3-10-2010

Development of Measures to Assess Product Modularity and Reconfigurability

Amie C. Stryker

DEVELOPMENT OF MEASURES TO ASSESS PRODUCT MODULARITY AND RECONFIGURABILITY

DISSERTATION

Amie C. Stryker, Lieutenant Colonel, USAF

AFIT/DS/ENV/10-M01

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY

AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

The views expressed in this dissertation are those of the author and do not reflect the official policy or position of the United States Air Force, Department of Defense, or the United States Government.

AFIT/DS/ENV/10-M01

Development of Measures to Assess Product Modularity and Reconfigurability

DISSERTATION

Presented to the Faculty

Graduate School of Engineering and Management

Air Force Institute of Technology

Air University

Air Education and Training Command

in Partial Fulfillment of the Requirements for the

Degree of Doctor of Philosophy

Amie C. Stryker, BS, MS

Lieutenant Colonel, USAF

March 2010

AFIT/DS/ENV/10-M01

Development of Measures to Assess Product Modularity and Reconfigurability

Amie C. Stryker, BS, MS
Lieutenant Colonel, USAF

Approved:

David R. Jacques, PhD (Chairman) — 10 MAR 2010 — Date

Adedeji B. Badiru, PhD (Member) — 10 March 2010 — Date

Richard G. Cobb, PhD (Member) — 10 MARCH 2010 — Date

Jonathan T. Black, PhD (Member) — 10 MAR 2010 — Date

Accepted:

Marlin U. Thomas, PhD
Dean, Graduate School of
Engineering and Management — 15 Mar 2010 — Date

Abstract

Several fundamental benefits of modularity are agreed upon by industry including reusability, flexibility, reconfigurability and extensibility. Interfaces within or between modules which establish provide/depend relationships are the focus of current modularity measures. This research outlines a new method and measures for assessing product modularity in terms of degree of coupling and the recognized modularity benefits. A five–step analysis process is developed and used to guide the modularity assessment. Defining and decomposing products are performed first. Using the resultant functional model from the first step, the identified functions are mapped to modules in a product in the second step. In the third and fourth steps, module-to-module interfaces are identified and captured in design structure matrices or a tensor plot. Finally, using results from steps 1–4, the Vector Modularity Measure that includes a reconfigurability measure can be calculated. The measures and analysis process are demonstrated using two precision guided munitions in the United States Air Force inventory. After this demonstration, the research focuses on extending the approach to a modular satellite design problem, namely AFRL's Plug-and-Play Satellite (PnPSat) concept for Operationally Responsive Space. Using the resulting analysis, recommendations to the existing PnPSat design to further increase modularity and its derived benefits are given. Lastly, the modularity analysis process and applications are used to draw conclusions and make recommendations for future research to include identifying factors that influence both modularity and the timeline to perform product assembly and check-out.

Acknowledgements

First and foremost, thank you to my family, both immediate and extended. This journey was made possible by your love and support, and for that I am forever grateful.

Thank you to my advisor, Dr. Jacques, you never gave up on me and supported me throughout this journey.

Thank you to my committee members, your assistance and encouragement was greatly appreciated.

Thank you also to my fellow PhD students, it was great going through this experience with you. Thank you to those that provided valuable information to make this journey a little smoother. And thank you to my fellow runners, it was great to have such a positive outlet when encountering road blocks during the research. Happy trails to you as you continue your running adventures.

Thank you to the USAF for another great assignment. Thank you for allowing me the opportunity to pursue this degree.

Last but not least, thank you Lord for the many blessings in my life and for your guidance along the way.

Amie C. Stryker

Table of Contents

Page

Page

List of Figures

List of Tables

List of Symbols

List of Abbreviations

Development of Measures to Assess Product Modularity and Reconfigurability

I. Introduction

This chapter introduces the dissertation research and its documentation. The motivation for conducting the research is first provided in Section 1.1, followed by a problem statement stemming from the research motivation. Next, the objectives of the research are given followed by a method overview for achieving the stated research objectives. The chapter concludes with an overview of the dissertation document in Section 1.6.

1.1 Research Motivation

The motivation for beginning this research on modularity grew from initial responses from several Department of Defense (DoD) offices that were interested in modularity, standardization, and research efforts that are believed necessary to enable the operationally responsive space (ORS) concept to become a reality. This ORS concept is defined in [41] and broadly in the DoD as assured space power focused on timely satisfaction of Joint Force Commanders' (JFCs) tactical level needs.

In the ORS acquisition construct, increased risk tolerance is acceptable for the potential operational gain that will be realized. Additionally, the ORS acquisition will use streamlined processes to field key capabilities as soon as possible, not waiting for 100 percent solutions, and emphasizing integration of off-the-shelf components where possible.

The U.S. Congress felt strongly enough about the ORS concept that it approved funding and strongly supported the establishment of an official Joint ORS office that stood up in May, 2007 [40]. The Department of Defense [40, 41] stated that it is

Table 1.1: Program Attributes or Themes Affected by Modularity

Attributes Affected by Modularity	
responsiveness to requirements	development timeline
rapid reaction payloads	deployment timeline
rapid reaction buses	innovation
responsive bus development	low cost
responsive payload development	flexibility

committed to improving the nation's means to develop, acquire, field and employ space capabilities in shortened timeframes and in more affordable ways. The ORS report [41] identifies that the overall approach to ORS is to expedite development and fielding of select responsive space systems by leveraging National Security Space (NSS)-wide technology development activities and operational capabilities.

According to a House Armed Services Committee (HASC) report [24], the National Defense Authorization Act for Fiscal Year 2008 reaffirmed support and the need for ORS. This support came in light of a Chinese anti-satellite test and other growing threats to space. The committee provided previous legislation stating that ORS shall consist of low-cost, rapid reaction payloads, buses, spacelift, and launch control capabilities. The committee indicated that it would like to see more support for responsive payload and bus development.

The preceding paragraphs highlight the relevance and current interest level in the ORS concept. In reviewing the preceding paragraphs and the literature, several recurring program attributes or themes were revealed as being areas of desired improvement over existing systems as listed in Table 1.1. Surveying the attributes or themes highlighted in the table and reviewing the literature, one prevalent theme continued to appear that policy makers feel would go a long way in helping to make the ORS concept a reality, and that theme was *modularity*. The last two decades have seen an increase in focus on modularity benefits and modular design methods. The creation of the customizable personal computer in a short time frame or on demand has become the impetus of trying to capture the benefits of both customization and time to get a product to the market, or the warfighter in the case of military products.

This notion that modularity is beneficial depends on the context and purpose behind modularizing a system. System designers must understand when an increase in modularity is desirable from a design standpoint and from a system goal(s) perspective.

Modularity has been studied in various scientific areas. Russell [39] gives a comprehensive though not exhaustive list of all of the disciplines that have used modularity in their discipline evolution including: cognitive processes of the brain [8], computers, computer networks, computer programming languages, theoretical biology, psychology, speech perception, economics, politics, neuroscience, architecture, production, education, orthopedic implants, and robots. While modularity had been studied in the above disciplines, it was not clear whether or not the same benefit can be realized in evolving the ORS construct. Moreover, it was not clear if designing systems that are more modular is warranted despite the push from policy makers.

In line with the responsiveness aspect of the ORS construct, it has been recognized that satellite or mission [3] responsiveness has become important in more than just military situations. It can also be used in disaster monitoring like the tsunami in southeast Asia or hurricane Katrina in New Orleans. Responsiveness can also support science missions; for example, when transient phenomena occur it would provide the ability to get a spacecraft on orbit on short notice to allow observations in space to occur. It would also support the idea of "responsive science [3]" such that a quick turnaround of results from one day's mission can be incorporated into tomorrow's experiments. Responsiveness would support a growing need to bring science and math into classrooms. In a responsive space environment, a student can become involved beginning with the payload manifest and follow it from initial concept to mission completion which is currently a rare occurrence.

The typical timeline for major missions to occur is on the order of a decade or more. Even small satellite missions can take 5-7 years. As a result of a joint analysis team review of the oversight and review process for DoD space systems acquisition, interim guidance was provided in 2009 from the Under Secretary of Defense for Ac-

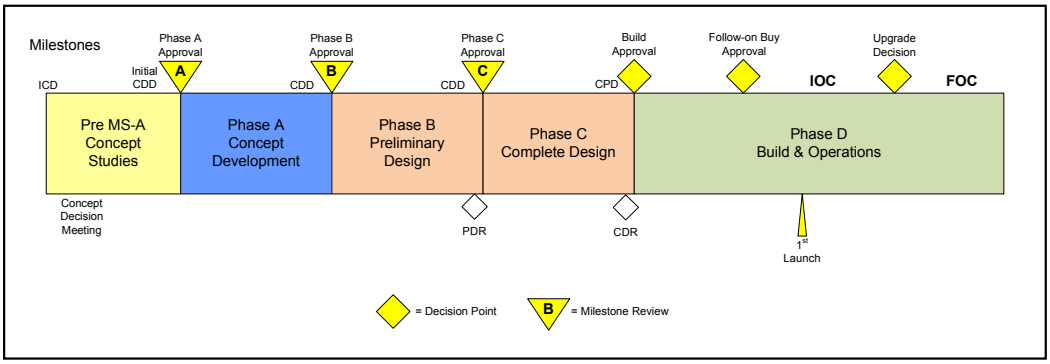


Figure 1.1: DoD Space Systems Acquisition Process – Small Quantity System Model [57]

quisition, Technology, and Logistics (USD/AT&L). The interim guidance states that space acquisition policies from the rescinded National Security Space Acquisition Policy 03-01 will be incorporated into the Department of Defense Instruction 5000.02. Figure 1.1 illustrates the acquisition process of the Small Quantity System Model based on the interim guidance that is to be used for space based systems among others [57]. One of the key developments that will enable the ORS concept and the idea of responsive space in general to progress toward an operational reality is the ability to reduce the overall timeline from mission concept to spacecraft launch and employment. This timeline involves several key events or phases that are shown in Figure 1.2. The last phase shown in the figure is assembly and checkout that prepares the spacecraft for launch and deployment. A typical timeline for assembly and checkout is on the order of months for smaller missions and years for larger missions. The goal is to reduce this timeline to be on the order of days or weeks.

It has been hypothesized, without proof, that one way to accomplish this timeline reduction is through spacecraft designs that incorporate this idea of modularity early in the design process. Current research by Mikkola [34] has shown systems that target customization as a goal are dependent on two factors: 1) the degree of modularity embedded in product architectures; and 2) the extent to which components are standardized. Studying the first factor, degree of modularity, recent research [23] has

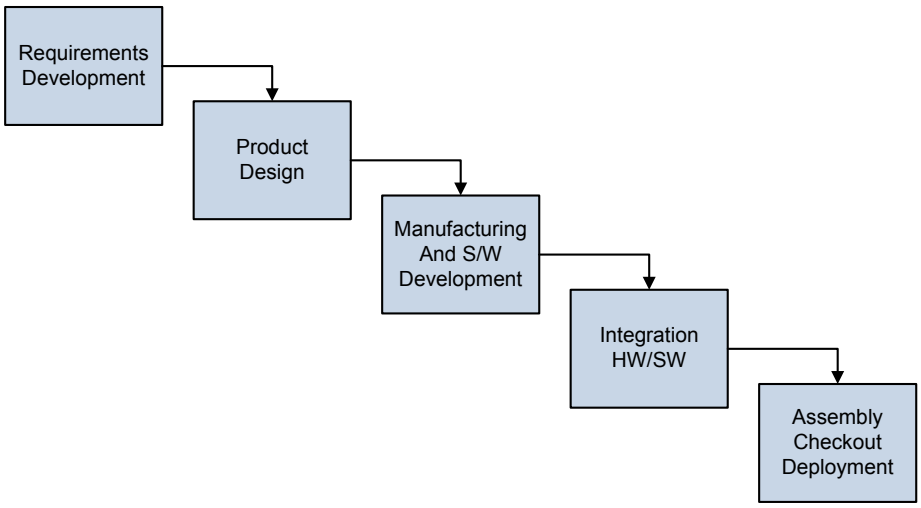


Figure 1.2: Typical Product Development Process - Simplified

begun to compare goals and design decisions of choosing modular architectures versus integral architectures. The research has shown that systems with higher degrees of modularity, tending toward more modular architectures, target goals such as reduced life cycle cost and getting a product to the warfighter or user quicker. This is in contrast to systems with lower degrees of modularity, tending toward more integral architectures, can achieve greater technical performance.

1.2 Problem Statement

It is hypothesized that increasing the modularity of a system will increase the responsiveness of getting a product to the market or the warfighter in a reduced timeframe. In proving/disproving this hypothesis, the following questions need to be answered:

- How is product modularity measured?
- What design influences increase product modularity?
- When is a more modular system, as compared to an an integral system, desirable from a design standpoint and from a system goal(s) perspective?

- What affect do temporal constraints on assembly have on the modularity versus design goals relationship?

1.3 Research Objectives

The intent of this research is to investigate a complete measure of product modularity. The benefits of modularity are investigated in order to better understand how to develop a measure for it. There are three main thrusts of this investigation. First, it is important to identify the benefits being realized by modularizing a product. Only then can a measure be developed to capture product modularity. The second thrust of this research involves using two Air Force precision guided munitions to further refine the modularity measure development and to understand design influences that increase modularity. The third thrust is to extend the use of the modularity measure from the simpler precision guided munition application to a more complex modular satellite example. This third thrust also involves relating modularity to the product assembly and checkout process.

1.4 Method Overview

The overall approach in meeting the research objectives and answering the questions in the problem statement is outlined below and further expounded upon in Chapters 3–6. A five-step analysis process is developed and used to guide the assessment of product modularity. Defining and decomposing products is performed first since the remainder of the research builds upon these definitions and decompositions. Using the resultant functional model from the first step, the identified functions are mapped to modules in a product in the second step of the analysis process. In the third and fourth steps, module-to-module interfaces are identified and captured in a set of design structure matrices (DSMs) or a tensor plot. Finally, using the results from steps 1–4, the Vector Modularity Measure that includes a reconfigurability measure can be calculated. Once the measures are calculated, the results can be used to compare the modularity of alternative product architectures. After illustrating

the research methodology on a simplified munitions example, the research focuses on extending the approach to a modular satellite design problem, namely AFRL's Plug-and-Play Satellite (PnPSat). The resulting analysis is used to recommend changes to the existing PnPSat design to further increase its modularity. The resulting analysis is also used in conjunction with an assembly and checkout analysis of the three applications to begin to characterize the modularity versus temporal constraints relationship. This initial characterization uses the first factor in the VMM, degree of coupling. Lastly, the modularity analysis process and applications are used to draw conclusions and make recommendations for future research in Chapter 7. These recommendations include identifying factors that influence both modularity and the timeline to perform product assembly and check-out.

1.5 Research Contributions

1. A process for accomplishing module identification in conjunction with performing a system decomposition was defined. Module identification is the starting point for the Reconfigurability Measure and Vector Modularity Measure calculations.
2. This research extended previous work to capture the interface types in a layered or tensor approach. The tensor plot graphically provides the designer with feedback to identify predominant interface types.
3. In order to begin assessing the modularity of any product, several key steps must be performed to identify modules and capture associated characteristics of each module. This research developed a process to do this in a manner that is repeatable.
4. This research developed a repeatable measure to assess product reconfigurability recognizing that the measure should account for more than the mathematical number of reconfigurations possible stemming from module options.

5. This research extended the current research on measuring product modularity in the development of the Vector Modularity Measure (VMM). The VMM uses degree of coupling, reusability, reconfigurability, and extensibility factors to assess product modularity.

6. The utility of the RM and VMM were demonstrated using the GBU-24 and the GBU-31 precision guided munitions. This application highlighted reasons why the GBU-24 is more modular than the GBU-31 (e.g. the GBU-24 has less pair-wise constraints than the GBU-31).

7. The applications of the RM and VMM were extended from a simpler PGM example to a more complex example, PnPSat. The analysis process was also applied resulting in recommendations for future design changes to increase the modularity of PnPSat and associated modularity benefits being realized.

8. An initial approach to characterizing the modularity versus temporal constraints relationship was developed and applied to the GBU-24, the GBU-31, and to the PnPSat. This relationship was characterized using the first factor in the VMM, degree of coupling.

1.6 Dissertation Overview

This document follows the scholarly article dissertation format for documenting the research. Using this format, the document is divided into seven chapters and contains two appendices. Chapter 2 presents relevant technical background information on major concepts and techniques used to conduct the research. These major concepts and techniques were used as a foundation for the work covered in Chapters 3–6. Sufficient technical detail is presented such that the fundamental research approach is repeatable and the key contributions are verifiable. The majority of the documentation presented in Chapter 2 was submitted to and published in the 2009 Conference on Systems Engineering Research Proceedings (see reference [47]).

Several factors are used in assessing a product's modularity, one of which is reconfigurability. Chapter 3 provides the development of a measure to assess reconfigurability of modular products. It identifies key concepts that are used in Chapter 4, extending the Reconfigurability Measure (RM) to the overall Vector Modularity Measure (VMM). The RM is demonstrated using two precision guided munitions in use by the United States Air Force. Chapter 3 has been submitted as a research paper to the ASME Journal of Mechanical Design.

Chapter 4 describes the Vector Modularity Measure (VMM) that is used to assess a product's modularity. The VMM uses the RM along with degree of coupling, reusability, and extensibility to assess product modularity. The VMM is demonstrated using the precision guided munitions example from Chapter 3. Chapter 4 has been submitted as a research article to the Taylor & Francis Journal of Engineering Design.

Chapter 5 presents and extends the application of the VMM to a more complex product, AFRL's PnPSat. The VMM analysis process was used to evaluate modularity benefits being realized by PnPSat and recommend future design changes to further increase PnPSat's modularity. Chapter 5 has been submitted as a research article to the AIAA Journal of Spacecraft and Rockets.

Chapter 6 uses the PGM applications to develop an approach to characterizing the modularity versus temporal constraints relationship. The temporal constraints referred to in this chapter are associated with a product assembly and checkout process timeline. This approach is then used to analyze the PnPSat modularity versus assembly and checkout process relationship. This relationship focuses on the degree of coupling factor in the VMM. Lastly, emerging trends from using all three applications are stated as preliminary findings.

Chapter 7 concludes the main document by providing an overall summary of the research findings, a summary of key contributions, and recommendations for future research. This is followed by two appendices that provide a glossary of key terms and some of the MATLAB® code used to support the research.

II. Background

This chapter presents pertinent technical background information on major concepts and techniques used to conduct the research. The benefits of modularity along with key terms associated with modularity are given first as they introduce the foundation of the research. Tools and techniques for performing system decomposition are described in the first section of this chapter. System decomposition is pervasive to all of the modularity measures studied in the literature survey as well as in the research documented in this dissertation. Summaries of each of the current modularity measures are given next. Lastly, a few concepts that are also used in the literature but not as pervasively as system decomposition are given as they are fundamental to this research.

Before diving into the specific techniques and methodologies from the literature survey, the recognized benefits of modularizing a product are reviewed so they can be balanced and used when making design decisions. According to a 2005 Space Power Journal article [28], Lee points out the benefits of using components and standards for space systems, and some of these benefits are also applicable to modular systems in general. Gershenson et al. [16] list and summarize the benefits of modularity as cited from numerous previous works. These benefits are summarized in Table 2.1. As Gershenson et al. point out, while many benefits of modularity are widely accepted, there has been only little anecdotal evidence and scientific proof of these benefits. The proof that has been shown has been on simple products and there is even less of a degree of certainty that these benefits will carry over to more complex products. Product complexity, as used in this research, is defined similarly as in [36] as having three main elements: 1. number of modules; 2. number of interfaces between modules (degree of coupling); and 3. degree of product novelty. Increasing any of the three elements corresponds to an increase in product complexity.

After reviewing a few related articles on modularity measures, it became readily apparent that a few definitions need to be established based on some terms that have

Table 2.1: Modularity Benefits (summarized from [16])

Category	Benefits
Product Functionality	Reconfiguration; customization; modularity allows for flexible designs that can respond to changes in functional requirements over the life of a product
Product Development	Dividing tasks for parallel development; re-use of existing designs; economies of scale; increased feasibility of component/product change; increased product variety; decoupling risk
Production	Streamlined suppliers; reduced inventory, fewer works in progress, faster process time; learning curve effect; parts and material price breaks
Other	Decreased life-cycle costs; maintenance fault analysis; recycling, re-use, and disposal

been used thus far as well as additional terms that will be introduced. Gershenson et al. [16] surveyed the literature for a common definition or agreed upon definition of modularity among many disciplines. What they found was that there was some consensus in some areas but few definitions of modularity captured product modularity. Gershenson et al. came up with three fundamental elements to modularity: the independence of a module's components from external components, the similarity of components in a module with respect to their life-cycle processes, and the absence of similarities to external components. They consider modularity to be a relative property and so systems or products can have higher or lower degrees of modularity. They also clarify that products with higher degrees of modularity are either considered to have a higher percentage of components that are modular or contain components that are more modular on average. An even more general definition for modularity along with some additional definitions that are used in this research are given below[1]:

[1]http://en.wiktionary.org/wiki/Wiktionary:Main_Page and http://www.wikipedia.org were used as guide markers in finalizing the definitions except as indicated where the exact definition was used.

- **Modularity** – grouping of components into well defined entities, such as modules or sub-assemblies, that can be further described by the interfaces between them.
- **Interface (I/F)** – spatial, informational, material, energy, or structural connection or coupling of one module to another module within a product [42]. I/F types given below are defined similarly as in [42].
  - **Spatial I/F** – physical adjacency for alignment, orientation, serviceability, assembly or weight.
  - **Informational I/F** – transference of signals or controls.
  - **Material I/F** – transference of airflow, oil, fuel, or water.
  - **Energy I/F** – transference of heat, vibration, electric, or noise energy.
  - **Structural I/F** – transference of loads or containment.
- **Modular System or Product:** – a system or product composed of modules according to a particular system architecture.
- **System Architecture:** – the conceptual design that defines the structure and/or behavior of a system[2]; it represents the functional, physical and/or operational architectures of the system [7].
- **Modular:** – an entity containing one or more modules that can be arranged in a flexible way such that the modules are interchangeable.
- **Module:** – group of components or sub-assemblies that perform one or more functions; a module has at least one interface with other modules within a system or subsystem.
- **Component:** – smaller, self-contained part of a larger entity[3]; components have at least one interface with another component or module.

[2]http://en.wikipedia.org/wiki/System_architecture
[3]http://en.wiktionary.org/wiki/component

- **Modular Component:** – components that can be interchanged with other components in a module to achieve the functions of the module.
- **Function** – technical process involving energy, material and/or signals being converted and/or channeled.
- **Flow** – material, signal, and/or energy that can be converted or channeled.

Additional definitions will be included as they are introduced and are also included in Appendix A for reference.

2.1 System Decomposition

2.1.1 Functional Basis Language. One of the foundations of this work is decomposing systems in order to analyze their degree of modularity and how it relates to responsiveness. Two system decompositions are performed, one at the functional level; and one at the module level that resulted from the functional level decomposition. The latter is captured using design structure matrices (DSMs) described in Section 2.1.3 and follows Suh's axiomatic design principles described in Section 2.6 [54].

Hirtz et al. [21] extended the previous work by [38] to create a functional basis vocabulary. This vocabulary defines a standardized language to decompose a system into functions and flows to a level of abstraction needed for a given analysis. Three levels of abstraction are used to describe the decomposition: class (or primary), secondary, and tertiary. Functional decomposition is not new. Functional decomposition is typically done in the early stages of design conceptualization, transforming user requirements into functional requirements that result in a functional model [11, 43, 54]. This functional decomposition or modeling provides an abstract method for understanding and representing the overarching function of a product [21]. Functional decomposition begins at the top level outlining the overarching function of a product. This overarching function is then decomposed into the three levels of abstraction listed above that are used in the zigzagging technique given in [54] and described in

Section 2.6. For purposes of this analysis, the functional decomposition abstraction level stops at the class level.

2.1.2 Module Identification. After functional decomposition has been accomplished at the class level, a product's components and/or modules can be mapped to their corresponding function(s). For existing products, one method to identify module boundaries is to use reverse-engineering. Even though the product exists, clear boundaries may not present themselves, requiring iterations of function-to-module mappings until the boundaries are clearly defined. For new products, identifying the module boundaries also will likely require several iterations of these mappings. Another technique to identify modules is to use the dominant flow heuristic developed by Hirtz *et al.* ([44–46]). This heuristic groups components performing similar functions into modules.

2.1.3 Design Structure Matrix. One use of the design structure matrix (DSM) is to capture results of performing a system decomposition. Browning [6] points out that the DSM is a square matrix with identical rows and columns that displays relationships between components of a system in a compact, visual, and analytically advantageous format. The row and column headings capture the modules identified in the system decomposition. As shown in Figure 2.1, the DSM matrix also shows the dependency and provider relationship between elements where an element can be a module, a component, activity, parameter, etc. When viewing the rows and columns of the example matrix (as adapted from [6]), specifically at the Element 3 row, Element 3 *depends* on information *from* Elements 1, 5, 6, 9, and 10. Similarly, reading the column of Element 3 shows that it *provides* something *to* Elements 8 and 9. DSMs are one of the tools used to model the module-to-module interfaces in this research.

Another example of using DSMs, with a similar presentation, is given by Hölttä et al. [22] where it is used to represent the internal connectivity structure of a product. Hölttä et al. give examples of what these DSMs would look like for the extremes of

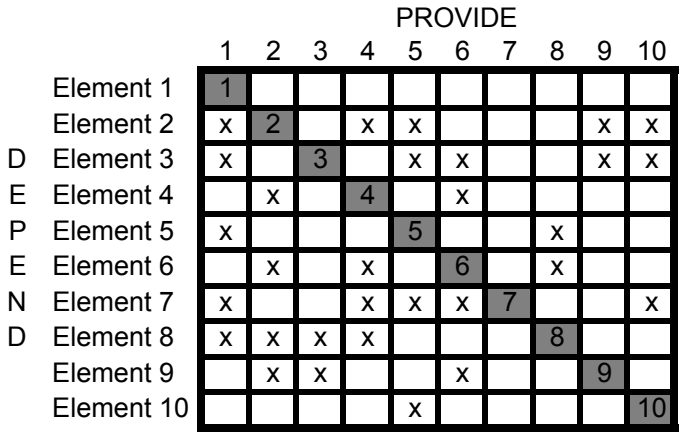


Figure 2.1: Example DSM showing dependencies

a fully integral system and a fully modular system of components. The examples shown in Equation 2.1 are adapted from the examples given by Hölttä et al.

$$\mathrm{DSM}_{\mathrm{integral}} = \begin{bmatrix} 0 & 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 & 1 \\ 1 & 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 & 0 \end{bmatrix} \qquad \mathrm{DSM}_{\mathrm{modular}} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix} \tag{2.1}$$

2.2 Modularity Measures

A survey of the literature revealed several methods to quantify the degree of modularity of systems. Each of the methods tackle the concept of measuring the degree of modularity of a system, a module, or a component in slightly different ways. In this section, the various methods to quantify the degree of modularity is reviewed individually. Very little work was found on comparing modularity versus performance of systems. That work will be summarized in this section as well.

2.2.1 Gershenson, Guo, Prasad and Zhang [17, 19, 20]. Gershenson et al. [17, 19] provide a good summary of the work that has been done in the literature regarding product modularity measures and design methods to achieve modularity in product design. Gershenson et al. looked qualitatively in 2002 [17] and quantitatively in 2003 [19] at works in various fields in addition to engineering to include computer science, biology, architecture and art. They also reviewed their own modularity measure [15, 17] that they developed in 1999 where they considered modularity to be a relative property, which is commonly done. Their measure was the ratio of intra-module similarities to all similarities, both intra- and inter-module, which is added to the ratio of intra-module dependencies to all dependencies, both intra- and inter-module as seen in Equation 2.2.

$$Modularity \;=\; \frac{intra_{sim}}{all_{sim}} + \frac{intra_{dep}}{all_{dep}} \tag{2.2}$$

where :

$$all_x = intra_x + inter_x$$
$$x \in \{sim\ (similarities),\ dep\ (dependencies)\}$$

The similarities that Gershenson et al. considered were component-process similarities whereas the dependencies considered were both the component-process and the component-component dependencies. Their conclusion after reviewing the literature was that the measures and methods varied widely in both purpose and process and that some were highly quantitative in nature whereas others were highly qualitative in nature. Ultimately, their review of the literature showed no clear consensus other than those found in defining modularity. They did find that most of the measures centered on measuring *dependencies* with components external to modules but there was always some subjectivity in the measures. They used this finding in their quantitative analysis when comparing the various modularity measure methods. They used cluster analysis to recommend one of the 9 measures as an industry standard

for other researchers and design engineers to adopt. At this time, no such standard exists. Finally, they noted that the measures and methods lacked rigorous verification and validation. Gershenson et al. reviewed works for the previous 30 years, and that historical review is not repeated here. The background literature survey in this section focuses on work done since their article appeared in 2003.

2.2.2 Mikkola [34]. The first of the modularity measures that are reviewed in this section were initially developed by Mikkola in 2003 and further developed in 2006 [34]. They include degree of modularity, degree of mass customization, and degree of product variety. These relations taken together make up the modularity function (MF) which "is a mathematical model that measures the degree of modularization embedded in a given product architecture by taking into account the number of components, interfaces, degree of coupling, and substitutability." Mikkola noted that modularity is a key enabler for mass customization. She also noted that the extent of mass customization is dependent on two factors: 1) the degree of modularity embedded in product architectures; and 2) the extent to which components are standardized. With this, Mikkola also suggests that the degree of modularity embedded in product architectures is related to the tradeoffs between the amount of standardization and uniqueness of components. Mikkola goes on to categorize the degree of customization into four component strategies:

1. standard - noncustomizable components
2. standard - customizable components
3. unique - noncustomizable components
4. unique - customizable components

These categories lead to a customization strategy spectrum shown in Figure 2.2, adapted from [34]. When looking at Figure 2.2, traditional spacecraft fall into the right-side of the spectrum. The ORS concept is trying to move this towards the middle of the spectrum through higher component standardization and an increase

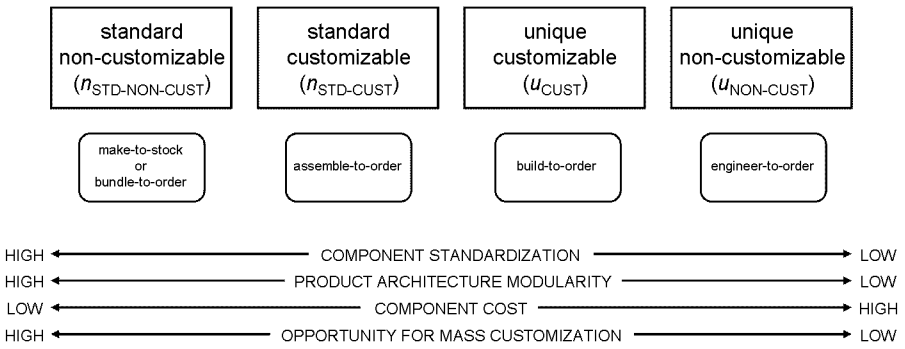


Figure 2.2: Customization Strategy Spectrum

in product architecture modularity. According to the model, this will promote lower component costs and it is believed will result in an increased opportunity for mass customization of spacecraft.

This leads to the modularity function that measures the degree of modularization embedded in product architectures. The terms used in the preceding explanations, along with a few more key factors, define the degree of modularity, $M(u)$, developed by Mikkola.

$$M(u) = e^{-u^2/2Ns\delta} \tag{2.3}$$

In Equation 2.3, the degree of modularity is defined with respect to the number of unique components (u) embedded in a given product architecture where N is the total number of standard and unique components, $N = n_{STD} + u$, δ is the average degree of coupling between components and between modules; and s is the substitutability factor of the unique components into other products or systems. Interfaces k and degree of coupling δ are related and approximated using the relationship $\delta \sim (k/n)$ where k is the number of interfaces for a module or subsystem and n is the number of components or modules in a module or subsystem, respectively. This relationship is an approximation of the average number of interfaces per component or module, respectively. The substitutability factor, s, which is the substitutability of unique components in the product architecture, is estimated as the total number of families in which the unique components are used (in addition to the product being analyzed)

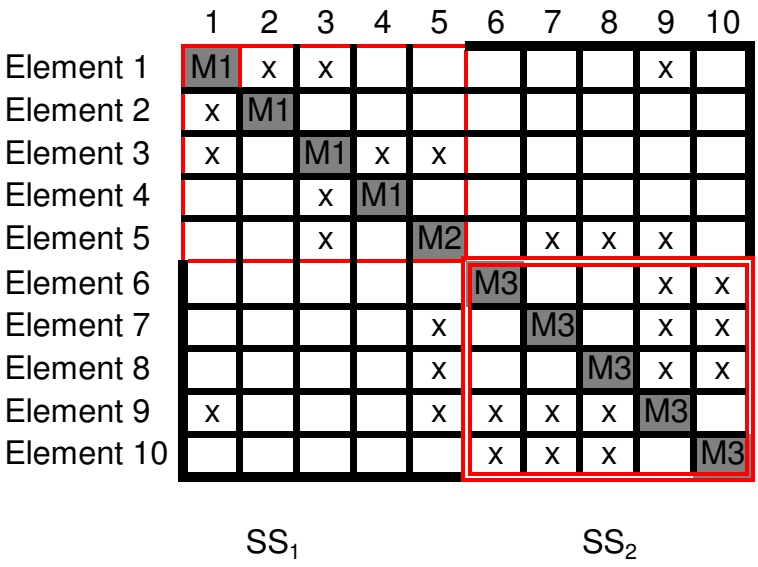


Figure 2.3: Example system DSM for System 1 (S1) showing 2 subsystems, 3 modules, and 10 components along with their various relationships

divided by the number of interfaces required for functionality, $s \sim n_{PA}/k_A$. A couple of observations or insights can be made at this point in reference to the setup and Equation 2.3. The lower the number of unique components the higher the degree of modularization. A perfectly modular product architecture results in $M(u) = 1.0$ and has zero unique components. Unique components that are used across product families have a higher substitutability factor and hence result in a higher degree of modularity. Mikkola goes on to outline an algorithm for measuring $M(u)$ which is illustrated here using a similar system example adapted from [34].

The degree of modularity measure begins with defining the product architecture through decomposition of the system into subsystems, modules, and components. For this example calculation, the DSM in Figure 2.1 is used but is modified as a symmetric DSM. The DSM represents an example system, System 1 (S1), which is decomposed into 2 subsystems, SS_1 and SS_2, respectively. Subsystem 1 is decomposed into Module 1 (M1) and Module 2 (M2), which are in turn decomposed into Components 1 through 4 and Component 5, respectively as shown in Figure 2.3. By using the DSM, Mikkola's second step of the algorithm is already accomplished, that is identifying the interfaces between components and modules, both inter- and intra-. In setting up

Table 2.2: Degree of Coupling for S1, Subsystems, and Modules

Module	$\delta_{modules}$	Subsystem	$\delta_{subsystem}$	δ
M1	0.75	SS_1	0.5	-
M2	0	SS_2	0	-
M3	1.2	-	-	-
M_{avg}	0.65	SS_{avg}	0.250	0.9

the DSM, the components and modules should also be identified as being unique or standard. In our example, M1 and its components are considered unique such that 4 out of the 10 components are unique. The next step is to assess the substitutability factor of the unique components (or modules) by counting the number of product families enabled by the component, divided by the number of interfaces required by the component for functionality. For S1, if M1 can also be used in 2 other systems, then the substitutability factor, s, is 2/2 or 1 component per interface. The next step is to compute the degree of coupling, δ, for each module, and then compute the average degree of coupling per module. This is followed by computing the degree of coupling for each subsystem which is then used to find the representative value of δ for S1. The δ for S1 is simply the sum of the average δs for the modules and subsystems. The calculated values are shown in Table 2.2. Using the calculated δ along with $s = 1.0$, N= 10, and $u = 4$, yields $M(u) = M(4) = 0.41$ which can be seen in the plot of M(u) versus u in Figure 2.4.

It is important to note that this measure is influenced by the way in which components and modules are assembled, hence the measure is dependent on the particular product architecture and its boundaries. Also looking at the modularity function equation itself again (Equation 2.3), for a given product architecture we can only change the number of unique components and the substitutability factor and observe the influence of the two on the inherent modularity of a given product. This can be viewed graphically, similar to Figure 2.4, as M(u) versus u where $u \in \{0,\ 1,\ ,..., N\}$ and each set of data points refers to a particular substitutability factor, s. For the definition and setup of the modularity function measure developed

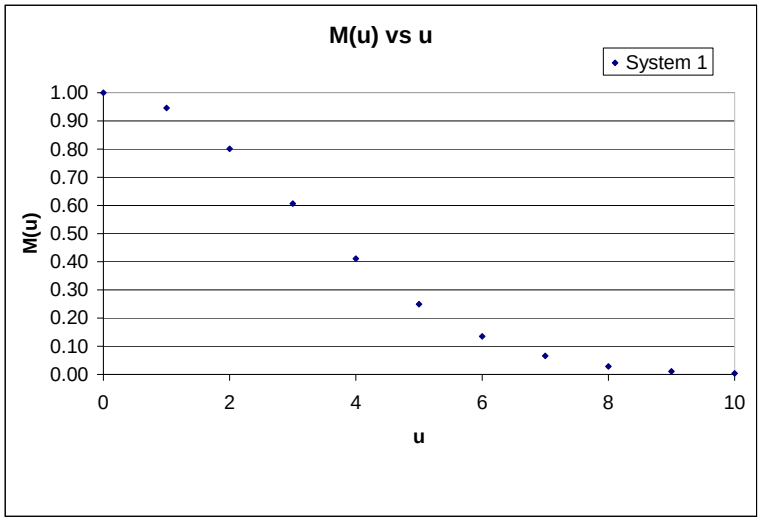


Figure 2.4: M(u) versus u

by Mikkola [34], in order to increase the degree of modularity unique components should be incorporated into as many other systems as possible (increasing s), as few unique components as possible should be used (minimizing u), or a combination of the two techniques. The author notes that another possibility of increasing degree of modularity is by increasing degree of coupling, δ, which is mathematically a correct statement using Equation 2.3. However, this goes against the prevailing principle in systems engineering that lower the degree of coupling increases a products modularity.

