Paper #181

Screening for Real Options "In" an Engineering System: A Step Towards Flexible Weapon System Development

PART I: The Use of Design Matrices to Create an End-to-End Representation of a Complex Socio-Technical System

Jason E. Bartolomei
Massachusetts Institute of Technology
77 Massachusetts Ave., 41-205
Cambridge, MA 02139
jason.bartolomei@mit.edu

Daniel E. Hastings
Massachusetts Institute of Technology
77 Massachusetts Ave., 4-110
Cambridge, MA 02139
hastings@mit.edu

Richard de Neufville
Massachusetts Institute of Technology
77 Massachusetts Ave., E40-245
Cambridge, MA 02139
ardent@mit.edu

Donna H. Rhodes
Massachusetts Institute of Technology
77 Massachusetts Ave., E40-215A
Cambridge, MA 02139
rhodes@mit.edu

Abstract

The goal of this research is to develop an analytical framework for screening for real options "in" an engineering system. Real options is defined in the finance literature as the right, but not the obligation, to take an action (e.g. deferring, expanding, contracting, or abandoning) at a predetermined cost and for a predetermined time. These are called "real options" because they pertain to physical or tangible assets, such as equipment, rather than financial instruments. Real options improve a system's capability of undergoing classes of changes with relative ease. This property is often called "flexibility." Recently, the DoD has emphasized the need to develop flexible system in order to improve operational, technical, and programmatic effectiveness. The aim of this research is to apply real options thinking to weapon acquisitions in order to develop weapons system programs able to deftly avoid downside consequences or exploit upside opportunities.

The practice of real options in systems engineering is a nascent field of inquiry. One of the most significant challenges in applying real options to engineering systems is problem of identifying the most efficacious points within the system to create options. In order to identify the points of interest, systems engineers require knowledge about the physical and non physical aspects of the system, insight into sources of change, and the ability to examine the dynamic behavior of the system. We propose a two-phase process to perform this analysis. The first phase is a system

representation phase that seeks to create an end-to-end representation of engineering system that includes endogenous interactions across system views and interactions with a systems environment. The next phase is an analysis phase that models the evolution of the engineering system in order to identify the real options in the system. This paper presents the system representation phase and proposes a methodology for creating an end-to-end representation of an engineering system.

The methodology for representing an engineering system extends existing systems engineering and architecting methods in two dimensions. First, the framework couples traditional architecting views to represent traceability and endogenous interactions within an engineering system. Second, the framework includes views of the system not represented in traditional engineering frameworks that includes social networks and environmental interactions. The framework uses coupled Design Structure Matrices (DSM) to represent the traditional and new architecting views. The coupled DSMs are organized into an Engineering System Matrix (ESM), which is a holistic representation of an engineering system that captures all of the critical variables and causal interactions across architectural elements. The result is an analytic framework that captures the qualitative understanding of the system into a single view that is conducive for deep quantitative inquiry.

This paper presents a discussion of pertinent literature, an overview of the ESM framework and underlying theory. In addition, this paper previews ongoing research using the ESM to identify options for a mini-air vehicle (MAV) weapon development system.

Defining an Engineering Systems

MIT has recently embarked on the creation of a new academic discipline called *Engineering Systems* that is devoted to the study of complex socio-technical systems. There is a debate within the community on what the term "Engineering Systems" means. Some interpret the phrase as a verb-noun—"A new discipline for the better engineering of systems." Others interpret the phrase as an adjective-noun—"The Big-Dig is one of many engineering systems." One definition for an engineering system comes from an MIT White Paper where MIT Professor Joseph Sussman proposes a definition that is congruent with the latter interpretation. Sussman suggests an "Engineering System" is a type of a Complex, Large-Scale, Integrated, Open System (CLIOS). (Sussman 2000) He defines "complex" as having a large number of interacting parts, which exhibit non-trivial behavior. "Large-Scale" refers to not only physical size, but also impact. Sussman lists a planet (for its size) and the automobile (for its impact on society) as examples of "Large-Scale" systems. "Integrated" refers to the tight coupling of subsystems through feedback loops. Finally, "open" refers to social, political, and economic aspects affecting a system. According to Sussman's definition, what differentiates an engineering system from other CLIOS, is a significant technology component.

In 2004, Joel Moses offers a simplified definition for an engineering system as follows: "Engineering systems are systems designed by humans having some purpose and are composed of interacting parts." (Moses 2004) Moses' definition does not limit engineering systems to large-scale systems (though he does later say that large-scale systems are of particular interest to the field). In addition, the inclusion of human design and system purpose are important in that they distinguish an engineering system from other systems, such as ecological or biological

systems. Although Moses' speaks to the lifecycle issues of an engineering system throughout his discussion, these ideas seem discounted in his definition. Moses' use of the word "design" seems too limiting in that readers might infer a front-end activity rather than the human/system lifecycle interactions emphasized in the monograph. Lastly, Moses' seems to draw the system boundary around the technical system, leaving the social, political, and economic interactions beyond the system boundary. This boundary seems too limiting and makes it difficult to differentiate an engineering system with an "engineered" system or complex artifact.

