Applying the Engineering Systems Multiple-Domain Matrix Framework to Nanosatellite Space Systems

by

Kyle B. Hurst

Bachelor of Science, Engineering Management
United States Military Academy at West Point, 2008

Submitted to the System Design and Management Program
in Partial Fulfillment of the Requirements of the Degree of

Master of Science in Engineering and Management

at the

Massachusetts Institute of Technology

June 2017

© 2017 Kyle B. Hurst. All rights reserved.

The author hereby grants to MIT permission to reproduce and to distribute publicly paper and electronic copies of this thesis document in whole or in part in any medium now known or hereafter created.

Signature redacted

Signature of Author

Signature redacted

Kyle B. Hurst
System Design and Management Program
May 01, 2017

Certified by

Signature redacted

Dr. Donna H. Rhodes
Principal Research Scientist, Socio-technical Systems Research Center
Director, Systems Engineering Advancement Research Initiative
Thesis Advisor

Accepted by

Signature redacted

Joan S. Rubin
System Design and Management Program
Executive Director

Signature redacted
Disclaimer

The views expressed in this thesis are those of the author and do not reflect the official policy or position of the U.S. Army, Department of Defense, or the U.S. Government
Applying the Engineering Systems Multiple-Domain Matrix Framework to Nanosatellite Space Systems

by

Kyle B. Hurst

Submitted to the System Design and Management Program on May 01, 2017
in Partial Fulfillment of the Requirements for the Degree of
Master of Science in Engineering and Management

Abstract

The nanosatellite industry is expanding rapidly, as academic and private institutions develop new technologies for experimentation on orbit. These “CubeSats” are resource constrained, complex socio-technical systems that have complicated interdependencies across multiple domains. To improve understanding and reduce ambiguity, systems engineers apply a variety of modeling frameworks to model system behavior. Introduced in 2007, the Engineering Systems Multiple-Domain Matrix (ES-MDM) framework addresses the interdependencies of a complex engineering system, such as a CubeSat, across five domains: environmental, social, functional, technical and process.

Using the Free-space Lasercom and Radiation Experiment (FLARE) CubeSat constellation as an example engineering system case, the ES-MDM is constructed using the qualitative knowledge construction framework to model and analyze the system drivers, stakeholders, objectives, function, objects and processes of the system.

The primary objective of this analysis is to provide a structured systems design approach for nanosatellite development that encompasses the entire system holistically. The second objective is to analyze the interactions and interdependencies within a highly-constrained system and determine key design nodes that are critical to system flexibility. The third objective is to evaluate the ability of the ES-MDM methodology to analyze a highly-constrained system. The fourth objective of this thesis is to provide recommendations for future work to improve the ES-MDM framework and the systems engineering field.

Thesis Supervisor: Dr. Donna H. Rhodes
Title: Principal Research Scientist, Socio-technical Systems Research Center
Acknowledgements

This thesis is dedicated to my wife, Elizabeth. Thank you for your love and support throughout our travels together. “God only know what I’d be without you.” – Brian Wilson

I would like to thank the U.S. Army and SDM for this amazing opportunity to broaden my educational horizons.

Lastly, I would like to thank my thesis advisor Dr. Donna Rhodes. Her invaluable contributions to my research and guidance throughout the thesis process was critical to the success of the thesis.
## Table of Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Abstract</td>
<td>3</td>
</tr>
<tr>
<td>Acknowledgements</td>
<td>4</td>
</tr>
<tr>
<td>List of Figures</td>
<td>7</td>
</tr>
<tr>
<td>List of Tables</td>
<td>8</td>
</tr>
<tr>
<td>Acronyms</td>
<td>9</td>
</tr>
<tr>
<td>Chapter 1: Introduction</td>
<td>11</td>
</tr>
<tr>
<td>1.1 Background and Motivation</td>
<td>11</td>
</tr>
<tr>
<td>1.2 Research Scope</td>
<td>12</td>
</tr>
<tr>
<td>1.3 Research Objectives</td>
<td>13</td>
</tr>
<tr>
<td>1.4 Organization of Thesis</td>
<td>13</td>
</tr>
<tr>
<td>Chapter 2: Literature Review</td>
<td>15</td>
</tr>
<tr>
<td>2.1 Introduction to Engineering Systems</td>
<td>15</td>
</tr>
<tr>
<td>2.1.1 What is Systems Engineering?</td>
<td>15</td>
</tr>
<tr>
<td>2.1.2 Why do we need Systems Engineering?</td>
<td>16</td>
</tr>
<tr>
<td>2.1.3 System value</td>
<td>17</td>
</tr>
<tr>
<td>2.1.4 Life-cycle properties</td>
<td>19</td>
</tr>
<tr>
<td>2.2 Current Modeling Framework Overview</td>
<td>20</td>
</tr>
<tr>
<td>2.2.1 Department of Defense Architecture Framework</td>
<td>20</td>
</tr>
<tr>
<td>2.2.2 Unified Modeling Language (UML)</td>
<td>22</td>
</tr>
<tr>
<td>2.2.3 System Modeling Language (SysML)</td>
<td>23</td>
</tr>
<tr>
<td>2.2.3 Object Process Methodology (OPM)</td>
<td>25</td>
</tr>
<tr>
<td>2.2.4 Design Structure Matrix (DSM)</td>
<td>28</td>
</tr>
<tr>
<td>2.2.5 Domain Mapping Matrix (DMM)</td>
<td>30</td>
</tr>
<tr>
<td>2.2.6 Multiple-Domain Matrix (MDM)</td>
<td>32</td>
</tr>
<tr>
<td>2.3 ES-MDM</td>
<td>33</td>
</tr>
<tr>
<td>2.3.1 ES-MDM Background</td>
<td>34</td>
</tr>
<tr>
<td>2.3.2 System Drivers</td>
<td>35</td>
</tr>
<tr>
<td>2.3.3 Stakeholders</td>
<td>36</td>
</tr>
<tr>
<td>2.3.4 Objectives</td>
<td>36</td>
</tr>
<tr>
<td>2.3.5 Functions</td>
<td>36</td>
</tr>
<tr>
<td>2.3.6 Objects</td>
<td>37</td>
</tr>
<tr>
<td>2.3.7 Activities</td>
<td>37</td>
</tr>
<tr>
<td>2.3.8 Existence Attributes</td>
<td>37</td>
</tr>
<tr>
<td>Section</td>
<td>Page</td>
</tr>
<tr>
<td>------------------------------------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>2.3.9 Interaction Analysis</td>
<td>38</td>
</tr>
<tr>
<td>Chapter 3: CubeSat Background and Evolution</td>
<td>39</td>
</tr>
<tr>
<td>3.1 Introduction to CubeSat Specifications</td>
<td>39</td>
</tr>
<tr>
<td>3.2 CubeSat Design</td>
<td>40</td>
</tr>
<tr>
<td>3.3 CubeSats as an Engineering System</td>
<td>43</td>
</tr>
<tr>
<td>Chapter 4: FLARE Case Study</td>
<td>45</td>
</tr>
<tr>
<td>4.1 FLARE Program Background</td>
<td>45</td>
</tr>
<tr>
<td>4.2 FLARE System Engineering Challenges</td>
<td>47</td>
</tr>
<tr>
<td>4.3 Motivation for ES-MDM Analysis</td>
<td>51</td>
</tr>
<tr>
<td>Chapter 5: Constructing the ES-MDM</td>
<td>53</td>
</tr>
<tr>
<td>5.1 Information Procurement and Abstraction</td>
<td>53</td>
</tr>
<tr>
<td>5.2 ES-MDM Development and Analysis</td>
<td>56</td>
</tr>
<tr>
<td>Flexible Design Opportunities</td>
<td>67</td>
</tr>
<tr>
<td>Network Analysis</td>
<td>74</td>
</tr>
<tr>
<td>5.3 Conclusions from ES-MDM analysis</td>
<td>78</td>
</tr>
<tr>
<td>Chapter 6: Conclusions and Future Work</td>
<td>81</td>
</tr>
<tr>
<td>6.1 ES-MDM CubeSat Applications</td>
<td>81</td>
</tr>
<tr>
<td>6.2 Assessment of ES-MDM Methodology</td>
<td>82</td>
</tr>
<tr>
<td>6.3 Future Work</td>
<td>84</td>
</tr>
<tr>
<td>Works Cited</td>
<td>86</td>
</tr>
</tbody>
</table>
List of Figures

Figure 1 – Systems Engineering Effort vs. Actual/Planned Cost Ratio
Figure 2 – DoDAF View Points
Figure 3 – UML Diagram Tree
Figure 4 – SysML Diagram Tree
Figure 5 – Example of an OPD with corresponding OPL using the OPM framework
Figure 6 – Example DSM matrix before and after clustering analysis
Figure 7 – The DSM-DMM Periodic Table
Figure 8 – DMM mapping customer requirements with product specifications
Figure 9 – Multiple-Domain Mapping across process, product and organizational matrices
Figure 10 – Engineering System Multiple-Domain Matrix
Figure 11 – ES-MDM Domain Interactions Summary
Figure 12 – Example 3U CubeSat frame courtesy of FLARE PDR
Figure 13 – Proposed FLARE Design CAD diagram, courtesy of FLARE Team
Figure 14 – FLARE CONOPS, PDR 2017
Figure 15 – FLARE Structural Overview, PDR 2017
Figure 16 – Object Process Diagram for FLARE Satellite
Figure 17 – Qualitative Knowledge Construction Example
Figure 18 – System Driver Network Diagram
Figure 19 – Stakeholder Value Network Diagram
Figure 20 – Functional Decomposition Diagram
Figure 21 – Formal Decomposition Diagram
Figure 22 – Process Architecture Diagram
Figure 23 – Example of QKC Coding in FLARE Technical Document
Figure 24 – FLARE ES-MDM
Figure 25 – FLARE ES-MDM Network Graph
Figure 26 – Example Subgraph
Figure 27 – Example Network Graph
Figure 28 – Example Change Graph
Figure 29 – Graph of Objects by Desired Flexibility Score
Figure 30 – DoDAF 2.0 viewpoints overlaid in an ES-MDM
**List of Tables**

Table 1 – SysML and OPM Comparison

Table 2 – CubeSat Standard Dimensions

Table 3 – Change Scenario Probability

Table 4 – Change Initiator/Relationship

Table 5 – Example Component Expected Expense for Change Scenario 1, Lasercom Payload Initiate, Power Relationship

Table 6 – Object Betweenness Comparison

Table 7 – Object Centrality Comparison

Table 8 – External Stakeholder Betweenness and Centrality Comparison

Table 9 – Internal Stakeholder Betweenness and Centrality Comparison
Acronyms

3U – Three Unit (10 cm x 30 cm)
ADCS – Attitude Determination and Control System
C&DH – Command and Data Handling
C4ISR – Command, Control, Communications and Computers Intelligence Surveillance and Reconnaissance
CAM – Cambridge Advanced Modeler
CEE – Component Expected Expenses
CIRT – Change Initiator Affects Relationship Type
CONOPS – Concept of the Operation
CS – Component Switch Cost
CSD – Corporation Canisterized Satellite Dispenser
DFS – Desired Flexibility Score
DSM – Design Structure Matrix
DMM – Domain Mapping Matrix
DoDAF – Department of Defense Architecture Framework
EOL – End of Life
ES-MDM – Engineering Systems Multiple-Domain Matrix
FDO – Flexible Design Opportunity
FLARE – Free-space Lasercom and Radiation Experiment
FPGA – Field Programmable Gate Array
FSO – Free-space Optical
INCOSE – International Council on Systems Engineering
ISO – International Organization for Standardization
ISS – International Space Station
IT – Information Technology
LEO – Low Earth Orbit
MATLAB – Matrix Laboratory
MBSE – Model-Based Systems Engineering
MDM – Multiple-Domain Matrix
MIT – Massachusetts Institute of Technology
MLI – Multilayer Insulation
OMG – Object Management Group
OPCAT – Object Process Case Tool
OPD – Object Process Diagram
OPL – Object Process Language
OPM – Object Process Methodology
$P_c$ – Probability of Change Propagation
$P_{cs}$ – Change Scenario Probability
$P_{CIRT}$ – Probability Change Initiator Affects Relationship Type
P-POD – Poly Picosatellite Orbital Deployer
PDR – Preliminary Design Review
QFD – Quality Function Deployment
QKC – Qualitative Knowledge Construction
SE – Systems Engineering
STAR – Space Telecommunication, Astronomy, and Radiation
SVN – Stakeholder Value Network
SysML – Systems Modeling Language
UML – Unified Modeling Language
UNP – University Nanosat Program
Chapter 1: Introduction

1.1 Background and Motivation

The field of systems engineering is constantly evolving. New systems are being designed with increasingly complex interdependencies and interfaces. Because of this complexity, new methodologies and frameworks are being introduced to model system behavior and to better understand emergent properties. One such framework, the design structure matrix, or DSM, uses a $N \times N$ matrix to highlight design interactions and provide insight into architectural design decisions. Further iterating on the DSM’s single domain insights, new frameworks have evolved from the DSM methodology to reflect more holistic approaches to system analysis, deriving additional insights and interactions over multiple domains.

The application of these frameworks to complex engineering systems can be observed in almost all fields and industries; however, the focus of this research is solely on the application of these frameworks toward complex space systems, more specifically nanosatellites. Nanosatellites are resource constrained systems, typically weighing less than 5 kg, which must operate for extended periods of time in austere conditions, without maintenance or repairs. These systems are an amalgamation of commercial off the shelf components, science payloads, and designed components engineered for to satisfy interface requirements. Due to cost and weight constraints, the interactions among subsystems within the nanosatellite represents a complex tradespace where achieving pareto efficiency is extremely difficult.

In such a resource constrained system, the ability to identify feasible architectural options is seemingly limited. Tradeoffs between performance and weight are common, where costs are the driving factor. In order for the system design to maximize value, it must achieve its primary value function objective while meeting all project and design requirements. This is difficult to achieve with space systems, since these system designs are often new and untested, relying on innovative technologies that must perform without human intervention on orbit. The perceived limitation of architectural decisions in space systems has led to project teams limiting the scope of design and tradespace analyses.

There has been an increased interest in the design and deployment of small satellites due to their improving performance and relatively low costs. Improvements to solar cell efficiencies, energy storage, data processing, and micro-thruster efficiency have created a new competitive market for small satellite deployment. This cutting-edge area of interest has launched multiple
programs across the globe, energizing governments, universities, and private companies to research and develop new technologies for small satellites. (Macdonald and Lowe 2014) More than ever there is a need for a methodology that can abstract detail and reduce complexity of inherently complex systems and provide valuable insights into key design nodes and architectural interdependencies.

The motivation for this thesis is to support the field of systems engineering in identifying and evaluating frameworks and methodologies for the design of complex systems. Often engineering design decisions are limited in scope and analysis, excluding the socio-technical aspects of an engineering system. Specifically, project teams often overlook the complex nature of socio-technical systems with regards to the social and environmental domains, resulting in potential missed opportunities. Using a highly resource constrained system case, such as a nanosatellite, this thesis aims to demonstrate the strengths and weaknesses of one particular systems engineering framework, known as an engineering systems multiple-domain matrix, to pave the foundation for further research and development in this area of study.