Mikkola [34] goes on to relate the degree of modularity embedded in product architectures with mass customization, $MC[M(u)]$ and is given in Equations 2.4 and 2.5. This relation includes the degree of modularity $M(u)$ and a new factor m that is outlined in Equation 2.5. The new factor m "is an indicator of the degree of product

$$MC[M(u)] = me^{-u^2/2Ns\delta} \qquad 0.0 \leq m \leq 1.0 \tag{2.4}$$

$$m = \frac{k_1 n_{STD-NON-CUST} + k_2 n_{STD-CUST} + k_3 u_{NON-CUST} + k_4 u_{CUST}}{N} \tag{2.5}$$

where :

$$0.0 \leq k_1,\ k_2,\ k_3,\ k_4 \leq 1.0$$

$$N = n_{STD-NON-CUST} + n_{STD-CUST} + u_{NON-CUST} + u_{CUST}$$

$$k_1 = \frac{n_{STD-NON-CUST}}{\sum n_{STD-NON-CUST} \text{ (from aggregate MPS)}}$$

$$k_2 = \frac{n_{STD-CUST}}{\sum n_{STD-CUST} \text{ (from aggregate MPS)}}$$

$$k_3 = \frac{u_{NON-CUST}}{\sum u_{NON-CUST} \text{ (from aggregate MPS)}}$$

$$k_4 = \frac{u_{CUST}}{\sum u_{CUST} \text{ (from aggregate MPS)}}$$

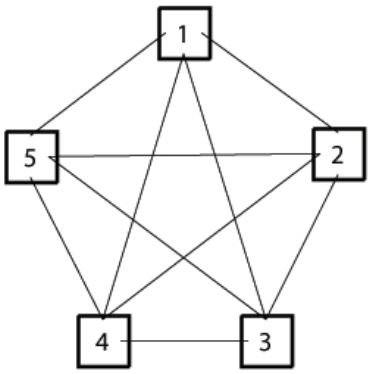
variety present in a given product architecture, which is reflected in the number of components that are used for creating product variety where $k_1,\ k_2,\ k_3$, and k_4 are contribution percentages per component type that is used in all production lines, which can be obtained from the Bill of Materials (BOM) and the Master Production Schedule (MPS). The BOM lists the quantity of all the components used in a given product, including respective types and prices. The MPS lists the volume of components needed in production to satisfy demand. [34]" It is unclear at this point whether or not this relationship, $MC[M(u)]$, will be useful but is included for completeness from the literature survey. This relationship will be revisited as the research effort progresses. The degree of modularity measure, $M(u)$ is relevant and will be used as the research progresses.

2.2.3 Hölttä-Otto, de Weck and Suh [22, 23]. Hölttä-Otto et al. created a new measure of modularity in 2005 [22] and further developed it in 2007 [23] that combines two metrics. The two metrics are the non-zero fraction (NZF) and the singular value modularity index (SMI). The NZF evaluates the sparsity of the interrelationships between components and assumes values between zero and one. The SMI also assumes a value between zero and one but looks at the degree of internal coupling. This new common measure was then compared to the common measure created by Gershenson and Guo [20].

The SMI is the measure that was developed in the earlier work [22] which is an unambiguous way to quantify the degree of modularity of a product based on its internal connectivity structure. This internal connectivity structure is represented by a design structure matrix (DSM), specifically a binary DSM that was chosen for the sake of simplicity. The DSM was introduced in Section 2.1. Hölttä-Otto et al. [23] use the binary DSM such that the diagonal entries are zeros and the off-diagonal elements are set to one or zero if two components have a connection or not, respectively. Again the binary DSM was used, but according to Hölttä-Otto et al., a non-binary DSM can also be used which enables differentiation between connections of different strengths. The DSM examples used in Section 2.1 are shown again in Figure 2.5. Also shown in Figure 2.5 are example integral and modular systems in Figures 2.5(a) and 2.5(b) respectively that were adapted from [22] that the DSMs would represent.

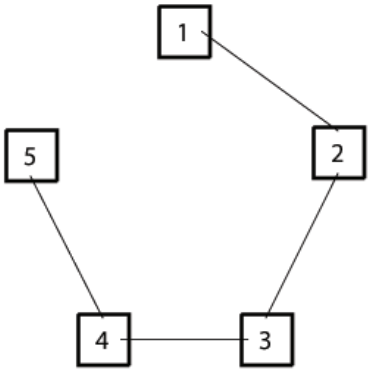
Figure 2.5(a) shows that every component connects to every other component in the integral system, whereas for the modular system in Figure 2.5(b), each component only connects with its immediate neighbor(s). Both systems have 5 components. The total number of components in the system is referred to as N, and in our example $N = 5$.

A common technique for analyzing multivariate data is the Singular Value Decomposition (SVD) technique [58]. The SVD can be performed on the DSM matrices to reveal the singular values. The values for the example system, where $N = 5$, were computed using MATLAB and are plotted in Figure 2.6. The singular values are σ_i through σ_N and are produced in descending order. Using σ_1, the singular values were normalized through σ_i/σ_1. Hölttä-Otto et al. note, as can be noted with our example in Figure 2.6, that there is a significant difference in the plots for integral versus modular systems. The modular system shows a more gradual decay of its singular values. Hölttä-Otto et al. explain the reason for this; the information that describes the system is more broadly distributed in modular versus integral architectures. More importantly, the authors go on to explain that a system with a higher decay rate of singular values can more easily be reduced to a smaller set without much loss of infor-



$$\mathrm{DSM_a} = \begin{bmatrix} 0 & 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 & 1 \\ 1 & 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 & 0 \end{bmatrix}$$

(a) Integral System



$$\mathrm{DSM_b} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

(b) Modular System

Figure 2.5: Product structures and their associated binary DSMs for a (a) fully integral system, and (b) fully modular system

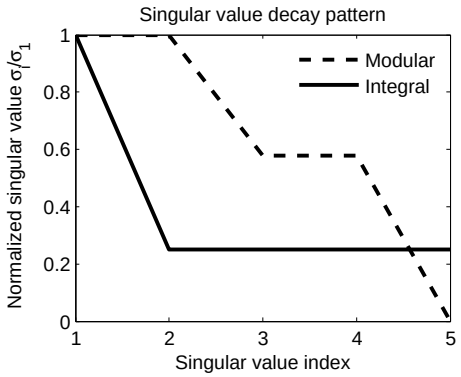


Figure 2.6: Singular value decay pattern

mation. So for an integral system, it can be described by focusing on the components that are the most connected. Conversely for a modular system, information from a wider set of components is required to describe the system.

Hölttä-Otto et al. [23] postulate that the modularity index reflects the degree to which the important information for describing system connectivity is concentrated in a few components that are highly connected across the system. The SMI measures the decay rate of the sorted, normalized singular values in the system and is shown in Equation 2.6.

$$\text{SMI} = \frac{1}{N} \arg\min_{\alpha} \sum_{i=1}^{N} \left| \frac{\sigma_i}{\sigma_1} - e^{-[i-1]/\alpha} \right| \tag{2.6}$$

In Equation 2.6, developed by Hölttä-Otto et al., it is assumed for the sake of comparison that singular values in all systems decay exponentially according to $exp(-(i-1)/\alpha)$. Equation 2.6 can be rewritten in the form shown in Equation 2.7 such that α^* is equal to the α from Equation 2.6 that minimizes the sum shown on the right hand side of the equation.

$$SMI = \frac{\alpha^*}{N} \tag{2.7}$$

Continuing with our example problem, "fminsearch" was used in MATLAB to find the α that minimized the sum in Equation 2.6. This α, denoted as α^*, was then used to calculate the SMI using Equation 2.7. The results are tabulated in Table 2.3.

Table 2.3: SMI and NZF Values for the Example System, N=5

System	N	α^*	SMI	NZF
Integral	5	1.44	0.29	1.0
Modular	5	3.64	0.73	0.4

Hölttä-Otto et al. outline several fundamental characteristics of the SMI to include:

- $0 \leq \text{SMI} \leq 1$
- An SMI closer to one indicates a more modular system whereas an SMI closer to zero indicates a more integral system.
- The SMI is independent of subjectively drawn boundary lines between modules which is not the case in the Modularity Function developed by Mikkola [34].
- The SMI is scale free and can be computed for systems of varying sizes.

The second metric used and developed by Hölttä-Otto et al. is the non-zero fraction (NZF). The NZF is the fraction of non-zero entries in the DSM to the total number of entries, after subtracting out the main diagonal. The NZF is computed using Equation 2.8.

$$\text{NZF} = \frac{\sum_{i=1}^{N} \sum_{j=1}^{N} DSMij_{i \neq j}}{N(N-1)} \tag{2.8}$$

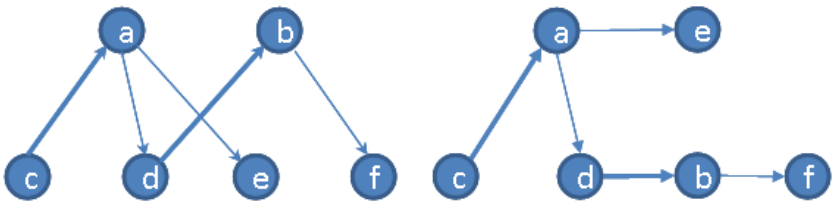
The NZF is useful in that it indicates the degree of connectedness or the degree of sparsity of the underlying DSM. This is different than the SMI in that the SMI measures the degree of modularity of a system. In our example, the integral system has an NZF = 1 which indicates that each component is connected to every other component. The example modular system has an NZF = 0.4 which is shown in Table 2.3.

2.2.4 Sosa, Eppinger, and Rowles [42]. Sosa et al. [42] developed a modularity measure in 2007 by further developing and combining several existing measures using graph theory. A few basics regarding graph theory are presented in Section 2.5. Sosa et al. break the modularity measures into three groups of measures including degree modularity, distance modularity, and bridge modularity. These groups of

measures are based on the notion of centrality. The degree modularity measure represents how components share direct interfaces with adjacent components. The distance modularity measure represents how design interfaces may propagate to nonadjacent components. Finally, the bridge modularity measure represents how components may act as bridges among other components through their interfaces. Sosa et al. consider complex products as a network of components that share technical interfaces in order to function as a whole system. In defining component modularity, they look at the lack of interfaces or connectivity among the components. These connections represent design dependencies.

Sosa et al. [42] point out that modularity can be measured at varying levels such as at the product level, the system or subsystem level and at the component level. The latter level is the focus of Sosa et al. who define component modularity as the level of independence of a component from the other components within a product, where the product is made up of systems or subsystems which in turn are made up of components. A component is considered more modular if it is more independent as indicated by its number of connections or its disconnectedness. Conversely, the more connected or dependent a component is with other components, the less modular it is. In trying to measure modularity, Sosa et al. aim to look at the patterns of a component's connections or design dependencies with the other components in the product.

To illustrate the three groups of modularity measures, from Sosa et al., the example bipartite graph from Figure 2.10 in Section 2.5 will be used here with a few changes. The bipartite graph vertices are grouped back into a single group of nodes instead of having them split into variables and relations, thus removing the "bi-" part of the graph. The nodes are then renumbered (re-lettered) as a, b, c, d, e, and f. The graph still has the same edges and same directional markings with the addition of two edges that are made more prominent than the others thus showing a "stronger" connection between the two nodes (between c and a, and between d and b). The stronger connections (design dependencies) occur when they are considered



(a) Example bipartite graph from Chapter 1

(b) Example bipartite graph rearranged into a digraph

Figure 2.7: Example graph transformed from the bipartite graph in Figure 2.10

		PROVIDE					
		a	b	c	d	e	f
D	a	*		2			
E	b		*		2		
P	c			*			
E	d	1			*		
N	e	1				*	
D	f		1				*

Figure 2.8: DSM matrix for graph in Figure 2.7

to be of higher importance or influence as subjectively assessed by the designers, users or stakeholders. The last change is to simply rearrange the nodes pictorially for ease of directly seeing the components and their connections. These changes are shown in Figure 2.7. The graphs in Figure 2.7 represent a product that is broken down into components which also show their design dependencies among and between the components. The graph in 2.7(b) is used to define the network of components that will be analyzed over the next few paragraphs.

After breaking down the product, Sosa et al. represent products using the DSM (refer to Section 2.1.3) which is shown in Figure 2.8 for our example system. For the DSM, any edge between two nodes will take on a nonzero value. In this example, adapted from Sosa et al., each edge receives a value of one unless a "strong" dependency exists, noted by the prominent directional arrows, which is then represented in the matrix entry as a 2.

Having developed the DSM, the first group of modularity measures can be calculated: degree modularity. To do this, Sosa et al. use a normalized expression

for component modularity using a ratio shown in Equation 2.9 of actual component disconnectivity to the maximum possible component disconnectivity.

$$\text{ComponentModularity} = \frac{\text{Actual component disconnectivity}}{\text{Maximum possible component disconnectivity}} \tag{2.9}$$

The idea of disconnectivity is based on the concept of independence and how independent a component is with respect to design dependencies. Having setup the normalized measure for component modularity, the first of the degree modularity measures, the in-degree of modularity, $M(\text{ID})_i$ can be calculated using Equation 2.10. This equation translates mathematically into Equation 2.11 where $x_{i+} = \sum_{j=1,j\neq i}^{n} x_{ij}$ and x_{max} is the maximum value that x_{ij} can take. The in-degree of a component is a measure of the number of other components that it depends on for functionality.

$$\begin{aligned} M(\text{ID})_i &= \frac{\text{Actual indegree disconnectivity}}{\text{Max. indegree disconnectivity}} \\ &= \frac{\text{Max. indegree disconnect.} - \text{Actual indegree connect.}}{\text{Max. indegree disconnectivity}} \end{aligned} \tag{2.10}$$

$$\begin{aligned} M(\text{ID})_i &= \frac{x_{max} \cdot (n-1) \; - \; x_{i+}}{x_{max} \cdot (n-1)} \\ &= 1 \; - \; \frac{x_{i+}}{x_{max} \cdot (n-1)} \end{aligned} \tag{2.11}$$

The out-degree modularity of component i is calculated similarly to the in-degree using Equations 2.12 and 2.13 where $x_{+i} = \sum_{j=1,j\neq i}^{n} x_{ji}$.

$$M(\text{OD})_i = \frac{\text{Actual outdegree disconnectivity}}{\text{Max. outdegree disconnectivity}} \tag{2.12}$$

$$M(\text{OD})_i = 1 \; - \; \frac{x_{+i}}{x_{max} \cdot (n-1)} \tag{2.13}$$

Sosa et al. point out a couple of notes about degree modularity including:

- $0 \leq M(\text{ID}),\ M(\text{OD}) \leq 1$
- Maximum degree modularity occurs when a component is not connected (has no design dependencies) to any other component in the product.
- Minimum degree modularity reflects *strong* design dependencies (connections) between a component and all other components of the product, in other words the component would be considered highly integral.
- If there are no design dependencies, the component is considered disconnected for the given direction, in- or out-.

The values, x_{i+}, $M(\text{ID})_i$, x_{+i}, and $M(\text{OD})_i$, for the example problem that were calculated for each component i are shown in Table 2.4.

Table 2.4: Degree, Distance, and Bridge Modularity for the Components in the Example System, *n=6*

Component i	x_{i+}	$M(ID)_i$	x_{+i}	$M(OD)_i$	$M(IT)_i$	$M(OT)_i$	$\sum_{i\neq a, i\neq b, a\neq b} \frac{nd_{ab}(i)}{nd_{ab}}$	$M(B)_i$
a	2	0.8	2	0.8	0.03	0.23	4	0.8
b	2	0.8	1	0.9	0.2	0.03	3	0.85
c	0	1.0	2	0.8	1	0.4	0	1.0
d	1	0.9	2	0.8	0.1	0.1	4	0.85
e	1	0.9	0	1.0	0.1	1	0	1.0
f	1	0.9	0	1.0	0.33	1	0	1.0

The next group of modularity measures is the distance modularity. Degree modularity captures the relationship between components when they are directly linked to other components. Sosa et al. [42] use distance modularity to capture the relationship between components when they are not directly linked. Sosa et al. point out that the measure of modularity of a component should also take into account how "distant" a component dependency is from another component. This concept brings in the notion of centrality that the authors are trying to capture and characterize. The first measure of distance modularity is in-distance modularity, $M(\text{IT})_i$, and is calculated using Equations 2.14 and 2.15. The authors define distance modularity as being proportional to the sum of the geodesics of component i with all other components in a product. It is the direction and not the strength of design

dependencies that defines distance modularity.

$$M(\text{IT})_i = \frac{\text{Actual indistance disconnectivity}}{\text{Max. indistance disconnectivity}} \tag{2.14}$$

$$M(\text{IT})_i = \frac{\sum_{j=1, j\neq i}^{n} d(i,j)}{n(n-1)} \tag{2.15}$$

In Equation 2.15, $d(i,j)$ represents the geodesic distance of the design dependency between components i and j. If multiple paths exist, the shortest path is used.

Similarly, the out-distance modularity is calculated using Equations 2.16 and 2.17. The difference is in the path direction where $d(j,i)$ is used. The calculated values, $M(\text{IT})_i$ and $M(\text{OT})_i$, for component i for our example problem are shown in Table 2.4.

$$M(\text{OT})_i = \frac{\text{Actual outdistance disconnectivity}}{\text{Max. outdistance disconnectivity}} \tag{2.16}$$

$$M(\text{OT})_i = \frac{\sum_{j=1, j\neq i}^{n} d(j,i)}{n(n-1)} \tag{2.17}$$

As with degree modularity, Sosa et al. point out a few notes about distance modularity:

- High values of $M(\text{IT})_i$ or $M(\text{OT})_i$ indicate that component i is far from the other components and hence is more modular.
- Disconnected components have a distance modularity of 1 since it is assumed that a disconnected component is n steps away from all other components in the product.
- The minimum distance modularity is 1/n, which happens when a component is adjacent to all other components and hence is completely integral.

This leads to the final grouping of modularity measures, bridge modularity denoted by $M(\text{B})_i$. Bridge modularity looks at component modularity in yet another

way. Bridge modularity looks at the relationships between components that are not directly adjacent to each other. It looks at the components that "bridge" the design dependencies between two end components in a design dependency path. Bridge modularity is calculated using the Equations 2.18 and 2.19.

$$M(\mathrm{B})_i = \frac{\text{Actual bridge disconnectivity}}{\text{Max. bridge disconnectivity}} \tag{2.18}$$

$$M(\mathrm{B})_i = 1 - \frac{\sum\limits_{i \neq a, i \neq b, a \neq b} \frac{nd_{ab}(i)}{nd_{ab}}}{[(n-1)(n-2)]} \tag{2.19}$$

The quantity $\sum\limits_{i \neq a, i \neq b, a \neq b} \frac{nd_{ab}(i)}{nd_{ab}}$ is the sum of the ratios of all geodesics between components a and b that contain component i to the total number of geodesics between a and b. Once the ratios are summed, this value can be used in Equation 2.19 to calculate the bridge modularity for component i. The values for the example product are shown in Table 2.4.

Sosa et al. point out a few observations with regards to bridge modularity which are also consistent with the values calculated for the example product:

- The more a component bridges between other components, or in other words the more dependency paths it is on, the less modular it is.
- Components lying on the most geodesics are those bridging the most components and, therefore, are considered to be the least modular.
- Maximum bridge disconnectivity, 1.0, occurs when a component does not bridge any other pair of components because it is not on any of the (n-2)(n-1) maximum possible paths between the other (n-1) components.
- Minimum bridge disconnectivity, 0.0, occurs when it is at the center of a star-shaped configuration with bidirectional ties to all peripheral components.
- A component is more modular if it has a higher $M(\mathrm{B})_i$ which occurs when it appears on fewer geodesics.

Sosa et al. consider their measures to be complementary to each other since each measure looks at component modularity from related yet slightly different perspectives. It should be noted at this point that the design dependencies used in the example and up to this point were considered to be of the same type, e.g. spatial, material, etc.. The authors point out that the design dependencies can be broken down into categories to include: spatial, structural, material, energy, and information. Each of the dependency categories need to be characterized when trying to measure the component modularity. The authors attempt to study how the measures relate to each other both within and across design dependency category types by performing two correlation analyses. This correlation analysis appears to be insufficient in correlating the various modularity groups with the various design dependency category types and provides an opportunity to improve the current state of the research in this area. Sosa et al. do recognize in the recommendation for future work that a way to combine the three groups of measures to attain an aggregated measure of component modularity still needs to be done.

2.3 Comparison of Modular Design Methods

One of the benefits of a modular design method should be the ability to facilitate redesign of a complex product, and subsequently improving the overall product. While many redesigns can occur using various design methods, Guo and Gershenson [20] looked at a way to measure the overall value of each of the design methods in order to compare and select the best modular design method. Here the best modular design method is one that can produce a more modular design with the greatest efficiency or in the least amount of iterations. In order to make a comparison among the various design methods, a common modularity measure first had to be developed that could be calculated before and after the redesign methods were applied.

In 2004, Guo et al. [20] created Equation 2.20 as a common measure of modularity. This common measure calculates modularity by averaging the relationships

within modules and subtracting the averaged relationship external to modules.

$$\text{Modularity} = \frac{1}{M}\left(\sum_{k=1}^{M}\frac{\sum_{i=n_k}^{m_k}\sum_{j=n_k}^{m_k}R_{ij}}{(m_k-n_k+1)^2} - \sum_{k=1}^{M}\frac{\sum_{i=n_k}^{m_k}\left(\sum_{j=1}^{n_k-1}R_{ij}+\sum_{j=m_k+1}^{N}R_{ij}\right)}{(m_k-n_k+1)(N-m_k+n_k-1)}\right); \tag{2.20}$$

Where: n_k: index of the first component in k^{th} module

m_k: index of the last component in k^{th} module

M: total number of modules in the product

N: total number of components in the product

R_{ij}: the value of the i^{th} row and j^{th} column element in the modularity matrix

Guo et al. used this measure while comparing matrix based redesign methods but their common measure could also be applied to function based redesign methods if they are put into a matrix form. An example matrix, Figure 2.1, is revisited here to demonstrate the calculation method of Equation 2.20, but as Guo et al. point out, the actual number that is calculated is only relatively useful. That is, the result that is calculated can indicate whether a system is more or less modular but the real benefit lies in the comparison of the calculation before and after a system is altered by a redesign or some other change. This aids in identifying the effects of a redesign on a system, e.g. is it more or less modular due to the change.

The common modularity measure begins with the system decomposition and the development of the DSM. For the example calculation, the DSM in Figure 2.1 is used but is modified as a symmetric DSM, as was done in Figure 2.3 (minus the subsystem groupings), that is grouped into 3 modules comprised of 4, 1, and 5 components each as shown in Figure 2.9.

Once the DSM is established, Equation 2.20 can be used to calculate the common modularity measure. While the equation looks complicated, it is simply a matter

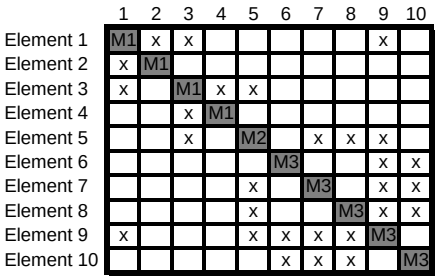


Figure 2.9: Example system DSM showing 3 modules and their components and relationships

of summing and averaging the number of boxes that have entries within modules and then between modules and finally performing the subtraction operator. The parameters for the example problem are shown in Table 2.5.

Table 2.5: Values used from example DSM to calculate common modularity measure

Module k	n_k	m_k	M	N
M1	1	4	3	10
M2	5	5	3	10
M3	6	10	3	10

Using the values in Table 2.5, the common modularity measure was calculated to be 0.44 for the modular system. This low number shows that there is room for improvement with regards to the modularity of the system. Again, the useful part of this measure is in the redesign and re-calculating the value after the design changes to see if there is an improvement in modularity. Improvement is indicated by the system being more modular in this case and hence yielding a higher value for the measure. For the sake of comparison, the common modularity measure was calculated using the integral and modular system DSMs in Equation 2.1. These values are shown in Table 2.6. Our integral system had interfaces between each component and every other component and so the modularity measure is the minimum achievable, -1. For the modular system, it was assumed that one interface existed between modules 1 and 2 and modules 2 and 3 whereas, because of the setup, two interfaces exist with module 2. The modularity measure for the modular system is not the theoretical

Table 2.6: Common modularity measures for fully integral and modular systems with 3 modules each

System	M	N	m
Fully integral	3	5	-1.0
Fully modular	3	5	0.44

maximum. For the true maximum (+1) to occur, there would be zero interfaces between modules and only interfaces within the modules would exist. This is the case when analyzing a product from several design interface types, e.g. spatial, structural, material, energy, and information. A module may have no material interfaces (e.g. functional requirements to transfer airflow, oil, fuel, water, etc.) and would have a +1 modularity measure for this domain. Whereas the same component may have spatial and structural interfaces that would have a less than maximum value for degree of modularity.

2.4 Modularity versus Performance

Limited research has been found on modularity versus performance of a system. Hölttä et al. [22] offer the only article discussing tradeoff between modularity and performance for engineered systems and products. They recognize the benefits of modularity and integrality of systems and the research that has gone into understanding both approaches. They also found that the evaluation of which is better and when is often subjective, qualitative, or speculative. Therefore, Hölttä et al. attempt to show more quantitatively through two examples what effect technical performance (efficiency) constraints have on the degree of modularity of two pairs of functionally equivalent products: a cell phone versus a desk phone, and a laptop computer versus a desktop computer.

Hölttä et al. first decompose the system using a DSM which in turn is used to analyze the coupling and modularity of the system. They then compute the SMI which is described in Section 2.2. While this does present a quantitative method for comparing whether a system is more modular versus more integral, it really doesn't

provide any new information. The SMI that is calculated for a cell phone versus a desk phone is 0.78 versus 0.87, respectively. The SMI that is calculated for the laptop computer versus the desktop computer is 0.83 versus 0.87, respectively. In both cases, as shown in Table 2.7, where technical performance constraints such as light weight and compactness (smaller volume) were design parameters, the SMI was smaller versus its non-constrained counterpart. In both cases, this result also indicates a higher degree of integrality for the constrained system than its counterpart which is considered to be more modular. Ultimately, while the SMI does provide a quantitative measure, it doesn't really provide a quantitative measure of modularity versus performance.

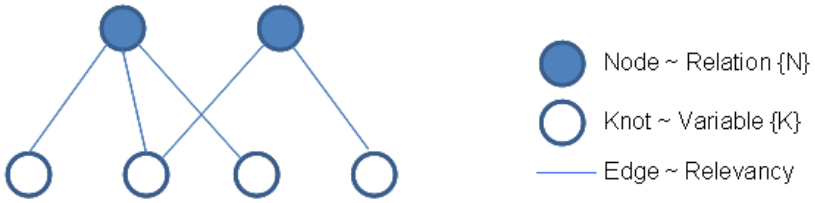
Table 2.7: SMI and performance constraints for two pairs of functionally similar systems, adapted from [22]

System	Performance Property/Constraint	SMI
Desk phone		0.87
Cell phone	Compact, Light weight, Mobile	0.78
Desktop computer		0.87
Laptop computer	Compact, Light weight, Mobile	0.83

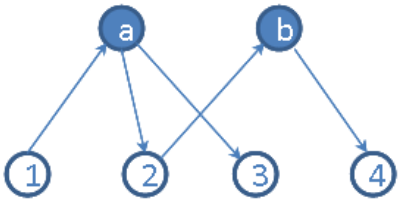
2.5 Graph Theory

A bipartite graph [35] is a simple graph whose vertices can be partitioned into two sets such that each edge of the graph joins a vertex in the first set to a vertex in the second set. In constraint theory [12], the two sets of vertices represent nodes and knots with edges connecting the two. A node corresponds to the model's relations whereas the knots correspond to the model's variables as depicted in Figure 2.10.

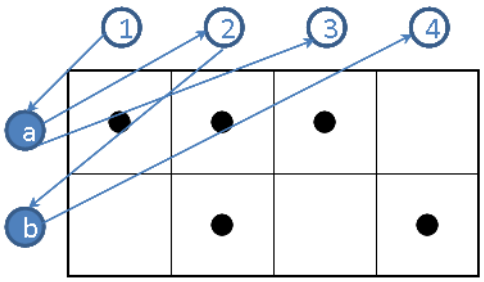
The graph is transformed and represented by the constraint matrix where the columns of the matrix are the knots or variables and the rows are the nodes or relations. The elements of the matrix will be filled if there is relevancy between a knot and a node, otherwise it will be empty. The arrows of the bipartite graph determine whether the elements in the matrix take on values of -1, 0, or +1 which indicate constraint flow from node to knot, constraint flow from knot to node, or relevancy



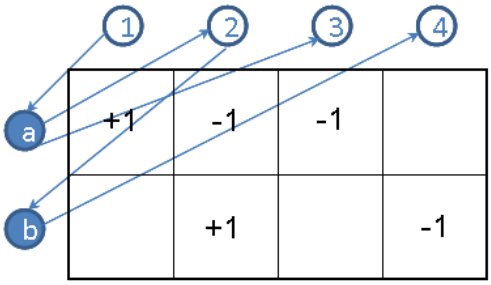
(a) Undirected Bipartite Graph



(b) Directed Bipartite Graph



(c) Relevancy between Directed Bipartite Graph and the Constraint Matrix



(d) Constraint Matrix Developed

Figure 2.10: Development of the Constraint Matrix from the Bipartite Graph Model (a), (b), (c), and (d)

exists with no constraint flow, respectively. In this research, bipartite graphs are used to map directional interfaces between modules.

2.6 Axiomatic Design Principles

There are various methods in practice to decompose a system. In this research, as mentioned previously, the first axiomatic design principle [54] is used to decompose a system by mapping functions to modules. The axiomatic design framework is described as it is also be used as one of the building blocks of this research. Lindholm et al. [29] generalize the framework developed by Suh [53] by breaking it down into five concepts:

1. domains

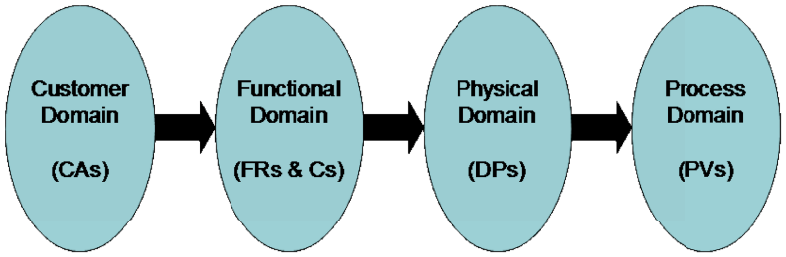


Figure 2.11: Four domains used in the axiomatic design framework along with the associated characteristics

2. hierarchies
3. zigzagging
4. design axiom 1
5. design axiom 2

There are four domains that are used in the design framework: 1) customer domain, 2) functional domain, 3) physical domain, and 4) process domain. Each domain has a characteristic vector associated with it. The customer domain has the needs or attributes (CAs) that the customer is looking for in the end product or process. These needs are mapped into the functional domain using functional requirements (FRs) and constraints (Cs). Design parameters (DPs) are conceived or derived from the FRs and Cs in the physical domain. Finally, the DPs are mapped into processes, characterized by process variables (PVs), to produce the product. These mappings are shown in Figure 2.11.

Hierarchies allow a product or system to be rolled up or broken down to appropriate levels of detail or abstraction. Each of the characteristic vectors (FRs, DPs, and PVs) can be decomposed according to the hierarchy. An example from [55] is used to demonstrate this decomposition. The example is for a refrigerator design with two functional requirements as follows:

FR_1 = Freeze food for long-term preservation.

FR_2 = Maintain food at cold temperature for short-term preservation.

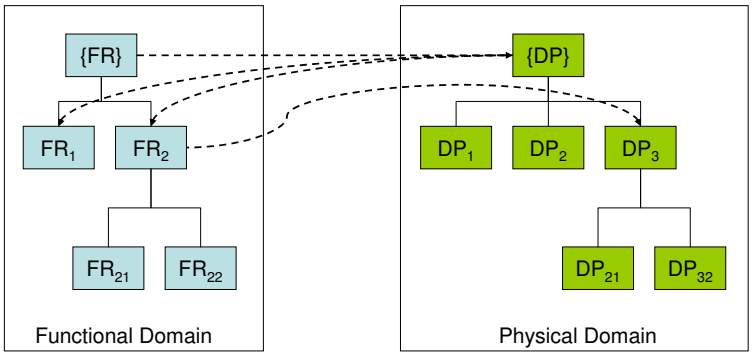


Figure 2.12: Axiomatic design zigzag technique between functional and physical domains

Design parameters are then chosen based on satisfying the FRs. In this example the following two DPs are chosen which sets up the refrigerator to be designed into two compartments:

DP_1 = The freezer section

DP_2 = The chiller (i.e. refrigerator) section

The resulting design equation is shown in Equation 2.21.

$$\begin{Bmatrix} FR_1 \\ FR_2 \end{Bmatrix} = \begin{bmatrix} X & 0 \\ 0 & X \end{bmatrix} \begin{Bmatrix} DP_1 \\ DP_2 \end{Bmatrix} \tag{2.21}$$

It is important to note that in the hierarchy framework, in order to decompose FRs into lower levels of abstraction, DPs and PVs that satisfy the upper level FRs need to be selected. Once the process of selecting DPs to satisfy FRs and PVs to satisfy DPs is complete, the process is repeated until the required level of abstraction is reached. This process is referred to as zigzagging as it zigzags between the four domains. This zigzag process is illustrated in Figure 2.12.

The final two concepts in the axiomatic design framework are the two design axioms.

1. **Independence Axiom** - Maintain the independence of the functional requirements (FRs) [55].

2. **Information Axiom** - Minimize the information content of the design [55].

The set of FRs are defined as the minimum set of independent requirements that the design must meet in order to satisfy the design goal(s). It is important to note that the independence of the FRs are with respect to the functions, not necessarily interpreted as physical independence. As for the second axiom, information content refers to the information that is required in order to satisfy the FRs. The second axiom relates the information content into a probability of success of satisfying the FRs. The second axiom translated states that system designs with the smallest set of required information content are the best as they require the least amount of information to achieve the design goals.

III. Development of a Measure to Assess Reconfigurability of Modular Products

3.1 Introduction

One of several recognized benefits of modularizing a product is the flexibility that is gained ([16]). Flexibility in this context refers to a product's ability to adapt to changing requirements. Flexibility has two elements, a reconfigurability element and an extensibility element. This paper focuses on the reconfigurability element. What does it mean to be reconfigurable? Is it a measure of the number of reconfigurations possible using the options available in a plug-and-play type architecture? If so, how does one compare a product with three modules versus a product with ten modules when each module has three options to choose from when building the product? It is hypothesized that the number of reconfigurations possible is only one aspect of measuring reconfigurability. A measure of reconfigurability should also take into account the number of options available for each module as well as the number of modules that have options. This paper develops a measure to assess the reconfigurability of modular products using these aspects followed by an application of the measure to two U.S. Air Force munitions. The measure presented strictly focuses on a mathematical viewpoint.

3.2 Modularity and Reconfigurability

In general, a product can be decomposed into modules and the associated functions that they perform (Fixson [11], Pahl [38], and Suh [54]). The definition of a module varies among disciplines but here it is defined as a grouping of components or subassemblies that perform one or more functions. Modules are arranged according to a product's architecture. Each of the modules has one or more ways in which it interfaces with the rest of the product. By interface, we refer to one or more interfaces of the five categories of interface types (spatial, informational, material, energy,

or structural) given by Sosa *et al.* [42]. To gain the recognized benefits of modularity, including reconfigurability, it is advantageous to have multiple options for each of the modules that make up a product according to its architecture. Using these options and the product's architecture, multiple reconfigurations of a product can be assembled.

3.3 Reconfigurability Components

In order to compare the reconfigurability across products with varying numbers of modules and varying numbers of options, more than just the total number of reconfigurations, r, is needed. To show this, consider a product with three modules where each module has two options to choose from when assembling the product. Compare this to a product with ten modules where each module also has two options to choose from. In this example of two products, the former has 2^3 reconfigurations possible whereas the latter product has 2^{10} reconfigurations possible. How does the number of reconfigurations change when the total number of options, S, changes from six and 20, as in the current example, to 20 and 20? Both products now have the same number of total options, S, but the mean number of options across the modules with options is now different. For the product with 20 options and ten modules, the mean is still 2. For the former product, the mean is now 6.67 (instead of 2). The number of reconfigurations is no longer 2^3 and 2^{10}, respectively; they are now a minimum of $2 \cdot 2 \cdot 16$ and 2^{10}. The value of r depends on how the number of options is varied across the modules. The maximum number of possible combinations for a given number of modules with options, t, and total options, S, occurs when the standard deviation, σ, is closest to zero, and is in fact maximized at $\sigma = 0$ (the latter is not possible if $S/t \notin \mathbb{N}$). The number of reconfigurations that are possible, r, for a product of a given architecture follows Eq. (3.1). In this equation, r is the number of reconfigurations possible, s_i is the number of options for the i^{th} module with options, and k_i is the number of options to be chosen for the i^{th} module, which is limited to

Example Product	Module A	Module B	Module C s_1	Module D s_2	Module E s_3
Number of Options	1	1	2	3	7

Table 3.1: Example Product Composition of Modules and Options, $n = 5$

one for the purposes of the present reconfigurability measure (RM) development.

$$r = \binom{s_1}{k_1}\binom{s_2}{k_2}\cdots\binom{s_i}{k_i} \tag{3.1}$$

$$1 \leq i \leq t, \quad k_i = 1$$

As seen in Eq. (3.1), the number of reconfigurations or combinations of modules takes on a s_i-choose-k_i arrangement. For example, assume a product has $n = 5$ modules (see Table 3.1) where three of those modules have multiple options to choose from when constructing the product; $t = 3$. In this case, where $k_i = 1$, the product has $r = 42$ possible reconfigurations (see Eq. (3.2)).

$$r = \binom{1}{1}\binom{1}{1}\binom{2}{1}\binom{3}{1}\binom{7}{1} = 2 \cdot 3 \cdot 7 = 42 \tag{3.2}$$

Modules A and B form the product platform since those modules have no options; the product *must* contain those two specific modules as shown in Fig. 3.1. Modules C, D, and E have two, three, and seven options available to choose from, respectively, when assembling the product. The remainder of the reconfigurability measure development and analysis focuses on only modules with options since only these modules add to the overall number of reconfigurations possible. Thus, Modules C, D, and E, become Modules 1, 2, and 3, as depicted by s_1, s_2, and s_3 in Table 3.1. The mean number of options per module is 4. The standard deviation, σ, is 2.65. The sum of the options across the three modules, S, is 12. Using the various elements given thus far, the reconfigurability measure can be developed.