We would like to propose another definition for an engineering system that integrates the ideas presented above. *An engineering system is a complex socio-technical system that is designed, developed, and actively managed by humans in order to deliver value to stakeholders.* An engineering system is complex in that it consists of many interacting parts that exhibit non-trivial behavior. "Socio-Technical" extends the system boundary to include technical and non-technical interactions within the system. These interactions include but are not limited to social, political, technical, and economic interactions. The terms "designed, developed, and actively-managed" extend Moses' word "design", by implying human agent interactions occur beyond the front-end. This also implies there is a system boundary which differentiates what is inside (endogenous) and outside (exogenous) the system. For an engineering system, the extent at which the system can be controlled by the human entities responsible for system synthesis defines the system boundary. Finally, what separates an engineering system from other complex systems is the value delivery aspect of the system. The phrase "in order to deliver value to stakeholders" is an expanded description of Moses' purpose. An engineering system's purpose is to deliver value to stakeholders. When an engineering system ceases to deliver value the system no exists as an engineering system.

Representing an Engineering System:

By definition, an engineering system consists of many inter-related parts that exhibit non-trivial behaviors. As with all complex systems, this creates a problem for those responsible for synthesizing the system. One of the major hurdles when analyzing a complex socio-technical system is bridging the gap between the qualitative and quantitative understanding of the system. Due to the increasing complexity of engineered systems, the engineering community has developed many methods and tools designed manage complexity. Some methodologies include the following: Unified Program Planning, Quality Functional Deployment, Object-Process Methodology, CLIOS method, System Architectures, and the Design Structure Matrix.

Quality Functional Deployment is decision-making methodology created in the 1960s and is still widely used by marketing and engineering design communities to map the following relationships: Customer needs to engineering characteristics, interactions between engineering characteristics, and target values for the engineering characteristics. (Cohen 1995) Analysts use the framework to prioritize customer needs and understand engineering parameter interactions and parameter performance for an engineering system. The methodology ensures that design decisions are aligned with stated customer needs.

QFD is well documented in texts and academic journals and is used extensively as a tool for process improvement and quality management. Some of the strengths of QFD include repeatability, ease of use, and that it provides valuable insights into a product development

effort. There are also several limitations in the methodology. For example, the methodology generally assumes a homogenous set of stakeholders and stakeholder preferences, which is almost never the case for a complex engineering system. Next, the methodology aggregates the technical details of the system into performance parameters, which is a very limited representation of the system. Lastly, QFD does not attempt to represent relationships between the system and the environment. Social, political, and economic factors affecting the system are not explicitly represented in the framework.

In an effort to present a more holistic view of a product development system, John Warfield and Douglas Hill developed a Unified Systems Engineering methodology in the mid-1970s. Hill and Warfield published a seminal paper entitled "Unified Program Planning" (UPP) which proposed the use of matrices to represent the planning efforts for a product development system. (Warfield and Hill 1972) Their methodology was a first attempt to develop a multidisciplinary framework for developing a complex engineered system. (Warfield and Hill 1972) Hill and Warfield expanded the methodology beyond the product development domain and proposed the methodology for use as a policy analysis methodology for non-engineering systems. (Warfield 1973; Warfield 1976) Hill and Warfield created elaborate tools and proposed methods to aid in the development of a complex engineered system. Warfield and Hill's methodologies went far beyond QFD method by including multiple stakeholders, mapping interactions between customer requirements, showing organizational responsibilities, and including social, political, and economic constraints and alterables. Still lacking in the methodology was the absence of a physical architecture, organizational interactions, and interactions between the system and the environment. The tools and methods far exceeded the computational capabilities for the day, thus the qualitative value outweighed tangible quantitative benefits. Despite UPPs promise, it is very difficult to comment on the methodology because there are few documented examples in academic literature.

Since UPPs introduction, John Warfield has developed a general framework for analyzing complex systems called the Science of Generic Design. (Warfield 1990) Warfield presents a philosophical foundation for his science and several theories, methods, and tools (including UPP). Warfield displays a vast knowledge of literature and proposes several theories which cross-disciplinary lines. Like UPP, few examples of this methodology exist in the literature.

More recently, several other methodologies have been developed to represent an engineering system. In the mid-1990's, Dov Dori developed Object-Process Methodology for mapping function to form engineering systems. The methodology uses elaborate graphs of nodes and links to represent an engineering system graphically. Like UPP and QFD methods, OPM maps function (processes) to form (objects). The value of the OPM methodology is the ability represent different types of systems, since all systems can be generically described using processed and objects. (Dori 2002) Bartolomei demonstrated how OPM can be translated into matrices to represent a small engineered system. (Bartolomei 2004)

A limitation of the OPM methodology is the conventions for the different nodes and links are not intuitive and require quite a bit of training in order to understand. In addition, using the methodology to represent a large-scale system is difficult as the networks become very large and difficult to interpret. The methodology is quite new and although many examples exist in the

product/artifact domain, we were did not find examples that apply OPM in representing an complete complex socio-technical system.

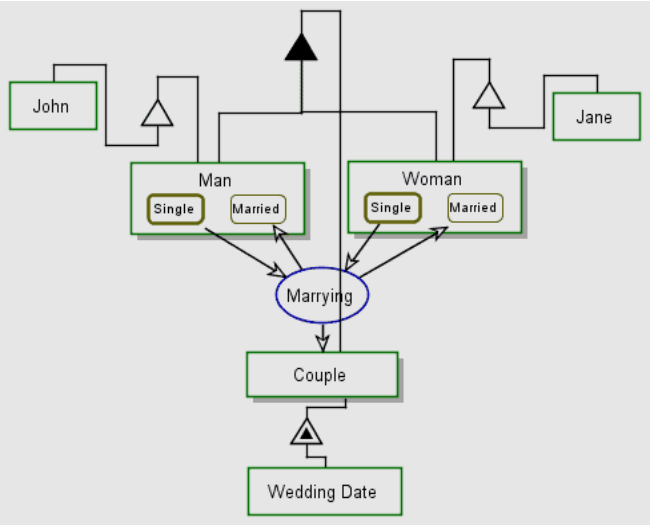


Figure 2. Example OPM of a Wedding

Like OPM, CLIOS is another methodology that uses graphs of nodes and links to represent a complex system. The CLIOS modeling methodology is based on the framework discussed previously. The methodology is still in development and very little documentation is available that describes the theoretical underpinnings of the approach.