1.2 Research Scope

The research analyzes a nanosatellite, referred to as a CubeSat, using the Engineering System Multiple-Domain Matrix (ES-MDM) framework. The project analyzed, known as the Free-space Lasercom and Radiation Experiment or FLARE project, is a Massachusetts Institute of Technology (MIT) program that is sponsored by the University Nanosatellite Program (UNP). The FLARE project, which is discussed in Chapter 4, is a constellation of two 3U CubeSats that will test the feasibility of small scale optical communications between nanosatellites. The FLARE project is still in its nascent design stages, with the preliminary design review scheduled for March 2017. This timeline provides an ideal candidate for evaluation, since the system design architecture is still fluid.

Since the FLARE project design is still under development, the research data collected reflects an intermediate stage of FLARE’s development, and may not reflect the final design decisions of the project. For the purpose of this thesis, the data collected at the preliminary design review will be the assumed values regardless if changes are made to the actual project. Furthermore, the research focus extends beyond the pure technical domain of the system and
explores interactions between the system, stakeholders, and the project team. The objective of this research is to analyze the effectiveness of the ES-MDM and not the FLARE project.

The scope of research into the FLARE system is focused on deriving elements and relationships of that system for the purpose of constructing an ES-MDM and evaluating the ES-MDM methodology's ability to provide value to a resource constrained program. The strengths and weaknesses of the methodology, its ability to identify critical design nodes, and ability to provide information valuable to decision makers is discussed and evaluated, with insights identified for potential future work. Specifically, the thesis looks to identify insights providing value toward design option flexibility derived from the ES-MDM.

1.3 Research Objectives

The intent of the research is to investigate the design flexibility opportunities in a highly resource constrained system. The primary objective of the analysis is to provide a structured systems design approach for nanosatellite development that encompasses the entire system holistically. The second objective is to analyze the interactions and interdependencies within a highly-constrained system and determine key design nodes that are critical to system flexibility. The third objective is to evaluate the ability of the ES-MDM methodology to analyze a highly-constrained system. The fourth objective of thesis is to provide recommendations for future work to improve the ES-MDM framework and the systems engineering field.

1.4 Organization of Thesis

This thesis is organized into six chapters. The first chapter, introduction, sets the framework for the following chapters by introducing key concepts and ideas regarding the formulation of systems engineering methodologies, the need to understand key interactions of a complex system, and the difficult in identifying flexibility in design options of a resource constrained systems, such as a nanosatellite. Additionally, the first chapter outlines the scope of the thesis and the research objectives.

The second chapter provides the background and context for the thesis. The chapter sets the foundation by building an understanding of systems engineering and key definitions such as system value and flexibility. The chapter continues to identify key systems engineering methodologies and frameworks that are widely used within the field. The second chapter ends
with the introduction of the ES-MDM, an overview of its cross-domain approach to an engineering system and the results that can be derived from the ES-MDM methodology.

The third chapter provides the background to nanosatellite development and the CubeSat program and the challenges associated with the development of a resource constrained complex system.

The fourth chapter provides the background for the FLARE project, the project’s scope, and the project’s research objectives. Additionally, it introduces some of the challenges currently faced by the project team and the motivation for applying the ES-MDM methodology to the FLARE project.

The fifth chapter outlines the construction of the ES-MDM for the FLARE project, to include research and detail abstraction, step by step ES-MDM development, interim analysis, and the final analysis and drawn conclusions.

The sixth chapter discusses the results of the analysis, the potential impacts with respect to CubeSat design and the use of the ES-MDM methodology, as well as the potential for future work in the field of study.
Chapter 2: Literature Review

The focus of the literature review aims to provide an understanding of the field of systems engineering, the development of model-based systems engineering (MBSE) frameworks, and the development of the ES-MDM framework. This chapter briefly summarizes the key concepts and definitions of systems engineering and provides structure to enhance the review and discussion of incumbent MBSE frameworks. Each of the frameworks discussed are commonly applied to engineering systems and provide context for the creation and evaluation of the ES-MDM framework. Finally, the literature review thoroughly examines the development and application of the ES-MDM framework.

2.1 Introduction to Engineering Systems

Advances in technology over the past century have created a variety of new systems that are integrated into our daily lives. These systems over time have increased in complexity to improve functionality and maximize value delivery to the user. Ranging from handheld electronic devices to large scale transportation networks, these systems rely upon a network of mechanical, electronic, and information interfaces to achieve the system’s primary value function. Furthermore, modern systems are shaped by societal requirements that include, but are not limited to, cost and scope requirements, safety requirements, disposal requirements, and a need for information connectivity across system interfaces. Due to this trend of evolving complex systems, an alternative approach to designing systems is needed.

An engineering system can be defined as “a class of systems characterized by a high degree of technical complexity, social intricacy, and elaborate processes, aimed at fulfilling important functions in society.” (MIT Engineering Systems Division Strategic Report, 2011) These systems are more than just complicated technical systems, but also are highly integrated social systems incorporating stakeholders, project teams, operations team, and end users. Due to these interdependencies, these systems are often referred to as socio-technical systems, but that label does not provide a complete understanding the dynamic nature of engineering systems.

2.1.1 What is Systems Engineering?

Systems Engineering is an interdisciplinary engineering field that addresses the increasing complexity of engineering systems. A system is “a set of interacting components – technical artifacts – with well-defined behavior and a well-defined function or purpose...” (de
Engineering Systems, are more complex systems due to their technical and social interdependencies as well as their dynamic nature. Since these systems often blur the lines between traditional engineering disciplines, a new paradigm is needed. Systems Engineering “is a unique perspective on reality – a perspective that sharpens our awareness of wholes and how the parts within those wholes interrelate.” (INCOSE 2014) Systems Engineers apply systems thinking to abstract information from complex systems, model behavior, identify interrelationships, identify casual loops, and understand the system holistically to achieve system objectives, meet stakeholder requirements, while minimizing undesirable system emergence.

At its core, systems engineering is a design perspective. Systems Engineers must understand the context in which a system will operate, how the system will behave, and how it will be managed. (INCOSE 2014) A fundamental principle of systems engineering is a holistic systems approach – an understanding that the system is greater than the sum of its parts. Modern systems are typically nonlinear – that the change in the output of a system does not necessarily correlate to the scale of the change of the input to a system. The nonlinearity of systems has created new fields of research into modeling and predicting the emergent behavior of complex systems. Systems Engineering encompasses social aspects of a system, such as understanding a systems use context or identifying stakeholder needs, and technical aspects of a system, such as identifying technical requirements and interfaces, in order to identify emergent behavioral issues generated from dynamic socio-technical interdependencies.

2.1.2 Why do we need Systems Engineering?

Systems Engineering seeks to optimize engineering systems to meet cost, schedule and scope requirements. Often tradeoffs between performance characteristics of an engineering system are difficult to quantify and optimize. Further exacerbating tradeoff ambiguity are emergent properties of a system that can create unexpected or unpredictable behavior. This nonlinearity of engineering systems requires new methods for analysis and modeling to reduce ambiguity and make informed decisions. Systems Engineering provides that methodology enabling engineers to identify and solve complex issues in dynamic nonlinear systems.

Furthermore, systems engineering includes not just technical analysis, but also managerial processes involved in system design and implementation. Systems Engineering focuses on the whole life cycle of the system, following a system from idea generation to the customer acceptance or transfer to operations. Since systems engineering provides a
methodology for the entire life cycle process, many management functions are included, such as: stakeholder management, requirements management, cost management, scope management, and schedule management. Systems Engineers are not solely focused on designing the best technical system, but are also integrating key management processes to create the best technical system that meets stakeholder objectives and requirements, while staying on schedule and on budget.

Finally, recent research suggests that there is a correlation between project success and systems engineering activities, especially for complex engineering systems. “The quantifiable correlations between SE effort and program success all evidence a ‘bathtub’ behavior, in which there is a clear optimum value of SE effort in each relationship… the optimum amount of effort for a median program is 14.4% of the total program cost.” (Honour 2013) This analysis indicates that projects that budget approximately 14% of total cost toward systems engineering activities are less likely to experience cost or schedule overruns. Additional effort that exceeds 15% indicates minimal additional cost or schedule benefits for the project. Systems Engineering provides not just socio-technical analysis and system behavior modeling, but is strongly correlated with project success and financial benefits. See figure 1 for visualization.

![Figure 1: Systems Engineering Effort vs. Actual/Planned Cost Ratio (Honour 2013)](image)

2.1.3 System value

Value is benefit at cost. (Crawley, Cameron, and Selva 2016) System value is the net utility or worth a system provides for users or stakeholders, with cost considered. This simple
definition of system value understates its importance with regards to a system’s design. The value a system provides is delivered externally, across that system’s boundary, by its primary value function. Although systems typically have multiple functions operating internally or across system boundaries, the delivery of the primary value function is in essence the purpose of the system and determines whether or not a system will be a success or failure.

The primary value function follows the value pathway through the system; the processes, operands, and instruments within a system that directly support the primary value function. A function is “is the activity, operation, or transformation that causes or contributes to performance” (Crawley, Cameron, and Selva 2016) and is typically abstracted into two components: an operand and a process. A process is an action, and in its fundamental definition can be classified into: transform, transport, store, exchange, and control. (de Weck, Roos, and Magee 2011) Operands are objects that the processes within a function are acting upon. Specifically, the processes change the state of an operand to achieve a functional purpose and create value. Operands can be abstracted into three basic classifications for technical systems: matter, energy and information. (de Weck, Roos, and Magee 2011) Although there are other operand classifications that have evolved from increased socio-technical interdependencies, these three classifications will suffice for this discussion. Finally, instruments are objects that execute the process, but are not always present. For instance, in the function “human walking” the human is both the instrument and operand.

The principle of system value is straightforward in simple technical systems; however, as the technical and social interdependencies increase in complexity, it is possible to lose sight of value pathway. Supporting secondary functions can be necessary to support the primary value pathway and value creation, but these functions can exist independently to support additional system functionality or features. Many modern systems today deliver value through their primary value functions but add features to the system to remain competitive within the market. As features or secondary functions evolve in a complex system, it is possible for the value pathway and the purpose of the systems primary value function to become ill-defined.

An example of a system whose value pathway is difficult to define due to system evolution would be the cellular phone. When the cellular phone was introduced, the value function was simple and straightforward: human communicates. The speaker, microphone, radio and antenna were design to maximize the quality of the signal transmitted and received and
translate those signals into sound understandable by the user. However, as cellphones evolved with the introduction of cameras, text messaging, web browsing, touch screens, Wi-Fi and Bluetooth interfacing, the primary value pathway of the cell phone has shifted. Where the microphone, speaker, radio and antenna were part of the primary value pathway, these instruments arguably now support secondary system functionality. Although human communicates may still be a viable primary value function for the system, the definition of the communication process and the value pathway to achieve the function have been blurred. This example illustrates the importance of understanding the primary value function and value pathway of a system, and thereby its purpose, when designing a complex socio-technical system.

2.1.4 Life-cycle properties

Systems Engineering life-cycle properties, or *ilities*, are “desired properties of systems such as flexibility or maintainability (usually but not always ending in “ility”), that often manifest themselves after a system has been put to its initial use.” (de Weck, Roos, and Magee 2011) The ilities of a system have evolved over the past 150 years, focusing initially on resilience and flexibility attributes of a system and growing to include safety, reliability, quality, maintainability and sustainability to list a few. The evaluation of system behavior or performance with respect to life-cycle properties is a constant endeavor. As systems become more complex and interconnected, users demand performance in many of these ilities.

As these complex systems evolve, the constraints placed upon the system increase. The size, weight, and energy constraints on cell phones are a constant tradeoff with functionality and performance. These factors greatly affect the ability of system designers to successfully incorporate life-cycle property attributes into a system. Flexibility, specifically is an interesting system property, that represents the relative ease a system changes to embrace uncertainty or the ability of a system to reconfigure to perform multiple functions. (de Weck et al., 2011) Alternatively, “‘flexibility’, defined as the measure of ease for which a system can change over time, is an emergent property of the system and can only be understood by examining the social and technical domains.” (Bartolomei, 2007)

The essence of flexibility within a system revolves around the ability of that system to embrace uncertainty. Uncertainty can take a variety of forms. Uncertainty can be internal to a system, such as unexpected emergent system behavior or external to a system, such as unexpected influence by system drivers or stakeholders. Regardless of the source, the ability of a
system designer to anticipate uncertainty and incorporate flexibility in a system is challenging prospect and a significant driver of systems engineering research.

2.2 Current Modeling Framework Overview

As systems become more complex and the interactions between system functionality and the environment become intertwined, a need for a formalized language to describe and analyze systems has emerged. Models are “simplifications of reality... [used to] identify complexity, increase our understanding and communicate in an unambiguous (or as unambiguous-as-possible) manner.” (Holt and Perry 2013) Model-based systems engineering (MBSE) frameworks use a generic language to illustrate system structure and flow of operations of a system to enhance design and development of complex systems. (de Weck, Roos, and Magee 2011) These models, typically constructed digitally, enable system designers to understand and predict system behavior, abstract detail, reproduce digital copies and modify system characteristics in order to reduce design error discovery during the development process. (Micouin 2014) Although there are many industry standard modeling frameworks that are widely used and accepted, such as the Unified Modeling Language (UML) and the Systems Modeling Language (SysML), the field of MBSE is constantly evolving and introducing new methodologies and approaches.

2.2.1 Department of Defense Architecture Framework

The US Department of Defense Architecture Framework (DoDAF) is comprehensive modeling framework that enables the development of architectures for acquisition decision making and information sharing purposes. (Department of Defense Deputy Chief Information Officer 2011) The DoDAF evolved from Command, Control, Communications and Computers Intelligence Surveillance and Reconnaissance (C4ISR) framework in 2003, with the goal of providing a methodology for analysis, a universal language for better information dissemination, and improved integration with emerging information technologies. (Giachetti 2015) However, where the DoDAF prescribes data and terminology requirements for each modeling view, the actual DoDAF framework does not specify modeling languages or architectural development tools. This enables the DoDAF to have sufficient freedom for system designers to apply modeling principles to a variety of systems and organizations.
The current version of the DoDAF, DoDAF 2.0, consists of six core processes: Capabilities Integration and Development, Planning Programming Budgeting and Execution, Acquisition Systems, Systems Engineering, Operations Planning, and Capabilities Portfolio Management. The fourth core process, systems engineering, has been updated in the most recent version of the DoDAF to include models or what the DoDAF refers to as viewpoints for capability, data and information, project, services, standards, and systems. These systems engineering viewpoints offer a broad summary of information for the business and operational needs of the DOD while still enabling project and system managers to access more narrowly focused information regarding that specific viewpoint. (Department of Defense Deputy Chief Information Officer 2011) The ability of the framework to abstract system information and “zoom” in to enhance granularity on a specific topic enhances the shared understanding of the organization and reduces the need for recreating models for executive level and system level discussions.