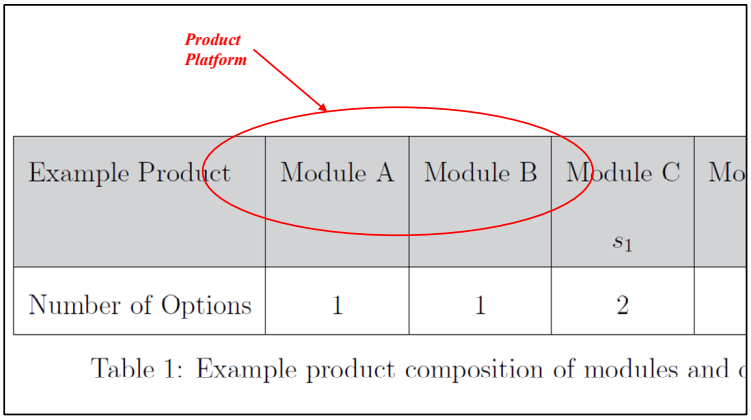


Figure 3.1: Product Platform

3.4 Reconfigurability Measure

All of the pieces (r, t, n, S) that go into and make up the reconfigurability measure (RM) are now available. The RM is stated first and discussed further in subsequent subsections. The RM is broken down into four ratios (y_1, y_2, y_3, and y_4) that are captured in vector form, $\boldsymbol{Y}$, and are given by Eqs. (3.3) and (3.4), and Fig. 3.2.

$$\boldsymbol{Y} = [\, y_1 \;\; y_2 \;\; y_3 \;\; y_4 \,] \tag{3.3}$$

$$= \left[\frac{r}{S} \;\; \frac{r}{t} \;\; \frac{t}{n} \;\; \frac{r}{r_{u.b.}} \right] \tag{3.4}$$

where: $S = \sum_{i=1}^{t} s_i$ = Total number of options

n = Number of modules in a product

t = Number of modules with options

r = Number of possible reconfigurations

$r_{u.b.}$ = Upper bound number of reconfigurations for a given S and t pair

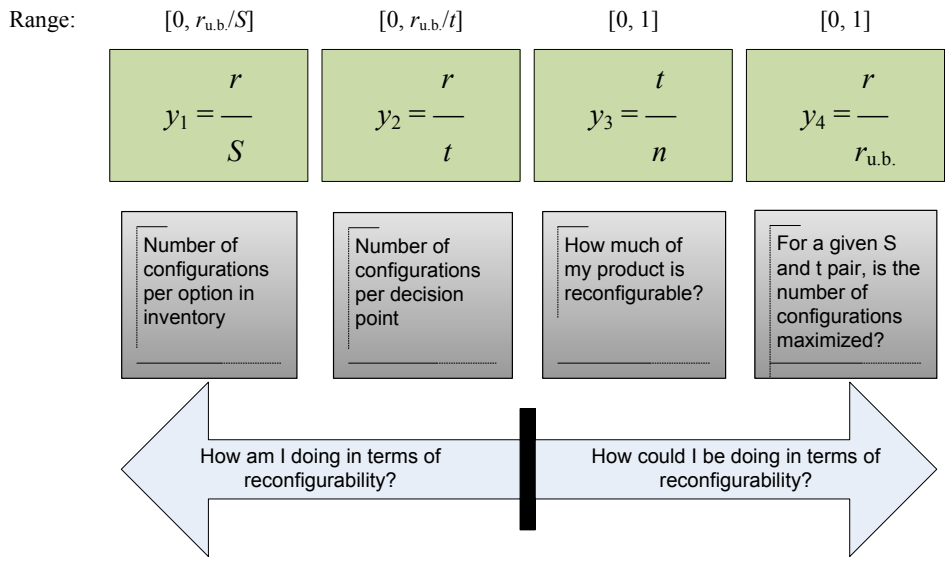


Figure 3.2: Reconfigurability Measure

Equation (3.4) uses the mathematical number of reconfigurations possible, r. In reality, the actual number of reconfigurations realizable, r_{act}, will be less than or equal to r due to pair-wise incompatibilities between option choices for the i^{th} and j^{th} modules.

$$r_{\text{act}} \leq r \tag{3.5}$$

To improve the quality of the RM, r_{act} should be used when available. This may or may not be possible to determine early in a conceptual design. For more mature products and in reverse-engineering cases, it may be easier to determine the number of realizable reconfigurations.

As hypothesized, a RM should take into account not only r, but also S and t. The number of reconfigurations possible, r, is a function of S and t. In developing a RM, there are four possible combinations of S and t that need to be addressed which are summarized in Table 3.2. These relationships are used to evolve the equations stated in Eqs. (3.3) and (3.4). Two designs can have the same number of modules with options and the same number of total options, which turns the emphasis towards the number of reconfigurations possible, r. The two designs can have the same number

Situation #	Total # of Options (S)	Total # of Modules w/Options (t)
1	Same	Same
2	Same	Different
3	Different	Same
4	Different	Different

Table 3.2: Four Situations When Varying Total Number of Options, S, and Total Number of Modules with Options, t

of modules with options, t, but different number of total options, S, and vice versa. Lastly, the two designs can have different numbers of modules with options and different numbers of total options.

3.5 **Y** *Measure Development*

A product is comprised of n modules. Of those n modules, t have a minimum of two or more options that can be used in the product's architectural build. By definition, in order to have a reconfigurable product, the product must have at least $t = 1$ module with options. The case where $t = 1$ is trivial; the total number of reconfigurations (where $k_1 = 1$) will only vary by the number of options, S or s_1, available to that module. Increasing the reconfigurability of the product is straightforward; either more options for that one module, or adding options to other modules, will increase the overall product reconfigurability. The real analysis comes when $t \geq 2$. In this case, after assessing the reconfigurability, design decisions can be made using the results of the reconfigurability measure. Now that there are multiple modules with options, decisions must be made as to which modules to focus on in order to increase the overall reconfigurability. In making these decisions, several factors come into play that are highlighted and discussed subsequently. These factors evolve from each of the situations listed in Table 3.2.

3.5.1 Situation 1. The first situation arises when two designs have the same number of total options, S, and the same number of modules with options, t. One can

envision this situation if a manufacturer chooses to use an existing product and add module options to keep up with technology and perhaps discontinue some obsolete module options. Consider Product A with $S = 12$, and $t = 3$. Product A.1 also has $S = 12$, and $t = 3$. Table 3.3 shows the distribution of options for each of the products. Table 3.4 shows the number of reconfigurations possible, r, along with the first ratio, y_4, to be discussed for Product A and Product A.1.

The y_4 term of the RM is a ratio of the number of reconfigurations possible for a given product divided by the maximum number (or upper bound) of reconfigurations possible based on the given S and t pair. The y_4 ratio is an indication of how the product measures in terms of reconfigurations achievable compared to the maximum number of reconfigurations possible for the given S and t pair as shown in Fig. 3.2. If $y_4 = 1$, then the current design is achieving the maximum number of reconfigurations possible. As noted earlier in Section 3.3, r is maximized when the standard deviation between the modules with options, s_i, is equal to zero or is minimized for a given S and t pair. Beyond this, the only way to increase r is to increase S or t which leads to Situations 2, 3, or 4. In calculating $r_{\text{u.b.}}$ in the natural number domain ($\mathbb{N}$) for the given S and t pair, s_i values are chosen such that σ is minimized. For example, if $S = 20$ and $t = 3$, then the mean of the s_i values used in calculating $r_{\text{u.b.}}$ is $S/t = 6.67$. Since $S/t \notin \mathbb{N}$, then as noted in Section 3.3, σ cannot equal zero but should be minimized. Therefore, a theoretical distribution of s_1, s_2 and s_3 values that minimizes σ includes a combination of sixes and sevens. In this example, $r_{\text{u.b.}}$ is calculated as shown in Eq. (3.6), noting the sum of the s_i values equals 20 and $\sigma = 0.58$.

	Module 1 s_1	Module 2 s_2	Module 3 s_3	S	t
Product A	2	2	8	12	3
Product A.1	4	4	4	12	3

Table 3.3: Sample Product A and Product A.1 Module Option Distribution

	r	$y_4 = \frac{r}{r_{u.b.}}$
Product A	32	0.5
Product A.1	64	1

Table 3.4: Sample Product A and Product A.1 Total Number of Reconfigurations and y_4

	S	t	r	Std dev	y_4
Product A	12	3	32	3.46	0.5
Product A.1	12	3	64	0	1

Table 3.5: Summary of Reconfigurability Measure for Product A and Product A.1

$$r_{\text{u.b.}} = 6 \cdot 7 \cdot 7 = 294 \tag{3.6}$$

Returning to Situation 1, Table 3.4 shows that Product A.1 has twice the number of reconfigurations as Product A for the same number of total options and the same number of modules with options. A benefit of this analysis is that the manufacturer can keep the same number of options in inventory but by changing which module options to carry, he/she can maximize the product variety without increasing inventory. The summary of the reconfigurability measures of the two sample products thus far is given in Table 5.

3.5.2 Situation 2. The first situation highlighted the importance of r and the distribution of s_i values when comparing the reconfigurability of two designs. Continuing with the example products from Situation 1, the next situation can arise when, for example, two products having the same number of total options, S, are being compared, but they have varying numbers of modules with options, t. Sample Product B and Product C are used to illustrate this situation as shown in Tables 3.6 and 3.7.

In this second situation, even though both products have a standard deviation equal to zero and hence maximized number of reconfigurations possible for their

	Module 1	Module 2	Module 3	Module 4	S	t
Product B	4	4	4	-	12	3
Product C	3	3	3	3	12	4

Table 3.6: Sample Product B and Product C Module Option Distribution

	S	t	r	Std dev	$y_3 = \frac{t}{n}$	$y_4 = \frac{r}{r_{u.b.}}$
Product B	12	3	64	0	0.6	1
Product C	12	4	81	0	0.8	1

Table 3.7: Summary of y_3 and y_4 for Product B and Product C, $n = 5$

respective S and t pairs, Product C is more reconfigurable. The measure uses t to help differentiate two products that have the same number of total options. Using t also highlights the significance and potential for additional reconfigurations using modules in a product that have been identified as capable of having options. Product C has four modules with options and hence four potential opportunities to increase the number of options versus Product B that only has three modules with options. Thus, the t term acts as a "reconfigurability potential" factor. This potential is captured in the y_3 ratio that indicates to a designer how much of the product is reconfigurable as shown in Fig. 3.2.

The problem setup so far is one of maximizing the product of the s_i terms (or r) for a given sum of s_i terms, S. This setup is identical to the mathematical problem of partitioning a number, S, into t parts, such that the product of the partitions (s_i terms) is maximized (Krause [27]). In general, there is a strong correlation between decreasing the standard deviation of a set of partitions and increasing the number of reconfigurations. There are some exceptions to this relationship which are not enumerated here but involve situations where the number of options for multiple modules is three. This is due to the fact that, in the real number domain, the maximum number of product reconfigurations for a given S occurs when the average number of options per module is e (2.7182818), obviously not possible where $s_i \in \mathbb{N}$ (Krause [27] and authors). As proof of this, the first-order necessary condition ($\tilde{r}' = 0$)

is given below and it can easily be shown that the second-order sufficient condition ($\tilde{r}'' < 0$) is also satisfied (Arora [4]).

$$\tilde{r} = \left(\frac{S}{t}\right)^t \tag{3.7}$$

$$ln(\tilde{r}) = ln\left(\left(\frac{S}{t}\right)^t\right) \tag{3.8}$$

$$= t\ \ln(S) - t\ \ln(t) \tag{3.9}$$

Taking the derivative of both sides w.r.t. t

$$\frac{1}{\tilde{r}}\ \frac{d\tilde{r}}{dt} = \ln(S) - \ln(t) - \frac{t}{t} \tag{3.10}$$

$$\frac{d\tilde{r}}{dt} = \tilde{r}\,(\ln(S) - \ln(t) - 1) = \tilde{r}' \tag{3.11}$$

Setting the r.h.s. equal to 0

$$\Longrightarrow \ln\left(\frac{S}{t}\right) = 1 \tag{3.12}$$

$$\Longrightarrow \frac{S}{t} = e \qquad . \tag{3.13}$$

Having maximized the product of the s_i values in the real number domain, we turn the focus to a subset of that domain, the natural number domain, $\mathbb{N}$. In $\mathbb{N}$, the maximum number of reconfigurations, $r_{\max}$, for a given S, allowing t to vary in $\mathbb{N}$, occurs when the maximum number of modules with options have three options, and the remaining (if there are any) one or two modules have two options. We define $r_{\max} = r^*$, where t^* is the required number of partitions of the given S to achieve r^*. Beginning with the selection of t^* as the ceiling of $S/3$ shown in Eq. (3.14), r^* is calculated using one of the cases in Eq. (3.15).

$$t^* = \left\lceil \frac{S}{3} \right\rceil \tag{3.14}$$

$$r^* = \begin{cases} 3^{t^*} & : \quad \frac{S}{t^*} \in \mathbb{N} \\ 3^{(t^*-1)} \cdot 2 & : \quad \frac{S+1}{t^*} \in \mathbb{N} \\ 3^{(t^*-2)} \cdot 2 \cdot 2 & : \quad \frac{S+2}{t^*} \in \mathbb{N} \end{cases} \tag{3.15}$$

While $S/2$ is the mathematical upper bound for t, we restrict our domain of interest (DOI) using t^* as the upper bound for t given S. This t^* in turn yields an upper bound for r, r^*, for the given S. A lower bound for r, $r_{\text{l.b.}}$, is also included in defining our DOI. Equations (3.16), (3.17), (3.18), and (3.19) define further and summarize our DOI.

$$DOI: \quad \{s_1, s_2, ..., s_t, t\} \quad : \quad S = \sum_{i=1}^{t} s_i \geq 2t + 2 \tag{3.16}$$

$$r_{\text{l.b.}} = 2^{t-1} \cdot 4 = 2^{t+1} \tag{3.17}$$

$$2 \leq t \leq t^* \tag{3.18}$$

$$r_{\text{l.b.}} \leq r \leq \begin{cases} r_{\text{u.b.}} & : \quad t < t^* \\ r* & : \quad t = t^* \end{cases} \tag{3.19}$$

The DOI is established such that decisions can be made after gaining the insights from performing the reconfigurability measure analysis. When a product has values outside of the DOI, then design decisions are limited and straightforward such as the case when $t = 1$. For the given t, increasing reconfigurability simply involves increasing module options for s_1, thus not requiring an in-depth analysis that the RM assessment provides. Another example to illustrate the DOI suitability is if $S = 4$ or $S = 5$, and $t = 2$, then $r = 4$ and $r = 6$, respectively. Unless S or t changes, there are no changes to the s_i values that will produce a larger r. However, if $S = 6$ and $t = 2$, then there

	Module 1	Module 2	Module 3	Module 4	S	t
Product B	4	4	4	-	12	3
Product C.1	2	2	2	6	12	4

Table 3.8: Sample Product B and Product C.1 Module Option Distribution

	S	t	r	Std dev	$y_3 = \frac{t}{n}$	$y_4 = \frac{r}{r_{u.b.}}$
Product B	12	3	64	0	0.6	1
Product C.1	12	4	48	2	0.8	0.59

Table 3.9: Summary of y_3 and y_4 for Product B and Product C.1, $n = 5$

are two possible s_i distributions that impact r differently; specifically, $r = 2 \cdot 4 = 8$, or $r = 3 \cdot 3 = 9$.

Returning to Product B and Product C, they are both maximized for reconfigurations possible for each of the S and t pairs respectively. What happens when one of the products is not maximized? Table 3.8 shows Product B again and a non-maximized Product C.1 that is in fact minimized (r = minimum for the given S and t pair). Even though Product C.1 has more modules with options than Product B and hence has more potential for increasing the number of reconfigurations, the current option distribution achieves the least number of reconfigurations possible for the given S and t pair. The reconfigurability measure captures this in r and y_4 showing that Product C.1 is less reconfigurable than Product B. This is consistent with the maximum number of reconfigurations possible since Product C.1 has less reconfigurations possible than Product B. It should also be noted that Product C.1 also has less reconfigurations possible than Product C which has the maximum number of reconfigurations possible (for $t = 4$ and $S = 12$).

A practical example of Situation 2 is when a manufacturer determines the total number of options that he/she would like to keep in inventory and then uses that number, S, to determine the maximum number of reconfigurations possible if t was not limited by design, only by $S/2$. For example, if $S = 30$, then the theoretical maximum number of reconfigurations ($r = r^*$) would be $3^{10} = 59,049$ and $t = t^* = 10$.

	Module 1	Module 2	Module 3	Module 4	S	t
Product D	4	3	4	3	14	4
Product E	2	3	6	4	15	4

Table 3.10: Sample Product D and Product E Module Option Distribution

If only 7 modules currently have options then the manufacturer could focus on the modules without options to continue to maximize the number of reconfigurations.

A final note on the assessment thus far; y_3 and y_4 are indicators to designers on how the current design is performing in terms of reconfigurability as shown in Fig. 3.2. If $y_3 = y_4 = 1$, then the current product design is maximizing the number of reconfigurations possible for the given S and t pair. If either or both terms is less than one, then the current design is not maximizing the number of reconfigurations possible and could be increased for the given S and t pair.

3.5.3 Situation 3. The previous situation (Situation 2) prompted the use of the t term in the reconfigurability measure, specifically in y_3. Situation 3 prompts the use of S, total number of options, in the y_1 term, and reinforces the y_4 term. Situation 3 involves two products that have the same number of modules with options but have different numbers of total options.

Tables 3.10 and 3.11 show Product D and Product E having the same number of reconfigurations possible, r. The y_1 term in the RM was developed to further differentiate two products. This term identifies the average number of product reconfigurations made possible per module option in inventory as shown in Fig. 3.2. After calculating the reconfigurability measures for both Product D and Product E, it can be seen that Product D is slightly more reconfigurable in y_1 than Product E since Product D is able to achieve the same number of reconfigurations as Product E with less modules.

This example highlights where efforts would be worthwhile to increase the number of reconfigurations for Product E, namely on Module 1 (an additional option for

	S	t	r	Std dev	$y_1 = \frac{r}{S}$	$y_4 = \frac{r}{r_{u.b.}}$
Product D	14	4	144	0.58	10.29	1
Product E	15	4	144	1.71	9.60	0.75

Table 3.11: Summary of y_1 and y_4 for Product D and Product E, $n = 5$

	Module 1	Module 2	Module 3	Module 4	Module 5	S	t
Product D	4	3	4	3	-	14	4
Product F	4	3	3	6	6	22	5

Table 3.12: Sample Product D and Product F Module Option Distribution

Module 1 could achieve a 50% increase in the number of possible reconfigurations). Additionally, the y_1 term is an indicator to designers on how the current design is performing in terms of reconfigurability and the average number of reconfigurations that are achieved for each module that is kept in inventory, or for which (re)configuration control is maintained.

3.5.4 Situation 4. The last situation reinforces all of the constituent terms in the reconfigurability measure and gives rise to one additional term, y_2. Situation 4 arises when two products or two designs have $S_1 \neq S_2$ and $t_1 \neq t_2$ as seen in Table 12.

Table 3.13 shows the complete RM calculated for Product D and Product F when the total number of modules in the product, n, is eight. The y_2 term highlights the average number of reconfigurations being achieved per module with options. Another way to view this is as follows: on average, how many reconfigurations is the design achieving for each decision point in the architectural build? Certainly Prod-

	S	t	r	Std dev	y_1 $\frac{r}{S}$	y_2 $\frac{r}{t}$	y_3 $\frac{t}{n}$	y_4 $\frac{r}{r_{u.b.}}$
Product D	14	4	144	0.58	10.29	36	0.5	1.0
Product F	22	5	1296	1.52	58.91	259.2	0.63	0.81

Table 3.13: Summary of RM (y_1, y_2, y_3, and y_4) for Product D and Product F, $n = 8$

GBU-24	Warhead, forward adapter, guidance control section (GCS)‡ (s_1), airfoil group, support structure, fuze‡ (s_2), initiator‡ (s_3), and external aircraft (EA)‡ (s_4).
GBU-31	Warhead, guidance set, proximity sensor (PS)‡ (s_1), airfoil group, support structure, fuze‡ (s_2), initiator‡ (s_3), and external aircraft (EA)‡ (s_4).

Table 3.14: GBU-24 and GBU-31 Modules

	s_1	s_2	s_3	s_4	S	t
GBU-24	2	6	4	7	19	4
GBU-31	2	2	2	11	17	4

Table 3.15: GBU-24 and GBU-31 Module Option Distribution

uct F has more reconfigurations than Product D, but Product F also must maintain more than 50% more options than Product D. This may or may not be an issue. Additionally, while Product F has a higher number of reconfigurations possible and is considered better from reconfigurability and modularity viewpoints, from a logistics viewpoint, a reconfiguration control viewpoint, or from an assembly time viewpoint, more doesn't necessarily mean better.

3.6 Application

Two guided bomb units (GBUs) in the United States Air Force inventory are used to apply the reconfigurability measure developed in the previous section and subsections. These GBUs are the GBU-24 and GBU-31. They are commonly referred to as the Laser Guided Bomb (LGB) and the Joint Direct Attack Munition (JDAM), respectively. Reconfigurability for the two munitions follows Situation 3 where the numbers of modules with options are equal but the numbers of total options are different ($t_{\text{GBU-24}} = t_{\text{GBU-31}}$, $S_{\text{GBU-24}} \neq S_{\text{GBU-31}}$). The modules that comprise the GBU-24 and GBU-31 are shown in Table 3.14. The modules indicated with a (‡) in Table 3.14 refer to modules that have multiple options. These modules are listed again in Table 3.15 as s_1, s_2, etc., showing the number of options available for each. Both GBUs are comprised of eight modules ($n = 8$), each having $t = 4$ modules with options.

	S	t	r	Std dev	y_1 $\frac{r_{\text{act}}}{S}$	y_2 $\frac{r_{\text{act}}}{t}$	y_3 $\frac{t}{n}$	y_4 $\frac{r_{\text{act}}}{r_{u.b.}}$
GBU-24	19	4	84	2.22	4.42	21	0.5	0.17
GBU-31	17	4	33	4.5	1.94	8.25	0.5	0.10

Table 3.16: Summary of RM for GBU-24 and GBU-31, $n = 8$

Data values from Table 3.15 were used to calculate r, and subsequently, pair-wise constraints were used to calculate r_{act}. Next, the complete RMs for both GBUs were calculated with the results given in Table 3.16. This analysis shows that the GBU-24 is more reconfigurable and that it is achieving more of its potential number of reconfigurations than the GBU-31. The GBU-31 is achieving the least number of reconfigurations for the given S and t pair.

In this application, if a designer chooses one of the existing modules with options to increase by one option, then this would impact the increase in the number of reconfigurations possible by varying amounts. Using the GBU-24, it is easily seen that focusing on increasing the number of options for Module 1 would yield the highest increase in terms of more reconfigurations possible. The reason for this is easily seen when broken up in terms of multiplication and combinations.

$$
\begin{aligned}
&r = 2 \cdot 6 \cdot 4 \cdot 7 = 336, \quad S = 19 \qquad (Baseline)\\
&r = \langle 3 \rangle \cdot 6 \cdot 4 \cdot 7 = 504, \quad S = 20\\
&r = 2 \cdot \langle 7 \rangle \cdot 4 \cdot 7 = 392, \quad S = 20\\
&r = 2 \cdot 6 \cdot \langle 5 \rangle \cdot 7 = 420, \quad S = 20\\
&r = 2 \cdot 6 \cdot 4 \cdot \langle 8 \rangle = 384, \quad S = 20
\end{aligned}
$$

A product with $2{\cdot}6{\cdot}4{\cdot}7$ number of reconfigurations can gain $2{\cdot}6{\cdot}4$ reconfigurations by increasing the fourth term, seven, by one to eight. But, this same product can increase the number of reconfigurations by increasing the first term to three by $6 \cdot 4 \cdot 7$. This example is assuming the overall number of options, S, is increased by one. The same relation holds true if the second, third, or fourth term is decreased by one (holding S constant) as the product would still gain the most number of reconfigurations by

	S	t	r	Std dev	y_1 $\frac{r}{S}$	y_2 $\frac{r}{t}$	y_3 $\frac{t}{n}$	y_4 $\frac{r}{r_{u.b.}}$
GBU-24	19	4	336	2.22	17.68	84	0.5	0.67
GBU-24 (Actual)	19	4	84	2.22	4.42	21	0.5	0.17
GBU-31	17	4	88	4.5	5.18	22	0.5	0.28
GBU-31 (Actual)	17	4	33	4.5	1.94	8.25	0.5	0.10

Table 3.17: Comparison of Using r Versus r_{act} in the RM Calculation for the GBU-24 and GBU-31, Where $n = 8$

increasing the number of options for Module 1. The increase in reconfigurations for this example would be $5 \cdot 4 \cdot 7$, $6 \cdot 3 \cdot 7$, or $6 \cdot 4 \cdot 6$.

Table 3.16 uses the the actual number of reconfigurations which eliminates combinations from r due to pair-wise constraints. Both GBUs have pair-wise constraints that result in $r_{\text{act}} < r$. The impacts of r versus r_{act} on $\boldsymbol{Y}$ are shown in Table 3.17, effectively reducing three of the four $\boldsymbol{Y}$ terms. This highlights the importance of using r_{act} when it is available. Additionally, it shows the impact of pair-wise constraints on the achievable number of reconfigurations.

3.7 Conclusions

It was shown that measuring reconfigurability requires more than just calculating the number of reconfigurations possible for a given product. A reconfigurability measure (RM), as hypothesized, must also take into account the total number of options available to a product, S, and the total number of modules with options, t. Using these additional terms in the RM, a designer or decision maker can understand how the current design is performing compared to how well it could be performing in terms of the total number of reconfigurations.

If a product has one or more modules that do not currently have options, then by focusing on one of these modules, the highest increase in the number of reconfigurations will be realized if one or more options can be added to this module. Similarly, if the number of options for each module with options varies greatly (high standard deviation), then the highest increase in the number of reconfigurations would come

when making design changes such that each module with options has the same number of options for a given S.

Insight is also gained as to how well the current design is performing in terms of the number of reconfigurations possible per total options in inventory as well as per number of modules with options. The latter term, y_2, shows the number of reconfigurations being realized per decision point that must be made for a given architectural build. The higher the number of decision points, t, the higher the number of potential interfaces and reconfiguration controls will be needed.

Three of the four RM ratios use the number of reconfigurations possible, r, in their calculation. Pair-wise constraints effectively reduces r and hence the three ratios that use it. Minimizing the pair-wise constraints on modules will help to maximize the achievable number of reconfigurations for the given s_i distribution. In turn, this minimization of pair-wise constraints will help to increase the reconfigurability, flexibility, and modularity of a product.

The RM assesses the reconfigurability of modular products strictly from a mathematical viewpoint which stemmed from capitalizing on the benefits of modularity. While this viewpoint is an important starting point in analyzing the reconfigurability of product designs, a system viewpoint along with other viewpoints must also be considered before making decisions on design changes to a product. Increasing module options offers more reconfigurability possibilities mathematically, but it also requires understanding other ramifications and limitations. Module options have associated costs, logistics, pair-wise constraints, etc., that must be considered. Ultimately, the number of reconfigurations maintained will be a balance between user requirements and cost.

While using the reconfigurability measure as a benchmark for comparing products or designs is useful, a significant benefit of the reconfigurability measure also comes from setting up the problem. By examining the product architecture, one can key in on specific areas and even narrow down areas for the greatest increase in re-

configurability and hence modularity. After the initial problem set-up, focus can be applied to the appropriate modules to maximize the increase in the number of reconfigurations. One module may be easy to vary and so already has the highest number of options available. Higher returns, in terms of total number of reconfigurations, may be realized when increasing the number of options for other modules.

IV. Assessing Modularity – a Vector Approach

4.1 Introduction

Product modularity has gained an increase in focus over the last couple decades. The benefits of modularity in product design have been widely recognized and qualitatively captured by [16]. Some of these benefits include changeability, flexibility, reusability, reconfigurability, and extensibility. Several measures exist to quantitatively assess the modularity of a product in terms of interfaces within and between modules which is referred to as degree of coupling in this paper [23,34,42]. But, what does being modular really mean? When comparing the modularity between two products or two designs for a given product, what does it mean when the modularity measure indicates one is more modular than the other? A method to capture recognized benefits of modularity in a rigorous manner is proposed and then demonstrated using two precision guided munition examples.

When decision-makers or designers state they want a product to be more modular, they are indicating that there are one or more aspects of modularity that they want captured in a new design. Current modularity measures roll-up contributing factors to modularity which result in a real number between zero and one as the overall modularity value for a product [23, 34, 42]. This methodology gives no additional insight into the aspects of modularity being realized. The Vector Modularity Measure (VMM) presented in this paper uses degree of coupling as well as the benefits of modularity in a vector form to highlight the contributing factors to a product's modularity assessment. Each of the equations used in the VMM can be used to gain insight into the specific modularity benefits being realized. Designers and decision-makers alike can use this insight to improve existing designs and to aid in overall product selection based on priorities and goals of modularizing a product.

4.2 Background

Before introducing the Vector Modularity Measure, the analysis process, and the applications, a few key terms are introduced. The definitions given below are those of the authors, except where cited, for purposes of this paper and analysis.

- **Modularity** - grouping of components into well defined entities, such as modules or sub-assemblies, that can be further described by the interfaces between them.
- **Interface (I/F)** - spatial, informational, material, energy, or structural connection or coupling of one module to another module within a product [42]. I/F types given below are defined similarly as in [42].
  - **Spatial I/F** - physical adjacency for alignment, orientation, serviceability, assembly or weight.
  - **Informational I/F** - transference of signals or controls.
  - **Material I/F** - transference of airflow, oil, fuel, or water.
  - **Energy I/F** - transference of heat, vibration, electric, or noise energy.
  - **Structural I/F** - transference of loads or containment.
- **Module** - group of components or sub-assemblies that perform one or more functions
- **Reusability** - ability of modules within a product to be used in at least one other product variant.
- **Flexibility** - a product's ability to change or adapt to new requirements; measured in terms of a product's ability to be reconfigurable and extensible.
- **Reconfigurability** - product's ability to be assembled or built in multiple configurations according to its architecture.
- **Extensibility** - built in architectural options for upgrading, or adding functionality to a product.

- **Function** - technical process involving energy, material and/or signals being converted and/or channeled.
- **Flow** - material, signal, and/or energy that can be converted or channeled.

The idea of measuring modularity is not new. [17] reviewed the literature and summarized various methods for measuring modularity through 2004. [19] extended the concept and compared the various modularity measures based on consistency and sensitivity analyses. Several methods have been proposed since 2004 by [23,34], and [42]. These measures quantify the module-to-module connections, both inter- and intra-module, but ultimately focus on coupling of either design parameters or interfaces. Mikkola's measure also accounts for a module's reusability in an exponent term identified as a substitutability factor. None of the measures, however, takes into account the assumed benefits of modularity. While a common consensus exists on the benefits of modularizing a product, no method or measure captures these benefits.

4.3 Vector Modularity Measure

4.3.1 Measure Overview. The Vector Modularity Measure (VMM) proposed herein captures the degree of coupling in a product along with the recognized benefits of modularity in a vector form for further mathematical manipulation. Specifically, the following aspects of modularity are captured in the VMM introduced here: degree of coupling between/among the modules in a system; reusability of the modules; and the flexibility of a product to adapt to changing requirements which is assessed in terms of its reconfigurability and extensibility. Equation 4.1 defines the Vector Modularity Measure, and subsequent subsections detail each of the factors comprising the VMM.

$$\boldsymbol{VMM} = [\ V \quad X \quad \boldsymbol{Y} \quad Z\ ] \tag{4.1}$$

where: V = Degree of coupling
X = Reusability
$\boldsymbol{Y}$ = Reconfigurability

$$Z = \text{Extensibility}$$

The Vector Modularity Measure, VMM, is useful when comparing two similar products in terms of modularity. The composition of the VMM points to the benefits of modularity that are being realized for one product over another one. The measure is also useful when upgrading an existing product. The measure can be calculated for an existing product and then compared to iterations of proposed module designs. Does the new module design increase or decrease the degree of coupling? Is the new module reusable? Are there constraints on the interfaces that the module imposes that will limit its ability to be reconfigured with certain other modules in a product? All of these questions can be answered as a result of performing the VMM calculations. Another useful aspect of the VMM is that it focuses the designer's attention on the benefits of modularity which are the goals of modularizing a product in the first place. This focus of the designer's attention in and of itself is beneficial in increasing a product's modularity.

4.3.2 Degree of Coupling. The first factor in the VMM, the degree of coupling, V, is used to assess how connected/disconnected each module is from each of the other modules within a product. This factor can be used to identify which modules are loosely or highly coupled to the other modules in a product. This assessment can then be used by designers and decision-makers to guide future design decisions regarding which modules to target when trying to improve a product's modularity. For example, the interfaces that a module has to other modules impose constraints on that module according to the product's architecture. These constraints must be considered when redesigning the module. A module that is loosely coupled has fewer constraints than a module that is highly coupled from an interface viewpoint.

This idea of using degree of coupling is similar to the use of in- and out-degree modularity measures in [42]. Another similar concept is the non-zero fraction (NZF) term in the modularity measure of [23]. The NZF is useful in determining a product's connectedness or coupling and will be used in the modularity measure introduced in

this paper. The NZF uses a symmetrical binary design structure matrix (DSM), an n x n matrix where each column and row refers to a module in a product. If an interface exists between two modules, then an "X" is used to indicate the interface. The NZF is then calculated as the ratio of the total number of non-zero entries to the total number of entries minus the diagonal entries, n.

For the VMM, a DSM is built for each of the five interface types (spatial, informational, etc.). However, the DSMs used herein accommodate directional interfaces by type and hence are generally nonsymmetric. The NZF is calculated for each of the five DSMs using Equation 4.2.

$$\text{NZF} = \frac{\sum_{i=1}^{n}\sum_{j=1}^{n} DSMij_{i\neq j}}{n(n-1)} \tag{4.2}$$

where: n = Total number of modules in a product

The degree of coupling, V, between modules is then calculated by summing the five NZF terms over a product and dividing by the total number of interface types. The resultant ratio for the degree of coupling factor is the total number of interfaces divided by the total number of possible interfaces minus the diagonal entries over all five interface types. This calculation effectively results in averaging the NZF terms over the five interface types and is shown in Equations 4.3 and 4.4.

$$V = \frac{1}{5}\sum_{k=1}^{5} \text{NZF}_\text{k} \tag{4.3}$$

$$= \frac{\sum_{k=1}^{5}\left(\sum_{i=1}^{n}\sum_{j=1}^{n} DSMij_{i\neq j}\right)_k}{5n(n-1)} \tag{4.4}$$

The analysis in this paper uses integer values when calculating V. The analysis can be extended to include the real domain as well. One benefit of this extension is that

it accommodates the potential to evaluate design complexity. For example, if a real value is assessed to each interface based on the number of interfaces or the level of complexity for the interface type, Equation 4.4 would need to be slightly modified; the denominator would need to be removed that normalizes the term since an upper limit is no longer imposed on the number of interfaces. This extension as well as additional uses of Equation 4.4 are not expounded upon here but are stated for future research.

4.3.3 Reusability. The reusability factor, X, is an assessment of the percentage of modules of a product that are used in other products. In assessing the reusability of a product, modules are sorted into two categories: unique and reusable. This is similar to the categorization that [34] uses to categorize components. Mikkola identifies reusable components as standard components and then further categorizes each standard and unique components into customizable and noncustomizable components. In assessing reusability, it is not necessary to categorize modules beyond unique and reusable. In the reusability assessment, each module is assigned a binary value indicating whether or not it is used in at least one additional product. The values for the reusable modules are then summed across a product (n_{mp}) and divided by n to attain the overall percentage of a product's module reuse.

$$X = \frac{n_{\text{mp}}}{n} \tag{4.5}$$

where: n_{mp} = Number of modules used in multiple products
n = Total number of modules

The reusability factor highlights to designers what percentage of a product is being reused. In order to obtain the benefit of reusability, designers need to avoid or minimize using unique module designs where possible. For the analysis herein, assessing whether a product is reused or not is sufficient to glean the benefit of reusability being captured. Knowing the extent a module is reused, or the number of products containing the module, has potential benefits beyond the assessment in this paper. For

example, as the number of products that use a given module increases, the probability that the module is or will become a standard module increases. A future adaptation could account for the number of products each module *option* (see Section 4.3.4.1) is used in when building variant configurations of a product. Using this adaptation, module options that are peculiar to a product (i.e. not reusable in other products) are highlighted. In the current assessment, however, they are hidden by the overall categorization of "unique/reusable" if a given module has multiple options and a subset of those modules are reusable.

4.3.4 Flexibility. The flexibility of a product is a measure of its ability to change or adapt to new requirements. Flexibility in this paper is assessed in terms of a product's ability to be reconfigurable and extensible with respect to its architecture. These two components of flexibility are described in Sections 4.3.4.1 and 4.3.4.2.

