Dynamic System Perspective for Design: Ility-Driving Elements as Responses to Uncertainty

by

Nicola Ricci

B.S. Aerospace Engineering
Boston University, 2011

Submitted to the Department of Aeronautics and Astronautics and Engineering Systems Division
in Partial Fulfillment of the Requirements for the Degrees of

Master of Science in Aeronautics and Astronautics
Master of Science in Technology and Policy

at the

Massachusetts Institute of Technology

June 2014

Signature redacted

Signature of Author..
Department of Aeronautics and Astronautics, Engineering Systems Division
May 22, 2014

Certified by.. Signature redacted ....
Adam M. Ross
Research Scientist, Engineering Systems
Lead Research Scientist, Systems Engineering Advancement Research Initiative
Thesis Co-Advisor

Signature redacted

Certified by.. ....
Donna H. Rhodes
Principal Research Scientist and Senior Lecturer, Engineering Systems
Director, Systems Engineering Advancement Research Initiative
Thesis Supervisor

Certified by... Signature redacted ....
Daniel E. Hastings
Cecil and Ida Green Education Professor of Engineering Systems
Aeronautics and Astronautics Academic Advisor

Accepted by.. Signature redacted ....
Paulo Lozano
Associate Professor of Aeronautics and Astronautics
Chair, Graduate Program Committee, Department of Aeronautics and Astronautics

Accepted by.. Signature redacted ....
Dava J. Newman
Professor of Aeronautics and Astronautics and Engineering Systems
Director, Technology and Policy Program

Dynamic System Perspective for Design:

Ility-Driving Elements as Responses to Uncertainty

by

Nicola Ricci

Submitted to the Department of Aeronautics and Astronautics and Engineering Systems Division
on May 22, 2014 in Partial Fulfillment of the Requirements for the Degrees of

Master of Science in Aeronautics and Astronautics
Master of Science in Technology and Policy

Abstract

This thesis is concerned with the design of complex artificial systems. For such systems, there is a growing need to deliver value to stakeholders beyond the initial functional requirements and to cope with rapidly changing outer environments. This thesis presents a conceptual framework and a structured approach for thinking about and designing systems that can exhibit the emergence of desirable lifecycle properties (i.e., ilities). To set the ground for the research contributions, a literature overview on (1) complex sociotechnical systems, (2) uncertainty in such systems, and (3) ways to cope with such uncertainty is given. Furthermore, the larger research effort concerning a method for architecting Systems of Systems with ilities is discussed to frame the remainder of the thesis.

The dynamic system perspective for design is discussed, as well as a formal way of modeling the space of possibilities for designers of complex systems (i.e., what the system can be, as well as what its outer environment and expectations can be). In this perspective, uncertainty is modeled as perturbations, which are operators on these spaces of possibilities. Similarly, ility-driving elements (IDEs) are introduced and modeled as operators on such spaces as well. Two main types of ility-driving elements are discussed and formally defined: change options and resistance properties. The former, akin to real options in business, enable the system to change over time so as to cope with perturbations and sustain (or enhance) value delivery. The latter, on the other hand, impede undesired changes in system value delivery.

Lastly, IDE Analysis – a structured approach for generating, evaluating and selecting ility-driving elements – is introduced, and demonstrated on a running case application to a Maritime Security System of Systems. This approach requires an initial baseline design concept, and considers a set of relevant perturbations as a starting point. The thesis ends with general discussions around applicability of research and possible areas for future research, as well as conclusions regarding key contributions.

Thesis Supervisor: Dr. Donna H. Rhodes
Title: Principal Research Scientist and Senior Lecturer, Engineering Systems

Thesis Co-Advisor: Dr. Adam M. Ross
Title: Research Scientist, Engineering Systems

A *mamma e papà*, e la mia famiglia tutta – il mio centro di gravità permanente.

Acknowledgements

The past three years have been an extraordinary experience. The vibrant intellectual community I have found at MIT has shaped my thought in new and unexpected ways, and critically broadened my perspective on life and what can be known. At MIT, I have met the most eclectic array of people – from philosophers, to designers, to scientists – who have all contributed to my personal growth in some way. If you are one of them – and you would know it – thank you!

First and foremost, I'd like to thank **Dr. Adam Ross** and **Dr. Donna Rhodes**, who have given me the opportunity of working alongside them in the pursuit of a paradigm shift. Thank you, Adam, for the mind-blowing conversations and the brilliant insights you have offered me these three years. I can confidently say that you are the person that has contributed the most to my intellectual growth. Thank you, Donna, for your invaluable advice and support – on the professional and personal level – during this time. I didn't just feel appreciated for my work, but also cared for, and this I will never forget. Furthermore, I'd like to thank **Professor Daniel Hastings** for his academic mentorship and the pearls of wisdom I have gathered from each interaction with him.

A big thank you goes to all the members of SEAri – those I have met and interacted with, and those I haven't met, but that have laid the foundation for this research effort. In particular, I'd like to thank **Nirav Shah**, constant source of knowledge and ideas; **Matt Fitzgerald**, for making sure my conceptual frameworks were sound and for giving in to Italian soda; **Marcus Wu**, with whom I shared the Dual AeroAstro-TPP experience, supporting each other through thick and thin; **Paul Grogan**, for letting me pick his brilliant mind, and always providing invaluable feedback for improvement. A special thank you goes to **Michael Schaffner**, with whom I have shared the office for the past year, and from whom I have learned (mostly unlearned) a lot – but never admitted it. Mike, I know I will miss our conversations very much, but I also hope that this won't be the last time we share an office. Additional thanks, for their valued input, go to **Dan Fulcoly**, **Clark Beesemyer**, **Brian Mekdeci**, **Matt Frye**, **Hank Roark**, **Henrique Gaspar**, **Jeff Mekler**, **Li Qian Yeong** (the best cupcakes!), **Paul LaTour**, **Ben Putbrese**, **Michael Curry** (keep plucking that chicken!), **Alex Pina**, and **Hunter Zhao**.

Without **Pasquale Freda**, I would not be here now. Much of my gratitude goes to the person that has made it possible for me to come study in the United States from a little village in the South of Italy. Pasquale deeply and permanently changed the equations of motion underlying my life path, and I will be forever grateful to him.

The largest source of inner energy has always been my family: **mamma**, **papà**, **zio Salvatore**, **zia Rocchina**, **zia Maria**, **nonno Nicola**, **nonna Letizia**, **nonno Giovanni**, **nonna Carolina**, **zia Mena**, **zio Biagio**, **Gianrocco**, **Fede**, **Erica**, **Dino**... You are the strong foundation on which everything in my life rests. Thank you all so much for the unconditional love.

Speaking of which, I'd like to thank **Flavia** for her unconditional love and support during this time. I cannot stop thanking whatever entity has made it possible for us to cross paths. Here's to the new chapter in our lives that we'll soon start together… :*

The author gratefully acknowledges the financial support for this research from the Systems Engineering Advancement Research Initiative (SEAri), a part of the MIT Sociotechnical Systems Research Center (SSRC).

Biographical Note

"Un paese ci vuole, non fosse che per il gusto di andarsene via. Un paese vuol dire non essere soli, sapere che nella gente, nelle piante, nella terra c'è qualcosa di tuo, che anche quando non ci sei resta ad aspettarti."

– Cesare Pavese (1950)

Nicola Ricci was born and raised in Vallata, a small rural village in Irpinia, the heart of Southern Italy. In the quietness of his hometown, he grew up helping nonno Giovanni and nonna Carolina with their farm, playing soccer, and wandering down the streets of his neighborhood with his friends. In high school, he developed a strong passion toward theoretical sciences, especially math and physics, as well as philosophy and art history. In October 2006, during his senior year, he was exposed to an opportunity that would drastically alter the course of his life: the chance to win a full scholarship to attend Boston University. He went for it...

On July 14th, 2007, he left Vallata – all he ever knew or had – to fly to Boston. He had never spoken English before, but promised himself he'd learn. By January 2008, he was able to commence his undergraduate studies in Aerospace Engineering. As part of his undergraduate experience, he spent six months at the Technische Univerisität in Dresden (Spring 2009) for a study abroad program. The summer of 2010 he spent at the MIT SEAri lab, where he first got acquainted with (and enchanted by) the field of system design. In May 2011, he graduated from Boston University, and headed into a much-needed 3-month break in his beloved Southern Italy, before starting a Masters degree program at MIT in Aeronautics and Astronautics.

At MIT, he was exposed to a variety of disciplines and worldviews, which kept on broadening his horizons. While continuing his formal education in aerospace engineering (and the elegant reductions of physical reality therein), he became fascinated with the more open, sociotechnical systems, and decided to go for a second Masters degree in Technology and Policy, in the Engineering Systems Division. During these three years, Nicola published six articles on a variety of topics, ranging from strategies for designing systems with ilities to the usefulness of interactive visualization in the process of building stakeholder value models. This thesis presents only part of the academic journey Nicola has undergone in his graduate experience.

In the fall of 2014, Nicola will start a job as Data Scientist in New York City. He'll have to leave Boston, his adoptive city, which will be painful. However, he is confident great things are awaiting him in Brooklyn...

Table of Contents

List Of Figures

List of Tables

1 Introduction

"Nondimanco, perché il nostro libero arbitrio non sia spento, iudico poter essere vero che la fortuna sia arbitra della metà delle azioni nostre, ma che etiam ne lasci governare l'altra metà, o presso, a noi..."

– Niccolò Machiavelli (1532)

The term "design" comes from the Latin word "designare," a composite of "de"- (i.e., "out") and "signare" (i.e., "to mark"). Hence, at its core, it symbolizes the act of marking out, of defining – omnipresent in human life. Within what is in their might, human beings are always designing the next course of actions to undertake.

This thesis is concerned with the design of complex artificial systems. The designer of such systems must cope with three important moving targets: (1) the system, (2) its purpose, and (3) its outer environment (Simon, 1996). However, he or she holds control over the design of only one of them, the system. If the latter two were in perennial stasis (an ineffable state of affairs), the task of designing would be simple (relatively to the structural complexity of the system). However, if there is a truth one can grab on, that is the fact that things constantly change. This fact is the very basis and the very curse of system design.

1.1 Motivation

This section explains the motivation for this thesis. The hardship of designing complex systems is described, and some relevant empirical examples are given. Lastly, a summary of the main needs driving this research effort is given.

1.1.1 Complex Artificial Systems

Artificial systems are those designed and instantiated by human beings, rather than occurring naturally (Simon, 1996). The concept of complexity in such systems is multifaceted: it can be related to the structure of the system, to its interaction with the outer environment, to the way it is perceived, and so on (a more in depth description of the different types of complexity affecting system design is given in chapter 2). For systems that are very complex and feature a pronounced social component, the relationship among the three terms of design (system, purpose, and outer environment) becomes blurry very rapidly, making the task of design difficult. Complex artificial systems lie on a spectrum that goes from the mostly technical (e.g., cars, aircraft) to the sociotechnical (e.g., transportation systems, some military Systems of Systems, policy decisions). Considerations, frameworks and analyses in this thesis apply to systems that span across this spectrum.

Modern-day systems operate in highly dynamic and interconnected environments: technologies improve at an increasing pace; geo-political scenarios change; economies and markets fluctuate. If focused solely on the present state of affairs, designers may incur into designing systems that operate in contexts for which they were not intended. Furthermore, such systems may end up delivering capabilities that are no longer of interest to stakeholders. A report from the Systems Engineering Research Center (to the U.S. Department of Defense) summarizes this problem eloquently (Deshmukh et al., 2010, pp. 9):

> Complex DoD systems tend to be designed to deliver optimal performance within a narrow set of initial requirements and operating conditions at the time of design. This usually results in the delivery of point-solution systems that fail to meet emergent requirements throughout their lifecycles, that cannot easily adapt to new threats, that too rapidly become technologically obsolete, or that cannot provide quick responses to changes in mission and operating conditions.

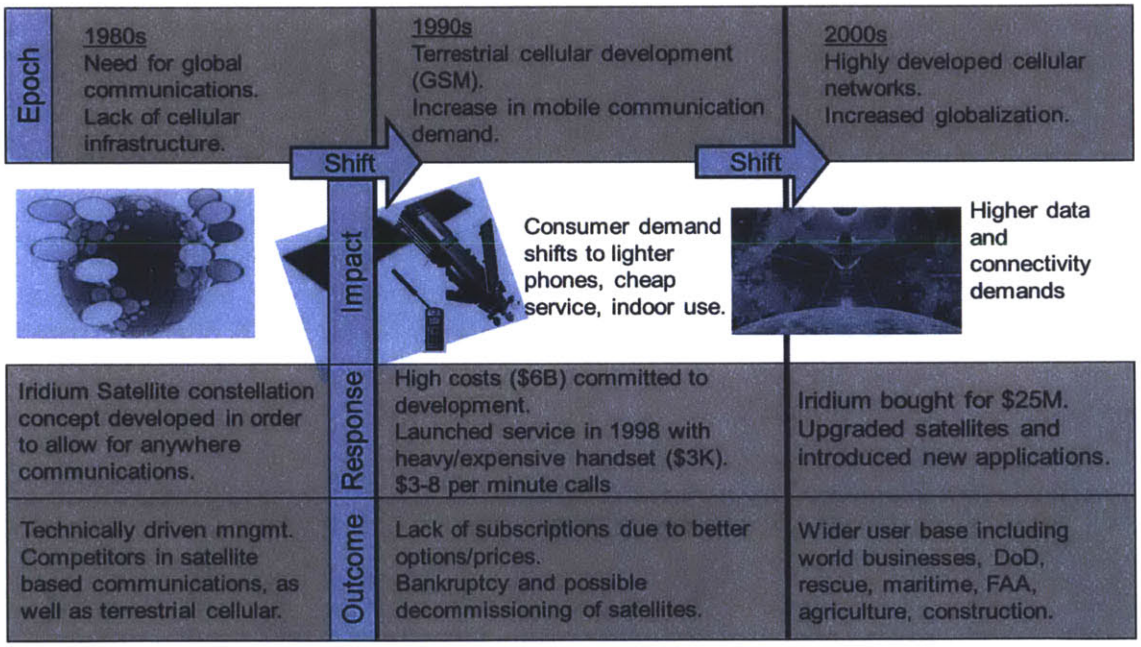
Hence, a designer must think not only about the immediate functional requirements, but also about the ability of the system to endure in a rapidly changing environment, with (possibly) changing goals. As Neches and Madni (2012) state in their manifesto on Engineered Resilient Systems (ERS), "it is important to realize that, when needs are prematurely translated into requirements or key performance parameters, both the process and the product of engineering suffers the consequence." However, traditional systems engineering[1] tends to define and freeze requirements early in the conceptual design phase. This happens mainly because of the need for clear guidance in terms of the functions and performance levels that are to be achieved by the system. As de Weck, Eckert, and Clarkson (2007) point out, "one of the dangers of this approach, when applied blindly, is that it does not adequately address situations where there is uncertainty."

1.1.2 On the Effects of Change and Uncertainty

The presence of irreducible uncertainty, coupled with the absence of strategic ways to respond to it, has led to some systems that have failed to achieve their operational goals over time. In other cases, such uncertainty was well managed by systems. Some important examples are described below.

[1] "The process of selecting and synthesizing the application of the appropriate scientific and technical knowledge in order to translate system requirements into system design." (Chase, 1984) As opposed to "advanced" systems engineering (treated herein): "a branch of engineering that concentrates on design and application of the whole as distinct from the parts... looking at the problem in its entirety, taking into account all the facets and variables and relating the social to the technical aspects." (Booton and Ramo, 1984)

Iridium. The Iridium satellite constellation was initially conceived as a way to provide a global communications system that would enable people to communicate by telephone anywhere on Earth. The technical goals of the system were very ambitious, as it attempted to make space communications viable by using Low-Earth Orbit (LEO) satellites. Over the course of a decade, the technical design of the system was completed, leading to over 1,000 patents. However, despite such incredible technical breakthroughs, Iridium resulted in a colossal commercial failure, going bankrupt after losing about $5 billion. The major cause behind such failure was the fact that ground-based competitors quickly captured the market for wireless telephony. These came to be in the time period between Iridium's conception (1980s) and its launch (in 1998). Iridium management team failed to hypothesize unfolding of events that did not align with their forecasts and (unrealistic) expectations, and this led to a system that was not capable of responding to changes in the operational environment. In fact, the system was not capable of downsizing or changing its operational goals. A snapshot of the events for the Iridium system is shown in Figure 1-1.



1980s
Need for global communications.
Lack of cellular infrastructure.

1990s
Terrestrial cellular development (GSM).
Increase in mobile communication demand.

2000s
Highly developed cellular networks.
Increased globalization.

Consumer demand shifts to lighter phones, cheap service, indoor use.

Higher data and connectivity demands

Iridium Satellite constellation concept developed in order to allow for anywhere communications.

High costs ($6B) committed to development.
Launched service in 1998 with heavy/expensive handset ($3K).
$3-8 per minute calls

Iridium bought for $25M.
Upgraded satellites and introduced new applications.

Technically driven mngmt.
Competitors in satellite based communications, as well as terrestrial cellular.

Lack of subscriptions due to better options/prices.
Bankruptcy and possible decommissioning of satellites.

Wider user base including world businesses, DoD, rescue, maritime, FAA, agriculture, construction.

Figure 1-1: Unfolding of events for the Iridium satellite system, using the Epoch shift-Impact-Response-Outcome framework (Beesemyer et al., 2012)

M1 Abrams tank. This tank was conceived and designed in the 1980s. At the time, because of the Cold War, the main location for any war engagement was expected to be central Europe. This led to the design of a system that was going to perform in moderate climate conditions and a specific type of terrain. When the system was then fielded in the Middle East years later, it could not perform optimally. Sand would infiltrate into the system and clog up mechanisms, leading to the failure of certain subsystems and parts much earlier then expected. In this case, the inability to anticipate such different environments undermined the overall performance of the tank.

Boeing B-2 Stratofortress. A (somewhat serendipitously) successful design is that of the B-2 aircraft. The design of this system was initially formalized in 1945, and the aircraft became operational ten years later. Through time, it has demonstrated great ability to manage the unfolding of uncertain events. In fact, the aircraft has participated in many missions for the U.S. Air Force, performing a variety of tasks: from air interception, to offensive counter-air, to maritime operations. The B-2 is a highly reconfigurable system, and, as such, it has managed to provide value in many different contexts with diverse missions and stakeholders. In fact, the aircraft has been capable of dispensing a variety of weapon systems (from gravity bombs, to cluster bombs, to guided missiles), as well as other payloads (e.g., UAVs). Part of the secret of its success is hidden behind its inherently simple design, characterized by ample margins for reconfigurability. As Saleh et al. (2003) point out, the U.S. Air Force has had many other aircraft that were much more complex (from a technical standpoint) than the B-2, and whose lifecycle has lasted only a fraction of the still operational B-2 (e.g., the B-58).

1.1.3 Need

The unfolding of uncertain events may result in major risks the system must be able to manage and respond to. However, such uncertain events may also have in stock great opportunities for generating extra value. In any case, it is important that designers think strategically about conceptualizing systems that are capable of responding to such uncertainty. Due to this dynamic complexity, it has become crucial that, early in the conceptual design phase, attention be devoted to system lifecycle properties, such as flexibility, robustness and affordability (Neches and Madni, 2012). It is in this light that these ilities[2] (i.e., system lifecycle properties) have become a very desirable trait in systems.

Designing for ilities is a strategic effort that involves not only technical knowledge, but also creativity. First, one must think about possible unfolding of events, and then about possible ways to design systems that are able to cruise this space of possibilities (and more) by resisting value loss and capturing extra value. So, in addition to identifying a meaningful uncertainty space, it is fundamental that system designers incorporate ways to enable a continual (or enhanced, when possible) value delivery in spite of the existence of uncertainty.

The main needs this thesis tries to address can be summarized as follows:

1. A formal way of modeling the key spaces of possibilities that characterize the effort of designing systems with desirable lifecycle properties
    a. A formal way of modeling and capturing information related to uncertainty characterizing the performance of the system over time

[2] It is important to clarify here that, for the purposes of this thesis, the use of the word "ility" (or "ilities") refers to these lifecycle properties that are related to the dynamic behavior of systems.

    b. A formal way of modeling and capturing information related to possible ways of responding to such uncertainty

2. A structured approach for instilling in systems the elements that drive the emergence of such lifecycle properties (i.e., ilities) over time
    a. A structured approach for exploring (and capturing) the uncertainty space related to a system
    b. A structured approach for generating, evaluating, and selecting ways to respond to the unfolding of the modeled uncertainty – thereby designing for ilities

1.2 Scope

The focus of the larger research project in which this thesis fits is that of delineating a method for architecting Systems of Systems with ilities. Chapter 3 gives a broad overview of this overarching method. The focus of this thesis is around two main steps in this method: that of brainstorming and modeling uncertainty, and that of identifying possible responses to it (which drive the emergence of ilities over time). In order to perform these two activities, this thesis discusses a formal way of modeling uncertainty and its responses based on a specific view on the design effort, the dynamic system perspective (Ross and Rhodes, 2008). From this perspective, and within the context of complex system design, the following research questions may be useful for the reader to keep in mind throughout the remainder of this thesis:

1. On the nature of uncertainty in system design
    a. How can uncertainty be modeled effectively?
    b. What can the dynamic impact of uncertainty be on the system?
    c. In what ways can the modeling of uncertainty be useful?
2. On the nature of responses to uncertainty in system design
    a. What drives the emergence of ilities throughout the lifecycle?
    b. How can one model these ility-driving elements?
    c. What can the dynamic impact of these ility-driving elements be in response to the unfolding of uncertainty?
    d. How have ility-related behaviors emerged in systems of the past (and the present)?
3. On the identification of ility-driving elements
    a. Can there be a structured approach for the generation, evaluation and selection of such elements?

1.2.1 MIT SEAri

The content of this thesis has been generated within the context of the larger research program of the MIT Systems Engineering Advancement Research Initiative (SEAri). SEAri is a research laboratory affiliated with the Engineering Systems Division (ESD) and the Department of Aeronautics and Astronautics. Since the former spans most departments within MIT (from those within the School of Engineering, to those within the School of Science, the School of Humanities, Arts, and Social Sciences, and the Sloan School of Management), SEAri finds itself in a unique position for interdisciplinary research in advancing systems engineering to meet contemporary challenges of complex sociotechnical systems.

SEAri seeks "to advance the theories, methods, and effective practice of systems engineering applied to complex sociotechnical systems through collaborative research." (SEAri, 2014) Research efforts within SEAri (this included) are predicated on the fact that the early phases of the conceptual design and development of a system require careful consideration. In fact, decisions made this early in the conceptual phase have a significant impact on the nature of the new system, and ultimately can enable or limit its success in time. In this perspective, and for the design of complex systems, it is useful to consider the trends of incurred and committed cost, management leverage, and knowledge over time. Blanchard and Fabrycky (2006) discuss the typical trends of these terms in the classic design paradigm (left side of Figure 1-2). Research in SEAri is aimed at influencing these trends, extending managerial leverage and delaying committed costs further into development, while increasing knowledge gained earlier in design.

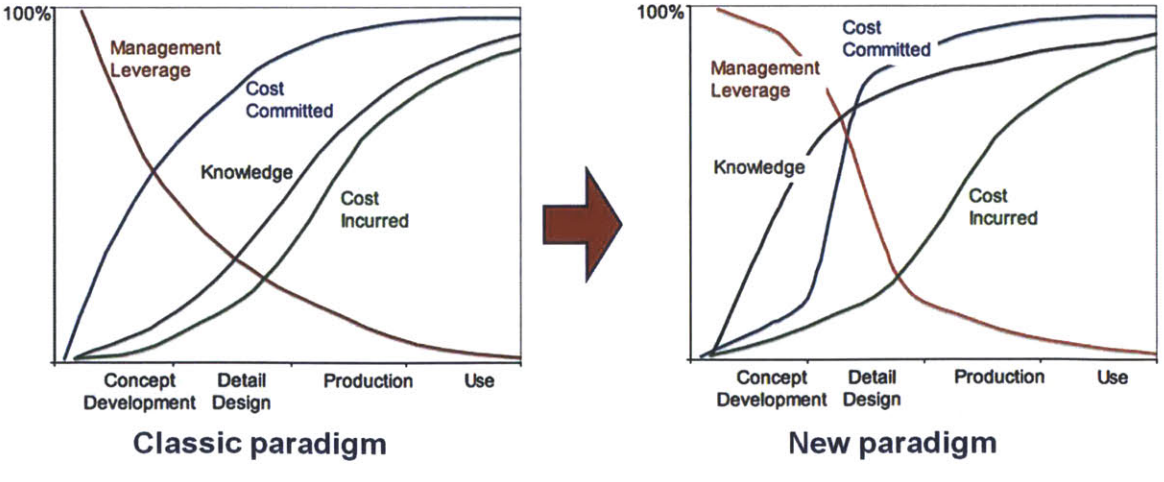


Figure 1-2: Desired Shift in Critical Front-End Complex System Design.

1.3 Methodology

Most research in SEAri, including the one presented in this thesis, is strategically placed in both theory-based and practice-based methods (Figure 1-3). While prescriptive methods seek to advance the state of the practice using normative principles, they must also be informed by practical limitations and constraints that come from descriptive research (Ross and Rhodes, 2008). This research focuses primarily on advancing a theory-based framework for the description of uncertainty and ility-driving elements (IDEs – formally described in chapter 5) within the context of system design. This work lays the foundations for a more prescriptive research thrust, wherein a structured approach for identifying ility-driving elements (appropriate for a given uncertainty space) is developed. Lastly, the research also features a descriptive thread, with the empirical investigation of a particular type of ility-driving element (change options) in military systems (as well as other fields).

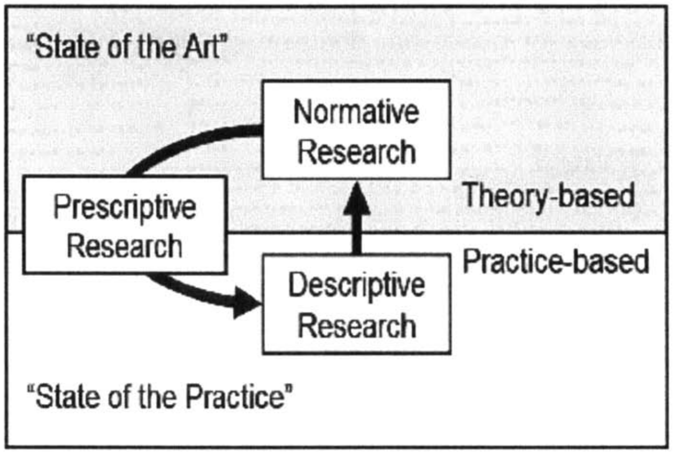


Figure 1-3: Underlying Structure of SEAri Research Program (Ross and Rhodes 2008).

1.3.1 Theory-based Research Thrust

Much research has been performed in the systems engineering community on what it means to design in the face of changing operational environments and needs, which has led to a large literature of definitions and constructs (building on several descriptive research efforts), as well as an array of methods for facilitating and improving system design in such circumstances. Part of this research aims at delineating a formal (set-theoretic) framework for modeling and defining some key concepts that have emerged over time (e.g., perturbations, change options), as well as new ones (e.g., resistance property). In such a formal framework, uncertainty and ility behaviors are characterized through perturbations and ility-driving elements.

1.3.2 Descriptive Research Thrust

Part of this thesis presents an empirical investigation on how and when change options (one type of ility-driving element, as discussed herein) have been instilled and used in systems of the past (that are still operational). In particular, change options related to the ability of some military System of Systems to evolve over time are presented. These are then linked to some generalized architect's intents for the design of such complex systems in the face of continual change. Lastly, a brief discussion is provided on planned adaptation in policymaking, and how well they this approach fits within the framework laid out in this thesis.

1.3.3 Prescriptive Research Thrust

Given the formal framework delineated in the theory-based research effort, a structured approach for the identification of design elements that can drive the emergence of ilities in systems is proposed. This approach proposes different ways of generating, evaluating and selecting such ility-driving elements. The approach is highly scalable in time and effort, and a (or a set of) stakeholder(s) can choose to perform it in the most appropriate way for his or her decision problem, and given limitations in time and knowledge.

1.4 Thesis Overview

This chapter has introduced the basic motivation behind this research effort, as well as the essential elements that constitute it. A brief description of what is discussed in the remainder of the thesis follows below.

Chapter 2: Literature Overview. This chapter provides an overview of the relevant literature with regard to the nature of complex sociotechnical systems, the nature of uncertainty in such systems, and how people have dealt with such uncertainty in various disciplines. The chapter ends with a brief discussion on Systems of Systems (SoS) and the Maritime Security SoS that is used throughout the thesis as a case study.

Chapter 3: A Method for Architecting Systems of Systems with Ilities. This chapter presents a broad overview of the SoS Architecting with Ilities (SAI) method and serves as a frame for the remainder of the thesis. Several activities involved in the architecting of Systems of Systems with an emphasis on enhancing lifecycle value sustainment from the early phases of design are discussed. The remainder of the thesis is centered around two of the steps in SAI: chapter 4 explicitly addresses the modelling and characterization of uncertainty (related to step 3 in the method); chapter 5, 6 and 7 discuss the nature, usage, and identification of change options and resistance properties in system design (related to step 5 in the method).

Chapter 4: Formal Modeling of Uncertainty via Perturbations. This chapter provides a more in depth investigation of the concepts and activities involved in Step 3 of the SAI method. It is concerned with the conceptualization and modeling of uncertainty. It provides a formal way of modeling uncertainty in system design under the dynamic system perspective. Furthermore, it discusses a variety of descriptive fields for the parameterized uncertainty. Lastly, it presents an application of some of the activities in Step 3 in SAI directed at the elicitation and parameterization of uncertainty to the Maritime Security SoS.

Chapter 5: Formal Modeling of Responses to Uncertainty: Ility-Driving Elements. This chapter provides an in depth description of the possible ways in which designers can cope with uncertainty. It introduces and models the concepts of change option and resistance property in a formal way. Furthermore, the chapter discusses the ways in which these two ility-driving elements can be used in the face of unfolding uncertainty. Lastly, several descriptive fields for the characterization of the two types of ility-driving elements are introduced.

Chapter 6: An Empirical Investigation of Change Options. This chapter presents the results of an empirical investigation of change options. First, it presents some of the results from an investigation on evolvability in military SoS. For a given evolution increment, the change options that enabled it were investigated. Then, the idea of planned adaptation in the policymaking realm is also discussed in relation to the concept of a change option. The link between the two is explained through the lenses of some policy cases (the most prominent of them being the regulation of particulate matter).

Chapter 7: Ility-Driving Element Analysis. This chapter builds on the foundations laid out in chapter 4 and 5 for describing uncertainty and ility-driving elements in the system design effort. It introduces Ility-Driving Element Analysis (IDE Analysis), a structured approach for the generation, evaluation and selection of ility-driving elements during the conceptual phase of system design. Using this approach, it is possible to formally think about the introduction of design- and enterprise-level elements that can drive the emergence of ilities in complex systems.

Chapter 8: Discussion. This chapter presents general discussions around three main topics: applicability of research; general considerations with regard to the main research contributions; and possible areas for future research.