Figure 2: DoDAF Viewpoints (Jeffries, Hayden, and Dafoe 2013)
Although the DoDAF offers flexibility for system designers and architects as well as requirements to ensure information completeness within its viewpoints, the DoDAF does have several flaws. First, the framework is extremely comprehensive, which is a strength in itself, but can be cumbersome to manage and difficult to interface with organizations not familiar with the framework. Secondly, the lack of specification for modeling languages or tools does allow flexibility for managers and designers; however, this same flexibility prevents standardization, model consistency, and IT interfacing across models and viewpoints. Finally, an evaluation of the DoDAF’s support of systems engineering by Ronald E. Giachetti infers that the viewpoints generated by the DoDAF framework are the result of separate systems analysis and are not the medium used in the MBSE process. He concludes that the DoDAF models “are not being used for systems engineering analysis and design activities,” which he links to the need for viewpoints that provide systems engineers with more useful information to compensate for the inherent oversights in the systems engineering process. (Giachetti 2015)

2.2.2 Unified Modeling Language (UML)

Unlike DoDAF, the UML framework provides not only the structure, but the tools for modeling systems. The fundamentals of UML originated in the 1960s from the need for object-oriented software programming. Difficulties communicating requirements to stakeholders and ambiguity caused through the use of natural language led system designers to develop new modeling languages. (Seidl et al. 2012) By the mid-1980s, compatibility issues amongst modeling languages led to a need for standardization and the development of an encompassing modeling language. In 1997, UML 1.0 was released by the Object Management Group (OMG) with the goal to standardize object-oriented software development language, “establish relationships between modeling concepts and executable program code... and create a modeling language that can be processed by machines and read by human beings.” (Seidl et al. 2012)

UML was one of the first modeling languages to become broadly accepted in industry, and multiple modeling languages developed since 1997 have their roots embedded in UML. UML consists of 13 diagrams or views that are categorized as either structural or behavioral in nature. Outlined in figure 3, these diagrams demonstrate the effort to create a universal modeling language able to meet the needs of the system designer while standardizing the information and format required for each diagram. However, UML does prescribe the use of these diagrams to define the system, but rather allows the system designer flexibility in execution.
Although UML was developed to standardize modeling languages for software development, the framework provides the foundation for future system object-oriented modeling languages and a method to abstract relationships graphically. The foundation of UML outlines the structure and behavior of a system, which can translate to physical system design outside of the digital domain. The use-case diagram depicts the function actors accomplish within the system the purpose they provided within a defined system boundary. The sequence diagram outlines the process actors within a system follow and attempts to depict those complex interactions. These fundamental concepts standardized in UML set the foundation for the development of future system oriented modeling frameworks.

2.2.3 System Modeling Language (SysML)

SysML was developed in 2003 in a joint effort with OMG and the International Council on Systems Engineering (INCOSE) adapting UML’s framework to extend outside of the software domain and address frameworks for system requirements and performance. (Crawley, Cameron, and Selva 2016) This adapted framework focused on creating a new standard for systems engineering modeling language, while maintaining flexibility and compatibility that existed already with UML. Where UML originally consisted of 13 diagrams or views, SysML
has been modified to describe physical systems in nine diagrams. As depicted in figure 4, SysML still decomposes system diagrams into structural diagrams and behavioral diagrams, but extends the framework with two new categories, requirement diagrams and parametric diagrams.

![Figure 4: SysML Diagram Tree (OMG 2017)](image)

The introduction of the requirement diagram captures the requirement hierarchical structure and depicts “client-supplier” dependencies within that structure. This allows the system requirements to be graphically depicted alongside system elements, enabling behavioral analysis and the creation of validation scenarios for each requirement. (OMG 2017) Further system decomposition into the subsystem level would also allow derived requirements to be analyzed in part with the system elements. The requirement diagram allows system designers “to relate a requirement to a model element that satisfies or verifies the requirement… [and] provides a bridge between the typical requirements management tools and the system models.” (OMG 2017)

Parametric diagrams are another addition to the SysML framework and represent constraints placed upon system property values. This allows systems engineers to analyze system performance against constraints for verification and validation processes. (Holt and Perry 2013) Similar to system requirements, the depiction of system constraints on system elements allows for in-depth behavioral analysis on emergent system properties such as performance and reliability. Using this methodology, systems engineers can monitor that cascade effect of
constraints at the system level upon subsystem design. Like the requirements diagram, the parametric diagram bridges the gap between system constraints and system behavior.

SysML represents a joint effort between INCOSE and OMG to standardize a technical modeling language that reduces complexity, increases understanding, and reduces ambiguity for complex system design. By decomposing physical systems into structural and behavioral viewpoints, systems engineers can analyze and communicate emergent system behavior, understand system requirements and constraints, and depict system interactions. The benefits of SysML are the it designed for flexibility and consistency, and has the backing of two international organizations as an industry standard for model-based systems engineering. Criticisms of SysML reference unclear requirement/behavior relationships, no representation of time, and overly complex or unclear semantics for viewers unfamiliar with the modeling language. (Hampson 2015) (Cole, Delp, and Donahue 2010)

2.2.3 Object Process Methodology (OPM)

Object Process Methodology is an alternative modeling framework to SysML developed by Dov Dori in 2002. Where SysML and DoDAF use multiple diagrams or viewpoints to describe various system attributes, OPM “combines simple graphics with natural language sentences to express the function, structure, and behavior of systems in an integrated, single model.” (Dori, 2002) OPM consists of a set of object-process diagrams (OPD) that visually depict a systems design and the corresponding object-process language (OPL) which translates the diagram using natural language. At the core of OPM is three entities that describe a system: objects, processes, and states. “Objects exist, and processes transform the objects by generating, consuming, or affecting them. States are used to describe objects, and are not stand-alone things.” (Dori, 2002) Objects can also assist processes in the form of instruments. Functions are defined by a process affecting an object, in this case an operand, and a change of state. Entities are connected by two types of links: structural links and procedural links. Using these basic building blocks, OPM is able to model complex systems.
Figure A.7. SD1.1.2.1 of the ATM system

Customer handles Password Keying and Reading.
Password Requesting requires Screen.
Loop can be uninitialized or initialized.
Message Displaying occurs if Loop is initialized, and if Number Of Trials is not greater than Max Trials.
Message Displaying requires CPU.
Message Displaying yields Displayed Message.
Reading requires Screen.
Reading invokes Password Keying.

Figure 5: Example of an OPD with corresponding OPL using the OPM framework (Dori, 2002)

Figure 5 depicts an example object process diagram for an ATM with corresponding object process language. Consistent with OPM, objects are drawn as rectangles, processes as ovals, and links as lines. In this example, the ATM system is decomposed into three objects: keyboard, CPU, and screen. Each of these objects are linked to corresponding processes. The process of password requesting is further decomposed, a mechanism called zooming in OPM, into a message displaying process which produces a message displayed object, and a reading process which is required for the customer to read the message and key the password. Zooming is a mechanism OPM uses to control model complexity, by abstracting into a higher level object or process. Beneath the example OPD is the OPL, which textually describes what is visually depicted using natural language. Although the OPL is not required to complete an OPD, it is useful to clearly convey meaning, especially when designing the OPD.

Like SysML, OPM is designed to interface with software tools to assist digital system modeling. However, unlike SysML, which has several open source options for software tools,
OPM primarily uses OPCAT software. OPCAT is a powerful tool that enables a system to be captured in a single diagram, referred to as a systems diagram. Additional granularity can be achieved on the systems diagram in OPCAT through zooming, which decomposes high level objects and processes into subordinate entities. Furthermore, OPCAT features model animation, which can assist with understanding and troubleshooting control flows for complex systems. (Perelman 2011) The OPM modeling language is approachable, holistic, and understandable framework that captures structure and behavioral characteristics of systems on a single diagram, possesses mechanisms to manage complexity of diagrams, and was formally specified by ISO in December, 2015. (“Automation Systems and Integration - Object-Process Methodology” 2015) Table 1 illustrates the fundamental differences between SysML and OPM.
Table 1: SysML and OPM Comparison (D. Dori 2016)

<table>
<thead>
<tr>
<th>Feature</th>
<th>SysML</th>
<th>OPM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Theoretical foundation</td>
<td>UML: Object-Oriented paradigm</td>
<td>Minimal universal ontology: Object-Process Theorem</td>
</tr>
<tr>
<td>Standard documentation number of pages</td>
<td>~1670=700 (UML Infrastructure) + 700 (UML Superstructure) + 270 (OMG SysML)</td>
<td>~180=100 (ISO 19450 main standard) + 80 (appendices)</td>
</tr>
<tr>
<td>Standardization body</td>
<td>OMG</td>
<td>ISO</td>
</tr>
<tr>
<td>Number of diagram kinds</td>
<td>9</td>
<td>1</td>
</tr>
<tr>
<td>Top-level concept</td>
<td>Block (UML object class)</td>
<td>Thing (object or process)</td>
</tr>
<tr>
<td>Complexity management guiding principle</td>
<td>Aspect-based decomposition</td>
<td>Detail-level-based decomposition</td>
</tr>
<tr>
<td>Hierarchical decomposition</td>
<td>In some diagram kinds</td>
<td>Yes</td>
</tr>
<tr>
<td>Number of symbols</td>
<td>~120</td>
<td>~20</td>
</tr>
<tr>
<td>Graphic modality</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Textual modality</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Built-in physical-informatical distinction</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Systemic-environmental distinction</td>
<td>Partial (using boundaries)</td>
<td>Yes</td>
</tr>
<tr>
<td>Logical relations (OR, XOR, AND)</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Probability modeling</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Execution, animated simulation, validation and verification capability</td>
<td>Partial (in some tools for some diagram kinds)</td>
<td>Yes</td>
</tr>
<tr>
<td>Tool availability</td>
<td>Many, some free</td>
<td>Currently one free (OPCAT) from <a href="http://esml.iem.technion.ac.il/">http://esml.iem.technion.ac.il/</a> Cloud-based tool under development</td>
</tr>
</tbody>
</table>

2.2.4 Design Structure Matrix (DSM)

Design Structure Matrices are a “network modeling tool used to represent the elements comprising a system and their interaction, thereby highlighting the system’s architecture (or designed structure). (Eppinger and Browning 2012) The DSM is a powerful modeling tool that can visually analyze the dependencies of a system in a N x N matrix. There are four recognized DSM types: component-based, team-based, activity-based, and parameter-based. (Jason Bartolomei et al. 2006) Through multiple matrices or by color coding system attributes and dependencies on a single matrix, DSMs are able to capture the physical and non-physical flows and the dependency strength of a system. Additionally, DSMs can be used in project management to analyze team structure, system processes, system parameters, system
specifications and requirements, and project scheduling dependencies. (Danilovic and Browning 2007)

Figure 6: Examples DSM matrix before and after clustering analysis (Lindemann 2016)

In figure 6, two simple DSMs are presented with seven system elements. On the left, the DSM is in its original form with no analysis conducted regarding the interactions of system dependencies. The cells along the diagonal of each DSM represent the system elements along the rows, and are either labeled with the system element name or left empty due to the inability of system elements to be dependent upon themselves. In typical DSM convention, a cell can be considered to have inputs originating from the left and right, and outputs leaving the cell above and below. For example, system element C would have elements A and E as inputs, with elements A, B, F, and G as outputs. That would indicate that C is dependent on A and A is dependent upon C, indicating that these elements are coupled. There are other DSM conventions that have the inputs for a system element on the column and outputs on the row.

The second DSM in figure 6 has four darkened areas within the matrix. These areas represent the partitioning of the matrix. DSM models “can be partitioned or rearranged using a variety of analytical methods, the most common of which are clustering and sequencing.” (Eppinger and Browning 2012) Clustering involves rearranging system elements to combine system elements that are interdependent into subsystems or substructures. Clustering is typically used for structural DSM, where the system element interactions are mostly symmetrical along the diagonal of the matrix. Sequencing involves an algorithm that groups system elements with regards to the directional or temporal interactions of the system elements. Sequencing is primarily used with process DSMs, which are likely not symmetrical along the diagonal of the
matrix. (Eppinger and Browning 2012) Clustering and sequencing algorithms typically seek to minimalize feedback loops and identify coupled elements.

DSM construction and analysis can be done manually using most spreadsheet software or through a variety of software tools, such as Lattix or the Cambridge Advanced Modeler (CAM), which are DSM architectural analysis tools. Other analysis tools can be found in the form of Microsoft Excel macros that can be installed and used to analyze and partition DSMs built in Excel. Recent work by MIT Alumnus Sydney Do introduced MATLAB code that could import an OPD created in OPCAT, transfer the OPD into a network diagram, and then convert that information into a DSM for use in Lattix. (Do 2016) This type of flexibility and the capacity to interface with multiple software tools and system modeling frameworks illustrates the DSM’s strength as a tool to concisely visualize and analyze a system and its properties. However, have several limitations, such as the inability to model a system’s life-cycle dynamics over time, inability to measure environmental interactions, and limitations associated with mapping interactions across domains. (Bartolomei, 2007)

2.2.6 Domain Mapping Matrix (DMM)

The domain mapping matrix (DMM) is a DSM-based model that creates typically a non-square matrix by mapping the domain of one DSM or DMM to the domain of another. A domain is defined as the “realm of the elements comprising a DSM model of a system.” (Eppinger and Browning 2012) Domains are commonly defined in five areas for a project: the product system, the process system, the organization system, the tool system, and the goal system. (Danilovic and Browning 2007) Figure 7 outlines the relationships between each domain DSM and the resulting DMMs used to analyze domain interactions. Like DSMs, DMMs can illustrate the existence of a dependency or indicate strength or type of relationship between system elements. Unlike DSMs, however, DMMs do not necessarily have marks or unused cells along the diagonal of the matrix, largely due to their rectangular shape.
Figure 8 provides an example DMM where a customer requirements DSM is combined with a product specification DSM to map the relationships between specifications and requirements. In this example, the mapped relationships have already been clustered with some conclusions drawn. Based upon those results, the project manager can determine how a specific cluster of customer requirements are satisfied by certain product specifications, how teams should be organized during product design and investigate why two product specifications are not linked to customer requirements. Other common uses in industry for the DMM are function to component mapping, product goals to process mapping, quality function deployment (QFD) analysis, which is similar to the requirement to specification mapping in figure 8, and process to organization mapping. (Browning 2016) The use of a DMM model to analyze relationships across domains can capture the dynamics of product development, improve system interaction understanding, and decrease ambiguity for decision makers.
2.2.6 Multiple-Domain Matrix (MDM)

The multiple-domain matrix (MDM) model is two or more DSM or DMM models in different domains that are represented simultaneously. (Eppinger and Browning 2012) MDMs map these interconnected domains across multiple matrices to analyze relationships across domain interfaces and infer element relationships. MDM models can be used to verify DSMs and DMMs that predict system behavior, track change propagation, and analyze engineering design. (Browning 2016) MDM analysis can result in better design decisions, reduced architectural risks, and better product to organization alignment. Figure 9 illustrates an example of a MDM, where DSMs and DMMs interface for analysis. Although it is not always possible to create and analyze a MDM in such a concise structure, MDMs can still provide interesting results in a variety of applications regardless of visual complexity.
Figure 9 demonstrates an example of multiple-domain mapping across matrices for use in project management for software design. By comparing the process architecture with the product architecture, the MDM captures actor interaction with each subsystem during each phase of product development. Using this methodology for cross-mapping, managers are able to predict potential interactions between personnel and product design processes and model conflicts associated with design or process changes and technical personnel. (Sosa 2008) The MDM framework is a promising systems engineering model that enables systems engineers and engineering managers to analyze large quantities of interconnected data, draw conclusions and make informed decisions. Similar limitations exist to MDMs as with DSMs and DMMs, such as the inability to model a system’s dynamics over time. An example of a MDM system engineering framework is the engineering systems multiple-domain matrix.