4.3.4.1 Reconfigurability. The definition of reconfigurability used in this analysis is a product's ability to be assembled or built in multiple configurations according to its architecture. The authors hypothesize in [48] that a measure of reconfigurability of a product needs to take into account more than the number of (re)configurations made possible by module options. The reconfigurability measure (RM), previously developed by the authors [48] is used in this paper as part of the overall VMM, as the $\boldsymbol{Y}$ term. The reconfigurability factor, $\boldsymbol{Y}$, is defined by the four ratios given in Equations 4.6 and 4.7.

$$\boldsymbol{Y} = \begin{bmatrix} y_1 & y_2 & y_3 & y_4 \end{bmatrix} \tag{4.6}$$

$$= \begin{bmatrix} \frac{r}{S} & \frac{r}{t} & \frac{t}{n} & \frac{r}{r_{\text{u.b.}}} \end{bmatrix} \tag{4.7}$$

where: $S = \sum_{1}^{t} s_i =$ Total number of options

$n =$ Number of modules in a product

$t =$ Number of modules with options

r = Number of possible configurations

$r_{\text{u.b.}}$ = Upper bound number of configurations for a given S and t pair

A product is comprised of n modules that are arranged according to a product's architecture. Each of the n modules has one or more ways in which it interfaces with the rest of the product. Each of the n modules may or may not have options to choose from when assembling the product. Each module that has options is counted in an s_i term. The s_i term represents the number of module options for the i^{th} module where $1 \leq i \leq t$ such that t is the total number of modules with options. The sum of the s_i terms is equal to S as shown above.

The mathematical number of reconfigurations made possible by each module option is the product of each of the s_i terms assuming only one option for each module can be chosen, and no pairwise incompatibilities exist. The mathematical number of reconfigurations possible is used in conceptual design analysis as well as when in-depth knowledge of a product is not available. When possible, the actual number of configurations, r_{act}, should be used in an assessment to improve the quality of the RM. In reality, $r_{\text{act}} \leq r$ due to pair-wise incompatibilities between option choices for the i^{th} and j^{th} modules.

Returning to Equations 4.6 and 4.7, the y_1 and y_2 ratios refer to the reconfigurability of a product design. Specifically, y_1 indicates the average number of reconfigurations made possible per option being maintained in inventory. The y_2 ratio is an indicator of the average number of configurations made possible per module with options. Alternatively, this second ratio can be assessed as the average number of reconfigurations made possible per decision point.

Whereas y_1 and y_2 are assessments of the current design, y_3 and y_4 are assessments of how configurable a product design is compared to how reconfigurable it could be given its architecture. The y_3 term represents how much of a product is reconfigurable as well as the maximum t achievable ($t \leq n$) for the given product. The latter

point is important since the number of reconfigurations possible is a function of S and t. The S and t pair imposes an upper bound limit on the number of reconfigurations possible, $r_{\text{u.b.}}$. The last ratio, y_4, is an indication of how much of the $r_{\text{u.b.}}$ a product is achieving. When y_3 and y_4 are equal to one, then the current product design has maximized its reconfigurability for the given S and t pair. If either y_3 or y_4 (or both) is less than one, then the product is not maximizing the number of configurations possible and is not maximizing its reconfigurability. In order to maximize the the number of configurations for a given S and t pair, the standard deviation, σ, of the s_i factors should equal zero (only possible if $S/t \in \mathbb{N}$) or be minimized.

Increasing the number of module options (s_i and hence S) and increasing the number of modules with options (t), increases the combinations possible. The increased number of possible combinations or reconfigurations causes an overall increase in flexibility. If two products with the same S and t are assessed, the product with the lower σ will in general have more configurations possible and is considered more reconfigurable, and through extension, more flexible. In general, to maximize the number of possible configurations, r, for a product, regardless of the S and t values, σ should be minimized. It should be noted that $\sigma = 0$ may not be achievable since $s_i \in \mathbb{N}$, and generally $S/t \notin \mathbb{N}$.

The reusability factor only considers whether or not a module is reused. The reconfigurability factor takes into account the numerous modules that can fit within a product's architecture to form different configurations. This factor implies that increased possible reconfigurations are better than fewer reconfigurations from a modularity viewpoint. That's not to say from a logistics viewpoint, from a configuration management viewpoint, or from an assembly time viewpoint that more is necessarily better. Further, operational or user needs will ultimately determine the number of configurations that are needed.

4.3.4.2 Extensibility. Extensibility is a measure of a product's ability to be extended either through adding functionality or upgrading existing functional-

ity [18]. The latter component, upgrading functionality, is a characteristic of performance and is not assessed in the current measure. However, if a module has built-in architectural options that adds functionality, then it will be included in the extensibility factor. For example, if a navigation module that provides position information is upgraded to increase the position accuracy, it still performs the same function and will not be included in the Z factor. If the same navigation module has built-in architectural options to provide velocity information as well as position information, then the additional functionality will be included in the Z factor. The additional product functionality previously mentioned is referred to here as architectural design options, a, similar to "hooks" and "scars" in software and hardware design respectively [32], that allow for design evolution. They are the functions that will be performed by modules that may or may not exist, but are not in the current inventory of module options. When assembling products with one of the functions in the a term, the product is considered to be built in an engineer-to-order framework. On the other hand, the modules that perform the m functions are built in a configure-to-order framework since the module options are kept in inventory [26]. The extensibility factor in the VMM focuses on capturing the built in architectural design options for adding anticipated functionality to a product as shown in Equation 4.8.

$$Z = \frac{a}{m}, \qquad 0 \leq a \leq m \tag{4.8}$$

where: a = Number of anticipated architectural or functional options
m = Total number of functions

The range for a is assumed to be 0 – m. This range is based on the assumption that a product would not be fielded with less than 50% anticipated functionality. While Z has no hard upper limit, it has a practical limit of 1 based on the previous assumption. This assumption is consistent with the three use cases analyzed. Future use case analysis should be performed to confirm and refine this assumption. Z is a relative order of merit as it is a measure based on a percentage of original primary functionality. It

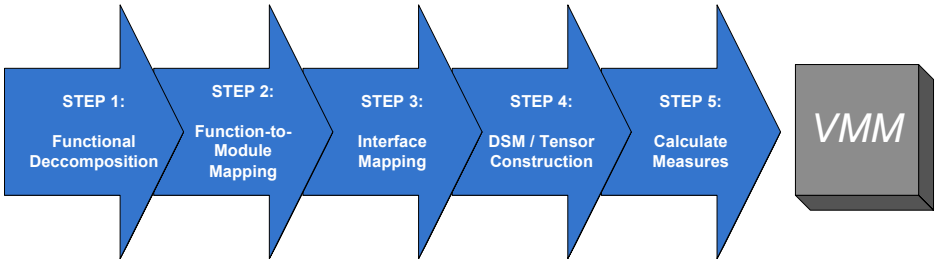


Figure 4.1: Vector Modularity Measure Analysis Process

is important to keep the functions in a at the same level of abstraction as the functions in m and to follow Suh's independence axiom [53]. Assessing extensibility requires in-depth knowledge of a product's design. In cases where reverse-engineering is used to upgrade products, extensibility is harder to evaluate but is still an important benefit of modularity.

4.4 Analysis Process

The analysis process used to calculate a product's VMM begins with a functional model that is accomplished through a functional decomposition of the product. Hirtz et al. [21] extended the previous work by [38] to create a functional basis vocabulary. This vocabulary defines a standardized language to decompose a system into functions and flows to a level of abstraction needed for a given analysis. Three levels of abstraction are used to describe the decomposition: class (or primary), secondary, and tertiary.

Functional decomposition (Step 1) is not new. Functional decomposition is typically done in the early stages of design conceptualization, transforming user requirements into functional requirements [11,43,54]. This functional decomposition or modeling provides an abstract method for understanding and representing the overarching function of a product [21]. Functional decomposition begins at the top level outlining the overarching function of a product. This overarching function is then decomposed into the three levels of abstraction listed above. For purposes of this analysis, the functional decomposition abstraction level stops at the class level.

After functional decomposition has been accomplished at the class level, a product's components and/or modules can be mapped (Step 2) to their corresponding function(s). For existing products, one method to identify module boundaries is to use reverse-engineering. Even though the product exists, clear boundaries may not present themselves, requiring iterations of Step 2 until the boundaries are clearly defined. For new products, identifying the module boundaries also will likely require several iterations of Step 2. Another technique to identify modules is to use the dominant flow heuristic developed by Hirtz *et al.* ([44–46]). This heuristic groups components performing similar functions into modules. Iterations of Step 2 should continue until the function-to-module ratio is 1:1 or is minimized [7]. The module-to-function ratio can be 1:1 or 1:many [7].

Using the identified modules, a bipartite graph can be constructed and used to understand and illustrate the interface mapping (Step 3) between modules. These interfaces, along with the functional decomposition, require in-depth subject or domain knowledge best gleaned from subject matter experts (SMEs). The interfaces between the modules are categorized similarly as was done by [42] into five categories - spatial, informational, material, energy, and structural. A design structure matrix (DSM) can be constructed (Step 4) with the identified modules as row and column labels. The five matrices (by interface type) can then be populated with the interface data. An optional tensor graphic can also be constructed to help illustrate the types and degree of interfaces (see Section 4.5). Finally, assessment of the VMM (Step 5) of the product can begin starting with the degree of coupling, V. V is an assessment of the connectedness/disconnectedness between and among the modules which is also considered the degree of coupling between and among the modules. After subtracting the diagonal entries in each of the five DSMs, Equation 4.4 can be used to solve for V using the off-diagonal entries in the DSM.

The reusability factor, X, can be assessed using the modules identified in the DSM in previous steps. Each of the identified modules, at a minimum, are in the product being assessed. Additionally, the modules could be used in other products

or product families. If a module is used in other products, then it is counted as 1. X is the number of these modules divided by the total number, n, of modules. Keeping track of each module as being reused or unique is straight forward and the reusability factor is easy to calculate even for products with a large number of modules. At this point in the modularity assessment, however, it is worthwhile to keep a list of each module and its associated products. This tracking will aid in the reconfigurability assessment later in the modularity analysis as well as future adaptations of the reusability factor (see Section 4.3.3.

To calculate the reconfigurability factor, $\boldsymbol{Y}$, more knowledge of the product architecture is needed. Each module in the product performs a function or multiple functions. In some cases, more than one option for a module can accomplish these functions and the designer or builder can choose from multiple module options when constructing the product. The number of options for each of these modules needs to be identified, starting with the modules identified in the DSM. The number of modules with multiple options, t, can then be identified as can S, the total number of options for modules with options. Using each of the s_i terms, the number of configurations and the standard deviation, r and σ, respectively, can then be calculated. Lastly, the four reconfigurability ratios can be calculated using Equations 4.6 and 4.7.

Lastly, extensibility, Z, can be calculated. The identification of additional architectural options requires in-depth knowledge of the architecture of the product under analysis. Each additional architectural option is counted and summed into a, which is then factored into Z in Equation 4.8.

4.5 Application

The modularity analysis process is demonstrated using two precision guided munitions (PGMs) in the Air Force inventory. The first PGM is the GBU-24, also referred to as a laser guided bomb or LGB. The second example is the GBU-31, also referred to as the Joint Direct Attack Munition or JDAM. Both munitions, shown in Figures 4.2 and 4.3, can use multiple bomb bodies as the main weapon module. The

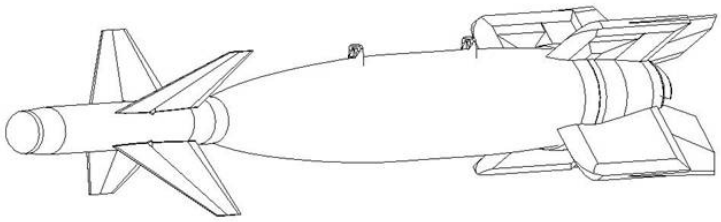


Figure 4.2: GBU-24 (LGB), Mk 84 Variant

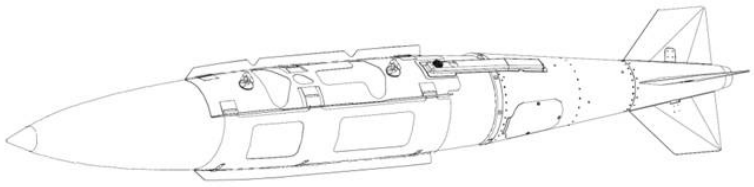


Figure 4.3: GBU-31 (JDAM), Mk 84 Variant

Mk 84, 2000 lb, general purpose bomb was used in this analysis for both PGMs. These two PGM examples were chosen for their similar modular architectures and continued use by the Air Force. Both munitions are mainstays in the Air Force weapons arsenal and have evolved over the years due to changing requirements, upgraded technologies, and employment effectiveness.

4.5.1 GBU-24. The first step in the modularity analysis process is the functional decomposition or function structure. Figure 4.4 depicts the GBU-24 function structure which is very similar for both munitions. The functional basis language was used to represent the functional decomposition. The Appendix gives a lay translation from the functional basis language to a more general language. For example, the overarching function of the weapon is to "Channel: Dumb Bomb," or, move the munition from the aircraft carriage location to the ground target. In this case, "channel" refers to movement from one location to another.

The second step is to map the functions identified in the function structure from Figure 4.4 to modules as shown in Table 4.1. Module identification in the GBU-24 application was straight forward. In less modular designs, this may not be the case.

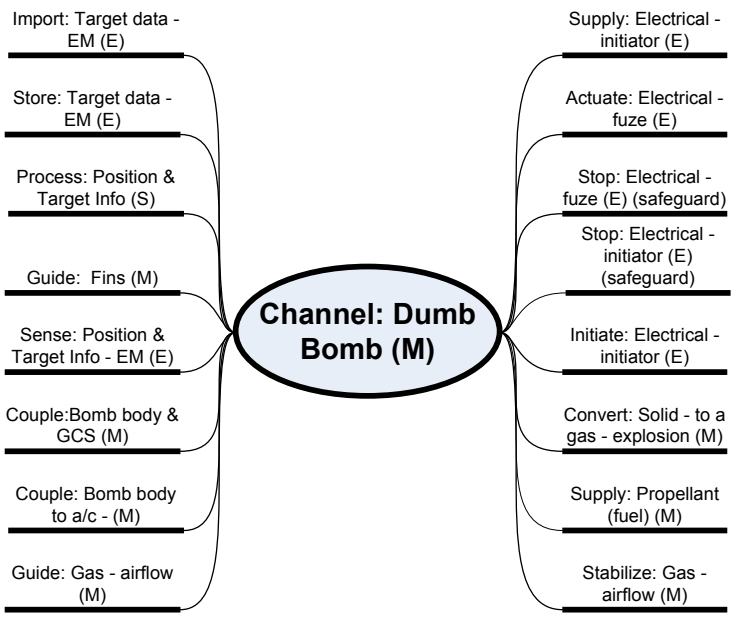


Figure 4.4: GBU-24 Function Structure

It is important to note however, the identification of modules in a product is pivotal to three of the four contributing factors in the modularity vector (V, X, and $\boldsymbol{Y}$).

Once module identification was performed, Steps 3 and 4 of the analysis process can be accomplished either sequentially or in parallel. These two steps were chosen to be accomplished in parallel and the design structure matrix (DSM) was constructed with the GBU-24 modules as row and column labels. For this initial application and for simplification, a binary symmetric matrix was chosen that identifies only that an interface (by type) exists between two modules. After discounting the diagonal entries, the assessment of interfaces between the modules was made. These interfaces, along with the functional decomposition, were accomplished using SMEs and hands-on experience. The DSM/tensor for the GBU-24 is shown in Figure 4.5. Each vertical layer of the tensor represents an interface type. Only half of the tensor plot is shown for simplification since it is symmetrical. Each box represents an interface existence between the two modules identified in the row and column headings in the horizontal axes. The relationship between the modules, by interface type, is given in a typical

Table 4.1: GBU-24 Function to Module Mapping

GBU-24 MODULE	Warhead	Forward Adapter	Guidance Control Section	Airfoil Group	Support Structure	Fuze	Initiator
FUNCTION							
Import: Target data			X				
Store: Target Data			X				
Process: Position & Target Info			X				
Guide: Fins			X				
Sense: Position & Target Info			X				
Couple: Bomb body & aircraft					X		
Couple: Bomb body & GCS		X					
Guide: Gas				X			
Supply: Electrical							X
Actuate: Electrical						X	
Stop: Electrical						X	
Stop: Electrical							X
Initiate: Electrical							X
Convert: Solid						X	
Supply: Propellant	X						
Stabilize: Gas	X						

DSM provide/depend association. Using the tensor plot, it is readily seen which interface types require more or less coupling for a given product.

After developing the GBU-24 tensor, the degree of coupling factor in the Vector Modularity Measure can be calculated. Using the eight modules identified in the DSMs/tensor plot, each module was categorized as unique or reused. After categorization, reusability was assessed. Continuiing on with the reconfigurability assessment, a list of products was created for each module that is used in additional products beyond the GBU-24. The S, t, r_{act}[1], and σ values were then calculated using the lists created for each module in the GBU-24. The values for these parameters, 19, 4, 84, and 2.36 respectively, led to the final calculation of the reconfigurability measure. Lastly, each of the functions identified for the product was summed in the m value. While zero, one, or two fuze modules (and associated functions) can be used to build a complete GBU-24, it was assumed that the build would include one fuze module. The ability to use a second fuze module was considered as additional functionality.

[1]See [48] for the impact on the reconfigurability measure, Y, of using r_{act} versus r for the GBU-24 and GBU-31.

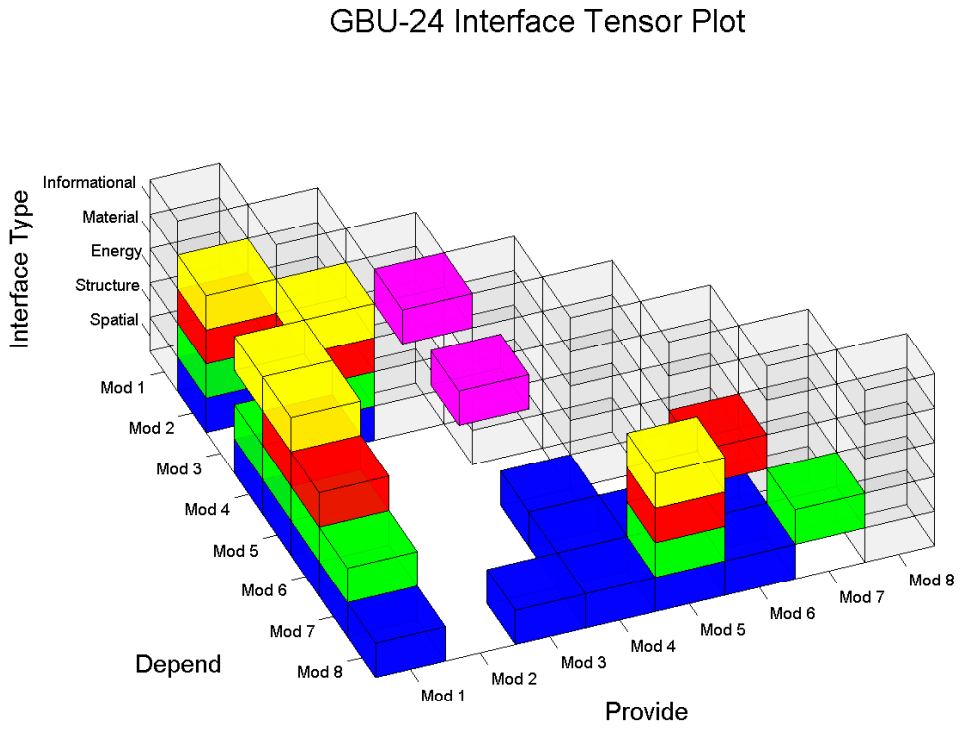


Figure 4.5: GBU-24 Tensor

The additional functionality was captured in the a parameter for extensibility. The equations for all of the modularity factors are summarized in Equations 4.9 and 4.10. The results for the GBU-24 modularity assessment, including the reconfigurability measure, are given in Equations 4.11 and 4.12.

$$\boldsymbol{VMM} = \left[\frac{\sum_{k=1}^{5} \left(\sum_{i=1}^{n} \sum_{j=1}^{n} DSMij_{i \neq j} \right)_k}{5n(n-1)} \quad \frac{n_{mp}}{n} \quad \boldsymbol{Y} \quad \frac{a}{m} \right] \tag{4.9}$$

$$\boldsymbol{Y} = \left[\frac{r_{\text{act}}}{S} \quad \frac{r_{\text{act}}}{t} \quad \frac{t}{n} \quad \frac{r_{\text{act}}}{r_{\text{u.b.}}} \right] \tag{4.10}$$

$$\boldsymbol{VMM}_{\textbf{GBU-24}} = [\ 0.25 \quad 0.63 \quad \boldsymbol{Y}_{\textbf{GBU-24}} \quad 0.06\ ] \tag{4.11}$$

$$\boldsymbol{Y}_{\textbf{GBU-24}} = [\ 4.42 \quad 21 \quad 0.5 \quad 0.17\ ] \tag{4.12}$$

4.5.2 GBU-31. The analysis process for the GBU-31 was also accomplished and is summarized in the following steps and figures. This analysis process is summarized separately from the GBU-24 to outline the complete analysis process from start to finish. The S, t, r_{act}, and σ values used for the GBU-31 were 17, 4, 33, and 4.5, respectively.

STEP 1: Figure 4.6

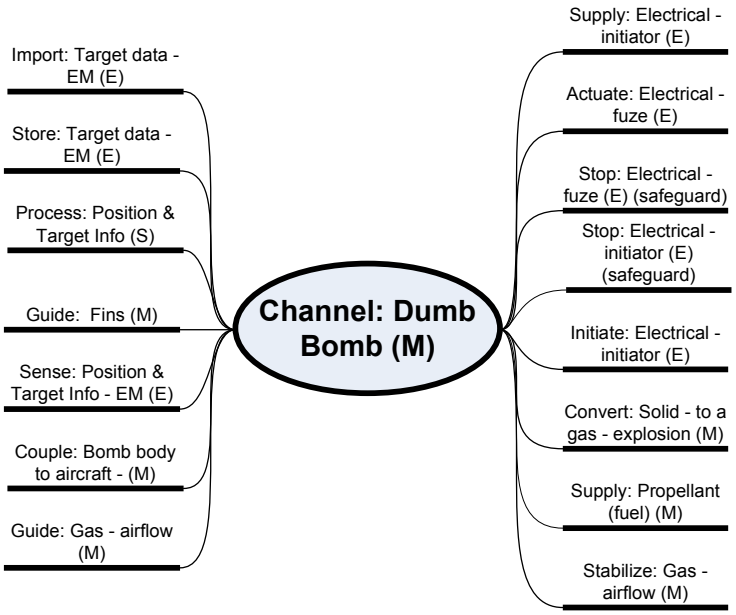


Figure 4.6: GBU-31 Function Structure

STEP 2: Table 4.2

STEPS 3 and 4: Figure 4.7

STEP 5:

$$\boldsymbol{VMM} = \left[\begin{array}{cccc} \dfrac{\sum_{k=1}^{5} \left(\sum_{i=1}^{n} \sum_{j=1}^{n} DSMij_{i \neq j} \right)_k}{5n(n-1)} & \dfrac{n_{mp}}{n} & \boldsymbol{Y} & \dfrac{a}{m} \end{array} \right] \tag{4.13}$$

$$\boldsymbol{Y} = \left[\begin{array}{cccc} \dfrac{r_{\text{act}}}{S} & \dfrac{r_{\text{act}}}{t} & \dfrac{t}{n} & \dfrac{r_{\text{act}}}{r_{\text{u.b.}}} \end{array} \right] \tag{4.14}$$

Table 4.2: GBU-31 Function to Module Mapping

GBU-31 MODULE	Warhead	Guidance Set	Proximity Sensor	Airfoil Group	Support Structure	Fuze	Initiator
FUNCTION							
Import: Target data		X					
Store: Target Data		X					
Process: Position & Target Info		X					
Guide: Fins		X					
Sense: Position & Target Info		X	X				
Couple: Bomb body & aircraft					X		
Guide: Gas				X			
Supply: Electrical							X
Actuate: Electrical						X	
Stop: Electrical						X	
Stop: Electrical							X
Initiate: Electrical							X
Convert: Solid						X	
Supply: Propellant	X						
Stabilize: Gas				X			

Table 4.3: PGM Vector Modularity Measure Results, $\boldsymbol{VMM}$

PGM	Coupl. V	Reusab. X	Reconfig. $\boldsymbol{Y}$	Extens. Z
GBU-24	0.25	0.63	$\boldsymbol{Y}_{\mathbf{GBU\text{-}24}}$	0.06
GBU-31	0.26	0.5	$\boldsymbol{Y}_{\mathbf{GBU\text{-}31}}$	0.00

$$\boldsymbol{VMM}_{\mathbf{GBU\text{-}31}} = [\ 0.26 \quad 0.50 \quad \boldsymbol{Y}_{\mathbf{GBU\text{-}31}} \quad 0.00\ ] \tag{4.15}$$

$$\boldsymbol{Y}_{\mathbf{GBU\text{-}31}} = [\ 1.94 \quad 8.25 \quad 0.5 \quad 0.10\ ] \tag{4.16}$$

4.5.3 Results. The results of the modularity assessment, including the reconfigurability measure, for the GBU-24 and GBU-31 precision guided munitions are shown in Tables 4.3 and 4.4.

Both PGMs perform the same overarching function, to guide or channel a bomb to a target on the ground. Both munitions have similar function structures, modules, and interfaces. This similarity is furthered characterized in the modularity assessment,

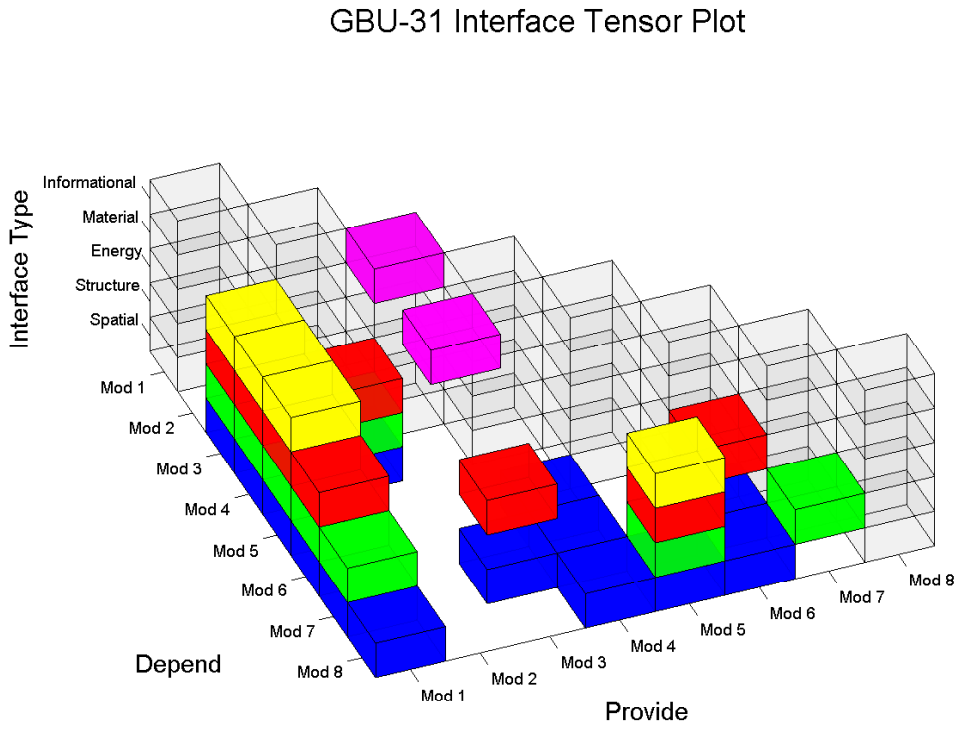


Figure 4.7: GBU-31 Tensor

Table 4.4: PGM Reconfigurability Measure Results, $\boldsymbol{Y}$

PGM	y_1 $\frac{r_{\text{act}}}{S}$	y_2 $\frac{r_{\text{act}}}{t}$	y_3 $\frac{t}{n}$	y_4 $\frac{r_{\text{act}}}{r_{\text{u.b.}}}$
GBU-24	4.42	21	0.5	0.17
GBU-31	1.94	8.25	0.5	0.10

specifically by the degree of coupling, V, and extensibility, Z, factors. Both factors show less than a 6% difference between the two PGMs. This small difference between the PGMs is the result of a two interface difference for V, and a one function difference in Z.

As mentioned previously, earlier modularity measures [23, 34, 42] focus on coupling of either design parameters or interfaces which is referred to as degree of coupling in this paper. Stopping at this point (assessing degree of coupling) in the assessment would yield an insignificant difference in the two products in terms of modularity

and would result in the two designs being relatively equal in terms of modularity. The modularity measure equations given by [23,34] and [42] were calculated for both PGM examples and in all three cases, the modularity measures indicated the same results as the Vector Modularity Measure introduced in this paper, that the GBU-24 is more modular than the GBU-31. While the analysis was consistent as to which munition is more modular, previous measures do not offer the additional insight into the specific benefits of modularity being realized by each munition.

Continuing the analysis process identified in this paper generates the next level of granularity in assessing the differences in modularity between the two PGMs. Specifically, the GBU-24 (Mk 84 variant) is identified as being more reusable and reconfigurable than the GBU-31 (Mk 84 variant). It is important to note that the Mk 84 variants of both munitions were used in the application in this research. The BLU-109 bomb body variant can also be used in both PGMs resulting in different modularity values and conclusions. While both munitions have similar modules, the GBU-24 has one module more than the GBU-24 that is reused in multiple products. The additional module results in the assessment identifying the GBU-24 as more reusable than the GBU-31. Neither weapon, however, is completely reusable ($n_{\text{mp}} < n$) from a modularity viewpoint.

The fuze-initiator constraints imposed on both GBUs as well as other pair-wise constraints hampered the total number of configurations achievable, r. Additionally for the GBU-31, the distribution of the s_i terms achieves the minimum number of configurations possible for the given S and t pair. Since three out of the four ratios in $\mathbf{Y}$ uses r in the reconfigurability factor, the result of the lower r for the GBU-31 extended to the overall reconfigurability as rating lower than the GBU-24. While the analysis showed a small percentage difference between the two degree of coupling measures (less than 6%), this difference could be greater if non-realistic or non-achievable interfaces were eliminated. Ultimately, performing the analysis outlined in this paper identified differences in the two PGMs not previously realized when stopping the analysis after assessing the degree of coupling.

Finally, this analysis process can also be used when comparing the modularity of two designs of the same product or of a product that is being upgraded. Using the tensor plot, for example, modules that are highly coupled to each other and through which interface type(s) can be visualized. If the intentions of a designer or decision-maker are to increase modularity of a product, then the analysis can show contributing factors to the product's modularity and the benefits of increasing the modularity.

4.6 Conclusions

Traditional modularity measures produce one real number, between zero and one, that can be used to compare relative modularity among multiple designs. Whereas these traditional modularity measures focus on coupling, whether between design parameters or interfaces among modules, the measure here builds upon that initial real number. The Vector Modularity Measure presented captures not only the coupling attribute but also the reusability and flexibility attributes. The flexibility attribute is measured in terms of a product's ability to be adaptable to changing requirements which are specifically measures of reconfigurability and extensibility.

The VMM presented can be used to evaluate and compare multiple designs from a modularity viewpoint. Whether these designs are for similar products, the same product, or an upgrade of an existing product, the VMM presented here helps to illuminate various aspects of the product's modularity. This is especially helpful in highlighting where one product design is more modular than another as in the demonstrated case of the PGM. When comparing designs, the various benefits of modularity identified through the analysis process can be taken into account when making design decisions. It is hypothesized that the analysis process can also be used on conceptual designs as well as existing designs but was not attempted as part of this research.

Through the two PGM applications, it was demonstrated that while the two munitions are similar in function structures, modules, and interfaces, they are different

in terms of reusability and reconfigurability. The particular modularity benefits of the GBU-24 over the GBU-31 were only highlighted once the analysis process presented in this paper was accomplished. If gaining the benefits of modularity is a design goal for a product, the VMM presented here helps to evaluate that design and highlight the benefits being realized.

Beyond measuring the four factors that make up the VMM, designers can use each equation of the calculations to determine where improvements to modularity can be made thus increasing the modularity benefits. For example, looking at Equation 4.7, a product with a lower σ will in general result in a higher number of configurations, r, for a given product with the same S and t which will increase the reconfigurability of a product. Another example, using Equation 4.5, is to increase n_{mp} and hence X by using modules in a product that have been used in other products.

Another use of this analysis is to refine the functional decomposition of a product. The second step in the analysis process maps modules to functions. This paper analyzed existing products and used reverse engineering to identify the modules and then map them to the corresponding functions they perform. The function to module mapping highlights where coupling exists between two or more modules. That is, two or more modules are necessary to accomplish one function. This information can then be used to reevaluate the functional decomposition or the module boundaries and hence the interfaces.

Two observations, based on the specific PGM application, are interesting and worthy of further investigation. The first observation is that the GBU-31 had a slightly higher degree of coupling that coincided with it being less reconfigurable than the GBU-24. Using this observation, a second observation is prompted in the form of a question. That is, does higher product complexity tend to discourage higher reconfigurability due to the number of interfaces, the types of interfaces, or a combination thereof?

The DSM/tensor in the degree of coupling factor, V, currently takes a binary approach. The next step, for future research, is to use directional information such that an interface can take on values of 0, 1, or 2 in the DSM/tensor for a given interface type. A "0" would represent no interface exists between two modules. A "1" would represent that a one-way directional interface exists. Lastly, a "2" would represent a bi-directional interface exists. Another future step in advancing the fidelity of this analysis process is to eliminate the non-realistic/non-achievable interfaces from the overall calculation in the V factor. Currently, all matches between modules for each of the interface types are treated as realistic/achievable. Eliminating combinations of modules when calculating the number of reconfigurations would also advance the fidelity of this analysis process. The PGM application in this paper eliminated most, if not all, of the constrained reconfigurations but leaves the process of reconfiguration elimination to the analyst performing the Vector Modularity Measure assessment outlined herein.

V. PnPSat – A Modularity Assessment

5.1 Introduction

When decision makers or designers state they want a spacecraft to be more modular, they are indicating that there are one or more aspects of modularity that they want captured in a new design. The benefits of modularity in product design have been widely recognized and qualitatively captured in [16]. Some of these benefits include changeability, flexibility, reusability, reconfigurability, and extensibility. The Vector Modularity Measure (VMM) used in this paper uses degree of coupling as well as the benefits of modularity, in a vector form, to highlight the contributing factors to a spacecraft's modularity assessment. Each of the equations used in the measure can be used to gain insight into the specific modularity benefits being realized. Designers and decision-makers alike can use this insight to improve existing designs and to aid in overall product selection based on priorities and goals of modularizing a product like a spacecraft.

This paper will briefly review the Vector Modularity Measure developed by the authors in [51]. A brief summary of the analysis process is then given. This summary is followed by the application of the analysis process to a more complex product, PnPSat, than originally presented in [51]. A PnPSat function structure is developed first in the analysis process using a functional basis developed in [21], followed by module identification using a dominant flow heuristic developed in [44–46]. PnPSat functions are mapped to modules and module-to-module interfaces are determined. The VMM is calculated and assessed next for PnPSat followed by a summary of the results. These results are used to propose future design implementations to increase the modularity of PnPSat.

5.2 Background

5.2.1 ORS. Traditional, large, complex satellites typically require 10 to 15 years to develop. These satellites are typically in operational use for 5 to 15 years.

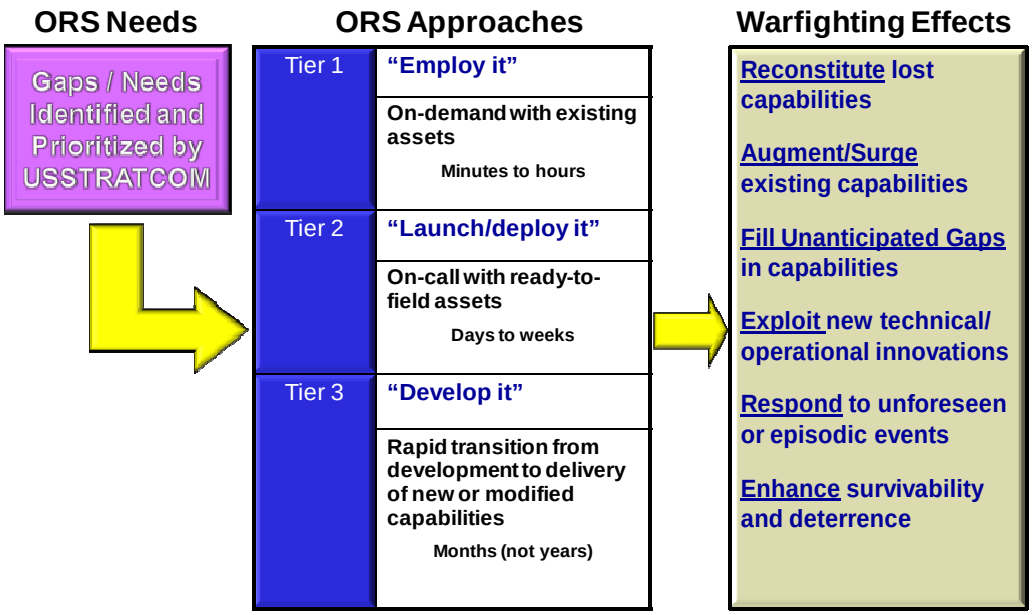


Figure 5.1: ORS 3-Tier Approach [59]

On the other extreme, "simpler," smaller satellites take 12 to 18 months to develop and are operational for 6 months to several years [60]. There still exists a gap of a year or more between when a need is identified by a Joint Force Commander and when a potential need is addressed. Congress recognized this shortfall and called for the creation of the Operationally Responsive Space Office and associated concept development [41]. Operationally Responsive Space was defined by the same report as:

> "...assured space power focused on timely satisfaction of Joint Force Commanders' needs. This definition considers ORS as a subset of space activities designed to satisfy Joint Force Commanders' (JFCs') needs, while also maintaining the ability to address other users' needs, for improving the responsiveness of space capabilities to meet national security requirements."

