techniques or methods occur in many case studies [41]. The focus of this work is in systems engineering and therefore a look at the cost estimates frequently encountered and used in systems engineering activities is prudent. One area often found in a negative view due to cost overruns and schedule slips are major defense acquisitions [4], [8], [23]. Recent critical Nunn-McCurdy breaches include

the following programs and or updates to the program: Excalibur, Joint Strike Fighter (JSF-35), MQ-8 Firescout UAV, and the Global Hawk UAV all of which occurred since FY2009 [42]–[44]. Some programs had more widespread negative attention than others did, but frequency and dollar figures associated with these breaches warrant attention.

Major defense acquisition cost estimates and methods then seem appropriate to discuss. The end goal of the defense acquisition process from an affordability perspective is to deliver the user defined need or capability at a reasonable price [7]. Validated cost estimates provide a way to determine the amount of necessary financial capital, the timing of those resources, and show progress or difficulties to achieve scheduled acquisition activities through Earned Value Management (EVM) or other means [41]. In a broad sense, cost estimates provide long-term planning, budgeting, and alternative comparisons for the DOD [40]. Over time, the uncertainty highlighted in Figure 2.9 of a program diminishes as the cost of any changes become increasingly expensive and infeasible to implement because estimates are being replaced by actual acquisition costs and are used for historical context and future estimates [16], [17], [40].
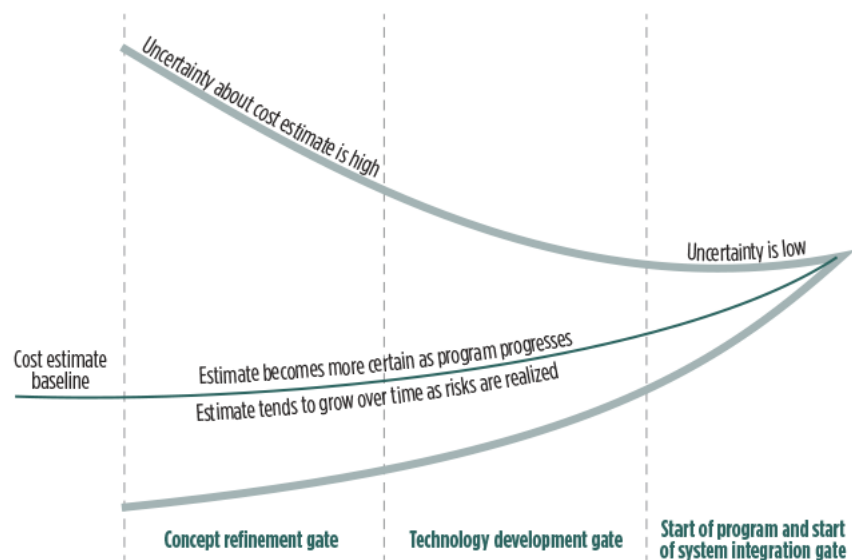
Figure 2.9: Cost Uncertainty Over Time. Source: [7, p. 38].

24

By nature, the cost estimate must maintain the pace of the changes to the program and update as necessary to reflect reality for the program. The more accurate and up to date a cost estimate, the better equipped a decision maker is to make decisions and increase the likelihood of informed decisions. The uncertainty of these cost estimates is due mostly to the assumptions, techniques, or methods used to develop the estimate. The utility of data reuse and data efficiency of model-based systems engineering can offer support to cost estimators.

### 2.3.2   Primary Cost Estimation Techniques

The following section provides a brief overview of the commonly acceptable cost estimation methods. Each description gives insight into executing the technique and highlight advantages and disadvantages of the cost estimating method. In general, terms, the cost estimation methods discussed highlight an increase in depth of information and data as the speed of developing the cost estimate decreases. This concept refers to either a top-down analogy approach (analogy method), a bottom-up approach (engineering method), or a hybrid approach (a combination of top-down and bottom-up methods) to fit specific needs. Figure 2.10 adapted from [45], [46] is a graphical illustration of these ideas with the inclusion of the related DOD modeling and systems engineering aspects. It is important to note that other cost estimation methods and models exist, but these methods also achieved acceptance in DOD cost estimation community [40] and defense acquisition and simulation domains.

### 2.3.3   Analogy Method

As the name applies, the analogy method takes the similarity of a proposed system to existing systems and scales the cost estimate for the new system appropriately under a given set of assumptions. This method provides simplicity and speed with a relatively large level of abstraction when necessary detail is unknown. The personal bias of a particular impact of the analogy and reliance on a single system or scarce data points provide some enumeration of disadvantages of this cost estimating method [40], [41].
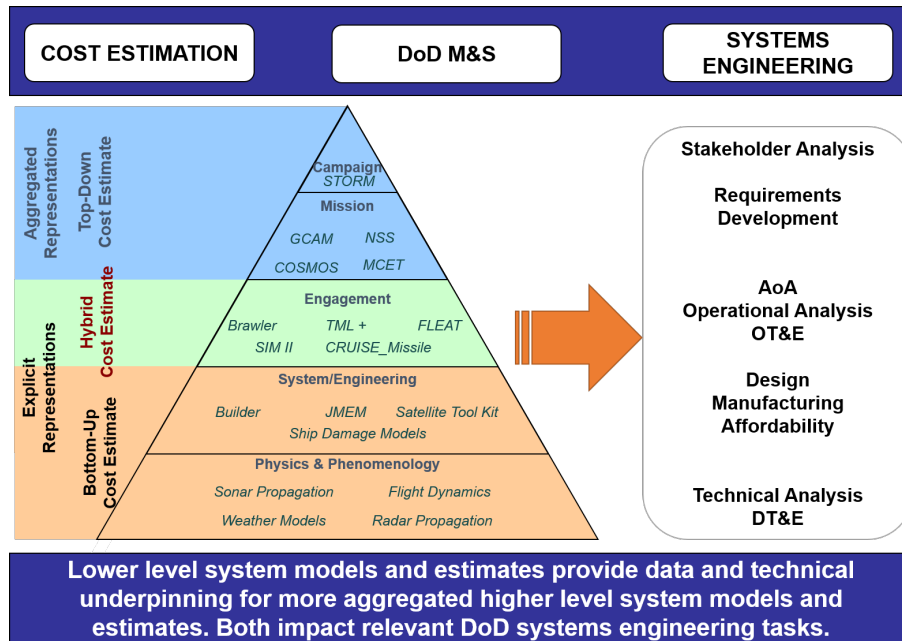
Figure 2.10: Model and Cost Complexity Level. Adapted from [45], [46].

### 2.3.4 Parametric Cost Method

The parametric analysis method also takes a historical perspective when finding information and data for cost estimation, but the documented assumptions and relationships use statistics. The cost estimating relationship (CER) is the statistical relationship between the dependent variable of cost and the independent factor of the system. This relationship is the result of statistical methods used on historical data most often some form of regression. Given its essential nature to the parametric analysis, the CER requires rigorous development and continued maintenance to ensure that a parametric estimate remains within the limits of the underlying CER data sets. The accuracy of information outside the boundaries of the CER data is questionable and could violate the estimates assumptions. COSYSMO the external cost model discussed in this thesis utilizes this cost estimation technique and follows in detail later in this chapter.

### 2.3.5 Engineering Build-Up Method

The last method discussed in this section is the engineering build-up method; the most detailed and accurate model, but also the slowest. Commonly referred to as the bottom-up methodology, this approach takes a physical description of all required tasks and estimated

26

resources for labor, materials, and other direct costs and in defense acquisitions, this is typically tied to a work breakdown structure (WBS). With the help of the engineering team and models, information transforms into a detailed justification to support an associated cost for each item of the WBS. The previously mentioned detail of such an estimate is apparent and by extension, how tedious and time-consuming this type of assessment is to build or inevitably change.

### 2.3.6 DOD Required Cost Estimates

The U.S. GAO and various cost estimating agencies from each military service provide support for developing quality cost estimates for defense acquisitions and their respective services [21]. This cost estimation support includes maintaining cost databases for the research and development costs, operation and support costs, and finally various reports, assessments, and system baselines. The legislative code and DOD acquisition process provides justification and mandated timelines for which cost estimate to complete and what if any external or third party reviews are required. Figure 2.11 highlights various funding categories throughout a system lifecycle [7] for a major defense acquisition system. A particular cost estimate occurs at each milestone, to help determine the requisite funding to support the system lifecycle [7], [16], [21]. Operating and sustainment cost are always a significant portion of the life-cycle cost, early decisions in the research and development phase have a lasting impact.

Many analysis and consultative aspects of the cost and risk understanding for U.S. defense programs involve the Cost Assessment and Program Evaluation (CAPE) office, and the approval of the CAPE director. This advisory role includes assessing alternative weapon systems and force structures, defense program alternatives development and evaluation, and the cost-effectiveness of these systems [47]. For more aspects of the history and mission of CAPE, the impacts of the 2009 Weapon System Acquisition Reform Act, and the various government cost databases and repositories, the reader is encouraged to view the OSD CAPE website at http://www.cape.osd.mil/ or refer to section 4.4 of reference [40]. The key point to highlight is that cost estimation relates directly to the defense major acquisition process, JCIDS, and decision support for the DOD both in practice and by law and relies on historical information to provide a reasonable and defendable cost estimate. Affordability concerns include both technical and non-technical risk considerations.
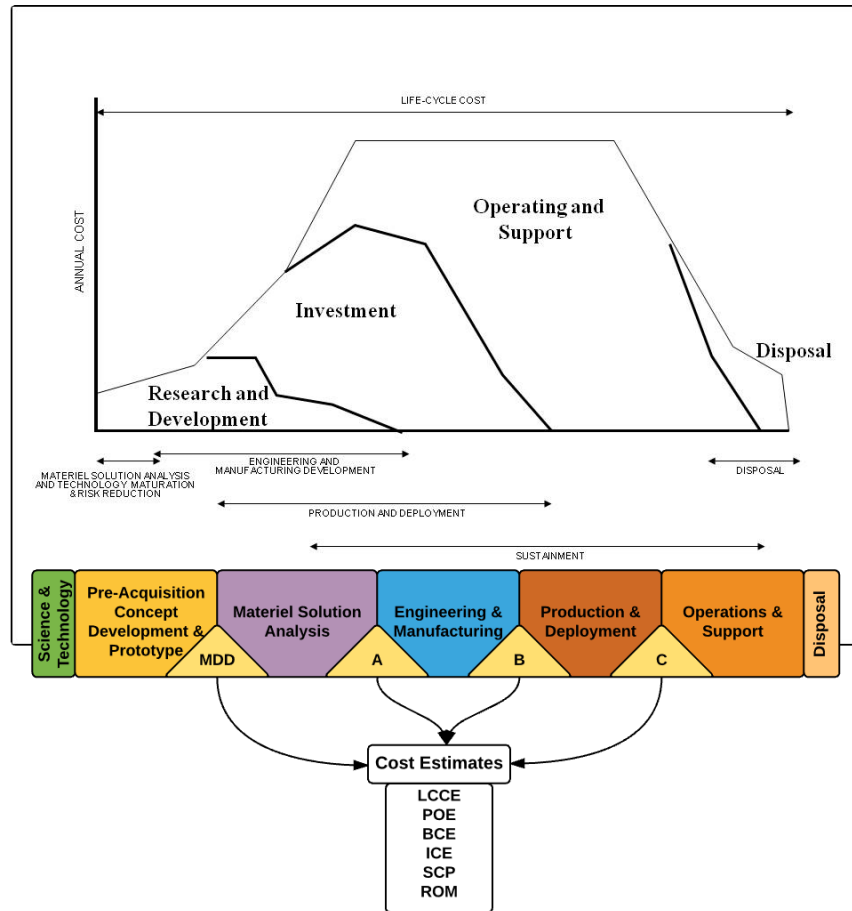
Figure 2.11: Life-Cycle Cost Example: Links to Cost Estimates. Adapted from [16, Figure 3.1.2.F1 and Figure 4.2.1.F1 ].

As a defense system progresses through the prescribed milestones and decision points of the defense acquisition process, the uncertainty of a given system diminishes over time, and iterative reviews provide opportunities to improve among other things cost estimates. The next list introduces an overview and brief description of common types of DOD cost estimates. Particular emphasis on the life cycle cost estimate occurs due to its relevance to future topics. Cost focused inquiries and studies common in systems engineering such as analysis of alternative (AoA), cost-benefit analysis (CBA), and trade-off studies augment these cost estimates.

**Life cycle cost estimation (LCCE)-** most widely produced cost estimate which provides an estimate of all program costs of a system from the conceptualization phase to

the system disposal. Often termed a "cradle to grave" [18] assessment, this cost estimation is required for each major system milestone of the Joint Capabilities Integration and Development System (JCIDS) process and aligns with the system engineering definition of lifecycle

**Program office estimate (POE)-** the local estimate of the program manager concerning a specific system

**Baseline cost estimate (BCE)-** as the name implies this cost estimate is the initial starting point or baseline for a given program. This cost estimate is a reference point for an acquisition program's variance

**Independent cost estimate (ICE)-** performed by an external and unbiased cost estimating team, this estimate is meant to evaluate an organization's cost estimating methods, check completeness, and provide credibility to previously developed estimates. All ACAT I, or programs in excess of $480 million (RDT&E) or $2.79 billion (total FY14 constant dollars) and specific ACAT II programs require an ICE from either OSD or CAPE [8], [11]

**Service cost position (SCP)-** when differences between a POE and ICE exist, the SCP provides a reconciliation of the differences to form this new estimate

**Rough order of magnitude estimate (ROM)-** gross estimate of the cost of a system that is in the early life-cycle stages with very little information about a given system

**What if exercise-** also called a sensitivity analysis; this estimate focuses on the variation of programmatic or underlying assumptions or parameters, to highlight the resulting consequence in cost terms

The discussion of life-cycle cost estimation methods, types of cost estimates, and its link to the United States' defense acquisition system highlights the compliance aspect of cost estimation and its direct tie to Milestones previously discussed in the defense acquisition system. Project maturity, system type, and project complexity will also inherently affect the information and data available to aid the cost estimation team to meet these mandated system decision points. As the project matures, the information known about the system or system of system (SoS) increases and estimating techniques may transition from an initial analogy to a more defined parametric method to a more engineering-focused approach, but not each item in the cost estimate will have such fidelity, see Figure 2.12.

In a resource-constrained environment regarding personnel or time, the most significant

contributors are focused on first and even with near perfect information, the outcome is not a single number or point estimate, but instead is an interval. This cost range is highly sensitive to the underlying assumptions used to create it, and assessment occurs after the realization of the actual cost of the system. Often this realization occurs long after the estimate is complete and reflects a transition from estimates to actual costs [40]. The beginning aspects of the MSA Phase provide opportunities for model integration and improvements for cost estimation. The integration has the potential to provide a push concept vice the current data pull concept in place. Providing the data in a fixed format, using familiar terms and acceptable cost techniques reduces the time for data normalization, keeps pace with system design changes, and links the system architecture to design.
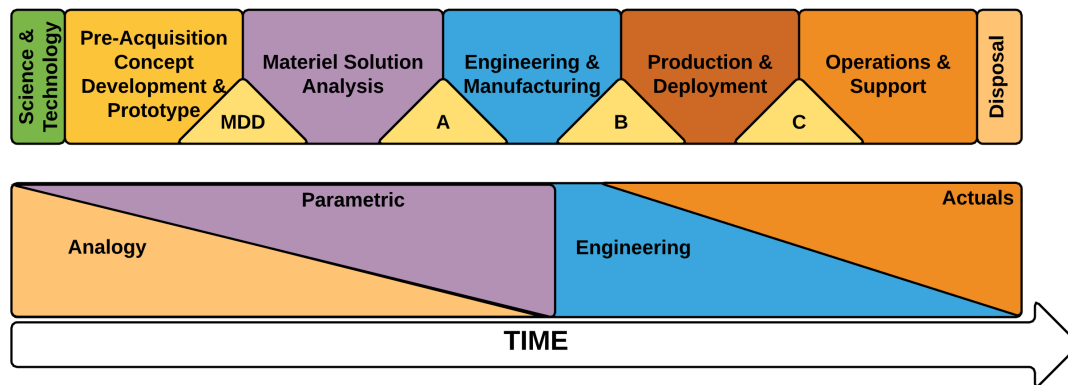


Figure 2.12: Use of Cost Estimation Techniques Over System Life-Cycle. Adapted from [16, Table 9-2 and Figure 9.6].

### 2.3.7 Systems Engineering Costs

In MIL-STD-881C, systems engineering costs are specifically listed as one of eleven elements associated with all major defense systems or common elements [48], [49]. This military standard also provides common WBSs typically utilized for several U.S. systems. Often, this WBS becomes the structure that the cost estimation team builds from, but as discussed historical information, normalization, and analysis are required to develop sufficient cost estimates. Appendix L of [48] provides detailed examples of all the activities, analysis, and costs associated with systems engineering. These tasks include a breadth of considerations from requirements, verification, and validation, to aspects of spare parts assessment and modernization, which again cover the various facets of a system lifecycle and link to system design.

The author's overall goal for this section was to provide context and discussion about the resources used in cost estimate as a matter of current methods, organizational compliance, and an assessment tool. The methods included conventional techniques and estimating processes. Compliance included adherence to regulatory guidance and mandates, but also acceptance of a standard set of acceptable cost estimating techniques for defense acquisition systems. Likewise, the assessment included analysis of long-term trends at both the DOD enterprise level down to the system component level. Data serves as a resource to help make better life-cycle cost estimates, and common to all cost estimates is systems engineering costs. With access to specific data, a cost estimate is achievable by several methods. Parametric techniques dominate early decisions.

## 2.4 COSYSMO

A constructive model brings utility and transparency to cost estimation to help decision makers. This family of models allows an estimator to understand the complexities of the effort to be completed and provides the rationale for the output of the estimate. Equation 2.1 provides the formula for the underlying core effort associated with the constructive family of models taken from [50]. Barry Boehm published the original constructive cost model in 1981 [50]–[52].

$$Effort = A * Size^B * \prod_{i=1}^{N} EM_i \qquad (2.1)$$

where:

- *Effort* is in Person-Months (PM)
- *A* is a constant derived from historical project data.
- *Size* is a measure of the work product
- *B* is an exponent for the diseconomy of scale
- $EM_i$ is an effort multiplier for the ith cost driver. The geometric product of *N* multipliers is an overall Effort Adjustment Factor *(EAF)* to the nominal effort.

Size represents the context of the specific work estimated and units depend on the work products associated with the project. The effort multipliers cover factors for product,

platform, personnel and project attributes that affect cost and are associated with risk understanding [50], [53]. The relationship between a basic schedule and an effort exist in Equation 2.2.

$$Schedule = C * Effort^D \qquad (2.2)$$

where:

- *Schedule* is in calendar months
- *C* is a constant derived from historical project data
- *D* is a constant derived from historical project data

The quotient of *Effort* over *Schedule* provides an estimate for the average number of people required to staff the project with a constant staffing level for the project duration. See Equation 2.3.

$$Average\ Staff = \frac{Effort}{Schedule} \qquad (2.3)$$

Collectively these terms can provide estimates of the necessary staffing, effort, and time required to complete a given project. Standard practices exist across the various hardware, software, and systems life-cycle phases that provide historical percentages of the total effort for each life-cycle phase. Applications, extensions, and modifications to these basic formulations are the link between the various constructive family of models and their updates. Over nearly thirty years, this group of models has evolved to meet industry needs while leveraging public and open cost estimating relationships to its decision makers, especially for software. The next section highlights the specifics of COSYSMO, which estimates systems engineering costs.

### 2.4.1   COSYSMO Origin

The Constructive Systems Engineering Cost Model (COSYSMO) originated from the 2005 doctoral dissertation of Ricardo Valerdi, and is one variant of a constructive model. He applied parametric cost estimation techniques to determine the systems engineering effort

required for a given program as a single system estimate or level. The COSYSMO research leveraged the collected knowledge base and data from industry leaders, government representatives, academic institutions and the model development process using parametric cost estimation techniques and constructive cost model designs of the early 2000s [51], namely Constructive Cost Model (COCOMO) II. These organizations provided expert information on major space, information, military aircraft, and radar systems using two variants of the Delphi technique and proven cost estimating methods. The inclusion of multiple sources and a large sector of the defense industry provided necessary support for COSYSMO as an acceptable cost estimation technique for the Defense Contract Audit Agency (DCAA) [51].

### 2.4.2 COSYSMO Parameters

Validated and acceptable cost estimating relationships (CERs) found by parametric cost estimation techniques enable rapid and repeatable cost estimates [40], [41], but the development and validation of these CERs requires large amounts of data, time, and also calibration [51]. Equation 2.4 presents COSYSMO in its mathematical form.

$$Effort = A * \left( \sum_k (w_{e,k}\Phi_{e,k} + w_{n,k}\Phi_{n,k} + w_{d,k}\Phi_{d,k}) \right)^E * \prod_{j=1}^{14} EM_j \qquad (2.4)$$

where:

- *Effort* is in Person-Months (PM) for a nominal schedule.
- *A* is a constant derived from historical project data.
- *k* = (Requirements, Interfaces, Algorithms, Scenarios)
- $w_x$ = weight for "easy", "nominal", or "difficult" size driver.
- $\Phi$ is the quantity of "k" size driver.
- *E* is an exponent for the economy or diseconomy of scale.
- $EM_j$ is an effort multiplier for the jth cost driver. The geometric product is an overall Effort Adjustment Factor (EAF) to the nominal effort.

The following sections provide a more detailed discussion on each of the variables in COSYSMO, their derivations and provide descriptions for the model parameters.

### 2.4.3 Effort

The effort or estimated amount of Person-Months is the dependent variable or result of the formulation. In the COSYSMO formulation, this provides an estimate of the labor cost for systems engineering activities. A Person-Month of effort is the equivalent of an average working month with a default of 152 hours/Person-Month [51]. Historical data calibrates the coefficients A and E using regression and constrains the estimate to within certain boundaries. The model uses ANSI/EIA 632 as the basis for estimating the percentage of the total effort by life-cycle phase [50], [51], [53].

### 2.4.4 Calibration Constant or Calibration Factor *A*

The calibration constant or factor *A* is a local calibration. The term calibrates COSYSMO to the business productivity level of the organization performing the estimate [51]. This calibration is a foundational aspect of developing CERs. In COSYSMO, *A* defaults to 38.55 without local calibration.