2.3 ES-MDM

Within this chapter, several MBSE frameworks have been discussed and evaluated to provide context for the development of the engineering systems multiple-domain matrix. DoDAM is a comprehensive systems architecture framework that does not prescribe analysis tools or
methodology, and therefore lacks internal consistency. UML and SysML are object oriented modeling languages that are flexible, comprehensive, and are considered industry standards but are criticized for lack of analytical depth in some systems engineering domains and the inability to display complete system information on a single viewpoint. OPM is emerging as another standard for object oriented modeling languages and is noted for its approachability and conceptual simplicity; however, OPM fails to fully capture some system level interactions and can lack completeness as a modeling language. DSMs and DMMs are excellent tools for modeling structural, process, and domain interactions, but lack the ability to model a system over time and struggle to illustrate interactions across system boundaries. MDMs expand the role and capabilities of DSM and DMM mapping, but MDMs are still unable to capture a system’s life-cycle across domains. As a response to limitations of existing MBSE frameworks, Jason Bartolomei in his 2007 MIT doctoral thesis established the foundations for the engineering systems multiple-domain matrix. (Bartolomei, 2007)

2.3.1 ES-MDM Background

To address the limitations of DSMs and DMMs, Bartolomei developed the engineering systems multiple-domain matrix, which is a MDM that provides a holistic view of an engineering system across the five domains: social, technical, functional, process, and environmental. ES-MDMs also are designed to “represent how the graph (nodes, relations, and attributes) change over time.” (Bartolomei, 2007) Nodes are defined as classes of objects, relations are defined interactions between two nodes, and attributes are defined as parameters or characteristics for each node or relation, and can be binary, numeric, or a function. (Bartolomei, 2007) The ES-MDM is an adjacency matrix, with the row and column headers referencing classes of nodes, the diagonal representing a particular class of node of system components, and the off-diagonal cells representing relationships between components. Figure 10 depicts a visual representation of the structure of the ES-MDM methodology.
Bartolomei decomposes the classes of nodes into six categories: system drivers, stakeholders, objectives, functions, objects, and activities. These six node classes describe an engineering system across the five domains and over the engineering system’s lifecycle. By analyzing the engineering system across system boundaries and multiple domains, the ES-MDM framework attempts to capture traceability and interrelationships in a single view. (Jason Bartolomei et al. 2006) These interrelationships provide insight into a system’s important nodes, how coupled those nodes are, and what system properties those nodes generate, like flexibility, and the potential impact on the rest of the system.

2.3.2 System Drivers

System Drivers represent the exogenous elements that act on the system or are acted upon by the system within the environmental domain. However, what contrasts system driver nodes from stakeholder nodes are that the environmental factors affecting the system are executed by non-human actors. (Bartolomei, 2007) The factors are typically regulatory, legal, political,
economic, and technical constraints and enablers that affect the system. The system driver X system driver DSM interactions also can provide interesting context into how environmental factors couple and create feedback loops outside the system boundary.

2.3.3 Stakeholders

Stakeholders represent the other exogenous set of elements within the social domain that act on the system or are acted upon by the system typically across the system boundary, but not always. These elements are typically human actors and can be external to a system, such as customers, regulators, and financially responsible organizations or internal to a system, such as the engineering design and testing teams. (Bartolomei, 2007) External actors have no direct control over the elements within the system boundary, but can be affected by the system or can affect the system indirectly. Internal actors are actors that do have direct control over elements within the system boundary. The stakeholders X stakeholders DSM can provide interesting insight into the relationships and interdependencies between actors external and internal to the system. Furthermore, stakeholder DSMs can analyze organizational structure, communications flow, financial flow, and can analyze actor influence over other actors operating external or internal to the engineering system.

2.3.4 Objectives

Objectives refer to the system’s purpose or goal, which is defined by stakeholders for the engineering system. (Bartolomei, 2007) Objectives are explicitly or implicitly derived from the needs of stakeholders and are usually the result of a detailed stakeholder analysis. The result of the stakeholder analysis generates system requirements, system objectives or goals, and potential system constraints. These system objectives are modified contextually to be from the point of view of internal stakeholders, in order to provide clear requirements or goals for the design of the engineering system. The objectives X objectives DSM can provide coupling information for two objectives, which can provide information for tradespace analysis and isoperformance allocation.

2.3.5 Functions

Functions are the actions “for which a system exists,” and can be defined as “the activity, operation, or transformation that causes or contributes to performance.” (Crawley, Cameron, and Selva 2016) Like objectives, the functional elements exist in the functional domain. Functions must relate to an objective directly or relate to an objective by supporting another function. (J. Bartolomei et al. 2012) Functions can also have attributes that map form to function within the
system. Examples potential attributes are structural, material, informational, and energy connections. The functions X functions DSM analyzes potential hierarchical relationships or interactions between functions within a system, and can be used to locate and analyze the system’s value function(s).

2.3.6 Objects

Objects are the physical components of the system, and can be defined as a thing or artifact that is stable and has the potential for unconditional existence for a period of time. Objects can support system functions as instruments or objects can be created, modified, or consumed as operands. (Crawley, Cameron, and Selva 2016) These system components can include objects needed for the infrastructure of a system, objects to include software, needed to support system functionality, and physical entities that are used by internal stakeholders to interact with the system. (Bartolomei, 2007) The objects elements existing within the technical domain of the engineering system. The objects X objects DSM analyzes how physical components interact, the level of interaction, and clustering effects.

2.3.7 Activities

Activities exist within the process domain and represent “the process, sub-processes, procedures, tasks, and work units associated with an engineering system.” (Bartolomei, 2007) Specifically, a process is “a pattern of transformation undergone by an object. Processes generally involve the creation of, destruction of, or a change in an operand.” (Crawley, Cameron, and Selva 2016) In this way, activities act upon object components and support function components to achieve system objectives. The traceability of interdependencies between the process, technical, functional, social and environmental domains can be traced between the system driver elements and the activity elements within an engineering system. The activities X activities DSM analyzes task dependencies, coupling, and potential feedback loops within a system.

2.3.8 Existence Attributes

One of primary critiques of DSM/DMM models are that they fail to capture change propagation and system behavior over time. In order to correct this deficiency, the ES-MDM framework captures time through an “existence attribute that defines the time intervals each node or relation exists within the system.” (J. Bartolomei et al. 2012) The existence attribute can be binary in nature for a particular time interval or can be expressed a time dependent function. The
existence attribute provides each cell within the ES-MDM a temporal modifier, enabling the ES-MDM framework to track system behavior and change propagation over time.

2.3.9 Interaction Analysis

The ES-MDM framework provides a useful analysis architecture to trace system drivers to system activities and analyze the various interactions, interdependencies, and internal system hierarchies across the five engineering system domains. The interactions among ES-MDM’s element DSMs and DMMs, featured in figure 11, provide an interesting look at important nodes within an engineering system. These nodes, or hot/cold spots, are key elements within a system that exist on a spectrum, with high volatility or high risk of change propagating nodes (hot) to low volatility or low risk of change propagating nodes (cold). (Jason Bartolomei et al. 2006) The identification of key nodes within an ES-MDM allows system designers to evaluate node tradespace, identify potential real options, model engineering systems’ emergent behavior and attributes, and optimize the engineering system’s architectural design.
Chapter 3: CubeSat Background and Evolution

In 1999, the California Polytechnic State University proposed the concept of the CubeSat – a standardized specification for picosatellite or nanosatellite structures and respective interfaces to enable efficient integration into the satellite launcher and the launch vehicle. (Thakker and Shiroma 2010) Since the CubeSat specification was introduced, improvements in available technologies and reduction in the cost of commercial off the shelf equipment has resulted in the rapid expansion of picosatellite and nanosatellite programs. These programs, which are funded from a variety of private and government sources, engage universities to develop their own satellite systems to support a science payload. Although many academic institutions compete in these satellite development programs, only about 8% of all satellites are chosen for launch. (Thakker and Shiroma 2010) There have been about 600 CubeSat launches to date, with the majority of satellites originating from commercial ventures since 2014, with universities averaging 18 launches a year since 2010. (Swartwout 2017)

3.1 Introduction to CubeSat Specifications

CubeSat specifications originated as a measure to reduce costs to participate in space activities through the standardization of launch interfaces, structural measurements, and weight limitations. These standards are designed to ensure that all CubeSats are able to interface with the deploying mechanism and launch vehicle systems, and prevent damage to the launch vehicle and the primary payload. (CalPoly 2014) Although waivers can be requested, waiver acceptance must pass tests and be approved by the launch vehicle mission manager. The standardization of many of the CubeSat’s system elements creates many interesting challenges for system designers.

CubeSats are broken down by unit sizes, which are typically 1U, 2U, 3U and 6U, although other variations do exist. A “U” is a single cube unit, which measures 10mm x 10mm x 10mm. Each unit has its dimensional and weight specifications, which are outlined in Table 2. There are other designs, such as the 1.5U CubeSat, but are less common. Deviations from the existing specifications must be evaluated through the waiver request process. The purpose of the structural dimension and total weight requirements extends to the orbital deployer and the launch vehicle requirements. Aside from the 6U CubeSat specified below, CubeSats use the poly picosatellite orbital deployer, or P-POD. The P-POD is capable of carrying 3x 1U CubeSats, 1x
1U and 1x 2U CubeSats or a 3U CubeSat per deployer. However, 6U and 12U CubeSats deploy using a Planetary Systems Corporation Canisterized Satellite Dispenser (CSD) deployer. The CSD is designed to launch larger satellites with greater mass that exceed the specifications of the P-POD. Multiple deployers can be mounted to a launch vehicle. (CalPoly 2014)

Table 2: CubeSat Standard Dimensions (CalPoly 2014)

<table>
<thead>
<tr>
<th>UNIT SIZE</th>
<th>LENGTH</th>
<th>WIDTH</th>
<th>HEIGHT</th>
<th>WEIGHT</th>
</tr>
</thead>
<tbody>
<tr>
<td>1U</td>
<td>10mm</td>
<td>10mm</td>
<td>10mm</td>
<td>1.33kg</td>
</tr>
<tr>
<td>2U</td>
<td>20mm</td>
<td>10mm</td>
<td>10mm</td>
<td>2.66kg</td>
</tr>
<tr>
<td>3U</td>
<td>30mm</td>
<td>10mm</td>
<td>10mm</td>
<td>4.00kg</td>
</tr>
<tr>
<td>6U</td>
<td>30mm</td>
<td>20mm</td>
<td>10mm</td>
<td>8.00kg</td>
</tr>
</tbody>
</table>

In addition to the dimension and weight specifications, CubeSats have several safety constraints that must be incorporated into the design, such as limits to the amount of chemical energy that can be stored, hazardous material restrictions, center of gravity locations, material requirements, and testing requirements. These specifications create a narrow design space for system designers to incorporate a science payload and the required subsystems to support CubeSat operations. Although the standardization of CubeSat design specifications resulted in increased access to space activities by academic institutions, it has also imposed constraints limiting design flexibility.

3.2 CubeSat Design

The design of CubeSats and the design of larger conventional satellite systems vary in scale and features, but are comprised of similar subsystems. All satellites are designed around a payload, which can vary from observation missions, communication missions, navigation missions, or remote sensing missions. The function performed by the satellite’s science payload is typically the primary value function of the system. The remaining subsystems exist to support the primary value function or other supporting functions to enable satellite operations while on orbit. Subsystems are typically decomposed into: attitude determination and control system (ADCS), command data and handling (C&DH), position and orbit determination and control, communications, propulsion, power, thermal control, and structures.
The design of these subsystems must support the needs of the science payload and the operational needs of the satellite. For instance, the design of the power subsystem is particularly challenging in satellite systems. Power generation for low earth orbit (LEO) satellite systems is almost exclusively restricted to solar panel arrays. The generation capacity and efficiency of solar panels are dictated by size, weight and budgetary constraints. Since LEO orbits pass through the Earth's shadow approximately every 90 minutes, energy storage systems must be designed to accommodate satellite operations during periods of limited or no power generation. Finally, as satellite solar arrays age, they become less efficient, forcing system designers to design the power system for end of life (EOL) efficiencies. Exacerbating these challenges are effects of the on-orbit environment on subsystem performance and the need for robust power generation to meet dynamic energy demands.

The other subsystems within a satellite are less dynamic, but are critical to provide functionality for satellite operations. The ADCS subsystem uses a suite of sensors and actuators to monitor and adjust the orientation of the satellite. The sensors use the location of the sun, the location of the Earth's horizon against space, or the location of stars relative to an existing catalogue to determine the attitude of the satellite. The attitude data is then processed through the C&DH using an algorithm, and the actuators are engaged to adjust the attitude of the satellite by engaging thrusters, reaction wheels, or by creating a magnetic field using magnetic torque rods that interact with the Earth's magnetic field. Similar to the ADCS, the position and orbit determination control subsystem uses sensors, usually a GPS sensor for LEO, to determine the position and altitude of the satellite on orbit. The satellite then uses thrusters, if equipped, to correct navigational deviations and maintain orbit altitude.

Propulsion subsystems are uncommon for CubeSats due to their size and weight constraints and limited operational life, but some propulsion subsystems do exist. CubeSats can employ cold gas thrusters, which release measured amounts of compressed gas from a tank to create thrust or monopropellant thrusters, which releases propellant that reacts with a catalyst to combust and rapidly expand to create thrust. (Wertz, Everett, and Puschell 2015) Other options exist for micro-thruster integration such as electrostatic propulsion systems. These systems can use a variety of gases that are electrically charged and are passed through a magnetic field to accelerate particles and create thrust. Electrostatic propulsion systems can achieve a higher thrust
impulse than cold gas and monopropellants, but with increased complexity and cost. (Wertz, Everett, and Puschell 2015)

The communications subsystem is critical for maintaining contact between the ground station and the orbiting satellite. Depending on the altitude and inclination of the orbit, the ground station and satellite may only have line of sight a few times a day, making data transfer critical during these periods. The communications subsystem must possess enough data bandwidth and link margin to facilitate data transfer during these periods. The C&DH subsystem collects information from the communications subsystem, the ADCS sensors, the position and orbit determination and control subsystem, feedback from the power and thermal subsystems, and processes that data to autonomously control the satellite system and report system status back to the ground station. Since communications with the ground station may only be achieved several times during a 24-hour period, the C&DH must be built with enough processing power and on board data storage to manage the subsystems of the satellite and collect and store data from the science payload.

The structure subsystem is largely specified by CubeSat specifications. Figure 12 is an example of a 3U CubeSat frame. The materials that can be used in accordance with CubeSat specifications are highly specified in order to prevent potential issues during launch, on orbit, and during reentry. CubeSat frames are constructed out of aluminum and must adhere to NASA’s outgassing approved materials. (CalPoly 2014) Other material requirements may specify melting point energy to manage reentry, hazardous material constraints, electromagnetic interference shielding, fundamental frequency, and stress, shock, and vibroacoustic standards. Additional structural specifications may dictate coatings preventing surface charging on satellites and promote thermal management. (University Nanosatellite Program NS-9 2016)

Figure 12: Example 3U CubeSat frame courtesy of FLARE PDR (Cahoy et al. 2017)
The thermal subsystem is designed to monitor and regulate the internal and external temperatures of the satellite in order to maintain acceptable operating temperatures in the on orbit environment. Thermal management on orbit is more difficult, since the primary means of heat transfer is through radiation and conduction. Conduction is useful to transfer heat through a medium, such as the structure of the satellite, to equalize internal temperatures. Radiation is the primary transfer process of thermal energy to and from the satellite system. The satellite can be coated in reflective or absorbent coatings to manage thermal energy. Use of multilayer insulation (MLI) can also assist in mitigating the absorption or loss of thermal energy. In larger satellite systems, the thermal subsystem may consist of internal heaters, radiators, and potentially a two-phase fluid-flow heat pipe system. (Wertz, Everett, and Puschell 2015) However, due to the size, weight, and cost constraints of a CubeSat, CubeSat systems typically rely on insulation, coatings, and sometimes radiators to manage thermal energy and operating temperatures while on orbit.