ORS is focused not on strategic or long-term needs, but on time sensitive needs of the JFCs and other users. In meeting the responsiveness needs of the JFCs, ORS is implemented in a 3-tier approach as depicted in Figure 5.1. The first tier or method of meeting an identified JFC's need is to use current capabilities such as existing

satellites or unmanned aerial vehicles (UAVs). If the need cannot be met, then a tier two approach is used, that is to launch an asset in a time frame of days to weeks using existing assets in inventory. If an on-call asset is not available, then a new capability is sought to meet the identified time-sensitive need. The ORS office is charged with focusing on tier two and tier three approaches. It is thought that modularizing spacecraft will enable the goal time frame of an asset launch on the order of weeks to be realized [2, 30, 33].

5.2.2 Modularity. A method to assess the modularity of modular products was previously developed by the authors in [51]. This method is applied to the Plug-and-Play Satellite (PnPSat) that is being proposed by the Air Force Research Lab (AFRL) in response to the tier two approach to meeting the JFCs' needs. The idea of measuring modularity is not new. A literature review and summary of the various methods for measuring modularity through 2004 is given in [17]. The concepts are further extended in [19] along with comparisons of the various modularity measures based on consistency and sensitivity analyses. Several methods have been proposed since 2004 in [23, 34, 42]. These measures quantify the module-to-module connections, both inter- and intra-module, but ultimately focus on coupling of either design parameters or interfaces. The modularity measure in [34] also accounts for a module's reusability in an exponent term identified as a substitutability factor. None of the measures, however, take into account the assumed benefits of modularity.

5.3 PnPSat

5.3.1 PnPSat Overview. PnPSat is being designed as a modular, reconfigurable small satellite to meet the tier two approach to meeting the JFCs' needs. It has open standards and interfaces, self describing components, and an auto-configuring system [13]. PnPSat performs many of the same functions, on a smaller scale, as traditionally larger satellites. One of the more complex functions that is not performed by PnPSat is propulsion.

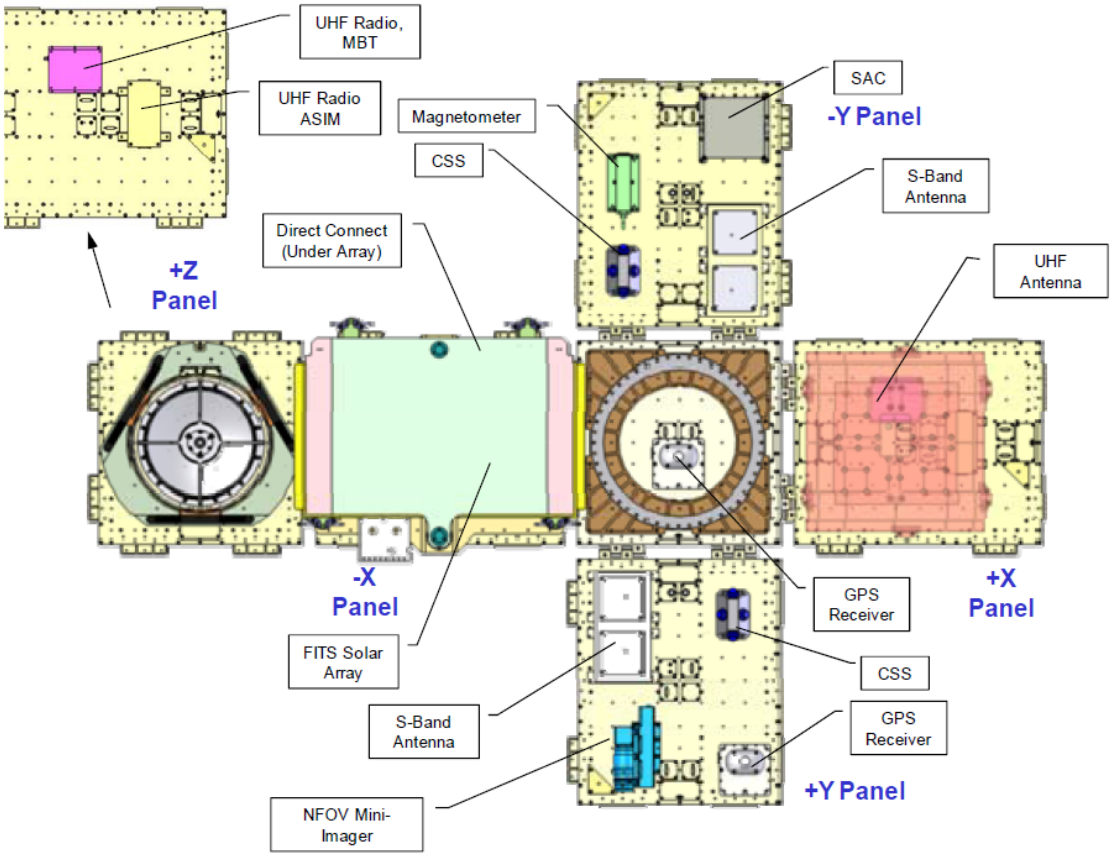


Figure 5.2: Example Exterior of PnPSat [9]

PnPSats are built using a four step process [14]:

1. Spacecraft design
2. Bus assembly and test
3. Payload assembly, alignment, and test
4. Spacecraft integration and test

The first step, spacecraft design, is performed using a software tool called the Mission to Satellite Design Tool (MSDT) that translates user requirements into a parts list for subsequent steps in the build process. The second and third steps of the building process are accomplished in parallel culminating with the integration and testing of the bus and payload in step four.

The proposed assembly, integration, and test flow of PnPSat is given in Figure 5.3. In general, the test function drives the overall time of spacecraft delivery as its intended purpose is to verify adequacy of the spacecraft design and assembly processes [60]. One of the challenges in meeting the ORS goal of a 6- or 7-day spacecraft to become a reality is the reduction of this test cycle. The goal of PnPSat is to have the first type of testing, design adequacy, performed prior to selecting a component or module from inventory during the assembly sequence. This reduces the overall testing function of the components or modules to verifying the assembly process. Whereas the qualification test of a traditional spacecraft is a lengthy and demanding process, a reduced set of functional and environmental tests are being drafted for PnPSat in order to reduce the overall timeline from design to launch [60].

The architecture of PnPSat involve three basic parts [13]: basic spacecraft; spacecraft components; and payload or mission sensors for customization. The basic spacecraft includes the spacecraft structure, the power grids (both main and charging), the space plug-and-play avionics (SPA) infrastructure, and thermal control. The spacecraft components include the autonomous flight software; the quantity of high-performance computing; power generation and storage; guidance, navigation, and

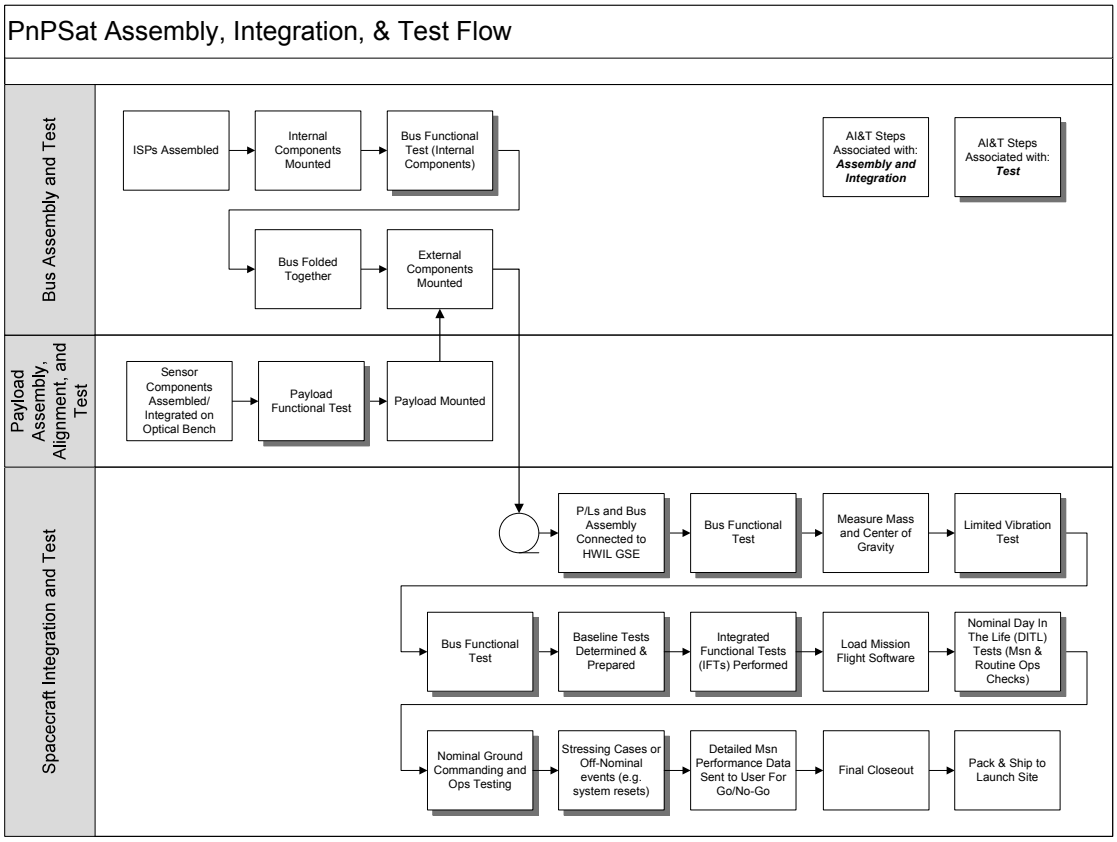


Figure 5.3: PnPSat Assembly, Integration, & Test Flow

control components; and the communications radios for both tactical and TT&C. Thirdly, the mission sensors are customized to the needs of the mission given by the warfighter.

The current PnPSat includes 25 components plugged onto the basic spacecraft. These components include [13]:

- 2 batteries (plus charge control component)
- 2 coarse sun sensor assemblies
- FITS solar array (plus control component)
- GPS radio (2 components)
- TT&C radio (4 antennas)
- 2 packages of HPCOO processors
- 3 magnetic torque rods
- fine sun sensor
- 3 reaction wheels
- intelligent data store
- magnetometer

PnPSat makes use of standardizing mechanical and electrical interfaces in an attempt to reduce the integration timeline. These standard interfaces can accommodate 48 experiments such that the components can be located on either the interior or exterior surfaces. The mechanical interface involves a 5 x 5 cm grid pattern that goes across both the internal and external surfaces of each of the six spacecraft panels. The electrical interface uses a 25-pin micro-D electrical connector that includes up to 4.5A @ 28v, data, time synchronization pulse, test bypass interface, and a single point ground. Finally, the current spacecraft structure is 51 x 51 x 61.2 cm and has a mass of 34.7 kg excluding the launch vehicle adapter [13].

5.4 Vector Modularity Measure

The Vector Modularity Measure (VMM) previously developed by the authors in [51] captures the degree of coupling in a product along with the recognized benefits of modularity in a vector form for further mathematical manipulation. Specifically, the following aspects of modularity are captured in the VMM: degree of coupling between/among the modules in a system; reusability of the modules; and the flexibility of a product to adapt to changing requirements which is assessed in terms of

its reconfigurability and extensibility. Equation (5.1) defines the Vector Modularity Measure, and subsequent subsections briefly describe each of the factors comprising the VMM. The VMM analysis process identified by the authors will be described in the next section.

$$\boldsymbol{VMM} = [\ V \quad X \quad \boldsymbol{Y} \quad Z\ ] \tag{5.1}$$

where: V = Degree of coupling
X = Reusability
$\boldsymbol{Y}$ = Reconfigurability
Z = Extensibility

5.4.1 Degree of Coupling. The first factor in the VMM, the degree of coupling, V, is used to assess how connected/disconnected each module in a product is from each of the other modules. This factor can be used to identify which modules are loosely or highly coupled to the other modules in a product. This assessment can then be used by designers and decision-makers to guide future design decisions regarding which modules to target when trying to improve a product's modularity.

This idea of using degree of coupling is similar to the use of in- and out-degree modularity measures in [42]. Another similar concept is the non-zero fraction (NZF) term in the modularity measure in [23]. The NZF is useful in determining a product's connectedness or coupling and is used in the VMM in this paper. The NZF uses a symmetrical binary design structure matrix (DSM), an n x n matrix where each column and row refers to a module in a product. If an interface exists between two modules, then an X is used to indicate the interface. The NZF is then calculated as the ratio of the total number of non-zero entries to the total number of entries minus the diagonal entries, n.

A DSM is built for each of the five interface types (spatial, informational, etc.). However, the DSMs used herein accommodate directional interfaces by type and hence are generally nonsymmetric. The NZF is calculated for each of the five DSMs using

the same calculation procedure as in Equation (5.2).

$$\text{NZF} = \frac{\sum_{i=1}^{n}\sum_{j=1}^{n} DSMij_{i\neq j}}{n(n-1)} \tag{5.2}$$

where: n = Total number of modules in a product

The degree of coupling, V, between modules is then calculated by summing the five NZF terms over a product and dividing by the total number of interface types (5). The resultant ratio for the degree of coupling factor is the total number of interfaces divided by the total number of possible interfaces minus the diagonal entries over all five interface types. This calculation effectively results in averaging the NZF terms over the five interface types and is shown in Equations (5.3) and (5.4).

$$V = \frac{1}{5}\sum_{k=1}^{5} \text{NZF}_\text{k} \tag{5.3}$$

$$= \frac{\sum_{k=1}^{5}\left(\sum_{i=1}^{n}\sum_{j=1}^{n} DSMij_{i\neq j}\right)_k}{5n(n-1)} \tag{5.4}$$

5.4.2 Reusability. The reusability factor, X, is an assessment of the percentage of modules of a product that are used in other products. In assessing the reusability of a product, modules are sorted into two categories: unique and reusable. This is similar to the categorization of components used in [34]. In the reusability assessment, each module is assigned a binary value indicating whether or not it is used in at least one additional product. The values for the reusable modules are then summed across a product and divided by n to attain the overall percentage of a product's module reuse.

$$X = n_{\text{mp}} \ / \ n \tag{5.5}$$

where: n_{mp} = Number of modules used in multiple products

n = Total number of modules

The binary use of numbering or counting modules in the reusability factor is used in order to avoid cross-coupling factors.

Since reusability is one of the benefits of modularity, the reusability factor highlights to designers what percentage of a product is being reused. In order to claim the benefit of reusability, designers need to avoid using unique module designs where possible. For the analysis herein, assessing whether a product is reused or not is sufficient to glean the benefit of reusability being captured. Knowing to what extent a module is reused, or the number of products containing the module, has potential benefits beyond the assessment in this paper. For example, the greater the number of products using a module the higher the probability that the module is or will become a standard module.

5.4.3 Flexibility. The flexibility of a product is a measure of its ability to change or adapt to new requirements. Flexibility in this paper is assessed in terms of a product's ability to be reconfigurable and extensible with respect to its architecture. These two components of flexibility are described subsequently.

5.4.3.1 Reconfigurability. The definition of reconfigurability used in this analysis is a product's ability to be assembled or built in multiple configurations according to its architecture. Reconfigurability, Y, captures several attributes. These attributes include the number of modules in a product that have multiple options, the total number of options for all modules with options, the number of possible configurations of the product, and how the number of options varies across the modules with options.

The reconfigurability factor, $\boldsymbol{Y}$, is defined by the four ratios given in Equations (5.6) and (5.7). These ratios, used to assess product reconfigurability, were previously developed by the authors in [51].

$$\boldsymbol{Y} = [\ y_1 \quad y_2 \quad y_3 \quad y_4\ ] \tag{5.6}$$

$$= \left[\frac{r}{S} \quad \frac{r}{t} \quad \frac{t}{n} \quad \frac{r}{r_{\text{u.b.}}}\right] \tag{5.7}$$

where: $S = \sum_{1}^{t} s_i =$ Total number of options

$n =$ Number of modules in a product

$t =$ Number of modules with options

$r =$ Number of possible configurations

$r_{\text{u.b.}} =$ Upper bound number of configurations for a given S and t pair

A product is comprised of n modules. Each of the modules may or may not have options to choose from when assembling the product according to its architecture. Each module that has options is counted in an s_i term. The s_i term represents the number of module options for the i^{th} module where $1 \leq i \leq t$ such that t is the total number of modules with options. The sum of the s_i terms is equal to S as shown above.

The mathematical number of reconfigurations made possible by each module option is the product of each of the s_i terms. The mathematical number of reconfigurations possible is used in conceptual design analysis as well as when in-depth knowledge of a product is not available. When possible, the actual number of configurations, r_{act}, should be used in an assessment to improve the quality of the RM. In reality, $r_{\text{act}} \leq r$ due to pair-wise incompatibilities between option choices for the i^{th} module and the j^{th} module.

Returning to Equations (5.6) and (5.7), the y_1 and y_2 ratios refer to the reconfigurability of a product design. Specifically, y_1 indicates the average number of reconfigurations made possible per option being maintained in inventory. The y_2 ratio is an indicator of the average number of configurations made possible per module

with options. Alternatively, this second ratio can be assessed as the average number of reconfigurations made possible per decision point.

Whereas y_1 and y_2 are assessments of the current design, y_3 and y_4 are assessments of how configurable a product design is compared to how reconfigurable it could be given its architecture. The y_3 term represents how much of a product is reconfigurable and hence the maximum t achievable ($t \leq n$) for the given product. The latter point is important since the number of reconfigurations possible is a function of S and t. The S and t pair imposes an upper bound limit on the number of reconfigurations possible, $r_{\text{u.b.}}$. The last ratio, y_4, is an indication of how much of the $r_{\text{u.b.}}$ a product is achieving. When y_3 and y_4 are equal to one, then the current product design has maximized its reconfigurability for the given S and t pair. If either y_3 or y_4 (or both) is less than one, then the product is not maximizing the number of configurations possible and is not maximizing its reconfigurability. In order to maximize the number of configurations for a given S and t pair, the standard deviation, σ, of the s_i factors should equal zero (only possible if $S/t \in \mathbb{N}$) or be minimized.

Increasing the number of module options (s_i and hence S) and increasing the number of modules with options (t), increases the combinations possible. The increased number of possible combinations or reconfigurations causes an overall increase in flexibility. If two products with the same S and t are assessed, the product with the lower σ will in general have more configurations possible and is considered more reconfigurable, and through extension, more flexible. In general, to maximize the number of possible configurations, r, for a product, regardless of the S and t values, σ should be minimized. It should be noted that $\sigma = 0$ may not be achievable since $s_i \in \mathbb{N}$, and generally $S/t \notin \mathbb{N}$.

The reusability factor only considers whether or not a module is reused. The reconfigurability factor takes into account the numerous modules that can fit within a product's architecture to form different configurations. This factor implies that increased possible reconfigurations are better than fewer reconfigurations from a mod-

ularity viewpoint. That's not to say from a logistics viewpoint, from a configuration management viewpoint, or from an assembly time viewpoint that more is necessarily better. Further, operational or user needs will ultimately determine the number of configurations that are needed.

5.4.3.2 Extensibility. Extensibility is a measure of a product's ability to be extended either through adding functionality or upgrading existing functionality [18]. The latter component, upgrading functionality, is a characteristic of performance and is not assessed in the current measure. However, if a module has built-in architectural options that adds functionality, then it will be included in the extensibility factor. For example, if a navigation module that provides position information is upgraded to increase the position accuracy, it still performs the same function and will not be included in the Z factor. If the same navigation module has built-in architectural options to provide velocity information as well as position information, then the additional functionality will be included in the Z factor. The additional product functionality previously mentioned is referred to here as architectural design options, a, similar to "hooks" and "scars" in software and hardware design respectively [32], that allow for design evolution. They are the functions that will be performed by modules that may or may not exist, but are not in the current inventory of module options. When assembling products with one of the functions in the a term, the product is considered to be built in an engineer-to-order framework. On the other hand, the modules that perform the m functions are built in a configure-to-order framework since the module options are kept in inventory [26]. The extensibility factor in the VMM focuses on capturing the built in architectural design options for adding anticipated functionality to a product as shown in Equation 5.8.

$$Z = \frac{a}{m}, \qquad 0 \leq a \leq m \tag{5.8}$$

where: a = Number of anticipated architectural or functional options
m = Total number of functions

The range for a is assumed to be $0 - m$. This range is based on the assumption that a product would not be fielded with less than 50% anticipated functionality. While Z has no hard upper limit, it has a practical limit of 1 based on the previous assumption. This assumption is consistent with the three use cases analyzed. Future use case analysis should be performed to confirm and refine this assumption. Z is a relative order of merit as it is a measure based on a percentage of original primary functionality. It is important to keep the functions in a at the same level of abstraction as the functions in m and to follow Suh's independence axiom [53]. Assessing extensibility requires in-depth knowledge of a product's design. In cases where reverse-engineering is used to upgrade products, extensibility is harder to evaluate but is still an important benefit of modularity.

5.5 Analysis Process

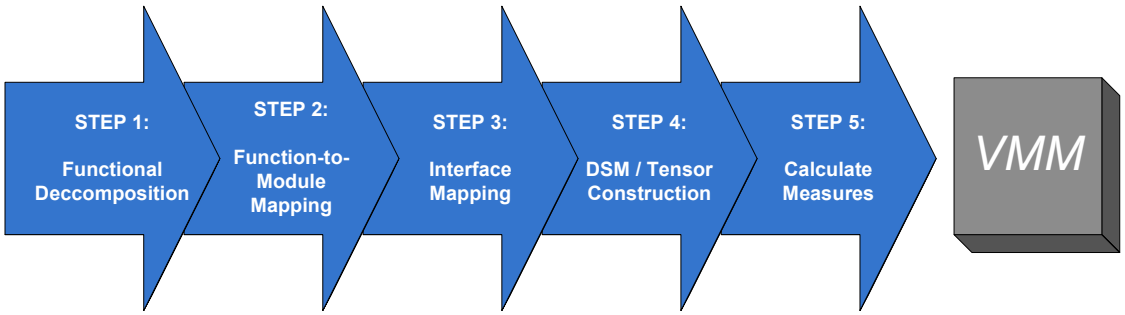


Figure 5.4: Modularity Analysis Process

The analysis process used to calculate a product's Vector Modularity Measure begins with a functional model that is accomplished through a functional decomposition of the product. Hirtz et al. [21] extended the previous work in [38] to create a functional basis vocabulary. This vocabulary defines a standardized language to decompose a system into functions and flows to a level of abstraction needed for a given analysis. Three levels of abstraction are used to describe the decomposition: class (or primary), secondary, and tertiary.

Functional decomposition (Step 1) is not new. Functional decomposition is typically done in the early stages of design conceptualization, transforming user requirements into functional requirements [11,43,54]. This functional decomposition or modeling provides an abstract method for understanding and representing the overarching function of a product [21]. Functional decomposition begins at the top level outlining the overarching function of a product. This overarching function is then decomposed into the three levels of abstraction listed above. For the purposes of this analysis, the functional decomposition abstraction level stops at the class level.

After the class level of functional decomposition has been accomplished, a product's components and/or modules can be mapped (Step 2) to its corresponding function. For existing products, reverse-engineering can be used to identify module boundaries. Even though the product exists, clear boundaries may not present themselves which will require iterations of Step 2 until the boundaries are clearly defined. For new products, identifying the module boundaries also will likely require several iterations of Step 2. Using the identified modules, a bipartite graph can be constructed and used to understand and illustrate the interface mapping (Step 3) between modules. These interfaces, along with the functional decomposition, require in-depth subject or domain knowledge best gleaned from subject matter experts (SMEs). The interfaces between the modules are categorized similarly as was done in [42] into five categories - spatial, informational, material, energy, and structural. A design structure matrix (DSM) can be constructed (Step 4) with the identified modules as row and column labels. The five matrices (by interface type) can then be populated with the interface data. An optional tensor can also be constructed to help illustrate the types and degree of interfaces (see Section 5.6). Finally, assessment of the VMM (Step 5) of the product can begin starting with the degree of coupling, V. V is an assessment of the connectedness/disconnectedness between and among the modules which is also considered the degree of coupling between and among the modules. After subtracting the diagonal entries in each of the five DSMs, Equation (5.4) can be used to solve for V using the off-diagonal entries in the DSM.

The reusability factor, X, can be assessed using the modules identified in the DSM in previous steps. Each of the identified modules, at a minimum, are in the product being assessed. Additionally, the modules could be used in other products or product families. If a module is used in other products, then it is counted as 1. X is the number of these modules divided by the total number, n, of modules. Keeping track of each module as being reused or unique is straight forward and the reusability factor is easy to calculate even for products with a large number of modules. At this point in the modularity assessment, however, it is worthwhile to keep a list of each module and its associated products. This tracking will aid in the reconfigurability assessment later in the modularity analysis.

To calculate the reconfigurability factor, $\boldsymbol{Y}$, more knowledge of the product architecture is needed. Each module in the product performs a function or multiple functions. In some cases, more than one option for a module can accomplish these functions and the designer or builder can choose from multiple module options when constructing the product. The number of options for each of these modules needs to be identified, starting with the modules identified in the DSM. The number of modules with multiple options, t, can then be identified as can S, the total number of options for modules with options. Using each of the s_i terms, the number of reconfigurations, r, and the standard deviation can then be calculated. Next, each of the four reconfigurability ratios can be calculated Equations (5.6) and (5.7).

Lastly, extensibility, Z, can be calculated. The identification of additional architectural options requires in-depth knowledge of the architecture of the product under analysis. Each additional architectural option is counted and summed into a, which is then factored into Z in Equation (5.8).

5.6 Application

The Vector Modularity Measure used in this paper was originally developed and applied using two simpler products, two precision guided munitions (PGMs) [48]. This section extends the original application to a more complex product, PnPSat. Whereas

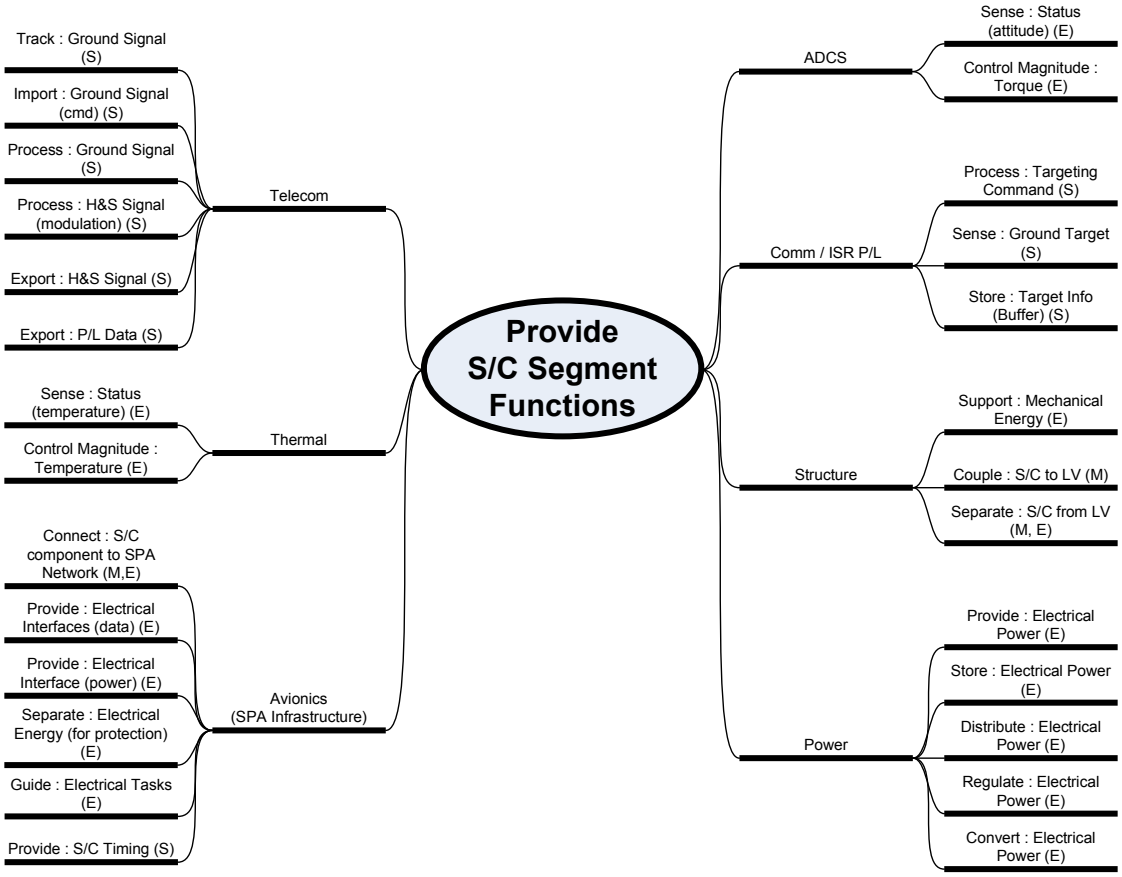


Figure 5.5: PnPSat Function Structure

the precision guided munitions have clear cut modules and interfaces, PnPSat does not. By calculating the VMM for PnPSat, it can then be compared to iterations of proposed modular designs. Does the new modular design increase or decrease the degree of coupling? Are new modules being proposed reusable? Are there constraints on the interfaces that new modules impose that will limit its ability to be reconfigured with certain other modules in PnPSat? All of these questions can be answered as a result of performing the VMM calculations for PnPSat. Another useful aspect of the VMM is that it focuses the designer's attention on the benefits of modularity which are the goals of modularizing a product in the first place.

Applying the modularity analysis process given in the previous section, the first step is the functional decomposition or function structure. An initial iteration of

identifying PnPSat functions was developed first to get the analysis process started. This initial iteration of identified functions were used throughout the first iteration of estimating the VMM. After the first iteration, a second iteration and refinement was made resulting in the function structure for PnPSat given in Figure 5.5. The functional basis language developed in [21] was used to represent the functional decomposition.

The second step is to map the functions identified in the function structure from Figure 5.5 to modules as shown in Table 5.3. Module identification began with the component list for PnPSat from a bill of materials (BOM). Module identification was performed using the dominant flow heuristic developed in [44–46] such that components performing similar functions are grouped into modules. Iterations of module identification should continue until all functions have been mapped to modules in a 1:1 ratio [7]. The module-to-function ratios can be 1:1 or 1:many [7]. If a module performs more than one function, then less modules will be required. Functions that require more than one module should be minimized. In these cases, another iteration of module identification should be performed. If more than one module is required, then the dominant flow heuristic of grouping similar functions into modules was incomplete. In less modular designs, module identification may not be straightforward. It is important to note however, the identification of modules in a product is pivotal to 3 of the 4 contributing factors in the Vector Modularity Measure (V, X, and $\boldsymbol{Y}$). The modules that comprise PnPSat are shown in Table 5.1. The modules indicated with a (‡) in Table 5.1 refer to modules that have multiple options. These modules are listed again in Table 5.2 as s_1, s_2, etc., showing the number of options available for each. PnPSat is comprised of 8 modules ($n = 8$), having $t = 4$ modules with options.

Once module identification was performed, Steps 3 and 4 of the analysis process could be accomplished either sequentially or in parallel. These two steps were chosen to be accomplished sequentially beginning with interface determination between modules by type. For this initial assessment and for simplification, a binary

Table 5.1: PnPSat Modules

Label	Module
Mod 1	Structure
Mod 2	Payload (P/L)‡ (s_1)
Mod 3	Power‡ (s_2)
Mod 4	Thermal
Mod 5	Telecom
Mod 6	Avionics
Mod 7	Attitude determination and control (ADCS)‡ (s_3)
Mod 8	Launch vehicle (LV)‡ (s_4)

‡ represents modules with options

Table 5.2: PnPSat Module Option Distribution

Product	s_1	s_2	s_3	s_4	S	t
PnPSat	4	2	5	3	14	4

Table 5.3: PnPSat Function-to-Module Mapping

PnPSat **MODULE**	Structure	P/L	Power	Thermal	Telecom	Avionics	ADCS
FUNCTION							
Sense : Status (attitude)							X
Control Magnitude : Torque							X
Sense : Status (temperature)				X			
Control Magnitude : Temperature				X			
Track : Ground Signal					X		
Import : Ground Signal (cmd)					X		
Process : Ground Signal					X		
Process : H&S Signal (modulation)					X		
Export : H&S Signal					X		
Export : P/L Data					X		
Connect : S/C component to SPA Network						X	
Provide : Electrical Interfaces (data)						X	
Provide : Electrical Interface (power)						X	
Separate : Electrical Energy (for protection)						X	
Guide : Electrical Tasks						X	
Provide : S/C Timing						X	
Process : Targeting Command		X					
Sense : Ground Target		X					
Store : Target Info		X					
Support : S/C and P/L Modules	X						
Couple : S/C to LV	X						
Separate : S/C from LV	X						
Provide : Electrical Power			X				
Store : Electrical Power			X				
Distribute : Electrical Power			X				
Regulate : Electrical Power			X				
Convert : Electrical Power			X				

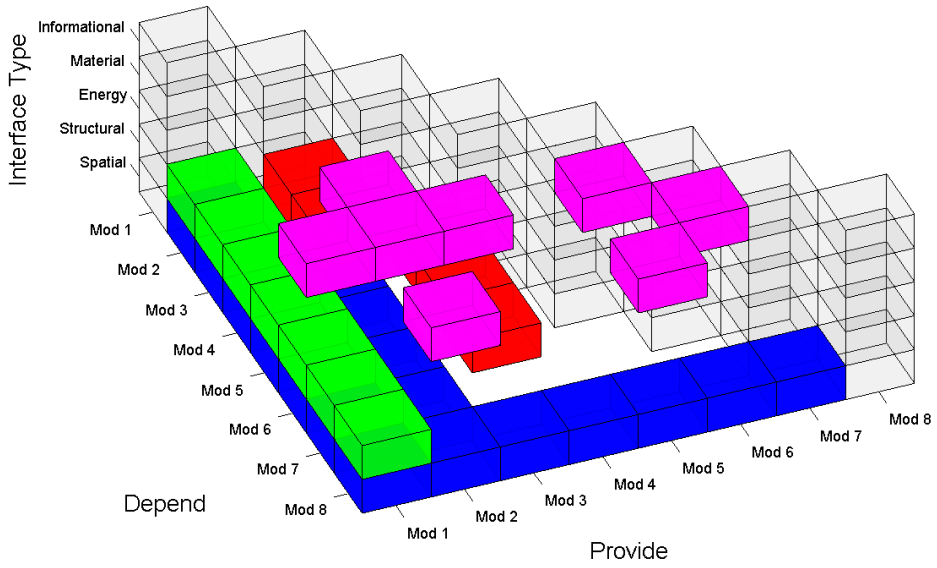


Figure 5.6: PnPSat Tensor

symmetric matrix was chosen that identifies only that an interface (by type) exists between two modules. After discounting the diagonal entries, the assessment of interfaces between the modules was made. These interfaces, along with the functional decomposition, were initially accomplished using PnPSat documentation and later refined using SMEs and hands-on experience. Using the results from Step 3, the design structure matrix (DSM) was constructed with the previously identified PnPSat modules as row and column labels. Using the entries in the DSM, a PnPSat tensor was plotted using MATLAB® for PnPSat as shown in Figure 5.6. Each vertical layer of the tensor represents an interface type. Only half of the plot is shown for simplification since it is symmetrical. Each box represents an interface existence between the two modules labeled as row and column headings in the horizontal axes that correspond to the modules listed in Table 5.1. The relationship between the modules, by interface type, is given in a typical DSM provide/depend association. Using the tensor plot, it is readily seen which interface types require more or less coupling for a given product.

After developing the PnPSat tensor, the degree of coupling factor in the VMM was calculated. Using the eight modules identified in the DSMs/tensor plot, each module was categorized as unique or reused. After categorization, reusability, X, was assessed.

The reconfigurability assessment began by creating a list of products for each identified module from step 2 that is used in additional products beyond PnPSat. The S, t, r, n, $r_{\text{u.b.}}$, and σ values were then calculated using the lists created for each module in PnPSat. The values for these parameters, 12, 4, 72, 8, 81, and 0.82 respectively, led to the final calculation of the four reconfigurability measure ratios.

Lastly, each of the functions identified for the product was summed in the m value. Using information from PnPSat SMEs, the additional functionality, or architectural options, not currently being used was included in the a term and the extensibility factor, Z, was assessed. The equations used in the ***VMM*** are summarized in Equation (5.9) and the results for the PnPSat ***VMM*** are given in Equation (5.10).

$$\boldsymbol{VMM} = \left[\frac{\sum_{k=1}^{5} \left(\sum_{i=1}^{n} \sum_{j=1}^{n} DSMij_{i \neq j} \right)_k}{5n(n-1)} \quad \frac{n_{\text{mp}}}{n} \quad \boldsymbol{Y} \quad \frac{a}{m} \right] \tag{5.9}$$

$$\boldsymbol{VMM}_{\mathbf{PnPSat}} = [\, 0.27 \quad 0.88 \quad \boldsymbol{Y}_{\mathbf{PnPSat}} \quad 0.92 \,] \tag{5.10}$$

$$\boldsymbol{Y} = \left[\frac{r}{S} \quad \frac{r}{t} \quad \frac{t}{n} \quad \frac{r}{r_{\text{u.b.}}} \right] \tag{5.11}$$

$$\boldsymbol{Y}_{\mathbf{PnPSat}} = [\, 6 \quad 18 \quad 0.5 \quad 0.89 \,] \tag{5.12}$$

5.6.1 Results. The results of the Vector Modularity Measure for PnPSat are shown in Tables 5.4 and 5.5. As mentioned previously, earlier modularity mea-

Table 5.4: PnPSat Modularity Measure Results

Product	Coupl. V	Reusab. X	Reconfig. $\boldsymbol{Y}$	Extens. Z
PnPSat	0.27	0.88	see Table 5.5	0.92

Table 5.5: Reconfigurability Measure Results

Product	Configurations / option (y_1)	Configurations / decision point (y_2)	% of Product that is reconfigurable (y_3)	% Maximum possible Configurations (y_4)
PnPSat	6	18	0.5	0.89

sures [23, 34, 42] focus on coupling of either design parameters or interfaces which is referred to as degree of coupling in this paper. Stopping at this point would result in insufficient detail about the modularity of PnPSat. Using the NZF term in [23] would yield an assessment of PnPSat being highly coupled as seen in Figure 5.7. Figure 5.7 shows the five DSMs, by interface type, from Figure 5.6 from a bird's eye view. The NZF in [23] doesn't distinguish between interface types and thus assesses an interface as either existing or not. The resulting degree of coupling, $V = 0.27$, shows a similar result as the simpler precision guided munition products previously assessed in [48]. The degree of coupling assessment shows that spatial interfaces are the dominant type of interfaces and require the most coupling. One reason for this is that the spatial arrangement of modules is influenced by the mission sensor FOV requirements. There are currently no material interfaces; however, this could change with future active cooling or propulsion requirements.