Chapter 9: Conclusion. This chapter presents some general conclusions, as well as a brief recapitulation of the key contributions of the thesis.

2 Literature Overview

"Hegel was right when he said that we learn from history that man can never learn anything from history."

– George B. Shaw

This chapter presents a review of the most relevant literature with regard to (1) the nature of complex engineered systems and the need for ilities; (2) the conceptualization of uncertainty around complex systems; (3) addressing uncertainty in the design of complex systems through the use of (real) options. The chapter closes with a short discussion of Systems of Systems (SoS) – to which most examples in this thesis relate – and a description of the Maritime Security (MarSec) SoS, used as a case study throughout the thesis.

2.1 Basic Motivation

Due to the complex and highly dynamic operational environments in which engineering systems operate nowadays, it has become crucial that, early in the conceptual design phase, attention be devoted to system lifecycle properties, such as flexibility, robustness, etc. (Neches and Madni, 2012). That is, it is important to identify key uncertainty categories and associated stimuli that could disrupt system value delivery, or that could introduce opportunities for delivering more value (de Weck, Eckert, and Clarkson, 2007). As pointed out in chapter 1, in addition to identifying a meaningful uncertainty space, it is fundamental, then, that systems engineers incorporate in the design of their systems ways to enable a continual delivery of value (or enhancement thereof) in spite of the existence of such uncertainty.

2.2 Complex Artificial Systems

The application domain for this research is the design of engineered systems, aimed at providing value to a (set of) stakeholder. Such systems can span from the very technical to the more sociotechnical (e.g., an aircraft and a mass transportation system, respectively). The complexity of such systems has been growing over time, not only due to scale and interconnectedness, but also due to increased scope in our ability to describe the systems themselves and their continual evolution over time.

2.2.1 On the Design of Complex Engineered Systems

Simon (1996) explains that, "... everyone designs who devises courses of action aimed at changing existing situations into preferred ones." This idea is pervasive in all professions, and very much so in the design of complex engineered systems, where a system is aimed at improving the current situation of a stakeholder (e.g., the government, the scientific community, society as a whole). An artificial system (i.e., man-made, different from natural systems like the human body, or a cell) constantly interacts with the external environment. Simon (1996) discusses how there are three principal components in the design of a system: (1) the system itself, whose design is controlled (the "inner environment"); (2) the environment in which the system operates, which is exogenous to the control of designers (the "outer environment"); and (3) what is desired of the system, i.e.: a set of objectives. He goes on to state that the good designer is able to "insulate the inner system from the environment, so that an invariant relation is maintained between inner system and goal." This is often a hard task when designing complex engineered systems (a special kind of artificial system), most of which are inherently open. The openness of such systems makes it so that even the goals of the system may change over time, upon the unfolding of uncertain events. This is related to the so-called "wicked problem," first described by Horst Rittel in the 1960s. Churchman (1967)[3] provides the first published report of Rittel's idea, in which wicked problems are described as a "class of social system problems which are ill-formulated, where the information is confusing, where there are many clients and decision makers with conflicting values, and where the ramifications in the whole system are thoroughly confusing." Buchanan (1992) points out how this description evinces a "fundamental indeterminacy in all but the most trivial design problems." Rittel (1969) elucidates ten key properties of wicked problems (e.g., among others: no definitive formulation; no stopping rule; solutions are not true-false, but good-bad; no possibility of testing solution; every solution is a "one-shot operation").

In designing complex engineered systems of the future, models have become essential tools. Analysts and engineers are often forced on using models and simulations in order "to explore...system performance without actually producing and testing each candidate system" (Blanchard and Fabrycky, 2006). The nature of the data produced by these models is inherently *artificial* – i.e., it is not obtained by direct measurement of system properties, as no identical (or even similar) system may yet exist. Hence, the artificial data (as well as the model) "cannot be classified as accurate or inaccurate in any absolute sense" (Blanchard and Fabrycky, 2006).

In addition to models of system performance, there are value models, which attempt to capture stakeholder preferences on performance (i.e., their values). The need for

[3] CW Churchman and HA Simon are often seen as the founding fathers of Management Science, albeit with somewhat different views of it. Ulrich (1980) provides a (very insightful) fictitious debate between these two illustrious figures.

constructed value models arises from the fact that the many dimensions of value (and their interactions) are often beyond the capability of a stakeholder's mental model, and that stakeholders are often interested in comparing many alternatives. Value models are more intricate than performance models: while assessing performance can have a certain degree of objectivity and measurability, stakeholder satisfaction is inherently subjective and difficult to measure. Fischoff (1991) discusses a spectrum of philosophies with regard to the nature (and elicitation) of values. On the one end of the spectrum is the philosophy of articulated values, which assumes that values are self-evident in people's choices (e.g., empirically gathered willingness-to-pay measurements, i.e.: demand curves in microeconomics). On the other end of the spectrum is the philosophy of basic values, which assumes that a core set of values exist, from which it is possible to derive value models through an inferential process (e.g., interviews). This assumption pervades classical decision theory (Keeney, 1996), and it also underlies the remainder of this thesis (in which the problem of sustainment of value delivery over time is addressed). Examples of constructed value models are customer value models (i.e., a representation of the worth – in monetary terms – of what a company does for its customers), or multi-attribute utility functions from classical decision theory (Keeney and Raiffa, 1976). The value model's reliability is critical to success in system design; for as Hall (1989) points out, "to design the wrong value system is to design the wrong system."

Lastly, it is important to mention that in many large-scale engineered systems, there exists a pronounced social component (these systems are often referred to as "sociotechnical systems"). This adds another layer of complexity in the design of such systems because, unlike the natural sciences for which there are universal laws[4], in the case of the social sciences it is only possible to construct models by means of "situational analysis" (Popper, 1967), (i.e., models of typical social situations). These models can't be "animated" from the universality of natural principles, but have to compromise for the assumption of a different principle: the rationality principle[5]. This principle has been proven false under many circumstances, e.g.: Tversky and Kahneman (1986). Popper (1967, pp. 345) defines it an "almost empty principle," but also admits its inescapable nature:

> I hold, however, that it is good policy, a good methodological device, to refrain from blaming the rationality principle for the breakdown of our theory: we learn more if we blame our situational model.

2.2.2 Five Aspects of Complexity

As discussed in the subsection above, Simon (1969) points out the involvement of three terms in the design of an artifact (or system): "the purpose or goal, the character of the artifact, and the environment in which the artifact performs." These key terms each

[4] Universal only within the current scientific paradigm (Kuhn, 1962).

[5] Which Popper (1967) defines as "the principle of acting appropriately to the situation," where a situation can be summarized by what is known and what is aimed for.

induce different types of complexity, which make engineering complex systems an arduous endeavor. First of all, the system (artifact) is complex in its *structural* form, as well as in the way it *operates*. Furthermore, there is complexity embedded in the interaction of the system with its outer *environment* over *time*. Lastly, it is complex to decipher and quantify the way in which stakeholders *perceive* value delivered by the system (over time).

Rhodes and Ross (2010) describe in depth these types of complexity with their five-aspect framework for the engineering of complex systems. In their work, they acknowledge that modern day systems have extraordinary levels of complexity and uncertainty. In fact, these systems exist in a very dynamic world, whose pace of change continues to accelerate. They state that, "while the structural and behavioral aspects [of complexity] remain at the core of the systems engineering method, there is an urgent need to more effectively address three additional aspects: contextual, temporal and perceptual." The five aspects of complexity in Rhodes and Ross (2010) are described here:

- Structural: related to the form of system components and their interrelationships. For instance: heterogeneous components and constituent systems; elaborate networks; loose and tight couplings; layers; vertical or horizontal structures; multiplicity of scales.
- Behavioral: related to performance, operations, and reactions to stimuli. For instance: variance in response to stimuli; unpredictable behavior of technological connections; emergent social network behavior.
- Contextual: related to circumstances in which the system exists. For instance: many complexities and uncertainties in system context; political, and economic variations; market factors; stakeholder needs profile and overall worldview.
- Temporal: related to dimensions and properties of systems over time. For instance: decoupled acquisition phases; context shifts; systems with long lifespan and changing characteristics; time-based system properties (flexibility, survivability, evolvability, etc.)
- Perceptual: related to stakeholder preferences, perceptions, and cognitive biases. For instance: many stakeholder preferences to consider; perception of value changes with context shifts; cognitive biases[6].

This research is not concerned with (static) structural and behavioral complexity of a system. Rather, it focuses on addressing concepts around the remaining three types of complexity. Particular attention is devoted to the temporal aspect of complexity: i.e., complexity in the interaction of the system with its *outer environment* and it *purpose or*

[6] Cognitive biases are omnipresent in people's actions. For an in-depth (much recommended) read on such biases (e.g., availability, contaminations, anchoring, etc.), refer to the collection of articles in Gilovich, Griffin, and Kahneman (2002).

goal over *time*. This aspect of complexity is the fundamental motivation behind the need for ilities.

2.3 Uncertainty in the Design of Complex Engineered Systems

The design of complex engineered systems is unavoidably plagued with irreducible uncertainty. Part of this thesis focuses on a way to model the uncertainty space (chapter 4) – i.e., contextual, temporal, and perceptual complexity, as well as how to respond to it (chapters 5 and 7) – using generalized ility-driving elements.

Uncertainty is ubiquitous, especially in highly dynamic environments. When considering complex systems, it can take a variety of different shapes and impacts. In general, uncertainty can stem from endogenous and exogenous sources, where the latter are usually related to context and expectation changes. The following subsections cover some salient aspects regarding uncertainty.

2.3.1 On the Nature and Characterization of Uncertainty in Design

This subsection presents literature on the nature, classification and identification of uncertainty from the fields of system design and policymaking. It discusses how prediction (and its uncertainty) in science is different from that in the field of design; it then presents some relevant taxonomies for classifying uncertainty that can be used in the process of modeling uncertainty in design (see chapter 4), as well identifying relevant sources of uncertainty.

Simon (1996) discusses at length the notion of design, describing it as "devising courses of action aimed at changing existing situations into preferred ones." However, since the consequences of design lie in the future, "it would seem that forecasting is an unavoidable part of every design process." Hence, when devising courses of action for future states of the world, one is inevitably faced by uncertainty. Pielke (2001) discusses the important difference between prediction in science and prediction in decision making, where the latter is "forward looking," formulating alternative courses of action extending into the future, and selecting among alternatives by expectations of how things will turn out. He then defines uncertainty as the state in which more than one outcome is consistent with reasonable expectations. He goes on to state (pp. 116):

> Expectations are a result of judgment, are sometimes based on technical mistakes and interpretive errors, and are shaped by values and interests. As such, uncertainty is not some feature of the natural world waiting to be revealed but is instead a fundamental characteristic of how human perceptions and understandings shape expectations.

Furthermore, Pielke (2001) discusses the difference between aleatory and epistemic uncertainty. The former is associated with random processes, and can be studies using statistics. This kind of uncertainty, by definition, can be known but can't be reduced. The latter is associated with incomplete knowledge of a phenomenon, as well as of the limits of one's knowledge (Hoffman and Hammonds, 1994; Stewart, 2000). Decisions made in the conceptual phases of engineering complex systems are often plagued with epistemic uncertainty, which is inevitably carried into the models used for prediction.

van Asselt and Rotmans (2002) propose a taxonomy for different sources of uncertainty (Figure 2-1) in the context of Integrated Assessment modeling for decision makers (in the realm of complex policy issues). At the highest level, they assert that two major sources of uncertainty can be distinguished:

- *Variability.* The system/process under consideration can behave in different ways or is valued differently. Variability is an ontological attribute; it includes (but is not limited to) aleatory uncertainty.
- *Limited knowledge.* Limited knowledge is a property of the analysts performing the study and/or of our state of knowledge (epistemological).

These two types of uncertainty are referred to differently across the literature, depending on the academic field. Variability is also referred to as: "objective uncertainty" (Natke and Ben-Haim, 1996), "stochastic uncertainty" (Helton, 1994), "primary uncertainty" (Koopmans, 1957), "external uncertainty" (Kahneman and Tversky, 1982) or "random uncertainty" (Henrion and Fischoff, 1986). Limited knowledge is also referred to as: "subjective uncertainty" (Helton, 1994; Natke and Ben-Haim, 1996), "incompleteness of the information" (von Schomberg, 1993), "informative uncertainty" (Klir, 1996; Natke and Ben-Haim, 1996; van Witteloostuijn, 1987), "secondary uncertainty" (Koopmans, 1957) or "internal uncertainty" (Kahneman and Tversky, 1982).

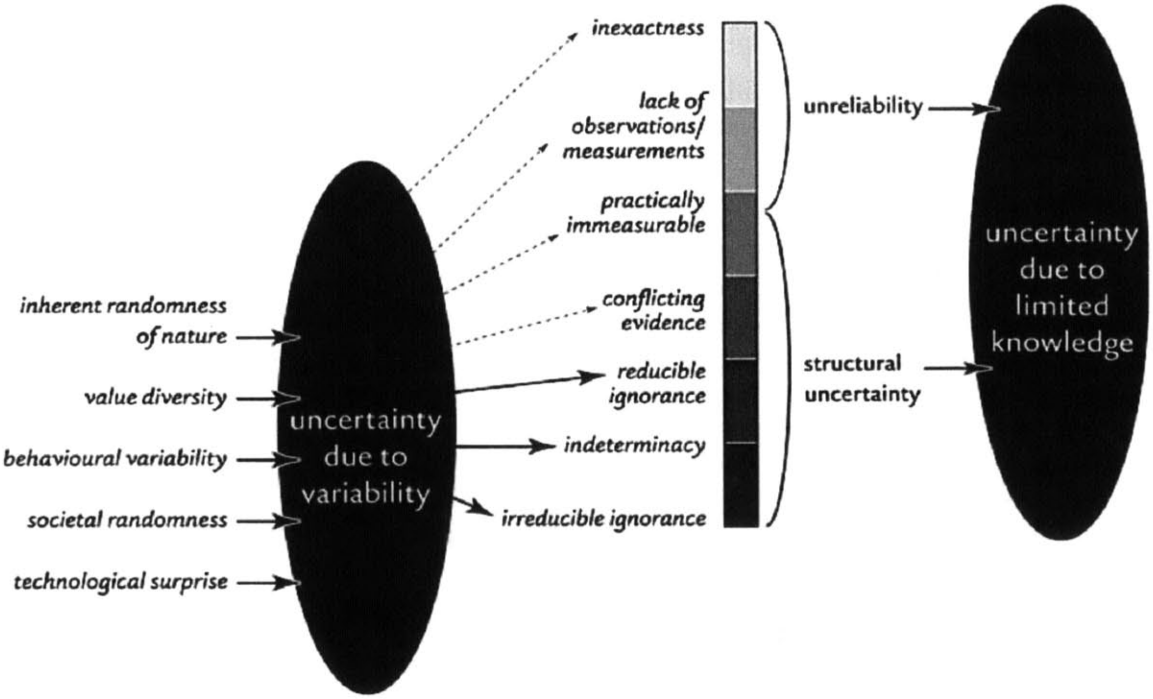


Figure 2-1: Typology of sources of uncertainty (van Asselt and Rotmans, 2002).

van Asselt and Rotmans (2002) further distinguish different sources of variability (providing some examples related to the climate change policy issue):

- Inherent randomness of nature. The non-linear, chaotic and unpredictable behavior of natural processes (e.g., the behavior of clouds).
- Value diversity. Differences in people's (stakeholders') mental maps, worldviews, moral norms and values, due to which problem perceptions and definitions differ. This is also referred to as moral uncertainty (de Marchi, 1995) (e.g., discounting rate in cost-benefit analysis for environmental impacts of CO_2 emissions).
- Human behavior (behavioral variability). Non-rational behavior, discrepancies between what people say and what they actually do (cognitive dissonance) (e.g., consumption patterns)
- Social, economic and cultural dynamics (societal variability). The non-linear, chaotic and unpredictable nature of societal processes (macro-level behavior) (e.g., infrastructural changes in energy supply).
- Technological surprises. New developments or breakthroughs in technology or unexpected consequences ('side-effects') of technologies (e.g., renewable energy options).

van Asselt and Rotmans (2002) further discuss how limited knowledge partly results from variability, but knowledge with regard to deterministic processes can also be incomplete and uncertain. They introduce a continuum of sources of limited knowledge:

- Inexactness. This source of uncertainty is concerned with the fact that it is only possible to "roughly" know. There is always going to be lack of precision, inaccuracy, and measurement errors (e.g., life-times of greenhouse gases).
- Lack of observations/measurements. Lacking data that could have been collected, but haven't been (e.g., temperature feedbacks).
- Practically immeasurable. Lacking data that in principle can be measured, but not in practice (too expensive, too lengthy, infeasible experiments).
- Conflicting evidence (Zimmermann, 1996). Different data sets/observations are available, but allow room for competing interpretations.
- Reducible ignorance (Funtowicz and Ravetz, 1990; Wynne, 1992). Processes that are not observed, nor theoretically imagined at a given point in time, but may be in the future.
- Indeterminacy (Wynne, 1992). Processes for which it is possible to understand the principles and laws, but that can never be fully predicted or determined (e.g., weather dynamics).

- Irreducible ignorance. There may be processes and interactions between processes that cannot be (or not unambiguously) determined by human capacities and capabilities.

These uncertainties discussed in the context of decision making for complex policy issues are relevant also in the design of complex systems. McManus and Hastings (2006) propose a framework to aid in the understanding of uncertainties and techniques for mitigating and even taking advantage of them. In their work, they attempt to clarify the wide variety of uncertainties that can affect complex systems. They describe two major classes of uncertainty: lack of knowledge and lack of definition. The former concerns "facts that are not known, or are only known imprecisely, that are needed to complete the system architecture in a rational way." The latter concerns "things about the system in question that have not been decided or specified." They further specify that, for both classes of uncertainty, there can be different flavors, of which they give three points in a continuum:

- Statistically characterized (random) variables/phenomena: Things that can't always be known precisely, but which can be statistically characterized, or at least bounded (e.g., weather conditions). This type of uncertainty can be handled by analytical techniques (e.g., risk analysis).
- Known unknowns: Things that it is known are not known (e.g., performance of new technologies, future political adversaries, or future budgets). These are most often handled qualitatively, and at best semi-analytically.
- Unknown unknowns: "Gotchas." By definition, these can 't be known.

A remarkable literature survey of uncertainty definitions and classifications from various fields was performed by Thunnissen (2003). First he provides a summary of classifications and definitions of uncertainty in social sciences, physical sciences, and engineering. Then, he discusses how none of the uncertainty classifications described for these fields are directly applicable to the design and development of complex systems. In an attempt to remedy this shortcoming in the literature, he proposes a new classification of uncertainty for the design and development of complex systems, shown in Figure 2-2.

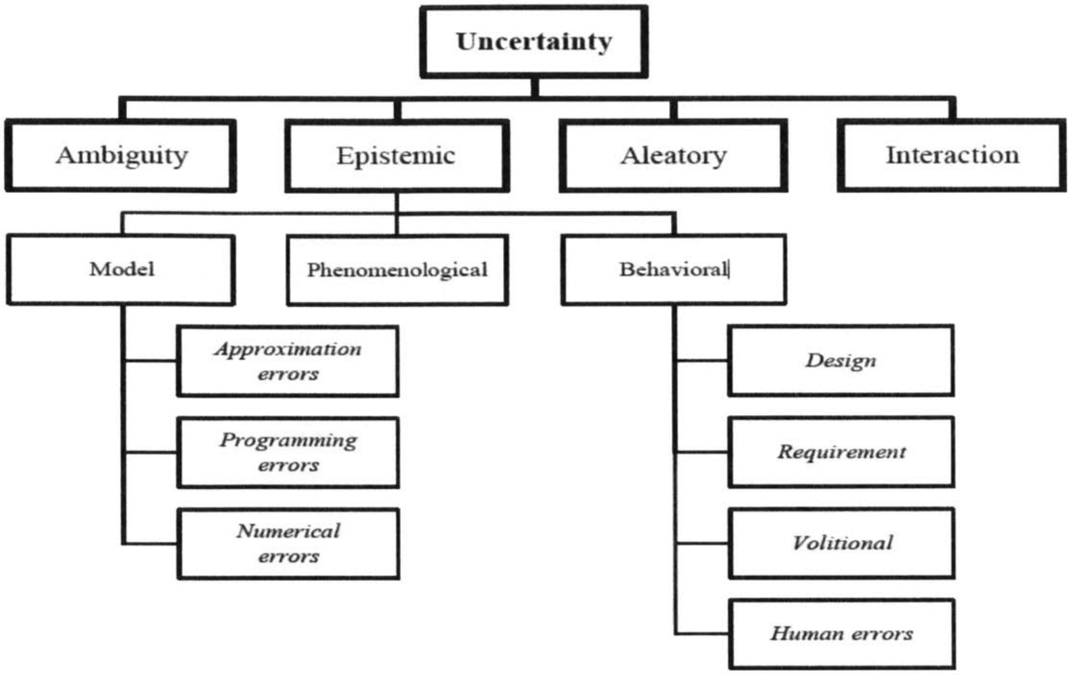


Figure 2-2: Uncertainty Classification for the Design and Development of Complex Systems (Thunnissen, 2003)

Within the context of engineering design, de Weck, Eckert, and Clarkson (2007) reduce the relevance of uncertainty to the following two questions:

1. Will the product, system or artifact that is being designed meet its functional and form requirements once it is on sale or in use? Will it function properly and perform adequately?
2. Are the functional and form requirements the right ones that will lead to market success?

Then, they also examine the generic classification of uncertainty. Similar to McManus and Hastings (2006), their sources of uncertainty range from known to unknown in a continuum. Furthermore, they take a system-centric view and, after making the distinction between endogenous and exogenous uncertainties, they describe specific sources of uncertainty in engineering design (e.g., technology, suppliers, contractual arrangements, etc.). Figure 2-3 illustrates the sources of uncertainty de Weck, Eckert, and Clarkson (2007) identify in their work. Such diagram can also be very helpful in terms of recognizing uncertainty – categories and instantiations thereof – that can impact the ability of the system to meet its functional and form requirements over time. A similar approach – the enterprise boundary analysis – is described in Ross et al. (2008) within the context of the Responsive Systems Comparison method. This approach is discussed in further detail in chapter 3.

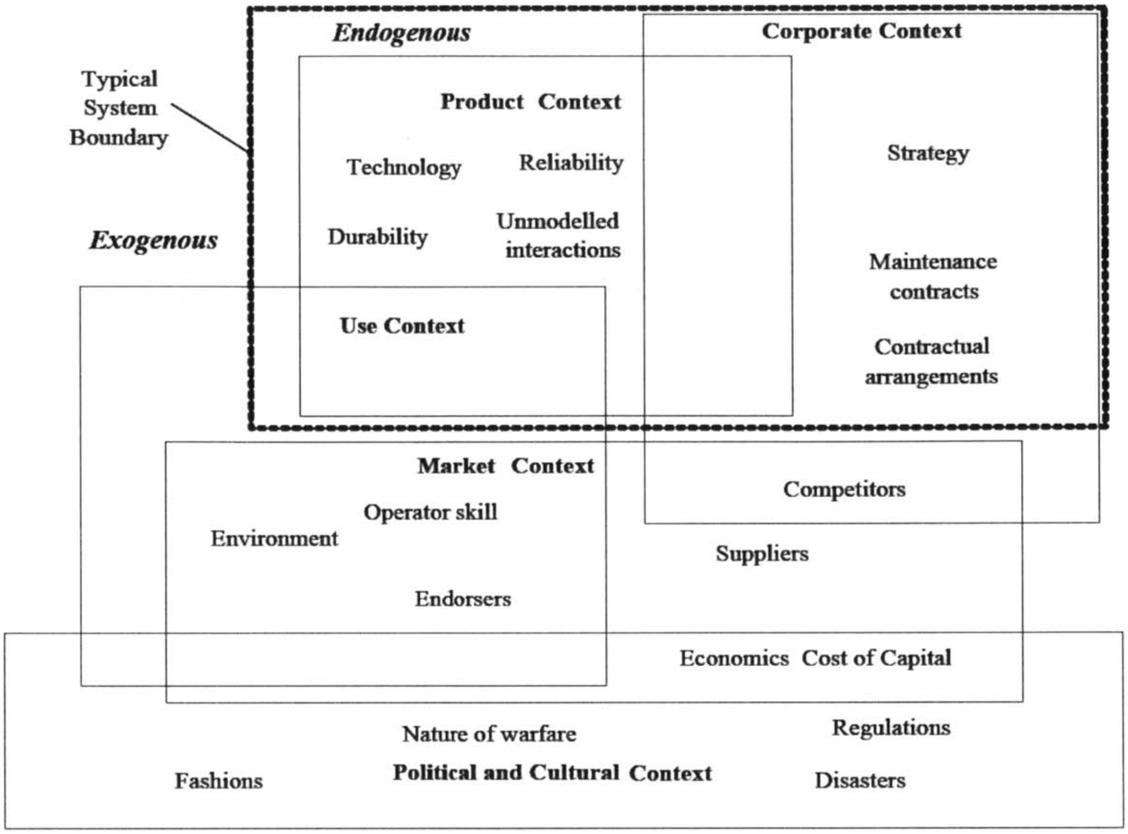


Figure 2-3: Sources of uncertainty (de Weck, Eckert, and Clarkson, 2007).

2.3.2 On the Modeling of Uncertainty in Design

In this subsection, a literature review on the effort of modeling uncertainty – and the implications therein – is discussed. Formal and practical approaches for modeling uncertainty are discussed, as well as the implications of using models that are intended for the explorations of possible future states of the world. These considerations play a key role in the practical uncertainty modeling effort discussed in chapter 4.

There are many different approaches for modeling uncertainty in the literature. de Weck, Eckert, and Clarkson (2007) distinguish between formal approaches and practical approaches. The former are theories grounded in logic and epistemology, and which require a numeric expression of the likelihood of future events. Two examples of such theories are Probability Theory and Possibility Theory. Probability Theory is the traditional setting for representing uncertainty in Artificial Intelligence. It is the so-called Bayesian approach, first introduced in Bayes (1763). In this setting, a cognitive state is represented by means of a single probability measure (p) on the set of possible worlds. In Possibility Theory, a cognitive state of the world is modeled by a possibility distribution instead. As such, Possibility Theory can be thought of as an imprecise Probability Theory (or a complement to it), devoted to the handling of incomplete information. It differs from the latter because of the use of a pair of dual set-functions – possibility and necessity – instead of only one (probability). Possibility Theory was first introduced in Zadeh (1977) as an extension of his theory of fuzzy sets and fuzzy logic.[7]

More practical approaches to uncertainty modeling can be distinguished into two main categories (de Weck, Eckert, and Clarkson, 2007): continuous variables and discrete events (or scenarios). The former represent uncertainty as random variables. This approach is particularly useful when representing uncertain demand or supply. Within the context of system design, among the most common methods in this category are continuous diffusion models (de Weck, de Neufville, and Chaize, 2004) and discrete lattice models (de Neufville et al., 2004). Discrete methods like lattice models rely on statistical sampling methods, such as Monte Carlo simulations (Rader, Ross and Rhodes, 2010).

The latter category (discrete events and scenarios) is used for representing uncertainty that is more discrete in nature. It involves "either estimating the likelihood, time of occurrence, and magnitude of certain discrete events (e.g. earthquakes, hurricanes, etc.), or representing the future as a set of more or less well defined scenarios." (de Weck, Eckert, and Clarkson, 2007) Examples of methods within this category are scenario planning (Schoemaker, 1993) and Epoch-Era Analysis (EEA) (Ross and Rhodes, 2008). Such techniques basically consist of defining a finite set of possible futures ("scenarios" for scenario planning and "epochs" for EEA) that, collectively, capture a set of possible future circumstances that are deemed relevant to decision makers at a given point in time (e.g., design). Epoch-Era Analysis is discussed in further detail in chapter 4.

It is important to recognize that, although there exist closed systems for which uncertainty can be directly quantified (e.g., error analysis in engineering, probabilities in blackjack, actuarial science for insurance), most complex engineered systems are inherently open, and it is very difficult to characterize uncertainty in these instances. For these systems, it is arduous (often impossible) to predict the future environment they are going to operate in, as well as any new need that may emerge. In these cases, the most one can do is to explore the behavior of the system under a variety of different (probable) circumstances (Bankes, 1993). Particularly useful in doing so are scenario-based models of uncertainty. In fact, when used to explore possible contingencies, models can help mitigate (not solve!) the problem of coping with uncertainty of prediction during the decision making process. Bankes (1993) describes such models as "prostheses for the intellect." Exploratory models – such as scenario planning and EEA – can be beneficial in many ways (Pielke, 2001, 115):

> First, they can shed light on the existence of unexpected properties associated with the interaction of basic assumptions and processes (e.g., complexity or surprises). Second, in cases where explanatory knowledge is lacking, exploratory models can facilitate hypothesis generation to stimulate further investigation. Third,

[7] For an in-depth comparison of the two theories, refer to Dubois and Prade (1994).

> the model can be used to identify limiting, worst-case, or special scenarios under various assumptions and uncertainty associated with the model experiment.

Abbass et al. (2008) reinforce this idea in their discussion of computational scenario-based capability planning (defense planning). They state that scenarios represent different frames for plausible futures, and that the aim of these frames is "to focus the planner's mind on establishing future contexts and help make a case for the development and justification of strategies. They also assist the planner in defining intermediate goals along the path towards the future." Abbass et al. (2008) highlight how common to all scenarios is that they deal with uncertainties and illustrate major issues a planner has to deal with. Quoting Schoemaker (1993), they assert that the elements in a scenario can be thought of as uncertainty-bounding. And thus, "one can think of scenarios as the edges that bound a multidimensional sub-space of uncertainty." They point out that there are many critics of the scenario method, who rightfully claim that the future is inherently unpredictable and that using scenarios to anticipate futures is an illusion. However, they also admit that entering such philosophical debate would be useless from the pragmatic standpoint of someone that has to make a decision without falling into paralysis. Similarly, in response to social constructivist invectives (Jasanoff and Wayne, 1998) claiming that modeling – by definition – denies the essence of uncertainty, van Asselt and Rotmans (2002) argue that downplaying the usefulness of models as structuring devices corresponds to "throwing away the baby with the bath water."

Lastly, Pielke (2001) points out how conventional wisdom mistakenly holds that uncertainty is best understood or reduced by advancing knowledge. In fact advances in knowledge can add significant uncertainty. In the end, "one of the most critical issues in using models to develop information for decision making is to understand uncertainty, its sources, and its potential reducibility." As Weber (1999, pp.43) observes:

> If uncertainty is measurable and controllable, then forecasting and information management systems serve a high value in reducing uncertainty and in producing a stable environment for organizations. If uncertainty is not measurable and controllable, then forecasting and predictions have limited value and need to be understood in such context. In short, how we view and understand uncertainty will determine how we make decisions.

2.4 System Lifecycle Properties: Ilities

The unfolding of uncertain events (related to temporal complexity) makes the design effort much more difficult. Unfortunately, the degree to which a system meets static functional requirements does not describe much insofar as how well the system can cope with the unfolding of uncertainty. Hence, non-traditional design criteria (e.g., flexibility, survivability or robustness) – i.e., ilities – that describe the aptitude of a system in coping with changing contexts and needs (or forced changes in the design itself) have become a fundamental interest in the systems engineering community. McManus et al.