3.3 CubeSats as an Engineering System

Recall that an engineering system can be defined as “a class of systems characterized by a high degree of technical complexity, social intricacy, and elaborate processes, aimed at fulfilling important functions in society.” (“MIT Engineering Systems Division Strategic Report” 2011) On the surface, CubeSats appear to be technically complicated systems that are reducing complexity through standardization; however, CubeSats are becoming more socially interdependent, and thereby increasingly complex, as satellite miniaturization shifts focus for longer term space ventures. (Canis 2016) The reliance on miniaturized satellites for commercial and government ventures creates unique challenges and tradeoffs to achieve each system’s objectives and provide value.

Technical complexity of a system is defined as, “a system that has many elements or entities that are highly interrelated, interconnected, or interwoven.” (Crawley, Cameron, and Selva 2016) Already satellite subsystems are highly interconnected with dynamic trade spaces and narrow pareto fronts. Adding to the complex nature of satellites, the size, weight, and cost constraints inherent with CubeSat design require further trade evaluation to accomplish the system’s primary value function. Finally, CubeSats exist in a dynamic environment largely absent from human intervention once on orbit. These systems must employ elaborate processes autonomously to manage and maintain the system while continuing to provide value, even in the
event of a component failure. Despite CubeSat standards and commercial off the shelf equipment available to support CubeSat design and fabrication, the dynamic environment and internal system interdependencies of a CubeSat preserve its classification as a complex system.

The social interdependencies of satellites on orbit are a developing domain. New private markets supporting commercial launch or satellite design services are providing businesses, private organizations, and academic institutions access to space. With the launch hundreds of autonomous, non-government owned satellites, concerns over orbit collisions, reentry control, debris reduction, and physical and technological security have arisen, as well as questions about authority and protections granted to systems on orbit. These emerging trends and the reliance on space systems worldwide couple these complex systems to the social domain and drive system design and system requirements.

These technical and social complexities are not atypical for an engineering system; however, the size, weight, and often cost limitations placed upon CubeSats create a very narrow design space which is difficult to evaluate. CubeSats are not alone in this engineering space. Common engineering systems, such as the smart phone, have similar constraints inherent to the system design and system drivers. Although this is true for many artifacts and devices used in modern societies, CubeSats are largely unique designs, created for the purpose of data collection or testing, and are largely deprived of human intervention activities once deployed on orbit, with the exception of potential software updates and commands. The uniqueness of CubeSat design and the inability to modify hardware or design aspects once deployed emphasize the importance of understanding system interdependencies and the identification of critical nodes during the design phase.
Chapter 4: FLARE Case Study

The Free Space Lasercom and Radiation Experiment, or FLARE is a project led by MIT’s Space Telecommunication, Astronomy, and Radiation (STAR) Lab, and represents a joint venture between MIT’s Department of Aeronautics and Astronautics and Department of Nuclear Engineering. The project, which began in December 2015, is competing for funding and launch services through the University Nanosat Program (UNP). The UNP has existed for over fifteen years, is a partnership with the United States Air Force’s Office of Scientific Research and the Air Force’s Research Laboratory Vehicle Directorate. Together, these organizations, through the UNP, provides funding, information, and guidance to university satellite programs as well as a chance to complete or launch services. (University Nanosatellite Program NS 2016) Through the UNP, 11 nanosatellites have been launched and deployed on orbit.

4.1 FLARE Program Background

Figure 13: Proposed FLARE Design CAD diagram, courtesy of FLARE Team (Cahoy et al. 2017)

FLARE is a dual-purpose satellite constellation composed of two identical 3U CubeSats that will be deployed on orbit simultaneously. The first mission of the FLARE constellation is to demonstrate a free space optical (FSO) communication between nanosatellites. FLARE
accomplishes this objective through employment of optical transceivers on either CubeSat. Using precise attitude control of each system, FLARE hopes to establish a crosslink between the satellites with the intent to demonstrate higher data rates than comparable RF systems. The second mission of the FLARE constellation is to investigate high energy radiation environments in low earth orbit that lead to spacecraft degradation and anomalies. FLARE accomplishes this through the “Sparrow” particle detector, which discriminates against specific high energy electrons and protons, measuring the frequency of collisions. The intent of the Sparrow mission objective is to measure the variations and energy of space weather affecting satellites on orbit, using a low cost, low impact sensor package.

FLARE’s concept of operations (CONOPS) is designed for deployment of the constellation from the International Space Station (ISS) via the ISS CubeSat launcher. After a tandem deployment as a single 6U unit, each CubeSat will separate, independently conduct activation procedures, de-tumble activities, solar panel deployment and systems checks before proceeding to normal operations. The FLARE constellation will begin normal operations and orbit keeping following a successful system diagnostic and confirmation via the ground station. FLARE’s FSO communication tests and Sparrow’s particle measurements will be initiated via commands from the ground station, and the two systems are not expected to operate simultaneously due to power constraints. Following completion of a successful crosslink
demonstration, the FLARE constellation will continue to test crosslink data rates and varying distances, starting at 25km and continuing to 250km, or the end of the mission. Although the mission life of FLARE is designed for a six-month duration prior to de-orbit operations, it is possible for CubeSats to endure the space environment for longer durations.

4.2 FLARE System Engineering Challenges

CubeSats represent a difficult design space due to the size, weight and cost constraints associated with nanosatellite development and program funding limitations. FLARE possesses difficult trade spaces due to two science payloads and the necessity to have each satellite system’s attitude precisely controlled. During the design process, the FLARE project team struggled with the prioritization of subsystem performance against system constraints. Furthermore, the performance requirements for each of the system’s supporting functions required systems engineers to find creative solutions and make difficult design decisions to balance the cost, volume, and weight constraints.

The primary value functions of each FLARE system is FLARE communicating using Lasercom payload and FLARE measuring using Sparrow payload. At its core, these two functions must be supported for FLARE to meet its mission and system objectives. Using the OPD in figure 16 on page 49 as a model, we are able to graphically depict the primary value pathway of a FLARE system and the functions that support value creation, which are depicted in light yellow. The primary value functions modify operands across the system boundary to produce value. Both payloads yield raw data that must be processed prior to being sent to the
The lasercom payload processes data using a field programmable gate array (FPGA) and the Sparrow payload processing data internally through the use of a microcontroller. However, the lasercom payload also consumes data for transmission, indicating a multi-directional interface across the system boundary. Finally, the processed data is sent the CD&H subsystem for further processing and storage.

Supporting the primary value functions of the FLARE system are the remaining subsystems. The power subsystem is instrument to consuming solar radiation through a transforming process to yield electrical current. The ADC subsystem is instrument to the sensing process, which consumes physical signals across the system boundary, processes the data, delivers the processed data to the CD&H, processes and consumes commands from the CD&H to control actuators to modify the CubeSat’s attitude. The thermal subsystem senses thermal radiation across the system boundary to yield temperature data, which it then processes. Using the processed temperature data, the thermal subsystem relays that information to the CD&H, and the CD&H sends commands to the thermal subsystem, which the thermal subsystem processes and consumes to control internal heating elements within the CubeSat and regulate heat. The communications subsystem processes data received from the ground station to send to the CD&H and likewise processes data from the CD&H to send the ground station.

Using this OPD as a paradigm to understand each subsystem’s functionality, the systems engineer can begin to evaluate performance requirements and available trade spaces with respect to the primary value functions. Given that the payload sizes for the FSO laser communication system and Sparrow are 1U and 0.5U respectively, which is outlined in figure 15, system designers have approximately 50% of the available space within the 3U CubeSat bus to design remaining. Within this space, the systems designers must fit the remaining subsystems, structural connections, and wiring connections to satisfy internal system interfaces as well as spatial and functional dependencies.
Figure 16: Object Process Diagram for FLARE Satellite
To enable the primary value function of an optical communications crosslink, supporting functions must exist to properly stabilize and control the attitude of each FLARE CubeSat. Due to the sensitive nature of optical communications, the ADCS subsystem must be designed to achieve attitude stabilization without generating vibration or structural frequency anomalies. The need for a high quality, robust ADCS subsystem limits the development options to purchasing commercial off the shelf components. Given the cost, performance, and weight tradeoffs of using a robust ADCS subsystem, other subsystems must be reduced in scope or removed. Because of this, FLARE has made a trade decision between the ADCS and propulsion subsystems.

Originally, the FLARE design included an electrostatic propulsion system, which is characterized by small thruster size and reduced propellant mass. The propulsion subsystem would maintain each satellite’s orbit as well as the distance between satellites for crosslink operations. However, FLARE designers determined that the ADCS is a higher priority and that orbit maintenance is unnecessary for a satellite with a mission lifetime of six months. This eliminated the need for many of the performance benefits a propulsion subsystem would provide. Alternatively, to change the distance between satellites while on LEO, FLARE designers opted to control the drag forces acting upon each satellite by manipulating the cross-sectional area of the solar arrays. By choosing to use the solar arrays to control drag forces, the FLARE team removed the need for a propulsion system and used the budget, weight, and volume surplus to improve the ADCS subsystem performance.

The expanded ADCS package includes multiple sensors, such as: sun sensors, gyroscopes, star trackers, beacon signals to target crosslink optics, and a GPS receiver. These sensors provide raw data that must be processed in real time to determine the CubeSat’s attitude and generate the appropriate commands to the ADCS actuators. These actuators then generate force, through the use of reaction wheels or a magnetic torque, to modify the CubeSat’s orientation. The ADCS subsystem generates data and processing power requirements that must be met in real time. Further data and processing power requirements are generated from the FSO payload, the Sparrow payload, the communications subsystem, the thermal subsystem, and the power subsystem.

The ability of the CD&H subsystem to manage data requirements in real time is critical to the success of the FLARE mission. There are many tradeoffs between the design of the CD&H system and budgetary limitations, as well trades between CD&H performance and
electricity consumption or thermal energy management. Changes to science payloads and subsystem designs may have impacts upon the performance requirements of the CD&H, and thereby impact the power and thermal subsystems. The design specifications for the CD&H subsystem are representative of the many interdependencies of CubeSat systems and reflect the difficulty system engineers encounter when attempting to modify the design of a dynamic and interrelated system.

The discussion of the trades between the ADCS, propulsion, and CD&H subsystem for the FLARE project highlight the system engineering challenges of designing a highly interdependent system. The size, weight, and cost constraints of nanosatellite design provide a narrow space for trade analysis. The interdependencies of critical subsystem performance add a layer of complexity that makes design decisions difficult. Provided multiple design options and emergent system performance characteristics, it becomes necessary for the system engineers and designers to focus efforts on the most critical elements of a system and the relevant trades within that design space.

4.3 Motivation for ES-MDM Analysis

The complex and dynamic nature of evaluating interdependent system elements within a narrow design space makes it difficult for system engineers and designers to prioritize design efforts. The ES-MDM framework analyzes the relationships of system elements across the five domains of a system, highlighting interaction clustering. Using the relationships and dependencies of the system within those domains, the system designer can identify critical elements or nodes within the system. These critical design nodes must be prioritized over other elements to reduce system ambiguity and allow for a more effective examination of tradeoffs within the design space.

During the concept design phase of the FLARE system, several design architectures were identified and evaluated. Even through the course of detailed design, major changes occurred to the FLARE system architecture, causing rework and delays. Although some of the design changes were influenced by factors external to the project, such as technical limitations or vendor issues, many changes were caused by the ambiguity and uncertainty of designing an engineering system. Applying the ES-MDM framework to a nanosatellite system provides a new
paradigm to view CubeSat design, provide a point data source for system element prioritization, and assist future nanosatellite system designers to focus efforts for performance trade analysis.
Chapter 5: Constructing the ES-MDM

The construction of the ES-MDM requires an understanding of the technical and social aspects of the project system and the project team. These interactions are often difficult to discern on the surface, but become apparent through various qualitative analysis techniques. One such technique, the Qualitative Knowledge Construction (QKC) involves the coding and organizing of raw qualitative data based upon a system and its objectives. Coined by Jason Bartolomei in his 2007 MIT doctoral thesis, QKC builds upon the ground theory method, which systematically approaches analysis of qualitative data. (Bartolomei, 2007) Coupling the QKC technique with interviews and surveys of project team members and access to FLARE design documentation, a picture of the FLARE system as both a technical and a social system emerged. Using this information, the ES-MDM elements and relationships were discerned and the DSM and DMMs constructed.

5.1 Information Procurement and Abstraction

The primary source of technical information for the FLARE project came from the project design documents generated and revised for the PDR review in March 2017. These documents represent a snapshot in time for the FLARE system and are the framework for the technical and process domains. The documents included: the FLARE project proposal, organization charts, requirement verification matrix, interface documents, block diagrams, systems engineering discussion interviews, and other specific PDR deliverables for each subsystem. The technical design and architecture diagrams outlined the system’s object elements, processes, and functions of the system and subsystems, as well as indicated basic relationships and dependencies, interfaces, and information and energy flows.

The design documentation for the FLARE project was well documented, decomposing system objects down to the third or fourth level; however, provided the purpose of the ES-MDM analysis and the siloed nature of subcomponents, elements within the function and technical domains were abstracted to only the second level of decomposition for this analysis. The second level of decomposition represents the major components of each respective subsystem within FLARE. These major components, such as reaction wheels, sun sensors, solar panels, etc. largely interact only at the component level when decomposed past level 2 decomposition. Therefore,
the choice to focus on major components of each subsystem would not reduce the benefits from conducting an analysis of FLARE using the ES-MDM framework.

Although the technical and functional domains of the FLARE system were clearly outlined within supporting project documentation, the social and environmental domain information was less obvious. The task organization of the FLARE project team was established within the documentation; however, in practice the organization was flat, without a clear hierarchical structure. This is not surprising given that FLARE is a student led project, where concrete organizational hierarchies are unnecessary and team members work voluntarily. Furthermore, the prioritization of objectives and stakeholders within the system was not explicitly stated. To better understand the FLARE system in these domains, qualitative analysis techniques, interviews with key personnel, observed behavior and surveys were used to create a wealth of qualitative data.

The qualitative analysis technique used was the QKC method. This method analyzes qualitative documentation through a 9-step process:

Step 1: Identify the system of interest
Step 2: Define objectives for analysis
Step 3: Collect Data
Step 4: Code Data *(see figure 17 for example)*
Step 5: Organize coded data into system-level modeling framework
Step 6: Examine data for missing or conflicting information
Step 7: Resolve missing or conflicting data through additional research
Step 8: Perform Analysis
Step 9: Iterate
The results of the QKC analysis provided another layer of information from the available technical documents. This information formed the foundation for system elements and relationships within the environmental and social domains of the FLARE system, but still lacked adequate prioritization. To better understand the organization and the prioritization of stakeholders, objectives, and system elements within the technical domain, interviews were conducted with key project personnel, to include the project advisor, project manager, and chief engineer. Additionally, a survey was distributed to the project team that asked project engineers to list and prioritize what they thought were the system’s drivers, stakeholders, objectives, functions, objects, and activities. The goal of the survey was to assess the qualitative prioritization of subsystem designers and gain insight into the true prioritization of system elements.