It is currently assumed that each of the modules, with the exception of avionics, are to be used in additional spacecraft. The resulting reusability assessment, $X = 0.88$, yields a highly reusable product. This is a preliminary assessment since PnPSat has not yet flown.

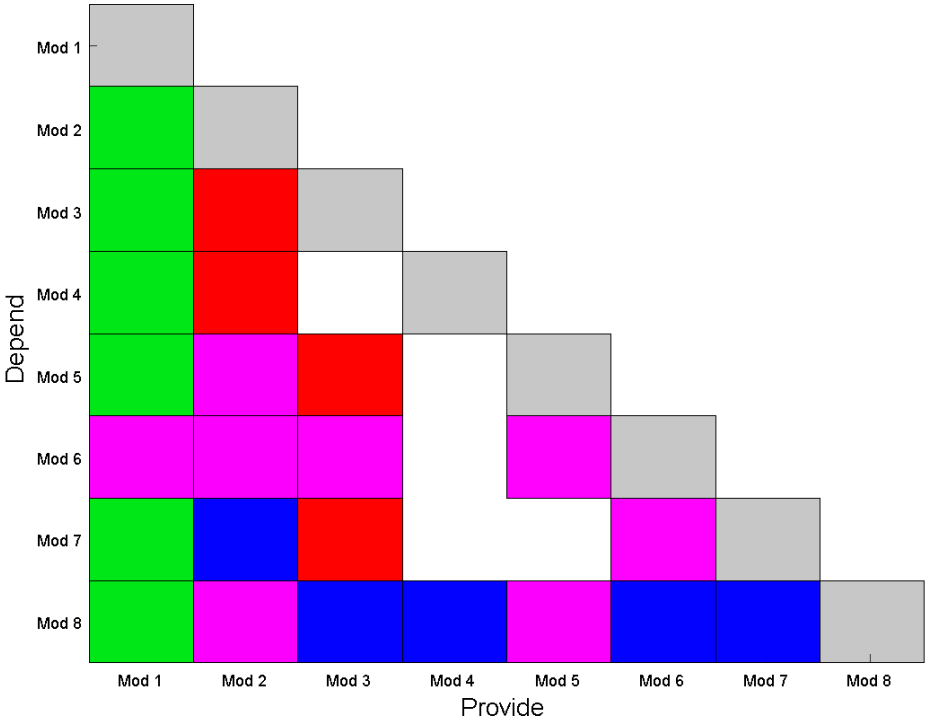


Figure 5.7: Bird's Eye View of the PnPSat Tensor

Looking at the y_3 and y_4 terms in the reconfigurability measure, PnPSat is coming close to maximizing the number of configurations possible for the given S and t pair. One way to increase r would be to examine one of the modules not currently in the t term and look to see if there are additional module options that could be added. Increasing r would in turn result in the y_1 and y_2 terms being increased which is the ultimate goal in increasing the reconfigurability of PnPSat. Currently, it was assumed that there are no constraints on the PnPSat component and module combinations. The r achievable may be decreased in future iterations if it is deemed that this assumption is invalid. One design feature of PnPSat that is instrumental in minimizing the pair-wise constraints is the mounting pattern on the bus structure and the data and electrical endpoints at various panel locations.

One researcher studying the path to making the ORS construct realizable utilizes a constraint-based approach to minimize the number of satellite configurations. This approach is used to quickly evaluate a wide variety of satellite configurations

in order to identify the best configuration to meet the end user's needs [25]. While increasing the number of reconfigurations possible is desirable from a reconfigurability viewpoint, from an evaluation viewpoint, more is not necessarily better.

The designers of PnPSat certainly had extensibility in mind when designing the spacecraft. There are currently 25 functions being performed by PnPSat. Due to built-in options through mechanical and electrical interfaces, an additional 23 interfaces to components or modules exist for adding functionality resulting in a extensibility factor of $Z = 1.92$. This is one of the stronger benefits of modularity being captured by PnPSat.

Finally, this analysis process was used for PnPSat specifically, but it can also be used to compare the modularity of multiple designs of spacecraft trying to fulfill the ORS Tier 2 concept. Using the tensor plot, for example, modules that are highly coupled to each other can be immediately visualized by interface type. Also, if the intentions of a designer or decision-maker is to further increase the modularity of PnPSat, then the analysis can show contributing factors to PnPSat's current modularity and the benefits of increasing the modularity.

5.7 Future Design Implementations

The Vector Modularity Measure and included Reconfigurability Measure can be used to highlight where to make improvements to an existing design to increase modularity and each of the assumed benefits. This section uses the results of the PnPSat analysis and each of the VMM factors to make future design recommendations to further increase modularity.

5.7.1 Degree of Coupling. The first term in the VMM analysis, degree of coupling, shows that of the five interface types, spatial interfaces have the highest number of module-to-module interfaces. This result can be used to identify which module(s) to focus on that have the highest number of spatial interfaces. In reducing the number of spatial interfaces for this target module(s), the overall degree of cou-

Table 5.6: PnPSat Spatial Interfaces

Module	Structure	P/L	Power	Thermal	Telecom	Avionics	ADCS	Ext Sys
Structure	–	X	X	X	X	X	X	X
P/L	X	–	X	X	X	X	X	X
Power	X	X	–					X
Thermal	X	X		–				X
Telecom	X	X			–			X
Avionics	X	X				–		X
ADCS	X	X					–	X
Ext Sys	X	X	X	X	X	X	X	–

pling will also be decreased. For PnPsat, it appears that the design has the minimum number of spatial interfaces. Reducing this set further would require a change in the launch vehicle design. The launch vehicle (or external system) imposes constraints on the spacecraft's size and weight. This translates to spatial interfaces with each module in the PnPSat. Additionally, each structural interface imposes a corresponding spatial interface or constraint. The bus structure module is designed to support all of the subsystems and the subsystems are designed to support the payload. As a result, spatial interfaces exist between the structure and all of the other modules as well as between the payload and all of the other modules. Due to the modular design of each of the remaining modules, no other spatial interfaces exist. This assessment shows the number of spatial interfaces cannot be reduced further without changing designs of the launch vehicle. The goal of reducing the degree of coupling requires studying the other interfaces types to determine if reductions can be made. Currently, no material interfaces exists but there has been consideration given to adding active cooling in the future. If this happens, the number of material interfaces in future design iterations of PnPSat will increase. Structural interfaces are also minimized for PnPSat since the only structural interfaces that exist are between the bus structure and each of the other modules in PnPSat. This leads to the remaining two interface types as the only two types of interfaces that can be further minimized in future PnPSat design iterations. The author is not a PnPSat SME, but at first glance, it doesn't appear that either of these interface types can reduced further. Having stated

that, it appears that the PnPSat design has achieved the minimum set of interfaces due to its modular design approach from the initial concept inception.

The current VMM identifies interfaces between modules by type. It does not currently take into account the number of each of the interface types between modules. While the types of interfaces may not be able to reduced, it is possible that the number of each type of interface can be reduced further. The current VMM measure does not capture this, but it would be worthwhile to investigate this further as it potentially impacts the total time to assemble the satellite. Assembly and checkout time of PnPSat is one area of focus in enabling the ORS 6- or 7-day construct to become a reality.

5.7.2 Reusability. Due to the modular design of the PnPSat, all of the modules are designed to be used in multiple products except the bus structure. PnPSat is scoring extremely high in terms of reusability; seven out of eight modules are reusable. To further increase the reusability measure, the remaining bus structure module could be designed for reuse. There exists another family of PnPSat, PnPSat2, that is in the early stages of design. It currently plans on using a hexagon panel for two of the sides with the remainder sides being rectangular shape panels. In this case, it would be easy to reconfigure the panels back into the PnPSat bus structure shape. If this new PnPSat2 design becomes operational, then the remaining module will be considered reusable and the reusability factor in the VMM will be 1.0.

5.7.3 Reconfigurability. While the goal is to have multiple options for each of the PnPSat modules, the current PnPSat build only has one set of modules. If maximizing reconfigurability is a design goal for PnPSat, then an initial starting point for accomplishing this is to have three options for each of the modules. This will result in an inventory of 21 modules and 3^7 number of reconfigurations possible, assuming no pair-wise constraints exist. Once this happens, then focus can be put on further increasing the number of options for each module. This is strictly from a numerical perspective. In reality, the number of options for a given module will

depend on mission needs. For example, a thermal blanket is the only "module" currently being used to control spacecraft temperature (i.e. no active cooling). If thermal requirements are satisfied using this thermal blanket, then there may not be a need for active cooling to improve the thermal range of the spacecraft. This is consistent with the ORS objectives of meeting mission requirements without trying to optimize a given design as is done in traditional (larger) satellite designs.

Another recommendation to increase the reconfigurability is to increase the number of bus structure modules. This has been considered in the past a possible future direction but has not come to fruition. By adding multiple bus structures, additional flexibility will be gained and potential spatial pair-wise constraints that exists for one bus structure may not exist for another structure. Another consideration to further reduce the number of spatial constraints is to embed the data and power endpoints into each of the panels. Doing this will give more module mounting flexibility and is currently being considered for PnPSat2.

5.7.4 Extensibility. The PnPSat design already includes 23 extension points for adding components or modules, and hence functionality. These extension points provide data and power to potential components or modules. The current PnPSat design does not include a propulsion function. One design recommendation is to add mechanical interfaces (i.e. "scars") that could be used by a propulsion module. Adding these interfaces early in the design can help reduce major design changes in the future. It would also give interface specifications for designing the propulsion module.

5.8 Conclusions

Traditional modularity measures produce one real number, between 0 and 1, that can be used to compare relative modularity among multiple designs. Whereas the traditional modularity measures focus on coupling, whether between design parameters or interfaces among modules, the Vector Modularity Measure here built

upon that initial real number. The VMM presented captures not only the coupling attribute but also the reusability and flexibility attributes. The flexibility attribute is measured in terms of a product's ability to be adaptable to changing requirements which are specifically measures of reconfigurability and extensibility.

The VMM presented can be used to evaluate and compare multiple designs from a modularity viewpoint. Whether these designs are for similar products, the same product, or an upgrade of an existing product, the VMM presented and demonstrated here helps to illuminate various aspects of the product's modularity. This is especially helpful in highlighting where one product design is more modular than another. When comparing designs, the various benefits of modularity identified through the analysis process can be taken into account when making design decisions.

Through the PnPSat application, the original Vector Modularity Measure was reinforced using a more complex product than originally used in the development of the VMM and analysis process. The particular modularity benefits being realized the most by the current design of PnPSat is reusability and extensibility. The benefit of reconfigurability is also being realized to a large extent for the given S and t pair. The number of configurations possible, r, with PnPSat could be increased by increasing t. These benefits of modularity being realized by PnPSat were only highlighted once the VMM analysis process was accomplished.

Beyond measuring the four factors that make up the VMM, designers can look at each equation of the calculations to determine where improvements to modularity can be made thus increasing the modularity benefits. For example, using the reconfigurability measure, a product with a lower σ will result in a higher number of configurations, r, for a given product with the same S and t which will improve three of the four reconfigurability ratios of a product. Another example, using the reusability factor, is to increase n_{mp} and hence X by using modules in a product that have been used in other products.

Another use of this analysis is to refine the functional decomposition of a product. The second step in the analysis process maps modules to functions. The function-to-module mapping highlights where coupling exists between two or more modules. That is, two or more modules are necessary to accomplish one function. This information can then be used to reevaluate the functional decomposition and/or the module boundaries and hence the interfaces. If a module performs more than one function, then less modules will be required. Functions that require more than one module should be minimized. In these cases, another iteration of module identification should be performed. If more than one module is required, then the dominant flow heuristic of grouping similar functions into modules was incomplete.

An observation, based on the previous PGM application, is that one of the PGMs has a slightly higher degree of coupling that coincides with it being less reconfigurable than the other PGM also coinciding with a higher number of pair-wise constraints between modules. If the assumption of no pair-wise constraints for PnPSat proves false, will the number of reconfigurations be reduced? The answer is certainly yes. Another observation previously noted was whether or not higher product complexity tends to discourage higher reconfigurability due to the number of interfaces, the types of interfaces, or a combination thereof. After applying the Vector Modularity Measure to PnPSat, the answer seems to be a combination thereof as well as pair-wise constraints imposed on module-to-module interfaces.

VI. Temporal Constraints

It was hypothesized in the first chapter that increasing the modularity of a system will increase the responsiveness of getting a product to the market or the warfighter in a reduced timeframe. In proving/disproving this hypothesis, the following questions needed to be answered:

- How is product modularity measured?
- What design influences increase product modularity?
- When is a more modular system, as compared to an an integral system, desirable from a design standpoint and from a system goal(s) perspective?
- What affect do temporal constraints on assembly have on the modularity versus design goals relationship?

This research has addressed the first three questions in the research articles in Chapters 3–5. This chapter addresses the fourth question by identifying temporal constraint influences on the modularity versus design goals relationship associated with assembly and checkout (A&CO). These influences are related to the first factor in the Vector Modularity Measure (VMM), degree of coupling. These influences are studied using the two PGM applications to understand the influences as well as to develop a process to analyze these influences that can subsequently be applied to other product applications (e.g. AFRL's Plug-and-Play Satellite (PnPSat)). A brief summary of the results are given for the two PGM applications followed by a detailed application of the analysis process to PnPSat. The analysis process itself and results are given in Section 6.2. Next, the results from the two PGM applications and the PnPSat application are used to identify emerging trends. Lastly, the next steps in characterizing the modularity versus A&CO relationship are provided for the three remaining factors in the VMM as well as other future research recommendations.

6.1 PGM Assembly and Checkout

The GBU-24 and GBU-31 assembly and checkout (A&CO) processes are detailed in Technical Order 11A-1-63, Munitions Assembly Procedures [56]. These procedures guide munition handlers in the assembly and inspection or checkout process. Each step in the A&CO procedure was mapped to the modules required to accomplish the step and the associated interface types. These modules were first identified through the VMM analysis process. Data was provided by the 9^{th} Munitions Squadron (9^{th} MUNS), Air Force Combat Ammunition Center, Beale AFB, CA that identifies minimum, maximum, and average times associated with each step in the A&CO process. Finally, each of the steps in the A&CO process, both main and sub-processes, and their associated interface(s) were mapped to assembly and checkout times given by the 9^{th} MUNS. These mappings are shown for the GBU-24 and GBU-31 in Tables 6.1 and 6.2, respectively.

While clock times were mapped to the process and sub-process steps in the A&CO procedures, they are not given here due to operational concerns. A brief summary of the results are given, however, for the modularity versus assembly and checkout relationship assessment. First, for both PGMs, the warhead module had the highest number of total interfaces that corresponded to the highest amount of handling or clock time among all of the modules in each PGM. Additionally, both PGMs were considered to be the Mk 84 variant of their respective munition, GBU-24 and GBU-31. Another variant of these PGMs uses the BLU-109 warhead. In both cases, the warhead was the module that the rest of the product was built around during the assembly process. This "base" module, the warhead, was handled for the greatest amount of clock time compared with the other modules in the munition. The average clock time to handle each module was 30 seconds longer for the GBU-31 that was less modular and had a slightly higher degree of coupling compared to the GBU-24. The average clock time to handle a GBU-24 module versus a GBU-31 module was 7.21 versus 7.72 minutes. This finding is consistent with the total time to assemble each of the PGMs. The GBU-31 takes slightly longer to assemble than the GBU-24;

Table 6.1: GBU-24 Modules and Interface Types Mapped to Each Step in the A&CO Procedure

		Modules								Interfaces				
Process	Sub-process	WHD	FA	GCU	AG	SS	FUZE	INIT	ACFT	SP	I	M	E	ST
Bomb body prep (BBP)	BBP Sub-process-1	X	X	X	X	X	X	X		X				X
	BBP Sub-process-2					X			X	X				X
Fuze prep (FP)	FP Sub-process-1	X					X			X			X	
	FP Sub-process-2	X					X						X	
	FP Sub-process-3	X					X						X	
Forward adapter install (FAI)	FAI Sub-process-1		X	X						X				X
	FAI Sub-process-2		X	X						X				X
	FAI Sub-process-3	X	X							X				X
	FAI Sub-process-4	X	X							X				X
	FAI Sub-process-5	X	X							X				X
	FAI Sub-process-6	X	X							X				X
	FAI Sub-process-7	X	X	X						X				X
	FAI Sub-process-8	X	X							X				
	FAI Sub-process-9		X	X						X				X
	FAI Sub-process-10		X	X						X				X
	FAI Sub-process-11		X	X						X				X
	FAI Sub-process-12		X	X						X				X
	FAI Sub-process-13	X	X							X				
	FAI Sub-process-14	X					X			X				X
	FAI Sub-process-15	X	X											X
Fuze installation (FI)	FI Sub-process-1	X					X	X					X	
	FI Sub-process-2	X					X	X		X			X	
	FI Sub-process-3	X					X	X		X			X	
	FI Sub-process-4	X					X	X		X			X	
	FI Sub-process-5	X					X			X				X
	FI Sub-process-6	X					X	X		X				
	FI Sub-process-7	X					X	X		X			X	
	FI Sub-process-8	X					X	X		X			X	
	FI Sub-process-9	X					X	X		X			X	
	FI Sub-process-10	X					X	X		X				X
	FI Sub-process-11	X					X						X	

Table 6.1: GBU-24 Modules and Interface Types Mapped to Each Step in the A&CO Procedure (*continued*)

		Modules								Interfaces				
Process	Sub-process	WHD	FA	GCU	AG	SS	FUZE	INIT	ACFT	SP	I	M	E	ST
Wing assembly install (WAI)	WAI Sub-process-1	X			X					X				X
	WAI Sub-process-2	X			X					X				X
	WAI Sub-process-3	X			X					X				
	WAI Sub-process-4	X			X					X				
	WAI Sub-process-5	X			X					X				X
	WAI Sub-process-6	X			X									X
	WAI Sub-process-7	X			X					X				X
	WAI Sub-process-8	X			X					X				
Post assembly inspection (PAI)	PAI Sub-process-1	X	X		X	X	X	X		X			X	X

Modules: WHD=warhead, FA=forward adapter, GCU=guidance control unit, AG=airfoil group, SS=support structure, INIT=initiator, ACFT=aircraft
Interfaces: SP=spatial, I=informational, M=material, E=energy, ST=structural

Table 6.2: GBU-31 Modules and Interface Types Mapped to Each Step in the A&CO Procedure

		Modules								Interfaces				
Process	Sub-process	WHD	PS	GCS	AG	SS	FUZE	INIT	ACFT	SP	I	M	E	ST
Bomb body prep (BBP)	BBP Sub-process-1	X	X	X	X	X	X	X		X				X
	BBP Sub-process-2					X			X	X				X
Fuze prep (FP)	FP Sub-process-1	X					X			X			X	
	FP Sub-process-2	X					X							
	FP Sub-process-3	X					X						X	
Guidance set prep (GSP)	GSP Sub-process-1	X		X	X					X				
	GSP Sub-process-2	X		X	X					X				
	GSP Sub-process-3	X		X	X						X			
Fuze installation (FI)	FI Sub-process-1	X					X	X					X	
	FI Sub-process-2	X					X	X		X			X	
	FI Sub-process-3	X					X	X		X			X	
	FI Sub-process-4	X					X	X		X			X	
	FI Sub-process-5	X					X			X				X
	FI Sub-process-6	X					X	X		X				
	FI Sub-process-7	X					X	X		X			X	
	FI Sub-process-8	X					X	X		X			X	
	FI Sub-process-9	X					X	X		X			X	
	FI Sub-process-10	X					X	X		X				X
	FI Sub-process-11	X					X							
Proximity sensor install (PSI)	PSI Sub-process-1	X	X										X	
	PSI Sub-process-2		X					X					X	
	PSI Sub-process-3		X					X					X	
	PSI Sub-process-4	X	X					X		X			X	
	PSI Sub-process-5	X	X					X		X			X	
	PSI Sub-process-6	X	X							X				X
	PSI Sub-process-7	X	X					X		X			X	
	PSI Sub-process-8	X	X					X		X			X	
	PSI Sub-process-9		X					X					X	
	PSI Sub-process-10		X					X					X	
	PSI Sub-process-11	X	X					X		X				X
	PSI Sub-process-12	X	X					X		X				

Table 6.2: GBU-31 Modules and Interface Types Mapped to Each Step in the A&CO Procedure (*continued*)

		Modules								Interfaces				
Process	Sub-process	WHD	PS	GCS	AG	SS	FUZE	INIT	ACFT	SP	I	M	E	ST
Tail assembly install (TAI)	TAI Sub-process-1	X			X					X				X
	TAI Sub-process-2	X			X					X				X
	TAI Sub-process-3	X			X					X				X
Umbilical cover install (UCI)	UCI Sub-process-1			X	X				X	X	X			
	UCI Sub-process-2			X	X				X	X	X			
	UCI Sub-process-3			X	X				X	X	X			
	UCI Sub-process-4			X	X				X	X	X			
	UCI Sub-process-5			X	X				X	X	X			
Aero surface assembly install (ASAI)	ASAI Sub-process-1				X	X				X				
	ASAI Sub-process-2	X			X					X		X		
	ASAI Sub-process-3	X			X					X		X		
	ASAI Sub-process-4	X			X					X		X		
	ASAI Sub-process-5	X			X					X		X		
	ASAI Sub-process-6	X			X					X		X		
Post assembly inspection (PAI)	PAI Sub-process-1	X	X	X	X	X	X	X		X	X	X	X	X

Modules: WHD=warhead, PS=proximity sensor, GCS=guidance control set, AG=airfoil group, SS=support structure, INIT=initiator, ACFT=aircraft
Interfaces: SP=spatial, I=informational, M=material, E=energy, ST=structural

Table 6.3: PnPSat Jumpstart Exercises and Demonstrations [5]

Trial / Demo	Date(s)	Purpose	A&CO Steps Performed[1]
Trial 1	9-11 Feb 2009	Baseline configuration used; process and procedure validation; A team primary; train B team	1–4, 6-18
RS7 Demo	28–30 Apr 2009	Baseline configuration used; A team primary; team B assists; demo A&CO process during 2009 7^{th} AIAA Responsive Space Conference	1–4, 6–18
Trial 2	11-13 May 2009	Baseline configuration used; personnel investigation – training, skill set, number of personnel; B team primary; team A assists	1–4, 6–18
Trial 3	8-10 Jun 2009	New configuration (Sun-Sync AIS/Imaging); timed trial; refine personnel skill set required	1–4, 6–18
Media Day Demo	23 Jun 2009	New configuration (Sun-Sync AIS/Imaging); timed trial; refine personnel skill set required	1–4, 10
Trial 4	15-18 Dec 2009	Payload A&CO incorporated	1–8, 11–17

[1] From Table 6.4

again, times are not given here due to operational concerns. Lastly, handling times (clock times) associated with spatial and structural interface types were the greatest among the five interface types. The analysis process that was used and developed through analyzing the PGM A&CO procedures and the VMM degree of coupling term is given next in Section 6.2, followed by additional preliminary findings given in Section 6.3.

6.2 PnPSat Assembly and Checkout

Four PnPSat Jumpstart Exercises (time trials) and two rapid A&CO (also called assembly, integration and test (AI&T)) demonstrations have been performed to understand and develop the appropriate level of modularity and checkout (or tests)

required to enable the 6 or 7-day ORS construct to become a reality. Table 6.3 summarizes the Jumpstart Exercises and demonstrations using AFRL's Plug-and-Play Satellite (PnPSat) along with the A&CO steps that were accomplished (the steps are listed in Table 6.4. Fundamentally, a new paradigm in building and testing satellites in preparation for launch was conceived for PnPSat from the beginning during initial design conception. One of the key tenets in reducing the overall timeline for the A&CO phase was to minimize the tests required during and after satellite assembly to make a go/no-go decision for launch. The Jumpstart time trials have been instrumental in understanding and defining this minimum set of tests. The summary of the Jumpstart Exercises in Table 6.3 shows the progression of the time trials along with the purpose of each trial. Initially, trial one was used for validating the proposed A&CO procedures and to use the A team that consisted of personnel that were the most familiar with the build process and modules. Trial one was also used to train a secondary team, the B team, that was not as familiar with the build process and modules. The first of two demonstrations was used to demonstrate the procedures to academia and industry during the 2009 7^{th} AIAA Responsive Space Conference. The second time trial was used to continue to refine the A&CO procedures and to use the B team as the primary lead. After using a baseline configuration for the satellite build process through the second trial, a new configuration was established and refinement of the procedures and necessary skill set continued. The second demonstration was held using this new configuration that only involved mounting the internal and external components and performing a functional test. The last time trial used the new configuration and included, for the first time, the payload build and integration process steps.

The following four-step process, given below, resulted from analyzing the two PGMs described in Section 6.1 and was used to relate modularity and the PnPSat A&CO process. For the current analysis, only the first factor in the VMM was used to characterize this relationship with recommendations for future research using the other three VMM factors in Section 6.3.2. Although the current analysis has been

applied to a sample size of three, it is postulated that a potential use of the results of the four-step process is as a predictive tool in assessing the overall time for assembly and checkout in new product designs. While the small sample size isn't broad enough to be used to show causal relationships between interface types and clock time to assemble modules, potentially it can be used as an indicator as the sample size of analyzed products increases.

1. Identify A&CO procedural steps
2. Associate clock times with each step
3. Identify required modules to perform each step and associated interface types
4. Summarize clock times associated with handling each module and associated interface types

Step 1. The PnPSat A&CO procedures used in the fourth Jumpstart Exercise are based on refinement of previous Jumpstart Exercise procedures. The refined procedures are given in PNP-4025, PnPSat Rapid AI&T Procedures [9], and are graphically depicted in Figure 6.1. These procedures were first given in Section 5.3.1 and are described briefly here. The A&CO procedure begins with the assembly of the bus structure in a "flat-satellite" configuration. This is followed by the installation of the internal components and harnesses being mated to the power and data network. The vehicle is then powered-up on internal power followed by performing a bus functional test to verify internal devices. The bus panels are then folded up and external components are installed, including the solar array and payload items. The RF links are connected next from the vehicle to the ground station. The vehicle is powered-up again, this time with the solar array simulator power followed by another bus functional test. The vehicle is then lifted and the vehicle mass and center of gravity measured. The payload and bus assembly is then placed on a vibration table and sine sweeps are run. An optional single axis random vibration test can also be run. Following the vibration test, the payload and bus assembly are powered-up again and another bus functional test is performed. S-Band and UHF compatibility to the

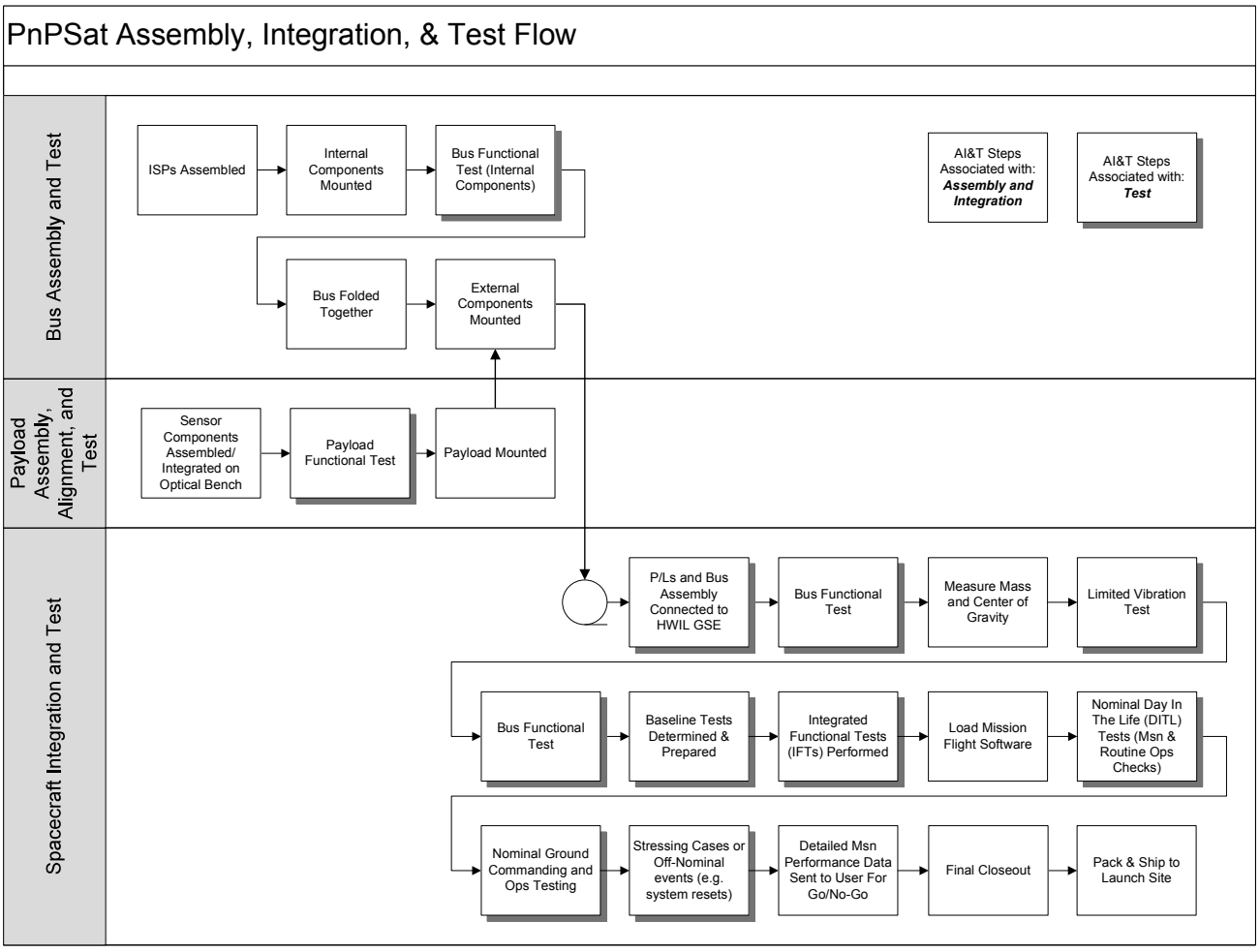


Figure 6.1: PnPSat Assembly, Integration, and Test Flow

ground station simulator are demonstrated next. This is followed by a demonstration of array first-motion deployment along with a full deployment and illumination of the array to verify end-to-end power flow. The mission flight software is loaded next followed by running a nominal "day in the life (DITL)" mission scenario. Included in this scenario is performing nominal ground commanding and operations testing with the RF links. Following the nominal DITL scenario, an off-nominal event or stressing case (e.g. system reset) is performed. Once all data is captured from the A&CO performed thus far, the data is sent to the user for a go/no-go decision. Lastly, the vehicle is prepared for shipment to the launch location where it will be integrated with the launch vehicle.

Step 2. The list of PnPSat A&CO steps were listed next and times were associated with each step from the fourth Jumpstart Exercise as shown in Table 6.4. Not all of the steps were accomplished in the fourth Jumpstart Exercise, times associated with these omitted steps are from the third Jumpstart Exercise and are indicated with a ($^{\mp}$).

Step 3. During the VMM analysis process, one of the steps is module identification. These modules are used here and are mapped to each step in the A&CO procedures. The interface types associated with each step in the A&CO procedures are also mapped. The results for this step for PnPSat are given in Table 6.5. The following question was used in accomplishing these mappings: if the design of a module changes, does it impact the step being accomplished and/or the interface types involved? If the answer was yes, an "X" was used to indicate the module(s) and interface type(s) associated with each step.

Step 4. Tables 6.4 and 6.5 were used to summarize clock time associated with handling each of the PnPSat modules and associated interface types. For PnPSat, the bus structure had the highest number of total interfaces that corresponded to the highest amount of handling time compared to the rest of the modules. Additionally, the bus structure was the module that the rest of the product was built around during

Table 6.4: PnPSat A&CO Times Associated With Each Step [10]

Step	Description	Duration (hours)
1	Assembly of the bus structure in a "flat-satellite" configuration	1.37[a]
2	Install the internal components and mate harnesses to the power and data network	1.37[a]
3	Power up the vehicle on internal power and run a bus functional test to verify internal devices	0.83
4	Fold up the bus panels and install external components, including solar array	1.48
5	Install payload items	3.5[b]
6	Connect the RF links from the vehicle to the ground station	0.25
7	Power-up the vehicle with solar array simulator power and run bus functional test	1.28
8	Lift and measure vehicle mass and center of gravity	0.98
9	Place on vibration table and run sine sweeps and single axis random vibration (Optional)	1.17[∓]
10	Power-up and run bus functional test	0.83[∓]
11	Demonstrate S-Band and UHF Compatibility to ground station simulator	0.33
12	Demonstrate array first-motion deployment	0.08
13	Deploy array and illuminate to verify end-to-end power flow	0.17
14	Load Mission Flight Software	0.25
15	Run nominal Day in the Life scenario and verify L/EO events	3.0[c]
16	Perform nominal ground commanding and ops testing with RF links	3.0[c]
17	Perform stressing cases or off-nominal events (device POR, system resets)	3.0[c]
18	Set vehicle for launch	1.5[∓]

[a] The panels were assembled in parallel and as needed during performance of steps one and two.
[b] The payload was assembled and tested in parallel before mounting externally to the bus structure.
[c] Total time to accomplish steps 15–17.
[∓] Times taken from the third Jumpstart Exercise.

Table 6.5: PnPSat Modules and Interface Types Mapped to Each Step in the A&CO Procedures

		Modules								Interfaces				
Step	Description	STRUC	P/L	PWR	THML	TEL	AV	ADCS	LV	SP	I	M	E	ST
1	Assembly of the bus structure in a "flat-satellite" configuration	X								X				X
2	Install the internal components and mate harnesses to the power and data network	X		X		X	X	X		X				X
3	Power up the vehicle on internal power and run a bus functional test to verify internal devices			X		X	X	X			X		X	
4	Fold up the bus panels and install external components, including solar array	X		X	X	X		X		X				X
5	Install payload items	X	X							X				X
6	Connect the RF links from the vehicle to the ground station	X				X	X				X			
7	Power-up the vehicle with solar array simulator power and run bus functional test		X	X	X	X	X	X			X		X	
8	Lift and measure vehicle mass and center of gravity	X	X	X	X	X	X	X						X
9	Place on vibration table and run sine sweeps and single axis random vibration (Optional)	X	X	X	X	X	X	X					X	X
10	Power-up and run bus functional		X	X	X	X	X	X			X		X	
11	Demonstrate S-Band and UHF Compatibility to ground station simulator					X	X				X			
12	Demonstrate array first-motion deployment	X		X			X			X			X	X

Table 6.5: PnPSat Modules and Interface Types Mapped to Each Step in the A&CO Procedures (*continued*)

		Modules								Interfaces				
Step	Description	STRUC	P/L	PWR	THML	TEL	AV	ADCS	LV	SP	I	M	E	ST
13	Deploy array and illuminate to verify end-to-end power flow	X		X			X			X			X	X
14	Load Mission Flight Software	X	X	X	X	X	X	X			X			
15	Run nominal Day in the Life scenario and verify L/EO events	X	X	X	X	X	X	X			X			
16	Perform nominal ground commanding and ops testing with RF links	X	X	X	X	X	X	X			X			
17	Perform stressing cases or off-nominal events (device POR, system resets)	X	X	X	X	X	X	X			X			
18	Set vehicle for launch	X	X	X	X	X	X	X	X	X				X

Modules: STRUC=bus structure, P/L=payload, PWR=power, THML=thermal, TEL=telecommunications, ADCS=attitude determination and control, LV=launch vehicle

Interfaces: SP=spatial, I=informational, M=material, E=energy, ST=structural

Table 6.6: Clock Times Associated with Handling Modules and Interface Types

	Modules							Interfaces				
Product	MOD 1	MOD 2	MOD 3	MOD 4	MOD 5	MOD 6	MOD 7	SP	I	M	E	ST
GBU-24	15.95	6.00	4.50	8.53	1.92	9.33	4.25	15.95	0.0	0.0	4.83	14.95
GBU-31	19.37	3.42	10.12	12.12	6.25	4.20	5.28	19.98	9.70	2.92	5.87	10.20
PnPSat	13.75	12.51	12.94	10.49	13.53	12.05	12.69	8.10	6.77	0.0	4.36	10.25

the assembly process. This "base" module, the bus structure, was handled for the greatest amount of time compared with the other modules in PnPSat. Lastly, handling times associated with spatial and structural interface types were the greatest among the five interface types. These results, along with the PGM results, were studied to begin to characterize the modularity versus assembly and checkout relationship. The preliminary findings for this relationship are given in Section 6.3.1.

6.3 Modularity and Assembly & Checkout

6.3.1 Preliminary Results and Findings. The assembly and checkout procedures for PnPSat were identified at the main process level. For the purposes of comparison, the PGM assembly and checkout procedures were also mapped at the main process level. The module and interface type mappings previously identified in the sub-process steps of the PGM A&CO procedures were rolled up into the main process mappings resulting in six and nine steps for the GBU-24 and GBU-31, respectively. After performing the four step process to relate modularity and the A&CO process, additional analysis was performed to identify emerging trends. The times that were associated with each step in the assembly process were also used to analyze the clock time associated with handling each module in each of the three products: GBU-24, GBU-31, and PnPSat. These clock times were again used in analyzing the handling time associated with each interface type. These associations are tabulated in Table 6.6. The modules are listed as MOD 1, MOD 2, etc. and refer to the modules in Tables 6.1, 6.2, and 6.5 in the same order given. The units of time for the PGMs and PnPSat are minutes and hours, respectively.