### 2.4.5 Size

The size input used in COSYSMO is a weighted sum of four size drivers *(k)* for systems engineering activities. These four size drivers represent the context of the specific work products for systems engineering and include the number of requirements, interfaces, algorithms, and operational scenarios for the system of interest. Mathematically, the size estimate or additive part of the constructive cost model details the weighted sum of all size drivers. This weighted sum is scaled by the exponential scaling factor *E* in Equation 2.5.

$$Size = \left( \sum_k w_e \Phi_e + w_n \Phi_n + w_d \Phi_d \right)^E \tag{2.5}$$

That scaling factor represents either an economy or diseconomy of scale for the exponential or non-linear impact on the result [51]. Additionally, Table 2.2 provides the traditional systems engineering documents or resources for finding these size drivers. For COSYSMO *(E)* is set to 1.06 by default. Each size driver input is categorized as easy *(e)*, nominal *(n)*, or difficult *(d)* and the count of each category $(\Phi_{e,n,d})$ is the total number of size driver inputs summed by these categories. The complete parameter description for each size driver

defines how to assess each driver input as easy, nominal, or difficult in [51]. A summary of each size driver is below.

**Number of System Requirements:** The quantity of requirements that includes physical, functional, performance, and service oriented features. One recommended method of finding requirements is to count the number of terms associated with contractual requirements, i.e. shalls, wills, should and mays in the system specification [50], [51].

**Number of System Interfaces:** The quantity of physical and logical boundaries shared between internal and external components and functions. ISO/IEC 15288-defines system elements and interfaces for this size driver.

**Number of Critical Algorithms:** The number of unique algorithms needed to meet the system performance levels. These algorithms require mathematical expressions and link to system performance.

**Number of Operational Scenarios:** The number of operational scenarios or threads that a system must satisfy. The number of use cases associated with a system typically estimates this size driver.

Table 2.2: COSYSMO Size Drivers and Traditional Systems Engineering Data Sources. Source: [51, Table 5 p. 35].

| Driver Name | Data Source |
|---|---|
| Number of requirements | System specification |
| Number of major interfaces | Interface control documents |
| Number of critical algorithms | System specification or model description |
| Number of operational scenarios | Test or use case documents |

### 2.4.6   Effort Multipliers

Also referred to as cost drivers, 14 factors with effort multipliers exist in COSYSMO. These 14 factors relate to a project risk taxonomy. This taxonomy includes four risk categories:

Product, Process, People, and Platform risk categories. The *Expert* COSYSMO tool [53] utilizes these risk categories. The effort multipliers provide the multiplicative effect across the system independent of the system size [51], [53]. The original five COSYSMO groupings included understanding, complexity, operations, people, and environmental factors. The grouping emerged in an effort to capture the intricacies of systems engineering versus software engineering [51]. The understanding factors captured team understanding and system familiarity of the systems engineering team for the project. Complexity factors discussed the difficulty, risk, and program-related factors of the systems engineering effort. Operations captured the systems engineering planning and execution. Finally, the people and environmental factors addressed the capability of the systems engineering team and the external factors in which the team must perform.

### 2.4.7 COSYSMO Systems Engineering Risk Categories

Each of the 14 cost drivers in COSYSMO belongs to one of the following four groups: product, process, people, and platform and are described next and summarized from [53]. Product risk considerations account for variability in a system engineering effort due to the system being under development. Product risks include requirements understanding, architecture understanding, the level of service requirements, and the technology risk. Process risk considerations account for influences of process maturity, such as CMMI, and the integration of tools to support the systems engineering [53]. People risk considerations inform decision makers on the collective team capability and experience to perform system engineering. People risks include stakeholders, teams, and their experience. Finally, platform risk considerations highlight impacts of legacy systems, migration complexity, and system design. Table 2.3 illustrates the 14 cost drivers, the discussed five groups, and the associated risk categories and show the connection between risk, the definition of the groupings, and the COSYSMO cost driver. For a complete description of each cost driver and the associated effort multiplier for each cost driver at a given rating, see [50], [51].

### 2.4.8 COSYSMO Integration-Bringing It All Together

Over the last decade, refinements of the initial model have occurred in several aspects. Refinement efforts have included calibrations, new size categories for reuse in COSYSMO 2.0 [52], introduction of future drivers in COSYSMO 3.0, and linking the COSYSMO

Table 2.3: COSYSMO Cost Drivers Groupings and Risk Categories. Adapted from [51, Table 16 and Figure 9 pp. 47-48], [53, Figure 1].

| Driver Name | Driver Grouping | Risk Category |
|---|---|---|
| Requirements Understanding | Understanding | Product |
| Architecture Understanding | Understanding | Product |
| Level of Service Requirements | Complexity | Product |
| Technology Risk | Complexity | Product |
| Process Capability | People | Process |
| Multisite Coordination | Environment | Process |
| Tool Support | Environment | Process |
| Documentation Match to Life-Cycle Needs | Complexity | Process |
| Stakeholder Team Cohesion | Understanding | People |
| Personnel Experience/Continuity | Understanding | People |
| Personnel/Team Capability | People | People |
| Number of Recursive Levels in the Design | Complexity | Platform |
| Number and Diversity of Installations/Platforms | Operations | Platform |
| Migration Complexity | Operations | Platform |

model to other tools. The size and cost drivers are based on applicable governing systems engineering standards of ANSI/EIA-632, ISO/IEC 15288, and CMMI supported by the International Council on Systems Engineering (INCOSE) [51], [52]. Yet we have not seen an overwhelming use of COSYSMO in the systems engineering aspects of the DOD despite its origins in defense industry systems, the models focus on systems engineering efforts over the system lifecycle, and its accepted method of parametric cost estimation by the DOD. Additionally, systems engineering effort has been shown to correlate directly to program schedule and performance [22] and improved cost estimation has been linked to improved risk management [4], [32], [50].

Supporters of COSYSMO offer that the mathematical rigor paired with the subjective rankings of the drivers and local calibration allow for a fine balance of both systematic

and custom features to estimate systems engineering costs using adequate documentation. Examples include extensions of the bid and proposal process [54], exploring system of systems (SoS) [55], risk add-ons with Expert COSYSMO [32], [53], [56]. Friedenthal, Moore, and Steiner specifically note that SysML can be used to model COSYSMO size drivers [12]–[14]. BAE systems and Lockheed Martin use extensions of COSYSMO to help understand systems engineering costs and also estimate total operating costs for a given system [50], [56]–[59] using this information.

Conversely, some argue that COSYSMO's rating scales for the various categories and subjective assessment leave considerable variability and that the calibration required is very dependent on the data and techniques of the cost estimator organization [60]. All, however, acknowledge the increased speed and simplicity in developing useful estimates for a given program. COSYSMO has achieved a relatively simple and data-driven method to understand key aspects of systems engineering cost. The drivers noted are easily understood and can be assessed early and often for a given system engineering effort. Figures 2.13 and 2.14 illustrate the simple data entry, user interface for COSYSMO, and sample estimate results, this works integration must provide.



Figure 2.13: COSYSMO- Example of Data Inputs Required. Source: [56].

The user can choose to input COSYSMO information from a file or simply manually input the data entries required for the tool. The results include an estimate of the systems engineering effort cost, an estimate of the effort duration in Person-Months, and finally given a labor rate, the tool will provide a point estimate of the systems engineering costs. The drop down menus provide several additional add-ins and enhances the tool. For example, if the Monte Carlo add-in is turned on, the output provides an estimate and specifies a confidence level for the system effort. If file input is selected, the tool performs an automated read of the given file for use into the tool. This version is available at http://csse.usc.edu/tools/COSYSMO.php [56]. Other tools are available at http://cosysmo.mit.edu/downloads/ [61]. These resources show the refinement and development that have occurred with COSYSMO.

**Results**
**Systems Engineering**
Effort =209.9 Person-months
Schedule = 8.8 Months
Cost = $2098963

Total Size =628 Equivalent Nominal Requirements

**Acquisition Effort Distribution (Person-Months)**

| Phase / Activity | Conceptualize | Develop | Operational Test and Evaluation | Transition to Operation |
|---|---|---|---|---|
| Acquisition and Supply | 4.1 | 7.5 | 1.9 | 1.2 |
| Technical Management | 7.9 | 13.6 | 8.9 | 5.4 |
| System Design | 21.4 | 25.2 | 10.7 | 5.7 |
| Product Realization | 4.1 | 9.4 | 10.1 | 7.9 |
| Product Evaluation | 11.7 | 17.6 | 26.0 | 9.8 |

Figure 2.14: COSYSMO- Example of Estimate Results. Source: [56].

The discussion of COSYSMO, its definition, and mathematical formulation is twofold. First, to show the large amount and breadth of information used in COSYSMO for systems engineering cost estimation, but also to highlight that given the right information or data there are potential gains in utility. Gains include the rapid and early understanding of new ideas and proposed concepts, simple quantification of tradeoffs for competing or conflicting ideas, and finally a cost estimate value for various life-cycle phases. COSYSMO was selected for integration into a model-based approach for this work due to its origin, purpose, and success in defense related industries and the success of similar models for use in software. All the necessary parts exist in isolation. Systems engineering provides the methodology to bring it all together, in the proceeding sections.

## 2.5 Model-Based Systems Engineering: A Growing Trend

Model-based or model-driven engineering is by no means new; it has been a staple in the mechanical and electrical disciplines for many years. As early as the 1980s, computer-aided design (CAD) and various circuit analysis tools provided invaluable insight into growing complexity of these domains [12]. Models are seemingly more widespread with the influence of technology and software [22]. Wynmore's 1993 work [62], highlights the mathematical theory associated with system elements and their various relationships including mathematical constructs of "coupling, classification, and model behavior." The premise of the MBSE concept is to shift to a more data-centric focus communicated by models [63]. Traditional aspects of engineering used models, but the model was simply a representation of the system that provided shared understanding. The further removed from the originator of the model, the more diluted or error prone the interpretation became. MBSE enhances the utility of the system representation by infusing the graphical representation of information that can be queried, analyzed, reused, and communicated at the necessary level directly from the model.

The system model spans assessment at a managerial or business viewpoint down to a very detailed technical view using the same data. A representative model expert maintains configuration control, but recurring analysis and reports generate directly from the real-time model of the system. In addition to this near real-time data pull, standardization of computer data exchange and the syntax and semantics of various languages, allow for efficient storage in model databases or repositories for extensions of the data.

As an example, consider a system model of an unmanned aerial system. As the systems engineers, step through their organizational process to evaluate, synthesize, and analyze alternatives [17], various propulsion and structural considerations can be passed to external mechanical engineering models while other life-cycle considerations such as reliability, availability, maintainability can be assessed using the same model data. This simple example shows how MBSE can help support trade studies, analyze potential performance, assess impacts of design alternatives, and finally, verify requirements, but also begs the questions of what makes a good MBSE model, and what languages and tool support MBSE? Both topics follow in the next sections.

### 2.5.1 What Makes A Good MBSE Model

Systems modelers use several approaches. The approach is typically a tailored response to the domain, the organization's systems view, and focus. The focus scales from product and services to enterprises and systems of systems [24]. In his 2008 survey of MBSE methodologies, Estefan highlights six different vendors and their associated methods [64]. Despite the varying use of the distinction between process, method, and tool, each of the six methodologies falls into either a traditional functional decomposition, state or scenario analysis, or a hybrid of the two dependent on the mix of software, hardware, and other systems. Each of the methods specifies the level of object-oriented or traditional modeling approach, but the underlying use of ontologies, languages, and governing standards wildly vary across those surveyed MBSE approaches.

The common thread of the methodologies was on the purpose of performing systems engineering, which reinforces that systems engineers are required to be conversant in a "wide range of modeling languages, tools, and techniques" [65, p. 1], [66]. The author adds, and flexible to continuous change, to this quote after surveying MBSE languages which included UML, SysML, lifecycle modeling language (LML), Business Process Model Notation (BPMN) to name a few. Each of these languages has specific lifecycles and limitations to consider. The use of a customized ensemble of models brings value.

To aid in the assessment of modeling languages and various tools, Friedenthal and Burkhart [67] and Vaneman [66] support the following eight effectiveness measures shown in Table 2.4 to compare available MBSE tools and languages. These measures are consistent with other sources, but provide a more complete listing of such measures [12]–[14], [63], [68]. The key takeaway is that an organization's MBSE approach, tool, and language will affect an organization's achievable options. Specifically, there may be various modeling techniques that incorporate model extensions, analysis, and reuse even though a more succinct or simple model is available. There is inherent loss or trade by the organization due to the model purpose, system domain, and stakeholder needs [35]. Considerations for these model language impacts and the certainty of change must occur, and for the DOD, history supports that efficient resource utilization and compliance to the various policies and regulations are important. A balance of the model scope and granularity is critical.

Table 2.4: MBSE Effectiveness Measures. Source: [67, p. 40].

| Effectiveness Measure | Explanation |
|---|---|
| Expressiveness | assessment of how well the model or language expresses the system concept |
| Precise | assessment of the ambiguity in the system model |
| Presentation / Communication | assessment of system model communication to diverse stakeholders |
| Model Construction | assessment of model intuitiveness and efficiency of construction |
| Interoperable | assessment of model ability to exchange and transform data with other models and structured data types |
| Manageable | assessment of model ability to change and manage various portions of the model |
| Usable | assessment of model use by stakeholder's in terms of efficiency and intuitiveness |
| Adaptable / Customizable | assessment of the model scalability to extend to support various domain concepts and terminology |

## 2.5.2  MBSE Benefits and Concerns: Finding a Balance For The DOD

This section provides an overview of model-based systems engineering as a technique that offers potential benefits of enhanced communication, reduce design risk, and improved productivity and quality, but is not without trades. Model breadth, depth, and fidelity are competing against model clarity, consistency, and utility required. The detailed discussion of specific vendor tools and languages did not occur, but one can infer that several combinations of systems engineering process, modeling tools, and languages can yield unique but limited responses. Many may also assert that the MBSE approach is relatively transparent and an enhancement to meet the digital age, but counter that the various languages and models might be great for engineers, but confuse decision makers [28]. Similar to understanding a system design space, certain combinations of tools, languages, and processes will require understanding and translation across various domains for model integration and communication efforts. The correct mix of tools, languages, and processes for the defense system acquisitions process is needed, but still developing in the DOD.

## 2.6 Systems Modeling Language: One of Many Available Modeling Languages

The following section serves as an overview of Object Management Group Systems Modeling Language (OMG SysML), for a complete and detailed discussion, see [65]. As the full name implies, OMG developed the systems modeling language in response to the specific needs of systems engineers who had to use a broad range of modeling languages, tools, and techniques when working with multi-disciplinary teams on large complex projects. The language specification states that SysML as it is more commonly referred to was developed to support the "specification, analysis, design, verification, and validation" of complex "hardware, software, information, process, personnel and facility" systems [65, p. 1]. The language reuses portions of UML 2 a predominate software development language [9], [12]–[14], [65], [68], and supplements this foundational language with the necessary extensions to address systems engineering requirements. The language's aim is a general-purpose modeling language that is effective at system requirements specification, system structure modeling, system behavior modeling, functional and physical allocations, and finally constraint modeling [65, p. 1]. The goal of SysML is a powerful, interoperable language that meets the needs of systems engineers.

The language took inputs and considerations from several major defense industry leaders, multiple U.S. government organizations including the Office of the Secretary of Defense (OSD), several modeling tool vendors, and multiple standardizations and governing bodies including the INCOSE [65, pp. 5-6]. The language is in its fourth revision. Find the current specification along with machine consumable files at www.omg.org. SysML typically uses a generic block or node with various notation and connectors or paths to represent systems, components, and other entities in a diagram. Variations of these model elements can represent the following entities listed in Table 2.5 [12]–[14].

### 2.6.1 Diagrams

SysML communicates using nine unique diagram types. Figure 2.15 summarizes each of the nine diagrams adapted from [12]–[14], [65], [68] and provides the model name and abbreviation, its purpose, and basic examples to help understand the diagram. Note the orange and gray distinction between structure and behavior diagrams in SysML. This delineation will be used later to distinguish links between DODAF, COSYSMO, and SysML.

Table 2.5: Key Representations of SysML. Adapted from [12]–[14].

| Model Feature | Representation Entities |
|---|---|
| Structure | composition, interconnection, and classification |
| Behavior | functional-based, message-based, or state-based |
| Constraints | physical and performance based |
| Allocation | physical, functional, behavioral, constraints |
| Requirements | identification, relationships with others requirements design elements and test cases |
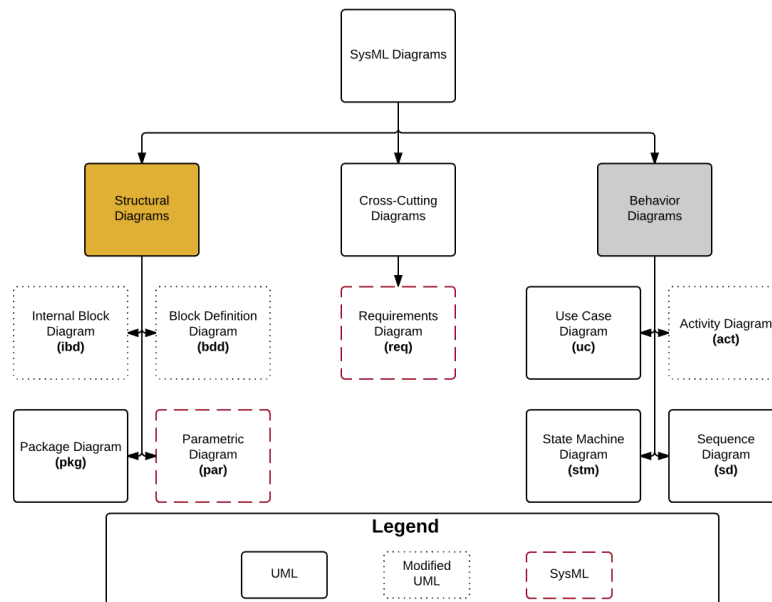


Figure 2.15: SYSML Models. Adapted from [12]–[14], [65], [68].

Each diagram has an acceptable representation of blocks, symbols, and notation that ensure compliance with the modeling language syntax. All nine of the diagrams can exist in tabular form, and the use case, sequence, state machine, and package diagram are the same as UML 2. The parametric and requirement diagrams are unique to SysML, and the activity, block definition, and internal block diagrams are modified UML diagrams.

**Activity Diagram (act)** is a behavior diagram that shows the ordering of actions based on the availability of inputs, outputs, control, and the actions which govern such behavior (modified from

| | |
|---|---|
| | UML) |
| **Sequence Diagram (sd)** | is a behavior diagram that uses a sequence of messages to convey the exchanges between parts or entities (UML diagram) |
| **State Machine Diagram (stm)** | is a behavior diagram that uses transitions between stages triggered by events to represent entities (UML diagram) |
| **Use Case Diagram (uc)** | is a behavior diagram that highlights the use of a system or entity by external entities or actors to accomplish a goal (UML diagram) |
| **Block Definition Diagram (bdd)** | is a structure diagram that uses blocks as structural elements to highlight their parts and classify these entities (modified UML diagram) |
| **Internal Block Diagram (ibd)** | is a structure diagram that represents the interfaces and connections between entities and their parts. This diagram uses ports and flows which highlight direction and source of flow (modified from UML) |
| **Parametric Diagram (par)** | is a structure diagram that demonstrates the constraints of various elements in a given diagram. This diagram is primarily used for engineering analysis. A common example is Newton's Second Law of Motion, Force=mass*acceleration or F=ma. Where a parametric diagram can provide the specific details which govern this formulation (specific to SysML) |
| **Package Diagram (pkg)** | is a structure diagram that allows for the partitioning and organization of models in terms of the elements in the package (UML diagram) |
| **Requirement Diagram (req)** | is the only diagram of its type. This diagram shows allows for traceability of requirements with other requirements, system elements, and test cases (specific to SysML) |

### 2.6.2 MBSE with SysML

There is no cure-all process for systems engineering. Instead, the defined problem, its stakeholders, and time available should drive each systems engineering endeavor. This scenario applies to the ideal, but not the current state of most organizations. If the organization has a systems engineering team at all, the process is typically a tailored form dominated by a particular domain, a tool, or an organizational culture [68]. While not the only language, SysML provides a way to identify, analyze, and communicate system level information among diverse teams without imposing a particular process. It also provides necessary granularity for analysis. As discussed in the MBSE section of this work, models have become a significant part of the engineering effort, and INCOSE went as far as to name MBSE as the future of systems engineering [3].

SysML then is one language to capture pertinent system information in a model form to represent a system. The collective team or organizational policy can determine what is required to meet a refined user need; SysML can be the standard method to describe the various facets of the problem and provide analysis. This idea highlights a key point, systems modeling tools like SysML do not stand alone, and instead, must interact with several other external tools, data exchange types, and domains. Likewise, it highlights that any SysML model is only as good as its true representation of the system. In addition, when modeling in SysML a clear purpose is required. Said more plainly, an accurate model is not necessarily a correct model. Likewise, a correct model may not serve the intended purpose. Figure 2.16 illustrates the MBSE approach taken by the author for this work influenced by Blanchard and Fabrycky's synthesize, analyze, and evaluate construct from [17]. Note the blue, yellow, and green arrows associated with system engineering tasks. This color scheme links specific systems engineering tasks to DODAF and SysML models in later discussions.

Two areas SysML has shown considerable utility for systems engineering is structured functional analysis or decomposition and use case or design reference missions [12]–[14], [68]. Performing the different techniques on the same system result in different diagram combinations, but each highlights a particular purpose. Due to its origin, SysML diagrams present the graphical representation of systems engineering tools in a way that performs common system engineering tasks, but the rigor and relationships of the underlying data allow additional performance analysis through executable models or reuse of the data and data relationships in external models. The latter is the focus for the remainder of this work.
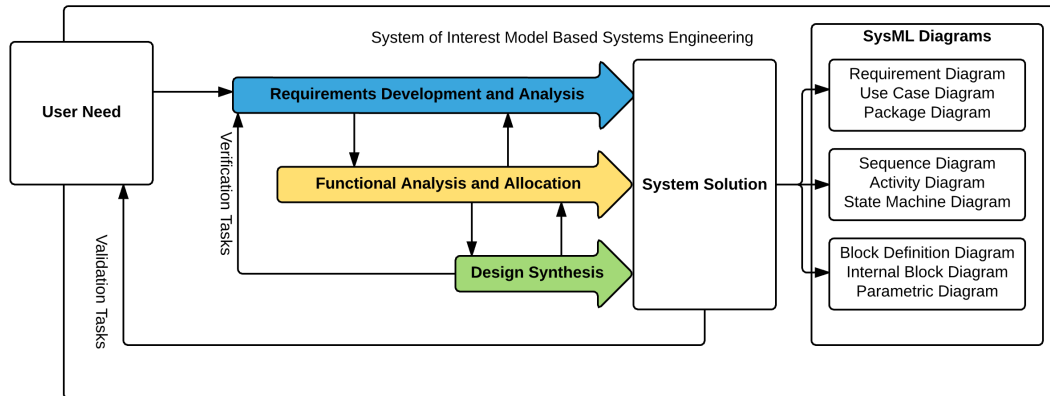
Figure 2.16: SYSML Diagram Link to MBSE Implementation.