The qualitative and quantitative data extracted from project documents, interviews and surveys built the foundation for system elements and relationships within the FLARE system. The QKC process identified several unanticipated elements within the environmental, social, and functional domains as well as a variety of element attributes. This analysis approach was well suited to the construction of DSMs for each system category, and enabled the complete picture of the system to be constructed within the ES-MDM framework. Although research was restricted to the system’s development in March 2017, the analysis techniques used to construct the ES-MDM provided enough data granularity to transform information into matrices.

Figure 17: Qualitative Knowledge Construction Coding Example (Jason Bartolomei 2007)
5.2 ES-MDM Development and Analysis

Using the QKC methodology as a framework to construct the ES-MDM, coded elements were separated into six categories: system drivers, stakeholders, objectives, functions, objects, and activities. These system categories holistically represent the environmental, social, functional, technical and process domains of an engineering system.

Step 1: Identify the system of interest

The system of interest is the FLARE project system, a student led CubeSat development project focused on the demonstration of new technological capabilities on orbit. The FLARE CubeSat project system is an engineering system with elements across the following domains:

*Environmental Domain:* The environmental domain consists of the non-human actors that influence the system across the system boundary. These often include regulatory agencies, technology innovations, and economic drivers. For FLARE, system drivers can be grouped into technology drivers, market drivers, and defense industry drivers. Figure 18 depicts the interdependencies among system drivers that influence the FLARE project.
**Social Domain:** The social domain consists of the human components that affect the FLARE project. These include project team members, the project advisor, STAR Lab, MIT, UNP, vendors and the Air Force Research Laboratory. An example of stakeholders external to the project are depicted in Figure 19, the FLARE Stakeholder Value Network (SVN).
STAKEHOLDER VALUE NETWORK

Public

ACADEMIC COMMUNITY
- Research
- Peer Review
- Public Information

MEDIA
- Press Release
- Public Opinion
- Public Information

PRIVATE SECTOR
- Capability Gap
- Research
- Donations

SOCIETY
- Public Opinion
- Donations

Public

Government

AIR FORCE
- Approved Project
- Project Information
- Program Support
- Public Information

NASA
- Project Funding
- Project Approval
- Best Practices
- Launch Vehicle

NOAA
- System Design
- Proposed System
- Purchased Components

Market

VENDORS
- Contracts

System Boundary

MIT & DEPARTMENTS
- Resources
- Project Lobbying
- R&D
- Lab Funding
- Project Workforce & Support

PROJECT
- Research Data
- Project Funding
- Project Approval

MIT Affiliates
- STAR LAB
- Equipment Access
- Research Data

FLARE TEAM
- Science Payload
- Public Opinion

SPARROW TEAM
- Public Opinion

Improved Services

Nonmarket Stakeholders
The Focal Organization
Market Stakeholders
Public Information
Political Information
Goods/Service
Financial
**Functional Domain:** The functional domain consists of the primary value function of the system and supporting technical functions to achieve the system objectives. Figure 20, the FLARE function decomposition, visually depicts the hierarchical breakdown of the FLARE technical system functions.

**Functional Decomposition**

- **FLARE Objective:** Demonstrate feasibility of nanosatellite optical crosslinks at a distance of 20km or greater and validate Sparrow particle spectrometer operations.

- **Primary Value Function(s):**
  - Lasercom Payload Message Communicating
  - Sparrow Payload Particles Measuring

- **Level 1 Supporting Functions:**
  - ADC Subsystem Attitude Controlling
  - CD&H Data Managing
  - Power Subsystem Powering
  - Thermal Subsystem Temp. Controlling
  - Comms Subsystem Message Communicating

- **Level 2 Supporting Functions:**
  - Lasercom Beacon Sensing
  - Light Sensing
  - GPS Signal Sensing
  - Commands Receiving
  - Magnetic Field Sensing
  - Data Receiving
  - Telemetry Housekeeping
  - Data Transmitting
  - Commands Transmitting
  - Data Storing
  - Data Processing
  - Energy Transforming
  - Electricity Storing
  - Electricity Processing
  - Actuators Rotating
  - Temperature Sensing
  - Temperature Heating
  - Transmitting Message
  - Receiving Message

*Figure 20: Functional Decomposition Diagram*

**Technical Domain:** The technical domain consists of the forms that comprise the system as operands or instruments. These physical entities support processes to achieve function objectives and generate system value along the value pathway. Figure 21 depicts the formal decomposition of FLARE objects. Components deemed sensitive with respect to FLARE specific technology or with respect to vendor proprietary information have been labeled with generic component terms.
**Process Domain:** The process domain consists of the processes and activities the technical system performs to support function objectives and generate value. The high-level processes of the FLARE system that support the technical system are depicted in figure 22 within their respective subsystems, highlighted as supporting instruments. The operands and instruments are abstracted at a high-level to highlight the interactions of the supporting processes.
**Process Architecture**

<table>
<thead>
<tr>
<th>Operands</th>
<th>Value Processes</th>
<th>Value Instruments</th>
<th>Supporting Processes</th>
<th>Supporting Instruments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Message</td>
<td>Communicating</td>
<td>Lasercom Payload</td>
<td>Heating</td>
<td>Thermal Subsystem</td>
</tr>
<tr>
<td>Particles</td>
<td>Measuring</td>
<td>Sparrow Payload</td>
<td>Sensing</td>
<td>Power Subsystem</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Transforming</td>
<td>CD&amp;H Subsystem</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Storing</td>
<td>ADC Subsystem</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Distributing</td>
<td>Comms. Subsystem</td>
</tr>
<tr>
<td>Electrical Current</td>
<td>Data</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Temperature</td>
<td>Attitude</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Figure 22: Process Architecture Diagram**

**System Boundary:** The system boundary is defined as the technical FLARE system, or the CubeSat, and the organic project team. With respect to the functional, technical and process domains, the system boundary is highlighted in figures 16 and 22 respectively. The environmental and social domain boundary is annotated in figure 19. The analysis assumes a systems level perspective for the FLARE project elements and relationships.

**Step 2: Define objectives for analysis**

*Objective 1:* Construct a model of the FLARE engineering system from a systems level perspective using the ES-MDM framework.

*Objective 2:* Conduct a network analysis of the ES-MDM to evaluate relationships and identify critical nodes within the system.

*Objective 3:* Conduct a change propagation analysis of the ES-MDM to identify “hot and cold” nodes within the system.
Step 3: Collect Data

Data for the FLARE project was collected from project technical documents, PDR documents, project member surveys, direct observation, interviews with the student project manager, the student chief systems engineering, and the project mentor. The majority of the data focused on the system design evolution at the preliminary design review in March 2017. Sample project documentation include, but are not limited to:

- FLARE UNP Proposal – March 14, 2017
- FLARE Mission Overview – February 16, 2017
- PDR System Engineer Interview – February 14, 2017
- FLARE Organizational Chart – February 26, 2017
- MIT FLARE PDR Slides – March 03, 2017
- FLARE Block Diagrams – February 16, 2017
- FLARE Assembly Procedures and Mechanical Design Package – February 15, 2017
- FLARE Interface Control Document – February 16, 2017
- FLARE Project Manager Interview – April 05, 2017
- FLARE Mentor Interview – March 21, 2017
- FLARE Survey Results – April 05, 2017
- FLARE Project Team Interviews – April 11, 2017

Step 4: Code Data

The documentation listed in step 3 was analyzed for content, but also coded using the QKC methodology outlined in Jason Bartolomei’s MIT doctoral thesis. This involved identifying key pieces of information, such as system drivers or stakeholders within a paragraph, highlighting that information, and recording specific attributes extracted from the document. For example, in figure 23 an excerpt from the PDR system engineer interview highlights several stakeholders, system drivers, and system objectives. These data points were coded in the margin of the document to highlight system elements and relationships pertinent to the ES-MDM framework.
Developing a compact, energetic particle spectrometer will be valuable to the defense and space weather community for deployment on a large number of spacecraft to allow environment monitoring, anomaly resolution, and environment model improvement and validation. Organizations such as the Space Weather Research Center at NASA Goddard, the Space Weather Prediction Center at NOAA, and the AFRL Space Weather Forecast Laboratory would benefit from additional data on the space weather environment in LEO. Data-driven models are essential to space weather forecasting. These data would also be relevant for industry seeking to optimize use of commercial off the shelf (COTS) parts and optimize designs using radiation tolerant or radiation-resistant components.

Figure 23: Example of QKC Coding in FLARE Technical Document

Step 5: Organize coded data in system-level modeling framework

Using the QKC coding results from step 4, the coded data was organized by system categories and constructed into N x N DSMs for system drivers, stakeholders, objectives, functions, objects, and activities. These DSMs were then analyzed across domains, producing DMMs for each system category combination. Combining these matrices into a 160 x 160 node matrix produces the visualization of the ES-MDM. The ES-MDM with the system boundary highlight in red is visible on the following page in figure 24. The gray box indicates a binary dependency between two nodes within the matrix. The black boxes along the diagonal of the DSMs reflect self-referential interactions, and therefore are not assessed. The software used to construct the ES-MDM was the Cambridge Advanced Modeler (CAM) software package. (Wynn et al. 2010)
Figure 24: FLARE ES-MDM
Steps 6 & 7: Examine data and resolve missing of conflicting information

For the purpose of the analysis, steps 6 and 7 were combined due to the interrelated nature of these QKC tasks. After the construction of the ES-MDM was completed, several interesting visual patterns emerged. First, the subsystem teams were visually represented as densely coupled nodes along the diagonal of the top left corner of the matrix, just under the system boundary (see figure 24). The team interactions are indicative of a flat organization, absent of hierarchy, which is collaborative in nature. Additional interviews were conducted with project team members to verify these assumptions. The reporting system within the project team was informal, with project team leads shifting regularly due to team member availability. Because of the simplistic interrelationships of each subsystem, subsystem teams could be abstracted to a single point node on the ES-MDM; however, the choice keep subsystem teams at the individual level was made to allow for changes amongst individual members to be represented in potential future time instantiations.

Other visual patterns that emerged from the ES-MDM involved project stakeholders and the function and technical domains. Specifically, there are tightly coupled node interactions between subsystem teams and their respective subsystem functions, objects, and activities. This is not surprising, given the collaborative nature of the project teams and the nascent design phase of the project. The stakeholders x functions, stakeholders x objects, and stakeholders x activities DMMs highlight the human components that contribute or support those respective functions, objects, and activities. (Bartolomei, 2007) Investigating the project team interactions through additional interviews concluded that only two project subsystem teams had assigned internal personnel on specific subsystem functions, objects, or activities due to the system architecture volatility prior to the PDR. Interviews with project team members highlighted additional engineer collaboration across subsystems that was not highlighted in the FLARE project documents. Future time instantiations of the FLARE project would likely yield less dense coupling among project team engineers and system technical elements.

Step 8: Perform Analysis

With the ES-MDM constructed and data and assumptions verified, the data set was prepared for analysis. The purpose of the analysis focused on the interactions among system elements, and the potential for change propagation to identify key nodes for system flexibility.
Although the ES-MDM reflects the initial representation of FLARE system interactions, network and change propagation analysis could yield important system design conclusions without needing a comparison of data sets across the temporal domain. Figure 25 depicts a visual representation of the FLARE ES-MDM using an undirected network graph.

The visual representation of the entire FLARE ES-MDM in figure 25 highlights the highly-interconnected elements of the system as well as potential key nodes and clusters that are focal points for design. These elements are potential “hot spots” or areas that propagate change within the system boundary. Bartolomei’s discusses in his thesis the concept of hot/cold spots within an ES-MDM and develops a methodology to highlight social and technical sources of uncertainty such as: organizational changes, funding or budget instability, technology innovation, and changing customer requirements. (Bartolomei, 2007) The analysis of the FLARE
ES-MDM explores an evolution Bartolomei’s hot/cold spot methodology, as well as apply network analysis metrics to identify and assess key elements within the system.

Flexible Design Opportunities

Building upon Jason Bartolomei’s ES-MDM work, Jennifer M. Wilds in her 2008 MIT master’s thesis conceived a framework to model change behavior across multiple domains. Highlighting nodes labeled as Flexible Design Opportunities (FDO), previously identified as hot spots, Wilds’ methodology used the following steps to determine the FDOs within a system (Wilds 2008):

Step 1: Construct the ES-MDM
Step 2: Identify Change Scenarios
Step 3: Identify Change Initiators and Relationships Types
Step 4: Reduce the ES-MDM to Subgraphs
Step 5: Conduct Change Propagation Analysis
Step 6: Calculate of the Desired Flexibility Score
Step 7: Recognize FDOs

Since the ES-MDM is already constructed, the analysis starts with step 2, or the identification where changes are likely to occur. Change scenarios are defined by the uncertainties affecting system drivers and the probability that the system driver will change in the future. (Wilds 2008) Analyzing the system drivers of FLARE, four system drivers were selected for FDO analysis due to their high uncertainty and change probability: technology innovation, technology affordability, and cyber security/telecom services. The selected system drivers and corresponding uncertainties are highlighted in the following change scenarios:

Change Scenario #1: Payload Innovation
Innovative technology becomes available or is developed internally during the FLARE design process that would improve payload performance and functionality if adopted into the system. The energy, data, size, and weight characteristics of this innovative technology are unknown.
Change Scenario #2: Component Price Increase
The cost of vital subsystem components that were planned to be procured by vendors rises substantially. Due to the cost increase, new components would need to be procured for alternate vendors, but have different performance characteristics, or components would need to be built internally, increasing risk and performance uncertainty.

Change Scenario #3: New Stakeholder Changes Mission Objective
New cyber security or telecom services influence from external entities increases interest in optical crosslink communications from previously unidentified system stakeholders. These new stakeholders influence the design process and change the FLARE mission priorities or objectives.