(2007) point out how ilities are system properties that are increasingly recognized as qualities that lead to successful systems.

Over the years, specific ilities are defined differently depending on the field they are conceived in (e.g., computer science, systems engineering, etc). Furthermore, the language used in the description of ilities is characterized by poorly defined terminology (McManus and Hastings, 2006). This research is generally concerned with the effort of engineering complex systems, and the meta-definition of an ility that underlies the remainder of the thesis is provided in de Weck, Ross and Rhodes (2012, pp. 1):

> The ilities are properties of engineering systems that often manifest and determine value after a system is put into initial use. Rather than being primary functional requirements, these properties concern wider impacts with respect to time and stakeholders.

Ilities, as defined above, are most related to the part of the conceptual-phase engineering effort that deals with strategic level thinking. The aforementioned contextual, perceptual and temporal aspects of complexity characterizing modern-day systems engineering put pressure on system engineers to think strategically about systems' lifecycle properties. In light of these considerations, Rhodes and Hastings (2004) point out how the practice of systems engineering must evolve to meet the requirements of increasingly dynamic operating environments.

McManus and Hastings (2006) describe how ility behaviors are a direct outcome of how well a system manages to respond to uncertainty. They introduce a framework (Figure 2-4) that relates the problem of uncertainty to the desired ility behavior. In this framework, it is evident how the numerous types of uncertainty (further discussed in the next section) can cause an array of consequences – both positive and negative (opportunities and risks, respectively). These consequences can be either mitigated or exploited through certain decisions or actions like introducing in the design of the system instantiations of design principles such as margin, redundancy, modularity, standardization, or scalability. These decisions enable the emergence of ilities over time. However, in most cases these decisions come at a cost. This cost is often hard to justify, given the fact that stakeholders are unable to draw any immediate benefit from it (Ross, Rhodes, and Hastings, 2008).

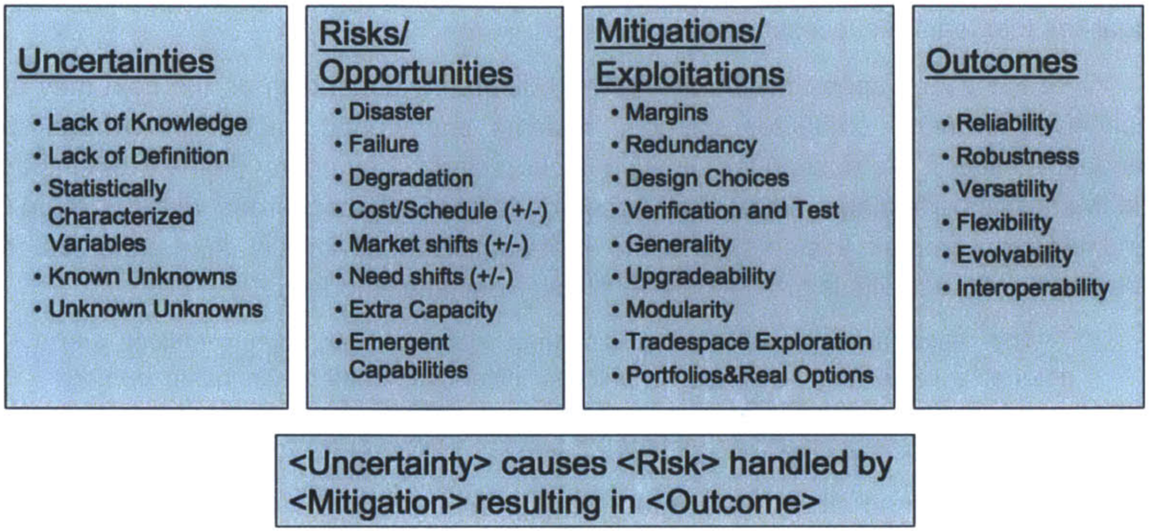


Figure 2-4: Framework that relates the existence of uncertainty to desired ility outcomes (McManus and Hastings, 2006)

Beesemyer (2012) discusses how at the core of the need of any type of ility, there is a key desire of maintaining system value delivery over time. He terms this all-encompassing goal *value sustainment*. Value sustainment, then, encapsulates the idea that the interest of a stakeholder is that systems *keep* providing value throughout their lifetime, in spite of the occurrence of value-disruptive uncertain events (i.e., perturbations, which are discussed in depth in chapter 4). This can be achieved in two different ways: either by (1) passively resisting changes (that ultimately cause reductions in value), or by (2) actively changing (to increase value delivery – perhaps after the occurrence of a negative perturbation – or reduce chances of value reduction). Example ilities related to (1) are robustness and survivability; example ilities related to (2) are flexibility and evolvability.

In the literature, different scholars and practitioners have defined the same ility differently. A vibrant example of this is flexibility, for which several different definitions have been provided over time (Fricke and Schulz, 2005; McManus and Hastings, 2006; Butterfield et al., 2008; Viscito and Ross, 2009). In an attempt to form a coherent semantic framework for defining ilities (as well as tracing them into verifiable system requirements), Ross et al. (2011) introduced a semantic basis for ilities. This is a structured approach for exploring the existence of one or more semantic fields, thereby enabling the classification of ilities over multiple categorical dimensions. Examples of such semantic fields introduced are: *agent* (internal, external or none), *parameter* (level or set), and *perturbation* (disturbance, shift, or none). For example, the definitions of flexibility and adaptability change only in the executive: external to the system for the former, and internal for the latter. Table 2-1 lists the definitions of some of the most relevant ilities within SEAri (de Weck, Ross, and Rhodes, 2012). It is important to remind the reader at this point that, in this thesis, an approach for formally thinking about the introduction of design elements that can drive the emergence of ilities in complex systems is introduced and discussed.

Table 2-1: MIT SEAri definitions for selected set of ilities.

Ility	Definition
Robustness	The ability of a system to maintain its level and set of specification parameters in the context of changing system external and internal forces
Changeability	The ability of a system to alter its form, and consequently possibly its function, or operations, at an acceptable level of resource expenditure
Flexibility	The ability of a system to be changed by a system-external change agent with intent
Adaptability	The ability of a system to be changed by a system-internal change agent with intent
Evolvability	The ability of an architecture to be inherited and changed across generations (over time)
Survivability	The ability of a system to minimize the impact of a finite duration disturbance on value delivery
Versatility	The ability of a system to satisfy diverse needs for the system without having to change form (measure of latent value)
Scalability	The ability of a system to change the current level of a system specification parameter
Modifiability	The ability of a system to change the current set of system specification parameters
Interoperability	The ability of a system to effectively interact with other systems
Reconfigurability	The ability of a system to change its configuration (component arrangement and links)
Agility	The ability of a system to change in a timely fashion
Extensibility	The ability of a system to accommodate new features after design

2.5 Managing Uncertainty: (Real) Options

Options theory was originally conceived in finance, a field also plagued by unavoidable uncertainty (but one that, in general, can leverage a lot of data). Inspired by the more abstract financial options, real options theories emerged over time for capital investment decisions in business. The appeal of such systematic approaches for the management of uncertainty were so high that the field of systems engineering also adopted and adapted some of the concepts around options theory. Real options in system design have been linked to flexibility – one of the most frequent ilities discussed in the systems engineering literature. This section presents an overview of options theory and valuation techniques in these three fields.

2.5.1 On the Concept of Option

The word "option" comes from the Latin noun "optĭo, optionis," which means: "power of choosing."[8] This connotation has survived the years (and expanded into new ones as well), and it has become the main concept behind the following theories in finance, business and system design.

2.5.1.1 Options in Finance

The idea of a financial option can be dated back to Bachelier (1900), who derived a closed formula for the pricing of standard call and put options in his theory of speculation. However, the fame and importance of financial options rose only with the groundbreaking work of Black and Scholes (1973), who directly connected option pricing to hedging strategies. Ever since, option theory has been very successful in finance, and many books have been written on it (Cox, 1985; Hull, 2006).

In finance, an option is an instrument that gives its owner the *right*, but *not the obligation*, to buy or sell an underlying stock at a previously specified price (the strike price). He or she can do so on or before the established expiration date of the option. Two main types of financial options exist: call options and put options. The former provides the right to buy, while the latter the right to sell. Fundamental to the generation of profit is the strike price. In fact, for a call option, the right to buy the stock is only exercised when the strike price of the option is less than the price of the stock. Similarly, for a put option, the right to sell the stock is only exercised when the strike price exceeds the price of the stock. It is important to note that the value of the option lies in the fact that its acquisition limits downside uncertainty (i.e., risk) and exploits upside uncertainty (i.e., opportunity).

[8] Among other slightly different connotations (http://www.latin-dictionary.net/search/latin/optio)

2.5.1.2 Real Options in Business

The field of Real Options emerged from the intention to apply concepts similar to financial options theory to capital investment decisions in business (i.e., in the context of strategic decision making). Myers (1984) first introduced the term "real option," with the adjective "real" signifying a certain tangibility of the assets underlying the option (as opposed to the more ethereal financial options). The whole idea behind the concept of real options in strategic decision making is to value investment decisions by taking into account also options (i.e., the power of making a choice) that are available in the future.

Real options are associated with flexible decisional frameworks, wherein it is possible to make decisions at later points in time, depending on the unfolding of uncertainty. In the literature, there are a variety of strategies that use real options to embed flexibility in the strategic design concepts. Trigeorgis (1998) describes six canonical strategies: (1) deferring capital investment until more favorable exogenous (market) conditions arise; (2) staging asset deployment strategically over time (as opposed to fielding all capacity at once); (3) scaling up or down the operations by expanding or reducing production capacity; (4) abandoning a project that is likely to fail (when there is the possibility of making profit from selling existent assets); (5) switching production output and/or input; (6) investing in research and development. Abandoning a project (4) and expanding an investment (3) are two important kinds of real options often taken into account when valuing the decisions of whether to invest. The former is often compared to a financial put option, insofar as it aids in managing downside uncertainty (risk). The latter is often compared to a financial call option, insofar as it aids in managing upside uncertainty (opportunity).

Important differences exist between real options and financial options. Mikaelian (2009) points out how "analogies have been made between financial and real options, mainly to justify the use of financial option valuation methods to value decisions." However, such "analogies are quite weak because of many differences between financial and real options." She discusses how, while financial options are precisely defined and parameterized, the definition of real options (as "the right, but not the obligation, to take an action at a future time") is more elusive. Mikaelian (2009) further discusses some of the main differences between the two types of option: financial options and their underlying financial assets can be traded, while real options are not publicly traded; in the context of valuation, financial options have well defined (fixed) strike prices, while it is not clear what a strike price really corresponds to in real options; unlike real options, financial options have a clearly defined action (buy or sell stock at strike price) that can be exercised before the expiration date of the option; finally, in the case of real options (and unlike financial options) there is not necessarily a legal contract that enforces the ability to exercise the future action (this makes the use of the term "right" in the definition of a real option controversial).

2.5.1.3 Real Options in System Design

As mentioned in previous subsections, the appeal of having the "power of choosing" at a later point in time is quite strong across various disciplines concerning strategic design. Hence, over the years, options analysis has translated from finance, to real options in business, to real options in the domain of system design. This is because the idea of an option is easily generalizable: any decision that can be postponed to be taken at future points in time can be considered a real option (as long as one has the right – but not an obligation – to execute such a decision). In the context of system design, Wang and de Neufville (2006) discuss an important distinction between real options "on" projects and real options "in" projects. The former refer to strategic decisions regarding strategic enterprise and project-related investments (Copeland and Antikarov, 2001); the latter refer to engineering design decisions.

Mikaelian (2009) discusses how real options on and in projects are located within two isolated silos in the system enterprise endeavor. This is due to the fact that engineers are traditionally concerned with the design of real options in systems, while managers and strategic analysts are concerned with the domain of capital investment and real options on projects. She argues that "this hinders a holistic approach to pro-actively designing flexibility in an enterprise, since real options implemented without consideration of factors and possibilities outside of each silo may lead to suboptimal means of managing uncertainty within enterprises." In response to this problem and motivated by the need for an integrated approach to real options analysis, Mikaelian (2009, pp. 90-91) introduces a new characterization of a real option, wherein she distinguishes between *mechanism* and *type* (Figure 2-5):

> **Mechanism**: A mechanism is defined as the set of actions or decisions that either directly or indirectly enables a real option. An active mechanism is defined as a mechanism that directly enables a real option. For example, designing a modular payload bay for a mini air vehicle is an active mechanism that directly enables the flexibility to switch the type of payload. A passive mechanism is defined as a mechanism that indirectly enables a real option. For example, the decision to buy a plant is an indirect enabler of the real option to shut down the plant. It is not a direct enabler because the flexibility to shut down the plant already existed and buying the plant simply enables the owner to exercise this flexibility.
>
> **Option type**: The option type is characterized by the set of actions or decisions that may be exercised by the owner of the real option. For example, the option to switch the payload of a mini air vehicle, the option to abandon a project and the option to enter a new market are different types of options, referred to as an operational option, abandonment option and growth option respectively.

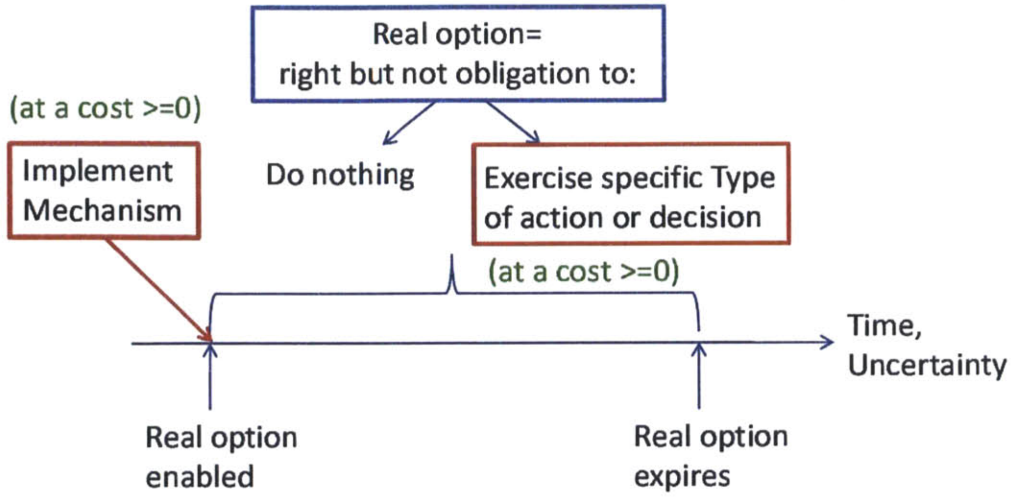


Figure 2-5: Anatomy of a real option (Mikaelian, 2009).

Furthermore, related to the concept of mechanism and option type, Mikaelian (2009) introduces the ideas of optionability and realizability. Optionability is the ability for a mechanism to enable more than one option type. Realizability is the ability for an option type to be enabled by more than one mechanism. Lastly, she defines flexibility as the ability for more than one option type to impact a given objective. It is important to note here that an option type is akin to one of the canonical real option strategies introduced in earlier subsections (e.g. expand, abandon). A mechanism is the actual action, decision, or entity that enables the execution of a real option.

This characterization introduced in Mikaelian (2009) resonates closely with that of path enabler and change mechanism discussed in (Ross, 2006). In fact, these two analogize well with the concepts of mechanism[9] and type, respectively. A path enabler is *what* gives the *option* of executing the change mechanism (i.e., the *method* through which a system goes from state A to state B). The union of the two is at the basis of a change option, discussed more in depth in chapter 5. In relation to the framework in Mikaelian (2009), optionability is the ability for a path enabler to enable more than one change mechanism; realizability is the ability for a change mechanism to be enabled by more than one path enabler.

Similar to real options in business, real options in system design diverge quite a lot from financial options. Myers (1984) discusses some of the major challenges encountered when trying to apply financial option theory to system design. Fitzgerald (2012) summarizes some of the most salient issues in trying to do so in Table 2-2.

[9] It is unfortunate that the concept of "mechanism" described in Mikaelian (2009) corresponds to that of path enabler (and not the more affine "change mechanism") described in Ross (2006).

Table 2-2: Issues with Applying Black-Scholes Assumptions to Engineering Systems (Fitzgerald, 2012)

Black-Scholes Assumption	Issue with Engineering System
European option (fixed exercise date)	Options are typically embedded and can be exercised at any time
Zero arbitrage	Large scale systems are not being traded on a perfect market, and inefficiencies may exist
Geometric Brownian Motion of asset, with constant drift and volatility	Long system lifetimes make it impossible to guarantee that these assumptions are true, or will remain true
Infinite divisibility of asset	Options are frequently binary, go/no-go operations
Existence of a risk-free interest rate	Engineering system value is frequently nonmonetary, making this hard to define

2.5.2 On the Identification of Options in System Design

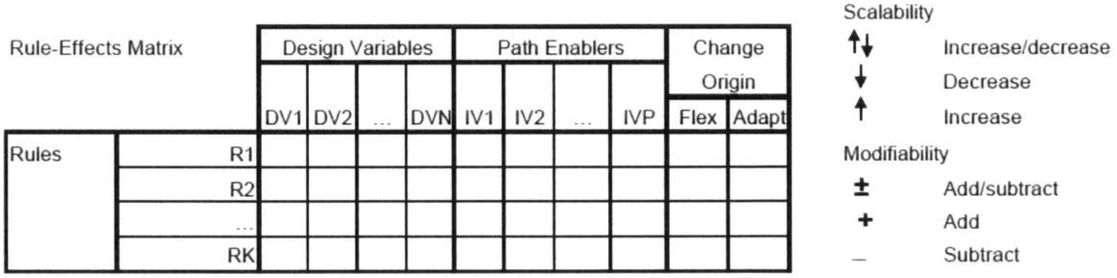
Given the strategic appeal of including real options in the design of systems, there has been increasing interest in the literature about ways to identify and introduce enablers for options in the design of systems. Some of the ideas and methods discussed in the literature are introduced below.

An important tool in engineering design is the Design Structure Matrix (DSM), first introduced in (Steward, 1981). Such a matrix can be thought of as a graphical representation of the design of a system, where the rows and columns list all the relevant design and management components. A careful analysis of a DSM can shine light on how the design and management components are connected, and how the information flows. Such a study of the matrix can enable the identification of specific instances where flexibility is more easily (and valuably) implementable.

Change Propagation Analysis (CPA) – e.g., described in Giffin et al. (2009) – leverages DSMs and statistical analysis to characterize change propagation in complex technical systems. CPA uses the absorber-multiplier framework, where change multipliers are effectively potential areas to insert flexibility. Sensitivity DSM is another method (conceptually similar to CPA) used for representing change propagation through the system. Sensitivity DSM provides a high-level view of the design representation, wherein it is possible to focus on the design elements most apt for the insertion of flexibility. Kalligeros (2006) describes such a method and applies it to the design of an offshore oil platform.

Silver and de Weck (2007) present the Time-expanded Decision Network (TDN), and an application thereof. A TDN aids insofar as modeling the lifecycle of a system in order to identify and value opportunities to insert real options into the system. These options enable flexibility. The TDN is initially created as a network of system configuration nodes connected by change paths (with associated switching costs). The nodes are then represented temporally, and, for a given demand profile, the maximum-value path between the nodes over time can be calculated. If different nodes are dominant design decisions under different demand profiles, the analysis can be iterated with the addition of potential real options to lower the switching costs between those domains in order to improve robustness against demand changes. Fitzgerald (2012) points out how the TDN is appealing because of its ability to not only value potential options but also to identify where the insertion of an option would be most useful, thereby "assisting the human-centered process of ideation." However, "the entire method is predicated on the same revenue and cost modeling necessary for the traditional financial calculations such as NPV, which is extremely difficult to model and justify for certain applications."

Ross (2006) introduces the Rule-Effects Matrix (REM) in order to help visualize and track transition rules – algorithms that instantiate change mechanisms (i.e., "the specification of how one design can be changed into another"). The REM contains rows as proposed rules, and columns as the design and path enablers. An entry in a cell of the matrix means that, with the presence of the appropriately marked path enabler, for a given transition rule (row *i*), there occurs a corresponding change in a design variable (column *j*). An example of a Rules-Effects Matrix (REM) is shown in Figure 2-6. The ability to change at a future point in time (i.e., executing an option to change) is given by the existence of one or more path enablers. This concept is revisited in chapter 5.



Rule-Effects Matrix		Design Variables				Path Enablers				Change Origin	
		DV1	DV2	...	DVN	IV1	IV2	...	IVP	Flex	Adapt
Rules	R1										
	R2										
	...										
	RK										

Scalability	
↑↓	Increase/decrease
↓	Decrease
↑	Increase
Modifiability	
±	Add/subtract
+	Add
–	Subtract

Figure 2-6: The Rules-Effects Matrix (Ross, 2006).

2.5.3 On the Valuation of Options

As mentioned in previous subsections, the idea of a financial option can be dated back to Bachelier (1900). However, a turning point for financial options theory was the work of Black and Scholes (1973), who directly connected option pricing to hedging strategies. Several quantitative techniques have been developed for valuing financial options since then, all based on the Black-Scholes model. They range from analytic solutions to the Black-Scholes equation, to binomial lattice methods, to Monte Carlo simulations.

The Black-Scholes model is a closed-form formula for pricing a special case of financial options: European options. These options can only be exercised on a specified date. Some of the key assumptions behind the Black-Scholes model are listed in Table

2-2. Binomial lattice methods are discrete time models for valuing financial options. Introduced by Cox, Ross and Rubinstein (1979), they have become widely used for options pricing. The underlying assumption is that the stock price follows a multiplicative binomial process over discrete periods. When at a given point in time (node), the stock price S can assume only two prices at the next time step – $u \cdot S$ or $d \cdot S$, with probability q and $1 - q$, respectively. In order to calculate the price of the option using such a model, it is possible to use dynamic programming techniques. Lastly, for modeling complex option types, it is possible to use Monte Carlo simulations (Boyle, 1977). Closed-form analyses of financial options have also expanded in scope through the years, as new classes of options have arisen. For instance, Câmara (2006) studies a new class of investment options (termed "event-contingent options") by constructing payoff functions and deriving closed-form solutions for the four types of event-contingent options he describes ("options to invest and divest in projects that are dependent on other projects of the same firm or that are conditioned by projects of other firms in its value chain").

Real Options Analysis (ROA) was developed in an attempt to apply options theory to business and management situations (as well as system design). In ROA, the end goal is to assign a monetary value to the flexibility provided by options in (or on) systems (e.g., option to increase production). To this end, many of the standard financial valuation techniques, such as Discounted Cash Flow (DCF) and Net Present Value (NPV) analysis, are used in ROA. However, as shown in Table 2-2 in section 2.5.1.3, some of the assumptions inherent in the Black-Scholes model make this jump questionable.

Fitzgerald (2012) discusses this problem at length, and, in an attempt to provide system designers with a more generally applicable method for valuing changeability, he introduces the Valuation Approach for Strategic Changeability (VASC). VASC is designed to "capture the multi-dimensional value of changeability while limiting the number of necessary assumptions." At the basis of the method is the conceptualization of uncertainty provided by Epoch-Era Analysis (mentioned in previous sections and further described in chapter 4). VASC also introduces a suite of new metrics (including Effective Fuzzy Pareto Trace, Fuzzy Pareto Number, and Fuzzy Pareto Shift), which differs from the monetary metrics of ROA, and which capture different types of valuable changeability information.

2.5.4 On the Selection of (Portfolios of) Options for Risk Mitigation

In finance, the two most important goals of an investor are (1) to maximize the return on investment, and (2) to minimize its riskiness. A common strategy for reducing risk is that of diversifying the investment's risk by investing in a portfolio of assets (a collection of investments – stock, bond or cash – held by an investor). As shown in Figure 2-7, this reduces total risk (but does not reduce the systematic risk inherent in the market). Markowitz (1952) first addressed the problem of the selection of efficient portfolios of financial assets in his Modern Portfolio Theory (MPT).

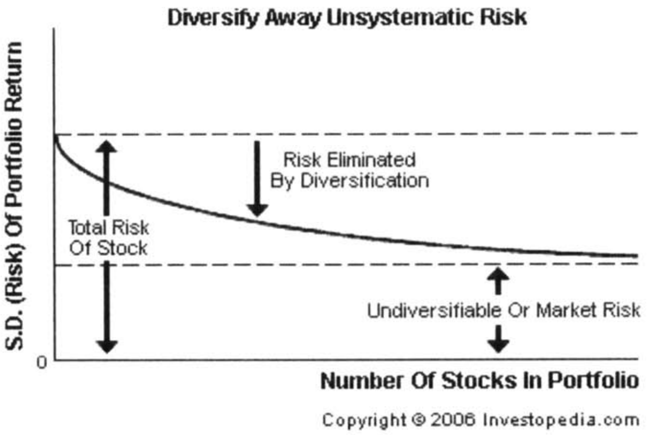


Figure 2-7: Systematic risk vs. risk that can be eliminated by diversification.

MPT represents the return of each asset as a normally distributed function, and risk as the standard deviation of this function. The portfolio is modeled as a weighted combination of assets, and its return is the weighted combination of individual assets' returns. A covariance matrix describes the uncertainty and correlation of assets. The canonical Markowitz problem is to allocate monetary resources across a number of n risky assets with mean expected returns $\boldsymbol{\mu}$ and return covariance matrix $\boldsymbol{\Sigma}$[10], over a weight vector $\boldsymbol{w} \in W$ (containing the normalization constraints). It is important to note how there are two different versions of the problem: (1) maximizing portfolio expected return, given a certain risk σ (standard deviation) investors are willing to take (i.e., $max\{\boldsymbol{\mu}^T\boldsymbol{w}: \boldsymbol{w}^T\boldsymbol{\Sigma}\boldsymbol{w} \leq \sigma^2\}$); or (2) minimizing portfolio risk, given an expected return r investors are interested in (i.e., $min\{\boldsymbol{w}^T\boldsymbol{\Sigma}\boldsymbol{w}: \boldsymbol{\mu}^T\boldsymbol{w} \geq r\}$). MPT relies on the concept of diversifying portfolios in such a way that allows the collection of assets to have lower risk than any individual asset.

Over the years, many attempts have been made to improve the model, especially by using more realistic assumptions. An interesting extension of MPT is Post-Modern Portfolio Theory (PMPT). This theory has no single author, but it combines theoretical research of several people and institutions. The advancement in PMPT essentially resides in the fact that it adopts non-normally distributed measures of risk, and in particular, seeks to minimize "downside risk" rather than mean variance (Swisher and Kasten, 2005).

There have also been efforts in the application of MPT to other fields. Of particular relevance is the work carried out by Walton (2002), who extended the use of MPT to space systems design selection. In his work, he recognized some of the key limitations of the theory and made opportune modifications to it. In particular, in an attempt to

[10] It is important to note here that the concept of risk is associated with that of variance, i.e.: a measure of the second moment around the expected return of the asset. This is an example of modeling uncertainty as a random aleatory process, and statistically characterizing it.

account for the upside potential from uncertainty (i.e., opportunities), he introduced the concept of semi-variance (upside and downside) as measures of one-sided uncertainty.

Despite the widespread popularity of MPT, a fundamental drawback from the practitioner's perspective is that $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ are rarely known with complete precision. Trying to run an optimization algorithm often times can only exacerbate the problem by finding solutions that are "extreme" allocations of resources. A response to this problem has come from the field of robust optimization (Bertsimas, Brown and Caramanis, 2011), which enables the use of robust mean return and covariance information models to attenuate the difficulty in quantifying such parameters. Lobo and Boyd (2000) and Tütüncü and Koenig (2004) propose different uncertainty structures (e.g., box uncertainty set, ellipsoidal uncertainty set, etc.) for the robust version of Markowitz's optimization problem.

2.6 Systems of Systems (SoS)

In the systems engineering community, the frequency of appearance of the term "System of Systems" has risen over the past two decades. Despite its common use, much ambiguity exists around its definition. Maier (1998) was quick to point out such ambiguity. He discusses how the concept of "system" is generally understood, and it falls along the lines of the following definition provided by the International Council on Systems Engineering (INCOSE): "a collection of components organized to accomplish a specific function or set of functions." However, the concept of "System of Systems" lacks formal definition, and it could lead to equivocality. For example, a laptop is a system; however, it could also be considered a "System of Systems" in that it is a system that includes other systems like a monitor, a hard drive, a processor, and so on (Maier, 1998). The main idea behind SoS was to describe a particular class of systems that is more than just simply applying the definition of "system" to a system. This class of system has unique characteristics that come about when one combines interacting systems in a way to achieve additional functionality and emergent properties. In an attempt to start clarifying some of the distinguishing features of SoS, Maier (1998) proposes five key characteristics of SoS:

- *Operational Independence of the Elements*: If the System of Systems is disassembled into its component systems, the component systems must be able to usefully operate independently. The System of Systems is composed of systems that are independent and useful in their own right.
- *Managerial Independence of the Elements*: The component systems not only can operate independently, they do operate independently. The component systems are separately acquired and integrated, and maintain a continuing operational existence independent of the System of Systems.
- *Evolutionary Development*: The System of Systems does not appear fully formed. Its development and existence is evolutionary with functions and purposes

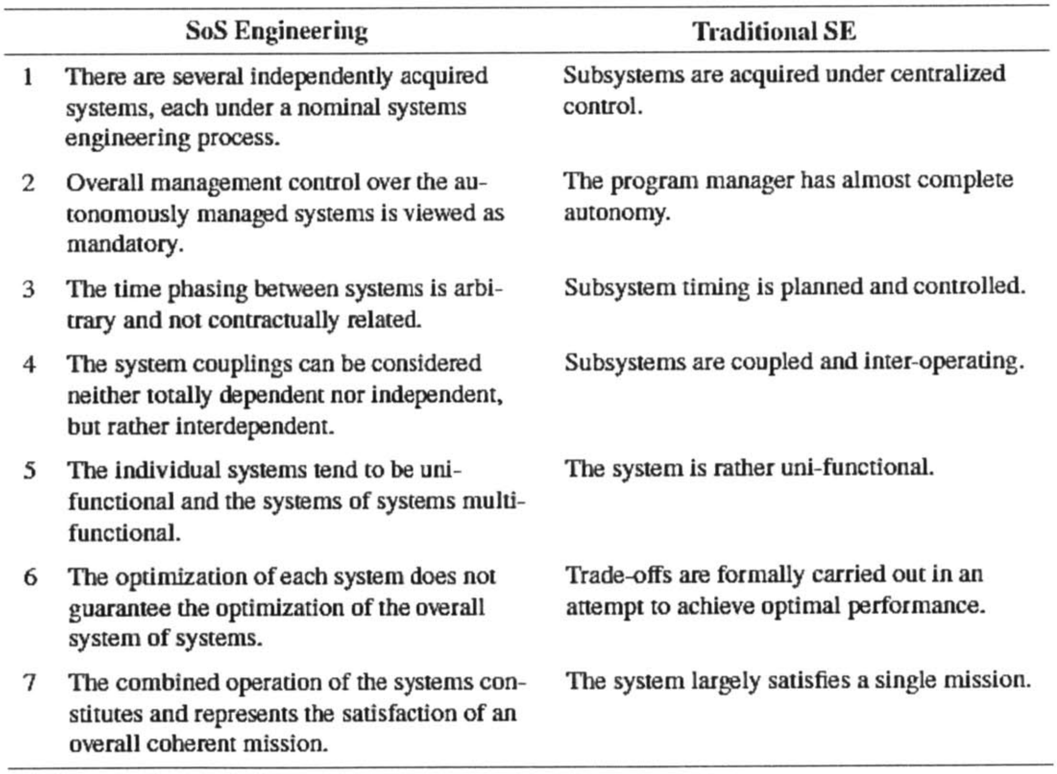
added, removed, and modified with experience, over time. Dahmann et al. (2011) discuss the "wave model" from an SoS implementer's perspective.