Figure 3 describes the methodology for constructing a CLIOS model for a complex socio-technical system. The methodology consists of three phases in constructing a CLIOS model. The first phase is the system representation phase which models the CLIOS structure and behavior. The next phase is the design and evaluation phase that models the performance and options for influencing a CLIOS. The final phase is the implementation phase, where CLIOS analysts and managers consider various strategic options and implement the selected options.

A foundational concept for CLIOS is the idea of nested complexity. Nested complexity refers to the interactions across subsystem layers and elements within the policy sphere of a CLIOS. The term layer seems to have different connotation in CLIOS. Layering in CLIOS refers to a boundary of a physical subsystem within a CLIOS as compared to a more traditional understanding of layering referring to hierarchic relationships described in the complex systems literature. (Simon 1996) The issue of hierarchy and aggregation is not directly addressed in the available literature on CLIOS. Like UPP, the CLIOS Process provides an approach that seeks to represent both physical and policy systems. The methodology seems very promising for both industrial and academic arenas.

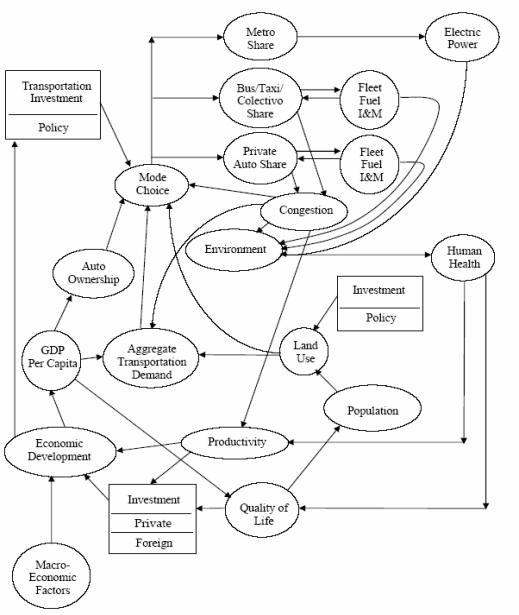


Figure 3. CLIOS Diagram for Mexico City Transportation Subsystem

System Architecture Frameworks

Another common method for representing engineered systems are architecture frameworks. Richards and Shaw provide a detailed examination of the many architecture frameworks currently used to represent systems. (Richards, Shah et al. 2006) Architecture frameworks are designed to manage system complexity by structuring data in a common language and format. System designers use architectures to characterize the form, function, and rules governing systems. System architectures serve as a communication tool to represent different views of a complex system. Each view provides details about the system valuable to the various stakeholders involved in developing or actively managing the system. These stakeholders might include customer, designer, and/or user.

In the DoD, the legally mandated architecture framework is the Department of Defense Architecture Framework (DODAF). The DoDAF consists of multiple views of an engineered system. (Cooper, Ewoldt et al. 2005; Richards, Shah et al. 2006) While the views of the DoDAF are well-defined, little documentation is provided on how the views are to be constructed. This lack of documentation, coupled with a focus on final view outputs in early user training, led to a work product-centric approach to DoDAF development. As a result, many early DoDAF work products were pictures (many done in PowerPoint) that were neither internally consistent nor complete in capturing relevant data. Furthermore, the DoDAF provided a discrete picture of each individual view, thus it is impossible to capture the dependencies and parallelisms among activities, processes, and supporting technologies. (Richards, Shah et al. 2006)

DSM Method

One of the most well known methods for representing a complex engineered system is the Design Structure Matrix methodology. The Design Structure Matrix methodology is an interdisciplinary methodology that seeks to represent a seemingly incomprehensible system on interacting parts in to a simple, compact framework that is conducive for quantitative and qualitative analysis.

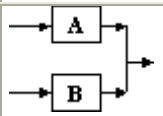
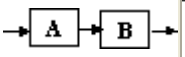
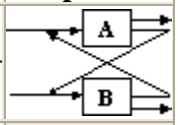
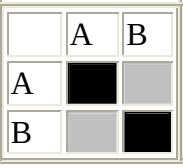
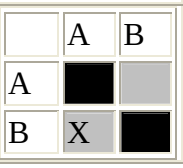
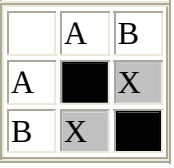
Three Configurations that Characterize a System			
Relationship	**Parallel**	**Sequential**	**Coupled**
Graph Representation			
DSM Representation			

Figure 4. Generic DSM

The figure above provides an example of a graphical representation and DSM representation of nodes and arcs. The Graph and DSM representations in figure 4 show the three fundamental configurations of graphs. The Graph representation consists of boxes (representing nodes) and lines with arrows (representing links). The DSM representation is simply a matrix representation of the graphs.