Table 3: Change Scenario Probability

<table>
<thead>
<tr>
<th>Change Scenario</th>
<th>Probability Occurring (Pcs)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Payload Innovation</td>
<td>0.3</td>
</tr>
<tr>
<td>Price Increase</td>
<td>0.5</td>
</tr>
<tr>
<td>New Stakeholder</td>
<td>0.7</td>
</tr>
</tbody>
</table>

Table 3 identifies the probability of each change scenario occurring (Pcs). The cyber security and telecom service system drivers are coupled together in change scenario #3 due to their interrelated nature. The probabilities are based upon observed discussions and risk analysis within the project teams. Using each of the three change scenarios, step 3 identifies the system change initiators and respective relationship types that could potentially affect the system if the change scenario occurred. In this instance, six change initiators were identified within the system. A change initiator is defined as “a component where the change enters the system,” or rather the system element that is affected by a system driver and generates a change within the technical system. (Wilds 2008) The change initiators in table 4 reflect the likely technical changes and relationships that would affect the system provided each of the three change scenarios. These initiators were derived from relationships between system objects and the system drivers in the ES-MDM.
Table 4: Change Initiator/Relationship

<table>
<thead>
<tr>
<th>Change Initiator</th>
<th>Relationship Types</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Power</td>
</tr>
<tr>
<td>Lasercom</td>
<td>0.8</td>
</tr>
<tr>
<td>Payload</td>
<td></td>
</tr>
<tr>
<td>Sparrow</td>
<td>0.8</td>
</tr>
<tr>
<td>Payload</td>
<td></td>
</tr>
<tr>
<td>ADCS</td>
<td>1</td>
</tr>
<tr>
<td>EPS</td>
<td></td>
</tr>
<tr>
<td>Lasercom</td>
<td>0.8</td>
</tr>
<tr>
<td>Payload</td>
<td></td>
</tr>
<tr>
<td>ADCS</td>
<td>0.2</td>
</tr>
</tbody>
</table>

The relationships identified in table 4, are listed on a scale of 0 to 1, where 0 indicates a change will not occur, and 1 indicates a change will occur. In her MIT master’s thesis, Wilds’ creates a scale to assess the likelihood of occurrence, but limits the numeric values to 0, 0.2, 0.8, and 1. She states that this differentiation amongst relationship probabilities discourages fence-sitting, or probabilities of 50%, and reduces subjectivity of assessing these likelihoods. (Wilds 2008) These probabilities, labeled $P_{CIRT}$ (change initiator – relationship type), represent the likelihood that a relationship type is effected when the change initiators are activated in response to the change scenario. The choice to keep change initiators as the subsystem level was made due to the strong interdependencies amongst subsystem components; changing a component or performance characteristic within a subsystem would likely have cascading effects within that respective subsystem. However, the analysis focuses on specific components that would like initiate change within that subsystem, which would potentially propagate through the FLARE system.

Step 4 of Wilds’ FDO methodology is the reduction of the ES-MDM to subgraphs, or the “removal of components unrelated to the change scenario for the change propagation analysis, thereby creating a subgraph of the system.” (Wilds 2008) Step 4 must be completed for each change scenario and each initiator/relationship pairing, totaling 18 subgraphs for this analysis. Figures 26 and 27 highlight an example subgraph and subgraph network diagram for change.
scenario 1, Lasercom Payload initiator, and power relationship of the FLARE system. At this stage of the analysis, the direction of change that propagates within the system has not been determined. The subgraph and network graph depict the components that could be impacted by a change initiator, specifically within the technical domain.

Figure 26: Example Subgraph for Change Scenario 1, Lasercom Payload Initiator, Power Relationship
After the subgraphs are created for each set of change initiator and relationship type pairs across the three change scenarios, the next step of the FDO methodology, step 5, analyzes the 18 subgraphs over four actions: indication of change propagation, estimation of probability of change (P_c), assignment of switch costs (CS), and calculation of expected expense (CEE). (Wilds 2008) The indication of change propagation uses a subgraph network diagram, like figure 27, to follow change propagation through a system from the originating change initiator element. Tracking the objects and probabilities that change will be propagated or absorbed results in a list of components that are affected downstream from the change initiator. This is graphically displayed as a change graph, figure 28. In the figure, component 3 initiates change within the power relationship type. Following the change through the system, change propagates from component 3 to component 2 and the EPS. Component 2 propagates change to component 4, and the EPS propagates change to the battery and solar panels. However, component 4 absorbs all additional change and the EPS absorbs change from affecting the ADCS and communications interface.
Following the change flow through the system, the next action of step 5 calculates the probability of change propagation ($P_c$) for each of the affected components identified in the change graph. The probability of change propagation acts as a weighting between components along the change flow. There is a 90% likelihood change will propagate ($P_c$) from component 3 to component 2. The EPS has a 10% $P_c$ of changing due to a change in component 3 and so on. The $P_c$ estimations were calculated based upon the operational dependencies of each component within the subsystem and the estimated likelihood that a change to one component would affect the performance characteristics of another.

Following the estimation of the probability of change propagation for each affected component, step 5 next calculates the component switch cost (CS), or the cost associated with modifying/replacing the component in response to the incoming change. (Wilds 2008) The component switch cost is the cost of labor or material, depending whether the component is sourced internally or externally, to change or modify a component within the system.

Finally, to calculate the component expected expense, the last action of step 5, for each component affected by the change propagation, the probability of change is multiplied by the switch cost of the component. This provides a weighted cost of each element based upon the likelihood change propagates to that component based on the change initiator and relationship.
The CEE was calculated across all 18 CIRT pairings over the three change scenarios. The calculations for change scenario #1, laser payload initiator, power relationship using the following formula are illustrated in table 5.

\[ \text{CEE} = P_e * CS \]

Table 5: Example Component Expected Expense for Change Scenario 1, Lasercom Payload Initiator, Power Relationship

<table>
<thead>
<tr>
<th>Object</th>
<th>( P_e )</th>
<th>CS</th>
<th>Component Expected Expense</th>
</tr>
</thead>
<tbody>
<tr>
<td>Component 3</td>
<td>1</td>
<td>5500</td>
<td>5500</td>
</tr>
<tr>
<td>Component 2</td>
<td>0.9</td>
<td>4000</td>
<td>3600</td>
</tr>
<tr>
<td>Component 4</td>
<td>0.9</td>
<td>240</td>
<td>216</td>
</tr>
<tr>
<td>EPS</td>
<td>0.1</td>
<td>1400</td>
<td>1400</td>
</tr>
<tr>
<td>Battery</td>
<td>0.3</td>
<td>4000</td>
<td>1200</td>
</tr>
<tr>
<td>Solar Panels</td>
<td>0.2</td>
<td>2000</td>
<td>400</td>
</tr>
</tbody>
</table>

Change costs in table 5 were extracted from FLARE documentation, but deliberately estimated to avoid potential release of proprietary information. The results in table 5 highlight one of 18 unique estimates for CEE across the change scenarios, relationship types, and change initiators. Combining the 18 data sets yields the total \( \text{CEE}_{cs} \) all components by change scenario. Summing the product of each scenario’s \( \text{CEE}_{cs} \) and \( P_{cs} \), yields what Wilds’ calls the desired flexibility score (DFS) for all of the objects within the system.

\[ \text{DFS} = (P_{cs1} * \text{CEE}_{cs1}) + (P_{cs2} * \text{CEE}_{cs2}) + (P_{cs3} * \text{CEE}_{cs3}) \]

The results for the system are displayed as a graph in figure 29. Wilds’ defines FDOs for a system as the physical objects ordered on a DFS graph prior to the DFS score curve nearing a slope of 0. Note that some system elements have a DFS of zero – this is due to the change scenarios not propagating change within those components. Using the zero slope heuristic, the identified FDOs for the FLARE system are: \( \text{EPS} \), reaction wheels, battery, and component 3 of the lasercom payload.
Network Analysis

As presented earlier, figure 25 displays the FLARE ES-MDM as an undirected network graph. A graph is a mathematical representation of a collection of connected objects. Each object in the graph is called a node, and nodes are connected by edges or links, indicating a relationship or dependency. (Boston University 2010) The network graph in figure 25 reflects the complex nature of the FLARE system, highlighting 160 nodes and 2,341 edges. Because of the complex network of connections and relationships, several algorithms can be applied to mathematically measure relationships within the network. Degree centrality is one such measurement, and is the sum of the number of relationships received and initiated by a node. Another measurement, betweenness or betweenness centrality, is the measure of how often a node sits in the shortest path connecting two other nodes. (Bartolomei, 2007)

These two metrics for network graph analysis will be used to compare different system categories within the ES-MDM, such as stakeholders or objects, against the respective stakeholder or object DSMs. The goal of this network analysis will be to rank the most connected elements within the system, evaluate the results, and compare those results to the
results compiled in the FDO analysis. Although other algorithms can be applied to the system, such as network diameter, average path length and average degree, these metrics for network analysis would only be useful to compare the entire FLARE ES-MDM with changes over time instantiations, and not useful for comparison internally.

The first elements of the FLARE system analyzed using network analysis are the physical objects of the FLARE system. The purpose of this analysis is to identify the most connected objects with respect to the objects DSM and the ES-MDM. This provides insight into the rankings of physical object connectedness within the technical domain but also compare physical object rankings with respect to the functional, process, social and environmental domains. Table 6 highlights the quantitative results of the betweenness centrality analysis of FLARE objects.

<table>
<thead>
<tr>
<th>Objects</th>
<th>Object DSM Betweenness</th>
</tr>
</thead>
<tbody>
<tr>
<td>3U Chassis</td>
<td>0.3461</td>
</tr>
<tr>
<td>EPS</td>
<td>0.2424</td>
</tr>
<tr>
<td>Internal Rail System</td>
<td>0.1764</td>
</tr>
<tr>
<td>Component 1</td>
<td>0.1056</td>
</tr>
<tr>
<td>Microcontroller</td>
<td>0.0871</td>
</tr>
<tr>
<td>ADCS Interface Board</td>
<td>0.0424</td>
</tr>
<tr>
<td>Component 3</td>
<td>0.0421</td>
</tr>
<tr>
<td>Tx/Rx Telescope</td>
<td>0.0364</td>
</tr>
<tr>
<td>Comm Interface Board</td>
<td>0.0315</td>
</tr>
<tr>
<td>Component 5</td>
<td>0.0288</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Objects</th>
<th>Object ES-MDM Betweenness</th>
</tr>
</thead>
<tbody>
<tr>
<td>3U Chassis</td>
<td>0.0651</td>
</tr>
<tr>
<td>EPS</td>
<td>0.0599</td>
</tr>
<tr>
<td>Component 1</td>
<td>0.042</td>
</tr>
<tr>
<td>Internal Rail System</td>
<td>0.0336</td>
</tr>
<tr>
<td>Reaction Wheels</td>
<td>0.0229</td>
</tr>
<tr>
<td>ADCS Interface Board</td>
<td>0.0185</td>
</tr>
<tr>
<td>Component 3</td>
<td>0.0143</td>
</tr>
<tr>
<td>Microcontroller</td>
<td>0.0136</td>
</tr>
<tr>
<td>Comm Interface Board</td>
<td>0.0115</td>
</tr>
<tr>
<td>Tx/Rx Telescope</td>
<td>0.0109</td>
</tr>
</tbody>
</table>

The results of the betweenness centrality calculations between the top 10 elements within the objects DSM and the ES-MDM yielded surprising results. Both the object DSM and the ES-MDM highlighted the FLARE chassis and EPS as the components with the highest betweenness. In this context, this means that these system nodes are most often on the shortest path between other nodes within the system. This is not surprising, since these two elements would have the most physical connections among other elements within the system. However, the differences between the object DSM and ES-MDM results provide interesting insight into the system. With respect to the ES-MDM, the reaction wheels make the top ten list, whereas in the objects DSM, the reaction wheels do not. This is largely due to the performance coupling of the lasercom
payload and the ADCS actuators, primarily the reaction wheels, which is not present in the objects DSM. Although the results of the betweenness centrality calculations of the objects DSM and ES-MDM provided comparable results, the analysis reinforced the importance of the interactions across domains within FLARE.

Next the objects DSM and ES-MDM are analyzed using the centrality measurement for network graph analysis. Where betweenness is amount a specific node is present on the shortest path between two vertices for all vertices, centrality focuses on the edges or relationships a specific node has. In a system context, centrality is a measurement of interconnectedness and may be seen as a measurement of influence within a system. Comparing the objects DSM to the ES-MDM provides context between connections within the physical technical system and connections within the engineering system itself. Table 7 displays the results on the centrality calculations for the object DSM and ES-MDM.

<table>
<thead>
<tr>
<th>Objects</th>
<th>Object DSM Centrality</th>
<th>Objects</th>
<th>Object ES-MDM Centrality</th>
</tr>
</thead>
<tbody>
<tr>
<td>EPS</td>
<td>44</td>
<td>EPS</td>
<td>58</td>
</tr>
<tr>
<td>3U Chassis</td>
<td>43</td>
<td>3U Chassis</td>
<td>55</td>
</tr>
<tr>
<td>Internal Rail System</td>
<td>40</td>
<td>Internal Rail System</td>
<td>52</td>
</tr>
<tr>
<td>ADCS Interface Board</td>
<td>27</td>
<td>ADCS Interface Board</td>
<td>45</td>
</tr>
<tr>
<td>Comm Interface Board</td>
<td>18</td>
<td>Component 1</td>
<td>43</td>
</tr>
<tr>
<td>Component 1</td>
<td>14</td>
<td>Component 3</td>
<td>38</td>
</tr>
<tr>
<td>Single Board Computer</td>
<td>13</td>
<td>Component 5</td>
<td>38</td>
</tr>
<tr>
<td>Thermal Interface Board</td>
<td>11</td>
<td>Tx/Rx Telescope</td>
<td>38</td>
</tr>
<tr>
<td>Microcontroller</td>
<td>10</td>
<td>Component 4</td>
<td>36</td>
</tr>
<tr>
<td>GPS Receiver</td>
<td>10</td>
<td>Component 6</td>
<td>36</td>
</tr>
</tbody>
</table>

The results of the object centrality comparison highlight the top ten most interconnected objects within the object DSM and ES-MDM. The EPS, FLARE chassis, rail system and ADCS interface board are the most interconnected elements across both models. However, the ES-MDM centrality results highlight the interconnectedness of the lasercom payload across domains, whereas the objects DSM does not. This is likely due to the stakeholder interest and system objective and functional focus upon the lasercom mission. The Sparrow payload did not make an appearance in either list, reaching the fifteenth most connected elements on the ES-MDM object centrality list. Sparrow is designed as a low impact sensor system, so having lower
centrality results makes sense; however, the lack of connections for the ES-MDM centrality results also indicates potential diverging objective focus for the system and Sparrow.

Looking at the social domain, specifically internal and external stakeholders, the same algorithms to calculate betweenness centrality and centrality can be applied. First assessing external stakeholders, this analysis can prioritize influence, and thereby affect communications management of the project team. With respect to internal stakeholders, mainly the project team, this analysis can provide insight into the management of personnel and importance or influence of key engineers. Tables 8 and 9 depict the top five results of the internal and external stakeholder analyses.