- *Emergent Behavior*: The System of Systems performs functions and carries out purposes that do not reside in any component system. These behaviors are emergent properties of the entire System of Systems and cannot be localized to any component system. The principal purposes of the Systems of Systems are fulfilled by these behaviors.
- *Geographic Distribution*: The geographic extent of the component systems is large. Large is a nebulous and relative concept as communication capabilities increase, but at a minimum it means that the components can readily exchange only information and not substantial quantities of mass or energy.

Mekdeci et al. (2011) perform a thorough literature survey, and identify additional characteristics of SoS that distinguish them from traditional "monolithic" systems (Jamshidi, 2009). Among them are: abstruse emergence, distributed authority (Boardman and Sauser, 2006), multi-functionality (Eisner, Marciniak, and McMillan, 1991), increased contextual diversity, decreased system awareness, and dubious validation (Ellison and Woody, 2007).

The practice of systems engineering applied to the case of SoS has become known as SoS engineering (SoSE). Eisner et al. (1991) summarize some key differences between SoSE and traditional systems engineering (Table 2-3).

Table 2-3: Seven differences between SoSE and traditional SE (Eisner et al., 1991).



	SoS Engineering	Traditional SE
1	There are several independently acquired systems, each under a nominal systems engineering process.	Subsystems are acquired under centralized control.
2	Overall management control over the autonomously managed systems is viewed as mandatory.	The program manager has almost complete autonomy.
3	The time phasing between systems is arbitrary and not contractually related.	Subsystem timing is planned and controlled.
4	The system couplings can be considered neither totally dependent nor independent, but rather interdependent.	Subsystems are coupled and inter-operating.
5	The individual systems tend to be uni-functional and the systems of systems multi-functional.	The system is rather uni-functional.
6	The optimization of each system does not guarantee the optimization of the overall system of systems.	Trade-offs are formally carried out in an attempt to achieve optimal performance.
7	The combined operation of the systems constitutes and represents the satisfaction of an overall coherent mission.	The system largely satisfies a single mission.

Furthermore, in order to distinguish various aspects of their management, there has been discussion in the literature about different types of SoS. In particular, Maier (1998) distinguishes among three different classes of SoS:

> **Directed**: Directed systems-of-systems are those in which the integrated system-of-systems is built and managed to fulfill specific purposes. It is centrally managed during long-term operation to continue to fulfill those purposes, and any new ones the system owners may wish to address. The component systems maintain an ability to operate independently, but their normal operational mode is subordinated to the central managed purpose. For example, most integrated air defense networks are centrally managed to defend a region against enemy systems, although its component systems retain the ability to operate independently, and do so when needed under the stress of combat.
>
> **Collaborative**: Collaborative systems-of-systems are distinct from directed systems in that the central management organization does not have coercive power to run the system. The component systems must, more or less, voluntarily collaborate to fulfill the agreed upon central purposes. The Internet is a collaborative system. The IETF works out standards, but has no power to enforce them. Agreements among the central players on service provision and rejection provide what enforcement mechanism there is to maintain standards. The Internet began as a directed system, controlled by the US Advanced Research Projects Agency, to share computer resources. Over time it has evolved from central control through unplanned collaborative mechanisms.
>
> **Virtual**: Virtual systems-of-systems lack both a central management authority and centrally agreed upon purposes. Large-scale behavior emerges, and may be desirable, but the supersystem must rely upon relatively invisible mechanisms to maintain it.

In addition to these three, another class was identified in Dahmann and Baldwin (2008):

> **Acknowledged**: Acknowledged SoS have recognized objectives, a designated manager and resources for the SoS, however, the constituent systems retain their independent ownership, objectives, funding, and development and sustainment approaches. Changes in the systems are based on collaboration between the SoS and the system.

2.6.1 Maritime Security (MarSec) SoS

A Maritime Security (MarSec) SoS is considered in many instances throughout the remainder of this thesis as the main case study. The operational goal of the MarSec SoS is to provide maritime security for a particular littoral area of interest (AOI). The system is required to detect, identify and board boats that constantly enter and exit the area of interest. Moreover, upon request, it must be capable of providing for search and rescue of sinking boats or entities in danger within the area of interest. This SoS is described at length in Mekdeci (2013).

Some of the components of the SoS are different types of Unmanned Aerial Vehicles (UAVs), manned patrol aircraft, helicopters, patrol boats, command centers, airbases, docks, and radar towers. Operational choices include the segmentation of the area (in terms of what is covered by different constituent systems), task assignment (what functions are performed by the different constituent systems), and the number of operators per UAV.

This SoS can be considered, to a large extent, a directed SoS, as its managers have full control over the constituent systems (with the exception of, for example, the satellite used for communication relay and Port Authority-managed patrol boats, for which the control is partial or effective under specific circumstances). Furthermore, the MarSec SoS exhibits some of the key characteristics that distinguish Systems of Systems from traditional systems discussed earlier (Maier, 1998; Mekdeci et al., 2011).

2.7 Summary

The literature review presented in this chapter is an attempt at elucidating relevant knowledge with regard to (1) the nature of complex sociotechnical engineering systems, (2) the conceptualization and modeling of uncertainty in such systems (literature drawn from the field of strategic system design and policy making), and (3) the conceptualization and modeling of responses to the presence of uncertainty in the fields of finance, business, and system design.

The next chapter (chapter 3) presents the overview of a method for architecting SoS with ilities, and excerpts of its application to the MarSec SoS case. This method is the output of a larger research effort, and the overview given in chapter 3 serves as a framing for the remainder of the thesis, which discusses in detail some of the steps in the method. Building on some of the concepts presented in this chapter (and introducing further relevant ones), part of this thesis (chapter 4 and 5) tackles the problem of systematically thinking about uncertainty and ilities by presenting a formal approach for modeling uncertainty and responses to it in the context of complex system design. Within chapter 4, a structured approach for identifying uncertainty is also discussed and applied to the MarSec SoS case. Then, using this modeling framework, some empirical instances (from the domains of military SoS and policymaking) of changing upon the resolving of uncertainty are discussed (chapter 6). Lastly, a structured approach for the identification of ility-driving elements (i.e., possible responses to uncertainty) is presented and applied to the MarSec SoS (chapter 7).

3 A Method for Architecting Systems of Systems with Ilities

"Fulfillment of purpose or adaptation to a goal involves a relation among three terms: the purpose or goal, the character of the artifact, and the environment in which the artifact performs."

– Herbert A. Simon (1996)

This chapter introduces the SoS Architecting with Ilities (SAI) method, which is intended to enable the architecting of Systems of Systems with an emphasis on enhancing lifecycle value sustainment from the early phases of design. Although the method is directly targeted at Systems of Systems, only certain sub-steps are particular to SoS, and most of the method is generalizable to traditional systems and engineering systems.

A broad overview of the SAI method is introduced in this chapter to frame the discussion around the design of systems with ilities. Alongside, excerpts from the application of the method to the Maritime Security SoS are presented. In successive chapters, some of the material related to certain steps in the method is described in more detail. Chapter 4 explicitly addresses the modelling and characterization of the outer environment (and the possibly changing goals) in which systems perform (directly related to step 3 in the method). Chapter 5 discusses the nature and usage of change options and resistance properties to design systems that are more likely to display ilities over their lifecycle (directly related to step 5 in the method).

3.1 Motivation

Chapter 1 presented the problem of designing modern-day complex systems, an endeavor subject to many different types of uncertainty. The literature review provided in chapter 2 further investigated the nature of complex, sociotechnical systems (and the challenges associated with them), as well as current state-of-the-art practices for the conceptualization of uncertainty, and the ways to strategically design systems that are able to deal with the unfolding of uncertainty over time.

As discussed in chapter 1, Neches and Madni (2012) criticize the traditional approaches and processes in systems engineering, as they do not address the various problems that affect modern-day system design. In their manifesto on Engineered

Resilient Systems (ERS), they point out that "it is important to realize that, when needs are prematurely translated into requirements or key performance parameters, both the process and the product of engineering suffers the consequence."

This need for thinking about the long-term dynamics of the system can be addressed (to the extent possible) by the introduction and analysis of ilities (i.e.: lifecycle properties). Ilities are properties that emerge through the interaction of a system (i.e., an artifact) and its outer environment. Such properties inevitably manifest themselves only after the system has been put in use. Modern ilities (e.g. flexibility) are one response to mitigate the impact of dynamic complexities on system value delivery over time (discussed in 2.2.2). Since a goal of systems engineering is to foster value sustainment throughout a system's lifecycle (Beesemyer, 2012), a method that can help to create SoS with explicit consideration for value sustaining ilities would be invaluable to the modern practicing systems engineer and architect.

With SAI, value sustainment is facilitated by guiding systems architects in designing not only for static functional requirements, but also for ilities (e.g., flexibility, adaptability, robustness), via the intentional inclusion of ility-driving elements (i.e., change options and resistance properties, discussed in depth in chapter 5). The SAI method enables the inclusion, quantification and tradeoff of desired ilities. The method ultimately results in specific requirements, targeted as such dynamic properties (as opposed to static functional requirements).

3.2 Overview of the SAI Method

The SAI method (Ricci, Fitzgerald, Ross and Rhodes, 2014) builds upon the Responsive Systems Comparison method (Ross et al., 2009) by adding more explicit steps and analytical tools for the identification and inclusion of relevant ilities early in the design process. Within SAI, ilities in SoS are driven by the introduction of ility-driving elements targeted at delivering contingent value, i.e.: value that materializes upon the resolving of a contingency (i.e., uncertainty).

The SAI method for SoS architecting is comprised of eight steps, each in turn composed of several sub-steps (described more in depth in the next section). It is an end-to-end process that guides systems architects throughout all phases of conceptual design: from value definition and elicitation, to alternative generation, to final selection. This chapter is intended to provide the reader with a general understanding of all the activities associated with the steps in SAI. Future chapters will put particular focus on those linked to the conceptualization of uncertainty in systems, as well as the introduction of ilities. SAI uses a variety of tools and methods developed over the course of a large research project. When possible, a general description of these is provided. Furthermore, some examples from the application of the SAI method to a Maritime Security SoS case study are presented.

The eight steps characterizing the SAI method are shown in Figure 3-1, along with the feedback and feed forward relationships that exist among them. The general flow of the SAI method is summarized in the following brief description of the eight SAI steps (Discussed in further detail in the next sections):

1. *Determine value proposition and constraints*: the first step involves identifying, understanding, and capturing the overall SoS architecture value proposition(s). In this step, the interactions and needs of key stakeholders are identified, and design constraints are also listed.

2. *Identify potential perturbations*: using an array of techniques, potential perturbations (in the design, context or needs) that can possibly interfere with SoS value delivery are identified and categorized.

3. *Identify initial desired ilities*: identify ilities that promote the desired long-term behavior of the SoS using a variety of analytical tools (e.g., ilities hierarchies, semantic basis tool, etc.). Here, information about relevant perturbations can lead to particular interest in certain ilities over others.

4. *Generate initial architecture alternatives*: the purpose of this step is to propose various value-driven (value proposed in step 1) SoS architecture alternatives in terms of design and operational variables, with associated concepts of operations.

5. *Generate ility-driving options*: this step is concerned with the generation and selection of elements that drive contingent value (i.e., options) to include in the initial architecture. These will eventually result in enabling desired ilities (identified in step 3). These elements are the key linkage to the emergence of lifecycle properties over time, because they are what enables change – or resistance to change – in the SoS when exogenous perturbations threaten SoS value delivery, or when opportunities to enhance value delivery arise.

6. *Evaluate potential alternatives*: here a model is built and executed to evaluate different SoS architecture alternatives in terms of various metrics, including performance (i.e. attributes and costs) and ility metrics.

7. *Analyze architecture alternatives*: the analysis in this step is aimed at developing insight and understanding in the trade-offs between static value and ility behaviors (i.e., contingent value) within various SoS architectures in terms of design and operations choices. Various analytical techniques – such as Multi-Epoch Analysis (Ross et al., 2009), Era Analysis (Ross et al., 2009), and the Valuation Approach for Strategic Changeability (VASC) (Fitzgerald et al., 2012) – are employed in this step.

8. *Trade-off and select "best" architecture with ilities*: in the final step, selection criteria for nominating the "best" architecture with ilities are justified and documented, and ilities requirements are generated.

The remainder of the chapter describes each step in more depth, highlighting some of the sub-steps and activities associated with them, as well as presenting some excerpts from the application to the MarSec SoS. Inputs and outputs to the steps are listed, as well as the role that they play in the feedback and feed forward relationships that exist amongst the steps.

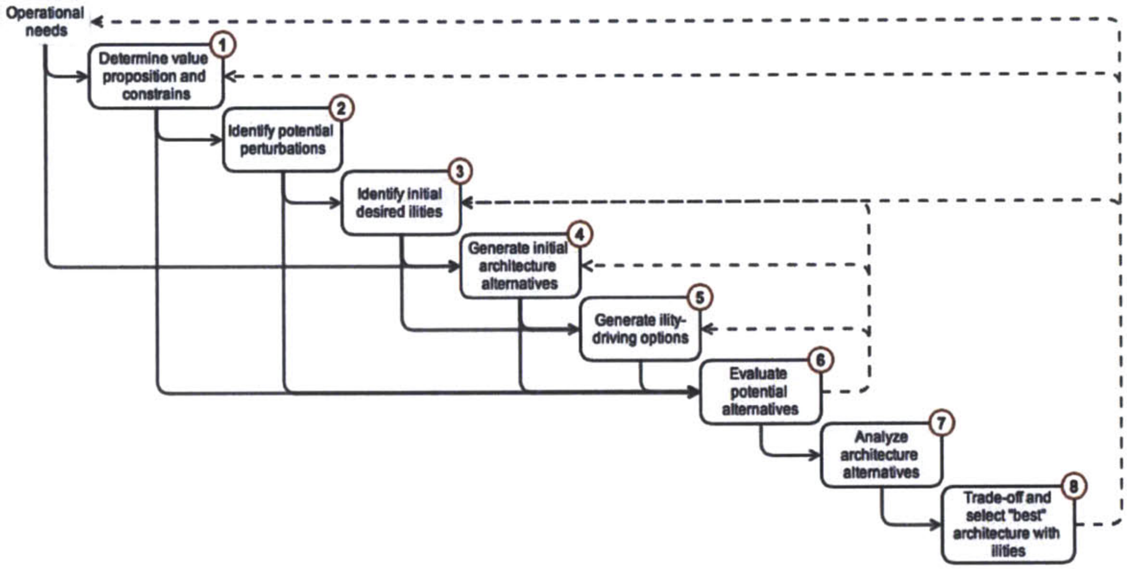

Operational needs
1
Determine value proposition and constrains
2
Identify potential perturbations
3
Identify initial desired ilities
4
Generate initial architecture alternatives
5
Generate ility-driving options
6
Evaluate potential alternatives
7
Analyze architecture alternatives
8
Trade-off and select "best" architecture with ilities

Figure 3-1: Steps in the SAI method. Feedback and feed-forward relationships among the steps are illustrated.

3.2.1 Determine Value Proposition and Constraints

The input to the first step (and the process as a whole) is a description of the overall operational needs statement of the SoS. In the case of the MarSec SoS, for example, such a statement is: "the main operational goal of the MarSec SoS is to provide maritime security for a particular littoral area of interest (AOI)." From such a statement, it is then possible to derive salient attributes (Keeney and Raiffa, 1976) for quantifying the performance of the SoS, which is then mapped onto the value preferences elicited from stakeholders.

Additionally, this step is for assessing any legacy constituent systems that may be available, or required, to be part of the SoS. It is usually better to avoid starting with a solution, and instead to focus on objectives and functions before identifying forms (Keeney and Raiffa, 1996). However, here it is recognized that most SoS are not developed from scratch, but rather inherit components and requirements to some degree (Bergey et al., 2009). Another important part of this step is to assess any important organizational constraints for the architecting of the SoS. Overall, this step is very similar to the standard practice of defining the problem scope, also focusing on identifying all external influences and their potential impact to the value delivery of the SoS.

Some selected key activities and associated outputs in step 1 are discussed in the following paragraphs (in order).

Assess currently available or required constituent systems. Given the high level problem to be solved, the first task is to determine what relevant constituent systems currently exist and could or should (or occasionally *must*) be considered later in the process.

Assess constraints. This task consists of identifying various types of constraints that pose limits or requirements on the SoS. These can include organizational, policy, physical, and geographic constraints, among others. Organizational and policy constraints could include acquisition processes and schedules, workforce skills, and national laws. It is important to note that some apparent constraints could be changed in the future, depending on the nature of the constraint. For example, national law could change, so too could the skill set of the workforce. Physical and geographic constraints include possible geographic scoping of the problem and identification of available infrastructure. A list of organizational and policy constraints ("socio-" type) and physical and geographic constraints ("technical" type) that limit potential SoS architectures are shown in Table 3-1 for the MarSec case.

Table 3-1: Example brainstorming of SoS contraints for MarSec.

Constraint Type	Issue with Engineering System
Organizational	- All constituent systems are under a directed authority (everyone belongs to the same team) - Workforce must not exceed given capacity.
Policy	- Budget constraints. - Constraints and limitations on who the partners can be: do not work with bordering countries (e.g., do not borrow assets from them). - Boarding done by patrol boats with personnel. (No interception UAV, for example: can't use x45.)
Physical	- Only certain types of UAVs and payloads are under consideration, and these have technical specifications that imply physical constraints (e.g., size, speed).
Geographic	- The area of interest is given; it includes both terrain and water. There is a high traffic area within the AOI (the port).

Define SoS enterprise boundary. This task is an important one insofar as helping to identify entities within the SoS over which program managers have complete, partial or no control. Figure 3-2 shows a diagram of the different types of boundaries that apply to an SoS. To help organize, a "supra-decision maker" (for MarSec, a Maritime Security SoS Manager in the program office) – who must represent all interests in order to have a successful program – must be defined.

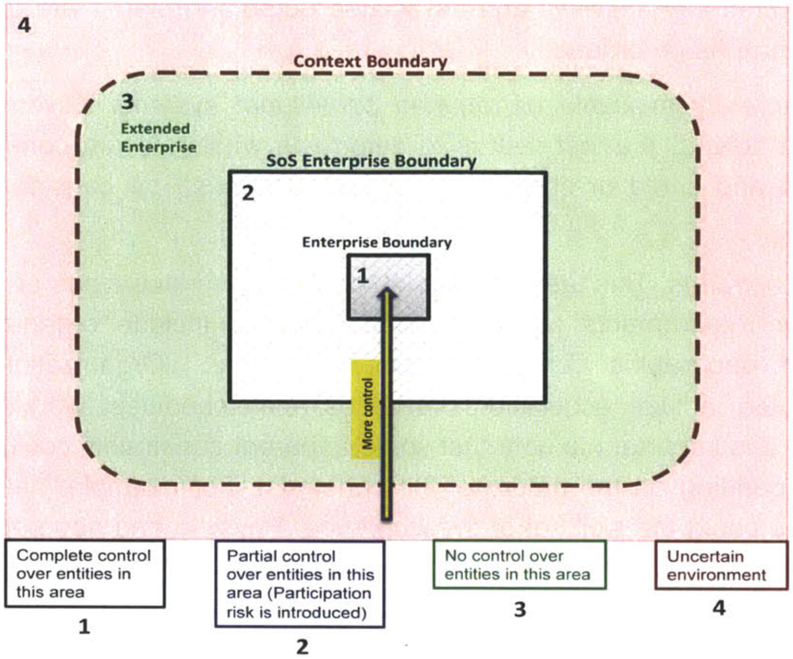

4
Context Boundary
3
Extended Enterprise
SoS Enterprise Boundary
2
Enterprise Boundary
1
More control
Complete control over entities in this area
1
Partial control over entities in this area (Participation risk is introduced)
2
No control over entities in this area
3
Uncertain environment
4

Figure 3-2: Enterprise boundary analysis.

This diagram is used to delineate boundaries for the MarSec SoS. Figure 3-3 illustrates entities identified to be within the first two boundary types (Enterprise Boundary and SoS Enterprise Boundary). MarSec SoS Program managers have full directorial power over the SoS entities that are within the Enterprise Boundary. On the other hand, some degree of uncertainty exists for entities within the SoS Enterprise Boundary, as Program Managers only have partial control over those: these assets may not always be available (participation risk) depending on other tasks they have to accomplish (functional independence) and decisions made by their owners (managerial independence).

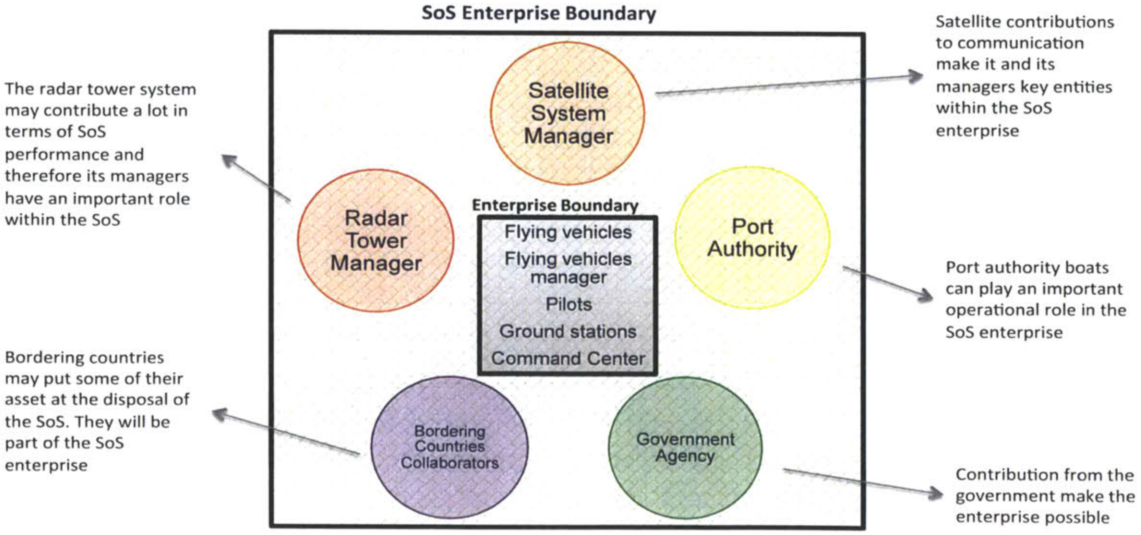

SoS Enterprise Boundary
Satellite System Manager
Radar Tower Manager
Enterprise Boundary
Flying vehicles
Flying vehicles manager
Pilots
Ground stations
Command Center
Port Authority
Bordering Countries Collaborators
Government Agency
The radar tower system may contribute a lot in terms of SoS performance and therefore its managers have an important role within the SoS
Satellite contributions to communication make it and its managers key entities within the SoS enterprise
Port authority boats can play an important operational role in the SoS enterprise
Bordering countries may put some of their asset at the disposal of the SoS. They will be part of the SoS enterprise
Contribution from the government make the enterprise possible

Figure 3-3: Core of MarSec SoS enterprise.

Further enlarging the view, Figure 3-4 shows the context boundary for the MarSec SoS. Between the context boundary and the SoS Enterprise Boundary there is the extended enterprise area. In this area there are entities still relevant for the success of the SoS, but for which the Program Managers have no control over. The clouds shown in Figure 3-4 are categories of exogenous uncertainties that can pierce through the context boundary and the SoS Enterprise Boundary and affect the performance of the SoS. These uncertainty categories are used in Step 2 for the generation of possible perturbations that can affect the value delivery of the SoS.

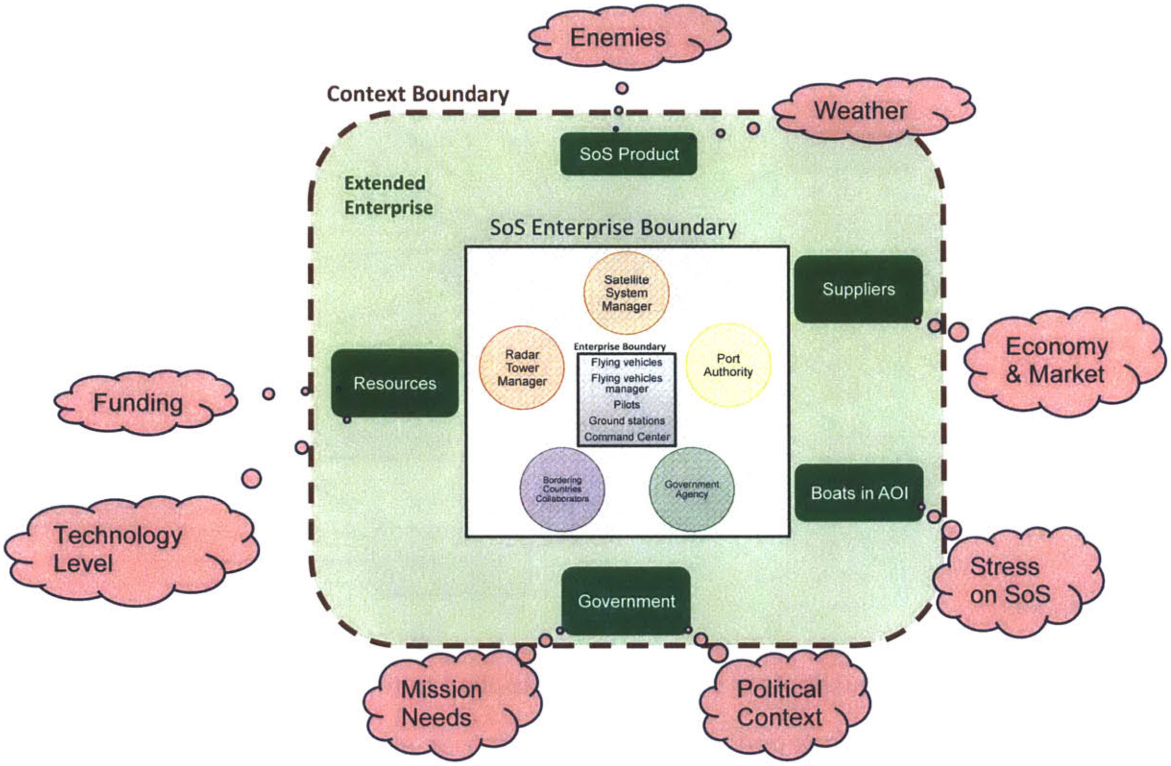


Figure 3-4: Enterprise boundary analysis applied to the MarSec SoS case.

Delineate stakeholder value network. In this task, relevant stakeholders are identified and classified, discerning among decision makers, SoS stakeholders and exogenous stakeholders.[11] Then, from the list of stakeholders, it is possible to develop a stakeholder value network. A stakeholder value network is "a multi-relation network consisting of a focal organization, focal organization's stakeholders, and the tangible and intangible exchanges between the focal organization and its stakeholders, as well as between the

[11] For SoS, stakeholders exist at both the SoS-level, as well as the constituent system level. Exogenous stakeholders might be those who: benefit directly or indirectly from the SoS concepts (but do not explicitly participate as SoS stakeholders), provide the resources needed to develop and maintain the SoS, control supporting infrastructural elements, provide strategic level guidance, oversight, and/or priorities (likely the same stakeholder who identified the overarching operational needs), or are adversely effected by the SoS concept. Exogenous stakeholders are located *outside* the SoS enterprise boundary, as opposed to the SoS stakeholders.

stakeholders themselves." (Feng and Crawley, 2008). This type of diagram can help understand the impacts of both direct and indirect relationships between stakeholders on the success of large engineering projects. An example of stakeholder value network developed for the MarSec SoS is shown in Figure 3-5.

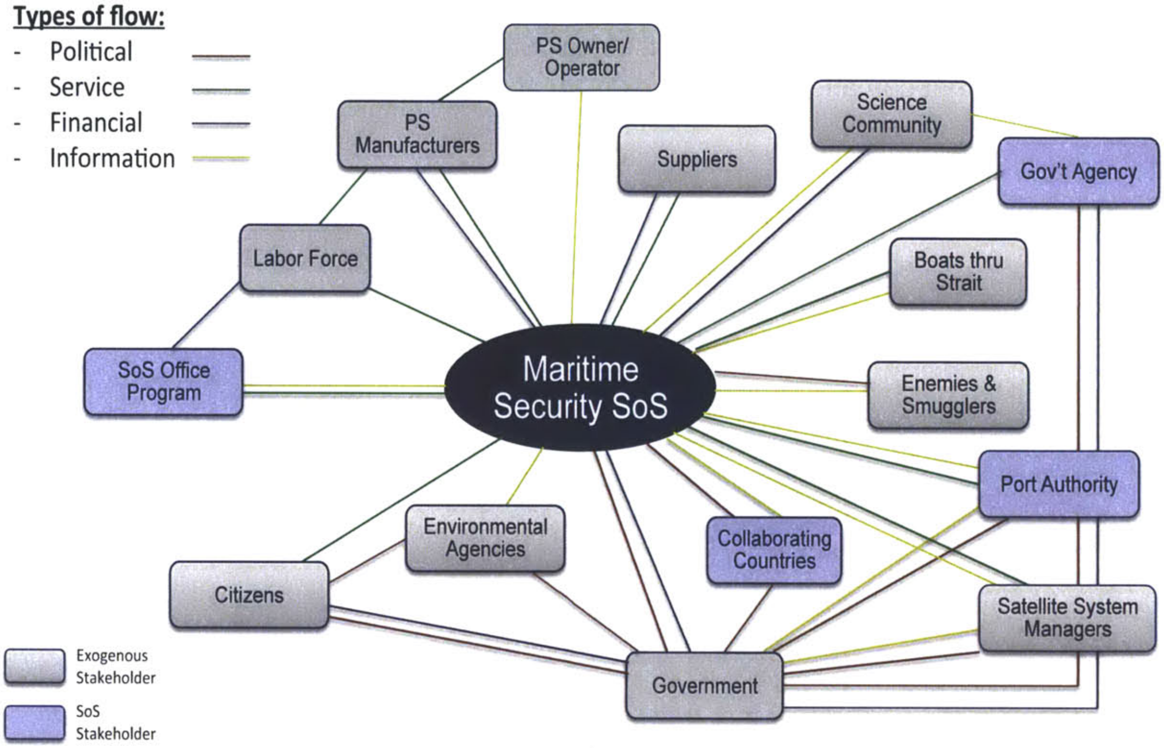


Figure 3-5: Stakeholder value network applied to the MarSec SoS case.