## 2.6.3 SysML Links to COSYSMO
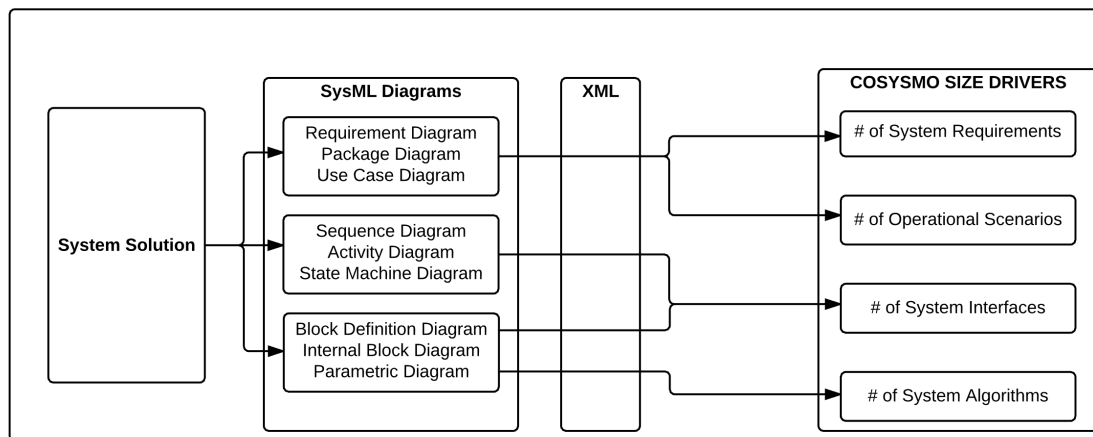


Figure 2.17: SYSML Model Links to COSYSMO Size Drivers.

COSYSMO uses 18 parameters and as discussed, includes four size drivers and 14 cost drivers. For the size drivers, requirements, interfaces, critical algorithms, and operational scenarios are counted and grouped by an assessed difficulty of implementation. Figure 2.17 provides a mapping of which SysML diagrams provide context for COSYSMO parameters. Recalling the general concepts of requirement development and analysis, functional development, and allocation, and design synthesis and analysis from Figure 2.16, we see that certain aspects of SysML diagram types include aspects of COSYSMO size drivers. For example, package diagrams can provide the partitioning of requirements into understandable and communicable segments, but the requirements diagrams show the links and relationships of various requirements for traceability. It seems logical then that these two diagrams

47

may provide information to estimate COSYSMO system requirements. Like- wise, under-standing the parametric diagrams of a system model may include information concerning system algorithms. The exploration of these suggested mappings between MBSE, SysML, and COSYSMO occurs in detail in Chapter 3 and is the foundation of the integration method proposed.

## 2.6.4   SysML Links to DODAF
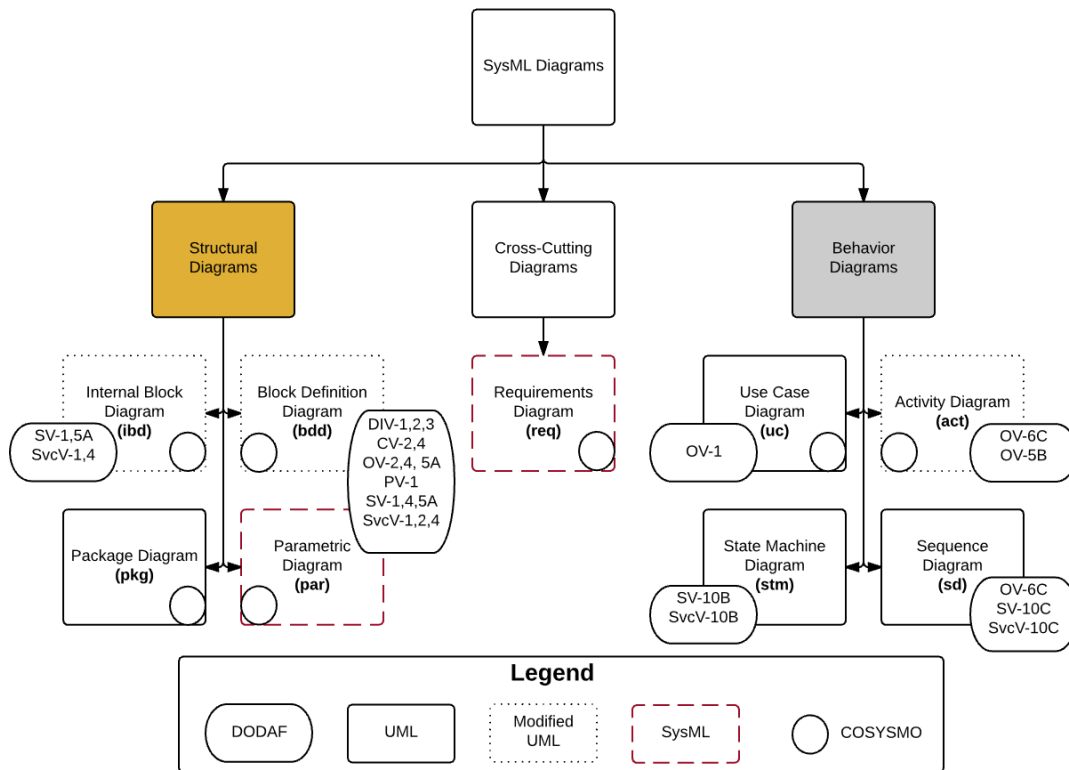


Figure 2.18: SYSML Model Links to JCIDS Required DODAF Models.

Figure 2.18 provides a context for how the JCIDS required models of DODAF version 2.02 relate to SysML, adapted from [9] a 2012 mapping of the two. The base layer includes the nine SysML diagrams, also included are the 23 models required for the CDD in the JCIDS process, recalling the JCIDS duplication of the OV-2 and SV-7 in its 2015 revision. The CDD is representative of the system understanding during the MSA activities of early DOD trade decisions. Originally mapped to UML in previous versions of the DOD architectural framework, we see links to SysML constructs that are identical to UML and

those that are modifications of UML discussed in Figure 2.15. Note the use case, internal block diagram, block definition diagram, and activity diagram frequency. The criticism of DODAF's poor performance analysis ability also seems evident, as there is no link to the SysML parametric diagram. These mappings suggest that similar model constructs and communication intentions exist between the DODAF, COSYSMO, and SysML constructs. The use of universal terms across the various defense organizations in which the ontological structure and the deep analysis capability occurs does not exist. Model trades and language mappings are fulfilling those voids. The next chapter discusses these gaps and focuses on bringing aspects of MBSE, SysML, and COSYSMO to support early trade decisions. Figure 2.19 provides a mapping of DODAF models related to the MBSE methodology presented. The orange items denote the DODAF diagrams that relate to SysML structure diagrams while the gray items denote those DODAF diagrams related to the behavioral diagrams in SysML. Lastly, the white items indicate those DODAF diagrams without a particular SysML link although many sources [9], [12]–[14], [28], [68] suggest using the matrix and tabular tools in SysML to meet these information requirements for DODAF.
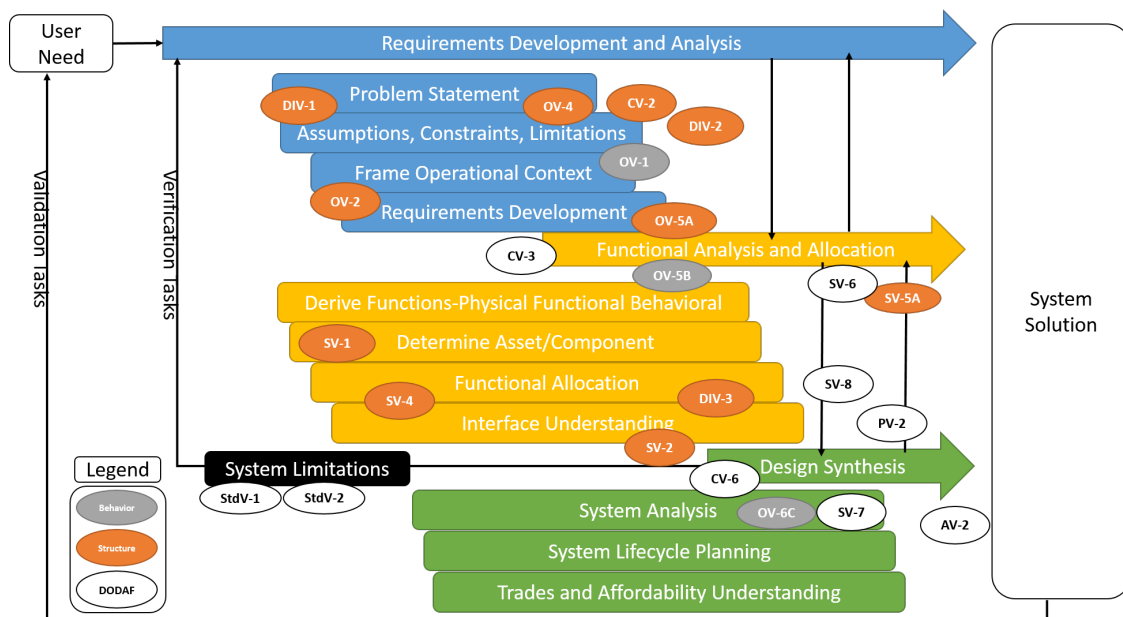


Figure 2.19: Links of MBSE Methodology and DODAF. Adapted from [28, pp. 89,92].

49

## 2.7 Chapter Summary: Background

Complexity in the defense industry requires increased interoperability for its defense systems. This organizational need requires a diverse set of domains and perspectives to help find an able, agile, and affordable solution. Traditional systems engineering for the DOD and many defense industries are shifting focus to a model-based systems engineering approach. Several languages, modeling techniques, and ontological approaches are under development but are also continuously improving to support this model-based approach. No single method, tool, or ontology has proved to be dominate in systems engineering. The aggregation or fusion of the various methods, tools, and ontological constructs shows benefit. Early trade space decisions influenced by system engineer input and analysis are critical during the MSA Phase of major defense system acquisition. Data reuse, the speed of access, and ability to keep pace with defense systems changes, are key benefits of integrating the COSYSMO into the MSA Phase of the acquisitions process. The aim is to increase defense system life-cycle cost awareness and fidelity early in the MDAP to identify cost risk. SysML is a current modeling language with distinguishable links to MBSE, COSYSMO, and DODAF and a general theme of integration and understanding cost impacts are stressed to support bringing the ideas together in Chapter 3 of this work.

# CHAPTER 3:
## Methodology: Bringing It All Together

## 3.1 Overview

This chapter outlines the methodology applied to the modeling aspects of this thesis with a focus on external model integration. An assumption of general understanding of the concepts and topics presented in the background discussion concerning MBSE, DODAF, COSYSMO, and SysML occurs. The model-based system approach discussed follows the DOD specific systems engineering tasks presented in Chapter 2. The approach mirrors Chapters 15 and 16 of Sanford Friedenthal, Alan Moore, and Rick Steiner's three editions of "A Practical Guide to SysML" [12]–[14] because the approach uses the same water distiller system documentation, data available from OMG, and figures from the publisher Elsevier. In addition, the guides, [12]–[14], are a current resource for developing system modeling techniques. A full discussion of the development of the water distiller and the iterative refinements that Friedenthal, Moore, and Steiner took to get to this point obviously requires an entire book on its own, but utilizing their validated model provides the focus to be on model integration and not modeling itself. Model integration requires a validated model to begin work. Elsevier provided copyright permissions for the use of the water distiller figures presented.

The water distiller humanitarian scenario provides an easily understood concept that highlights the necessary and sufficient steps to illustrate identification and extraction of COSYSMO size drivers from SysML models. The author had no part in creating this data or original model. Instead, the only contributions made were using the models and various representations of the information to identify and highlight relevant COSYSMO parameters. Minor modifications and model extensions provide amplifying or clarifying comments in the chosen modeling tool. This utility highlights the practicality and reuse aspect of MBSE and carefully structured data. This chapter serves as an overview and record of the actions taken to pull useful COSYSMO parameters from the water distiller model. Data output and analysis section follow this chapter.

The aim of this work is to understand the current state of COSYSMO and its use for the DOD. Thus far, that understanding has been qualitative in both the brief discussion of the formation of COSYSMO, introduction to the foundational concept of parametric cost estimation, and finally identifying current pre-Milestone A decision support and analysis efforts COSYSMO could augment. The next section will transition to a quantitative approach to determine what specific model entities can produce estimators of COSYSMO parameters from a representative system model. Figure 3.1 provides a conceptual overview of the method used, the system of interest, and external model integration. The information that follows discusses the MBSE techniques employed, the model manipulation and enhancement, and concludes with a recommendation on which model aspects are useful as COSYSMO parameters.
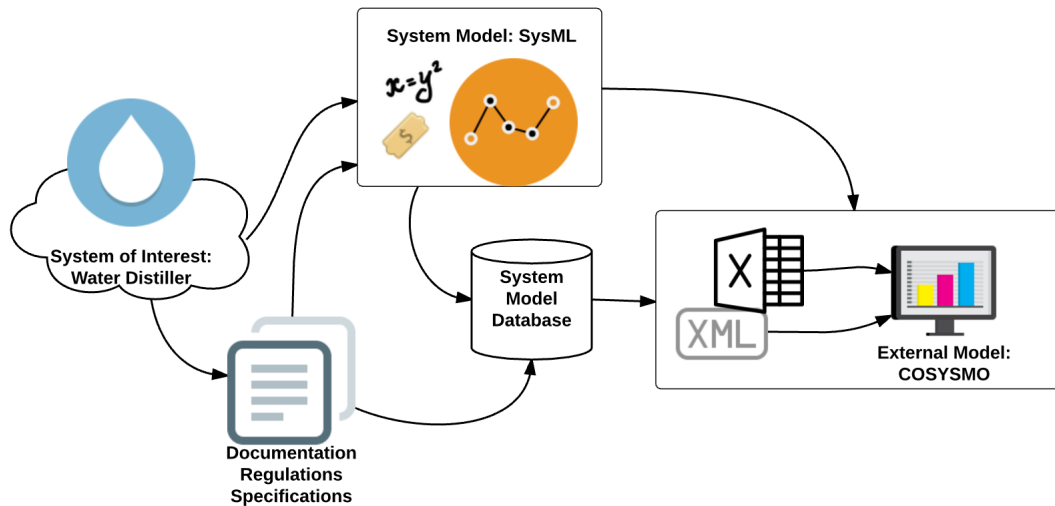


Figure 3.1: Conceptual Overview of External Cost Model Integration.

### 3.1.1 Methodology Specific Assumptions

The following assumptions applied to the water distiller scenario for consideration and use in the exploration of COSYSMO integration.

- COSYSMO calibration occurred from the industry or organizations developing the system of interest.
- The model presented is the current understanding of the system of interest.
- The techniques used will allow adequate scalability to a larger more complex system.
- The mapping between DODAF, SysML, and LML from [9] and Innoslate's [33] are valid.

### 3.1.2 Summary of Methodology

The six-step process detailed next provides a summary of the MBSE approach and follow-on integration of COSYSMO using a system model. An expansion of each of the six steps describes the specific actions taken. When necessary, background information ensures clarity for the reader, as many of the terms of systems engineering, SysML, DODAF deviate in different contexts by the various sources. One example includes the terms Measure of Effectiveness (MOE) and Measure of Performance (MOP) between the test and evaluation and systems engineering community [28] for the DOD. Additionally, the modeling tool of choice is Innoslate, which presents minor differences in presentation and has an LML specific schema. For example, package diagrams in Innoslate have connectors. In the SysML specification, these lines do not exist, although links between packages do.

**Step 1.** Identify a system of interest and selected modeling tool.

**Step 2.** Use MBSE approach to explore the models purpose and granularity for the system of interest. Highlight the SysML model links to COSYSMO size or cost drivers at that level.

**Step 3.** Identify a COSYSMO size or cost driver to estimate.

**Step 4.** Locate and document the system model entities and relationships that best estimate the size or cost driver from Steps 2 and 3.

**Step 5.** Count COSYSMO size or cost driver estimates from the models given Step 4 in a manual or automated process.

**Step 6.** Pass information to COSYSMO for an estimate of the systems engineering costs.

## 3.2 Model Selection: A Valid and Creditable Source

To explore the integration of COSYSMO and enable far reaching understanding across various levels of modeling and systems engineering experience, the author selected the water distiller and a traditional functional analysis originally used for the SysML specification [12]–[14]. It is one that has commonly appeared in SysML tutorials over the last decade. The water distiller scenario provides adequate system internal and external exploration and represents the modeling detail known for pre-Milestone A decisions. Next, because the model supported the initial evaluation of SysML, the information to create the water distiller system model is in a repository by OMG [15] and available from their website [65]. Using the water distiller example provides a means to focus on the COSYSMO integration

versus modeling. Additionally, all of the previously discussed mappings of COSYSMO, SysML, and DODAF diagrams from Chapter 2 are easily related and reinforced.

A few caveats about this information and about selecting the distiller example. First, the originator of this model performed the MBSE approach using a SysML plugin from Magic Draw, but the information exists in hypertext markup language (HTML) format for the reader's selected tool. Additionally, alternative models were considered and explored. Original model integration efforts included teaming with members of the Air Force Institute of Technology (AFIT) who were developing two separate, but related UAV system models. The first model by Ryan Pospisal, explored an executable architecture to compare three alternative architectural designs for requirements development and analysis [69]. The single modeler's purpose was creating an executable architecture and showing the utility as a system engineering activity. The resulting OV-6C showed statistical significance in the various architectures, but due to the scope and purpose of the model, only operational considerations emerged.

The second UAV model was a seven-person modeling effort and included over 1,650 and growing model entities. This model also focused on the operational considerations of a multi-tiered UAV system but had a larger scope. The publishing timeline for the second UAV model and this thesis work did not align properly. While this model has considerable potential, the timing of the model would not allow for concurrent exploration of the model in line with an initial development concept for external model integration. Additionally, with such a vast scope, configuration management, and change control was a challenge until responsibilities transferred to a sole member who is expanding the model for future thesis research. This model has the potential to serve as a source for future application and refinement of the discussed approach.

## 3.3   Tool Selection: Select A Tool, Know Its Limits

The selected tool was Innoslate from Systems and Proposal Engineering Company (SPEC) innovations. The modeling tool provided a graphical based interface, ability to switch between various modeling languages in the same modeling tool across a large team, the tools inclusion of DODAF and the IDEAS foundational schema, and finally, this author, had experience with the tool. The author has used ViTech Core, IBM Rationale, Magic Draw,

and EA Sparx and considered each alternative, but availability and licensing for many of these modeling tools proved cost prohibitive for a single model excursion and discussion, despite similar modeling experience with these tools. Each tool, however, provided a set of strengths and weaknesses. The need was for a SysML capable model, which supported MBSE and was conducive for illustrating COSYSMO factors and DODAF. Innoslate is sufficient, but uses LML, and thus some modeling decisions were required to translate LML entities and labeling conventions into SysML equivalents. For the complete specification of LML [33] and its application in Innoslate, please see the Innoslate documentation at www.innoslate.com [70]. Appendix A and B of the LML specification provide a precise translation of differences between the two languages, and the Innoslate webinar tutorials help explain how to express the various types of SysML diagrams. Table 3.1 is a summary of the translations between the two languages and their entities from [33, p. 61]. The validity of this mapping is a critical assumption for this work.

Table 3.1: SysML Diagram Mapping to LML Diagrams and Ontology. Source: [33, p. 61]

| SysML Diagram | LML Diagram | LML Entity |
|---|---|---|
| Activity Diagram | Action Diagram | Action, Input/Output |
| Sequence Diagram | Sequence Diagram | Action, Asset |
| State Machine | State Machine | Characteristic (State), Action (Event) |
| Use Case Diagram | Asset Diagram | Asset, Connection |
| Block Definition Diagram | Class Diagram, Hierarchy Chart | Input/Output (Data Class), Action (Method), Characteristic (Property) |
| Internal Block Diagram | Asset Diagram | Asset, Connection |
| Package Diagram | Asset Diagram | Asset, Connection |
| Parametric Diagram | Hierarchy, Spider, Radar Diagrams | Characteristic |
| Requirement Diagram | Hierarchy, Spider | Requirement and Related Entities |

Innoslate has several useful features. One benefit is the tool's ease of collaboration and

graphical interface. Import tools augment this benefit. Specifically, Innoslate allows for imports of IBM Rational Doors, plain text, pdf, and word files, and some XML types [70] The dashboard and various report executions allow for quick and informative queries and analysis of the model. Finally, the model supports some discrete event, eight probability distributions, and Monte Carlo simulation techniques. While the DODAF diagrams and exporting XML output files are extremely useful, there is a translation loss between the default LML schema and DODAF. This loss affects the ability to create a SysML parametric diagram and then exporting the model into the PES required by DODAF or AV-2. Specifically, the DODAF schema does not understand the port and equation aspects because the terms are not in the DM2. Innoslate uses a new class entity in the LML schema to accommodate these specific terms [33, p. 61].

Other translation losses occur, but this is the only translation loss that affected this work. The LML specification [33] and Innoslate's user guide [70] do not specifically discuss how the equation model entity fits into LML despite its inclusion in Innoslate's list of available entities, and at publishing, a full ontology is still under development. Additionally, although most DODAF models are achievable in Innoslate, not all models are templated in the tool. To account for ambiguity or translation, Appendix A reproduced in Table 3.1 and Appendix B of the LML specification [33] were used when relating entities back to DODAF or translating into SysML from LML tools. Finally, projects originated in Innoslate follow a user specified schema. This schema can be either the default of the model type selected or a custom schema. The Innoslate schema on a given system model is customizable for a particular project and encouraged by the Innoslate team [70] to support custom schemas. The water distiller project did not require this modification. With any specification, expect user feedback to prompt necessary revisions in the future. During this work, an update of Innoslate occurred. Backward compatibility with the previous model exists in the modeling tool, but items and labels in the schema changed see Section 3.9 and Appendix: Select XML Output.

## 3.4    MBSE Approach with Innoslate: A Water Distiller

The following section will provide sufficient detail to highlight the water distiller system, its purpose, and perform a functional analysis of the system with SysML diagrams. Where applicable the author will highlight the links of a particular diagram or its purpose to

COSYSMO and DODAF and discuss and modifications for use in Innoslate.