One advantage of the DSM methodology, as compared to the node and link representation of methods, like OPM and CLIOS, is for large networks interpreting network graph is very difficult. Recent studies have shown that the Matrix-Based Representations are more readable than Node-Link representation. (Ghoniem, Fekete et al. 2004)

Evolution of DSM Literature:

The DSM literature begins with Donald Steward's seminal work, where he first describes a Design Structure System methodology for analyzing a complex system. (Steward 1981) Steward used the DSS to represent the structure of a complex systems and applied mathematical methods to reorder and group elements to create a more efficient design. In 1990, Steve Eppinger and a team of MIT researchers resurrected the DSM through a series of theoretical and empirical studies. The MIT team's findings opened the door to a vibrant vein of research that is active today. Eppinger et al's first efforts applied the DSM to the design tasks for a product development to improve efficiency and reduce cycle-time. (Eppinger, Whitney et al. 1990; Eppinger 1991; Gebala and Eppinger 1991) The team used a DSM to map the process activities for the construction of a product system. The DSM was used to identify unnecessary feedback relations within the system. Steward's mathematical methods were applied to reorder the design activities in order to minimize the feedback within the system. This resulted in a more streamlined development process and reduced development cycle-time.

The methodology was extended to other aspects of the product development system. McCord and Eppinger proposed a team-based DSM to analyze the organizational structure necessary for an improved automobile engine development process.(McCord, Eppinger et al. 1993) Similar work was performed on military aircraft development projects. (Rowles 1999) DSM analysis was used to examine alignment structures between a product systems and the design organization responsible for developing the system. (Sosa 2000)

In addition to the task and organizational views of the system, DSM methodology was applied to analyze physical systems. Pimmler and Eppinger developed the component-based DSM that represented the physical interactions between elements in complex system architecture. (Pimmler and Eppinger 1994) This work served as a basis for examining modularity in design and is presented in Baldwin and Clark's Design Rules research. (Baldwin and Clark 2000) The method is also found in product platforming and change propagation literatures as well. (Clarkson, Simons et al. 2001; Eckert, Clarkson et al. 2004; Suh 2005)

The DSM community currently recognizes the following DSM types: Component-Based DSM, Team-Based DSM, Activity-Based DSM, and Parameter-Based DSM. The DSM types are shown in Table 1.

DSM Data Types	Representation	Application	Analysis Method
Component-based	Multi-component relationships	System architecting, engineering and design	Clustering
Team-based	Multi-team interface characteristics	Organizational design, interface management, team integration	Clustering
Activity-based	Activity input/output relationships	Project scheduling, activity sequencing, cycle time reduction	Sequencing & Partitioning
Parameter-based	parameter decision points and necessary precedents	Low level activity sequencing and process construction	Sequencing & Partitioning

Table 1. Traditional DSM Views

DSM State-of-the-Art

Today, there are over one hundred papers demonstrating the value and/or extending to use of DSM's to represent physical, task, and organizational views of an engineered systems. (Browning 2001) Below are few examples that are most aligned with our research interests.

Browning extended the DSM method to explore the dynamic behavior of an engineered system by using DSM and simulation to operationalize an aircraft product development program model. (Browning 1998) Sosa demonstrated how to examine technical communication within an aircraft development program by comparing an organizational DSM with the physical DSM. (Sosa 2000) Lastly, Sharman and Yassine demonstrated how DSM and Real Options can be used to examine flexibility in product architecture. (Sharman, Yassine et al. 2002; Sharman and Yassine 2004)

Extending the DSM Methodology

As demonstrated by the vast literature, the DSM methodology has proven effective for representing and analyzing complex systems. This research hopes to extend current DSM practice along two dimensions. First, this research proposes a framework that couples traditional

DSM views to represent traceability and endogenous interactions across an entire engineering system. Second, the framework proposes two new DSM views of the system: the Stakeholder DSM and System Drivers DSM.

Stakeholder DSM:
The Stakeholder DSM represents the social network of stakeholders in an engineering system. Within the Stakeholder DSM, there are internal and external stakeholders. By definition, the external stakeholders are stakeholders who prescribe value for the system, but do not control the system. Likewise, internal stakeholders are the human entities who prescribe value, interpret the value proposition of external stakeholders, and can control the system. The extent of the internal stakeholder's control of the system defines the system boundary. For example, in the case of an F-16 in operational combat, the pilot would be an example of an internal stakeholder, who prescribes value for the function of the system and interprets the needs of external stakeholders (the squadron commander). The extent to which the pilot and the other internal stakeholders are able to control the system would be included within the system boundary. Conversely, another ESM could be constructed at a higher level of aggregation. In this example, the new ESM represents an F-16 squadron. A new system boundary is established. For the F-16 squadron, the commander now becomes the internal stakeholder interpreting the objectives established by the combatant commander (external stakeholder) and the pilot mentioned above is an agent represented in the Team DSM.

System Drivers DSM
The variables beyond the control of the internal stakeholders that exist outside the system boundary are called system drivers. The system drivers include social, economic, political, and technical influences that affect the behavior of the system. Examples of these variables in the F-16 example might include price of jet fuel, weather, or a new surface-to-air threat. The addition of a System Drivers DSM is a unique aspect of this research as compared to traditional DSM methods. Traditional DSM methods are applied within the boundary of a product system. This is because DSM practitioners are primarily concerned with endogenous issues such as, eliminating rework, removing unnecessary parts, and improving communications. What makes the discipline of engineering systems unique is the inclusion of social, political, and economic factors with the technical system. It is only when one considers an engineered system in the context of these factors can one can affect system properties like flexibility (capable to undergo specified classes of change with ease), adaptability (able to change self), and robustness (able to operate in different environments w/o degradation in performance). The next section explains how DSM views can be coupled to create an end-to-end representation of an engineering system called an Engineering Systems Matrix (ESM).