Reconcile value proposition. This task is aimed at eliciting stakeholder value- and design-space preferences with regard to SoS concept, operations and objectives. Different ways of eliciting values exist, also dependent on the value model chosen (Slovic, 1995). For example: if using utility theory (Keeney and Raiffa, 1976), including attribute utility curves and weighting is preferred; if using AHP (Saaty, 2004), include the value tree with weightings from pairwise comparison. The reconciliation of value consists of going from the stakeholders' needs and preferences to specific (quantifiable) attributes of interest for the SoS (e.g., surveillance → detect → probability of detection). The SoS stakeholders have some strategic objectives that would like to ultimately fulfill. High-level objectives spin from the strategic objectives: they are actions that the SoS should be performing in order to achieve the strategic objectives. From the high-level objectives, it is possible to derive a list of attributes of interest; these should be quantifiable and will later be used in order to assess the performance of the various SoS alternatives considered. Figure 3-6 shows an example of mapping stakeholders' strategic objectives to quantifiable attributes.

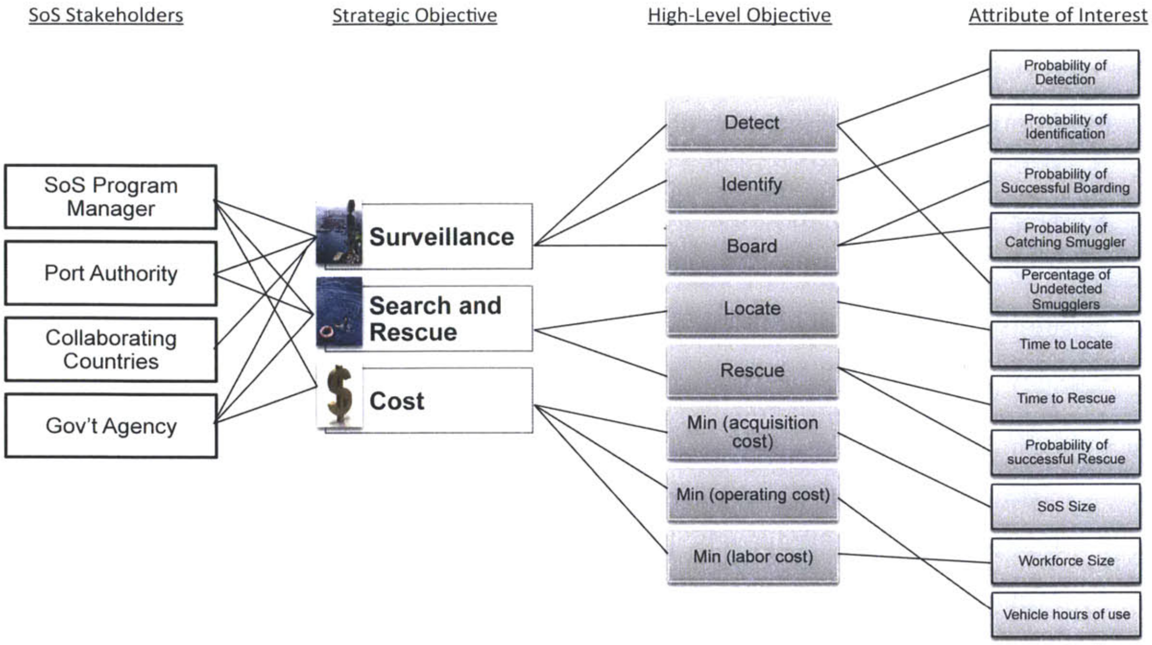


Figure 3-6: Mapping of stakeholders' strategic objectives to quantifiable attributes for the case of the MarSec SoS.

3.2.2 Identify Potential Perturbations

Perturbations can be thought of as a conceptualization (and parameterization) of the uncertain environment in which the system operates. The origin of the concept of perturbation is described in chapter 4, along with a formal way of modeling them within the context of system design.

When considering complex SoS, uncertainty can stem from endogenous and exogenous sources, where the latter are related to context and expectation changes (Ross and Rhodes, 2008). SoS enterprise boundary analysis and other activities performed in Step 1 are helpful toward the determination of possible sources of uncertainty. Furthermore, section 2.3.1 discusses other approaches for characterizing and identifying sources of uncertainty. These uncertainties are then parameterized into perturbations, for the goals of modeling and providing a structure in strategically thinking about uncertainty. In general, perturbations can be thought of as changes in either the design of the system, the context in which it operates, or the expectations of the system. Moreover, perturbations are subdivided into *shifts* – unlikely to revert back to initial state – and *disturbances* – finite, short duration changes that revert back to initial state. An "epoch" is defined as a fixed time period in which context and stakeholder's expectations don't change (Ross and Rhodes, 2008). Example epoch shifts could include changes in technology or regulations. Example disturbances include temporary weather events or short-term communication disruptions.

Step 2 is designed to provide a set of possible perturbations that will be used in later steps to motivate dynamic strategies (and ilities) for the SoS to maintain value delivery.

Some of the details of this step are presented in chapter 4, but a brief description of the main activities and outcomes involved in this step are presented here:

- Interview stakeholders and domain experts to get possible endogenous, exogenous and need-related uncertainties.
- Use enterprise boundary map and stakeholder value network (both generated in step 1) to brainstorm categories of uncertainty that can impact the SoS.
- Identify potential uncertainties surrounding stakeholder(s) attributes and utility ranges/weightings/value trees.
- Parameterize found uncertainty categories into perturbations, and brainstorm preliminary enumeration levels (e.g., uncertainty category "stress on SoS" is parameterized into perturbation "boat arrival rate", which in turn can have the following enumerated levels: [low; medium; high]).
- Finalize perturbation set, taking care to record fixed and assumed variables.
- Apply perturbation taxonomy (Mekdeci, 2012) to identified perturbations, according to possible categories and levels (more details about this provided in chapter 4). Perturbation taxonomy can assist in identifying the ways in which the system can fail to deliver value. The categorization of perturbation attributes helps architects design systems that prevent, mitigate and recover from perturbations (i.e. instill relevant ilities in systems).

3.2.3 Identify Initial Desired Ilities

This step enables the identification of the ilities desired in the SoS by the stakeholders (in order to promote the desired long-term behavior of the SoS). It is intended to help identify an initial list of ilities for explicit consideration during the architecting of the SoS. Parts of the step rely on the semantic basis for ilities, a tool that has emerged from recent research in the field (Ross et al., 2011; Beesemyer, 2012). This tool provides the means for associating a given ility to a specific definition, based on a set of differentiating categories (i.e., collectively defining semantic fields). The semantic basis is used in this step to identify (and differentiate between) stakeholders' desired ilities, as well as in step 8 to generate ility-based requirements. Some of the key activities performed in this step are:

- Gather directed and implied ility requests from stakeholders (e.g., "a survivable and flexible SoS").
- Trace perturbations to ilities: from the list of perturbations identified in step 2 and their taxonomy categorization, it is possible to infer relevant ilities. For example, if the perturbation space is described by many perturbations of the type "shift", then *robustness* is an ility of interest. If the perturbation space is dominated by needs-related perturbations, then *versatility* is a priority. Desiring to protect against

negative perturbations can lead to *survivability* and *robustness* (each with active and passive approaches). Desiring to take advantage of positive perturbations can lead to *changeability* and *evolvability*.

- The semantic basis tool can be used as an ilities statement generator to identify the desired ility behaviors from stakeholders' preliminary ility statements expressing non-value-driving desires. The semantic basis enables the classification of ilities over multiple different dimensions – among which are, for example, *agent* (internal, external or none), *parameter* (level or set), and *perturbation* (disturbance, shift, or none). Furthermore, it is important to recognize that there is no such thing as having an ility "in general," but rather with respect to particular aspects of the SoS. That is, one cannot be "survivable," but rather "survivable to disturbances X and Y." Likewise, one can be "scalable in parameter X from X_1 to X_2." The ilities statement generator, then, can be used to help generate or identify ilities based upon desired change (or change resistance) statements, which are in turn associated with the semantic categories. For example, a literal statement – such as: "In response to *perturbation*, require *increase* in coverage of Maritime Security SoS with *reaction* sooner than 7 minutes and *change span* shorter than 20 minutes" – is automatically linked to ilities like changeability, scalability, agility, and reactivity.
- Generate ility hierarchy maps (Beesemyer, 2012) of interest. The concept of ility hierarchy is tightly related with the categories of the semantic basis for ilities. The general idea is that, by prioritizing (i.e., ordering) different categories of the semantic basis, a hierarchy can be obtained. The hierarchy can be used to explain, trace and seed sets of ilities that may be of interest for the SoS under consideration. For example, if one cares about value sustainment in the face of finite duration impacts (i.e., disturbances), then one cares about survivability (as defined within fields of the semantic basis. Survivability can be achieved through numerous means, including reducing susceptibility, decreasing vulnerability, or increasing resilience. Increasing resilience can be achieved through adaptability and agility, which in turn can be achieved through modularity. The ilities hierarchy is intended to help structure and guide such considerations. An example of changeability hierarchy according to different filters (i.e., semantic fields) is provided in Figure 3-7.

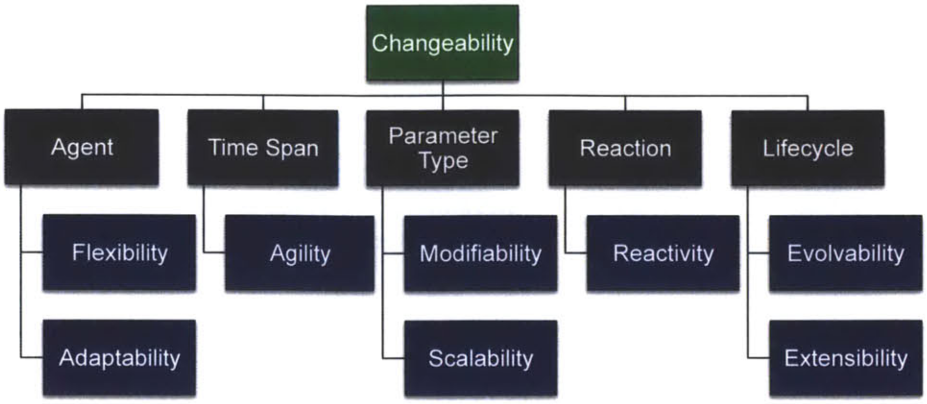


Figure 3-7: Example changeability hierarchy filtered by parameters such as agent, time span, etc.

After carrying out these activities, it is possible to finalize a list of potentially useful ilities, given mission needs and constraints. This list will be put forward into analysis and used to distinguish between architecture selections.

3.2.4 Generate Initial Architecture Alternatives

The goal of this step is to generate high-level concepts for SoS architectures, capable of delivering value despite utilizing significantly different means. It consists of brainstorming potential new constituent systems (form), as well as formulating various SoS concept-of-operations (CONOPs). The goal is to generate many possible SoS architectures, enumerating a preliminary SoS design space (see chapter 4) from the value propositions found in step 1 – and making sure to document all assumptions made. Key activities in this step are:

- Define high-level architecture concepts, given designated value proposition and constraints of step 1.
- Generate candidate SoS forms and CONOPs (Mekdeci et al., 2012).
- Conduct design-value mapping. The purpose of this task is to perform a qualitative assessment of the potential SoS concepts' fulfilment of stakeholders' needs, and it is performed by mapping from potential design trades to stakeholder value metrics (attributes). If some attributes are not affected by the current design space, or some design variables do not affect the value space at all, then the initial design space enumeration is revised (Ross et al., 2009).
- Develop an initial range for the levels of each design variable (e.g., number of patrol boats: [2; 4; 6]).
- Finalize initial design space, and record all assumptions made. An example of a first instantiation of the design space for the MarSec SoS is given in Table 3-2.

Table 3-2: Example of first instantiation of design space.

Form	Level	CONOPs	Level
Hermes UAV	[0-20]	Tech level upgrade	[Yes No]
Shadow UAV	[0-10]	Info sharing use	[Yes No]
Prop plane	[0-10]	Task assignment	[Dedicated Multi-role]
Helicopter	[0-4]	Geographic segmentation	[1 zone Multiple zones]
Manned patrol boat	[1-10]	Operators per UAV	[N:1 1:N]
Satellite relay	[Yes No]	Workforce buffer	[0% X%)
Land sensors	[Yes No]	Authority	[Central Distributed]

3.2.5 Generate Ility-driving Options

This step is concerned with the identification of relevant ility-driving elements that, when within the architecture, enable the system to generate contingent value (i.e., contingent upon the resolving of uncertainty), thereby resulting in desired ility properties. Two main types of elements are considered here: those that enable change (change options) and those that inhibit change (resistance properties). Chapter 5 will discuss these in more depth and a higher level of formality these two types of ility-driving elements.

The inputs to the step are the perturbation list of step 2, the initial desired ilities of step 3, and the architecture concepts delineated in step 4. This step is concerned with the generation, evaluation, and selection of relevant ility-driving elements to include in the SoS architecture. Although an in depth description and application of this step is provided in chapter 7, a brief summary of the activities and outcomes of this step is the following (in order):

- Conduct perturbation to architecture mapping to identify most “impactful” perturbations. This consists of tracing perturbations identified in step 2 to the design variables and attribute list to estimate which design variables and attributes are most impacted by perturbations.

- Select relevant design principles (Wasson, 2006). Each ility (from step 3) has a list of design principles associated with it (e.g., the design principle of margin is relevant for survivability considerations).

- Perform cause-effect mapping (Mekdeci, 2013). The cause-effect mapping technique is a way to trace out the cause and effect relationships between perturbations and the SoS, and can be used to identify links in causal chains that could be targeted for intervention by ility-driving elements. For example, a feedback loop between operator workload and operator error rate could be broken by an intervention through a change in CONOPs when threshold workloads are reached.

- Another way to generate potential change options and resistance properties is through design principle to perturbation mapping. For this task, the design principles related to ilities of interest are mapped to the list of perturbations that can potentially impact the SoS. The mapping consists of brainstorming instantiations of design principles that can inhibit or enable SoS changes, as a response to the perturbation. For example, the design principle of modularity can inspire the installation of modular payloads on the UAVs in the SoS, so that they can be swapped to accommodate different mission needs at a later point in time. Existing design variables may already instantiate design principles: these are *latent* ility-driving elements of the SoS architecture. For example, an SoS with distributed components already has latent instantiation of the survivability design principle of *distribution*.

- Generate formal list of candidate ility-driving elements. From what has been filled out in the design principle to perturbation matrix and the intervention points in cause-effect mapping, it is now possible to extract four lists containing path enablers, path inhibitors, change mechanism and resistance mechanism. These are what constitutes change options and resistance properties, respectively, and are discussed in further depth in chapter 5.

- Evaluate candidate ility-driving elements. After a comprehensive list of ility-driving elements to consider for inclusion in the SoS architecture has been generated, the next step is to evaluate and compare them so to select a final list. Examples of evaluation metrics are: optionability (Mikaelian, 2009), number of uses, cost (of different flavors), perturbation coverage (proxy for risk attenuation). These, too, are discussed in more depth in chapters 5 and 7.

- Finalize list of options. Given options evaluation above, a final list of options for consideration is assembled. Analytical and visualization tools can be used to facilitate this task.

3.2.6 Evaluate Potential Alternatives

The purpose of this step is to evaluate the various SoS architecture alternatives generated in step 4 in terms of different metrics, including value metrics (i.e. attributes and costs) and ility metrics. This step should be applicable at multiple levels of abstraction and fidelity: analysts could do detailed quantitative modelling and simulation, or architects could perform back of envelope evaluation with a few alternatives in mind. Some of the activities associated with this step are:

- Develop abstract architecture for SoS model, including important models: performance, cost, value, etc.
- Validate models (at least for a subset of the design space).
- Sample design space and epoch space using preferred Design of Experiment technique (Willcox, 2012).
- Evaluate performance (and value delivered) of architectures within each epoch (context and needs fixed).
- Given change mechanisms related to change options selected in step 5, generate transition matrices (generated by applying algorithmic logic that codifies the proposed change mechanisms – i.e., transition rule). This logic inspects a given alternative in the design-space and determines if it can transition to any other alternative in the design-space, given the criteria of each transition rule.

3.2.7 Analyze Architecture Alternatives

This step consists of the analysis of the generated data in step 6. Its purpose is to develop and understanding of (and insights on) trade-offs within various SoS architectures in terms of design and operations choices between static value and contingent value (ility behaviors). Some of the key activities are:

- Conduct single-epoch analyses. This can consist of generating utility (or whatever other measure of benefit) v. cost tradespace plots, conducting sensitivity analysis of results (with respect to design parameters or utility curve assumed), identifying drivers of utility v. cost trade-offs, calculating (fuzzy) Pareto efficient sets, as well as multi-stakeholder compromise solutions in any given epoch. For instance, Figure 3-8 shows the different impact (in value, i.e.: MAU v. Cost in this case) of using central vs. distributed authority in the MarSec case for two different stakeholders – one interested in the surveillance and interception mission, and the other more in the rescue type mission. For the former, central authority is a better choice across all designs; for the latter, it does not make a big difference.

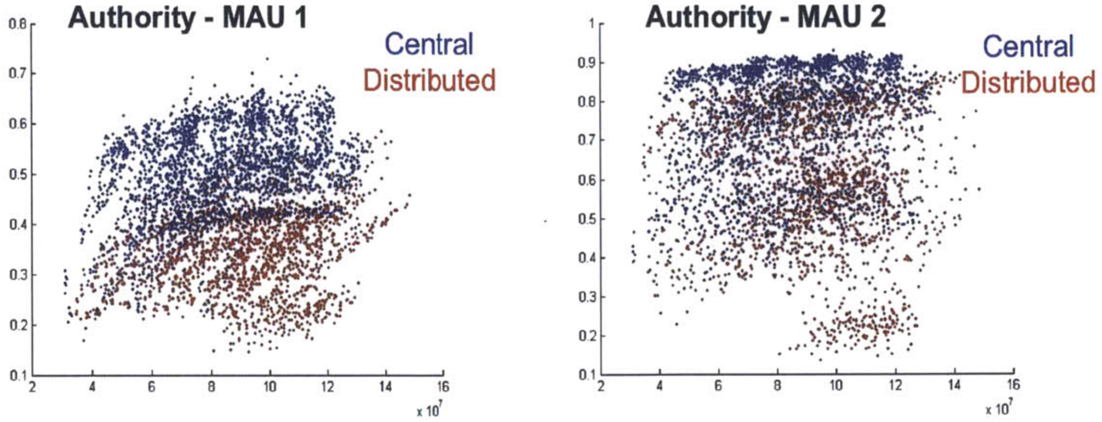


Figure 3-8: Using central authority vs. distributed has a larger impact for a stakeholder interested in surveillance than one interested in rescue missions.

- Propose change execution strategies, which are the logic behind how and when change mechanisms are executed. For example, a strategy could be "maximize utility" (i.e., execute change mechanisms whenever doing so would increase the utility of the alternative), or "maximize efficiency" (i.e., execute change mechanisms whenever doing so would move the alternative closer to the Pareto Front in that given epoch).

- Conduct Multi-Epoch Analysis (Fitzgerald, 2012). This activity is composed of three steps. First, ility-screening metrics of Filtered Outdegree and Normalized Pareto Trace should be calculated to identify potential alternatives of interest to focus on (Fitzgerald, 2012). (Filtered Outdegree is the number of transitions an architecture alternative can have, given filters in cost and time. Normalized Pareto Trace is the fraction of epochs in which a given alternative is Pareto efficient.) Then, select alternatives of interest based on these metrics. Finally, complete Multi-Epoch Analysis with a deeper analysis of the designs of interest, using techniques such as effective Normalized Pareto Trace, which evaluates the fraction of epochs in which a given alternative is Pareto efficient when it can change to other alternatives. This represents changeability-enabled value robustness, as opposed to passive value robustness.

- Conduct Era Analysis (Fitzgerald, 2012). This activity allows for the consideration of time-sequences of epochs. Epochs are selected and logically arranged in a sequence, with a duration applied to each epoch, resulting in a potential era for the SoS. Sequencing of epochs could take into account any likelihood knowledge of particular epochs, as well as logical constraints on progression of contexts (e.g. technology is not likely to get worse). Eras can be computationally generated or manually crafted using narrative-based approaches. For each constructed era, cumulative performance of alternative SoS can be evaluated, tracking long-term metrics like accumulated lifecycle utility and cost. These can be used to evaluate affordability of different SoS. Metrics such as frequency of change mechanism execution can be calculated using Epoch Syncopation Framework (Fulcoly, 2012). An example of Era analysis applied to the MarSec SoS is given in (Ricci et al., 2013).

- Collect set of alternatives of interest with ility metrics (some of which are shown in Table 3-3). This final sub-step collates all of the ility metrics calculated, grouped by ilities of interest specified in the earlier steps. Alternatives that perform well in the ility metrics can be identified to be traded with alternatives that perform well in other metrics, such as cost or utility. Alternatives with and without change options can be collected at this point as well, in order to directly compare the impact of including these options.

Table 3-3: Some ility metrics used in step 7.

Ility	Metric	Stands for
Robustness	NPT	Normalized Pareto Trace
	fNPT	Fuzzy Normalized Pareto Trace
Changeability	eNPT	Effective Normalized Pareto Trace
	efNPT	Effective Fuzzy Normalized Pareto Trace
	FPS	Fuzzy Pareto Shift
	FOD	Filtered Outdegree
Survivability	TAUL	Time-weighted Average Utility Loss
Affordability[12]	-	Accumulated Utility vs. Discounted Cost

3.2.8 Trade-off and Select "Best" Architecture with Ilities

The final step of the process involves backing out from the deep analysis of the previous step and using it to make decisions about the final selection of architecture and design. The designs evaluated in Step 7 are grouped by architecture and used to distinguish the ility performance of the different architecture concepts. When the "best" architecture is selected, the ility-driving elements of Step 5 that were not included in the tradespace are evaluated as potential extensions of the chosen architecture in order to

[12] New approaches and methods for considering and quantifying affordability in system design have been developed simultaneously to this research effort (Schaffner et al., 2014; Wu et al., 2014). Such efforts have been carried out by MIT graduate students Marcus S. Wu and Michael A. Schaffner, and are expected to be published in their full form (as theses) at the same time as this thesis (Wu, 2014; Schaffner, 2014).

improve its ility performance. Finally, the selection is documented and justified and either the process is repeated using the gained insight to improve the analysis or the architecture is used to generate requirements that will promote the desired ility behavior. An example of ility-based requirement produced with the aid of the ility statement generator is: "*In response to asset unavailability, it is required that spare vehicles should be on hand to maintain the level of the 'number of vehicles' in the form of the design, in order to maintain the same benefits with a reaction time of under 1 week.*" This statement is related to achieving robustness in performance through spares – which enable maintaining the number of vehicles as constant.

3.3 Summary

This chapter has introduced the reader to the activities involved in the various steps and sub-steps of the SAI method. It is important to note that the purpose of this chapter was not to present the detailed analyses involved in the various steps of the SAI method, but rather to expose the reader to the general overview of the method, and how it is specifically targeted at enabling SoS design with ilities. The SAI method builds upon the Responsive Systems Comparison method (Ross et al., 2009) by adding more explicit steps (e.g., step 5 and step 8) and analytical tools geared toward designing SoS with ilities.

One of the strengths of such an overarching method is that it is scalable in effort. Architecting with ilities rests on the achievement of a few core milestones along the conceptual design process: eliciting stakeholder values; identifying potential value-disrupting perturbations; listing ilities of interest; generating candidate ility-driving elements; and evaluating and selecting preferred SoS architectures with ility-driving elements. The last task is usually the most effort-intensive due to the evaluations involved. However, it is possible to perform such evaluations (corresponding to tasks in steps 5 and 7 in the overall SAI method) at multiple levels of abstraction and fidelity: from detailed quantitative modelling and simulation, to back of the envelope evaluation with a few alternatives.

The remainder of this thesis focuses on two of these milestones. Chapter 4 discusses the modeling of perturbations and the identification thereof. Chapter 5 introduces the concept of change option and resistance property in a formal way. Chapter 6 provides empirical examples of change options, and chapter 7 introduces a structured approach for the identification of ility-driving elements.

4 Formal Modeling of Uncertainty via Perturbations

"We abhor uncertainty, even when it is an irreducible part of the problem we are trying to solve."

– Nate Silver (2012)

This chapter provides a more in depth investigation of the concepts and activities involved in Step 3 of the SAI method introduced in the previous chapter. It is concerned with the conceptualization and modeling of uncertainty.

In chapter 2, the notion of designing complex systems has been discussed. Of particular relevance for the purposes of this chapter is the idea that, "since the consequences of design lie in the future, it would seem that forecasting is an unavoidable part of every design process." (Simon, 1996) However, the forecasting activity is inherently characterized with irreducible uncertainty (i.e., the state in which more than one outcome is consistent with reasonable expectations). It has also been discussed how Pielke (2011) points out that "expectations are a result of judgment, ... [and] As such, uncertainty is not some feature of the natural world waiting to be revealed but is instead a fundamental characteristic of how human perceptions and understandings shape expectations." Although there exist closed systems for which uncertainty can be directly quantified via statistical analysis, most engineering systems are inherently open and it is very difficult to understand uncertainty in these instances. In these cases, the most one can do is exploring the behavior of the system under a variety of different (probable) circumstances. This chapter formalizes a way of doing this.

4.1 Modeling in the Design of Artificial Systems

Models are essential tools in system design and are used by analysts and engineers throughout the design process. Engineers designing complex systems of the future are often forced to use complicated models and simulations in order "to explore...system performance without actually producing and testing each candidate system" (Blanchard and Fabrycky, 2006). These models and simulations have embedded in them causal and functional relationships, as well as empirical data from the past, which enable the synthesis of new data describing how the system is going to perform. In these cases, the data generated is *artificial* (synthetic) – i.e., it is not obtained by direct measurement of system properties, since no identical (or even similar) system may yet exist. As a result, the artificial data (as well as the model) "cannot be classified as accurate or inaccurate in any absolute sense" (Blanchard and Fabrycky, 2006). Thus, artificial data stands in stark contrast to empirical data (e.g., temperature readings, historical stock prices), which is directly measured and thereby holds a potentially higher degree of validity (i.e., reliability with respect to the relevant components).

Bankes (1993) distinguishes between two broad classes of model use: consolidative and exploratory. *Consolidative modeling* consists of constructing a model via a consolidation of known[13] facts into a single "box", which is then used as a surrogate of the actual system of interest. Such an approach is often used in the modeling of natural phenomena for which there is a significant knowledge of the physical facts that underlie them (e.g., Computational Fluid Dynamic model). When there is insufficient knowledge or the uncertainties can't be resolved, a surrogate model that closely reflects the behavior of the system can't be built. Under these circumstances, what is left to the modeler is exploring how the system would behave if certain assumptions held. *Exploratory modeling* is the use of series of "computational experiments" to explore the implications of a variety of assumptions and hypotheses. Such an approach is often used in the modelling of decision-making under uncertainty (e.g., Monte Carlo simulations) or systems for which there is no(t) (enough) data (e.g., behavior of subatomic particles at very high energies). In his work, Bankes (1993) argues that exploratory modeling is well suited for policy studies, but that oftentimes such exploratory nature is not recognized, leading to questionable strategies for the development and use of policy models. Similarly, Simon (1989) points out that, when building or using models for the design of complex systems of the future, one must be careful to realize that it is practically impossible to predict the behavior of such systems to the point of blindly relying on what the models says. Rather, models can extend the limits of human bounded rationality, and aid toward the practical concern of identifying plausible courses of action to be undertaken in the present, in order to bring about a certain desired future. His intent is to show that since the world and its systems are many orders of magnitude more complex than can be captured by a model, modelers must separate the 'essential' from the 'dispensable' in order to best capture a 'simplified view of reality' that can still enable useful extrapolations that will be helpful in achieving the (policy-related) aims of the modeling process. Evidently, this paradigm is based on the assumption that partial information and analysis is better than none.

In *The Sciences of the Artificial*, Simon (1996) discusses the great difficulties of designing complex artificial systems. He describes how there are three principal components in the design of a system: (1) the system itself, whose design is controlled (the "inner environment"); (2) the environment in which the system operates, which is exogenous to the control of designers (the "outer environment"); and (3) what is desired of the system, i.e.: a set of objectives, which are also subject to change. From here on,

[13] Known with reasonable confidence, at least. Popper (1967) argues that knowledge is in continuous evolution, and only tentative theories of the "world" can exist.

these three components are referred to as: the design space, the context space, and the needs space, respectively. It is important to note how designers hold control only over one of these three spaces (the design space), and must make their decisions about the designed system so that the system is able to continue to deliver value across changing environments and needs. These three key spaces form the backbone of the analyses performed at the Systems Engineering Advancement Research Initiative (SEAri) laboratory.

4.1.1 The Design Space and MATE

Conceptual design of complex systems involves a design-performance-value loop (Figure 4-1) – referred to as the "design loop." First, a design space of alternatives is generated based on a preliminary understanding of stakeholder needs. Each design alternative is mapped onto a performance space, spanned by attributes of interest quantified by a constructed performance model. The performance space is then mapped onto a value space via a value model (e.g., functional requirements, utility functions), through which the stakeholder evaluates the attractiveness of the alternatives. At this point, with an understanding of how each design alternative scores according to stakeholders' values, it is possible to either make a decision on what design (or set of designs) to focus on, or to go back and change the initial design space and repeat the loop. It is important to note that, in the design of the systems of the future, the nature of the data produced by performance and value models is artificial, and hence can't be directly validated against empirical data.

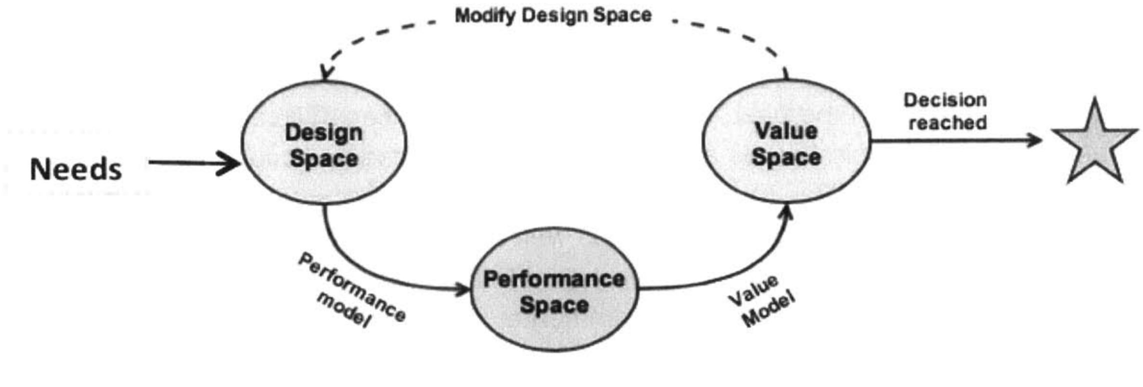


Figure 4-1: The design loop (Ricci et al., 2014).