### 3.4.1 Problem Statement

A need for clean drinking water exists as a problem for a humanitarian relief agency. Stakeholder analysis and user feedback suggest that water is typically available in many impoverished areas of concern for the organization, but the water contaminated is unsafe to drink. The shipping of potable water for the humanitarian relief agency is unsustainable for long-term relief efforts, and the use of filtering systems is unacceptable. Previous humanitarian efforts involving replacement parts found that both access to replacement filters in remote areas and the cost of such filters as prohibitive. One early alternative proposes a low cost, simple, and adaptable personal water distiller system for use in impoverished and remote areas [12]–[14]. See Figures 3.2 and 3.3.
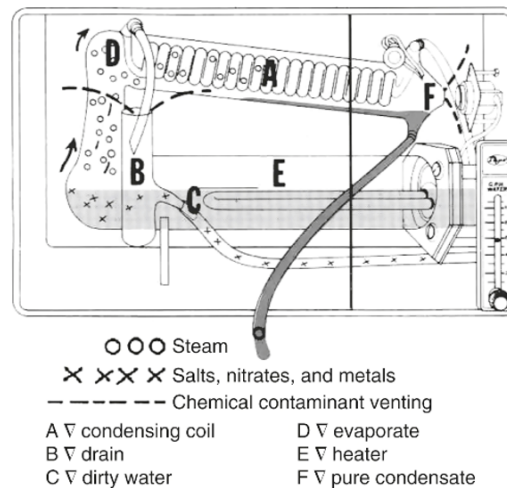


Figure 3.2: Example of a Distiller Process. Source: [13]. Reprinted with Permission.

This analysis will focus on the development and modeling of the water distiller scenario for the humanitarian relief agency. Figure 3.2 provides an initial understanding of the proposed functionality of the water distiller system and an example of a continuous water distiller. In contrast, Figure 3.3 represents a batch type water distiller. These figures are representative of system alternatives for a preferred solution that utilize heat source style distillers over filter type distillers. A comparison of Figures 3.2 and 3.3 highlights more

granularity in continuous distiller graphic than the batch distiller example. These figures enhance communication to stakeholders about the differences between the two solution alternatives of the batch versus the continuous distiller and the variation in the design type life-cycle considerations, especially when presented as OV-1 graphics.



Figure 3.3: Example of a Simple Batch Distiller. Source: [13]. Reprinted with Permission.

### 3.4.2   Modeling Organization and Partitioning

To provide a simple to follow and consistent frame of reference for discussion, the following package diagram, Figure 3.4, highlights a partitioning of the modeling effort to support the purpose of incorporating an external model. This model organization is critical to the clear and consistent representation of information in the project and separates those particular entities necessary for COSYSMO integration. Note the import annotated on the SI Definitions folder; this SysML marking denotes that a reusable library for the common International System of Units in this folder.

Each folder icon represents an office file or in the case of DODAF a viewpoint. A project manager or leader would consider these folders as the breakdown of his or her integrated project teams (IPT) assigned or analysis organizations. When a user opens a specific folder or package, he or she will see various diagrams to support that particular perspective. For example, if the user selected the distiller structure icon, he or she might expect to see Figure 3.2 and Figure 3.3, as they convey information about possible structures. Additionally, he or she would expect to see diagrams and models that address form such as the activity,
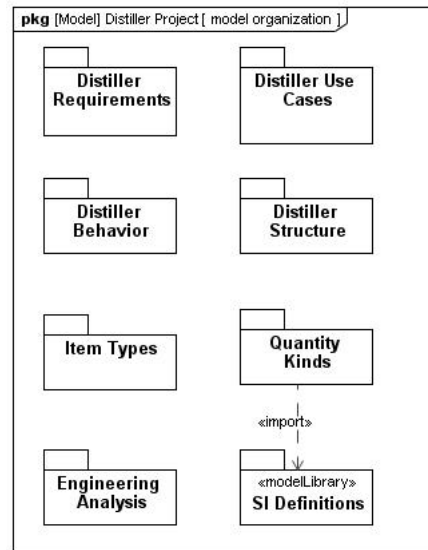
Figure 3.4: SysML Package Diagram Model Organization. Source: [13]. Reprinted with Permission.

sequence, state machine, and use case diagrams in SysML or the OV, SV, and SvcV models in DODAF. Figures 3.2 and 3.3 can be thought of as high-level representations whereas the block definition diagram (bdd) in Figure 3.5 could be at a more detailed and technical level for the batch distiller image. Comparison of Figures 3.3 and 3.5 shows both the possible perspectives of each user and how that perspective requires different representations of the same information. One view may need aspects of physical dimensions and weights included in the system description, where another user may need intricacies of the internal system to understand maintenance and supportability.

### 3.4.3 Requirements Development

A collection of source requirements documents for the humanitarian relief agency identifies key aspects of the water distiller. These materials include known information about the human relief agency organization and its mission, the refined need of the water distiller to achieve its humanitarian support mission, and finally mission effectiveness and distiller requirements. These documents and the requirements that populate them represent the results of current analysis efforts during the MSA Phase for the DOD. Recalling our folder analogy, Figure 3.6 shows the use of the SysML package diagram to organize the water distiller design for requirements development. Anyone can separate or categorize these

59
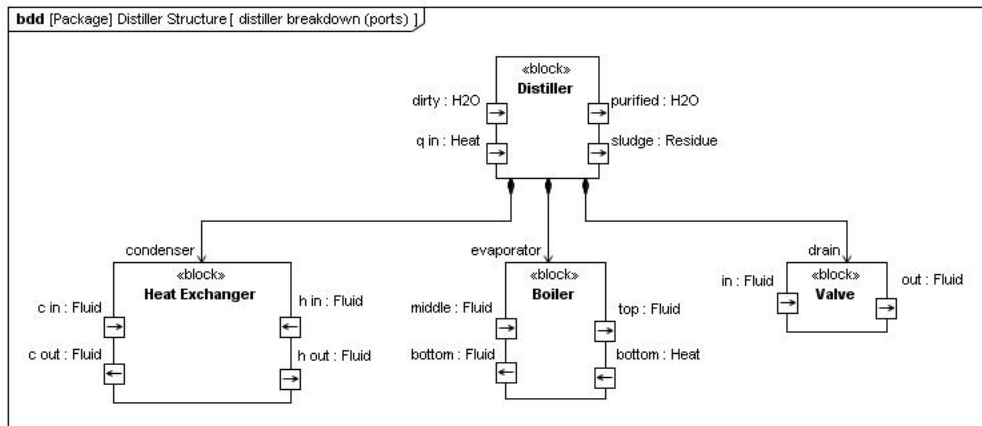
Figure 3.5: Initial Batch Water Distiller-Internal Block Diagram. Source: [13]. Reprinted with Permission.
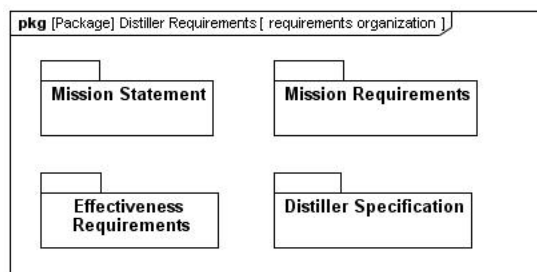


Figure 3.6: SysML Package Diagram Requirements Organization. Source: [13]. Reprinted with Permission.

requirements based on his or her preferences or organizational conventions. This separation can be similar to the categories shown or those deemed applicable to the system modeling purpose or perspective.

Figure 3.7 depicts a SysML requirement diagram using the folder contents from Figure 3.6. As shown, the DS designation here represents a design specification requirement, versus the MR designation for mission requirements; both are traceable back to a particular source document and the package diagram from Figure 3.6. When the model contents relate to an external model parameter, use the model data to provide an estimate for the external model parameter. Figure 3.6 and Figure 3.7 provide information concerning the water distiller requirements. The resulting total of requirements represents the count associated with that system for the parameter of interest. In this instance, that parameter is requirements.
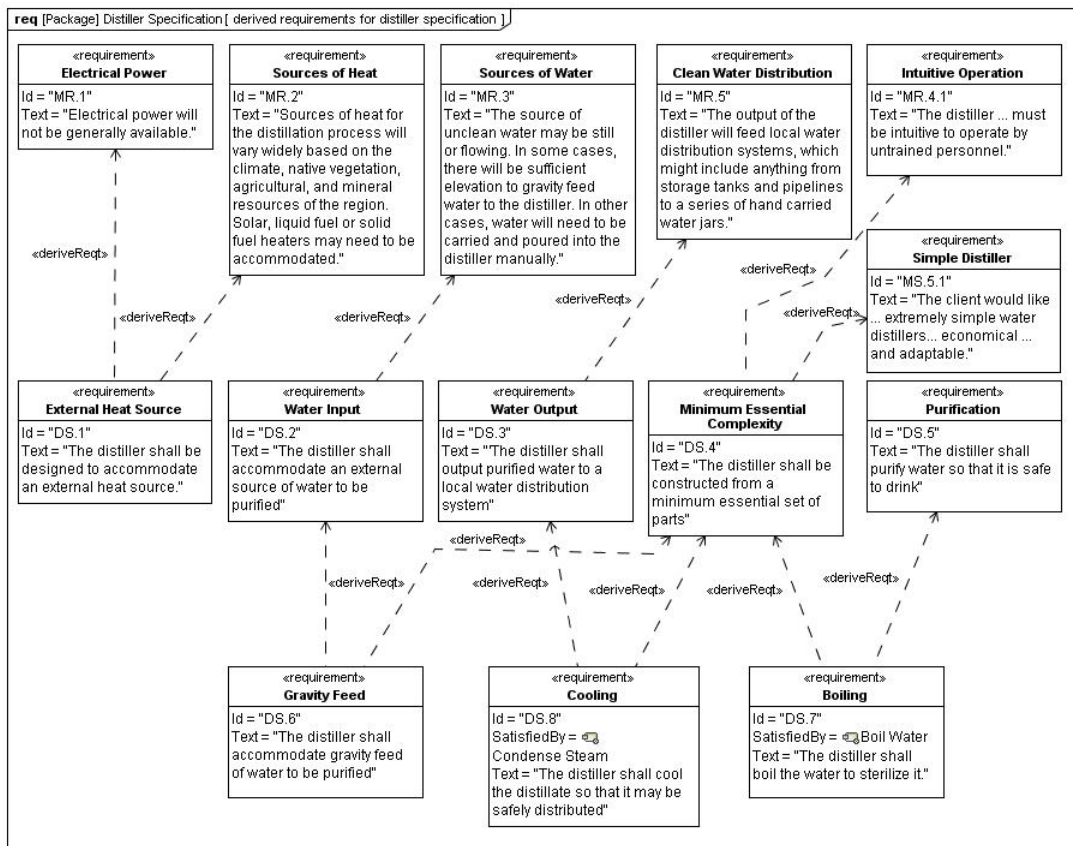
Figure 3.7: SysML Requirements Diagram-Derived Requirements and Relationships. Source: [13]. Reprinted with Permission.

For example, when the aggregation of the package diagram (Figure 3.6) and the requirements diagram (Figure 3.7) occurs for the system of interest, the result can help categorize, trace, and document information concerning the system requirements. Consider an organization that wanted to include the reliability, survivability, and or safety requirements of the water distiller system. Categorize these requirements and place them in an appropriate package diagram. Trace these requirements back to the representative document and model them to show the various dependencies. See the author's extension of this concept in Figure 3.8 using Innoslate's asset diagrams. The figure illustrates the model impacts when adding a set of artificial reliability requirements to the package and requirement diagrams. The top left diagram represents the package diagram, now updated to include *RR.0*, the reliability requirements, and the bottom right diagram shows the decomposition of the reliability requirements. Not shown but inferred are the system links and dependencies similar to

Figure 3.7. In the extension, four reliability requirements *RR.1-4* enhance the water distiller package diagram in Innoslate. If the previous total of system requirements was 31, then these four new reliability requirements included bringing the total system requirements to 35 for the system model. This example highlights how model changes propagate through the system model. Alternatively, Innoslate has a requirements label that allows the user to annotate a particular model entity as a system requirement. This labeling technique leverages model data, the package diagram example given shows leveraging the model architecture or structure. The structure approach is independent of modeling tool and language.
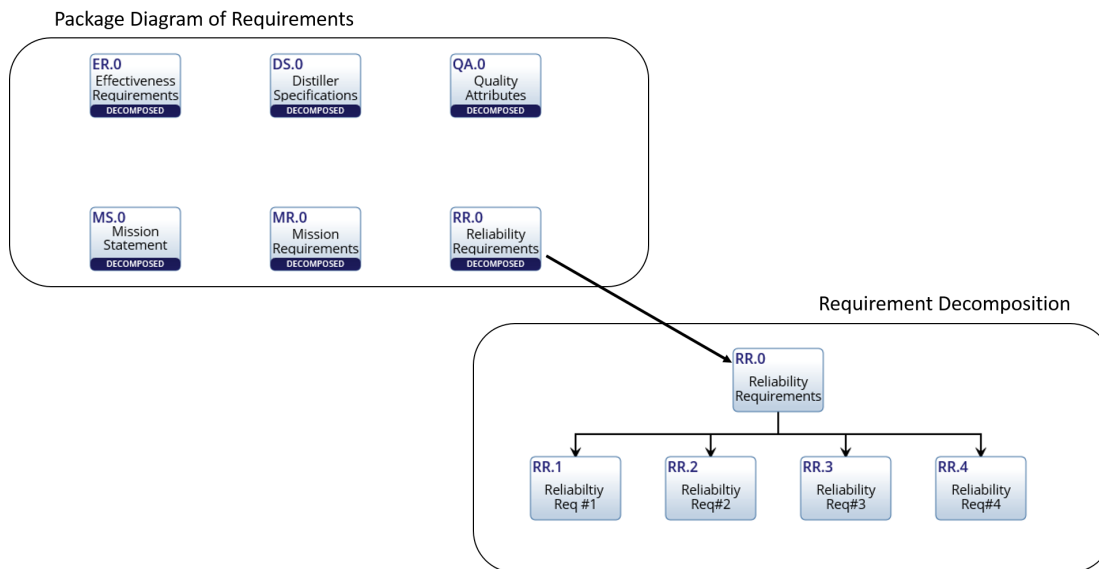


Figure 3.8: Extension of Requirements Diagram Using Innoslate.

Figure 3.8 in tabular form, provides a commonly used method for understanding the number of requirements. This tabular format provided a relatively quick method to count, filter, and sort through possible COSYSMO requirements when labeled and tailored to information related to COSYSMO. This tabular output could be in the form of an expressed labeling convention or modeling assessment parameter like easy, nominal, or difficult. This complexity assessment links to the various combinations and dependencies of the original requirements and the COSYSMO size drivers. See Figure 3.9, which is the exported system model information from Innoslate with a notionally assessed difficulty for two of the requirements discussed, distiller specifications and effectiveness requirements. This output could be the result of a search of all items labeled as a requirement or as discussed, the contents of a

package diagram that captured all the requirements.

| Class | Count | Model Entity Attributes | | Distiller Project Innoslate | COSYSMO |
| | | Name | Number | Description | |
|---|---|---|---|---|---|
| | 1 | Distiller Requirements | DR.0 | Overall Package Diagram-Partion | |
| | 2 | Distiller Specifications | DS.0 | Distiller Specifications Requirements of the Water Distiller-Package | |
| | 3 | External Heat Source | DS.1 | The distiller shall be designed to accommodate an external heat source. | E |
| | 4 | Water Input | DS.2 | The distiller shall accommodate an external source of water to be purified | E |
| | 5 | Water Output | DS.3 | The distiller shall output purified water to a local water distribution system | E |
| | 6 | Minimal Essential Complexity | DS.4 | The distiller shall be constructed from a minimum essential set of parts | H |
| | 7 | Purification | DS.5 | The distiller shall purify water so that it is safe to drink | E |
| Asset | 8 | Gravity Feed | DS.6 | The distiller shall accommodate gravity feed of water to be purified | E |
| | 9 | Boiling | DS.7 | The distiller shall boil the water to sterilize it. | E |
| | 10 | Cooling | DS.8 | The distiller shall cool the distillate so that it may be safely distributed | E |
| | 11 | Residue Removal | DS.9 | The distiller shall incorporate a mechanism to remove residue that results from the distillation process. | E |
| | 12 | Effectiveness Requirements | ER.0 | Effectiveness Requirements of the Water Distiller-Package | |
| | 13 | Cost Per Unit of Clean Water | ER.1 | Sustained cost per unit of clean water provided. This must consider labor, fuel, power, consumables and maintenance. | E |
| | 14 | Clean Water Quality | ER.2 | Quality of water provided (must be above a minimum accepted safety threshold) | E |
| | 15 | Cost Per Distiller Installed | ER.3 | Cost per distiller, including transportation. This drives the number of units that may be procured, and thus the number of people served. | E |
| | 16 | Usability | ER.4 | Usability by local, untrained operators. | N |

Figure 3.9: Tabular Form of Requirements Diagram Using Innoslate.

Requirements exist in many other places than source documents, some derived or inferred requirements exist. Although not as obvious, many of these requirements are substantial. The SysML internal block diagram (ibd) presented in Figure 3.10 provides one model example to illustrate how SysML diagrams can help identify derived requirements. The quality and consistency depend on the experience of the systems engineer and or individual modeling.

In this ibd, inferred or derived requirements make up several of the system model requirements. Specifically highlighted are the human factor considerations and other life-cycle considerations of the operator interactions with the heat source and the water source. Infer the same information from Figure 3.11, a SysML use case diagram. Again, note the representation of the same data in alternate presentations for a specific audience. This use of the SysML models to find requirements helps populate both the stated and derived requirements for the system.

This discussion highlights an additional consideration for COSYSMO integration. Explore all facets of the system model and database for requirements traceability and future use. Additionally, use cases are beneficial at identifying system operational scenarios and discussed
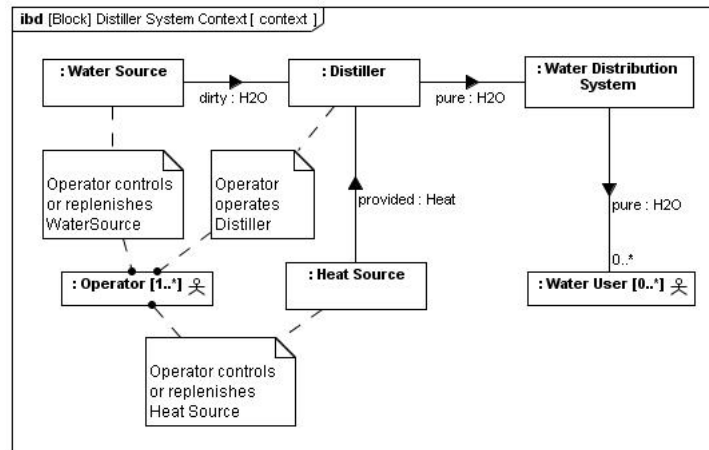
Figure 3.10: SysML Internal Block Diagram That Supports Deriving System Requirements. Source: [13]. Reprinted with Permission.
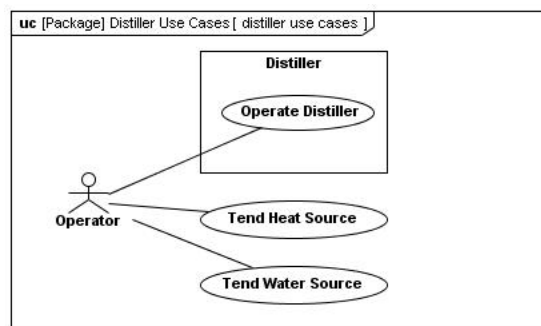


Figure 3.11: SysML Use Case Diagram That Supports Deriving System Requirements. Source: [13]. Reprinted with Permission.

in detail in a proceeding section.

Recall Figure 3.7 concerning derived requirements and their relationships for the simple water distiller. Unique to Innoslate, reports exported in CSV file types can present this structured data in an easily understood and quickly analyzed method. Many tools such as IBM Rationale Doors, ViTech Core, and Magic Draw provide this capability with variations in the particular output format. Model size may force clever package diagram selections to partition the model into manageable sections. Once all the system model requirements have been located, modeled, and packaged, then the complete listing of system requirements in the same tabular format as Figure 3.9 allow the number of requirements for the system, assessed by complexity, to be passed manually to a COSYSMO tool. This progress highlights that at

least a semi-automated integration of COSYSMO could include capturing of requirements from the model in a tabular form, labeled and sorted by assessed difficulty or complexity, and then placed into a COSYSMO tool.

## 3.5   Modeling Behavior (Function(s))

In systems architecture, there is a common phrase that "form follows function [63, p. 29]" and, as discussed in Chapter 2, functions satisfy needs and are synonymous with behavior. Our discussion to this point has highlighted the user needs and captured them as a series of requirements to address the potable water concern of a humanitarian relief agency. This next section will provide a similar walkthrough to highlight the use of SysML diagrams to model the water distiller functional aspects. SysML uses behavior diagrams, specifically the activity, sequence, state machine, and use cases to model the functional aspects of a system. These behavior diagrams locate system interfaces. See Figures 3.12 and 3.13.
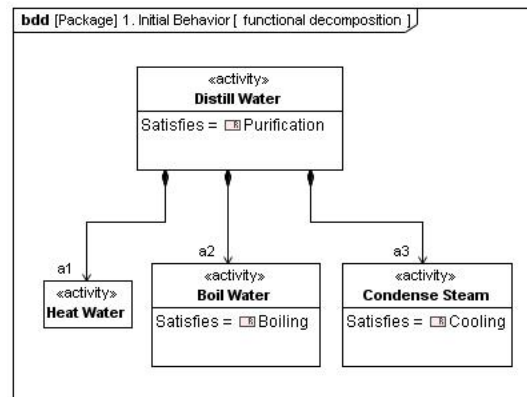


Figure 3.12:  SysML Block Definition Diagram for the Water Distiller. Source: [13]. Reprinted with Permission.

At the root of this analysis, Figure 3.12 highlights the block definition diagram for a batch distiller system. The diagram highlights both the functional aspects and the traceability of that requirement back to the users need. In Figure 3.12, the function *Distill Water* satisfies the *Purification* requirement. This model does not make any indication of where, the function will take place yet, but highlights the necessary and sufficient activities or functions to achieve water distillation as *(a1-a3) Heat Water*, *Boil Water*, and *Condense Steam* for the water distiller. This process is classic functional allocation.