The Engineering Systems Matrix
The ESM is a proposed framework for organizing the four different DSM-types, with other DSMs (e.g. Stakeholder DSM and System Drivers DSM), into a single view that captures traceability and interrelationships between views of an engineering system. The goal in constructing an ESM is to eliminate the epistemic uncertainty surrounding a complex socio-technical system. Epistemic uncertainty is defined as the uncertainty surrounding the system that can be known. (Ben-Haim 2001) The goal is to capture a qualitative understanding of a system into a format that is conducive for quantitative analysis.

Each of the DSM types is represented in the ESM framework as shown in Figure 5.

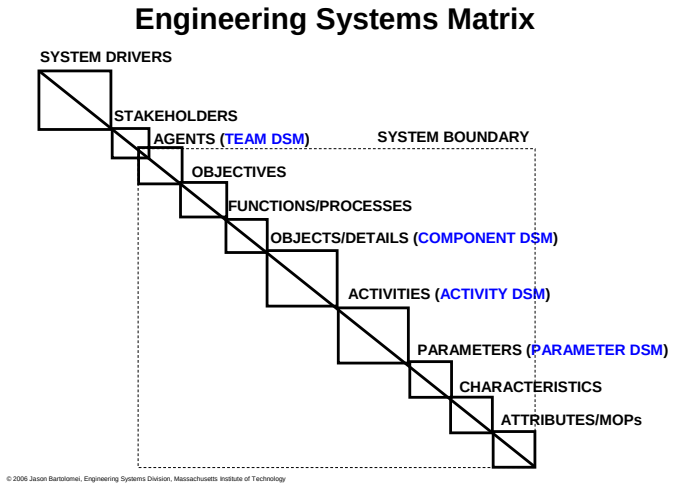


Figure 5. The Engineering Systems Matrix

A brief description of each DSM within an ESM is as follows:

- The Stakeholder DSM consists of nodes representing individuals or organizations. The links between stakeholder represent traditional social network links that can include monetary transactions, communication, etc.
- In the Objectives DSM, nodes are the objectives of the system. The links represent causal interactions. For example, a +1 represents a positive relationship, a 0 represents no relationship, and a -1 represents a negative relationship.
- In the Functions DSM, nodes represent the functions for the system. The links represents hierarchical relations between functions. Additional information regarding material, information, spatial and energy relations can also be included if desired.
- In the Components DSM, nodes represent the objects in the system. Physical network conventions can be used to represent links (e.g. physical, material, electrical, etc)
- In the Activities DSM, the nodes represent the activities performed in order to accomplish the system objectives. There are a variety of link representations for activities ranging from cost, schedule, and performance link attributes.
- In the Agents DSM, nodes represent individuals and/or organizations at work inside the engineering system. Social network conventions can be used to represent links.
- In the System Parameters DSM, the nodes represent the system parameters for the internal stakeholders, objects, activities, and agents. The links represent a relation between nodes.
- In the System Characteristics DSM, the nodes are the characteristics. The links are causal or mathematical relations between nodes.
- In the Attributes and Measures of Performance DSM, the nodes are the system attributes and MOPs. The links are causal or mathematical relations between nodes.
- In the System Drivers DSM, the nodes are social, political, economic, and technical system influences that are beyond the internal stakeholders control. The links can represent causal or mathematical relations.

Once the ESM is created, the system analyst has an end-to-end representation of the system that contains the critical system variables, interrelationships between variables, and boundaries for

the system. These are the essential elements for both qualitative and quantitative modeling methods. The next phase of this research seeks to use this information to identify and quantify spots in the engineering system that are candidates for real options.

Real Options "In" an Engineering System

A "real option" is defined in the finance literature as the right, but not the obligation, to take an action (e.g. deferring, expanding, contracting, or abandoning) at a predetermined cost and for a predetermined time. (Copeland and Antikarov 2003) Some experts believe the more real options that exist within a complex system, the more flexible the system will be to respond to future uncertainty. (de Neufville 2001) A common definition for flexibility in engineering systems is as follows: flexibility is the property of a system that allows it to endure sets of changes with ease. (Saleh 2001; Moses 2003; Whitney 2004; Rajan, Van Wei et al. 2005) This research hopes to contribute to the real options literature by providing a methodology for screening an engineering system to identify existing options and/or identify where to design options into the system. The hope is by analyzing the ESM system designers will enable to identify the real options within an engineering system.

Identifying Important Nodes

A working hypothesis guiding this research is that the qualities of certain nodes within the ESM are better candidates for real options than other nodes. The nodes that are best candidates for real options are called "Hot" spots in the ESM, while the nodes that do not require options are called "Cold" spots. Ongoing research seeks to determine the qualities of the "Hot/Cold" spots.

Although the idea of "hot" and "cold" spots is not explicitly mentioned in the literature, there are several well developed concepts from the risk/safety management and product design literature that inform this research. Within the risk and safety management literatures provides the risk matrix methodology. A risk graph, see figure below, is a method for examining each node within a system for its level of risk. Risk is defined as the product of the likelihood of a change (failure) occurring and the impact of that change. (Alexander 1996) In the Risk Matrix, nodes in the upper right would be "hotter" and nodes in the bottom left would be "cooler".