Four trends emerged from the modularity versus assembly timeline preliminary analysis using the GBU-24, GBU-31, and PnPSat. First, ***handling the module in***

Table 6.7: Number of Interfaces Affected During A&CO

	MOD 1	MOD 2	MOD 3	MOD 4	MOD 5	MOD 6	MOD 7
GBU-24	15	8	7	8	7	7	6
GBU-31	17	7	6	10	6	9	7
PnPSat	15	13	9	6	8	10	6

Table 6.8: Number of I/Fs and Clock Times for Handling the ADCS and Avionics Modules

PnPSat Module	Number of I/Fs	Handling Time (hrs)
ADCS	6	12.69
Avionics	10	12.05

a product that had the highest number of interfaces required the greatest amount of clock time. The handling time (or clock time) is used here to capture the total time that a module is handled during the assembly process not accounting for the number of personnel required (i.e. not man-hours). The total assembly time of a product is less than or equal to the handling time due to possible parallel processing of modules, thereby reducing the overall assembly time. An extension of this research could consider mapping man-hours required for each A&CO step as was done, similarly, with clock time. These mappings could then be used to relate man-hours and the components of the VMM.

Due to the analysis of a limited sample size, an extrapolation cannot be made that given any two modules in a product, the one with more interfaces will require a greater amount of handling time. This was demonstrated in the current PnPSat application; the ADCS module has six interfaces whereas the avionics module has ten interfaces. The associated handling times are shown in Table 6.8, showing the ADCS module requiring a greater amount of handling time than the avionics module.

Second, ***the module that was considered the base module was handled for the greatest amount of clock time*** in all three applications as shown in Table 6.6. The PGM variants used in the analysis are built around the Mk 84 warhead and correspondingly, the warhead is handled the most among the other modules in both the GBU-24 and the GBU-31. PnPSat is built around the "flat-satellite" concept

Table 6.9: Number of Module-to-Module Interfaces by Type

	SP	I	M	E	ST
GBU-24	15	1	5	6	8
GBU-31	14	2	4	8	9
PnPSat	18	8	0	5	7

and correspondingly the bus structure is handled for the greatest amount of time compared with the other modules that comprise the product. Preliminarily, there appears to be a correlation between the base module and the module in a product having the greatest number of interfaces. That is, they are the same. If this holds true upon further case studies analyses, then finding one and two will become redundant with each other.

The third "trend" is actually a single data point that resulted from comparing two similar products; the term trend is used loosely here. For similar products, ***the product with the higher degree of coupling will require a higher average clock time to handle each module***. This is a preliminary finding and more studies need to be conducted to confirm this result. An associated masters student is attempting to extend this finding using a discrete event model of the PGM A&CO procedures [37].

The fourth trend that emerged was: ***among the five interface types, required handling or clock times associated with spatial and structural interface types were the greatest***. For the two PGMs, this coincided with the two dominant (in number) interface types for each product. For the PnPSat, it did not; the two dominant interface types (by number) were spatial and informational with structural interfaces as the third dominant interface type by one interface (see Table 6.9). While these emerging trends are not surprising, the research behind this trend identification provides validation that was previously lacking. These results are also being extended by an associated masters student through a discrete model of the PGM A&CO processes whose preliminary findings are consistent with the research

findings herein [37]. That is, for the simpler PGM product, higher product modularity corresponds to a lower total assembly time.

6.3.2 Future Work. One limitation of the research was that the assembly procedures were not standardized, i.e. what was considered a main process versus a sub-process was not clearly defined but used as a general categorization. The analysis herein focused on the concept of a main process in characterizing the interfaces and assembly times. Future iterations should focus on defining and differentiating both main and sub-processes. Once this has been accomplished, then comparisons can be made on the modules and interface types associated with each step to further understand critical path issues during assembly and checkout. Due to the non-standardized grouping of procedural steps into main "chunks" or headings, the current analysis did not consider the number of steps in an assembly and checkout process in the overall assessment. Instead, the research used clock times associated with each step. Additionally for future research, design for assembly (DFA) concepts and techniques should be considered when addressing the process versus sub-process definitions as well as the assembly process itself. Some aspects of the A&CO process that need to be considered using DFA concepts and techniques include: number of personnel required to accomplish each step; skill level required; parallel processing of some steps; identification of the required path for product assembly and its associated timeline (i.e. critical path with respect to time).

In order to associate handling or clock times with each each specific interface, by type, the main process steps need to be broken down into sub-process steps. Clock times are currently associated with these main process steps, they would also needed to be broken down and associated with the sub-process steps. This decomposition of main process steps should continue until all interface types can be mapped to a single or a minimum set of sub-processes. Once this decomposition is accomplished, an assessment can be made of the average handling (or clock) time associated with each interface and by each type of interface.

The "8^{th}" module for each of the products was used in assessing modularity and was used in the modularity versus assembly and checkout relationship but was not used in analyzing product designs for trends in the A&CO process. The interfaces with the modules of a product and the aircraft or launch vehicle (the 8^{th} module) are important to capture but present themselves in a unique category during the analysis. This unique category should be further developed and understood.

In characterizing the modularity versus assembly and checkout relationship, each of the VMM factors should be considered individually for the influences they have on this relationship. The degree of coupling term has been the main focus of this relationship characterization thus far. The spatial and structural interface types had the most influence on the A&CO handling time for the current sample size. A larger sample size of applications needs to be analyzed to further characterize these influences.

Future research should focus on understanding the temporal constraint influences of A&CO and its relationship to modularity using the other three VMM factors (reusability, reconfigurability, and extensibility). For the reusability factor, the more a module is reused, the higher the probability that the module will become a standard interface. Standard interfaces in turn will tend to reduce the number of pair-wise constraints associated with that module and hence the number of reconfigurations possible will increase. While from a reconfigurability viewpoint, a larger number of reconfigurations possible is desirable, it is not clear how this impacts the assembly and checkout process. Is a separate A&CO process maintained for each reconfiguration? How are steps changed in the A&CO process to incorporate the desired configuration? PnPSat is trying to address this electronically and automatically but is in the infancy stage. The extensibility influences are harder to measure than the other VMM factors. If a product leaves "open slots" during the assembly process for adding functionality in the future, does this cause confusion or require extra verification steps thus increasing the total time to assemble a product?

Overall, this research has provided a starting point for characterizing the modularity versus assembly and checkout process. It has also provided some specific areas to focus on to develop this relationship further. Characterizing two other relationships are also worthy of future research, namely: 1. modularity versus mission assurance; and 2. modularity versus cost.

VII. Conclusion

This chapter provides an overall summary of research activities including a summary of key findings. This is followed by recommendations for future research. Lastly, it provides sponsor and collaboration acknowledgement. This final chapter in the main document is followed by two appendices: one summarizing key terms; and one that provides some of the developmental MATLAB® code used to support the research.

7.1 Research Summary

A goal of this research was proving/disproving the hypothesis in the problem statement. Specifically, will increasing the modularity of a system increase the responsiveness of getting a product to the market or the warfighter in a reduced timeframe? Several questions needed to be addressed in proving/disproving this hypothesis:

- How is product modularity measured?
  - Product modularity can be measured using the Vector Modularity Measure (VMM) and the Reconfigurability Measure (RM). The VMM incorporates degree of coupling along with the recognized benefits of modularity.
- What design influences increase product modularity?
  - If increasing a product's modularity is a design goal, then each of the terms in the VMM and RM can be used to identify the target factors that influence modularity and the associated benefits.
- When is a more modular system, as compared to an an integral system, desirable from a design standpoint and from a system goal(s) perspective?
  - A more modular system is desirable when trying to attain the recognized benefits of modularity; specifically: reusability, reconfigurability, and extensibility. It should be noted that difficult performance requirements

and/or tight space/weight constraints may not align with objectives supporting modularity.

- What affect do temporal constraints on assembly have on the modularity versus design goals relationship?
  - A preliminary finding shows a product with a lower degree of coupling (more modular) has a reduced average clock time to handle each module in the product. This finding is preliminary and related research suggest both the number of modules as well as the degree of coupling will impact temporal processing constraints [37]. See Section 7.2 for future recommendations.

The research that was conducted supports the initial hypothesis. See Section 7.2 for recommendations for furthering this research. The remainder of this section describes the resulting conclusions from this research.

Traditional modularity measures produce a real number, between zero and one, that can be used to compare relative modularity among multiple designs. These traditional modularity measures focus on coupling, whether between design parameters or interfaces among modules. After studying the literature on modularity measures, it was determined that these measures were insufficient in capturing the benefits of modularity. A Vector Modularity Measure (VMM) was developed as a result of this research; the VMM builds upon the initial real number of previous modularity measures. The VMM presented captures not only the coupling attribute but also the reusability and flexibility attributes. The flexibility attribute is measured in terms of a product's ability to be adaptable to changing requirements; specifically, measures of reconfigurability and extensibility.

The VMM presented and demonstrated can be used to evaluate and compare multiple designs from a modularity viewpoint. Whether these designs are for similar products, the same product, or an upgrade of an existing product, the VMM presented in this research helps to illuminate various aspects of the product's modularity. This

is especially helpful in highlighting where one product design is more modular than another as in the demonstrated case of the precision guided munitions (PGMs). When comparing designs, the various benefits of modularity identified through the analysis process can be taken into account when making design decisions.

One of the key factors in the VMM is the reconfigurability measure (RM). This research demonstrated that measuring reconfigurability requires more than just calculating the number of reconfigurations possible, r, for a given product. A reconfigurability measure, as presented, must also take into account the total number of options available to a product, S, and the total number of modules with options, t. Using these additional terms in the RM, a designer or decision maker can understand how the current design is measuring compared to how well it could be measuring in terms of the total number of reconfigurations. If a product has one or more modules that do not currently have options, then by focusing on one of these modules, the highest increase in the number of reconfigurations will be realized if one or more options can be added to this module. Similarly, if the number of options for each module with options varies greatly (high standard deviation), then the highest increase in the number of reconfigurations would come when making design changes such that each module with options has the same number of options, if possible, for a given S.

Insight is also gained into how well the current design is measuring in terms of the number of reconfigurations possible per total options in inventory as well as per number of modules with options. The latter ratio shows the number of reconfigurations being realized per decision point that must be made for a given architectural build. The higher the number of decision points the higher the number of potential interfaces and reconfiguration controls will be needed.

Three of the four RM ratios use the number of reconfigurations possible in their calculation. Pair-wise constraints effectively reduce the number of configurations possible and hence the three ratios that use it. Minimizing the pair-wise constraints on modules will help to maximize the achievable number of reconfigurations for the

given distribution of options among the modules. In turn, this minimization of pair-wise constraints will help to increase the reconfigurability, flexibility, and modularity of a product. One way to reduce the number of pair-wise constraints, as demonstrated with PnPSat, is through the use of standard interfaces.

The RM assesses the reconfigurability of modular products strictly from a mathematical viewpoint which stemmed from capitalizing on the benefits of modularity. While this viewpoint is an important starting point in analyzing the reconfigurability of product designs, a system viewpoint must also be considered before making decisions on design changes to a product. Increasing module options offers more possible configurations, but it also requires understanding other ramifications and limitations. Module options have associated costs, logistics, pair-wise constraints, etc., that must be considered. Ultimately, the number of reconfigurations maintained will be a balance between user requirements and cost.

Through the two precision guided munition (PGM) applications, it was demonstrated that while the two munitions are similar in function structures, modules, and interfaces, they are different in terms of reusability and reconfigurability. The particular modularity benefits of the guided bomb unit-24 (GBU-24) over the GBU-31 were only highlighted once the analysis process resulting from this research was accomplished. If gaining the benefits of modularity is a design goal for a product, the Vector Modularity Measure guides the evaluation of that design to highlight the benefits being realized.

Beyond measuring the four factors that make up the VMM and the four ratios that make up the RM, designers can use each component measure to determine where changes can be made to increase modularity and subsequently the benefits being realized. For example, using the RM from Equation 3.4, a product with a lower standard deviation among the modules with options (σ) generally will result in a higher number of configurations for a given product with the same S and t, thus increasing the reconfigurability of a product. Another example, using the numerator

in the reusability equation (Equation 4.5), is to increase the number of modules used in multiple products (n_{mp}) and hence reusability. By examining the product architecture, one can key in on specific areas and even narrow down areas for the greatest increase in reconfigurability and hence modularity. After the initial problem set-up, focus can be applied to the appropriate modules to maximize the increase in the number of reconfigurations. One module may be easy to vary and so already has the highest number of options available. Higher returns, in terms of total number of reconfigurations, may be realized when increasing the number of options for other modules.

Another use of this analysis is to refine the functional decomposition of a product. The second step in the analysis process maps modules to functions. This research analyzed existing products and used reverse engineering to identify modules and map them to the corresponding functions they perform. The function-to-module mapping highlights where coupling exists between two or more modules. That is, two or more modules are necessary to accomplish one function. This information can then be used to reevaluate the functional decomposition or the module boundaries and hence the interfaces. If a module performs more than one function, then less modules will be required. Functions that require more than one module should be minimized. In these cases, another iteration of module identification should be performed. If more than one module is required, then the dominant flow heuristic of grouping similar functions into modules was incomplete.

Through the PnPSat application, the original Vector Modularity Measure was reinforced using a more complex product than originally used in the development of the VMM and analysis process. The analysis process revealed that the particular modularity benefits being realized the most by the current design of PnPSat are reusability and extensibility. The benefit of reconfigurability is also being realized to a large extent for the given S and t pair. The number of configurations possible with PnPSat could be increased by increasing the number of modules with options.

An observation, based on the previous PGM application, was that one of the PGMs has a slightly higher degree of coupling. This coincides with it being less reconfigurable than the other PGM, and also coincides with it having more pair-wise constraints between modules. For simplicity purposes, an assumption was made that there are no pair-wise constraints for PnPSat; as the probability of this being true is extremely low, the number of reconfigurations realizable will certainly be reduced. Another noted observation is whether or not higher product complexity tends to discourage higher reconfigurability. If so, is this due to the greater number of interfaces, the types of interfaces, or a combination thereof. After applying the Vector Modularity Measure to PnPSat, the answer seems to be a combination thereof as well as a result of pair-wise constraints imposed on module-to-module interfaces.

An approach to characterizing modularity versus the assembly and checkout process was developed. Through applying this approach to the two PGMs and to PnPSat, four emerging trends were identified. The first trend revealed that handling of the module in a product that had the highest number of interfaces required the greatest amount of clock time. The second trend showed that the module that was considered to be the base module, the one that the other modules were built around, was handled for the greatest amount of clock time. There is likely some correlation between this trend and the first, as the base module often has more interfaces than other modules. The third "trend" was based on a single data point, thus the term trend is used loosely. The data point showed that for similar products, the product with the higher degree of coupling will require a higher average clock time to handle each module. Lastly, among the five interface types, required handling or clock times associated with spatial and structural interface types were the greatest.

7.1.1 Research Contributions. A summary of the research contributions and associated publications are listed below:

1. A process for accomplishing module identification in conjunction with performing a system decomposition was defined. This process used an existing func-

tional basis language to model the system in order to use one of several techniques to identify module boundaries in a product. The process of mapping functions to modules is defined as an iterative process. Module identification is the starting point for the Reconfigurability Measure and Vector Modularity Measure calculations. This work was submitted as part of [52] and [50].

2. Previous work identified five interface types between components or modules. Other work identified uses a matrix to capture the existence of interfaces between modules. This research extended both approaches to capture the five interface types in a layered or tensor approach. The layers are combined to form a tensor and are captured in a tensor plot developed and coded in MATLAB®. The tensor plot graphically provides the designer with feedback to identify predominant interface types. This work was submitted as part of [52] and [50].

3. In order to begin assessing the modularity of any product, several key steps must be performed to identify modules and capture associated characteristics of each module. This research developed a process to do this in a manner that is repeatable. This process is suitable for more than the VMM and RM metrics; it can also be extended to perform other product characterizations as they are developed. This work was submitted as part of [52] and [50].

4. The current research on measuring product reconfigurability is limited. This research developed a measure to assess product reconfigurability recognizing that the measure should account for more than the mathematical number of reconfigurations possible stemming from module options. This work was submitted as [48] and [49].

5. This work extended the current research on measuring product modularity. It recognized that a modularity measure needs to consider degree of coupling between modules. It also recognized that current measures in the literature did not consider the benefits of the modularity being realized. This work extended the current research on measuring product modularity to capture these benefits

that are used in the development of the Vector Modularity Measure (VMM). The VMM uses degree of coupling, reusability, reconfigurability, and extensibility factors to assess product modularity. This work was submitted as [48], [52] and [50].

6. The utility of the RM and VMM were demonstrated using the GBU-24 and the GBU-31 precision guided munitions. This application resulted in demonstrating that the GBU-24 is more modular than the GBU-31. This application also highlighted reasons why the GBU-24 is more modular than the GBU-31 (e.g. the GBU-24 has less pair-wise constraints than the GBU-31). This work was submitted as part of [52].

7. The applications of the RM and VMM were extended from a simpler PGM example to a more complex example, PnPSat. The analysis process was also applied resulting in recommendations for future design changes to increase the modularity of PnPSat and associated modularity benefits being realized; namely reusability and extensibility. This work was submitted as [50].

8. An initial approach to characterizing the modularity versus temporal constraint relationships was developed and applied to the GBU-24, the GBU-31, and to the PnPSat. Stemming from the preliminary analysis, four emerging trends identified which modules and interface types required the greatest amount of handling time during the assembly and checkout process. Related research further explored the relationship between modularity and temporal processing constraints and is discussed in [37].

7.2 Recommendations for Future Research

The recommendations given in this section focus on improving the fidelity and usefulness of the Vector Modularity Measure. The VMM developed due to this research is a departure from traditional modularity measures that result in a scalar number between zero and one. As such, examining its development will guide future

refinements of the measure thus improving its fidelity and usefulness. Recommendations for improving or extending each of the factors that comprise the VMM are given first. These are followed by recommendations for further characterizing the modularity versus temporal constraints relationship. Lastly, several general recommendations for future research are given.

7.2.1 Degree of Coupling, V. An extension of this analysis is to include the real domain as well. One benefit of this extension is that it accommodates the potential to evaluate design complexity. For example, if a real value is assessed to each interface based on the number of interfaces or the level of complexity for the interface type, Equation 4.4 would need to be slightly modified; the denominator that normalizes the term would need to be removed since an upper limit is no longer apparent based on the number of interfaces.

Advancing the fidelity of the degree of coupling term in the VMM can be accomplished by eliminating the non-realistic/non-achievable interfaces from the overall calculation in the V factor. Currently, all matches between modules for each of the interface types are treated as realistic/achievable which may result in a low degree of sensitivity to changes in module-to-module interfaces.

7.2.2 Reusability, X. For the analysis herein, assessing whether a product is reused or not is sufficient to glean the benefit of reusability being captured. Knowing the extent to which a module is reused, or the number of products containing the module, has potential benefits beyond the assessment in this research. For example, as the number of products that use a given module increases, the probability that the module is or will become a standard module increases. A future adaptation could account for the number of products each module *option* is used in when building variants of a product. Using this adaptation, module options that are peculiar to a product (i.e. not reusable in other products) are highlighted. In the current assessment, however, they are hidden by the overall categorization of "unique/reusable" if a given module has multiple options and a subset of those modules are reusable.

7.2.3 Reconfigurability, $\mathbf{Y}$. Eliminating combinations of modules (due to pair-wise constraints) when calculating the number of reconfigurations would advance the fidelity of this analysis process. The PGM applications in this research eliminated most, if not all, of the constrained reconfigurations but leaves the process of reconfiguration elimination to the analyst performing the Vector Modularity Measure assessment outlined herein.

The RM assesses the reconfigurability of modular products strictly from a mathematical viewpoint which stemmed from capitalizing on the benefits of modularity. This viewpoint is an important starting point in analyzing the reconfigurability of product designs, and should be combined with a system viewpoint along with other viewpoints before making decisions on design changes to a product. Increasing module options offers more possible reconfigurations, but it also requires understanding other ramifications and limitations. Module options have associated costs, logistics, pair-wise constraints, etc., that must be considered. Ultimately, the number of reconfigurations maintained will be a balance between user requirements and cost. A methodology to examine these various viewpoint should be pursued, building on the mathematical viewpoint presented in this research.

7.2.4 Extensibility, Z. The extensibility factor in the VMM is used to compare the built-in architectural design options for upgrading, or adding functionality to a product. The current VMM uses this factor in a limited capacity for comparison between product designs. A future direction is to relate this factor to the number of modules, the number of interfaces, and/or the types of interfaces required to be maintained to ensure the availability of these design options. This factor, together with the other measures, could possibly be used to evaluate redesign effort associated with product improvements. Another future use would be to study how often these options are utilized in upgrading products.

7.2.5 Modularity Versus Temporal Constraints Relationship. The future work recommendations in this subsection were first stated in Section 6.3.2 and are

included here in order to have all recommendations in one chapter/section. One limitation of the research was that the assembly procedures were not standardized, i.e. what was considered a main process versus a sub-process was not clearly defined but used as a general categorization. The analysis herein focused on the concept of a main process in characterizing the interfaces and assembly times. Future iterations should focus on defining and differentiating both main and sub-processes. Once this has been accomplished, then comparisons can be made on the modules and interface types associated with each step to further understand critical path issues during assembly and checkout. Due to the non-standardized grouping of procedural steps into main "chunks" or headings, the current analysis did not consider the number of steps in an assembly and checkout process in the overall assessment. Instead, the research used clock times associated with each step. Additionally for future research, design for assembly (DFA) concepts and techniques should be considered when addressing the process versus sub-process definitions as well as the assembly process itself. Some aspects of the A&CO process that need to be considered using DFA concepts and techniques include: number of personnel required to accomplish each step; skill level required; parallel processing of some steps; identification of the required path for product assembly and its associated timeline (i.e. critical path with respect to time).

In order to associate handling or clock times with each each specific interface, by type, the main process steps need to be broken down into sub-process steps. Clock times are currently associated with these main process steps; they would also need to be broken down and associated with the sub-process steps. This decomposition of main process steps should continue until all interface types can be mapped to a single or a minimum set of sub-processes. Once this decomposition is accomplished, an assessment can be made of the average handling (or clock) time associated with each interface and by each type of interface. Additionally, the number of personnel required can be associated with each interface and by each type of interface as well.

The "8^{th}" module for each of the products was used in assessing modularity and was used in the modularity versus assembly and checkout relationship, but it was not used in analyzing product designs for trends in the A&CO process. The interfaces with the modules of a product and the aircraft or launch vehicle (the 8^{th} module) are important to capture but present themselves in a unique category during the analysis. This unique category should be further developed and understood.

In characterizing the modularity versus assembly and checkout relationship, each of the VMM factors should be considered individually for the influences they have on this relationship. The degree of coupling term has been the main focus of this relationship characterization thus far. The spatial and structural interface types had the most influence on the A&CO handling time for the current sample size. A larger sample size of applications needs to be analyzed to further characterize these influences.

Future research should focus on understanding the temporal constraint influences of A&CO and its relationship to modularity using the other three VMM factors (reusability, reconfigurability, and extensibility). For the reusability factor, the more a module is reused, the higher the probability that the module will become a standard interface. Standard interfaces in turn will tend to reduce the number of pair-wise constraints associated with that module and hence the number of reconfigurations possible will increase. While from a reconfigurability viewpoint, a larger number of reconfigurations possible is desirable, it is not clear how this impacts the assembly and checkout process. Is a separate A&CO process maintained for each reconfiguration? How are steps changed in the A&CO process to incorporate the desired configuration? PnPSat is trying to address this electronically and automatically but is in the infancy stage. The extensibility influences are harder to measure than the other VMM factors. If a product leaves "open slots" during the assembly process for adding functionality in the future, does this cause confusion or require extra verification steps thus increasing the total time to assemble a product?

7.2.6 General Recommendations. Two observations, based on the specific PGM application, are interesting and worthy of further investigation. The first observation is that the GBU-31 had a slightly higher degree of coupling that coincided with it being less reconfigurable than the GBU-24. Using this observation, a second observation is prompted in the form of a question. That is, does higher product complexity tend to discourage higher reconfigurability due to the number of interfaces, the types of interfaces, or a combination thereof? Further investigation of the PGM applications and other product domain applications are warranted in order to answer these questions.

The VMM was applied to only one of the proposed concept satellites, PnPSat, attempting to fulfill the ORS 6- or 7-day satellite objective. Future research should focus on applying the measure to similar products that are attempting to fulfill the ORS construct. Another future direction of this research is to continue to refine the required testing deemed necessary to space qualify the spacecraft (with payload) while reducing the overall design to launch timeline.

The benefits of modularity were addressed by this research. The benefits included in the research are agreed upon generalized benefits of modularity. A future direction of this application is to identify any additional benefits not captured herein. Also, specific application domains may have additional benefits not identified in the general case that may be worthy of study and employment.

Having identified the benefits of modularity, one of the next steps in extending this research should be to compare the relationship between modularity and the product assembly and checkout timeline. This research has given a starting point to characterize this relationship. Studying the interfaces by types and degrees may yield a noticeable time difference required during assembly and checkout associated with each type of interface. If this proves true, designers can focus on eliminating or at least minimizing these types of interfaces, thus reducing the overall time for assembly and checkout. The preliminary analysis from Chapter 6.3 indicates that

spatial and structural interfaces are dominant drivers in the overall assembly and checkout process. These findings were based on a limited sample size and further analysis should be performed to confirm them.

A method proposed by Abdelsalam et al. [1] seeks to optimize the sequencing of design activities that affects both time and cost. This method was not attempted during this research but is stated here for consideration for future research since both time and cost are key contributors in getting products to the market or the warfighter in responsive timeframes. Abdelsalam et al. recognize that getting a product to the market (or the warfighter) in a timely fashion can make all the difference in whether or not the product is competitive (or useful). Abdelsalam et al. look at a method that is an excel-based framework to obtain an optimum sequence of design processes (comprised of activities) within a product development project. They seek to minimize the project total iterative time and cost using stochastic estimates for time and cost for the various activities in the product development.

Overall, this research has provided a starting point for characterizing the modularity versus assembly and checkout process. It has also provided some specific areas to focus on to develop this relationship further. Characterizing two other relationships are also worthy of future research, namely: 1. modularity versus mission assurance or probability of success; and 2. modularity versus cost.

7.3 Sponsor and Collaboration Acknowledgement

This research was sponsored in part by the Operationally Responsive Space Office, Kirtland AFB, NM, and supports their vision of improving the nation's means to develop and employ space capabilities in shortened timeframes, recognizing the importance of innovation and responsiveness in delivering space capabilities. This research also resulted in part due to the collaboration efforts between the author and the Space and Missile Systems Center, Development Planning Directorate (SMC/XR), Los Angeles AFB, CA; the Air Force Research Laboratory, Space Vehicles

Directorate, Kirtland AFB NM; and the 9^{th} Munitions Squadron (9^{th} MUNS), Air Force Combat Ammunition Center, Beale AFB, CA.

Appendix A. Glossary of Key Terms

A.1 Modularity Definitions

- **Modularity** - is the grouping of components into well defined entities, such as modules or sub-assemblies, that can be further described by the interfaces between them.
- **Interface (I/F)** - the spatial, informational, material, energy, or structural connection or coupling of one module to another module within a product [42]. I/F types given below are defined similarly as in Sosa et al. [42].
  - **Spatial I/F** - physical adjacency for alignment, orientation, serviceability, assembly or weight.
  - **Informational I/F** - transference of signals or controls.
  - **Material I/F** - transference of airflow, oil, fuel, or water.
  - **Energy I/F** - transference of heat, vibration, electric, or noise energy.
  - **Structural I/F** - transference of loads or containment.
- **Module** - a group of components or sub-assemblies that perform one or more functions
- **Flexibility** - a product's ability to change or adapt to new requirements; it is measured in terms of a product's ability to be reconfigurable and extensible.
- **Reusability** - the ability of modules within a product to be used in at least one other product variant.
- **Reconfigurability** - the ability to interface, with slight modification (or reconfiguration), with additional external systems.
- **Extensibility** - built in architectural options for upgrading, or adding functionality to a product.

- **Function** - a technical process involving energy, material and/or signals being converted and/or channeled.
- **Flow** - material, signal, and/or energy that can be converted or channeled.

A.2 PGM Function Definitions

Precision guided munition function definitions (from Figures 4.4 and 4.6) are listed in this appendix in operational terms. Each function-flow pairing follows the basic format given by [21] and which is shown in a slightly modified format given in Equation A-1. In this format, the function is an action verb from the functional basis terminology, the flow is a noun, and in () is the flow type. The three flow types are Material (M), Energy (E), and Signal (S).

$$\text{Function : flow (flow type)} \tag{A-1}$$

Channel : Dumb bomb (M) - Channel indicates movement from one location to another; it is used here to represent the movement of the munition from the aircraft to the target.

Import : Target data - EM (E) - Import is used to indicate or describe a flow entering the system boundary; target data is imported or downloaded from the aircraft to the munition guidance set.

Store : Target data - EM (E) - Store refers to the accumulation of a flow; target data is stored in memory in the munition guidance set for later use in munition guidance processing.

Process : Position and target information (S) - Process refers to submitting information to a treatment or method having a set number of operations or steps; the munition guidance set processes the position and target data or information to continually update the current position, desired position, overall flight path, control inputs, and fuzing timing.

Guide : Fins (M) - Guide is a secondary function (from channel) that indicates the direction of flow along a specific path; it is used here to indicate the reception of a control input from the guidance set that in turn provides an input to the mechanism to rotate the fins to achieve the desired flight path.

Sense : Position and target information (E) - Sense is to perceive or become aware of a flow; it is used here in the traditional way of sensing an energy source, the laser return for the GBU-24 and the GPS signal for the GBU-31, that is used in determining relative position to the target.

Couple : Bomb body to aircraft (M) - Couple is a secondary function (from connect) that indicates joining or bringing together flows such that the members are still distinguishable from each other; the use of coupling is also used in the traditional way, here it represents the attachment or mating of the munition with the aircraft.

Couple : Bomb body to GCS (M) - Couple is a secondary function (from connect) that indicates joining or bringing together flows such that the members are still distinguishable from each other; the use of coupling is also used in the traditional way, here it represents the joining of the bomb body with the guidance control section.

Guide : Gas - airflow (M) - Guide is a secondary function (from channel) that indicates the direction of flow along a specific path; it is used here to indicate the guidance of the airflow around the actuators (e.g. fins) to achieve the desired flight path.

Actuate : Electrical - fuze (E) - Actuate refers to the commencing of energy, signal, or material in response to an imported control signal; it is used here to represent the commencing the electrical signal that will ignite the explosive material in the munition.

Convert : Solid - to a gas - explosion (M) - Convert is used to represent the conversion of one form of flow to another; the conversion used here is the explosive material or fuel that is ignited and converted into explosive energy.

Stop : Electrical - fuze (E) (safeguard) - Stop is a secondary function (from control magnitude) used to indicate the ceasing, preventing or transferring of a flow; it is used here to represent the prevention of inadvertent fuzing which is one of two safeguard mechanisms.

Supply : Electrical - initiator (E) - Supply is a secondary function (from provision) used to indicate the provision of a flow from storage; upon release of the munition, the initiator is activated and electrical energy is generated, stored and supplied to the fuze.

Stop : Electrical - initiator (E) (safeguard) - Stop is a secondary function (from control magnitude) used to indicate the ceasing, preventing or transferring of a flow; it is used here to represent the prevention of inadvertent charging of the initiator that would result in fuzing which is one of two safeguard mechanisms.

Initiate : Electrical - initiator (E) - Initiate is a secondary function (from control magnitude) that refers to the commencing of energy, signal, or material in response to an imported control signal; upon release of the munition, the initiator is activated and electrical energy is generated that is subsequently supplied to the fuze.

Stabilize : Gas - airflow (M) - Stabilize is a secondary function (from support) to indicate the prevention of a flow from changing course or location; the strakes are used to stabilize the airflow around the bomb body and to help guide the airflow towards the aft of the bomb body.

Supply : Propellant - fuel (M) - Supply is a secondary function (from provision) used to indicate the provision of a flow from storage; the explosive material is carried or housed within the bomb body.

Space Plug-n-Play Avionics (SPA) The Operationally Responsive Space office has signed onto the AFRL definition of Space Plug-n-Play Avionics as a means for achieving modularity for their spacecraft. SPA is defined as an interface-driven standard that promotes the rapid development of spacecraft payloads and buses. The standard is comprised of an open system framework that combines commercial standards (e.g. USB) with hardware and software extensions necessary for modern real-time embedded systems [31].

SPA Concept: Architectural Overview The SPA approach an architecture of choosing components (such as sensors or actuators) in a pick and choose fashion that can be constructed in numerous arrangements and levels of complexity. By allowing this type of arrangement, the architecture setup lends itself easily to expansion and modification of components. This also allows the system to be more robust to component failures [31].

SPA-U (USB-Based SPA) SPA-U is an interface standard that is based on the current USB (version 1.1) standard that supports 12 Mbps data transport. A benefit of this standard is that it is suitable for interfacing with most spacecraft devices [31].

SPA-S (Spacewire-Based SPA) "Spacewire is a European Space Agency (ESA) standard that supports high data rate transport (up to 625 Mbps has been demonstrated)and routable interconnect using a switched fabric concept [31]."

SPA-U Applique Sensor Interface Module (ASIM) The ASIM is a compact reference design of the standard that provides a bridge between a compliant implementation of the SPA-U or SPA-S standard and a user design. The ASIM contains automatic support for useful services including power management, synchronization and electronic datasheets [31].

Appendix B. MATLAB® Code Used to Support the Research

The MATLAB® code used to support the research is given below. The code is used to plot the interface data resulting from the Vector Modularity Measure analysis process. The tensor plot provides a visual graphic to study the sparsity of the five design structure matrices. The code given is for the two precision guided munitions (GBU-24 and GBU-31) and can be readily adapted for future applications.