65
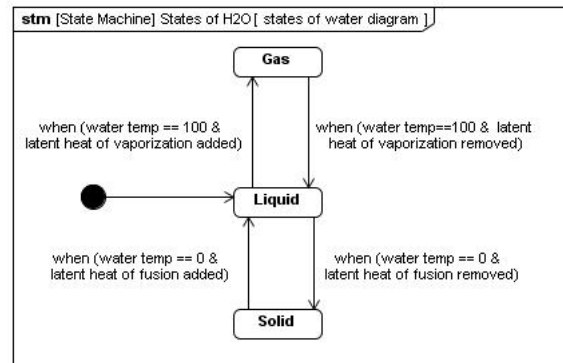
Figure 3.13: SysML State Diagram for the Water Distiller. Source: [13]. Reprinted with Permission.

Figure 3.13 is a state machine diagram for the water distiller example. It shows the phase changes of water and highlights scientific considerations for the *:H2O* in the system. Again, this provides additional details and considerations for the system and highlights other areas of system exploration in the MSA Phase analysis for the DOD. The state diagram combined with the block definition diagram (bdd) illustrate considerations for heat transfer considerations and also the differences between heating and boiling water from a physical property perspective. When combined, Figure 3.14 an activity diagram representing a batch distiller may result which shows the transition of the non-potable water to usable drinking water and highlights the required interfaces or ports depicted by the small boxes to achieve the desired functionality.

This water distiller system functionality is still at a behavioral level, but provides an initial assessment of another size driver parameter of COSYSMO, the number of major system interfaces. The small boxes in the activity diagram from Figure 3.14 show interactions between physical and logical objects. Similar to the discussion on system requirements, a tabular format of this interface information with an appropriate model annotation to capture the various interface complexity seems relevant to COSYSMO.

Refinement and exploration could continue for the distiller model and incorporate various design alternatives under consideration, but it is important to remember that each modification to the model happens through a database. As an example, Figure 3.15 provides the same behaviors or functions modeled for the continuous water distiller design for comparison. The same behavior analysis method presents a slightly different system model because the

66
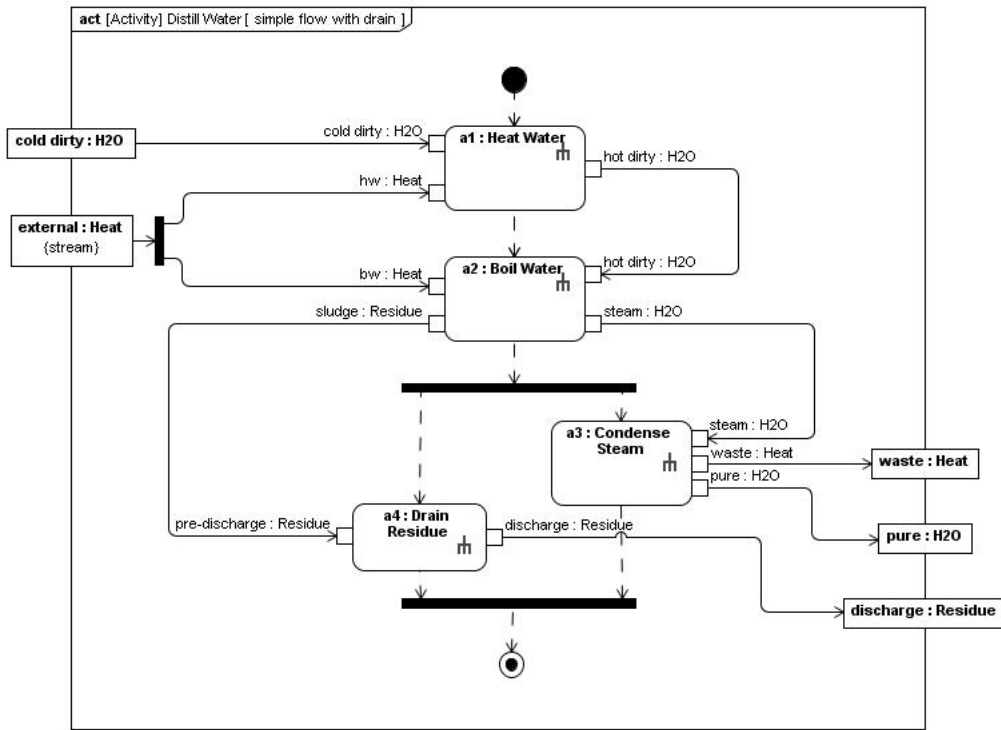
Figure 3.14: SysML Activity Diagram for the Water Distiller. Source: [13]. Reprinted with Permission.

two functional allocations for the models are different. Note that the steam from the *Boil Water* activity is now the external heat source illustrated by the additional port and flow in the model. A change in the system design alternative created a modification of the system interfaces. This slight variation has the potential for significant cost impacts as materials associated with the original design may not be as suitable for the model variation or significantly change weight or the pool of available vendors for tooling. These downstream effects are parallel, to the early trade space decisions during the MSA Phase of the DOD MDAP. Use of the system model allows for robust and speedy changes to a system design and provides insight into the cost impacts of the system design choices.

The functional allocation of the various activities *(a1-a4)* in Figure 3.15 requires specific physical components to achieve desired results. The next section addresses this topic. Figure 3.16 shows the proposed structural components and their intended purpose for the water distiller with the addition of three new components, a *user control panel*, a *user controller*, and a *tee fitting*. Simply focusing on interfaces associated with ports and flows,

67
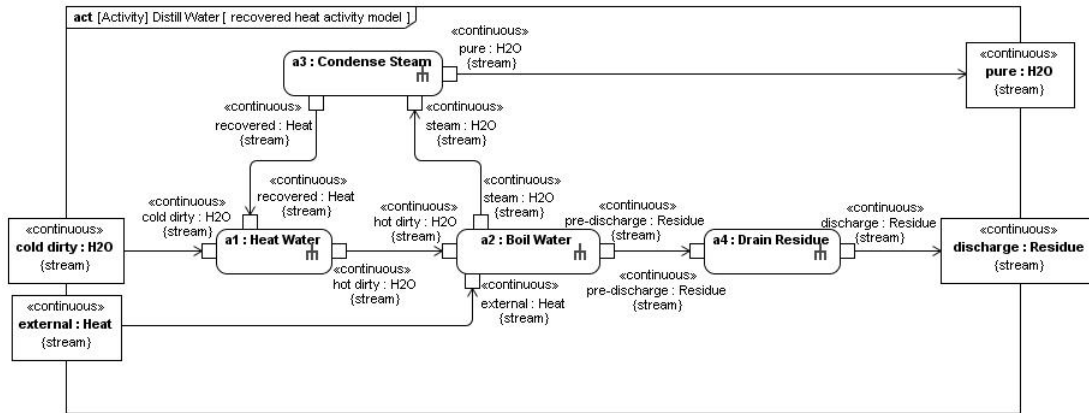
Figure 3.15: SysML Activity Diagram for the Continuous Water Distiller Alternative. Source: [13]. Reprinted with Permission.

the manual process discussed in the requirements section using tabular output can help find the number of interfaces of the system. Instead of the package diagram, a system functional model is required to locate the number of interfaces and pass that data to a COSYSMO tool. The type of diagram and parameter of interest would be the only difference from this estimate of interfaces and Figure 3.9.

One caution, the Figure 3.16 model exists at a relatively high level; more granularity in the model would illustrate more system interfaces. The granularity of the model to find these interfaces, however, must mirror the cost estimating level for the system engineering cost estimate in COSYSMO. Linking the cost estimate to the system model allows the estimate to adapt with the system model development over time. As the model improves, the cost estimate has the potential to improve also. Over time, the history of the cost estimates in a time series representation can provide information concerning trends of the model or areas with large variance. This type of information can help program managers and systems engineers identify and dedicate more resources or subject matter experts on the most critical aspects of their project or analysis efforts. Chapter 4 of this work estimates the number of interfaces and passes the information to COSYSMO for proof of concept.

## 3.6 Modeling Structure (Form)

System structure is synonymous to form. It represents the physical components and most detailed aspects of design. Figure 3.17 shows the allocation of the functional activities of the
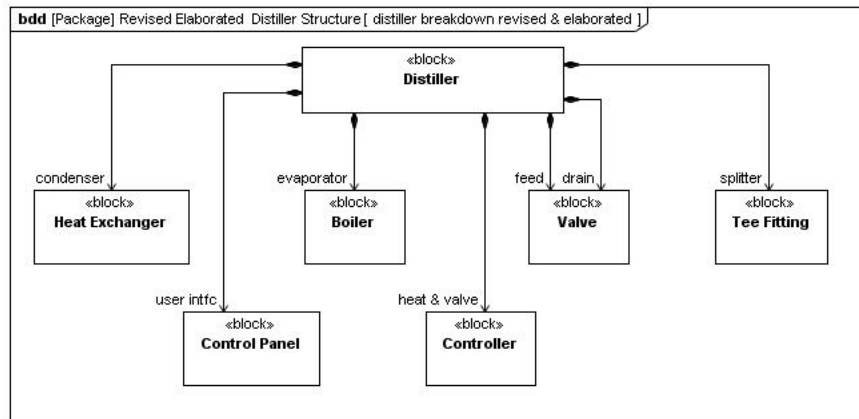
Figure 3.16: SysML Block Definition Diagram for the Water Distiller. Source: [13]. Reprinted with Permission.

water distiller's behavior to the proposed structural components to achieve these activities. Note the bold black lines denoting "swimlanes," [65, p. 191] which delineate the location of the various interfaces and ports and also provides general information about what the port or interface is providing regarding energy, matter, material wealth, or information [35].
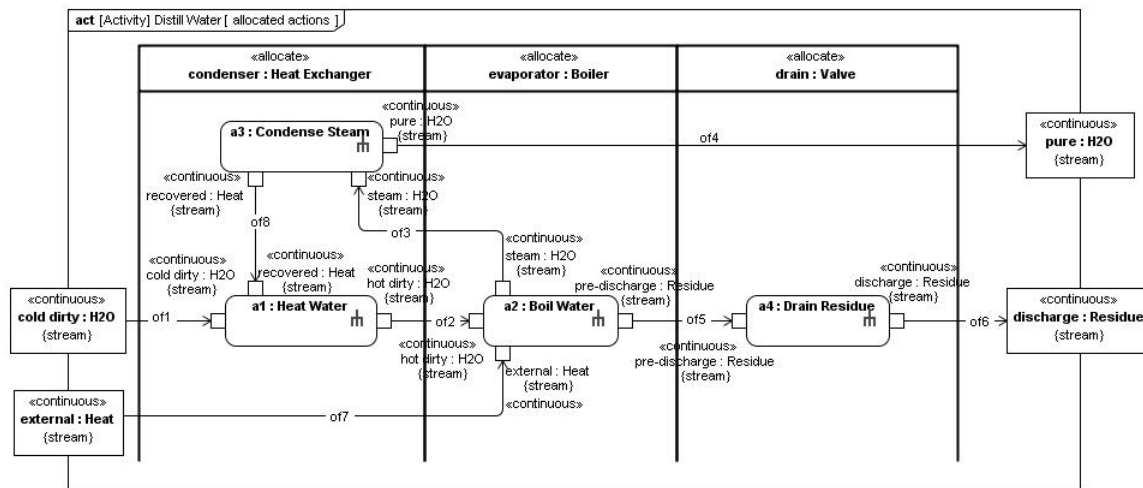


Figure 3.17: SysML Activity Diagram for the Water Distiller: Functional Allocation. Source: [13]. Reprinted with Permission.

When presented as an ibd like in Figure 3.18, the functional activity *a2:Boil Water* now occurs in the *Boiler* a physical component called the *evaporator*. This physical component has five ports, but six interfaces when considering directionality. See the two-way arrows.

The *Boiler* transfers fluid H20, heat information, and power signals to the other distiller components. This added level of detail and granularity provides more refinement of the number of system interfaces as a COSYSMO size driver. This interface understanding also serves as a method to analyze performance using the governing relationships and flow rates across the ports and interfaces. In the case of the water distiller, this analysis includes mathematical expressions for analyzing heat transfer, mass flow rate, and various electrical signals. The next section will illustrate these performance modeling capabilities using SysML diagrams and provide the link to our next COSYSMO size driver of interest, the number of algorithms.
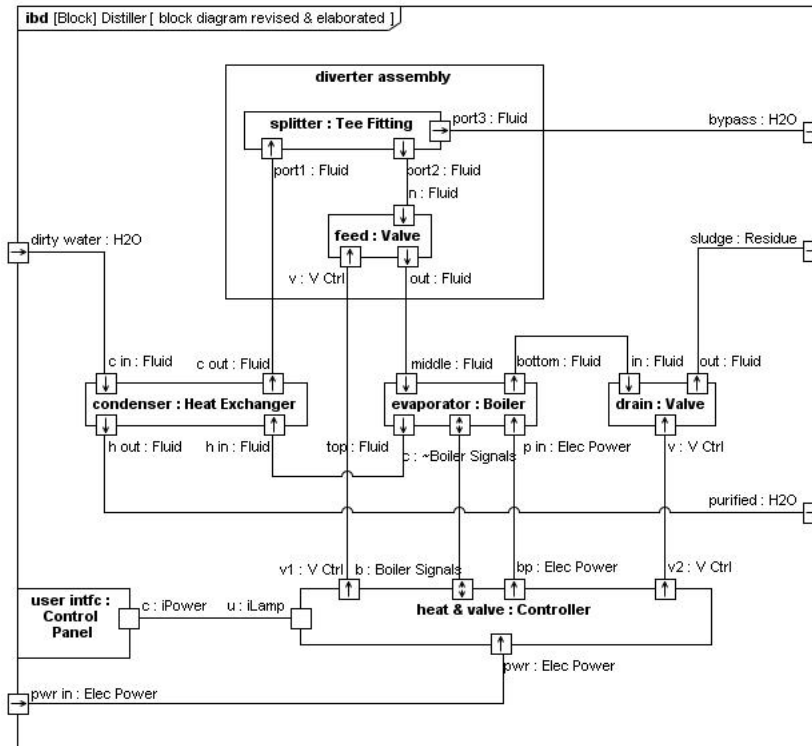


Figure 3.18: SysML Internal Block Diagram for the Water Distiller: Refined Design. Source: [13]. Reprinted with Permission.

## 3.7 Assessing System Performance

The one aspect of the MBSE discussion from Chapter 2 not discussed yet is system performance analysis. Our problem definition and requirements analysis highlighted key aspects

of the system engineering evaluation. The development and modeling of physical, functional (form), and behavior (functional) aspects of a water distiller provided the necessary synthesis to illustrate the analysis and development of feasible alternatives. Our scenario includes refinement of a batch or continuous water distiller, but this event is similar to the MSA Phase systems engineering tasks for the DOD. It highlights the trade decisions and analysis impacts between two competing system alternatives. This next section will focus on the system performance analysis as it relates to the water distiller.

### 3.7.1   In Search of Algorithms

Given the modeling effort to this point of the chapter, we have detailed the system requirements and interfaces to a component level, and two COSYSMO size drivers. The inclusion of two additional SysML diagrams associated with system performance analysis will complete the identification and extraction of COSYSMO size drivers applied to the water distiller example. The first diagram discussed is the parametric diagram. This SysML diagram highlights the mathematical relationships between various aspects of the distiller. Figure 3.19 provides an illustration of the batch distiller regarding the physical flows and their governing relationship.

For example, *s1*, the single phase heat transfer equation, is a constraint that governs the rate of heat flowing in the system expressed regarding mass flow rate, temperature difference, and specific heat. In an executable form, this model captures inconsistencies and faults. The identification of algorithms or mathematical expressions relates to the COSYSMO size parameter, the number of system algorithms. Additionally, the formulas presented as constraints in the parametric diagram are a means to assess the difficulty of these equations. COSYSMO presents basic algebra as easy, calculus and time constrained equations as nominal, and complex optimization and simulation as difficult [51]. Using this convention, we see three algorithms or equations, each requiring algebra. These equations represent three algorithms for the system, each assessed as easy.

### 3.7.2   Impact of Use Cases

The parametric diagram focuses on the internal workings and constraints and is a SysML structural diagram; the use case diagram assesses how the system interacts by external entities. As modeled by OMG and [12]–[14] the water distiller humanitarian relief scenario,
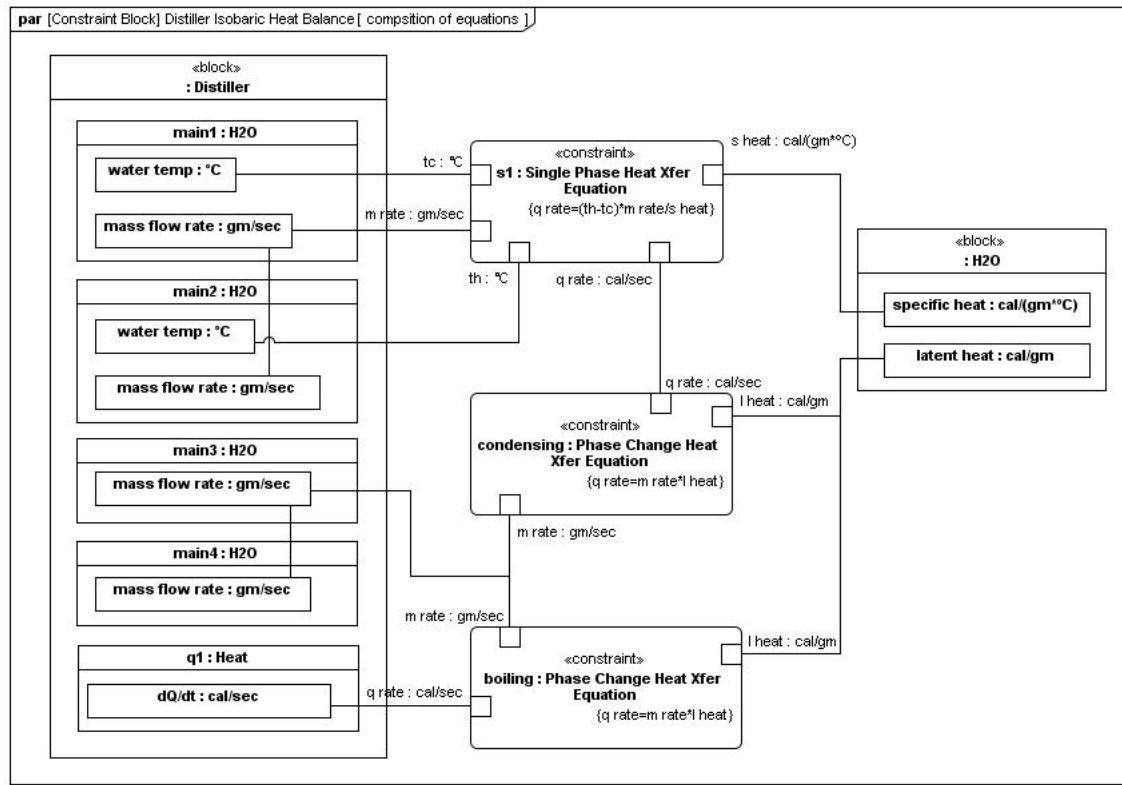
Figure 3.19: SysML Parametric Diagram for the Water Distiller. Source: [13]. Reprinted with Permission.

includes a single use case. The single use case is a product of the original problem statement and scope and illustrated in Figure 3.20. The figure highlights a single user or *Operator* running the distiller to provide the necessary drinking water for personal use. In a more complex system or system of systems, the use case diagrams may include additional operational scenarios of the system of interest and help provide information for the final COSYSMO size driver parameter, the number of operational scenarios.

An artificial expansion considering a manufacturer perspective allows for the inclusion of more than one operational scenario for the water distiller scenario. Figure 3.21 illustrates the insertion of these use cases as an example. The *Operate Distiller* activity now connects the *Manufacturer* and the *Operator*. The *Manufacturer* now has five activities related to the humanitarian relief scenario.
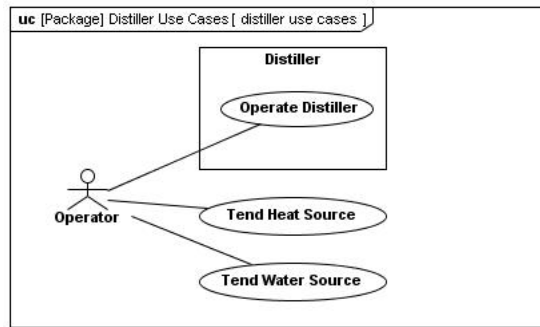
Figure 3.20: SysML Use Case Diagram for the Water Distiller. Source: [13]. Reprinted with Permission.
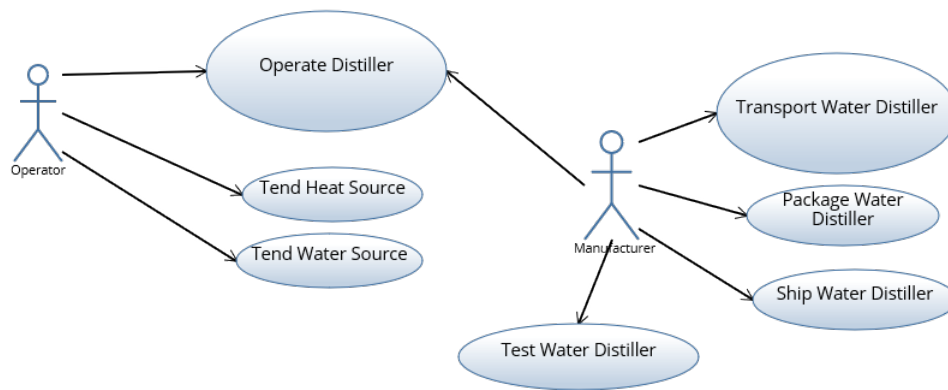


Figure 3.21: Innoslate Extension Use Case Diagram of Water Distiller Scenario: Manufacturing Concerns

## 3.8   Review of  COSYSMO Semi-Automated Integration

The previous six sections illustrate how manual labeling, grouping, and counting achieve reuse of the water distiller information for a cost estimate in COSYSMO. The process did not remove the need for significant systems engineering and modeling efforts up front, but provides a means to determine systems engineering cost while using near real-time and available data. Several COSYSMO tools exist, and the information from the distiller passed to any one of these tools could estimate the systems engineering cost. This manual method of counting, filtering, and grouping aspects from the system model database is rudimentary, so the next section will highlight an improved and more automated method to leverage the XML information located in the PES data layer of the system model. The aim is to

both have an automated means to find estimates of COSYSMO parameters from the system model XML output, but also automatically pass the extracted COSYSMO parameters to a COSYSMO tool. Specifically, the effort will look at leveraging the structure of the Innoslate XML output for the water distiller project to find the instances in the system model that correspond to the COSYSMO size drivers.