<table>
<thead>
<tr>
<th>External Stakeholders</th>
<th>Stakeholder DSM Betweenness</th>
<th>Stakeholder DSM Centrality</th>
</tr>
</thead>
<tbody>
<tr>
<td>STAR Lab</td>
<td>0.0651</td>
<td>21</td>
</tr>
<tr>
<td>MIT</td>
<td>0.0587</td>
<td>18</td>
</tr>
<tr>
<td>Media</td>
<td>0.0198</td>
<td>17</td>
</tr>
<tr>
<td>Lincoln Lab</td>
<td>0.0529</td>
<td>16</td>
</tr>
<tr>
<td>Society</td>
<td>0.0172</td>
<td>16</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>External Stakeholders</th>
<th>ES-MDM Betweenness</th>
<th>ES-MDM Centrality</th>
</tr>
</thead>
<tbody>
<tr>
<td>STAR Lab</td>
<td>0.0142</td>
<td>29</td>
</tr>
<tr>
<td>NASA</td>
<td>0.0128</td>
<td>27</td>
</tr>
<tr>
<td>Lincoln Lab</td>
<td>0.0089</td>
<td>21</td>
</tr>
<tr>
<td>MIT</td>
<td>0.0031</td>
<td>20</td>
</tr>
<tr>
<td>AFRL</td>
<td>0.0034</td>
<td>19</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Internal Stakeholders</th>
<th>Stakeholder DSM Betweenness</th>
<th>Stakeholder DSM Centrality</th>
</tr>
</thead>
<tbody>
<tr>
<td>Project Manager</td>
<td>0.1947</td>
<td>21</td>
</tr>
<tr>
<td>Chief Engineer</td>
<td>0.0938</td>
<td>18</td>
</tr>
<tr>
<td>Project Mentor</td>
<td>0.0817</td>
<td>15</td>
</tr>
<tr>
<td>Lasercom Engineer 1</td>
<td>0.0637</td>
<td>14</td>
</tr>
<tr>
<td>Sparrow Engineer 1</td>
<td>0.0612</td>
<td>10</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Internal Stakeholders</th>
<th>ES-MDM Betweenness</th>
<th>ES-MDM Centrality</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chief Engineer</td>
<td>0.0888</td>
<td>51</td>
</tr>
<tr>
<td>Sparrow Engineer 1</td>
<td>0.0507</td>
<td>51</td>
</tr>
<tr>
<td>Project Manager</td>
<td>0.0422</td>
<td>45</td>
</tr>
<tr>
<td>Systems Engineer</td>
<td>0.036</td>
<td>42</td>
</tr>
<tr>
<td>Lasercom Engineer 1</td>
<td>0.0238</td>
<td>74</td>
</tr>
</tbody>
</table>

The use of centrality as a measurement for analyzing internal stakeholder importance proved inaccurate due to the flat organization of the project team. Since the project team is absent of traditional subsystem team hierarchy, the centrality measurement included the highly-coupled relationships of all the team members on the subsystem. This resulted in the artificial prioritization of larger subsystem team members as more influential within the entire project team. Therefore, the internal stakeholder results are ordered by the betweenness centrality.
measurement. The results from the internal stakeholder DSM analysis highlight the most involved engineers on the project team. The internal stakeholder metrics calculated through the ES-MDM produced a slightly different ordering, but were overall very similar with one exception. The systems engineer moves into the top five influential engineers in the ES-MDM, most likely due to the impact of cross domain relationships, particularly in the objective and function system categories.

With respect to external stakeholders, the results were as expected. STAR Lab is the parent organization to the FLARE project, with many of the same stakeholder relationships that the project has. The changes to the external stakeholder prioritization from the stakeholder DSM to the ES-MDM highlight the holistic influence stakeholders have across the system. The addition of the AFRL and NASA into the top five influential stakeholders for the ES-MDM table reflects the interest in the FLARE’s mission objective and technological goals. The absence of UNP from the analysis results was surprising, since the FLARE project directly reports to UNP. This is likely due to UNP’s singular directional focus on project team mentorship and the minimal influence UNP has among system drivers that affect the goals and objectives of the FLARE system.

Step 9: Iterate

Due to the design progress of the FLARE system, it was not possible to reconstruct earlier ES-MDM iterations based upon available data. The limited observation period for the ES-MDM analysis failed to yield additional time instantiations based upon the FLARE development schedule. Therefore, the analysis conducted represents a single snapshot of the FLARE system at the time of the PDR. Further iterations of this process to incorporate new data sources and design changes would improve network analysis conclusions and add breadth to the study; however, this is beyond the scope of this thesis.

5.3 Conclusions from ES-MDM analysis

The ES-MDM process provided valuable tools and methodologies to extract qualitative data from varying sources and build a holistic view of the FLARE engineering system. The QKC methodology was critical to determine the system drivers and stakeholders of the FLARE system from FLARE documentation. Although, interviews and surveys of the FLARE team provided
insight into the system, many of the team members did not have broad enough of a view of the FLARE project to describe the environmental domain. However, the FLARE documentation was constructed with enough detail to extract the necessary information and develop the context within which FLARE is designed. The ES-MDM is a reflection of that information, which enabled the quantitative evaluation of qualitatively sourced data.

The flexible design opportunities approach created by Jennifer M. Wilds, built upon the hot/cold spot framework originally described by Jason Bartolomei’s doctoral thesis. Combining sensitivity techniques, relationships across domains, and weighting technical system changes by component cost yielded interesting flexibility opportunities. CubeSat design is typically highly constrained and requires trade analysis for performance, weight, power consumption, and cost. The weighting of component costs into change scenarios and change propagation enables system designers to understand potential change initiators and build flexibility into systems based upon perceived uncertainties or risks. Although cost constrained systems, such as typical CubeSat projects, may not have the budget flexibility to exercise robust design of FDOs, the knowledge that those nodes are considered “hot spots” or FDOs is invaluable to system designers.

With the regards to the four FDOs for the FLARE system, the EPS, reaction wheels, battery and what is identified as component 3 of the lasercom payload, the identification of these elements quantitatively and qualitatively makes sense with respect to system risks. The EPS and battery are likely change propagators, since if these components are designed to meet the projected energy needs in a degraded state, such as end of life, these systems are likely to lack performance robustness. In this scenario, any significant change to power consumption of the lasercom payload or ADCS would like propagate change through the power subsystem. The identification of the reaction wheels as a change propagator is likely due to their relatively high component cost. Although network analysis did not identify the reaction wheels as an influential component, the cost and potential risk of change propagation makes the reaction wheels significant.

The network analysis results with respect to system objects were less significant. The identification of the chassis, EPS and rail system amongst both the object DSM and the ES-MDM made sense with respect to the centrality and betweenness metrics. However, the identification of the Sparrow microcontroller and GPS receiver in the object DSM’s lists for betweenness and centrality are of questionable value. Since both components are relatively low
impact sensors, the number of physical connections within the objects DSM is minimal. However, the value of this type of analysis is improved through the application of these metrics across domains through the ES-MDM, which prioritizes objects based on the relationships within the system over the entire system. The ES-MDM network analysis results for system objects did identify those elements that were the most tightly coupled overall within the system. Without cost considerations, the betweenness metrics did highlight the reaction wheels as a significant component, likely due to their coupling with lasercom functionality and system objectives.

The evaluation of external and internal stakeholders for FLARE provided a quantitative methodology to rank stakeholders and analyze project team dynamics. Admittedly, this type of analysis is not well suited for the FLARE project, due to its fluid project team dynamics and informal, flat team structure. However, the insights that were assessed for key project engineers for the engineering system (across domains) and the influence stakeholders have upon the functional or technical domains of the system was particularly interesting. The re-evaluation of external stakeholders using the ES-MDM calculations for centrality and betweenness highlighted the stakeholders that influenced the design of the FLARE system the most, where the stakeholder DSM analysis only highlighted the stakeholders with the most connections. In all stakeholder analyses, the addition of the ES-MDM relationship measurements across multiple domains added value to the evaluation process and the conclusions that were inferred. This type of stakeholder analysis would benefit projects with large project teams or a large number of stakeholders spread over multiple organizations.

The FDO and network analysis provided interesting insights into the development of the FLARE system. Some of the insights are intuitive; however, the ability to quantitatively prioritize stakeholders and elements is invaluable for conducting tradeoff analysis. The ES-MDM reinforces the need for system designers to consider the social and environmental domains when designing a complex engineering system. Although the modularity of CubeSat subsystems simplifies some of this analysis, the relationships that emerge across domains with respect to the CubeSat payload(s) and supporting systems is a substantial realization. This provides system designers analytical foundation to prioritize system design and tradeoffs in a highly-constrained design space.
Chapter 6: Conclusions and Future Work

6.1 ES-MDM CubeSat Applications

The burgeoning space industry is accelerating rapidly, with innovative technologies entering the market at an alarming rate. As of April 2017, the world’s first reused stage one rocket successfully traveled to orbit a second time, and for a second time successfully landed on Earth. The expansion of on orbit opportunities will continue as new market dynamics, like reduced launch service costs or increased affordability of miniaturized technologies, make on orbit opportunities affordable and accessible to a mass market audience. The satellite industry itself is experiencing a revolution, with corporations investing in widespread constellations of small satellites. As the launch of secondary payloads, like CubeSats, becomes more common, the reality of academic institutions and research organizations having a constant presence on orbit becomes possible.

However, the design and fabrication of miniaturized satellites does not abate the challenges of developing a highly-constrained space system. These engineering systems are both social and technical in nature, displaying relationships across multiple domains. To reduce ambiguity and increase understanding, the systems engineering field developed many different frameworks and methodologies to better model the complexity and interactions of these complex systems. One such model, the ES-MDM, has been the focus of analysis for this thesis.

The ES-MDM is well suited for complex socio-technical systems. CubeSats remain a challenge for many frameworks due to the technical system constraints as well as the limitations of the organizations developing them. The application of the ES-MDM framework to the FLARE project highlighted many of the strengths of the model and some of the weaknesses. As a technical system, CubeSats are complex, interrelated entities that have difficult trades and easily propagate change. The ability to identify potential change scenarios, assess points of change propagation and develop methods to exercise flexibility is a powerful tool. In this way, system designers can work to maximize value and minimize risk through incorporating flexibility in design.

The ES-MDM helps project managers and systems engineers monitor the development of CubeSat systems over time. Although CubeSat project teams in an academic setting usually consist of a small group of students, the ability of project managers to track changes to project personnel over time and record the emergent effects of those changes is a management process.
Furthermore, the formal tracking of how system drivers and stakeholders influence the functional and technical domains of the system is a valuable design process that is often overlooked.

6.2 Assessment of ES-MDM Methodology

Despite the success of the ES-MDM methodology for the analysis of complex systems, the framework does possess several shortcomings. First, the building of the ES-MDM model is labor intensive and time consuming. Although previous research into the ES-MDM has had access to model specific software programs to construct the ES-MDM and generate model views, the research conducted in this thesis did not. A combination modeling platforms, to include the Cambridge Advanced Modeler (CAM), Object Process Case Tool (OPCAT), Matrix Laboratory (MATLAB), and the Pajek network analysis tool were used to construct and analyze elements of the ES-MDM. Although most of these tools are free to download or can be acquired with limited license agreements, the construction and analysis of the ES-MDM required integration management across several software platforms. The availability of a software platform to collect and organize data using the coding process outlined in the QKC methodology, to construct the corresponding DSMs and DMMs of the system to be analyzed, and to analyze the fully established ES-MDM using a variety of sensitivity and network analysis techniques would be invaluable.

The ES-MDM lacks the capability to highlight specific relationships or dependencies in the technical domain, such as information, energy, mass flow, or physical connection, and in the social domain, such as the flow of information, political influence, funding, and services. Although the ES-MDM was able to capture the relationships across multiple domains of the engineering system, the relationships were binary in nature and did not reflect relative weighting attributes. The lack of relationship attributes or relationship weighting decreased the level of granularity achieved within the model. This potentially skewed some of the results, specifically analysis across the environmental and social domains. Further analysis of incorporating relationship weights or relationship attributes into the ES-MDM is needed.

However, these framework deficiencies were overshadowed by the ES-MDM’s ability to model an engineering system across the environmental, social, functional, technical and process domains. The QKC methodology provided an approachable framework for extracting qualitative data from large quantities of information in a deliberate and organized manner. Although
interviews and surveys provided view of the environmental and social factors affecting the engineering system. The QKC coding approach extracted important qualitative data from project documents that would not have otherwise been used for the ES-MDM construction. This created a holistic view of the socio-technical system being analyzed, which allowed the system to be analyzed across domains. Specifically, the methodology for determining system hotspot outlined by Jason Bartolomei and expanded to determine FDOs by Jennifer M. Wilds highlights a unique capability of the ES-MDM.

With respect to DoDAF 2.0 and SysML frameworks, the ES-MDM framework outlines a methodology to understand and reduce ambiguity of an engineering system across the technical, behavioral, and parametric paradigms while still providing tools within the framework to enable analysis. With respect to SysML specifically, ES-MDM covers all system views except for the requirements view of the system. Although the objectives of the system are depicted in the ES-MDM, no explicit flow of requirements from the social and functional domain is translated to the technical and process domains of the system. However, the ES-MDM integrates well with the DoDAF 2.0 framework, with specific DMMs and DSM of the ES-MDM translating to DoDAF 2.0 views. See figure 30 for a visual depiction.

![Figure 30: DoDAF 2.0 viewpoints overlaid in an ES-MDM (J. Bartolomei et al. 2012)](image_url)
The ES-MDM framework also nests well with Dov Dori’s object process methodology, which focuses on the interactions between objects and processes that compose functions within a system to produce value. The relationships amongst objects and processes are captured within the object and process system categories of the ES-MDM, while the functional interactions and value creation are captured in the objectives and functions categories. Furthermore, previous research has worked to transcribe the visual interactions of objects and processes of the an OPD into adjacency matrices using MATLAB. Although the adjacency matrices produced using current methods do not directly correlate to the function, object, and process/activity DSMs created within the ES-MDM, there is potential for future iterations to directly correlate to the ES-MDM framework.

Overall, the ES-MDM framework approaches socio-technical systems holistically, successfully modeling the interrelationships and interdependencies of an engineering system in a single system view. The influence of the environment and social domains upon the function, technical, and process domains highlight the need for a framework that encompasses all domains, especially for highly coupled complex systems. The ES-MDM is an adaptive framework that is capable of analyzing a resource constrained system, such as a CubeSat, to assess flexible design points mapped to uncertainties. However, future research is needed to improve upon the existing framework and investigate analysis techniques for multiple-domain interactions.

6.3 Future Work

The majority of research conducted using the ES-MDM focuses around the analysis of the MAV-PD engineering system, originally conducted by Jason Bartolomei in 2007. To improve upon the QKC process and the ES-MDM framework, additional engineering systems must be analyzed. Furthermore, the scope of this thesis focused on a single instantiation of a CubeSat. This limited the results to analysis of the engineering system internally. Further investigations of future time instantiations of either the FLARE system or a similar CubeSat project could yield new and interesting results.

There are two research areas that should continue to be pursued with respect to the ES-MDM. First, the expansion of system dependencies and relationships outside of the simplistic binary relationships across multiple-domains needs to be evaluated. The interactions of technical
system components in the digital and physical domain provide an interesting realm to expand ES-MDM research and analysis. The relationships within the social and environmental domain may have positive or negative influences, which could impact the technical system in a variety of ways. The second area of interest within the ES-MDM framework that should be explored further is the introduction of a requirements DSM within the ES-MDM construct. Although the objectives DSM and functions DSM provide the link between the social and functional domains, there is a gap between how objectives drive functions. The impact of stakeholder needs on system objectives, and how the objectives develop into requirements that flow into the technical system should be explored.

Lastly, there are several successful holistic systems engineering modeling frameworks that are widely used within industry. DoDAF 2.0 and SysML have been discussed in this thesis, and although these are holistic frameworks or languages that model a system’s behavior, they do not internally provide the tools to analyze a system construct thoroughly. Recent work by David Kaslow involves the construction of a CubeSat MBSE reference model in SysML. Kaslow and the space systems working group are currently developing software for the integration of commercially available modeling tools and simulation software with their SysML based reference model. (Kaslow et al. 2015) The integration of the ES-MDM as a potential tool into Kaslow’s reference model or the ability to derive a complete ES-MDM construct across all domains from fully constructed SysML or DoDAF 2.0 models is an area worth future exploration.
Works Cited