In the product design literature, several recent studies are of particular interest. Clarkson, Simons et al developed a methodology using design matrices to calculate the likelihood and effect of changes propagating through a rotorcraft design using impact and likelihood DSMs for the physical system. (Clarkson, Simons et al. 2001) Eckert, Clarkson et al developed a change management framework that classifies types of changes and causes of change for complex engineered systems. (Eckert, Clarkson et al. 2004) Martin and Ishii propose a methodology called Design for Variety (DFV). (Martin and Ishii 2002) The authors propose a Generational Variety Index (GVI) and Coupling Index (CI) for product design elements. A GVI measures the amount of redesign required for a component to meet future market requirements. The CI is a measure of the strength of coupling between components in a product. The stronger the coupling between elements, the more likely changes in one will affect the other tightly coupled elements. Suh further develops these ideas by demonstrating how to pin point and value flexible elements in automotive platforms by considering both technical and economic aspects of the product system. (Suh 2005)

Rajan, Van Wei et al developed a methodology for measuring flexibility based on an empirical analysis of 20 simple engineered systems. (Rajan, Van Wei et al. 2005) In their analysis, they propose a Change Mode and Effects (CMEA) analysis for defining the Change Potential Number (CPN) for each element in the physical design. The CPN is based on number of factors including: likelihood of occurrence, readiness for change, and number of possible change modes. Each of these studies provides potentially useful approaches for identifying "hot" spots within a technical system.

The scope of previous research is generally limited to the product domain with very little consideration given to elements beyond the physical architecture. This research hopes to extend these existing methods and develop "hot/cold" spot criteria and measures for each DSM-type within the ESM. In addition, this research hopes to develop a repeatable methodology for screening an engineering system for these spots.

Figure 6 represents a notional illustration of "Hot/Cold" spots within an ESM. The intensities of colors represent the magnitude of hotness/coldness associated with the different nodes in the system.

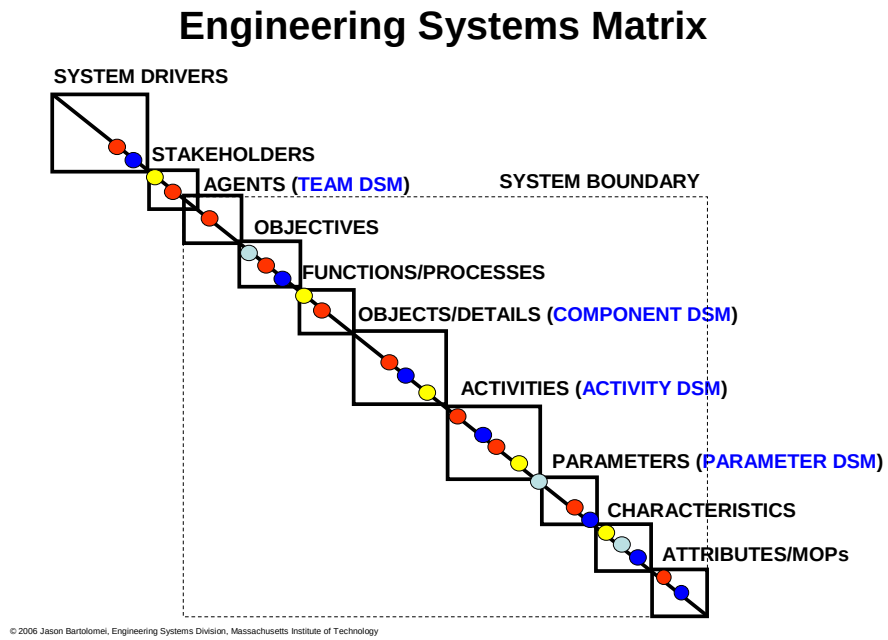


Figure 6. "Hot" and "Cold" Spots within a Engineering System Matrix

Ongoing Research:

Current research efforts focus on applying the ESM methodology to model a Miniature Air Vehicle (MAV) product development program in its entirety. The MAV was chosen because it has many advantageous characteristics. First, the MAV is a well-documented system that can be nearly completely described using the DSM types mentioned above. Second, the MAV is a rapidly evolving technology where system changes are relatively easily to observe. Lastly, the non-physical elements surrounding the MAV system, such as stakeholders dynamics,

development activities and operational applications, are so complex as to require an engineering systems perspective on system.