## 3.9    Improving the COSYSMO Integration

The original integration efforts focused on the identification and extraction from the system model database or repository. To improve this integration and utility, the author explored the XML data files that represent the model. The aim was to parse the XML in a way to bin COSYSMO cost drivers using the same MBSE techniques discussed with the water distiller. An algorithm-based approach enhances the manual method and tabular form output and passes that information from the XML output file to a COSYSMO tool. Similar automated passing has occurred [32] with HTML using COSYSMO, this work provides recommendations on what XML entities to pass for given system model. Figure 3.22 illustrates the process and highlights the XML link for finding the appropriate COSYSMO size drivers in the distiller example. As a clarifying point, all the XML files presented are from Innoslate and are in an LML based schema. Table 3.2 shows the instances noted that produce similar results to the manual method. The distiller project used Innoslate version 3.4 and XSD 3.0 during development. For reference C1, C8, C18, and CE represent action, asset, equation, and port respectively. P2, P4, P19, and PF represent duration, percentage complete, a text equation, and port directionality respectively. The two labels L5Z and L6W represent the use case and package diagram labels at the time of development. Note during this thesis an update to Innoslate occurred from version 3.4 to version 3.5. Some modifications of the schema items, labels, and relationships occurred. The package diagram label in Innoslate version 3.5 is L60 and the use case label is L6X. Ensure any algorithm used has the most current relationships identified, before parsing any system model XML output discussed.

Table 3.2 provided the necessary relationships to perform the data exploration of the system model XML output. The data analysis tool selected was R, version 3.2.3 [71]. Using the R package (XML) [72], the Innoslate XML file was read into R and parsed into a tree. The Innoslate XML output for the water distiller contained three XML nodes in the data

```
<?xml version="1.0" encoding="UTF-8"
standalone="true"?>
<innoslate xsd Version "3.0"......>
   <parent1
      <child1>
         <attribute>
            <relationship>
               ►Model Data----#####-----Model Data
            <relationship>
         </attribute>
      </child1>
   </parent2
</innoslate>
```
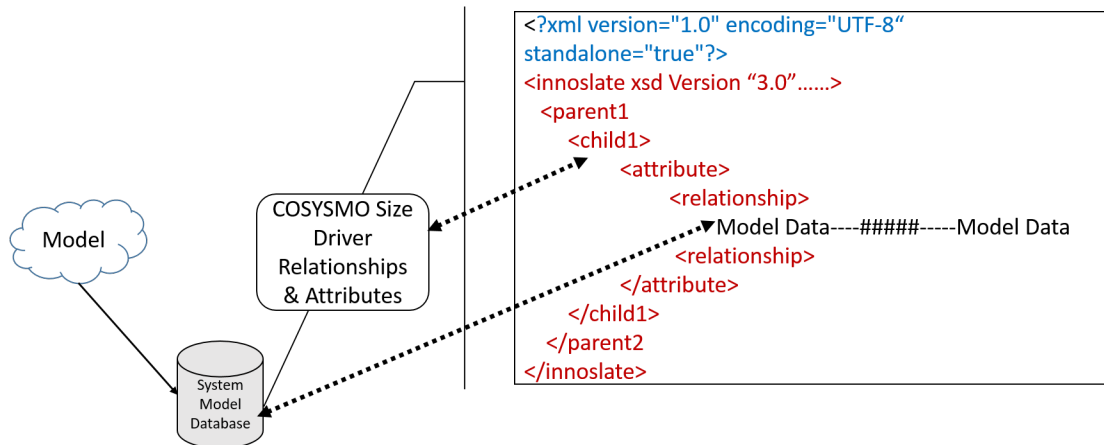
Figure 3.22: Proposed Improvements to Manual Method By Using System Model XML Data.

Table 3.2: XML Instances Used for Automated Method.

| COSYSMO Size Driver | XML Instances | Translation | Water Distiller Figure |
|---|---|---|---|
| Number of Requirements | "C8"/"L6W" | Every instance of an asset labeled as a package diagram | Figure 3.6 |
| Number of Interfaces | "CE"/"PF" | Every instance of a port with a specified direction | Figure 3.18 |
| Number of Critical Algorithms | "C18"/"P19" | Every instance of an equation with a math expression | Figure 3.19 |
| Number of Operational Scenarios | "C1"/"L5Z" | Every instance of an action labeled as a use case diagram | Figure 3.20 |

file: schema, labels, and database. The XML package in R written by Duncan Lang, allows for the use of the XPath interpreter, which allows R to leverage traditional query and relationship aspects of XML files [72]. These relationships include parents, children, siblings, and various size and attribute searches. A series of R scripts replicate the manual method presented in this chapter, using Lang's package. In general, the scripts take the relationships of Table 3.2 and ask R, to provide a count of those aspects. Appendix: Select XML Output has the entire Innoslate XML output for Figure 3.19, the parametric diagram

and all the R scripts used for future application and extensions. The scripts show a simple use of the R data analysis tool and XML package but highlight proof of concept. This lead to the development of the web-based tool presented in Chapter 4.

## 3.10   Lesson Learned from COSYSMO Integration Efforts

The following discusses lessons learned from using the two methods presented.

- Clearly communicate model purpose and scope across the entire modeling team. Conventions and modeling standards must clearly occur across the whole project. This clarity must include model extensions. Inconsistencies result in difficulties in extracting all relevant information and may eliminate potential links in the data. The system model granularity must match the granularity of cost estimate.

- Configuration and change management must remain with a small and select group of people and must include data, models, and data access.

- The MBSE method proposed can aggregate the subjective assessments of the COSYSMO cost drivers. The accuracy of this aggregation is unknown in its current form; parameterization is possible.

- Translation among modeling languages requires significant understanding of the schema and the specifics of the tool applied. LML is different from DM2, UML, and SysML, but similarities exist when presented using a specific ontology or modeling convention.

- DODAF is not a language; it is a framework. DM2 groups represent several potential paths and so instead of a single term, a pattern of terms will best identify a DM2 entity. Some, but not all DODAF models are achievable with SysML.

- A time series aggregation of system model information over time can help recognize problem areas for the project team, supported by data.

- The relationships to pull relevant COSYSMO parameters of a specific system model requires customization for a model unless a convention is developed and accepted before generating the system model. The same concepts of parsing the XML output would apply, but the modeling tool and its capabilities and limitations require understanding. For instance, the package diagram in Figure 3.6 estimating requirements used the underlying structure of the model. In Innoslate, that same output is achievable with using the label *C25* (Requirement). The modeler and system engineer must

decide on which provides the best utility for the intended purpose of the model.

## 3.11   Chapter Summary: Methodology

This chapter identifies and discusses the specific SysML models, data requirements, and model entities that relate to the four COSYSMO size drivers. Illustrations included modifications and extensions for the water distiller humanitarian relief scenario specific to COSYSMO and utilization of the modeling tool Innoslate. Presentations of both a manual and automated method reveal how to extract the discussed COSYSMO parameters to estimate systems engineering costs. The proof of concept provides a baseline for future refinement and enhancement. Seven points discuss lessons learned from the integration effort and highlight benefits. The next chapter will present the data results for the water distiller example as modeled.

THIS PAGE INTENTIONALLY LEFT BLANK

# CHAPTER 4:
## Data: Integration Results

## 4.1 Integration Overview: Estimating the COSYSMO Size Drivers

This chapter estimates COSYSMO parameters for the four size drivers of the water distiller humanitarian relief scenario. Output includes the results from both the manual and automated methods discussed and integration with a web-based tool. Only the four size drivers are considered and presented due to the subjective nature and organizational focus of the remaining 14 cost drivers and additionally, the lack of information available for the system of interest. When considering the MSA Phase context associated with DOD acquisitions, the information concerning these cost drivers is purely subjective, and a COSYSMO cost estimate is achievable by various Monte Carlo and simulation techniques or program manager input. The results of such simulation techniques provide a system engineering cost estimate with an associated confidence level and useful profiles for staffing and risk understanding. Additionally, similar cost estimation tools used by Galorath for extension and integration of the company's SEER cost model, have shown success by including project and organizational factors similar to COSYSMO cost drivers such as alternative staffing, funding, and schedule considerations in this manner [73].

Many of the COSYSMO cost drivers focus on the assessment of the organization and project team assigned to the project, to help generate worst, most likely, and best case cost estimates to better inform decision makers. For example, the individual abilities of the team are assessed in the personnel/team capability COSYSMO parameter. A decision maker could see which available team members, create an adequate knowledge base to support the project effort to an assessed level of need or conversely, for the limited resources present, decide what personnel/team capability level occurs and how that impacts the overall systems engineering cost. Valerdi also suggests that the size parameters are more impactful due to the exponential scaling [51]. Incorporating all the cost factors would require a system of systems approach [55] and is beyond the scope of this work but is discussed in the future work section of Chapter 5. The following section captures the results of estimating the four

COSYSMO size parameters as discussed in Chapter 3 and presents the integration results in a web-based tool developed from this research effort.

## 4.2   Number of Requirements (31)

### 4.2.1   Manual Method

The manual method presented took the underlying data from the water distiller using the SysML package diagram information, with the inclusion of additional requirements found in the HTML database for the water distiller from OMG [15]. Requirements exported in a tabular form provided a list of requirements to count manually. There was a total of 37 entities listed as requirements from the diagram. Upon further scrutiny six of those entities were due to the diagram convention and class definition associated with the model. Those six entities corresponded to the .0 or initial levels pictured in Figure 4.1, which note the means of packaging the diagram in an understandable way. DR.0 corresponds to the distiller requirements overall, and each subsequent .0 level represents the distiller specification DS, mission statement requirements MS, and effectiveness requirements ER from the original SysML requirement diagram in Figure 3.6. As modeled, there were 31 system requirements found in the water distiller system model. Notionally, the author assigned values to the distiller model to provide an assessment of the easy, nominal, and difficult for each of the size drivers to enhance the discussion and show the mix of modeling and system integration.

### 4.2.2   Automated Method

To incorporate the automated method for requirements development, the author exported the distiller requirement diagram using Innoslate XML exporter and searched those entities related to the requirement diagrams. See Appendix: Select XML Output A.2.2, for the R script and output performed on the SysML package diagram (Figure 3.6) of the water distiller to do this task. The entity names are specific to the model exported and Innoslate version used. Again, 37 children entities were noted when isolated to the specific diagram of interest from the entire XML output file, but there was no way to distinguish between the .0 naming convention and a specific requirement in the diagram information in this initial model. In the automated method used, it was possible to see if a .0 requirement had information, but not if it was truly a requirement or just a structural element to group
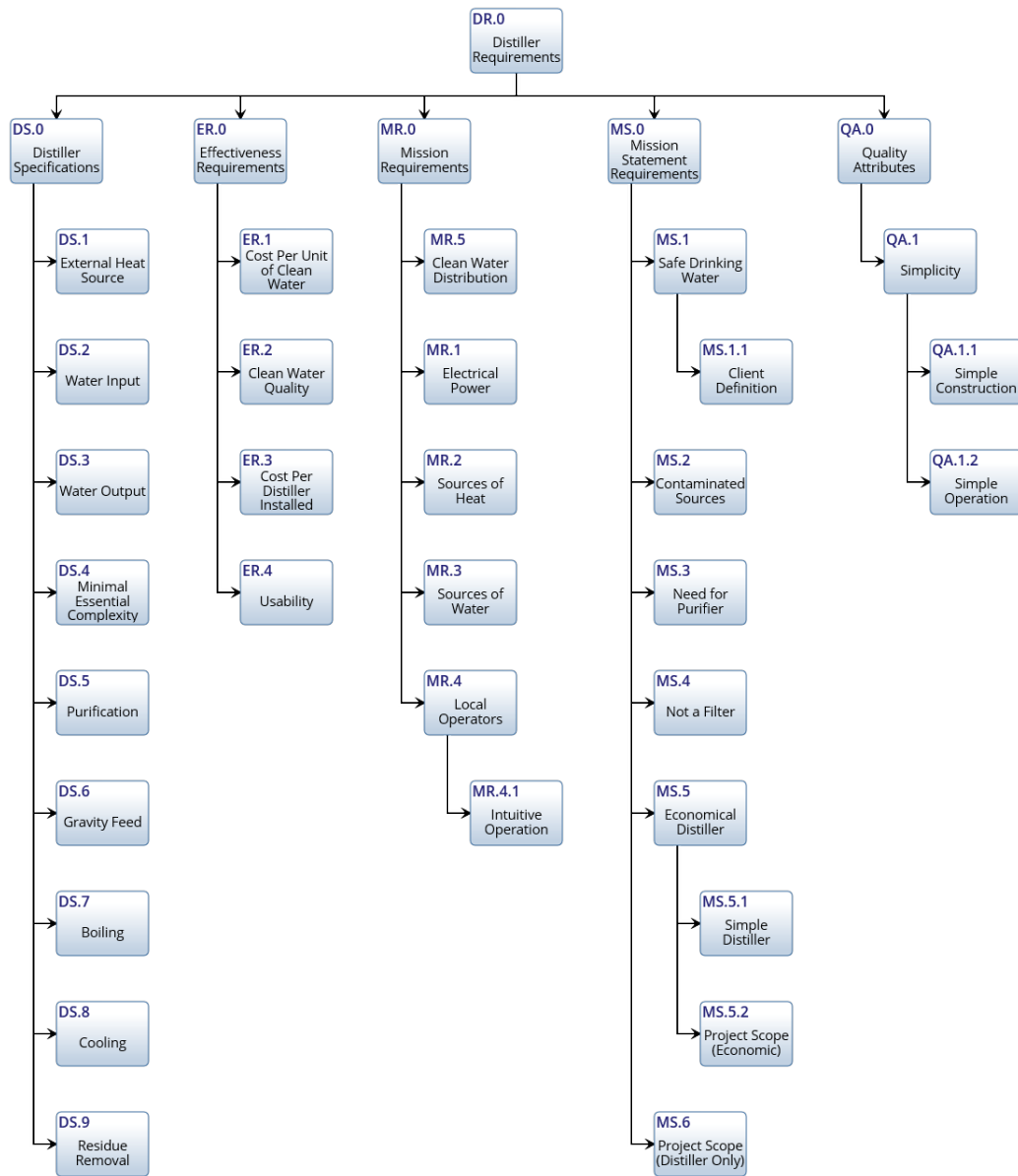
Figure 4.1: Innoslate Hierarchy Diagram of Figure 3.6 Original SysML Package Diagram of Distiller System Requirements.

requirements. The XML query resulted in 37 total requirements, with six model entities not containing any requirement information. This automated result is an overestimate, unless achieving the group of requirements is itself a requirement. Refinement must occur in the

R scripts to delineate between parent and children nodes of the package diagram either through labeling or an express modeling convention for the project. Using the Innoslate *C25* (Requirement) label in the XML query produces the expected 31 requirements but still requires adherence to a modeling convention or standard. This is an alternative method.

## 4.3 Number of Interfaces (32)

### 4.3.1 Manual Method

To estimate the number of major interfaces, the SysML ibd in Figure 3.18 for the water distiller was explored, with a focus on finding the specific directional ports in the model. These included both logical and physical constructs of the model. Again exporting the given model information into a tabular form, a total of 30 ports (32) interfaces were noted for the water distiller system model. Overall, 26 one-way ports, two two-way ports, and two ports that did not specify directionality exist in the model. The two-way ports were each counted as two interfaces, and the blank ports were assumed unknown from the present system model. This unclear aspect of the model could mean either 32-34 interfaces depending on the assumed outcome of those specific ports. A total of 32 interfaces were passed to COSYSMO, as it represented details presented in the model.

### 4.3.2 Automated Method

The automated method of integration searched for those instances of *CE* (Port) that included information *PF* (Direction: In, Out, Both or None). The result returned over 100 interfaces. This number represented every instance that an interface occurred in the water distiller project. Limiting the query to an ibd which represented the entire system, i.e., Figure 3.18, the query resulted in the same (32) interfaces found from the manual method. The query of the entire project database included those aspects of the model, which meet the instances specified, but previous editions and fit for purpose models were in the database. The ibd was one of the few models that were revised multiple times during the MBSE approach application to the water distiller. The query counts all instances which used the same construct as the ibd, and therefore had to be limited to the representation of the current system model. The author chose to use the last ibd presented in the methodology chapter for clarity and provide a graphic to reference. This example illustrates that a group of

models could be aggregated using the same instances discussed, but the need for some initial understanding of the model structure remains. A modeling convention or standard practice similar to a software specification can support this need. The automated output did provide information that two of the ports in the model did not have a directionality value associated with it. See Appendix: Select XML Output A.2.3, for R script and output performed on the SysML internal block diagram (Figure 3.18) of the water distiller model to do this task.

## 4.4   Number of Algorithms (3)

### 4.4.1   Manual Method

Three critical algorithms exist in the water distiller system model using the parametric diagram. They included *s1: Single Phase Heat Xfer Equation*, *condensing: Phase Change Heat Xfer Equation*, and *boiling: Phase Change Heat Xfer Equation* from Figure 3.19 the parametric diagram for the water distiller system. No other data was available to expand this aspect of the water distiller system model, but if Figure 3.19 represented the entire system model, then the same method applies. Due to the use of this single model, the entire Innoslate XML output for Figure 3.19 is in the Appendix: Select XML Output A.1 to allow the reader to explore how the Innoslate output files appear before applying either the manual or the automated method.

Looking at the Innoslate XML output in Appendix: Select XML Output A.1 the file has three nodes: schema, labels, and relationships. There are five total instances of *C18* in the entire file, one in the schema at line 4. There are no instances of *C18* in the label section of the file. Four instances exist of *C18* in the database sections at lines 119, 139, 173, and 193. Both line 4 and line 139 do not have a corresponding *P19* while the remaining three instances do. The values or in this specific case, equations associated with the instances of *P19* are found three lines later at 122, 176, and 196 respectively. The equations for line 196 and 122 are the same *(q rate=m rate* l heat)*, but represent the different aspects of *boiling:Phase Change Heat Xfer Equation* and *condensing:Phase Change Heat Xfer Equation* with different limitations. Line 176, *(g rate=(th-tc) *m rate/s heat)*, is *s1: Single Phase Heat Xfer Equation*, the third equation.

### 4.4.2   Automated Method

XML instances of *C18* (Equation) that included *P19* (Equation Descriptions) were queried from the SysML parametric diagram using the R script and returned the same three system model algorithms. The automated effort clearly detailed that two of the algorithms used were the same formula and highlighted the single blank instance in the schema file. See Appendix: Select XML Output A.2.1. Neither output, provided detail to inform decision makers that the algorithm occurred to different components or captured the unique equation limitations.

## 4.5   Number of Operational Scenarios (1)

### 4.5.1   Manual Method

Finally, the single operational scenario proposed in the water distiller system model of a single user operating the distiller was noted and passed to COSYSMO. The author passed the single scenario over the manufacturer extension from Chapter 3 to remain aligned to the original methodology assumptions expressed.

### 4.5.2   Automated Method

For the automated approach additional use case extensions were placed into the water distiller system model, to represent other likely operational scenarios, not in the OMG data repository. The purpose was to assess the ability of the proposed XML query to highlight the additional use cases in a format similar to the other scripts. During this query, R found all instances of *C1* (Action) labeled *L5Z* (Use Case) and recorded them. When recording only the original single use case, only one instance occurred. When adding the additional four use cases for the manufacturer user, the query resulted in the appropriate number of instances. The author passed the single use case to COSYSMO for the cost estimate, as the current representation of the system model.

## 4.6   COSYSMO Estimate for the Water Distiller

Table 4.1 provides a summary of the estimated COSYSMO size drivers and the associated value accessed from the water distiller system model. Figure 4.2 illustrates the COSYSMO

Table 4.1: Water Distiller Results Using COSYSMO Integration Methods

| Driver Name | Result | Easy | Nominal | Difficult |
|---|---|---|---|---|
| Number of requirements | 31 | 28 | 2 | 1 |
| Number of major interfaces | 32 | 29 | 2 | 1 |
| Number of critical algorithms | 3 | 3 | 0 | 0 |
| Number of operational scenarios | 1 | 1 | 0 | 0 |

cost estimate for those parameters in the nominal case using a notional $10,000 labor rate and the developed tool. Recalling Equation 2.1 provides us with the following estimate of the systems engineering costs using the system model. A and E for the presented tool are 38.55 and 1.06 respectively, the equivalent Size value is equal to 78, a working month includes 152 hours per Person-Month, and for the nominal case, the effort adjustment factor (EAF) is 1.00. The COSYSMO estimate for system engineering effort using the *SysML COSYSMO* tool [74] demonstrating this integration is 25.6 Person-Months. This developed tool uses the same relationships and model instances discussed, but utilizes web-based tools to parse and pass the system model information. See also the suggested breakdown of the system engineering effort across the lifecycle and the additional risk information in Figure 4.3.

$$PM = A * Size^E (EAF)$$
$$PM = 38.55 * 78^{1.06}(1.0)$$
$$PM = \frac{3884.0}{152}$$
$$PM = 25.6 \quad Person - Months$$

## 4.7   Chapter Summary: Integration Results

A presentation of both the manual and automated methods for finding relevant COSYSMO size drivers occurs using the discussed MBSE methodology and the XML output for a system model. The web-based COSYSMO tool [56] was successfully integrated per this approach

Figure 4.2: COSYSMO Cost Estimate Using *SysML COSYSMO* Tool [74].



Figure 4.3: COSYSMO Acquisition Effort Distribution and Monte Carlo Results [56].

by automating the system size input. The web-based application was extended to read in Innoslate files, parse the XML structure to extract the system size entities and automatically populate the size inputs. This tool called *SysML COSYSMO* [74] is demonstrated in Figure 4.2. It will be further updated, maintained, and applied to the joint AFIT study [75] for the exploration of model-centric UAV ISR analysis in future work.

Two instances of over estimates occurred using the algorithm based methods, due to lack of distinguishing characteristics between parent entities and their associated children and not specifying a particular model diagram to confine the XML queries. These errors, provided the necessary refinement to develop the web-based application of the COSYSMO discussed using the MBSE approach. Overall the SysML COSYSMO tool performed an estimate using 31 requirements, 32 interfaces, three algorithms, and one operational scenario. The SysML COSYSMO [74] tool estimated a systems engineering effort of 25.6 Person-Months given the water distiller system model and system defaults. This value roughly equates to a 4.5-month project estimated at just over $255,500. Additional, information relating to a recommendation of staffing and scheduling for that system engineering effort and enhanced risk understanding occurs. The results present proof of concept that data from a system model provides adequate details and relationships to support an estimate of systems engineering cost using XML. The PES foundation for the data layers associated with DODAF and required for major defense acquisitions programs utilize XML. The recommendations and concluding thoughts of these results follow in the next chapter.