Multi-Attribute Tradespace Exploration (MATE) (Ross, 2003; Ross et al., 2004) is a structured approach aimed at delineating and evaluating the design space. In MATE, a set of design alternatives of interest forms the design space, which is then evaluated into the performance space and, finally, the value space. MATE starts with the identification of stakeholder needs. From a list of elicited objectives, it is possible to derive quantifiable (sometimes proxy) attributes (Keeney and Raiffa, 1976), as well as stakeholder preferences over them (e.g., Single Attribute Utility curves, often used in MATE). A mapping of function (i.e. functional objectives) to form (often performed using design-value mapping matrix or other QFD methods) determines the initial design variables of interest, as well as the ranges (if ratio or cardinal scale) or levels (if discrete or nominal scale) they can take. This process is driven by engineering expertise and experience. At this point, and once the attributes that characterize the value space are defined, it is possible to enumerate the design variables and the levels that are going to be evaluated by software models and simulations (i.e., the performance model in Figure 4-1). It is important to note here that the enumerated design variables and levels form a set of design instances: *the design space*. The next step in MATE is to map the performance space into the value space. This consists of computing multi-attribute utility (MAU) for each design, given its attributes scores. A "valuable" design alternative is one that is efficient in the cost-MAU tradeoff. Usually, an efficient Pareto frontier (Pareto, 1906) of designs forms in the tradespace, wherein it is possible to select efficient design alternatives (i.e., designs for which it is impossible to improve the fulfillment of one objective without compromising the other). The last step in MATE is to use the actual decision makers for final evaluation and selection of a design (rather than their proxy preference functions).

4.1.2 Epoch-Era Analysis and the Context and Needs Spaces

Missing from the characterization of the design process in the previous section are the effects of changing context and needs on the decision made. In fact, in the diagram in Figure 4-1, there is no account of the outer environment and its variations over time. Traditional system design takes the approach of optimizing the design in terms of given objective functions, assuming a static context. However, this approach is valid only if the assumptions embedded in the analysis continue to hold true over time. Assuming a static outer environment has proven to be valid for the design of some complex engineering systems in the past. However, in other instances, failures of thinking deeply about possible realizable futures and ways to adapt to these has caused major letdowns (e.g., the Iridium Satellite constellation). Hence, in today's world, characterized by perpetual and often unexpected change, as well as limited budgetary and programmatic resources, failing to consider evolving environments is a luxury few decision makers (be them private organizations or environmental agencies) can afford. Traditional approaches to systems engineering are therefore not ideal when trying to design systems able to sustain value delivery over time (given changing contexts and needs). Exploratory models described in Bankes (1993) can be useful in order to explore the uncertainty space and gain familiarity with features of the system that make it robust or flexible to changing operational environments.

It is in light of these considerations that Ross and Rhodes (2008) discuss the difference between static and dynamic system perspective, where the former is associated with traditional systems engineering. Ross (2006) introduces Epoch-Era Analysis (EEA) – "an approach for conceptualizing system timelines using natural value centric timescales, wherein the context and expectations define the timescales" – as an enabler of the latter perspective. In EEA, an *epoch* is a discrete time period with fixed context and needs (i.e., value expectations). A sequence of different epochs forms an *era*, which can be used to model the full lifecycle of a system. A change in epoch occurs when there is a change in either the context or the needs (or both). Figure 4-2 illustrates the performance trajectory of a system within the unfolding of an era, as context and needs change. With EEA, tradespace exploration can be performed for a variety of different outer environments (i.e., epochs), or when new possible ones are recognized or anticipated. This aspect augments traditional systems engineering methods by enabling the exploration of not only the design space, but also the context space and the needs space – e.g. Beesemyer, Ross and Rhodes (2012). Finally, it is also important to note that, in the epoch-based paradigm, no absolute best design exists, and the identification of good designs is dependent on decision makers' preferred strategies across the epoch space or for eras of interest. Possible strategies may be maximizing benefit (e.g., utility), minimize cost, maximize cost-benefit efficiency, or minimize risk.

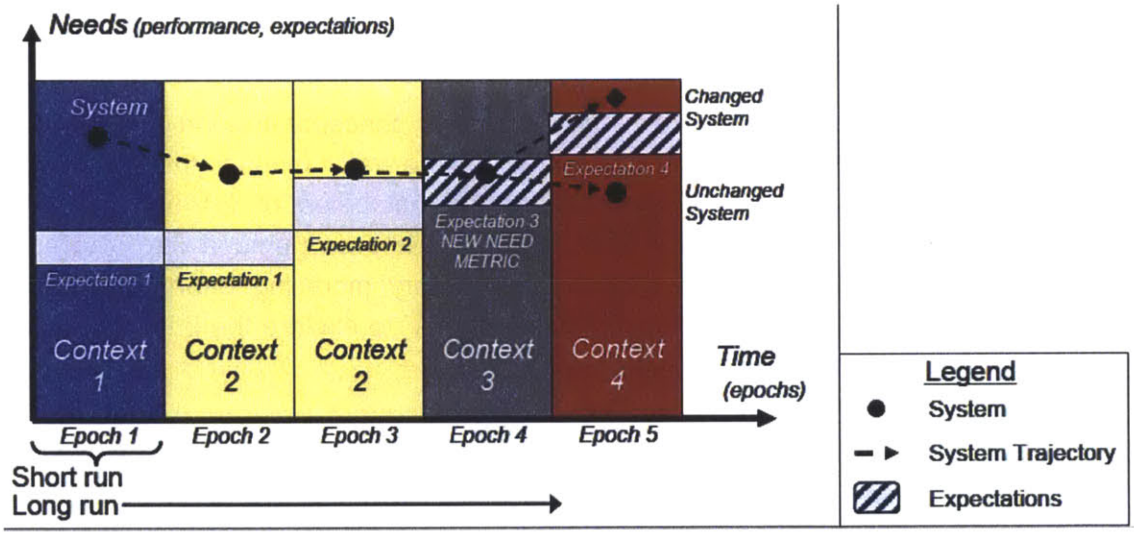


Figure 4-2: Trajectory of system performance (black dot) over an era, a succession of epochs (Ross and Rhodes, 2008).

4.2 Design under the Dynamic System Perspective

In much the same way a set of design variables and respective levels forms a set of designs – the design space, a set of epoch variables and respective levels forms a set of epochs – the epoch space. The set of epochs can be further subdivided into two sets: the context space (context variables and respective levels) and the needs space (needs-related variables and respective levels).[14] These three spaces form the backbone of the analysis performed in the design effort under dynamic system perspective.

[14] It is important to note here that, for the purposes of this thesis and the modeling efforts introduced in it, these two sets are considered mutually exclusive. However, it is likely that coupling exists between variables in the two sets. Incorporating this idea in the modeling framework proposed here could be an interesting area of future research.

It is now possible to enrich the classical design diagram shown in Figure 4-1 with the addition of the context and needs spaces (see Figure 4-3). The context space is related to the operational environment of the system (e.g., the meteorological conditions, or a regulation constraining emissions). The needs space is related to what is expected of the system (e.g., preferences over speed or range associated with an aircraft). Building on the work in (Ross, 2006), Beesemyer (2012) describes how design, context, and performance spaces concern what is "real," while the needs and value spaces concern what is "perceived." Spaces of the "real" type are grounded in reality and inherently carry with them a higher degree of objectivity. They describe potential actual systems or circumstances the systems may encounter, as well as measurable (or computable) performance metrics. Spaces of the "perceived" type are inherently more subjective as they reflect decision makers' perception of value accumulated from the performance of a system. The design space and context space are inputs to the performance model, which maps them into a performance space. The performance space and the needs space – the space of possible expectations of the system – are in turn inputs to the value model (e.g., utility functions), which maps them into the value space.

Two layers of exploration hence evince from this conceptualization of the design endeavor: (1) an exploration of the design space and the way it maps to the performance and value spaces, and (2) an exploration of the outcomes of (1) under different assumptions about the outer environment and the objectives pursued (i.e., the epoch space). The second directly relates to the exploratory modeling effort described in Bankes (1993): a series of "computational experiments" to explore the implications of a variety of assumptions and hypotheses.

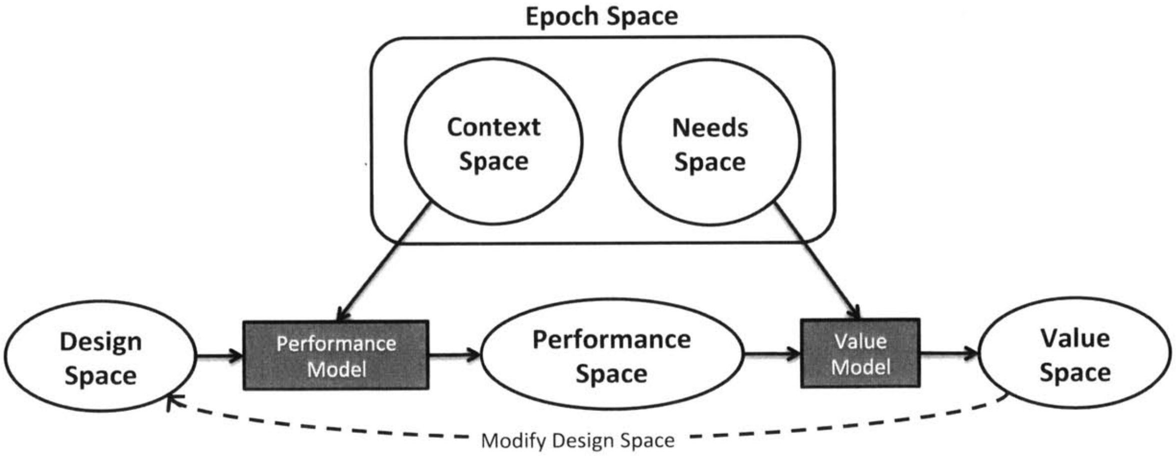


Figure 4-3: Design loop under dynamic system perspective.

4.2.1 Set-Theoretic Description of the Spaces

In the epoch-based tradespace exploration paradigm, the design space, context space, and needs space in Figure 4-3 are sets of design instances, context instances, and needs instances, respectively. Each instance can be described by a set of design, context, or needs variables at specific levels. During the course of the analysis, these spaces can be modified in order to reflect new information or insights. Each instance in the epoch space is an assumption about the system's outer environment under which the value trades associated with each design instance are explored. The design space and context space are inputs to the performance model, which, for any given context, evaluates the performance of each design instance with respect to the attributes of interest (among which can be cost). Each instance in the performance space is a set of quantified attribute indicating performance of each design-context pair. The performance space and the needs space – the space of possible expectations of the system – are in turn inputs to the value model (e.g., utility functions). Each instance in the value space is a representation of the value (e.g., MAU-cost pair) for each performance-needs pair (i.e., design-context-needs triad). As discussed in section 2.2.1, value here is what is perceived from the stakeholder as reflected in the value model. Not only can the given preference embedded in a value model change (e.g., multi-attribute utility functions in MAUT, discount rate in NPV), but it may also be the case that the value model changes as a whole (e.g., prospect theory vs. MAUT vs. NPV).

As mentioned in previous sections, the initial steps in MATE concern the elicitation of an initial design space of interest (using design-value mapping). Various techniques (some of which are presented in later sections of this chapter) also exist to elicit an initial epoch space of interest. These spaces are characterized by design and epoch variables, and their associated discrete or categorical levels. In the following paragraphs, a mathematical representation of these spaces is provided.

Design Space. Given the set of N design variables:

$$\boldsymbol{DV} = \{(dv_1, \dots, dv_i, \dots, dv_N) \mid i \in \mathbb{Z}, 1 \leq i \leq N\}$$

Each of which having a set of categorical or discrete levels:

$$\boldsymbol{l_i} = \left\{\left(l_{i_1}, \dots, l_{i_j}, \dots, l_{i_{M_i}}\right) \mid j \in \mathbb{Z}, 1 \leq j \leq M_i\right\}$$

Where M_i is the number of levels for a given design variable dv_i. It is possible to define a design space:

$$\boldsymbol{D} = \{(d_1, \dots, d_k, \dots, d_T) \mid k \in \mathbb{Z}, 1 \leq k \leq T\}$$

Where each design instance d_i is a vector with the design variables on specific levels. For example, in a space with five design variables, a design instance could be: $d_{153} = \{dv_1^3, dv_2^5, dv_3^1, dv_4^{12}, dv_5^5\}$, where the superscript indicates the level of the design variable. The cardinality of the design space is:

$$|\boldsymbol{D}| = T = \prod_{i=1}^{N} M_i = \prod_{i=1}^{N} |\boldsymbol{l_i}|$$

Which is: the Cartesian product of the number of levels of all the design variables.

For example, the description of a simple design space for a space tug system (Fitzgerald, 2012) can be given by three design variables:

$$\boldsymbol{DV} = \{manipulator\ mass, propulsion\ type, fuel\ load\}$$

Where each design variable has a discrete number of levels, e.g.:

$$manipulator\ mass = \{low, medium, high\}$$

$$propulsion\ type = \{bipropellant, cryogenic, electric, nuclear\}$$

$$fuel\ load = \{low, medium, high\}$$

The cardinality of this space is: $|\boldsymbol{D}| = 3\times4\times3 = 36$ possible designs.

Context Space. Given the set of N context variables:

$$\boldsymbol{CV} = \{(cv_1, \dots, cv_i, \dots, cv_N) \mid i \in \mathbb{Z}, 1 \leq i \leq N\}$$

Each of which having a set of categorical or discrete levels:

$$\boldsymbol{l}_i = \left\{\left(l_{i_1}, \dots, l_{i_j}, \dots, l_{i_{M_i}}\right) \mid j \in \mathbb{Z}, 1 \leq j \leq M_i\right\}$$

Where M_i is the number of levels for a given context variable cv_i. It is possible to define a context space:

$$\boldsymbol{C} = \{(c_1, \dots, c_k, \dots, c_T) \mid k \in \mathbb{Z}, 1 \leq k \leq T\}$$

Where each context instance c_i is a vector with the context variables on specific levels. For example, in a space with five context variables, a context instance could be: $c_{153} = \{cv_1^3, cv_2^5, cv_3^1, cv_4^{12}, cv_5^5\}$, where the superscript indicates the level of the context variable. The cardinality of the context space is:

$$|\boldsymbol{C}| = T = \prod_{i=1}^{N} M_i = \prod_{i=1}^{N} |\boldsymbol{l}_i|$$

Which is: the Cartesian product of the number of levels of all the context variables.

For example, the description of a simple context space for a space tug system (Fitzgerald, 2012) can be given by two context variables:

$$\boldsymbol{CV} = \{technology\ readiness\ level, market\ demand\}$$

Where each context variable has a discrete number of levels, e.g.:

$$technology\ readiness\ level = \{low, medium, high\}$$

$$market\ demand = \{low, medium, high\}$$

The cardinality of this space is: $|\boldsymbol{C}| = 3\times3 = 9$ possible contexts.

Needs Space. Given the set of N needs variables:

$$\boldsymbol{NV} = \{(nv_1, \dots, nv_i, \dots, nv_N) \mid i \in \mathbb{Z}, 1 \leq i \leq N\}$$

Each of which having a set of categorical or discrete levels:

$$\boldsymbol{l}_i = \left\{\left(l_{i_1}, \dots, l_{i_j}, \dots, l_{i_{M_i}}\right) \mid j \in \mathbb{Z}, 1 \leq j \leq M_i\right\}$$

Where M_i is the number of levels for a given needs variable nv_i. It is possible to define a needs space:

$$\boldsymbol{N} = \{(n_1, \dots, n_k, \dots, n_T) \mid k \in \mathbb{Z}, 1 \leq k \leq T\}$$

Where each needs instance n_i is a vector with the needs variables on specific levels. For example, in a space with five needs variables, a needs instance could be: $n_{153} = \{dn_1^3, nv_2^5, nv_3^1, nv_4^{12}, nv_5^5\}$, where the superscript indicates the level of the needs variable. The cardinality of the needs space is:

$$|\boldsymbol{N}| = T = \prod_{i=1}^{N} M_i = \prod_{i=1}^{N} |\boldsymbol{l}_i|$$

Which is: the Cartesian product of the number of levels of all the needs variables.

For example, the description of a simple needs space for a space tug system (Fitzgerald, 2012) can be given by one variable:

$$\boldsymbol{NV} = \{mission\ type\}$$

Where its levels are:

$$mission\ type = \{GEO\ satellites\ rescue, in\ orbit\ refueling, garbage\ collection\}$$

The cardinality of this space is: $|\boldsymbol{N}| = 3$ possible needs.

Epoch Space. Given the context space and the needs space, the set of epoch variables is:

$$\boldsymbol{EV} = \boldsymbol{CV} \cup \boldsymbol{NV} = \{ev \mid ev \in \boldsymbol{CV}\ or\ ev \in \boldsymbol{NV}\}$$

The cardinality of this set is:

$$|\boldsymbol{EV}| = |\boldsymbol{CV}| + |\boldsymbol{NV}|$$

The epoch space can be obtained through the Cartesian product of the context space and the needs space, and can be defined as:

$$\boldsymbol{E} = \{(e_1, \dots, e, \dots, e_T) \mid k \in \mathbb{Z}, 1 \leq k \leq T\}$$

Where each epoch instance e_i is a vector with the epoch variables on specific levels. For example, in a space with five epoch variables, an epoch instance could be: $e_{153} = \{ev_1^3, ev_2^5, de_3^1, ev_4^{12}, ev_5^5\}$, where the second subscript indicates the level of the epoch variable. The cardinality of the epoch space is given by:

$$|\boldsymbol{E}| = T = |\boldsymbol{C} \times \boldsymbol{N}| = |\boldsymbol{C}| \cdot |\boldsymbol{N}|$$

Which is: the product of the cardinalities of the context space and needs space.

For example, for the space tug case described above, the epoch variables are:

$$\boldsymbol{EV} = \boldsymbol{CV} \cup \boldsymbol{NV} = \{technology\ readiness\ level, market\ demand, mission\ type\}$$

With their respective levels listed above. The cardinality of the epoch space then is: $|\boldsymbol{E}| = 3 \times 3 \times 3 = 27$ possible epochs.

Performance Space. The performance space is evaluated through the performance model ($f\colon \boldsymbol{D}, \boldsymbol{C} \rightarrow \boldsymbol{P}$), which takes the design space and context space as inputs. Given the set of N attributes of interest:

$$\boldsymbol{A} = \{(a_1, \dots, a_i, \dots, a_N) \mid i \in \mathbb{Z}, 1 \leq i \leq N\}$$

(Each of which evaluated on a continuous or discrete scale and informing the decision maker about the degree to which an objective is fulfilled.) It is possible to define a performance space:

$$\boldsymbol{P} = \{(p_1, \dots, p_k, \dots, p_T) \mid k \in \mathbb{Z}, 1 \leq k \leq T\}$$

Where each performance instance p_i is a vector with the evaluated attributes associated to a specific design instance in a specific context instance. For example, in a space with three attributes of interest, a performance instance could be: $p_{153} = \left(a_1^{0.83}, a_2^{low}, a_3^{1237}\right)$, where the superscript indicates the score of attribute a_i in the given design instance d and context instance c resulting in p_{153}. The cardinality of the performance space is:

$$|\boldsymbol{P}| = T = |\boldsymbol{D} \times \boldsymbol{C}| = |\boldsymbol{D}| \cdot |\boldsymbol{C}|$$

Which is: the product of the cardinalities of the design space and context space.

It is important to note that, although each performance instance p_i contains the evaluation of all attributes envisioned by the needs space (e.g., for a car: speed, traction coefficient, comfort level), it is possible that only a subset of these attributes ($\boldsymbol{A_1} \subseteq \boldsymbol{A}$) is used in a given epoch, for a specific set of needs (e.g., only traction coefficient is used in the value space if driving in an epoch for which one is confined within the boundaries of a small and snowy mountain town).

Value Space. The value space is evaluated through the value model ($f\colon \boldsymbol{P}, \boldsymbol{N} \rightarrow \boldsymbol{V}$), which takes the performance space and needs space as inputs. Although a variety of different value models can be used[15], MATE typically uses a particular type of Cost-Benefit analysis, wherein Multi-Attribute Utility (MAU) represents benefit and (sometimes-discounted) monetary expense represents Cost. It is now possible to define a value space:

$$V = \{(v_1, \dots, v_k, \dots, v) \mid k \in \mathbb{Z}, 1 \leq k \leq T\}$$

Where each value instance v_i is a two-dimensional vector containing a MAU value (non-ratio cardinal scale – i.e., interval scale) and a Cost value (ratio scale) associated to a specific design instance in a specific epoch instance. For example, a value instance could be: $v_{1532} = \left(MAU^{0.67}, Cost^{\$2.3B}\right)$, where the superscript indicates the evaluated MAU and Cost instances for the given design-epoch pair (i.e., performance-needs pair or design-context-needs triad). The cardinality of the value space is:

$$|\boldsymbol{V}| = T = |\boldsymbol{D}{\times}\boldsymbol{E}| = |\boldsymbol{D}| \cdot |\boldsymbol{E}| = |\boldsymbol{D}{\times}\boldsymbol{C}{\times}\boldsymbol{N}| = |\boldsymbol{D}| \cdot |\boldsymbol{C}| \cdot |\boldsymbol{N}|$$

Which is: the product of the cardinalities of the design space and epoch space (i.e., the product of the cardinalities of the design, context and needs spaces). For example, for the space tug system described above, the cardinality of the value space is: $|\boldsymbol{V}| = |\boldsymbol{D}| \cdot |\mathbf{C}| \cdot |\mathbf{N}| = 36{\times}9{\times}3 = 927$.

Hence, design, context, and needs instances can be represented as elements in their respective sets (spaces), which are then mapped into the performance and value instances in the performance and value sets (spaces). Such representation is shown in Figure 4-4. It is important to note that instances within a space are often connected, as there is the possibility of (intentional or forced) transitions from one instance to another. In this way, networks of design, context, and needs instances form in their respective spaces. The conceptualizations of uncertainty and response to it introduced in later sections and chapters are linked to this idea of transitions within or outside such networks. In particular, two important ways in which these instances may be connected are discussed (i.e., perturbations and change mechanisms).

[15] Ross et al. (2010) perform a vast literature search on the different types of value models: e.g., Net Present Value (NPV), Multi-Attribute Utility Theory (MAUT), Cost-Benefit Analysis (CBA), Cumulative Prospect Theory (CPT), Value Functions (VFs), Analytic Hierarchy Process (AHP), and Technique for Order Preference by Similarity to Ideal Solution (TOPSIS).

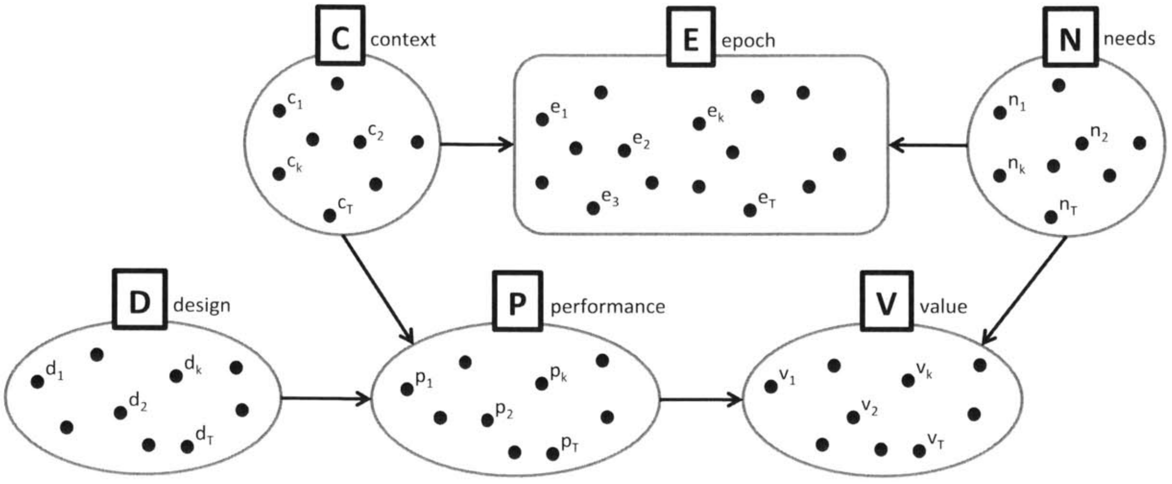


Figure 4-4: Representation of the different spaces in the dynamic system design perspective. Arrows symbolize the fact that relationships (e.g., the epoch space is given by the Cartesian product of context and needs spaces, P = f (D, C)) exist among spaces.

4.3 Conceptualization of Uncertainty via Perturbations

Ultimately, a stakeholder makes a decision based on the perceived value delivered by a specific design instance. The value space, however, is not only a function of the design space – controllable by designers, but also of the context space and the needs space. Hence, changes in any of these three spaces may cause changes in the value perceived by a stakeholder. Which is:

1. Unexpected or imposed changes in the design instance cause the performance (i.e., performance instance) of the system to change (e.g., car's windshield breaks), thereby changing the value delivered.
2. Changes in the context (i.e., context instance) in which the system operates cause the performance (i.e., performance instance) of the system to change (e.g., car goes from asphalt to dirt terrain), thereby changing the value delivered.
3. The expectations (i.e., needs instance) of the system change (e.g., need car to be more fuel efficient, as new job requires longer commutes), thereby changing the value delivered.

The *uncertainty* an engineer has to consider during the design process then can be conceptualized as variations in context and needs instances, as well as imposed or spontaneous variations in the design instance.[16] The evaluated performance and value spaces will automatically reflect these variations. Despite the existence of uncertainty, it

[16] Uncertainty can lie also in the assumptions behind the performance and the value model. While assumptions behind the performance model can be captured to a certain extent in the context space (e.g., modeling varying technological levels), it is very difficult to represent uncertainty associated with assumptions in the value model. In fact, this has to do with the existence of cognitive biases as well as explanation- and model-related biases (Ricci et al., 2014).

is of utmost importance to the stakeholder that the system keeps delivering a satisfactory level of value over time. (Beesemyer, 2012) describes in detail the concept of value sustainment. He discusses how, although it is important for a system to provide some level of value to the stakeholder in the present day, the real "interest of the stakeholder is that these systems *keep* providing value throughout their expected lifetime." Chapter 5 and 7 of this thesis introduce concepts and methods geared toward the achievement of value sustainment.

4.3.1 On the Concept of Perturbation

Within the context of SEAri methods and the spaces discussed in sections above, Mekdeci (2012) defines a perturbation as any "unintended state change in a system's form, operations, or context which could jeopardize value delivery." The system's form and operations (CONOPs) are a description of the design instance (Mekdeci, 2012), while the context is an instance in the context space. Beesemyer (2012) enriches the definition of perturbation by including changes in needs (i.e., from one needs instance to another). This way, he expands the concept of perturbation from the real space to the perceived one.

Hence, given the above discussion about the three spaces (design, context, and needs) that an engineer is faced with during the design and operations of a system, it is possible to describe a perturbation as an operator within these spaces. A perturbation operates on a given instance in a space, taking it from its current state to a different one. This new, different, state can be either within the enumerated space (i.e., set) or outside of it.

For a system in operations, given current design, context or needs instances, a perturbation is a change in one of these instances. As Mekdeci (2012) points out, such change is "*unintended.*" If the current design instance is d_i, a perturbation[17] λ^D can be defined as:

$$\lambda^D : \begin{cases} d_i \mapsto d_j \mid d_i, d_j \in \boldsymbol{D} \\ d_i \mapsto d_i' \mid d_i \in \boldsymbol{D}, d_i' \in \boldsymbol{D}' \end{cases}$$

Which is, an operator that transitions the current design instance d_i within the space $\boldsymbol{D}$ into either another design instance d_j within $\boldsymbol{D}$, or a design instance d_i' that does not exist in the current design space $\boldsymbol{D}$.[18] The two types of perturbations are illustrated in Figure 4-5. The new instance d_i' belongs to $\boldsymbol{D}'$, a slight variant of the design space $\boldsymbol{D}$ that includes d_i' (such that $\boldsymbol{D} \subseteq \boldsymbol{D}'$). For example, a UAV with a damaged wing is a

[17] The use of "$\boldsymbol{\lambda}$" to indicate perturbations is inspired by Perturbation Theory in quantum mechanics (Schrödinger, 1926). However, no meaningful relation exists between the two types of perturbation.

[18] In the modeling effort, the latter case can correspond to (1) a change to a level that was not modeled (for an existing design variable), or (2) to the expansion of the current space to include a new design variable altogether.

design instance not initially considered in the design space; however, the system may find itself in this design instance due to, perhaps, a physical attack.

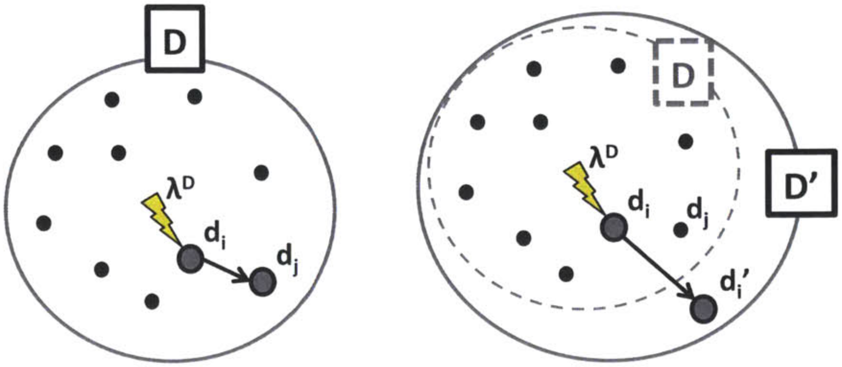


Figure 4-5: A perturbation shifting a design from an instance in the space *D* to another (left); and a perturbation shifting a design from an instance in the space *D* to a new one, outside of it (right).

Similarly, if the current context instance is c_i, a perturbation in context λ^C can be defined as:

$$\lambda^C : \begin{cases} c_i \mapsto c_j \mid c_i, c_j \in \boldsymbol{C} \\ c_i \mapsto c_i' \mid c_i \in \boldsymbol{C}, c_i' \in \boldsymbol{C}' \end{cases}$$

Which is, an operator that transitions the current context instance c_i within the space $\boldsymbol{C}$ into either another context instance c_j within $\boldsymbol{C}$, or a context instance c_i' that does not exist in the current context space $\boldsymbol{C}$. The two types of perturbations are illustrated in Figure 4-6. The new instance c_i' belongs to $\boldsymbol{C}'$, a slight variant of the context space $\boldsymbol{C}$ that includes c_i' (such that $\boldsymbol{C} \subseteq \boldsymbol{C}'$). For example, for a Maritime Security SoS, a context variable may be the percentage of pirates going through the area of interest. Although initially two levels, 2% and 5%, are used for exploration, it may happen that a level of 10% suddenly (or gradually) becomes plausible over time. If this were the case, it would signal a change in the initially considered context space.