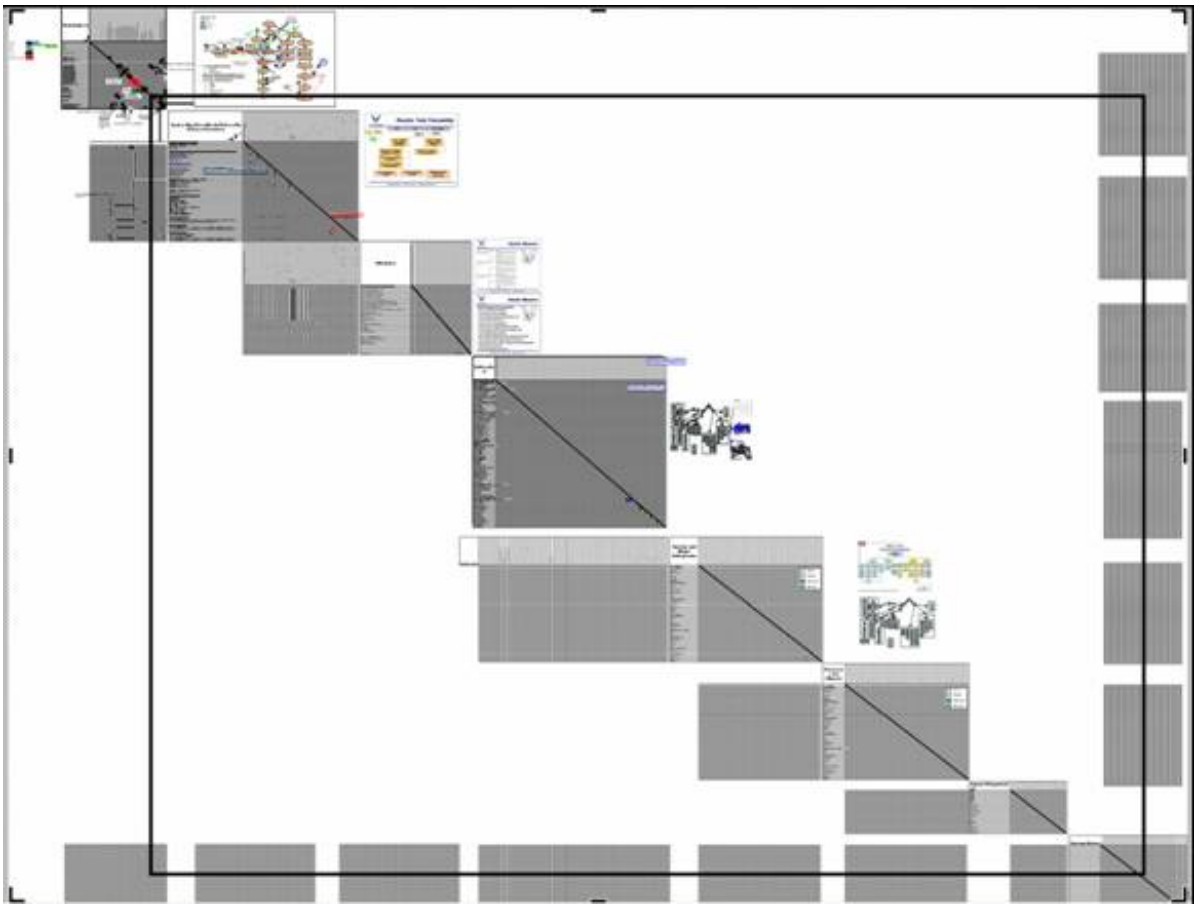


Figure 7. Current progress on MAV ESM

Figure 7 represents a current view of the MAV ESM. Data has been collected to map the evolution of the system since its inception. The method of analysis to identify the hot/cold spots is ongoing research.

Examples of Hot and Cold Spots:

During our preliminary analysis of the MAV engineering system, we have identified a few potential hot and cold spots. For example, within the organization we identified a few nodes in the social network that appear to be central to the development of the system. One agent was of particular instance since she was highly connected to several nodes in the engineering system and was very likely to leave the program in the near future. With this knowledge, program leadership must consider real options to ensure the system will progress effectively in the event that she leaves the system. A preliminary example of a cold spot in the engineering system was found in the engineered portion of the system. Using multidisciplinary optimization we identified the fuselage as a likely cold spot in the architecture. Based on the analysis, it was found that for a variety of wing configurations, the characteristics of the fuselage remained relatively unchanged to maximize operational performance. This analysis combined with information regarding stakeholder objectives, system constraints, and the pace of payload

technology reveals that the fuselage design was more stable than other elements of the system and may not require an investment in real options. Ongoing research seeks to develop an analytical method for identifying these spots both qualitatively and quantitatively.

Biographies

Captain Jason Bartolomei is an USAF acquisitions officer with experience as a systems engineer and an integrated product team lead at the F/A-22 System Program Office. Most recently he served as an Assistant Professor of Engineering Mechanics and Director of Systems Engineering at the US Air Force Academy. He is currently a PhD Candidate in MIT's Engineering Systems Division.

Daniel Hastings is a Professor of Engineering Systems and Aeronautics and Astronautics at MIT. Dr. Hastings has taught courses and seminars in plasma physics, rocket propulsion, advanced space power and propulsion systems, aerospace policy, technology and policy, and space systems engineering. He served as chief scientist to the U.S. Air Force from 1997 to 1999 and as director of MIT's Engineering Systems Division from 2004 to 2005. He is a member of the National Science Board, the International Academy of Astronautics, the Applied Physics Lab Science and Technology Advisory Panel, and the Air Force Scientific Advisory Board.

Richard de Neufville is a Professor of Civil and Environmental Engineering and Engineering Systems at MIT. He was the Founding Chairman of the MIT Technology and Policy Program, and author of 5 texts on systems analysis in engineering. This work has been recognized by a Guggenheim Fellowship, the NATO Systems Science Prize; the Sizer Award for the Most Significant Contribution to MIT Education, the Martore and MIT Effective Teaching Awards, and the US Federal Aviation Award for Excellence in Teaching. The French Government made him a Chevalier des Palmes Académiques. He has a Ph.D. from MIT and a Dr. h.c. from the Delft University of Technology.

Donna Rhodes holds a Ph.D. in Systems Science from the T.J. Watson School of Engineering at SUNY Binghamton. Her research interests are focused on systems engineering, systems management, and enterprise architecting. Dr. Rhodes has 20 years of experience in the aerospace, defense systems, systems integration, and commercial product industries. Prior to joining MIT, she held senior level management positions at IBM Federal Systems, Lockheed Martin, and Lucent Technologies in the areas of systems engineering and enterprise transformation. Dr. Rhodes is a Past-President and Fellow of the International Council on Systems Engineering (INCOSE).

Alexander, C. (1996). The handbook of risk management and analysis. Chichester ; New York, John Wiley.

Baldwin, C. Y. and K. B. Clark (2000). Design Rules, Vol 1: The Power of Modularity. Cambridge, MA, MIT Press.

Bartolomei, J. E. (2004). Using Design Structure Matrices to Represent an Object-Process Methodological View of a Mini Uninhabitated Air Vehicle, Massachusetts Institute of Technology**:** 30.

Ben-Haim, Y. (2001). Information-gap Decision Theory: Decisions Under Severe Uncertainty. San Diego, Academic Press.