THIS PAGE INTENTIONALLY LEFT BLANK

# CHAPTER 5:
# Recommendations and Conclusions

## 5.1 Discussion

Useful integration of multiple matters is often a challenge. Items working separately and independently typically experience a performance loss and or performance trade to integrate and work together [35]. A common current example is military UAVs, which on one extreme can be larger than some small planes to accommodate the sensor, and weapons payload required, but on the opposite end of that continuum can be carried by a single person. Both perform functions to support the warfighter needs, one is carried alongside the warfighter and launched as desired for immediate tactical level decisions, the other larger UAV flies overhead and is launched and prioritized by staffs and commanders for more far-reaching military objectives. Visible trades occur to the refined user need, but no one UAV meets all the user's need. The current DOD implementation of DODAF is still searching for the most effective utilization of the framework to support its entire enterprise from the business, technical, and programmatic perspectives. Some assert that a single inclusive method is unattainable [19], while others suggest alternative ontological modifications to simplify [28], [66] would move in a better direction. All support data-driven decisions.

Speed, efficiency, and accuracy of information in defense acquisitions and cost estimation could benefit from the integration of an external cost model that only requires an understanding of the number and assessed complexity or difficulty. Data exists in the organization, but bringing it all together for early trade decisions is one challenge for the DOD. COSYSMO is one simple option from a family of constructive models, which presents a promising start or launch point for cost estimation professionals input. To achieve this integration requires either inclusion of additional modeling languages and ontologies from the DOD or series of modeling conventions for the commercial industry to follow across the enterprise. Refinement of the various modeling conventions and translations present new challenges for the DOD. These challenges are due to the evolution and advancement trends of technology and software. This work uses specific tools, but the concepts of using the ontological basis of a specific language to translate among various modeling languages to leverage structured

data for storage, analysis, and reuse in the support of faster cost and risk understanding are takeaways.

While the DOD architectural framework brings necessary standardization of the underlying data and structure for the DOD, the rigidity and an overwhelming number of terms of the DM2, and relationships makes practical use of the over 400 terms unwieldy. The use of available modeling profiles and plug-ins produce a product that is mathematically rigorous and complete but lacks the clarity and utility of simple communication among diverse teams and executable aspect for systems analysis. When utilizing other available languages such, as SysML, LML, and BPMN, increased interoperability in communication and performance analysis occurs for a wider group of stakeholders, but at the loss of the completeness of the ontology. In its current state, DODAF compliance does not address how well the models created represent the true state of the system or how much reuse of the data occurs. Communication directly from models will take time to evolve as MBSE continues to improve in the discipline.

It is important to remember that DODAF in its current version is just over five years old and has had success in projects that operate in multinational and large-scale projects, most notably the Joint Polar Satellite System (JPSS) [30]. It appears then that a delicate balance must be struck for the DOD between the need for detailed analysis synonymous with granularity, against the scope of DODAF synonymous with breadth. Given the results of the simple water distiller discussed in this thesis, the inclusion of a few additional modeling languages at the logical data module could result in additional granularity and analysis without any change to the physical exchange schema XML format. This alternative supplements system design and external model integration for better cost and risk understanding. Another alternative for DODAF is to continue with translation efforts for the XMI-PES translator highlighted in 2009 [27]. In favor of a multinational and defense focus, the IDEAS ontology chosen by the DOD is rigorous and complete but provides challenges for some aspects of utilization by domain diverse and geographically diverse teams. The DM2 is also not currently well known or used in organizations outside of the DOD, which creates challenges for external model integration and commercial practices. Both must determine how to address physical and cyber security to ensure model integrity.

## 5.2 Conclusions

The background, methodology, and results presented in this thesis highlighted themes of compliance, use of system models, data utility, and an integration focus. At the onset of the work, four research questions were developed and proposed. The work presented suggests the following responses given with the original research questions.

**Research Question 1:** How does the integration of COSYSMO fit into the current DOD acquisition process?

Early trade space decisions have the greatest potential for long-term impacts to system life-cycle cost. During the acquisition of major defense systems, the event-driven process requires adherence to federal mandates and organizational policies. In its current form, many of the Milestone decision inputs are data centric. The data includes system analysis and source documents, and the Milestones have submission deadlines and temporal considerations associated with them. The three main objectives of the early trade decisions focus on the balance of cost, schedule, and performance for a given system. COSYSMO is one example of an external parametric model that utilizes current best practices for DOD cost estimation but only estimates systems engineering costs. It also provides useful information associated with effort scheduling and risk understanding. Similar parametric estimation techniques exist with the family of constructive models in the software and hardware domains. A group or ensemble of DOD recommended models appears useful. The ensemble would capture elements associated with hardware, software, systems, and other reoccurring costs. Linked to system models this ensemble can develop a powerful understanding of early system design trades.

The current cost estimation process for major defense systems requires significant amounts of data and system understanding, often down to a component level, which is rarely available in early trade decisions. A mix of rough order of magnitude, analogy, and parametric cost estimates suffice until system understanding progresses for more detailed estimates. Cost estimators are often required to locate, normalize, and estimate from various data sources and repositories. Systems engineering cost are often termed "below the line costs," [48] but exist in every major defense system acquisition. COSYSMO's parametric technique provides a means to estimate systems engineering costs for early trade space decisions and analysis when utilizing a system model. This work highlights one instance of the concept as evidence

and highlights useful information for early decisions system acquisitions. COSYSMO or similar constructive models could serve as an initial starting point for early cost estimates between the various analysis, system design, and development organizations.

**Research Question 2:** How does COSYSMO map to DODAF?

DODAF is an architectural framework and not a language or methodology. COSYSMO is a constructive cost model that utilizes parametric cost estimation techniques and is, therefore, a means to evaluate a project's system engineering cost, and is a method to express aspects of multiple viewpoints in DODAF. There was no apparent direct one to one mapping for COSYSMO input parameters to single DODAF terms, instead for a particular grouping of DODAF models, COSYSMO links may exist in those models. The instances of links must relate to the model purpose, modeling preference, and decision makers' perspectives. DODAF focuses on data structure at the expense of a preferred modeling convention. Additional model inclusion will increase DOD system model granularity for deeper analysis.

The DOD decision to focus on data interoperability, storage, and structure, has created some challenges for executable and system performance when utilizing the DM2. Modeling languages, underlying ontology, and overall purpose significantly affect the ability to formulate unique instances of COSYSMO in DODAF compliant models, because the models typically trend back to domain norms in the absence of modeling conventions. This work initially explored a DODAF modeling effort from AFIT, which focused on a multi-tiered UAV. This model or a DODAF compliant case study will serve as validation, refinement, or enhancement to the results of this work. Given the success of the Universal Profile for DODAF and MODAF (UPDM) model in recent works, finding a DODAF compliant model that uses this profile, may aid in finding more generalizable links between DODAF and COSYSMO, and challenge the results of this work.

**Research Question 3:** How does COSYSMO map to SysML?

SysML is one language that can express the size and cost drivers of COSYSMO as an independent model or can model a specific system of interest. The modeling language can extract system model entities that represent the COSYSMO size drivers. Critical to the external model integration is an accurate, accessible system model, clear adherence to

and understanding of the model schema, and previous industry calibration for the system domain. For traditional systems functional decomposition and allocation, clear links exist between the package, requirements, parametric, use case, internal block, block definition and activity diagrams for finding relevant COSYSMO parameters. This work's results provide proof of concept. The successes of a similar process for object oriented and use case or scenario based system modeling would lend credibility to the proposed method.

This effort utilized a well-known and documented model in MBSE methodology. This model choice alleviated uncertainty in model accuracy and creditability, but tests for scalability and oversimplification must occur due to the models relatively small size and the underlying methodology assumptions. Additionally, lessons learned highlighted that a model well suited for external model integration, typically experience a loss in completeness or rigor. The water distiller example utilized an LML translation of the DM2 ontology which may have limitations in completeness, whereas if a complete DM2 model was utilized, more completeness or rigor are achievable at the expense of model communication. The next steps are performing a similar methodology to an accepted DODAF model to explore additional model relationships between COSYSMO and SysML.

**Research Question 4:** What challenges exist with integrating COSYSMO into early DOD decisions?

Several levels of challenges exist for integrating COSYSMO and external models like it into DODAF decisions. The most difficult challenge is standard terminology and uses across all the aspects of the DOD and the systems engineering discipline. The continual change of frameworks for the DODAF presents large-scale configuration management issues. For instance, when an MDAP enters into the process, it assumes the current framework; this requires that backward compatibility and integration concerns will ultimately occur when updates and future modifications happen. Similarly, the lack of standard terminology negates some of the possible automated results discussed as often the framework and terminology change so quickly that the various domains are not keeping pace. Mappings and documented links provide reproducible changes and updates. Otherwise data and terminology use will be ad hoc and not adaptable to the continued growing complexity and rapid changes in defense systems. This integration improves data availability and quality concerns for system modeling use in cost estimation.

Similar to the idea of domains failing to keep pace, adequate training and development of the defense professionals in MBSE will take time, but must also sustain the current systems engineering knowledge. Both a working knowledge of systems engineering theory and the practice of modeling were essential for the proposed method. Continued acceptance of the COSYSMO parametric technique seems likely, but general acceptance of COSYSMO as more beneficial to current cost estimation relationships may also take time. As mentioned, cost estimators support and input to COSYSMO integration or an ensemble of models that the cost estimation and program manager's communities recommend will help external cost model integration. The calibration required for a full DOD implementation is a significant endeavor and will uncover vulnerabilities and data conflicts.

Finally, at the system modeling level, pulling generalizable relationships has aspects of both an art and science. The science includes an accurate representation of the system; the art encompasses enabling the use of the model for its intended purpose and audience. Without best practices and or modeling conventions across the DOD, COSYSMO integration may have difficulty overcoming organization resistance to change. The time required to achieve model validation will amplify this organizational resistance. A small subset of user-influenced standards for model use and communication in early trade decisions will provide reproducible and user defined utility. Getting the system engineers, systems architects, analysts, cost estimators, and acquisitions professionals together can help overcome this inertia and support bringing it all together with MBSE. The integration has the potential to support able, agile, and affordable defense systems for the warfighter. This data driven process represents faster, less biased, and risk aware information for decision makers.

## 5.3   Recommendations For Future Work

Most of this work was focused on highlighting the link between MBSE, SysML, cost estimation, and highlight its impact to MDAP. A single instance and methodology represent the extent of model integration in a developed web-based tool. The exploratory work's greatest contribution remains: suggest a means to bring cost estimators and early system designers together with the use of MBSE and identifying the relationships between various models and DOD organizations. There are several areas where future work seems promising. The following highlights the top three of those areas for future action.

The author's relatively straightforward and crude method of parsing XML needs refinement. Machine learning and computational statistics techniques exist for the development and training of an algorithm to identify, classify, and predict responses. As the system model repositories and case studies of successful MDAP programs using MBSE emerge over time; data sets will develop and provide the statisticians, computer scientists, and operations research analysts opportunities to refine this work. The expertise exists in these three domains to improve this aspect of the work as soon as the data is available. Depending on the field, data may be available now.

Next, challenging and expanding the methodology proposed to include other constructive models, modeling tools, and languages would strengthen and improve the analysis approach and provide support to the challenges of other cost estimation areas for the DOD. Software cost estimation, UML, and the COCOMO series seem to have several parallels with this work. The choice for exploring COSYSMO occurred for its relative ease of use, documentation, cost estimation method, and availability. Applying a systems engineering approach to identify requirements for effective integration of the result of this work or any extensions, would help improve external model integration. A recommended ensemble of DOD models linked to the KPP aspects of JCIDS seems promising.

Finally, exploring the cost estimation challenges of a system of systems would be a long term and noteworthy contribution to the systems engineering domain and extension of this works concepts. The systems of systems focus will experience similar struggles to find suitable systems models. There is a recent system of systems variant of COSYSMO completed in 2007 [55], which may provide a starting point for this work.

THIS PAGE INTENTIONALLY LEFT BLANK

# APPENDIX: Water Distiller Project

This Appendix provides one example of the XML output from the water distiller humanitarian relief scenario and then provides, the R code utilized for the automated methods discussed for finding each of the four COSYSMO size drivers. The XML output for the system model is relatively large. This file size is the rationale for providing only one XML output example. The parametric diagram example given, replicates Figure 3.19. This output file was the shortest at 239 lines of code and only a small portion of those lines contain information relevant to COSYSMO. A reminder before using the algorithms provided, check for specific updates to the underlying schema in Innoslate. During publishing of this thesis, Innoslate made modifications to its schema file, and some of the schemaId, labels, and relationship types changed.

## A.1 Select XML Output

This is the XML output for the SysML parametric diagram in Figure 3.19.

XML Output of Figure 3.19 Water Distiller Parametric Diagram

```
1  <?xml version="1.0" encoding="UTF-8" standalone="yes"?>
2  <innoslate createdDateTime="2016-05-01 03:38:46" xsdVersion="3.0">
3       <schema lmlVersion="1.6">
4           <schemaClass id="C18">
5               <name>Equation</name>
6               <description>An Equation entity specifies an equation (
                    mathematical or logical) that can be used to
                    describe a part of the model.</description>
7               <hidden>false</hidden>
8               <locked>true</locked>
9               <abstract>false</abstract>
10              <schemaProperty id="P19">
11                  <name>Value</name>
12                  <description>Value represents this Equation's text
                        .</description>
13                  <hidden>false</hidden>
14                  <locked>true</locked>
15                  <type>TEXT</type>
16                  <order>1</order>
```

```
17              </schemaProperty >
18          </schemaClass >
19          <schemaRelation  id ="R36">
20              <name>decomposed  by </name>
21              <description >Decomposed  by  identifies  the  children  of
                    this  entity . </ description >
22              <hidden >false </hidden >
23              <locked >true </locked >
24              <inverseRelationId >R37</inverseRelationId >
25              <schemaProperty  id ="P38">
26                  <name>Multiplicity </name>
27                  <description >Multiplicity  represents  if  this
                        decomposition  has  multiple  instances  of  the  child
                        . </ description >
28                  <hidden >false </hidden >
29                  <locked >true </locked >
30                  <type >MULTIPLICITY </type >
31                  <order >2147483647 </order >
32              </schemaProperty >
33          </schemaRelation >
34          <schemaRelation  id ="R4T">
35              <name>related  to </name>
36              <description >Related  to  identifies  the  entity  that  ties
                    in  a  peer−to−peer  way  with  this  entity . </ description >
37              <hidden >false </hidden >
38              <locked >true </locked >
39              <inverseRelationId >R4V</inverseRelationId >
40              <schemaProperty  id ="P4W">
41                  <name>Context </name>
42                  <description >Context  represents  a  description  of
                        this  relation . </ description >
43                  <hidden >false </hidden >
44                  <locked >true </locked >
45                  <type >TEXT</type >
46                  <order >2147483647 </order >
47              </schemaProperty >
48          </schemaRelation >
49          <schemaRelation  id ="R37">
50              <name>decomposes </name>
51              <description >Decomposes  identifies  the  parent  of  this
```

```
                            entity.</description>
52              <hidden>false</hidden>
53              <locked>true</locked>
54              <inverseRelationId>R36</inverseRelationId>
55              <schemaProperty id="P38">
56                  <name>Multiplicity</name>
57                  <description>Multiplicity represents if this
                        decomposition has multiple instances of the child
                        .</description>
58                  <hidden>false</hidden>
59                  <locked>true</locked>
60                  <type>MULTIPLICITY</type>
61                  <order>2147483647</order>
62              </schemaProperty>
63          </schemaRelation>
64          <schemaRelation id="R4V">
65              <name>relates</name>
66              <description>Relates identifies the peer-to-peer entity
                        that is tied to this entity.</description>
67              <hidden>false</hidden>
68              <locked>true</locked>
69              <inverseRelationId>R4T</inverseRelationId>
70              <schemaProperty id="P4W">
71                  <name>Context</name>
72                  <description>Context represents a description of
                        this relation.</description>
73                  <hidden>false</hidden>
74                  <locked>true</locked>
75                  <type>TEXT</type>
76                  <order>2147483647</order>
77              </schemaProperty>
78          </schemaRelation>
79          <targetedRelation>
80              <sourceId>C18</sourceId>
81              <relationId>R4V</relationId>
82              <targetId>C18</targetId>
83              <lowerLimit>0</lowerLimit>
84              <upperLimit>2147483647</upperLimit>
85              <popular>false</popular>
86          </targetedRelation>
```

```
87          <targetedRelation>
88              <sourceId>C18</sourceId>
89              <relationId>R36</relationId>
90              <targetId>C18</targetId>
91              <lowerLimit>0</lowerLimit>
92              <upperLimit>2147483647</upperLimit>
93              <popular>true</popular>
94          </targetedRelation>
95          <targetedRelation>
96              <sourceId>C18</sourceId>
97              <relationId>R37</relationId>
98              <targetId>C18</targetId>
99              <lowerLimit>0</lowerLimit>
100             <upperLimit>2147483647</upperLimit>
101             <popular>true</popular>
102         </targetedRelation>
103         <targetedRelation>
104             <sourceId>C18</sourceId>
105             <relationId>R4T</relationId>
106             <targetId>C18</targetId>
107             <lowerLimit>0</lowerLimit>
108             <upperLimit>2147483647</upperLimit>
109             <popular>false</popular>
110         </targetedRelation>
111     </schema>
112     <labels/>
113     <database>
114         <entity id="e8MFN2">
115             <name>boiling : Phase Change Heat Xfer Equation</name>
116             <description>&lt;br&gt;</description>
117             <hidden>false</hidden>
118             <locked>false</locked>
119             <schemaClassId>C18</schemaClassId>
120             <number></number>
121             <stringAttribute schemaPropertyId="P19">
122                 <value>{q rate=m rate*l heat}</value>
123             </stringAttribute>
124             <simulationData>
125                 <type>SERIAL</type>
126                 <controlStructure id="24f17539-d706-44c3-bc3c-0008
```

ed173e80">
127                  \<type\>START\</type\>
128                  \<sucessorStructure id="74209f40−c8d5−4470−b024−
                  d15a6d382ec6">
129                   \<type\>END\</type\>
130                  \</sucessorStructure\>
131               \</controlStructure\>
132            \</simulationData\>
133        \</entity\>
134        \<entity id="e7PJM1">
135           \<name\>[Constraint Block] Distiller Isobaric Heat Balance
           [composition of equations]\</name\>
136           \<description\>\</description\>
137           \<hidden\>false\</hidden\>
138           \<locked\>false\</locked\>
139           \<schemaClassId\>C18\</schemaClassId\>
140           \<stringAttribute schemaPropertyId="P19">
141              \<value\>\</value\>
142           \</stringAttribute\>
143           \<simulationData\>
144             \<type\>SERIAL\</type\>
145             \<controlStructure id="974c760b−bbc3−4c7d−9d0e−
             db8105cf5dbd">
146              \<type\>START\</type\>
147              \<sucessorStructure id="3fd981ac−7dfd−4976−b0fe
              −32dba609750b">
148               \<entityId\>e13CT2\</entityId\>
149               \<type\>SERIAL\</type\>
150               \<sucessorStructure id="745d6165−b5a4−4dad−
               b600−bbee557a3385">
151                \<entityId\>e6P7F3\</entityId\>
152                \<type\>SERIAL\</type\>
153                \<sucessorStructure id="0bf83d36−22bc−412
                e−88c7−b4c25da16faa">
154 \<entityId\>e8MFN2\</entityId\>
155 \<type\>SERIAL\</type\>
156 \<sucessorStructure id="85697a26−9b03−49bf−bd5b−13a83266b2ef">
157    \<type\>END\</type\>
158 \</sucessorStructure\>
159                 \</sucessorStructure\>

```
160                        </sucessorStructure>
161                      </sucessorStructure>
162                  </controlStructure>
163            </simulationData>
164            <diagramLayout type="LML_HIERARCHY">
165                <layout>{"vertexes":[{"id":"e7PJM1","isRoot":true ,"
                     width":100 ,"height":70 ,"y":50 ,"x":200},{"id":"
                     e7PJM1_e13CT2","isRoot":false ,"width":100 ,"height
                     ":70 ,"y":170 ,"x":50},{"id":"e7PJM1_e6P7F3","
                     isRoot":false ,"width":100 ,"height":70 ,"y":170 ,"x"
                     :200},{"id":"e7PJM1_e8MFN2","isRoot":false ,"width
                     ":100 ,"height":70 ,"y":170 ,"x":350}],"edges":[{"id
                     ":"e7PJM1-e7PJM1_e13CT2"},{"id":"e7PJM1-
                     e7PJM1_e6P7F3"},{"id":"e7PJM1-e7PJM1_e8MFN2"}],"
                     shapes":[],"serialVersion":2 ,"checksum":"'e7PJM1'
                     ,'e8MFN2','e6P7F3','e13CT2'"}</layout>
166            </diagramLayout>
167        </entity>
168        <entity id="e13CT2">
169            <name>s1: Single Phase Heat Xfer Equation </name>
170            <description></description>
171            <hidden>false </hidden>
172            <locked>false </locked>
173            <schemaClassId>C18</schemaClassId>
174            <number></number>
175            <stringAttribute schemaPropertyId="P19">
176                <value>q rate=(th-tc)*m rate/s heat}</value>
177            </stringAttribute>
178            <simulationData>
179                <type>SERIAL</type>
180                <controlStructure id="631e0d6c-788e-4ddb-9fc7-
                     d8da6c1fda74">
181                    <type>START</type>
182                    <sucessorStructure id="14f066de-f658-4e01-9333-0
                         e8c42c68f4d">
183                        <type>END</type>
184                    </sucessorStructure>
185                </controlStructure>
186            </simulationData>
187        </entity>
```

102

```xml
188          <entity id="e6P7F3">
189              <name>condensing : Phase Change Heat Xfer Equation </name>
190              <description>&lt;br&gt;</description>
191              <hidden>false</hidden>
192              <locked>false</locked>
193              <schemaClassId>C18</schemaClassId>
194              <number></number>
195              <stringAttribute schemaPropertyId="P19">
196                  <value>{q rate=m rate*l heat}</value>
197              </stringAttribute>
198              <simulationData>
199                  <type>SERIAL</type>
200                  <controlStructure id="24cc7305-64e4-4792-bbc9-04000
                         f8c019b">
201                      <type>START</type>
202                      <sucessorStructure id="ce6ef2eb-8b7b-40f4-89f1-
                             c9cfb94fd7f2">
203                          <type>END</type>
204                      </sucessorStructure>
205                  </controlStructure>
206              </simulationData>
207          </entity>
208          <relationship>
209              <sourceId>e8MFN2</sourceId>
210              <schemaRelationId>R37</schemaRelationId>
211              <targetId>e7PJM1</targetId>
212          </relationship>
213          <relationship>
214              <sourceId>e7PJM1</sourceId>
215              <schemaRelationId>R36</schemaRelationId>
216              <targetId>e13CT2</targetId>
217          </relationship>
218          <relationship>
219              <sourceId>e7PJM1</sourceId>
220              <schemaRelationId>R36</schemaRelationId>
221              <targetId>e6P7F3</targetId>
222          </relationship>
223          <relationship>
224              <sourceId>e7PJM1</sourceId>
```

```
225                <schemaRelationId >R36</schemaRelationId >
226                <targetId >e8MFN2</targetId >
227          </relationship >
228          <relationship >
229                <sourceId >e13CT2</sourceId >
230                <schemaRelationId >R37</schemaRelationId >
231                <targetId >e7PJM1</targetId >
232          </relationship >
233          <relationship >
234                <sourceId >e6P7F3</sourceId >
235                <schemaRelationId >R37</schemaRelationId >
236                <targetId >e7PJM1</targetId >
237          </relationship >
238      </database >
239  </innoslate >
```

## A.2   R Scripts to Parse XML Into COSYSMO Size Driver Parameters

The following four R scripts serve as an initial method to highlight the utility of the XML model output. The title of the script denotes which diagram the XML output is showing. The parametric diagram is presented first because it relates to the XML output example provided. This information corresponds to the number of critical system algorithms in COSYSMO. Table A.1 provides the SysML diagram used from this thesis, the COSYSMO parameter estimated, and the corresponding script that was used in the automated method for clarity ease to the reader. The output of each script is captured and displayed for reference.