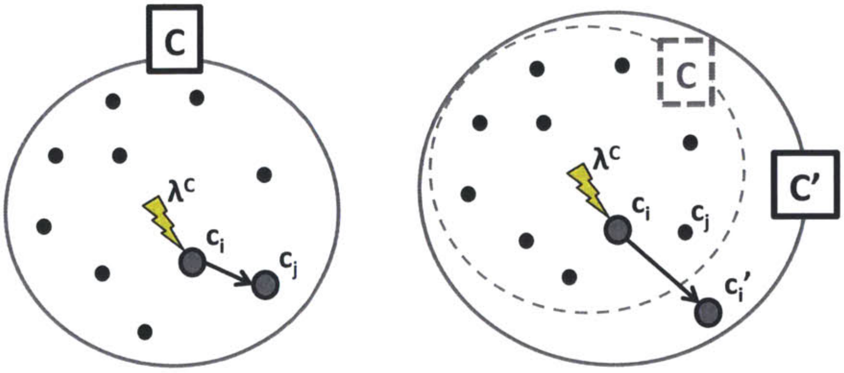


Figure 4-6: A perturbation shifting a context from an instance in the space *C* to another (left); and a perturbation shifting a context from an instance in the space *C* to a new one, outside of it (right).

If the current needs instance is n_i, a perturbation in needs λ^N can be defined as:

$$\lambda^N: \begin{cases} n_i \mapsto n_j \mid n_i, n_j \in \boldsymbol{N} \\ n_i \mapsto n_i' \mid n_i \in \boldsymbol{N}, n_i' \in \boldsymbol{N}' \end{cases}$$

Which is, an operator that transitions the current needs instance n_i within the space $\boldsymbol{N}$ into either another needs instance n_j within $\boldsymbol{N}$, or a needs instance n_i' that does not exist in the current needs space $\boldsymbol{N}$. The two types of perturbations are illustrated in Figure 4-7. The new instance n_i' belongs to $\boldsymbol{N}'$, a slight variant of the needs space $\boldsymbol{N}$ that includes n_i'. For example, in the case of a personal computer (PC), new and higher expectations with regard to graphics capabilities may arise that were not conceived when the PC was initially assembled. In this case, the initially conceived needs space changes.

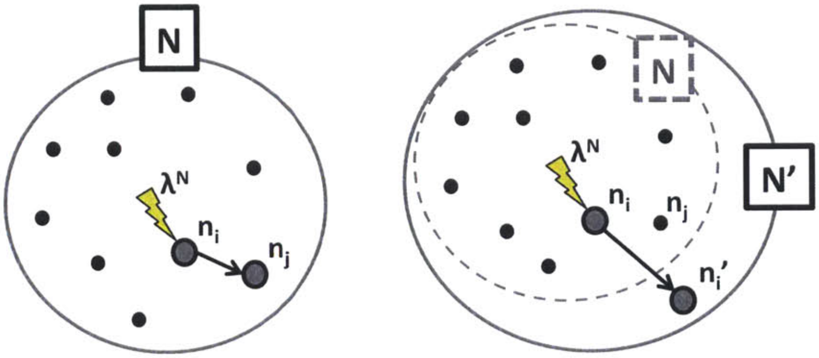


Figure 4-7: A perturbation shifting a set of needs from an instance in the space *N* to another (left); and a perturbation shifting a set of needs from an instance in the space *N* to a new one, outside of it (right).

The admission of $\boldsymbol{C}'$ and $\boldsymbol{N}'$ is representative of the fact that the modeling of the relevant context space and needs space is subject to change over time, as new information arises. In fact, not only is it possible for a context (or needs set) to change from an instance in the (presently envisioned) space to another, but also for the whole context (or needs) space to change, so to include new possible instances.[19] This consideration is particularly relevant for systems whose operations are monitored over time, and that allow for continuous redesign (e.g., the Ballistic Missile Defense System, which is discussed in chapter 6). Since this thesis is targeted mainly at the initial design effort, less attention will be devoted to $\boldsymbol{C}'$ and $\boldsymbol{N}'$ in the concepts and frameworks presented in the forthcoming chapters.[20]

Finally, it is important to note that, whenever there is a perturbation in context (λ^C) or needs (λ^N), it corresponds to a perturbation in the epoch: λ^E. When thinking strategically about the design of a system, it is convenient to represent perturbations as either changes in the level of epoch (i.e., context or needs) variables, imposed changes in the

[19] This parallels the idea of known unknowns and unknown unknowns (see chapter 2): a new context variable (or level in a context variable) can be the revelation of a previous unknown unknown (or of something that was deemed irrelevant, but it wasn't).
[20] This may be an interesting area for future research.

level of design variables, or imposed changes of a design into a new, non-enumerated one. This representation of perturbations is shown in Figure 4-8. It evinces from this diagram and the above discussion that two principal types of perturbation exist: (1) to the design of the system, and (2) to the epoch in which the system operates. These perturbations ultimately result in variations in value delivery of the current design instance. This is perceived in the value space (either directly, or indirectly through changes in the performance instance). Section 4.3.2, among other descriptive fields, discusses the impact of perturbations on the perceived value space.

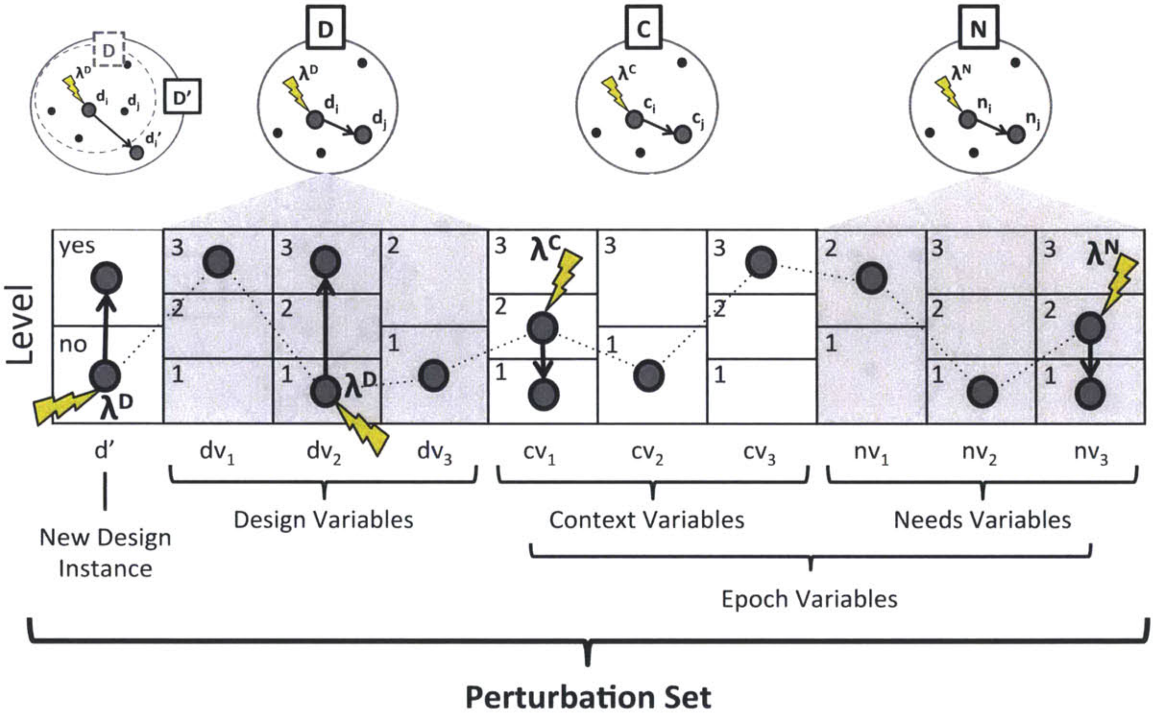


Figure 4-8: Variable-centric representation of perturbation set.

4.3.2 Descriptive Fields for Perturbations

Mekdeci (2012) first attempted at delineating a taxonomy for perturbations, in order to help organizing and categorizing them. A taxonomy for perturbations can assist in identifying the ways in which the system can fail to deliver value. The categorization of perturbation characteristics (i.e., descriptive fields) helps architects design systems that can prevent, mitigate and recover from perturbations (i.e. instill survivability, robustness, and other relevant ilities in systems). Beesemyer (2012) elaborated and augmented this taxonomy, as part of his work in the development of a semantic basis for ilities (Beesemyer et al., 2011). In the following paragraphs, some salient descriptive fields introduced in the past are presented, as well as some new ones.

Duration. A fundamental descriptive feature of a perturbation is its duration. Relative to the lifecycle of the system, a perturbation can be either finite (short-term) or irreversible (long-term). This difference lies at the root of two fundamentally different types of perturbation: disturbance and shift. Mekdeci (2012) defines a *disturbance* as "an unintended, finite duration, continuous state change of a system's form, operations, or context [i.e., changes in design or context instance] that could jeopardize value delivery". Beesemyer (2012) expands upon this definition allowing disturbances to affect not only the design or context, but also the needs. In this way, disturbances can affect both the real space and the perceived space. Ross and Rhodes (2008) define epoch shifts as "shifts in context and/or needs [instances]" which are unlikely to revert back to the initial instance. Similar to disturbances, the definition of *shift* can be expanded to include not only shifts in context or needs instances, but also imposed (or spontaneous) shifts in design (to either another enumerated design instance in $\boldsymbol{D}$ or a new, non-enumerated one in $\boldsymbol{D}'$). In the case of the Maritime Security SoS, a context disturbance can be a temporary communication jam, while a context shift a change in the emissions regulation; a design shift can be the permanent loss of a UAV. Lastly, it is important to note that there are perturbations that can be modeled as both a disturbance and a shift; e.g., change in the volume of incoming pirates in the area of interest. Whether to model it as either is a judgment of the modeler (together with the decision maker, perhaps), and it depends on the type of exploratory analysis he (they) set out to do. For instance, the epoch space in EEA uses only epoch shifts.

Space. Another important characteristic of a perturbation is the space that it affects, whether *design*, *context* or *needs*. This concept has been described at length in the pervious subsection of this chapter. An example of a perturbation affecting the design is the permanent loss of a UAV (shift); a perturbation affecting the context is the change in emissions regulation (shift); a perturbation in the needs set can be a change in the expectations of the system to have it perform search and rescue on a stranded boat (disturbance).

Origin. A perturbation can originate from inside or outside the system. The former kind is perturbations of *internal* origin, the latter of *external* origin. An example of internal perturbation can be the malfunctioning of the control system of a UAV; an external perturbation, once again, is the occurrence of a storm. Linked to perturbation's origin are the concepts of thread and hazard. A threat is an external set of conditions that may cause a perturbation, but has not impacted value delivery yet (e.g., a boats' abnormal and suspicious behavior). A hazard is an internal (inside a system) set of conditions that can cause a perturbation (e.g. flammable building materials).

Intent. This category refers to whether or not the perturbation is intentionally targeted at decreasing (or increasing) value delivery of the system; i.e., if an intelligent agent forced a transition in design instance or in epoch instance so to decrease (or increase) system value delivery. For example, an intentional perturbation in design is the loss of a UAV due to an enemy's missile strike; an unintentional perturbation in design is the decrease of communication among assets in an SoS due to bad weather.

Nature. The perturbation can be either natural or artificial: this informs its nature. If natural, then the cause of the perturbation is an event not human-generated. For example, a UAV is momentarily out of order due to a lightning strike (disturbance in design). If artificial, then the cause of the perturbation is the effect of a human action (that might perturb the system intentionally or unintentionally). For example, a UAV is lost due to an enemy missile strike.

Consequence. This field describes whether the consequence on system value delivery is positive or negative (or either depending on the reaction to the perturbation). For example, a missile strike taking out a UAV is a negative perturbation for the Maritime Security SoS; the decrease of gasoline price a positive one; lastly, the development of a new weapon technology can be positive or negative, depending on whether the weapon can be included in the operating SoS and whether enemies can use it as well. The consequence of the perturbation is tightly related with the concepts of opportunity (positive consequence) and risk (negative consequence). This will be discussed more in depth in chapter 7.

Effect. This is a field in which all the possible (conceivable) effects of a perturbation are listed. These may be effects in design, context, needs or performance that eventually translate to value loss or enhancement. For example, the loss of a UAV may cause a variety of effects: from an increase in stress on the SoS resulting in poorer performance, to distraction of human operators causing decreased detection and identification capabilities.

As it can be seen from Table 4-1, most of the descriptive fields have to do with either the causes or the effects of a perturbation. Mekdeci (2013) uses some concepts from the domains of Systems Safety (Leveson, 1995) and Systems Dynamics (Forrester, 1994; Sterman, 2000) to derive a cause-effect mapping method for perturbations, aimed at highlighting the dynamics of how perturbations in a particular system propagate. Trying to understand perturbations' causes and effects can be beneficial in terms of assessing perturbations' probability of occurrence, as well as their impact on value delivery. These two – probability of occurrence and impact on value delivery – are two very important descriptive fields for a perturbation. In fact, they play a key role when trying to decide what perturbation(s) the systems must be shielded from (or must exploit).

Table 4-1: Perturbations' descriptive fields and respective categorical levels.

Type	Space	Origin	Intentional	Nature	Consequence	Effect
Disturbance	Design	Internal	Yes	Natural	Positive	Various
Shift	Context	External	No	Artificial	Negative	
	Needs	Either	Either		Either	

Probability. The probability of occurrence describes how likely a perturbation is to strike within the lifecycle of a system. Although for some perturbations it is possible to obtain a meaningful quantitative assessment (using historical data or empirical observations), when dealing with the possible perturbations impacting a complex engineering system, it is much harder to obtain such data. For example, while it might be possible (after many observations) to generate a statistically meaningful number for the probability of a storm to occur, it is very hard to do so for the probability of the usage of magnetic levitation for transportation. In such instances, a qualitative assessment of the likelihood of a perturbation is all that is left to the designer. Such approaches are commonly used in the analysis of risk through risk matrices (National Aeronautics and Space Administration, 2012), as discussed in the next chapter.

One way of formalizing and improving the effectiveness of this task is using structured assessment techniques. One example of these is the Delphi method (Rowe and Wright, 1999). The Delphi method was developed during the initial phases of the cold war at the Rand Corporation, in order to attempt to forecast the impacts of technological advancement on warfare. Its goal is to obtain forecasts from a panel of independent experts. This may happen over two or more trials. The gist of the method is as follows: experts attempt to predict a quantity; after each round, a facilitator provides an anonymous summary of the experts' forecasts, as well as their reasons for them. When the variance of assessment between rounds is observed to have decreased significantly (or at least to a level that is deemed satisfactory by the group), the process is stopped. A key aspect of the Delphi method is the fact that it is anonymous and independently performed, thereby minimizing the effects of group dynamics and ego-driven behaviors.

Impact. The impact upon occurrence of a perturbation is related to the effects that it ultimately has on value delivery. Three ways of impacting value delivery exist within the framework developed so far[21]. First, a perturbation in design (to either an instance within or outside the initially enumerated design space):

$$\lambda^D : \begin{cases} d_i \mapsto d_j \mid d_i, d_j \in \boldsymbol{D} \\ d_i \mapsto d_i' \mid d_i \in \boldsymbol{D}, d_i' \in \boldsymbol{D'} \end{cases}$$

$$\Longrightarrow \begin{cases} p_i \mapsto p_j \mid p_i, p_j \in \boldsymbol{P} \\ p_i \mapsto p_i' \mid p_i \in \boldsymbol{P}, p_i' \in \boldsymbol{P'} \end{cases}$$

$$\Longrightarrow \begin{cases} v_i \mapsto v_j \mid v_i, v_j \in \boldsymbol{V} \\ v_i \mapsto v_i' \mid v_i \in \boldsymbol{V}, v_i' \in \boldsymbol{V'} \end{cases}$$

[21] As mentioned in previous sections, since this thesis is targeted mainly at the initial design effort, less attention is given to perturbations causing transitions to $\boldsymbol{C'}$ and $\boldsymbol{N'}$. However, instances within $\boldsymbol{D'}$ are considered.

Second, a perturbation in context (to either an instance within or outside the initially enumerated context space):

$$\lambda^C : \begin{cases} c_i \mapsto c_j \mid c_i, c_j \in \boldsymbol{C} \\ c_i \mapsto c_i' \mid c_i \in \boldsymbol{C}, c_i' \in \boldsymbol{C}' \end{cases}$$

$$\Longrightarrow \begin{cases} p_i \mapsto p_j \mid p_i, p_j \in \boldsymbol{P} \\ p_i \mapsto p_i' \mid p_i \in \boldsymbol{P}, p_i' \in \boldsymbol{P}' \end{cases}$$

$$\Longrightarrow \begin{cases} v_i \mapsto v_j \mid v_i, v_j \in \boldsymbol{V} \\ v_i \mapsto v_i' \mid v_i \in \boldsymbol{V}, v_i' \in \boldsymbol{V}' \end{cases}$$

Third, a perturbation in needs (to either an instance within or outside the initially enumerated needs space):

$$\lambda^N : \begin{cases} n_i \mapsto n_j \mid n_i, n_j \in \boldsymbol{N} \\ n_i \mapsto n_i' \mid n_i \in \boldsymbol{N}, n_i' \in \boldsymbol{N}' \end{cases}$$

$$\Longrightarrow \begin{cases} v_i \mapsto v_j \mid v_i, v_j \in \boldsymbol{V} \\ v_i \mapsto v_i' \mid v_i \in \boldsymbol{V}, v_i' \in \boldsymbol{V}' \end{cases}$$

These three ways of impacting value delivery are represented graphically in Figure 4-9, Figure 4-10, and Figure 4-11, respectively.

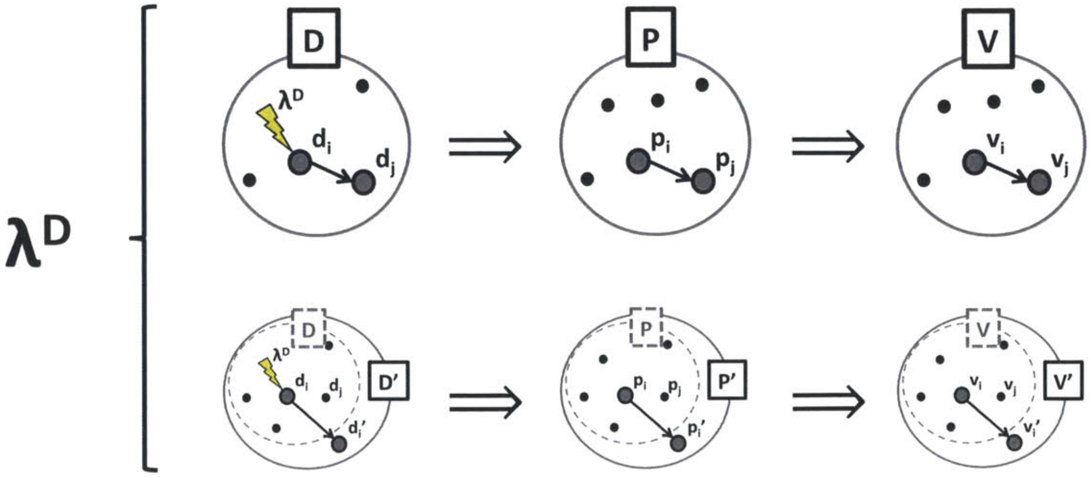


Figure 4-9: Perturbation in design impacts value delivery.

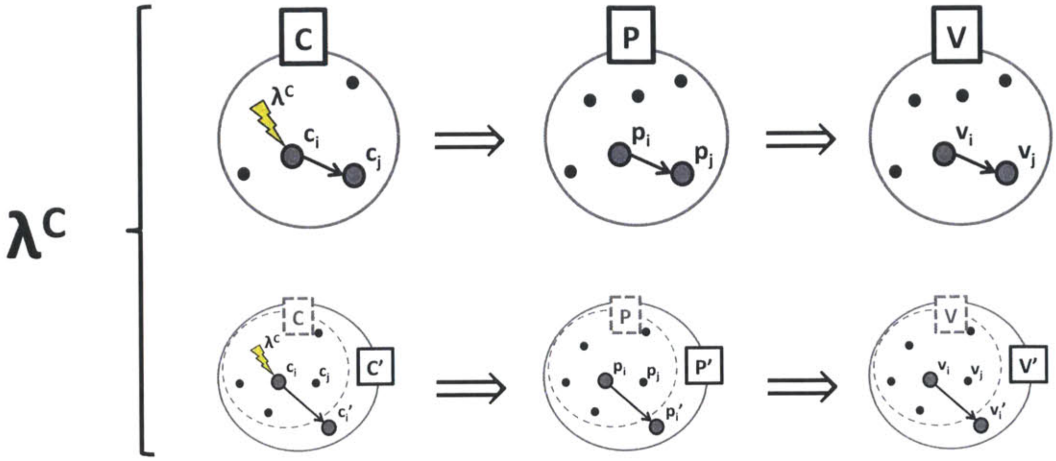


Figure 4-10: Perturbation in context impacts value delivery.

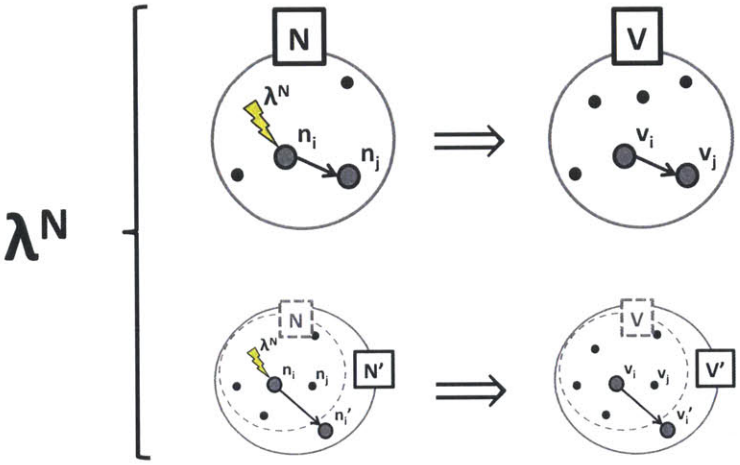


Figure 4-11: Perturbation in needs impacts value delivery.

In certain cases, the assessment of the impact of a perturbation on value delivery may be an easier task than the assessment of the probability of occurrence. For example, it may be possible to resort to modeling and simulation (based on physical laws or empirical evidence of previous instances) to approximate the impact of a perturbation that physically strikes the system. Similarly, it is possible to approximate the effects of the rise of gasoline price on the system's operations. In other cases, however, the assessment of impact is not any easier than that of probability of occurrence. For instance, it may be arduous to assess the impact of the rise of magnetically levitated cars on public transportation systems. In these cases, use of approaches like the Delphi method may be advisable.

4.3.3 Dynamic Impact of Perturbations

Richards (2009) defines survivability as "the ability of a system to minimize the impact of finite-duration environmental *disturbances* [emphasis added] on value delivery." He proposes a value-centric definition of survivability, and points out the fact that such a property is inherently dynamic – it emerges from the interaction of the system with its outer environment. He identifies three general design strategies for survivability: (I) susceptibility reduction, (II) vulnerability reduction, and (III) resilience enhancement. The first is related to the reduction of the likelihood of a disturbance to occur; the second to the minimization of the disturbance-induced losses on value delivery; and the third to the maximization of the recovery of value-delivery. Figure 4-12 shows these concepts within the value over time frame.

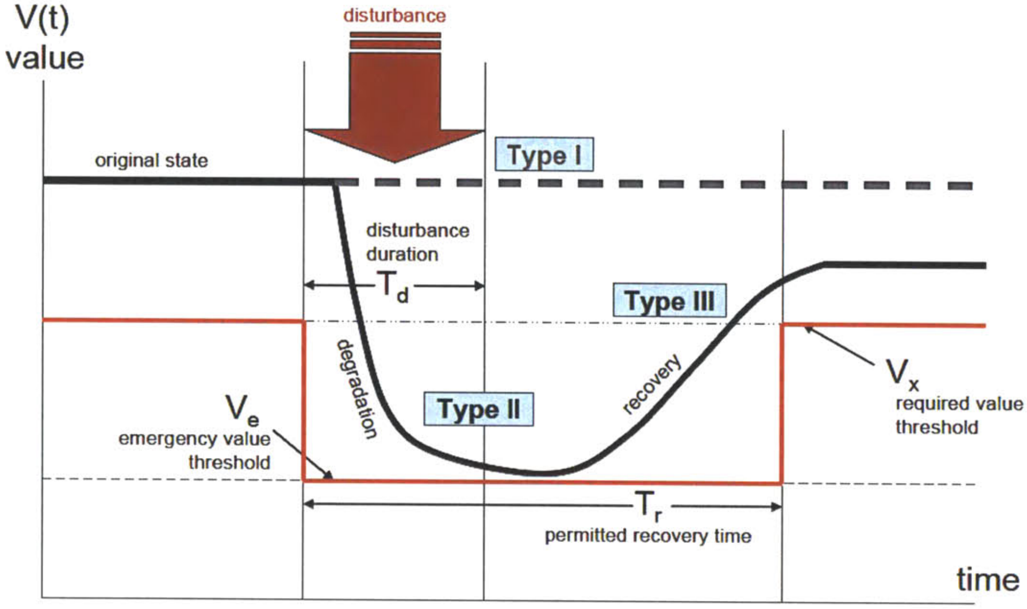


Figure 4-12: Value-centric disturbance lifecycle (Richards, 2009).

In a similar way, and using some of the concepts and notation developed so far, it is possible to derive a value-centric diagram for the dynamic impact of a perturbation – both shifts and disturbances (this is, an illustration of how a perturbation impacts value delivery over time). Figure 4-13 shows the impact of shifts, Figure 4-14 that of disturbances. It is important to note that, in Figure 4-13, only a shift in context or a shift in needs causes the epoch to shift, while a shift in design happens within the same epoch. Furthermore, in Figure 4-14, disturbances occur within the same epoch. Lastly, for these figures, it is possible to imagine a threshold for minimum acceptable value delivery, below which the system is in an unacceptable state from the perspective of stakeholders. Perturbations that are capable of bringing the value delivery of the system below this threshold are most impactful.

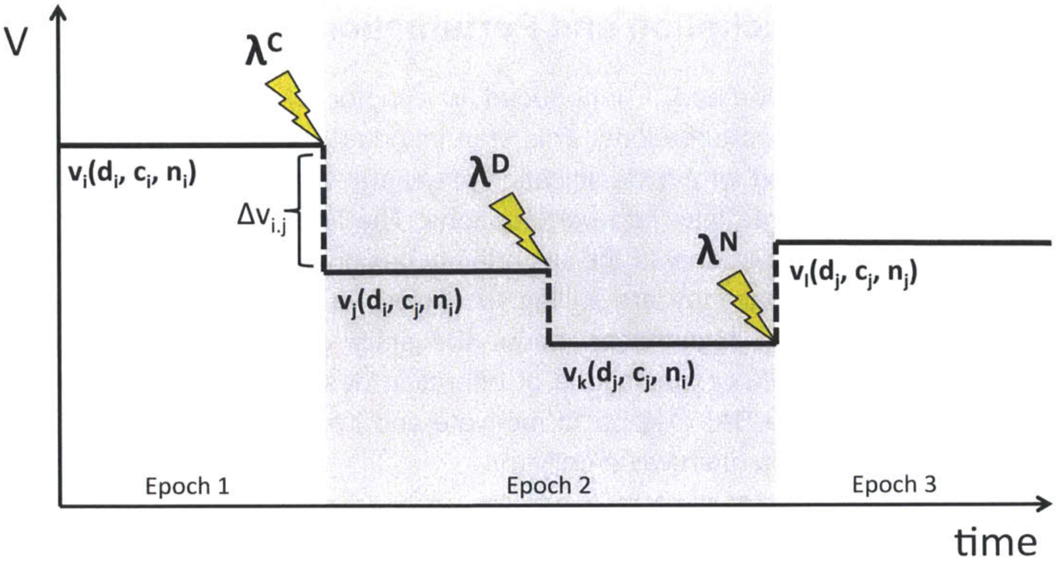


Figure 4-13: Dynamic, value-centric representation of shifts.

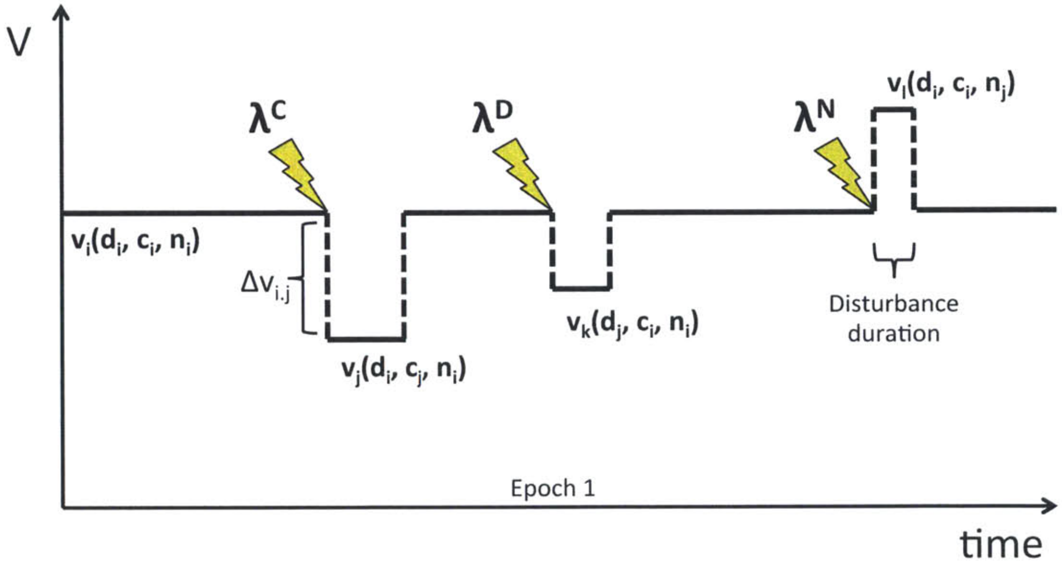


Figure 4-14: Dynamic, value-centric representation of disturbances.

In the diagrams in Figure 4-13 and Figure 4-14, susceptibility reduction has to do with decreasing the likelihood of a perturbation (λ^D, λ^C, or λ^N) to occur (e.g., using camouflage in order to not be attacked by enemies); vulnerability reduction with minimizing $\Delta v_{i,j}$ (e.g., using armor to absorb the damage of a physical attack); resilience enhancement with increasing the ability to quickly revert back to a preferred state in the face of a perturbation negatively impacting value delivery (e.g., having a rapidly deployable spare asset that can be put in use in case one is taken down by a physical attack).

4.4 Uncertainty Elicitation and Perturbations Parameterization

Step 2 of the SAI Method – introduced in chapter 3 – is concerned with the identification of potential perturbations. This step introduces an array of techniques and heuristics that can be used for the identification of general uncertainty categories, as well as the parameterization of these into perturbations. The final set of perturbations (both disturbances and shifts) represents the uncertainty designers decide to consider/model in their analysis: i.e., what they are willing to explore in terms of the implications of various assumptions and hypotheses about the state of (1) operating design, (2) operational context, and (3) expectations of operating design. These perturbations are used in later steps of the SAI method to motivate and inspire dynamic strategies (and ilities) for the system to maintain value delivery.