Browning, T. R. (1998). Modeling and Analyzing Cost, Schedule, and Performance in Complex System Product Development. TMP. Cambridge, MIT.

Browning, T. R. (2001). "Applying the design structure matrix to system decomposition and integration problems: a review and new directions." IEEE Transations on Engineering Management **48**(3).

Clarkson, P. J., C. Simons, et al. (2001). Predicting Change Propogation in Complex Design. ASME 2001 Seign Engineering Technical Conferences and Computers and Information in Engineering Conference, Pittsburg PA.

Cohen, L. (1995). Quality Function Deployment: how to make QFD work for you. Reading MA, Addison Wesley Longman, Inc.

Cooper, C. A., M. L. Ewoldt, et al. (2005). A Systems Architectural Model for Man-Packable/Operable Intelligence, Surveillance, and Reconnaissance Mini/Micro Aerial Vehicles. Department of Aeronautical Engineering. Wright-Patterson Air Force Base OH, AFIT.

Copeland, T. E. and V. Antikarov (2003). Real options : a practitioner's guide. New York, Texere.

de Neufville, R. (2001). "Real Options: Dealing With Uncertainty In Systems Planning and Design." Intergrated Assessment **4**(1): 26-34.

Dori, D. (2002). Object-Process Methodology: A Holistic Systems Paradigm. Verlag, Springer.

Eckert, C., P. J. Clarkson, et al. (2004). "Change and customisation in complex engineering domains." Research in Engineering Design **15**: 1-21.

Eppinger, S. D. (1991). "Model-based Approaches to Managing Concurrent Engineering." Journal of Engineering Design **2**: 283-290.

Eppinger, S. D., D. E. Whitney, et al. (1990). "Organizing Tasks in Complex Design Projects." MIT Working Paper **WP# 2083-89-MS**.

Gebala, D. A. and S. D. Eppinger (1991). Methods for Analyzing Design Procedures. ASME Third International Conference on Design Thoery and Methodology.

Ghoniem, M., J. D. Fekete, et al. (2004). A Comparison of the Readability of Graphs Using Node-Link and Matrix-Based Representations. Proceedings of InfoVis 2004, Austin TX.

Martin, M. V. and K. Ishii (2002). "Design for variety: developing standardized and modularized product platform architectures." Research in Engineering Design **13**: 213-235.

McCord, K. R., S. D. Eppinger, et al. (1993). Managing the integration problem in concurrent engineering. Cambridge, MA, International Center for Research on the Management of Technology Sloan School of Management Massachusetts Institute of Technology.

Moses, J. (2003). The Anatomy of Large Scale Systems. ESD Internal Symposium. Cambridge, MA, MIT**:** 17.

Moses, J. (2004). Foundational Issues in Engineering Systems: A Framing Paper. Engineering Systems Monograph. Cambridge MA**:** 16.

Pimmler, T. U. and S. D. Eppinger (1994). Integration analysis of product decompositions. Cambridge, Mass., Alfred P. Sloan School of Management Massachusetts Institute of Technology**:** 39.

Rajan, P. K., M. Van Wei, et al. (2005). "An empirical foundation of product flexibility." Design Studies **26**: 405-438.

Richards, M. G., N. B. Shah, et al. (2006). Managing Complexity with the Department of Defense Architecture Framework: Development of a System Archtiecture Model. CSER 2006, Los Angeles CA.

Rowles, C. M. (1999). System integration analysis of a large commercial aircraft engine**:** 106.

Saleh, J. H. (2001). Weaving time into system architecture : new perspectives on flexibility, spacecraft design lifetime, and on-orbit servicing**:** 224 leaves.

Sharman, D. M. and A. A. Yassine (2004). "Characterizing Complex Product Architectures." Systems Engineering **7**: 35-60.

Sharman, D. M., A. A. Yassine, et al. (2002). Architectural Optimisation Using Real Options Theory and Dependency Structure Matrices. ASME 2002 International Design Engineering Technical Conferences: 28th Design Automation Conference, Montreal, Canada.

Simon, H. A. (1996). The sciences of the artificial. Cambridge, Mass., MIT Press.

Sosa, M. E. (2000). Analyzing the effects of product architecture on technical communication in product development organizations**:** 152.

Steward, D. V. (1981). "The Design Structure System: A Method for Managing the Design of Complex Systems." IEEE Transactions on Engineering Management **28**: 71-74.

Suh, E. S. (2005). Flexible Product Platforms. Engineering Systems Division. Cambridge MA, MIT.

Sussman (2000). Toward Engineering Systems as a Discipline. Cambridge, Massachusetts Institute of Technology**:** 6.

Warfield, J. N. (1973). An assault on complexity. Columbus, Ohio,, Battelle Office of Corporate Communications.

Warfield, J. N. (1976). Societal systems : planning, policy and complexity. New York, Wiley.

Warfield, J. N. (1990). A science of generic design : managing complexity through systems design. Salinas, Calif., Intersystems Publications.

Warfield, J. N. and J. D. Hill (1972). "Unified Program Planning." IEEE Transactions on Systems, Man, and Cybernetics--Part A **SMC-2**(5): 14.1-14-14.

Warfield, J. N. and J. D. Hill (1972). A unified systems engineering concept. Columbus, Ohio,, Battelle Office of Corporate Communications.

Whitney, D. C., Edward; de Weck, Olivier; Eppinger,Steven; Magee,Christopher;Moses,Joel;Seering, Warren;Schindall,Joel;Wallace,David (2004). The Influence of Architecture in Engineering Systems. ESD Symposium. Cambridge, MA, MIT**:** 30.