Table A.1: COSYSMO Size Drivers and Corresponding R Scripts

| Driver Name | SysML Diagram-R Script and Thesis Figure |
|---|---|
| Number of requirements | Package Diagram-(PKG.R) Figure 3.6 |
| Number of major interfaces | Internal Block Diagram-(IBD.R) Figure 3.18 |
| Number of critical algorithms | Parametric Diagram-(PAR.R) Figure 3.19 |
| Number of operational scenarios | Use Case Diagram-(UC.R) Figure 3.20 |

### A.2.1 Parametric Diagram (PAR.R)- Figure 3.19

```
 1 #PAR.R
 2 install.packages("XML")# package required in R
 3 library(XML)
 4
 5 # read distller model parametric diagram export from Innoslate #parse into a tree
 6 distiller<-xmlParse("C:/Users/denni/Desktop/EDWARDS_DENNIS NPS_THESIS_LaTeX_Template/
      code/Parametric.xml")
 7
 8 #find instances of C18, C18 is the schema entity class for an equation
 9 C18 <- xpathSApply (distiller, "//entity[schemaClassId='C18']")
10
11 # Equations P19 is the string attribute for an expression.
12 P19 <- sapply (C18, function (x) xmlValue (xmlChildren (x)[["stringAttribute"]]))
13
14 #shown in tabular form 4 total instances,  3 which include mathematical expression
15 table(P19)
```

```
 1 # PAR.R Output P19
 2 N/A                    {q rate=m rate*l heat}        q rate=(th-tc)*m rate/s heat}
 3  1                               2                                        1
```

## A.2.2  Package Diagram (PKG.R)- Figure 3.6

```r
1 # PKG.R
2 install.packages("XML")# package required in R
3 library(XML)
4
5 # read distller model entire model export from Innoslate #parse into a tree/ this is
      the whole file
6 distiller<-xmlParse("C:/Users/denni/Desktop/EDWARDS_DENNIS NPS_THESIS_LaTeX_Template/
      code/Package.xml")
7
8 #find instances of C8, C8 is the schema entity class for an asset
9 C8 <- xpathSApply (distiller, "//entity[schemaClassId='C8']")
10
11 # L6W is package designation# Note check the current schema. L6W changed to L6X in
      recent version change
12 labelId <- sapply (C8, function (x) xmlValue (xmlChildren (x)[["labelId"]]))
13
14 #shown in tabular form #L6W is an package diagram and L6K is annotation for a block
      #17 assets are not labeled
15 sum (labelId == "L6W", na.rm=TRUE)
16 table(labelId)
```

```r
1 # PKG.R Output labelId
2     L6K L6W
3      25  37
```

### A.2.3 Internal Block Diagram (IBD.R)- Figure 3.18

```
 1 #IBD.R
 2 install.packages("XML")# package required in R
 3 library(XML)
 4
 5 # read distller model parametric diagram export from Innoslate #parse into a tree
 6 distiller<-xmlParse("C:/Users/denni/Desktop/EDWARDS_DENNIS NPS_THESIS_LaTeX_Template/
       code/IBD.xml")
 7
 8 #find instances of CE, CE is the schema entity class for a port
 9 CE <- xpathSApply (distiller, "//entity[schemaClassId='CE']")
10
11 # Equations PF is the string attribute for an expression giving the directionality of
       the port as In, Out, In/Out, None
12 PF <- sapply (CE, function (x) xmlValue (xmlChildren (x)[["stringAttribute"]]))
13
14 #shown in tabular form this captures the number of In and Out, Bi Directional and
       Unknown Ports
15 table(PF)
```

```
 1 # IBD.R Output PF
 2         In     In and Out       None        Out
 3         12          2            2          14
```

## A.2.4    Use Case Diagram (UC.R)- Figure 3.20

```
1 # UC.R
2 install.packages("XML")# package required in R
3 library(XML)
4
5 # read distller entire model export from Innoslate #parse into a tree/ this is the
      whole file
6 distiller<-xmlParse("C:/Users/denni/Desktop/EDWARDS_DENNIS NPS_THESIS_LaTeX_Template/
      code/UC.xml")
7
8 #find instances of C1, C1 is the schema entity class for an action
9 C1 <- xpathSApply (distiller, "//entity[schemaClassId='C1']")
10
11 # L5Z is use case designation
12 labelId <- sapply (C1, function (x) xmlValue (xmlChildren (x)[["labelId"]]))
13
14 #shown in tabular shows 6 total instances,1 which is L5Z, a use case #L5Q is an
      activity diagram
15 sum (labelId == "L5Z", na.rm=TRUE)
16 table(labelId)
```

```
1 # UC.R Output labelId
2     L5Q L5Z
3       5   1
```

THIS PAGE INTENTIONALLY LEFT BLANK

# List of References

[1] The MITRE Corporation, *The MITRE Systems Engineering Guide [ebook version]*, Bedford, MA, 2014. [Online]. Available: https://www.mitre.org/publications/systems-engineering-guide/about-the-seg

[2] M. Crawford. (2012, Sep.). Certified system engineers are in high demand. [Online]. Available: https://www.asme.org/career-education/articles/certification/certified-systems-engineers-are-in-high-demand

[3] INCOSE. (2014, June). A world in motion: Systems engineering vision 2025. Systems Engineering Vision 2025 Project Team of the International Council on Systems Engineering (INCOSE). San Diego, CA. [Online]. Available: http://www.incose.org/AboutSE/sevision

[4] M. Dwyer, B. Cameron, and Z. Szajnfarber, "A framework for studying cost growth on complex acquisition programs," *Syst. Eng.*, vol. 18, no. 6, pp. 568–583, Nov. 2015.

[5] C. Calvano and P. John, "Systems engineering in an age of complexity," *Syst. Eng.*, vol. 7, no. 1, pp. 25–34, Dec. 2003.

[6] A. G. Sedmak, Z. S. Taylor, and W. A. Riski, "Establishing the technical foundation: Materiel solution analysis is more than selecting an alternative," *Defense Acquisition Journal*, vol. 22, no. 4, pp. 364–393, Oct. 2015.

[7] Government Accountability Office. (2009, Mar.). GAO cost estimating and assessment guide: Best practices for developing and managing capital program costs GAO-09-3SP. GAO. Washington, DC. [Online]. Available: http://www.gao.gov/new.items/d093sp.pdf

[8] Government Accountability Office. (2012, Mar.). Defense acquisition: Assessment of selected weapon programs GAO-12-400SP. GAO. Washington, DC. [Online]. Available: http://www.goa.gov/assets/590/589695.pdf

[9] M. zur Muehlen, D. Hamilton, and R. Peak, "Integration of M&S (Modeling and Simulation), software design and DODAF (Department of Defense Architecture Framework (RT 24)," Stevens Inst. of Technol., Hoboken, NJ, Tech. Rep. A013, Apr. 2012.

[10] *Operation of the Defense Acquisition System, DOD Directive 5000.02,*, Under Secretary of Defense (AT&L), Washington, DC, 2015.

[11] *Joint Capabilities Integration and Development System JCIDS, CJCSI 3170.01I*, Under Secretary of Defense (AT&L), Washington, DC, 2015.

[12] S. Friedenthal, A. Moore, and R. Steiner, *A Practical Guide to SysML: The Systems Modeling Language*, 1st ed. Burlington, MA: Morgan Kaufmann OMG Press, 2008, pp. 359.396.

[13] S. Friedenthal, A. Moore, and R. Steiner, *A Practical Guide to SysML: The Systems Modeling Language [Knovel Version]*, 2nd ed. Morgan Kaufmann OMG Press/Elsevier, 2012, pp. 393-429. [Online]. Available: http://app.knovel.com/hotlink/toc/id: kpPGSMLTSX/practical-guide-sysml/practical-guide-sysml

[14] S. Friedenthal, A. Moore, and R. Steiner, *A Practical Guide to SysML: The Systems Modeling Language*, 3rd ed. Waltham, MA: Morgan Kaufmann OMG Press/Elsevier, 2014, pp 387-415.

[15] Distiller example. Object name Distiller Model Example. [Online]. Available: http://booksite.elsevier.com/9780123852069/distiller_example_html/Distiller_Example.html. Accessed Apr. 12, 2016.

[16] Defense Acquisition University. (2013, Sep.). Defense acquisition guidebook. Defense Acquisition University Press. Fort Belvoir, VA. [Online]. Available: https://acc.dau.mil/docs/dag_pdf/dag_complete.pdf

[17] B. S. Blanchard and W. J. Fabrycky, *Systems Engineering and Analysis*, 5th ed., W. J. Fabrycky and J. H. Mize, Eds. Prentice Hall, 2011, pp. 23-126.

[18] G. Hagan. (2015, Sep.). Glossary of defense acqusistions acronyms & terms. Defense Acquisition University Press. Fort Belvoir, VA. [Online]. Available: http://www.dau.mil/publications/publicationsDocs/Glossary_16th&20_ed.pdf

[19] M. W. Maier and E. Rechtin, *The Art of Systems Architecting*, 3rd ed. Boca Raton, FL: CRC Press Taylor & Francis Group, 2009.

[20] D. D. Walden, G. J. Roedler, K. J. Forsberg, R. D. Hamelin, and T. M. Shortell. (2015, Jan.). Systems engineering handbook: A guide for systems lifecycle process and activities. INCOSE. Hoboken, NJ. [Online]. Available: http://www.incose.org/ProductsPublications/sehandbook

[21] *Cost Analysis Guidance and Procedures, DOD Instruction 5000.73*, Cost Assessment and Program Evaluation, Washington, DC, 2015.

[22] J. P. Elm and D. R. Goldenson, "The business case for systems engineering: Comparison of defense-domain and no-defense projects," Carnegie Mellon Univ. Software Eng. Inst., Pittsburgh, PA, Tech. Rep. CMU/SEI-2014-SR-013, June 2014.

[23] M. J. Sullivan. (2011, Mar. 29). DOD cost overruns: Trends in nunn-mccurdy breaches and tools to manage weapon systems acquisition costs. GAO. [Online]. Available: http://www.gao.gov/products/GAO-11-499T

[24] BKCASE Editorial Board. (2016, Mar. 25). The guide to the systems engineering body of knowledge (SEBoK), v. 1.6. [Online]. Available: http://sebokwiki.org

[25] Department of Defense Deputy Chief Information Officer (DCIO), *Department of Defense Architectural Framework Version 1.5*. Washington, DC: DOD, 2007.

[26] Department of Defense Deputy Chief Information Officer (DCIO), *Department of Defense Architectural Framework Version 2.0*. Washington, DC: DOD, 2009.

[27] Department of Defense Deputy Chief Information Officer (DCIO), *Department of Defense Architectural Framework Version 2.02*. Washington, DC: DOD, 2010. [Online]. Available: http://dodcio.defense.gov/Library/DoDArchitectureFramework.aspx

[28] S. H. Dam, *DOD Architecture Framework 2.0- A Guide to Applying Systems Engineering to Develop Integrated, Executable Architectures*, E. Steiner and S. Campbell, Eds. Manassas, VA: SPEC Innovations, 2014.

[29] A. Morkevicius, S. Gudas, and D. Silingas. (2010, June). Model-driven quantitative performance analysis of UPDM-Based enterprise architecture. No Magic Inc. Allen, TX. [Online]. Available: https://www.nomagic.com/support/whitepapers/dodaf-modaf-updm/model-driven-quantitative-performance-analysis-of-updm-based-enterprise-architecture.html

[30] J. L. Hayden and A. Jeffries, "On using SysML, DODAF 2.0 and UPDM to model the architecture for the NOAA's joint polar satellite system (JPSS) ground system (GS)," presented at Space Ops Conference, Stockholm, Sweden, 2012.

[31] R. E. Giachetti, "Evaluation of the DODAF meta-model's support of systems engineering," *Procedia Comput. Sci.*, vol. 61, pp. 254–260, Nov. 2015.

[32] R. J. Madachy, "Heuristic risk assessment using cost factors," *IEEE Software*, vol. 14, no. 3, pp. 51–59, May 1997.

[33] LML Steering Committee, "LML specification 1.0," LML Steering Committee, Tech. Rep. LML Specification 1.0, October 2013. [Online]. Available: http://www.lifecyclemodeling.org/spec/1.0

[34] S. Gao and et al, "W3C XML schema definition language (XSD) 1.1 part 1: Structures," World Wide Web Consortium, Tech. Rep. XML Schema Definition Language (XSD) 1.1 Part 1: Structures, Apr. 2012. [Online]. Available: https://www.w3.org/TR/2012/REC-xmlschema11-1-20120405/

[35] G. O. Langford, *Engineering Systems Integration: Theory, Metrics, and Methods*. Boca Raton, FL: CRC Press Taylor & Francis Group, 2012, pp. 123-278.

[36] The DODAF 2.0 Meta Model. Object name DM2 HTML 130326. [Online]. Available: http://www.ideasgroup.org/dm2/. Accessed Nov. 12, 2015.

[37] J. A. Zachman. (2008, Jan.). John zachman's concise definition of the zachman framework. [Online]. Available: https://www.zachman.com/about-the-zachman-framework

[38] D. Taniar and J. W. Rahayu, *Web Semantics & Ontology*. Hershey, PA: Idea Group Publishing, 2006.

[39] R. Valerdi, M. Dabkowski, and I. Dixit, "Reliability improvement of major defense acquisition program cost estimates mapping DODAF to COSYSMO," *Syst. Eng.*, vol. 18, no. 5, pp. 530–547, Dec. 2015.

[40] G. K. Mislick and D. A. Nussbaum, *Cost Estimation: Methods and Tools*. Hoboken, NJ: John Wiley & Sons, 2015.

[41] G. A. Garrett, *Cost Estimating and Contract Pricing: Tools, Techniques and Best Practices*. Riverwoods, IL: CCH, 2008.

[42] Director Cost Assessment and Program Evaluation. (2011, Feb.). FY 2010 annual report on cost assessment activities. CAPE. Washington, DC. [Online]. Available: http://www.cape.osd.mil/files/Reports/CA_AR_071411.pdf

[43] Director Cost Assessment and Program Evaluation. (2012, Feb.). FY 2011 annual report on cost assessment activities. CAPE. Washington, DC. [Online]. Available: http://www.cape.osd.mil/files/Reports/CA_AR_20120508.pdf

[44] Director Cost Assessment and Program Evaluation. (2015, Feb.). FY 2014 annual report on cost assessment activities. CAPE. Washington, DC. [Online]. Available: http://www.cape.osd.mil/files/Reports/CA_AR_20150401.pdf

[45] "Campaign analysis, force structure and joint capability planning," class notes for Joint Campaign Analysis, Dept. of Operations Res., Naval Postgraduate School, Monterey, CA, winter 2016.

[46] C. F. Schied. (2009, July). Program manager e-tool kit: Hierarchy of models and simulations. [Online]. Available: https://acc.dau.mil/CommunityBrowser.aspx?id=294530

[47] Cost Assessment and Program Evaluation. (n.d.). About CAPE: Introduction. Cost Assessment and Program Evaluation. [Online]. Available: http://www.cape.osd.mil/. Accessed Jan. 5, 2016.

[48] *Department of Defense Standard Practice Work Breakdown Structures for Defense Materiel Items*, MIL-STD-881C, 2011.

[49] Defense Cost and Resource Center. (n.d.). Enhancing DOD cost analysis: Plan development and contracting. Defense Cost and Resource Center. [Online]. Available: http://dcarc.cape.osd.mil/csdr/Planning.aspx. Accessed Jan. 5, 2016.

[50] R. J. Madachy, "Systems engineering cost estimation workbook," Department of Systems Engineering, Naval Postgraduate School, unpublished, September 2015.

[51] R. Valerdi, *The Constructive Systems Engineering Cost Model (COSYSMO): Quantifying the Costs of Systems Engineering Effort in Complex Systems*. Saarbrucken, Germany: VDM Verlag Dr. Muller, 2008.

[52] J. Fortune, "Estimating systems engineering reuse with the constructive systems engineering model (COSYSMO 2.0)," Ph.D dissertation, Dept. Ind. and Syst. Eng., Univ. of Southern California, Los Angeles, CA, 2009.

[53] R. Madachy and R. Valerdi, "Automating systems engineering risk assessment," in *Proceedings of the 8th Annual Conference on Systems Engineering Research*, Hoboken, NJ, 2010.

[54] C. Smartt and S. Ferreira, "Advancing systems engineering in support of the bid and proposal process," *Syst. Eng.*, vol. 14, no. 3, pp. 255–266, Oct. 2010.

[55] J. A. Lane and R. Valerdi, "Synthesizing SoS concept for use in cost modeling," *Syst. Eng.*, vol. 10, no. 4, pp. 297–308, Aug. 2007.

[56] R. Madachy. (2016, May 4). COSYSMO - Constructive Systems Engineering Model. [Online]. Available: http://csse.usc.edu/tools/COSYSMO.php

[57] G. Wang, R. Valerdi, B. Boehm, and A. Shernoff, "Proposed modification to COSYSMO estimating relationship," in *INCOSE International Symposium*, vol. 18, no. 1, Utrecht, The Netherlands, 2008, pp. 249–262.

[58] Lockheed Martin, private communication, Feb. 2016.

[59] R. J. Madachy. (2014, Aug. 21). Total ownership cost modeling. Naval Postgraduate School. [Online]. Available: http://hdl.handle.net/10945/44234

[60] K. Liu and et al, "Better requirements decompositions can improve cost estimation of systems engineering and human systems integration," in *Proceedings of the 8th Annual Conference on Systems Engineering Research*, Hoboken, NJ, 2010.

[61] R. Valerdi. (2007, Mar. 5). COSYSMO 1.1. [Online]. Available: http://cosysmo.mit.edu/downloads/

[62] A. W. Wynmore, *Model-Based Systems Engineering: An Introduction to the Mathematical Theory of Discrete Systems and to the Triocotyledon Theory of System Design*. Boca Raton, FL: CRC Press, 1993.

[63] Vitech. (2011, Oct.). A primer for model-based systems engineering. Vitech Corporation. [Online]. Available: http://www.vitechcorp.com/resources/mbse.shtml

[64] J. A. Estefan. (2008, May). Survey of model-based systems engineering (MBSE) methodologies Rev. B. INCOSE MBSE Initiative. Pasadena, CA. [Online]. Available: http://www.omgsysml.org/MBSE_Methodology_Survey_RevB.pdf

[65] Object Management Group, "OMG systems modeling language (OMG SysML) version 1.4," Object Management Group, Needham, MA, Tech. Rep. Systems Modeling Language (OMG SysML) Version 1.4, Sep. 2015. [Online]. Available: http://www.omg.org/spec/SysML/1.4/

[66] W. K. Vaneman, "Enhancing model-based systems engineering with the lifecycle modeling language," in *Proceedings of the 10th Annual Systems Conference*, Orlando, FL, 2016, pp. 457–463.

[67] S. Friedenthal and R. Burkhart. (2015, Aug.). Evolving SysML and the system modeling environment to support MBSE. INSIGHT. [Online]. Available: http://dx.doi.org/10.1002/inst.12020. pp. 39-41.

[68] T. Weilkiens, *Systems Engineering with SysML/UML Architecture: Modeling, Analysis, Design*. Burlington, MA: The MK/OMG Press/Elsevier, 2007.

[69] R. M. Pospisal, "Application of executable architecture in early concept evaluation," M.S. thesis, Dept. Syst. Eng., Air Force Institute of Technology, Wright-Patterson AFB, OH, 2015.

[70] SPEC Innovations. (2015, Jan.). Innoslate users guide 3.4. SPEC Innovations. Manassas, VA. [Online]. Available: http://docs.innoslate.com/letest/users-guide/

[71] R Core Team, Vienna, Austria. (2015, Dec.). R: A language and environment for statistical computing. R Foundation for Statistical Computing. Vienna, Austria. [Online]. Available: https://www.R-project.org/

[72] Duncan Temple Lang and the CRAN Team. (2016). Xml: Tools for parsing and generating xml within r and s-plus r package version 3.98-1.4. [Online]. Available: https://CRAN.R-project.org/package=XML. Accessed Apr. 15, 2016.

[73] Galorath Incorporated. (2015, Nov.). SEER for systems engineering: Detailed estimation of systems engineering effort. Galorath Inc. [Online]. Available: http://galorath.com/wp-content/upload/2015/11/SEER-SYS-Data-Sheet-Nov2015.pdf

[74]  R. Madachy. (2016, May 4). SysML COSYSMO Tool. [Online]. Available: http: //csse.usc.edu/tools/SysML_COSYSMO

[75]  D. Jacques and R. Madachy, "Model-Centric UAV ISR Analysis," presented at Systems Engineering Research Center, 7th Annual SERC Sponsor Research Review, Washinton, DC, December 3, 2015.

THIS PAGE INTENTIONALLY LEFT BLANK

# Initial Distribution List

1. Defense Technical Information Center
   Ft. Belvoir, Virginia

2. Dudley Knox Library
   Naval Postgraduate School
   Monterey, California