B.1 GBU-24 Tensor Plot Code

Listing B.1: Code/gbu24tensor.m

```
clf
% Tensor Plot of 5 DSMs - Spatial, Informational, Material, Energy
%   and Informational
%   Uses plotcube function from the Mathworks website (www....
    mathworks.com)
%
%%%%%%%GBU-24 dsmtensor
%%%%%%%%%%%%Spatial .. Row zero blue
plotcube([1 1 1],[ 0 0 0],.9,[0 0 1]);
plotcube([1 1 1],[ 0 1 0],.9,[0 0 1]);
plotcube([1 1 1],[ 0 2 0],.9,[0 0 1]);
plotcube([1 1 1],[ 0 3 0],.9,[0 0 1]);
plotcube([1 1 1],[ 0 4 0],.9,[0 0 1]);
plotcube([1 1 1],[ 0 6 0],.9,[0 0 1]);
plotcube([1 1 1],[ 1 5 0],.9,[0 0 1]);
%plotcube([1 1 1],[ 2 0 0],.9,[0 0 1]);
plotcube([1 1 1],[ 2 4 0],.9,[0 0 1]);
plotcube([1 1 1],[ 3 0 0],.9,[0 0 1]);
plotcube([1 1 1],[ 3 1 0],.9,[0 0 1]);
plotcube([1 1 1],[ 3 2 0],.9,[0 0 1]);
plotcube([1 1 1],[ 4 0 0],.9,[0 0 1]);
```

```
plotcube([1 1 1],[ 4 1 0],.9,[0 0 1]);
plotcube([1 1 1],[ 5 0 0],.9,[0 0 1]);
plotcube([1 1 1],[ 5 1 0],.9,[0 0 1]);
%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%Structural .. Row one green
plotcube([1 1 1],[ 0 1 1],.9,[0 1 0]);
plotcube([1 1 1],[ 0 2 1],.9,[0 1 0]);
plotcube([1 1 1],[ 0 3 1],.9,[0 1 0]);
plotcube([1 1 1],[ 0 4 1],.9,[0 1 0]);
plotcube([1 1 1],[ 0 6 1],.9,[0 1 0]);
plotcube([1 1 1],[ 1 5 1],.9,[0 1 0]);
plotcube([1 1 1],[ 4 0 1],.9,[0 1 0]);
plotcube([1 1 1],[ 6 0 1],.9,[0 1 0]);
%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%Energy .. Row two red
plotcube([1 1 1],[ 0 2 2],.9,[1 0 0]);
plotcube([1 1 1],[ 0 3 2],.9,[1 0 0]);
plotcube([1 1 1],[ 0 6 2],.9,[1 0 0]);
plotcube([1 1 1],[ 1 5 2],.9,[1 0 0]);
plotcube([1 1 1],[ 4 0 2],.9,[1 0 0]);
plotcube([1 1 1],[ 5 1 2],.9,[1 0 0]);
%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%Material .. Row three yellow
plotcube([1 1 1],[ 0 3 3],.9,[1 1 0]);
plotcube([1 1 1],[ 0 4 3],.9,[1 1 0]);
plotcube([1 1 1],[ 0 6 3],.9,[1 1 0]);
plotcube([1 1 1],[ 1 5 3],.9,[1 1 0]);
plotcube([1 1 1],[ 4 0 3],.9,[1 1 0]);
%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%Information .. Row four
plotcube([1 1 1],[ 2 2 4],.9,[1  0 1]);
plotcube([1 1 1],[ 2 4 4],.9,[1  0 1]);
%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
%%%%%%%%%Row 0 of gray
plotcube([1 1 1],[ 0 7 0],.05,[0.5 0.5 0.5]);
plotcube([1 1 1],[1 6 0],.05,[0.5 0.5 0.5]);
plotcube([1 1 1],[2 5 0],.05,[0.5 0.5 0.5]);
plotcube([1 1 1],[ 3 4 0],.05,[0.5 0.5 0.5]);
plotcube([1 1 1],[4 3 0],.05,[0.5 0.5 0.5]);
plotcube([1 1 1],[5 2 0],.05,[0.5 0.5 0.5]);
plotcube([1 1 1],[ 6 1 0],.05,[0.5 0.5 0.5]);
plotcube([1 1 1],[7 0 0],.05,[0.5 0.5 0.5]);
%%%%%%%%
%%%%%%%%%Row 1 of gray
plotcube([1 1 1],[ 0 7 1],.05,[0.5 0.5 0.5]);
plotcube([1 1 1],[1 6 1],.05,[0.5 0.5 0.5]);
plotcube([1 1 1],[2 5 1],.05,[0.5 0.5 0.5]);
plotcube([1 1 1],[ 3 4 1],.05,[0.5 0.5 0.5]);
plotcube([1 1 1],[4 3 1],.05,[0.5 0.5 0.5]);
plotcube([1 1 1],[5 2 1],.05,[0.5 0.5 0.5]);
plotcube([1 1 1],[ 6 1 1],.05,[0.5 0.5 0.5]);
plotcube([1 1 1],[7 0 1],.05,[0.5 0.5 0.5]);
%%%%%%%%
%%%%%%%%%Row 2 of gray
plotcube([1 1 1],[ 0 7 2],.05,[0.5 0.5 0.5]);
plotcube([1 1 1],[1 6 2],.05,[0.5 0.5 0.5]);
plotcube([1 1 1],[2 5 2],.05,[0.5 0.5 0.5]);
plotcube([1 1 1],[ 3 4 2],.05,[0.5 0.5 0.5]);
plotcube([1 1 1],[4 3 2],.05,[0.5 0.5 0.5]);
plotcube([1 1 1],[5 2 2],.05,[0.5 0.5 0.5]);
plotcube([1 1 1],[ 6 1 2],.05,[0.5 0.5 0.5]);
plotcube([1 1 1],[7 0 2],.05,[0.5 0.5 0.5]);
%%%%%%%%
%%%%%%%%%Row 3 of gray
plotcube([1 1 1],[ 0 7 3],.05,[0.5 0.5 0.5]);
plotcube([1 1 1],[1 6 3],.05,[0.5 0.5 0.5]);
plotcube([1 1 1],[2 5 3],.05,[0.5 0.5 0.5]);
plotcube([1 1 1],[ 3 4 3],.05,[0.5 0.5 0.5]);
```

```
plotcube([1 1 1],[4 3 3],.05,[0.5 0.5 0.5]);
plotcube([1 1 1],[5 2 3],.05,[0.5 0.5 0.5]);
plotcube([1 1 1],[ 6 1 3],.05,[0.5 0.5 0.5]);
plotcube([1 1 1],[7 0 3],.05,[0.5 0.5 0.5]);
%%%%%%%%%
%%%%%%%%%Row 4 of gray
plotcube([1 1 1],[ 0 7 4],.05,[0.5 0.5 0.5]);
plotcube([1 1 1],[1 6 4],.05,[0.5 0.5 0.5]);
plotcube([1 1 1],[2 5 4],.05,[0.5 0.5 0.5]);
plotcube([1 1 1],[ 3 4 4],.05,[0.5 0.5 0.5]);
plotcube([1 1 1],[4 3 4],.05,[0.5 0.5 0.5]);
plotcube([1 1 1],[5 2 4],.05,[0.5 0.5 0.5]);
plotcube([1 1 1],[ 6 1 4],.05,[0.5 0.5 0.5]);
plotcube([1 1 1],[7 0 4],.05,[0.5 0.5 0.5]);
title('GBU-24 Interface Tensor Plot','FontSize',18)
xlabel('Provide','FontSize',14)
ylabel('Depend','FontSize',14)
zlabel('Interface Type','FontSize',14)
set(gca,'XLim',[0 8]);
set(gca,'XTick',0.5:7.5);
modules=['Mod 1';
    'Mod 2';
    'Mod 3';
    'Mod 4';
    'Mod 5';
    'Mod 6';
    'Mod 7';
    'Mod 8'];
set(gca,'XTickLabel',modules,'FontSize',9);
%
set(gca,'YLim',[0 8]);
set(gca,'YTick',0.5:7.5);
modulesrev=['Mod 8';
    'Mod 7';
    'Mod 6';
```

```
        'Mod 5';
        'Mod 4';
        'Mod 3';
        'Mod 2';
130     'Mod 1'];
    set(gca,'YTickLabel',modulesrev,'FontSize',9);
    %
    set(gca,'ZLim',[0 5]);
    set(gca,'ZTick',0.5:4.5);
135 %interfaces=['      Spatial';
    %    '   Structural';
    %    '       Energy';
    %    '     Material';
    %    'Informational'];
140 %set(gca,'ZTickLabel',interfaces,'FontSize',9);
    %NOTE labels have to have the same number of columns, i.e. same ...
       number of
    %items in the string e.g. mod 3, mod 4
    set(gca,'ZTickLabel',['      Spatial';'    Structure';'       ...
       Energy';
        '     Material';'Informational'],'FontSize',9);
145 %%%%%%%%
    % commented out for formateps purposes
    %print -depsc GBU24tensor
    %print -djpeg GBU24tensor
    print formateps
```

B.2 GBU-31 Tensor Plot Code

Listing B.2: Code/gbu31tensor.m

```
1 clf
  % Tensor Plot of 5 DSMs - Spatial, Informational, Material, Energy
  %   and Informational
  %   Uses plotcube function from the Mathworks website (www....
     mathworks.com)
5 %
```

```
%%%%%%%%%%GBU-31 tensor
%%%%%%%%%%Spatial .. Row zero blue
plotcube([1 1 1],[ 0 0 0],.9,[0 0 1]);
plotcube([1 1 1],[ 0 1 0],.9,[0 0 1]);
plotcube([1 1 1],[ 0 2 0],.9,[0 0 1]);
plotcube([1 1 1],[ 0 3 0],.9,[0 0 1]);
plotcube([1 1 1],[ 0 4 0],.9,[0 0 1]);
plotcube([1 1 1],[ 0 5 0],.9,[0 0 1]);
plotcube([1 1 1],[ 1 4 0],.9,[0 0 1]);
plotcube([1 1 1],[ 2 1 0],.9,[0 0 1]);
plotcube([1 1 1],[ 3 0 0],.9,[0 0 1]);
plotcube([1 1 1],[ 3 1 0],.9,[0 0 1]);
plotcube([1 1 1],[ 3 2 0],.9,[0 0 1]);
plotcube([1 1 1],[ 4 0 0],.9,[0 0 1]);
plotcube([1 1 1],[ 5 0 0],.9,[0 0 1]);
plotcube([1 1 1],[ 5 1 0],.9,[0 0 1]);
%%%%%%%%%%%%%%%%%%
%%%%%%%%%%Structural .. Row one green
plotcube([1 1 1],[ 0 1 1],.9,[0 1 0]);
plotcube([1 1 1],[ 0 2 1],.9,[0 1 0]);
plotcube([1 1 1],[ 0 3 1],.9,[0 1 0]);
plotcube([1 1 1],[ 0 4 1],.9,[0 1 0]);
plotcube([1 1 1],[ 0 5 1],.9,[0 1 0]);
plotcube([1 1 1],[ 1 4 1],.9,[0 1 0]);
plotcube([1 1 1],[ 4 0 1],.9,[0 1 0]);
plotcube([1 1 1],[ 6 0 1],.9,[0 1 0]);
%%%%%%%%%%%%%%%%%%
%%%%%%%%%%Energy .. Row two red
plotcube([1 1 1],[ 0 2 2],.9,[1 0 0]);
plotcube([1 1 1],[ 0 3 2],.9,[1 0 0]);
plotcube([1 1 1],[ 0 4 2],.9,[1 0 0]);
plotcube([1 1 1],[ 0 5 2],.9,[1 0 0]);
plotcube([1 1 1],[ 1 4 2],.9,[1 0 0]);
plotcube([1 1 1],[ 2 1 2],.9,[1 0 0]);
plotcube([1 1 1],[ 4 0 2],.9,[1 0 0]);
```

```
plotcube([1 1 1],[ 5 1 2],.9,[1 0 0]);
%%%%%%%%%%%%%%%%%%
%%%%%%%%%%Material .. Row three yellow
plotcube([1 1 1],[ 0 3 3],.9,[1 1 0]);
plotcube([1 1 1],[ 0 4 3],.9,[1 1 0]);
plotcube([1 1 1],[ 0 5 3],.9,[1 1 0]);
plotcube([1 1 1],[ 4 0 3],.9,[1 1 0]);
%%%%%%%%%%%%%%%%%%
%%%%%%%%%%Information .. Row four
%plotcube([1 1 1],[ 1 0 4],.9,[1  0 1]);
plotcube([1 1 1],[ 2 3 4],.9,[1  0 1]);
plotcube([1 1 1],[ 2 5 4],.9,[1  0 1]);
%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%Row 0 of gray
plotcube([1 1 1],[ 0 7 0],.05,[0.5 0.5 0.5]);
plotcube([1 1 1],[1 6 0],.05,[0.5 0.5 0.5]);
plotcube([1 1 1],[2 5 0],.05,[0.5 0.5 0.5]);
plotcube([1 1 1],[ 3 4 0],.05,[0.5 0.5 0.5]);
plotcube([1 1 1],[4 3 0],.05,[0.5 0.5 0.5]);
plotcube([1 1 1],[5 2 0],.05,[0.5 0.5 0.5]);
plotcube([1 1 1],[ 6 1 0],.05,[0.5 0.5 0.5]);
plotcube([1 1 1],[7 0 0],.05,[0.5 0.5 0.5]);
%%%%%%%%
%%%%%%%%%Row 1 of gray
plotcube([1 1 1],[ 0 7 1],.05,[0.5 0.5 0.5]);
plotcube([1 1 1],[1 6 1],.05,[0.5 0.5 0.5]);
plotcube([1 1 1],[2 5 1],.05,[0.5 0.5 0.5]);
plotcube([1 1 1],[ 3 4 1],.05,[0.5 0.5 0.5]);
plotcube([1 1 1],[4 3 1],.05,[0.5 0.5 0.5]);
plotcube([1 1 1],[5 2 1],.05,[0.5 0.5 0.5]);
plotcube([1 1 1],[ 6 1 1],.05,[0.5 0.5 0.5]);
plotcube([1 1 1],[7 0 1],.05,[0.5 0.5 0.5]);
%%%%%%%%
```

```
%%%%%%%%%%Row 2 of gray
plotcube([1 1 1],[ 0 7 2],.05,[0.5 0.5 0.5]);
plotcube([1 1 1],[1 6 2],.05,[0.5 0.5 0.5]);
plotcube([1 1 1],[2 5 2],.05,[0.5 0.5 0.5]);
plotcube([1 1 1],[ 3 4 2],.05,[0.5 0.5 0.5]);
plotcube([1 1 1],[4 3 2],.05,[0.5 0.5 0.5]);
plotcube([1 1 1],[5 2 2],.05,[0.5 0.5 0.5]);
plotcube([1 1 1],[ 6 1 2],.05,[0.5 0.5 0.5]);
plotcube([1 1 1],[7 0 2],.05,[0.5 0.5 0.5]);
%%%%%%%%%
%%%%%%%%%%Row 3 of gray
plotcube([1 1 1],[ 0 7 3],.05,[0.5 0.5 0.5]);
plotcube([1 1 1],[1 6 3],.05,[0.5 0.5 0.5]);
plotcube([1 1 1],[2 5 3],.05,[0.5 0.5 0.5]);
plotcube([1 1 1],[ 3 4 3],.05,[0.5 0.5 0.5]);
plotcube([1 1 1],[4 3 3],.05,[0.5 0.5 0.5]);
plotcube([1 1 1],[5 2 3],.05,[0.5 0.5 0.5]);
plotcube([1 1 1],[ 6 1 3],.05,[0.5 0.5 0.5]);
plotcube([1 1 1],[7 0 3],.05,[0.5 0.5 0.5]);
%%%%%%%%%
%%%%%%%%%%Row 4 of gray
plotcube([1 1 1],[ 0 7 4],.05,[0.5 0.5 0.5]);
plotcube([1 1 1],[1 6 4],.05,[0.5 0.5 0.5]);
plotcube([1 1 1],[2 5 4],.05,[0.5 0.5 0.5]);
plotcube([1 1 1],[ 3 4 4],.05,[0.5 0.5 0.5]);
plotcube([1 1 1],[4 3 4],.05,[0.5 0.5 0.5]);
plotcube([1 1 1],[5 2 4],.05,[0.5 0.5 0.5]);
plotcube([1 1 1],[ 6 1 4],.05,[0.5 0.5 0.5]);
plotcube([1 1 1],[7 0 4],.05,[0.5 0.5 0.5]);
title('GBU-31 Interface Tensor Plot','FontSize',18)
xlabel('Provide','FontSize',14)
ylabel('Depend','FontSize',14)
zlabel('Interface Type','FontSize',14)
set(gca,'XLim',[0 8]);
set(gca,'XTick',0.5:7.5);
```

```
modules=['Mod 1';
    'Mod 2';
    'Mod 3';
    'Mod 4';
    'Mod 5';
    'Mod 6';
    'Mod 7';
    'Mod 8'];
set(gca,'XTickLabel',modules,'FontSize',9);
%
set(gca,'YLim',[0 8]);
set(gca,'YTick',0.5:7.5);
modulesrev=['Mod 8';
    'Mod 7';
    'Mod 6';
    'Mod 5';
    'Mod 4';
    'Mod 3';
    'Mod 2';
    'Mod 1'];
set(gca,'YTickLabel',modulesrev,'FontSize',9);
%
set(gca,'ZLim',[0 5]);
set(gca,'ZTick',0.5:4.5);
%interfaces=['      Spatial';
%    '   Structural';
%    '       Energy';
%    '     Material';
%    'Informational'];
%set(gca,'ZTickLabel',interfaces,'FontSize',9);
%NOTE labels have to have the same number of columns, i.e. same ...
   number of
%items in the string e.g. mod 3, mod 4
set(gca,'ZTickLabel',['      Spatial';'    Structure';'       ...
   Energy';
```

```
    '     Material';'Informational'],'FontSize',9);
%%%%%%%%%
print -depsc GBU31tensor
print -djpeg GBU31tensor
```

B.3 PnPSat Tensor Plot Code

Listing B.3: Code/pnpsattensor.m

```
clf
%   ****UPDATED 14 Jan 2010
% Tensor Plot of 5 DSMs - Spatial, Informational, Material, Energy
%   and Informational
%   Uses plotcube function from the Mathworks website (www....
    mathworks.com)
%
% [ 0 0 0 0 0 0 0 0
%   1 0 0 0 0 0 0 0
%   0 1 0 0 0 0 0 0
%   1 0 1 0 0 0 0 0
%   1 0 0 0 0 0 0 0
%   1 0 0 1 0 0 0 0
%   1 0 0 1 1 1 0 0
%   1 0 0 1 1 1 0 0
%%%%%%%PnPSat dsmtensor
%%%%%%%%%%%%Spatial .. Row zero blue
%[ A B C]
% C = Interface type, here 0 = spatial
% A = Col 0, 1, 2 etc
% B = Row 7 = 0, 6 = 1, 5 = 4 etc
plotcube([1 1 1],[ 0 0 0],.9,[0 0 1]);
plotcube([1 1 1],[ 0 1 0],.9,[0 0 1]);
plotcube([1 1 1],[ 0 2 0],.9,[0 0 1]);
plotcube([1 1 1],[ 0 3 0],.9,[0 0 1]);
plotcube([1 1 1],[ 0 4 0],.9,[0 0 1]);
plotcube([1 1 1],[ 0 5 0],.9,[0 0 1]);
plotcube([1 1 1],[ 0 6 0],.9,[0 0 1]);
```

```
plotcube([1 1 1],[ 1 0 0],.9,[0 0 1]);
plotcube([1 1 1],[ 1 1 0],.9,[0 0 1]);
plotcube([1 1 1],[ 1 2 0],.9,[0 0 1]);
plotcube([1 1 1],[ 1 3 0],.9,[0 0 1]);
plotcube([1 1 1],[ 1 4 0],.9,[0 0 1]);
plotcube([1 1 1],[ 1 5 0],.9,[0 0 1]);
plotcube([1 1 1],[ 2 0 0],.9,[0 0 1]);
plotcube([1 1 1],[ 3 0 0],.9,[0 0 1]);
plotcube([1 1 1],[ 4 0 0],.9,[0 0 1]);
plotcube([1 1 1],[ 5 0 0],.9,[0 0 1]);
plotcube([1 1 1],[ 6 0 0],.9,[0 0 1]);

%%%%%%%%%%%%%%%%%%
%%%%%%%%%%Structural .. Row one green
% [ 0 0 0 0 0 0 0 0
%   1 0 0 0 0 0 0 0
%   1 0 0 0 0 0 0 0
%   1 0 0 0 0 0 0 0
%   1 0 0 0 0 0 0 0
%   1 0 0 0 0 0 0 0
%   1 0 0 0 0 0 0 0
%   1 0 0 0 0 0 0 0

plotcube([1 1 1],[ 0 0 1],.9,[0 1 0]);
plotcube([1 1 1],[ 0 1 1],.9,[0 1 0]);
plotcube([1 1 1],[ 0 2 1],.9,[0 1 0]);
plotcube([1 1 1],[ 0 3 1],.9,[0 1 0]);
plotcube([1 1 1],[ 0 4 1],.9,[0 1 0]);
plotcube([1 1 1],[ 0 5 1],.9,[0 1 0]);
plotcube([1 1 1],[ 0 6 1],.9,[0 1 0]);
%%%%%%%%%%%%%%%%%%
%%%%%%%%%%Energy .. Row two red

% [ 0 0 0 0 0 0 0 0
%   0 0 0 0 0 0 0 0
```

```
%    0 0 0 0 0 0 0 0
%    0 1 0 0 0 0 0 0
%    0 0 1 0 0 0 0 0
%    0 0 1 0 0 0 0 0
%    0 0 1 0 0 0 0 0
%    0 0 0 0 0 0 0 0
plotcube([1 1 1],[ 1 4 2],.9,[1 0 0]);
plotcube([1 1 1],[ 1 5 2],.9,[1 0 0]);
plotcube([1 1 1],[ 2 1 2],.9,[1 0 0]);
plotcube([1 1 1],[ 2 2 2],.9,[1 0 0]);
plotcube([1 1 1],[ 2 3 2],.9,[1 0 0]);
%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%Material .. Row three yellow
%
%%% PnPSat ... no Material I/Fs
%
% plotcube([1 1 1],[ 0 3 3],.9,[1 1 0]);
%
%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%Information .. Row four
% [ 0 0 0 0 0 0 0 0
%    0 0 0 0 0 0 0 0
%    0 0 0 0 0 0 0 0
%    0 0 0 0 0 0 0 0
%    0 0 0 0 0 0 0 0
%    1 1 1 0 1 0 0 0
%    0 0 0 0 0 1 0 0
%    0 1 0 0 1 0 0 0
%
plotcube([1 1 1],[ 0 2 4],.9,[1  0 1]);
plotcube([1 1 1],[ 1 0 4],.9,[1  0 1]);
plotcube([1 1 1],[ 1 2 4],.9,[1  0 1]);
plotcube([1 1 1],[ 1 3 4],.9,[1  0 1]);
plotcube([1 1 1],[ 2 2 4],.9,[1  0 1]);
plotcube([1 1 1],[ 4 0 4],.9,[1  0 1]);
```

```
    plotcube([1 1 1],[ 4 2 4],.9,[1  0 1]);
    plotcube([1 1 1],[ 5 1 4],.9,[1  0 1]);
100 %%%%%%%%%%%%%%%%%%
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    %%%%%%%%%Row 0 of gray
    plotcube([1 1 1],[ 0 7 0],.05,[0.5 0.5 0.5]);
105 plotcube([1 1 1],[1 6 0],.05,[0.5 0.5 0.5]);
    plotcube([1 1 1],[2 5 0],.05,[0.5 0.5 0.5]);
    plotcube([1 1 1],[ 3 4 0],.05,[0.5 0.5 0.5]);
    plotcube([1 1 1],[4 3 0],.05,[0.5 0.5 0.5]);
    plotcube([1 1 1],[5 2 0],.05,[0.5 0.5 0.5]);
110 plotcube([1 1 1],[ 6 1 0],.05,[0.5 0.5 0.5]);
    plotcube([1 1 1],[7 0 0],.05,[0.5 0.5 0.5]);
    %%%%%%%%
    %%%%%%%%%Row 1 of gray
    plotcube([1 1 1],[ 0 7 1],.05,[0.5 0.5 0.5]);
115 plotcube([1 1 1],[1 6 1],.05,[0.5 0.5 0.5]);
    plotcube([1 1 1],[2 5 1],.05,[0.5 0.5 0.5]);
    plotcube([1 1 1],[ 3 4 1],.05,[0.5 0.5 0.5]);
    plotcube([1 1 1],[4 3 1],.05,[0.5 0.5 0.5]);
    plotcube([1 1 1],[5 2 1],.05,[0.5 0.5 0.5]);
120 plotcube([1 1 1],[ 6 1 1],.05,[0.5 0.5 0.5]);
    plotcube([1 1 1],[7 0 1],.05,[0.5 0.5 0.5]);
    %%%%%%%%
    %%%%%%%%%Row 2 of gray
    plotcube([1 1 1],[ 0 7 2],.05,[0.5 0.5 0.5]);
125 plotcube([1 1 1],[1 6 2],.05,[0.5 0.5 0.5]);
    plotcube([1 1 1],[2 5 2],.05,[0.5 0.5 0.5]);
    plotcube([1 1 1],[ 3 4 2],.05,[0.5 0.5 0.5]);
    plotcube([1 1 1],[4 3 2],.05,[0.5 0.5 0.5]);
    plotcube([1 1 1],[5 2 2],.05,[0.5 0.5 0.5]);
130 plotcube([1 1 1],[ 6 1 2],.05,[0.5 0.5 0.5]);
    plotcube([1 1 1],[7 0 2],.05,[0.5 0.5 0.5]);
    %%%%%%%%
```

```
    %%%%%%%%%%Row 3 of gray
    plotcube([1 1 1],[ 0 7 3],.05,[0.5 0.5 0.5]);
135 plotcube([1 1 1],[1 6 3],.05,[0.5 0.5 0.5]);
    plotcube([1 1 1],[2 5 3],.05,[0.5 0.5 0.5]);
    plotcube([1 1 1],[ 3 4 3],.05,[0.5 0.5 0.5]);
    plotcube([1 1 1],[4 3 3],.05,[0.5 0.5 0.5]);
    plotcube([1 1 1],[5 2 3],.05,[0.5 0.5 0.5]);
140 plotcube([1 1 1],[ 6 1 3],.05,[0.5 0.5 0.5]);
    plotcube([1 1 1],[7 0 3],.05,[0.5 0.5 0.5]);
    %%%%%%%%%
    %%%%%%%%%%Row 4 of gray
    plotcube([1 1 1],[ 0 7 4],.05,[0.5 0.5 0.5]);
145 plotcube([1 1 1],[1 6 4],.05,[0.5 0.5 0.5]);
    plotcube([1 1 1],[2 5 4],.05,[0.5 0.5 0.5]);
    plotcube([1 1 1],[ 3 4 4],.05,[0.5 0.5 0.5]);
    plotcube([1 1 1],[4 3 4],.05,[0.5 0.5 0.5]);
    plotcube([1 1 1],[5 2 4],.05,[0.5 0.5 0.5]);
150 plotcube([1 1 1],[ 6 1 4],.05,[0.5 0.5 0.5]);
    plotcube([1 1 1],[7 0 4],.05,[0.5 0.5 0.5]);
    %title('PnPSat Interface Tensor Plot','FontSize',18)
    xlabel('Provide','FontSize',14)
    ylabel('Depend','FontSize',14)
155 zlabel('Interface Type','FontSize',14)
    set(gca,'XLim',[0 8]);
    set(gca,'XTick',0.5:7.5);
    modules=['Mod 1';
        'Mod 2';
160     'Mod 3';
        'Mod 4';
        'Mod 5';
        'Mod 6';
        'Mod 7';
165     'Mod 8'];
    set(gca,'XTickLabel',modules,'FontSize',9);
    %
```

```
    set(gca,'YLim',[0 8]);
    set(gca,'YTick',0.5:7.5);
170 modulesrev=['Mod 8';
        'Mod 7';
        'Mod 6';
        'Mod 5';
        'Mod 4';
175     'Mod 3';
        'Mod 2';
        'Mod 1'];
    set(gca,'YTickLabel',modulesrev,'FontSize',9);
    %
180 set(gca,'ZLim',[0 5]);
    set(gca,'ZTick',0.5:4.5);
    %interfaces=['      Spatial';
    %    '   Structural';
    %    '       Energy';
185 %    '     Material';
    %    'Informational'];
    %set(gca,'ZTickLabel',interfaces,'FontSize',9);
    %NOTE labels have to have the same number of columns, i.e. same ...
       number of
    %items in the string e.g. mod 3, mod 4
190 set(gca,'ZTickLabel',['      Spatial';'   Structural';'       ...
       Energy';
        '     Material';'Informational'],'FontSize',9);
    %%%%%%%%%
    print -depsc PnPSattensor
    print -djpeg PnPSattensor
```

Bibliography

1. Abdelsalam, Hisham M. E. and Han P. Bao. "Re-sequencing of Design Processes With Activity Stochastic Time and Cost: An Optimization-Simulation Approach". *Journal of Mechanical Design*, 129(2):150–157, February 2007.

2. Adang, Thomas and James Gee. "Creating an agile, all-space architecture". *Crosslink - The aerospace corporation magazine of advances in aerospace technology*, 10(1):6–11, Summer 2009.

3. Allen, Dr. Richard E. Van. "The future of responsive space: a summary of the results and papers from the 5th responsive space conference". *5th Responsive Space Conference*. AIAA, May 2007.

4. Arora, Jasbir S. *Introduction to optimum design*. Elsevier Academic Press, San Diego, CA, 2nd edition, 2004.

5. Baghal, Lisa A. "Trial Overview", 2009.

6. Browning, T. R. "Applying the design structure matrix to system decomposition and integration problems: a review and new directions", 2001.

7. Buede, Dennis M. *The engineering design of systems: models and methods*. John Wiley & Sons, Inc., New York, NY, 2000.

8. Carruthers, Peter. "The roots of scientific reasoning: infancy, modularity, and the art of tracking", 2002.

9. DiPalma, John. *PNP-4025 Rapid AI&T procedures*. SpaceWorks, Inc.

10. DiPalma, John. "Test flow time", December 2009.

11. Fixson, Sebastian K. "Product architecture assessment: a tool to link product, process, and supply chain design decisions". *Journal of Operations Management*, 23(3-4):345–369, 4 2005.

12. Friedman, George J. *Constraint theory: multidimensional mathematical model management*. Springer, New York, 2005.

13. Fronterhouse, Don and Jim Lyke. "Plug-and-Play Satellite (PnPSat)". *Proceedings of the AIAA Infotech/Aerospace 2007 Conference and Exhibit*, volume 2007. AIAA, May 2007.

14. Fronterhouse, Don and Jeff Prebble. "Building SPA PnP Satellites". *7th Responsive Space Conference*, volume RS7-2009. AIAA, April 2009.

15. Gershenson, J. K., G. J. Prasad, and S. Allamneni. "Modular Product Design: a Life-Cycle View". *Journal of Integrated Design & Process Science*, 3(4):13, December 1999.

16. Gershenson, J. K., G. J. Prasad, and Y. Zhang. "Product modularity: definitions and benefits". *Journal of Engineering Design*, 14(3):295, 2003.

17. Gershenson, J. K., G. J. Prasad, and Y. Zhang. "Product modularity: measures and design methods". *Journal of Engineering Design*, 15(1):33–51, February 2004.

18. Gerwin, Donald. "An Agenda for Research on the Flexibility of Manufacturing Processes". *International journal of operations & production management*, 7(1):38, January 1987.

19. Guo, Fang and J. K. Gershenson. "Comparison of modular measurement methods based on consistency analysis and sensitivity analysis". *ASME 2003 Design Engineering Technical Conferences and Computers and Information in Engineering Conference*. DETC2003, September 2003.

20. Guo, Fang and John K. Gershenson. "A Comparison of Modular Product Design Methods Based on Improvement and Iteration". *ASME Conference Proceedings*, 2004(46962a):261–269, January 2004.

21. Hirtz, Julie, Robert B. Stone, Daniel A. McAdams, Simon Szykman, and Kristin L. Wood. *A functional basis for engineering design: reconciling and evolving previous efforts*. NIST Technical Note 1447, U.S. Government Printing Office, 2002.

22. Hölttä, Katja, Nam P. Suh, and Oliver Olivier de Weck. "Trade-off between modularity and performance for engineered systems and products". *In Proc of International Conference on Engineering Design*. August 2005.

23. Hölttä-Otto, Katja and Oliver Olivier de Weck. "Degree of Modularity in Engineering Systems and Products with Technical and Business Constraints". *Concurrent Engineering*, 15(2):113–125, 2007.

24. House Committee on Armed Services. *NDAA FY2008: Report of the committee on armed services, House of Representatives on H.R. 1585*. Technical report, House of Representatives, May 2007.

25. Kahraman, Mesut Ö. *A constraint based approach for building operationally responsive satellites*. Master's thesis, Air Force Institute of Technology, 2008.

26. Kratochvíl, Milan and Charles Carson. *Growing Modular: Mass Customization of Complex Products, Services and Software*. Springer, Berlin, Germany, 2005.

27. Krause, E.F. "Maximizing the Product of Summands; Minimizing the Sum of Factors". *Mathematics Magazine*, 69(4):270–278, 1996.

28. Lee, Douglas E. "Space reform". *Air & Space Power Journal*, (Summer 2004), 2004.

29. Lindholm, D., D. Tate, and V. Harutunian. "Consequences of Design Decisions in Axiomatic Design". *Journal of Integrated Design & Process Science*, 3(4):1, December 1999.

30. Lyke, Jim. "Briefing: The six-day spacecraft: creating a plug-and-play approach for aerospace systems", September 2008.

31. Lyke, Jim, Don Fronterhouse, Scott Cannon, Denise Lanza, and Wheaton Byers. "Space Plug-and-Play Avionics". *3rd Responsive Space Conference*. April 2005.

32. Maier, Mark W. and Eberhardt Rechtin. *The art of systems architecting*. 2nd edition, 2000.

33. Martin, Maurice. "Space plug-and-play avionics (SPA) standards". *22nd Annual AIAA/USU Small Satellite Conference ORS Session*. August 2008.

34. Mikkola, J. H. "Management of Product Architecture Modularity for Mass Customization: Modeling and Theoretical Considerations". *Engineering Management, IEEE Transactions on*, 54(1):57–69, 2007.

35. Muhammad, Rashid B. "Graph Theory, Definitions and Examples". URL http://www.personal.kent.edu / ∼rmuhamma / Graph Theory / MyGraph Theory / defEx.htm, Last visited 4 March 2010.

36. Novak, Sharon and Steven D. Eppinger. "Sourcing By Design: Product Complexity and the Supply Chain". *Management Science*, 47(1):189, 01 2001.

37. Oyama, Kyle F., Amie C. Stryker, David R. Jacques, and David S. Long. "Linking modularity to system assembly – initial findings and implications". *8th Conference on Systems Engineering Research (*Accepted*)*. March 2010.

38. Pahl, G. and W. Beitz. *Engineering Design - A Systematic Approach*. Springer-Verlag, London, 1996.

39. Russel, Andrew. "Seminar: Modularity: from modern architecture to postmodern technologies". Seminar, online, March 2008. URL `http://fhi.duke.edu/`.

40. SECAF, Office of Public Affairs. "Press release: ORS Office stands up at Kirtland AFB", May 2007. For more information about the standup of the ORS Office, contact 377th ABW Public Affairs at 505-246-6068.

41. Sega, Ronald M. and James E. Cartwright. *Plan for operationally responsive space: a report to congressional defense committees*. Technical report, Department of Defense, April 2007.

42. Sosa, Manuel E., Steven D. Eppinger, and Craig M. Rowles. "A Network Approach to Define Modularity of Components in Complex Products". *Journal of Mechanical Design*, 129(11):1118–1129, 2007.

43. Stone, Robert B. and Kristin L. Wood. "Development of a Functional Basis for Design". *Journal of Mechanical Design*, 122(4):359–370, December 2000.

44. Stone, Robert B., Kristin L. Wood, and Richard H. Crawford. "A heuristic method to identify modules from a functional description of a product". *Proceedings of the ASME Design Engineering Technical Conference*. DETC, 1998.

45. Stone, Robert B., Kristin L. Wood, and Richard H. Crawford. "A heuristic method for identifying modules for product architectures". *Design Studies*, 21(1):5–31, January 2000.

46. Stone, Robert B., Kristin L. Wood, and Richard H. Crawford. "Using quantitative functional models to develop product architectures". *Design Studies*, 21(3):239–260, May 2000.

47. Stryker, Amie C. and David R. Jacques. "Modularity versus functionality – a survey and application". *7th Conference on Systems Engineering Research*. April 2009.

48. Stryker, Amie C. and David R. Jacques. "Development of a measure to assess reconfigurability of modular products". *8th Conference on Systems Engineering Research (*Accepted*)*. March 2010.

49. Stryker, Amie C. and David R. Jacques. "Development of a Measure to Assess Reconfigurability of Modular Products". Submitted to *Journal of Mechanical Design*, 2010.

50. Stryker, Amie C. and David R. Jacques. "PnPSat – A Modularity Assessment". Submitted to *AIAA Journal of Spacecraft & Rockets*, 2010.

51. Stryker, Amie C., David R. Jacques, and David M. Long. "A vector approach to assessing modularity". *20th INCOSE International Symposium (*Accepted*)*. July 2010.

52. Stryker, Amie C., David R. Jacques, and David S. Long. "Assessing Modularity – a Vector Approach". Submitted to *Journal of Engineering Design*, 2010.

53. Suh, Nam P. *The principles of design*. Oxford University Press, New York, 1990.

54. Suh, Nam P. *Axiomatic design: advances and applications*. Oxford University Press, New York, 2001.

55. Suh, Nam P. *Complexity: theory and applications*. Oxford University Press, New York, 2005.

56. United States Air Force. *T.O. 11A-1-63 Munitions assembly procedures*.

57. University, Defense Acquisition. "DoD Space Systems Acquisition". website, Mar 2009.

58. Wall, Michael E., Andreas Rechtsteiner, and Luis M. Rocha. *Singular value decomposition and principal component analysis*, 91–109. A practical approach to microarray data analysis. Kluwer, Norwell, MA, 2003.

59. Wegner, Peter. "Operationally responsive Space - Briefing to Commercial Space Transportation Advisory Committee", 2009.

60. Wertz, James R. and Wiley J. Larson. *Space Mission Analysis Design*. Microcosm Press and Kluwer Academic Publishers, El Segundo, CA, 3rd edition, 1999.

Vita

Lieutenant Colonel Amie C. Stryker graduated from Mandeville High School in Mandeville, Louisiana. She entered undergraduate studies at Embry-Riddle Aeronautical University in Daytona Beach, Florida where she graduated with a Bachelor of Science degree in Aerospace Engineering in April 1993. She was commissioned through the AFROTC Detachment 157 at Embry-Riddle Aeronautical University.

Her first assignment was at Vandenberg AFB as a student in Undergraduate Missile Training in November 1993. In September 1994, she was assigned to the 448th Missile Squadron, Grand Forks AFB, North Dakota where she served as a missile combat crew deputy commander, instructor, and missile combat crew commander. In September 1997, she was assigned to the DoD Space Test Program, operating location Houston, Texas where she served as a branch chief for cargo payloads and space station payloads destined for the Mir and international space stations. In September 2000, she was re-assigned to the 31st Test and Evaluation Squadron, Edwards AFB, California where she served as the F-22 avionics lead engineer, operational test. While stationed at Edwards, she was selected for a special duty assignment to Germany at the German Aerospace Center, Oberpfaffenhofen as part of the Engineer and Scientist Exchange Program after attending the German language course at the Defense Language Institute, Monterey, California. In July 2004, she successfully completed the language course and was awarded an Associate Arts degree in German and subsequently completed a two-year tour overseas. In August 2006, she entered the Graduate School of Engineering and Management, Air Force Institute of Technology. Upon graduation, she will be assigned to the Pentagon.

Permanent address: 2950 Hobson Way
Air Force Institute of Technology
Wright-Patterson AFB, OH 45433

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704–0188

1. REPORT DATE *(DD–MM–YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From — To)*
March 2010	Doctoral Dissertation	September 2006 – March 2010

4. TITLE AND SUBTITLE

Development of Measures to Assess Product Modularity and Reconfigurability

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)

Stryker, Amie C., Lieutenant Colonel, USAF

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Air Force Institute of Technology
Graduate School of Engineering and Management (AFIT/EN)
2950 Hobson Way
WPAFB OH 45433-7765

8. PERFORMING ORGANIZATION REPORT NUMBER

AFIT/DS/ENV/10-M01

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

Operational Responsive Space Office
Attn: Director, ORSO (Dr. Peter Wegner)
3550 Aberdeen Ave SE, Bldg 404, Rm 2
Kirtland AFB NM 87117-5776
(505)846-4282
peter.wegner@kirtkand.af.mil

10. SPONSOR/MONITOR'S ACRONYM(S)

ORSO

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION / AVAILABILITY STATEMENT

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED

13. SUPPLEMENTARY NOTES

14. ABSTRACT

This dissertation outlines a method and measures for assessing product modularity in terms of coupling, reusability, and flexibility. A five-step analysis process is developed and used to guide the modularity assessment. Defining and decomposing products are performed first. Using the resultant functional model from the first step, the identified functions are mapped to modules in a product in the second step. In the third and fourth steps, module-to-module interfaces are identified and captured in design structure matrices or a tensor plot. Finally, using results from steps 1–4, the Vector Modularity Measure that includes a reconfigurability measure can be calculated. The measures and analysis process are demonstrated using two precision guided munitions in the United States Air Force inventory. After this demonstration, the research focuses on extending the approach to a modular satellite design problem, namely AFRL's Plug-and-Play Satellite (PnPSat) concept for Operationally Responsive Space. Using the resulting analysis, recommendations to the existing PnPSat design to further increase modularity and its derived benefits are given. Lastly, the modularity analysis process and applications are used to draw conclusions and make recommendations for future research to include identifying factors that influence both modularity and the timeline to perform product assembly and check-out.

15. SUBJECT TERMS

Modularity, flexibility, reconfigurability

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			Dr. David R. Jacques (ENV)
U	U	U	UU	190	19b. TELEPHONE NUMBER *(include area code)* (937)255-3355x3329; email:david.jacques@afit.edu