4.4.1 Eliciting Uncertainty Categories

In this subsection, a list of activities aimed at the elicitation of relevant uncertainty categories is presented (step 2 in the SAI method), alongside an illustrative application to the MarSec SoS case. Although the descriptions focus on SoS, it is possible to extend the applicability of these tasks to all system types. Scenario generation and identification of potential changes in context and needs is one of the most creative activities in the strategic design of a system. The design team may not have the detailed domain knowledge required to generate a list of all possible context scenarios. Thus, during the creative process that leads to the perturbation list, it is valuable to consult existing technical roadmaps, large-scale strategic plans and failure reports. Also, the input of program managers and decision makers is advised. The important details to investigate with regard to these uncertainties include: what is the uncertainty; how does it affect the value proposition(s); when does this effect occur; who controls this uncertainty; what would be a response to this uncertainty.

Identify endogenous uncertainties. This brainstorming activity is aimed at the generation of possible key system-related uncertainties. In the case of SoS, Mekdeci (2012) discusses how the line between system and environment can be blurred. The Enterprise Boundary analysis performed in Step 1 of the SAI Method can aid the identification of such endogenous uncertainties. Endogenous uncertainties come from what is within the Enterprise Boundary in Figure 3-4 (i.e., components and processes that are entirely under the control of program manager). An example list of endogenous uncertainties for the MarSec SoS is shown in Table 4-2.

Table 4-2: Example of brainstorming endogenous uncertainties for the MarSec SoS case.

Key SoS Uncertainty	Example
UAV Operator Mistake	Operator presses the wrong button and UAV performs wrong task (e.g., goes in the wrong direction)
Random Component Failure	Camera on UAV stops working and Identification cannot be performed anymore
Component Degradation	Receiver/Transmitter efficiency is reduced with time causing unwanted SoS behaviors
Miscommunication	Operators do not properly communicate to each other the current state of the system
Increase in Time per Task	Operators are tired and take longer to identify boats causing identification rate to go down
Collision between Vehicles	Pre-determined UAV routes intersect and UAVs collide
Unknown Error	UAV route changes for unknown reasons

Identify exogenous uncertainties. Exogenous uncertainties come from elements outside of the enterprise boundary in Figure 3-4. (i.e., components and processes that are not at all under the control of program manager). Once again, the enterprise scoping exercise can be useful when performing this task. In fact, the "clouds" identified in the boundary analysis can be considered exogenous uncertainty categories, which can be used to inform the types of epoch variables encountered by the system, as well as what imposed changes in design instance can occur upon the realization of this uncertainty. These span from weather conditions to the geo-political context in which the SoS may be operating. Further activities (discussed in 4.4.2) are targeted at drawing inspiration from these uncertainty categories, in order to parameterize uncertainty into a set of perturbations. Table 4-3 shows the different exogenous uncertainty categories identified for the MarSec SoS case.

Table 4-3: Exogenous uncertainty categories

Technology Level	Enemies	Economy and Market	Stress on SoS	Mission Needs	Funding	Weather	Political Context

Interview stakeholders to get future context uncertainties. Another way of gathering information about exogenous uncertainties is to interview SoS stakeholders. From the interviews, it is possible to generate a list of anticipated context uncertainties, both technical (e.g. changes in UAV technology) and non-technical (e.g. changes in political landscape leading to change in policies). An illustrative example of the outcome of this task is shown in Table 4-4.

Table 4-4: Example outcome of potential interviews with SoS stakeholders.

SoS Stakeholder	Future Context Uncertainty
SoS Office Program	- Perform a different type of identification routine - Must change the SoS architecture due to asset availability issues - Inclusion of new asset in existing SoS architecture - New mission requested by government
Port Authority	- Participation risk - Using constituent systems for different purposes
Collaborating Countries	- Enter war period - Change of political relationship - Using constituent systems for different purposes
Gov't Agency	- Less funding for project

Identify possible future context-related uncertainties. This is a narrative-based activity that consists of describing possible contexts in which the SoS might find itself in the future. SoS experts, in collaboration with SoS stakeholders, can perform this activity. An example outcome of this task is shown in Table 4-5.

Table 4-5: Scenario-based activity consisting of describing possible future contexts in which the MarSec SoS might find itself operating.

Context	Description
High stress – Low Tech	The stress on the system due to traffic volume, probability of enemies in the AOI, communication issues, etc. is very high in this scenario. However, the technology hasn't improved much and the performance of the system is the same.
Low stress – High Tech	In this scenario, the stress on the system hasn't changed, but the level of the technology available has improved. For example, a new UAV capable of performing detection in a much more efficient way has been included in the SoS. In this scenario the SoS is expected to perform better.
Increased Criminality	The number of smugglers and pirates attacks in the AOI has increased notably, but the system does not experience many communication issues and traffic volume is constant. In this case, the system has the same capabilities, but it has to deal with more criminals. It is not easy to predict whether it will do it successfully or not.
Economy Crisis	In this scenario, the whole geo-political area is experiencing an economic recession. The budget to maintain and operate the SoS is limited, and the SoS size and workforce size have to be reduced.

Identify potential needs-related uncertainties. In this task, uncertainties related to changes in needs for SoS Program Managers and stakeholders are identified. Changes in needs could require the SoS to perform tasks it was not intended to perform in the first place. Over time, strategic and high-level objectives of the SoS may shift. For example, when using Multi-Attribute Utility to quantify benefits, changes in needs can be expressed via changes in Single-Attribute Utility curves, as well as in the weighting of the single attributes. So, the goal of this task is to identify potential preference sets that could arise both in response to and independent of potential future contexts listed in the previous task. Table 4-6 shows a list of possible new needs that may arise for the MarSec SoS with a brief description.

Table 4-6: Example brainstorming of potential needs-related uncertainties.

New Need	Description
Random Identification of AOI Entities	At some point in the future, if the stress on the SoS is not high, the stakeholders might be interested in achieving a higher number of boats identified. This could be done by changing the identification routine and introducing some identification for random boats, on top of the suspects.
Search and Rescue	It might happen that a vehicle in the AOI is stranded and its owners are in need of help. In this case it would be important that the target is located by the SoS, and rescued by (perhaps) patrol boats.
Interception	The number of smugglers, pirates and possibly terroristic attacks in the AOI has increased notably, but the system does not experience many communication issues and traffic volume is constant. In this case, the system has the same capabilities, but it has to deal with more criminals. In order to suppress emergent criminality, the SoS might need to intercept hostile boats.
Low Cost	In this scenario, the whole geo-political area is experiencing an economic recession. The budget to maintain and operate the SoS is limited, and some of the SoS size and workforce size have to be reduced.

4.4.2 Parameterization of Uncertainty

As introduced in previous sections, when thinking strategically about the design of a system, it is convenient to represent perturbations (disturbance or shift) as either changes in the level of epoch (i.e., context or needs) variables, imposed changes in the level of design variables, or imposed changes of a design into a new, non-enumerated one (Figure 4-8). It is exactly these variables and variations thereof that are parameterized here from the uncertainty categories elicited above.

In this parameterization exercise, one considers all previously identified types of uncertainties and generates possible perturbations in (1) the design of the operating system, and (2) the epoch the system operates in. It is important to note here that this list of perturbation can quickly become very large. When, possible, then, it can be useful to group different parameters into one variable (especially if these variables are intended for use in computational models and simulations). Furthermore, once a final list is aggregated, it is possible to down-select perturbations of interest based on factors such as likelihood, impact, or ease of implementation in the modeling and simulation effort. Table 4-7 shows a possible way of parameterizing the uncertainty categories in Table 4-3. These span from weather conditions to the geo-political context in which the SoS may be operating. Further activities (discussed in 4.4.2.) are targeted at drawing inspiration from these uncertainty categories, in order to parameterize uncertainty into a set of perturbations. For each type of uncertainty category, three possible factors that may have a role within that category are listed. Then, they are ranked in terms of perceived relevance and described.

Table 4-7: Parameterization of uncertainty. From the relevant uncertainty categories, related factors are derived and ranked in terms of perceived relevance to the SoS.

Uncertainty Category	Possible Factor	Rank Within Category	Description
Technology Level	New UAV	1	A new UAV with enhanced capabilities is available
	Detection Methods	3	More reliable detection methods are developed
	Communication	2	Higher gain receivers are on market
Enemies	Smugglers Volume	1	Illegal activity near the area increases
	Pirate Attacks	2	For some reasons, pirates are more (or less) active
	Terrorist Attacks	3	Possibility of terroristic attacks to boats and/or SoS
Economy and Market	Alliance with other countries	3	Allows for beneficial deals with other countries
	Goods' price	1	Fuel price increase
	Workforce salary and availability	2	Reduction in workforce salary and size
Stress on SoS	Communication interruption	2	Lightning strike interrupts communication b/w UAV and ground station
	UAV out of order	3	Pirate attack brings UAV down
	Increased traffic volume	1	Boat arrival rate increases for a specific period

Uncertainty Category	Possible Factor	Rank Within Category	Description
Mission Needs	Intercept boats	2	Need to take down a dangerous enemy in the AOI
	Search & Rescue	1	Must be able to perform S&R in case of emergency
	Random Search	3	New identification policy
Funding	Budget cuts on research	2	Will not be able to investigate new technologies
	Budget cuts on Operations	1	Can not pay the current workforce and have to downsize
	No working overtime	3	Can not ask for extra hours in the case of intense activity periods
Weather	Lightening strike	2	Lighting strike put UAV out of service
	Tsunami	3	Tsunami causes damage to the whole SoS
	Storm	1	Storm reduces visibility and situational awareness
Political Context	War time	3	The AOI might become a military intense zone
	Conflict with bordering country	2	Might undermine the state of the operating SoS
	Environmental Policy	1	Must fly less UAVs

After having selected the final list of variables that are deemed relevant, the next step in the parameterization process is to brainstorm the possible discrete or categorical levels that these variables can have. As defined in previous sections, perturbations represent the realization of uncertainty, whereby a variable's state is pushed from one level to another (or to places outside the discretely enumerated space). Table 4-8 shows an illustrative final list of design and epoch variables that are subject to change upon the resolving of uncertainty for the MarSec SoS. Although not congruent with the formal definition of perturbation[22], this list is informally referred to as the *perturbation set*, considered in the modeling and analysis. The number of levels enumerated is arbitrary

[22] As defined earlier, a perturbation is the operator that changes the level of these variables.

and dependent on the objective behind performing it: e.g., for pure exploration of the uncertainty space or for modeling and simulation purposes. The levels in Table 4-8 have been derived with the underlying knowledge that they could have been used for a discrete-event simulation of the MarSec SoS.

Table 4-8: Example of a perturbation set for the MarSec SoS case.

Perturbation	Number of levels	Levels
Pirate percentage variation	3	L1 – 0% of entering boats L2 – 1% of entering boats L3 – 5% of entering boats
Boat arrival rate variation	2	L1 – 1/640 seconds L2 – 1/320 seconds
Comms jamming	2	L1 – Inactive L2 – Active
Smuggler percentage variation	2	L1 – 1% of entering boats L2 – 5% of entering boats
Workforce availability change	2	L1 – 100% L2 – 80%
Optical sensor TRL change	2	L1 – Low L2 – High
UAV loss	2	L1 – No L2 – Yes
Variation in situational awareness	2	L1 – Regular L2 – Limited
Heavy storm	2	L1 – No L2 – Yes
Perform rescue mission	3	L1 – No L2 – Yes (calm seas) L3 – Yes (rough seas)
Information attack	2	L1 – No L2 – Yes

After such parameterization activity, it is advisable to provide a narrative-based description of the perturbations, as well as define what they represent both to keep as a reference for future efforts (re-modeling or re-designing) and to enable precise description of the perturbations. Furthermore, it is important to perform a formal description of them using the perturbation taxonomy and descriptive fields introduced in earlier sections. This description (an example of which is given for the MarSec SoS in Table 4-9) enables a better understanding of the entirety of the uncertainty space considered, which is useful when trying to architect for ilities (as discussed in chapter 3).

Table 4-9: Perturbation taxonomy applied to perturbations listed in Table 4-8.

Perturbation	Type	Space	Origin	Intentional	Nature	Consequence	Effect
Pirate percentage variation	D/S	C	Ext	No	Art	Negative	Increase work
Boat arrival rate variation	D/S	C	Ext	No	Art	Negative	Stress on SoS
Comms jamming	D/S	C	Ext	Yes/No	Art	Negative	Hinder comms
Smuggler percentage variation	D/S	C	Ext	No	Art	Negative	Increase work
Workforce availability change	D/S	D	Either	No	Art	Negative	Degrade perform
Optical sensor TRL change	S	C/N	Ext	No	Art	Either	Number of UAVs
UAV loss	S	D	Either	Either	Art/Nat	Negative	Degrade perform
Variation in situational awareness	D	C	Either	Either	Art/Nat	Negative	Degrade perform
Heavy storm	D	C	Ext	No	Nat	Negative	Harsh environ.
Perform rescue mission	D	N	Ext	No	Art	Either	Change in needs
Information attack	D	C	Ext	Yes	Art	Negative	Hinder comms

4.5 Summary

This chapter introduced a discussion of the exploratory modeling effort in system design and its link to uncertainty characterization. A formal description of the spaces involved in system design (i.e., design, context, needs, performance and value spaces) has been provided, as well as one of the instances within them. Perturbations, then, have been defined as operators on three of these spaces (design, context or needs), and as a useful way of characterizing uncertainty in the dynamic system perspective for design. Finally, an application of uncertainty elicitation and its parameterization into perturbations has been discussed for the case of the MarSec SoS.

5 Formal Modeling of Responses to Uncertainty: Ility-Driving Elements

"... [La fortuna] dimostra la sua potenzia dove non è ordinata virtù a resisterle; e quivi volta li sua impeti dove la sa che non sono fatti gli argini e li ripari a tenerla."

– Niccolò Machiavelli (1532)

This chapter investigates in more depth the concepts underlying the activities in Step 5 of the SAI method introduced in chapter 3. Chapter 4 has introduced the concepts of exploratory modeling of uncertainty via perturbations. This chapter is concerned with the topic of designing systems that are able to prevent, deal with, or react to the unfolding of hypothesized uncertain events.

5.1 On Responding to Uncertainty

If the universe were in perennial stasis, design would lose much of its meaning. However, as discussed in chapter 4, a designer must cope with the unfolding of uncertain events. Simon (1996) considers this problem of designing artifacts that are immersed in unpredictable outer environments. He discusses "two complementary mechanisms for dealing with changes in the external environment, ... homeostatic mechanisms that make the system relatively *insensitive to the environment* [emphasis added] and retrospective feedback *adjustment* [emphasis added] to the environment's variation." "In one way or another, the designer insulates the inner system from the environment, so that an invariant relation is maintained between inner system and goal." This last statement implies a fixed goal. However, as discussed in chapters 2 and 4, in many complex systems, designers are confronted with the fact that even goals (expectations of the system) can change.[23] In these cases, the systems must be insensitive or adjust in response to environment or goal variations.

Expanding on Simon's ideas, there are two ways in which a system can cope with perturbations (as defined in chapter 4) that impact value delivery: either by being relatively insensitive to them, or by adjusting to them. In the former case, the system is *passively resisting* either the occurrence or the impact of a perturbation; in the latter

[23] This is the "wicked problem," a phrase first introduced in Churchman (1967) to describe a problem that is difficult to solve because of incomplete, contradictory, and changing requirements.

case, the system is *actively changing* to either reduce the likelihood of occurrence of a perturbation, mitigate the immediate impacts of a perturbation, or recover from the impacts of a perturbation.[24] It is important to note that a value-centric perspective is adopted here, whereby "impacts" on value delivered are considered (see 4.3.2 and 4.3.3). That is, although perturbations can occur in any of the three spaces discussed in chapter 4 (design, context, or needs), the designer is interested in these mechanisms insofar as they are targeted at value sustainment over time. These two mechanisms of dealing with perturbations constitute the backbone of what are here referred to as *ility-driving elements*: i.e., *resistance properties* and *change options*. The next two sections provide an overview of these two elements in systems design, while the ones that follow attempt at formalizing the concepts behind such elements within the context of the dynamic system perspective for design discussed in chapter 4. It'll be described how, in such context, these two types of mechanism can be viewed as operators on the spaces of design, just like perturbations. However, with the difference that these mechanisms are intentionally embedded in the system by the designer, and are aimed at enabling value sustainment over time.

5.2 Anatomy of a Change Option

One way for the system to react to (or influence the likelihood of) perturbations is to be able to change. Ross et al. (2008) introduce the concept of "change events as paths." They characterize change events with three elements: (1) the agent of change, (2) the mechanism of change, and (3) the effect of change. The agent of change is the instigator for the change; it can be internal and implied (related to adaptability) or external and intentional (related to flexibility). The mechanism of change describes the path taken to transition the system from the current state to the future state, including any costs, both temporal and monetary. The effect of change is the actual difference between the origin and destination states. In a value-centric approach, this is important because the two different states can correspond to two different levels of value delivery. Figure 5-1 shows these concepts related to a change event.

[24] The word *passive* refers to the fact that, unlike active adjustment, it requires no execution decision by an agent (internal or external to the system – e.g., an algorithm changing flight path in a control system, or an operator pushing a button to activate windshield wipers on car) in order to be insensitive to external variations. Hence, a multi-terrain tire is a passive mechanism – a resistance mechanism (see later section) – while the activation of windshield wipers is an active adjustment – a change mechanism (see later sections). It is important to note that this is a modeling assumption made in the context of this research, and it can be relaxed in future research and contexts. In fact, the definition of whether the system is changed or not by small adjustments is relative to what one considers the system to be. It relates closely to a long philosophical debate: "Theseus' paradox." This is a thought experiment that raises the question of whether an object that has had all its components replaced (the ship of Theseus in the original problem) remains fundamentally the same object. The standpoint taken in this thesis is that it doesn't: every time an adjustment is made, an intentional change by an agent has occurred (most likely to sustain value delivery).

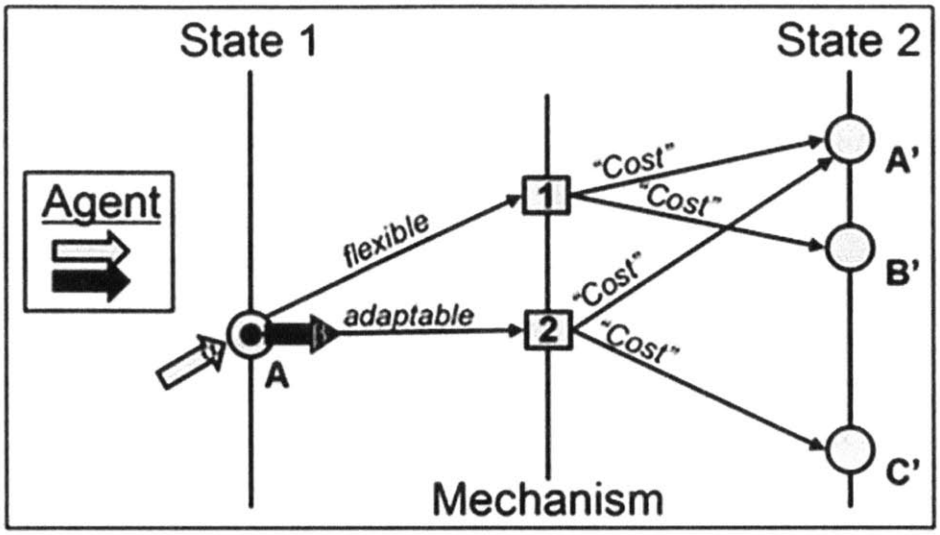


Figure 5-1: Agent-mechanism-effect representation of change event.

The ability of a system to undergo a change event at a future point gives designers a *change option*. Much like a real option, a change option gives the possibility, but not the obligation of changing. In fact, the execution of a change event is determined by either an internal or an external agent. Furthermore, similarly to the idea of real option *in* and *on* projects (discussed in chapter 2), a change option can be either a decision regarding the design of the engineered system or a strategic decision at the enterprise level. For perturbations with negative consequences on value delivery, a change option can be targeted at either reducing their likelihood of occurrence, mitigating their impact or recovering from their impact. For perturbations with potentially positive consequences on value delivery, a change option can be targeted at either increasing their likelihood of occurrence, or exploiting their potential positive impact on value delivery.

Abstractly, a change option is the union of two distinct concepts: change mechanism and path enabler. A change mechanism is the *method* through which a system goes from state A to state B (e.g., swapping payload on UAV); a path enabler (i.e., a physical object, an action or a decision) is *what* gives the *option* of executing the change mechanism (e.g., modular payload bay in original design). A path enabler can be either added to the baseline design of a system (a spare UAV), or latent in the baseline design instance (e.g., a UAV that already has a modular payload).

Options bring about contingent value, which materializes only upon the (imminent) occurrence of an event (e.g., perturbation). This enables the emergence of ilities over time. If one were only interested in "static" functional requirement satisfaction, options would not be considered during the system (and enterprise) design effort. Change options are associated with change-type ilities that involve a change agent: changeability, evolvability, flexibility, adaptability, etc.

5.3 Anatomy of a Resistance Property

In the same way systems can undergo "desired" and intentional change events that increase value delivery to stakeholders, they can undergo "undesired" and unintentional change events that disrupt value delivery. For example, the loss of a UAV due to a hostile missile attack in the MarSec SoS is an undesired change event. Another example can be the passing of a regulation that raises taxes on emissions to the extent that value is badly impacted. Being able to resist (by either prevent or mitigate the impacts of) such unintentional and undesired changes is a desirable property in systems.

The ability of a system to passively prevent or resist the impacts on value delivery of an unwanted change event (most perturbations described in chapter 4) at a future point in time is a *resistance property*. A resistance property enables the system to sustain its value delivery by either (1) reducing the likelihood of an unwanted perturbation or (2) passively mitigating (or recovering from) the negative effects of an unwanted perturbation.[25] Differently from a change option, a resistance property does not have an executive agent. However, similarly to change options, a resistance property can be either a decision regarding the design of the system or a strategic decision at the enterprise level.

Abstractly, a resistance property is the union of two distinct concepts: resistance mechanism and path inhibitor. A resistance mechanism is the *method* through which a system (or enterprise) resists (imposed) unintentional change from state A to state B (e.g., absorbing the hit of a physical attack); a path inhibitor (i.e., a physical object, an action or a decision) is *what* enables the resistance mechanism (e.g., armor).

Resistance properties bring about contingent value, which materializes only upon the (imminent) occurrence of an event (e.g., perturbation). This enables the emergence of passive-type ilities over time (e.g., passive value robustness). If one were only interested in "static" functional requirement satisfaction, resistance properties would not be considered during the system (and enterprise) design effort. Resistance properties are usually associated with resist-type ilities: survivability, robustness, etc.

5.4 Defining Ility-Driving Elements in the Design Effort

Both change options and resistance properties enable the emergence of lifecycle properties (active and passive, respectively). As such, in the context of the strategic design of a system and its enterprise, they are herein referred to as *ility-driving elements* (IDEs). This section is aimed at formally defining IDEs within the context of design under the dynamic system perspective introduced in chapter 4. In addition to the five spaces defined in chapter 4 (design, context, needs, performance, and value), two more spaces

[25] The word "passive" here is related to the fact that resistance property has no agent of execution and, hence, no execution decision. This assumption is discussed in further detail in later sections of this chapter.

are introduced here: the path enabler space and the path inhibitor space. Change mechanisms and resistance mechanisms, then, are operators that emerge in time (similar to perturbation), and that are enabled by the existence of path enablers and path inhibitors in the system, respectively.

5.4.1 Path Enablers and Path Inhibitors

Path enabler and path inhibitor spaces, just like the design space, are controllable by the designer. Unlike the design (or other spaces described), they don't form a network, but rather contain separate instances.[26] Furthermore, while only one instance of the design space is operationalized, it is possible to include more than one path enabler or inhibitor in the operating design of the system. It is the included set of path enablers or inhibitors that will result into change or resistance mechanisms at later points in time. In the following paragraphs, a mathematical representation of these new spaces is provided.

Path Enabler Space. Given N path enablers, it is possible to define a path enabler space:

$$\boldsymbol{PE} = \{(\varepsilon_1, \dots, \varepsilon_i, \dots, \varepsilon_N) \mid i \in \mathbb{Z}, 1 \leq i \leq N\}$$

Each path enabler ε_i represents *what* (perhaps partly) gives the *option* of executing a change mechanism. A basic distinction is made here between path enablers that are engineered *in* the system, and those that are *on* the system, at the strategic, enterprise level. An example of path enabler *in* the system is a modular payload bay in the original design instance. An example of path enabler *on* the system can be a strategic partnership with a supplier. Two mutually exclusive subsets then form from $\boldsymbol{PE}$. A subset of path enablers *in* the system:

$$\boldsymbol{PE}_{in} \subseteq \boldsymbol{PE}$$

And a subset of path enablers *on* the system:

$$\boldsymbol{PE}_{on} \subseteq \boldsymbol{PE} \mid \boldsymbol{PE}_{on} + \boldsymbol{PE}_{in} = \boldsymbol{PE}\,, \boldsymbol{PE}_{on} \cap \boldsymbol{PE}_{in} = \emptyset$$

A path enabler *in* the system can be either added to the baseline design of a system (a spare UAV), or latent in the baseline design instance (e.g., a UAV in the baseline design that already has a modular payload). Hence, it is possible to define a set of latent path enablers:

$$\boldsymbol{PE}_{latent} \subseteq \boldsymbol{PE}_{in}$$

[26] This is an assumption made in the formalization of the spaces presented in this chapter. The (non) connectedness of path variables can be an interesting question for future research.

An interesting problem (which is indirectly discussed in chapter 7) is that one of selecting a subset (i.e., a portfolio) of all possible path enablers in $\boldsymbol{PE}$ that best induces changeable behavior in the system (according to one's preferences). This is a combinatorial space of possibilities, which, as the cardinality of $\boldsymbol{PE}$ increases, soon becomes computationally intractable. The space of possible portfolios of path enablers can be defined as:

$$\boldsymbol{PE}^{\boldsymbol{Portfolio}} = \{(\varepsilon_1{}^{Portfolio}, \dots, \varepsilon_k{}^{Portfolio}, \dots, \varepsilon_T{}^{Portfolio}) \mid k \in \mathbb{Z}, 1 \leq i \leq T\}$$

And its cardinality is:

$$|\boldsymbol{PE}^{\boldsymbol{Portfolio}}| = \sum_{k=0}^{T} \binom{t}{k}$$

Path Inhibitor Space. Given N path inhibitors, it is possible to define a path inhibitor space:

$$\boldsymbol{PI} = \{(\iota_1, \dots, \iota_i, \dots, \iota_N) \mid i \in \mathbb{Z}, 1 \leq i \leq N\}$$

Each path inhibitor ι_i represents *what* (perhaps partly) inhibits the system from changing from state A to unwanted state B. Again, a basic distinction is made here between path inhibitors that are engineered *in* the system, and those that are *on* the system, at the strategic, enterprise level. An example of path inhibitor *in* the system is mounting armor on the original design instance. An example of path inhibitor *on* the system can be the existence of reserve budget funds to use in periods of crisis. Two mutually exclusive subsets then form from $\boldsymbol{PI}$. A subset of path inhibitors *in* the system:

$$\boldsymbol{PI_{in}} \subseteq \boldsymbol{PI}$$

And a subset of path inhibitors *on* the system:

$$\boldsymbol{PI_{on}} \subseteq \boldsymbol{PI} \mid \boldsymbol{PI_{on}} + \boldsymbol{PI_{in}} = \boldsymbol{PI}, \boldsymbol{PI_{on}} \cap \boldsymbol{PI_{in}} = \emptyset$$

As for path enablers, a path inhibitor *in* the system can be either added to the baseline design of a system (armor on a UAV), or latent in the baseline design instance (e.g., a UAV swarm is inherently distributed, making it hard to be taken out all at once). Hence, it is possible to define a set of latent path inhibitors:

$$\boldsymbol{PI_{latent}} \subseteq \boldsymbol{PI_{in}}$$

As for the case of path enablers, an interesting problem is that one of selecting a subset of all possible path inhibitors in $\boldsymbol{PI}$ that best enables resisting behavior in the system. This is a combinatorial space of possibilities, which, as the cardinality of $\boldsymbol{PI}$ increases, soon becomes computationally intractable. The space of possible portfolios of path enablers can be defined as:

$$\boldsymbol{PI^{Portfolio}} = \left\{\left(\iota_1{}^{Portfolio}, \dots, \iota_k{}^{Portfolio}, \dots, \iota_T{}^{Portfolio}\right) \mid k \in \mathbb{Z}, 1 \leq i \leq T\right\}$$

And its cardinality is:

$$|\boldsymbol{PI^{Portfolio}}| = \sum_{k=0}^{T} \binom{t}{k}$$

The two spaces $\boldsymbol{PE}$ and $\boldsymbol{PI}$ can be added to the representation of the different spaces in the dynamic system perspective for design discussed in chapter 4. Unlike the others, the elements in these two spaces are herein assumed not be able to connect in a network.[27] However, the spaces of possible portfolios within them can. Figure 5-2 shows the new representation of the dynamic system perspective. This is a revisited representation of the spaces in Figure 4-3 with the addition of path enablers and path inhibitors. The path enabler and path inhibitor spaces give system designers change and resistance mechanisms to respond to the unfolding of uncertain events (i.e., perturbations in design, context, or needs).

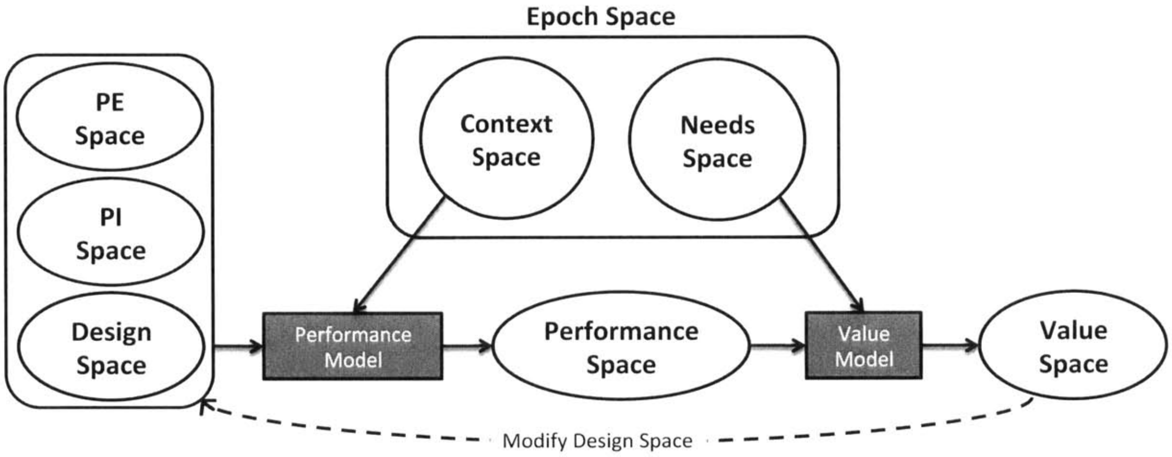


Figure 5-2: Revisited representation of the spaces in dynamic system design perspective.

5.4.2 Change Mechanisms and Resistance Mechanisms

Path enablers and path inhibitors are the necessary conditions for their respective mechanisms. In the same way perturbations are operators on design, context or needs instances, mechanisms can be thought of as operators on design instances (performance and value, also, in the case of resistance mechanisms) that enable either change to new ones (change mechanism) or resistance to change to new ones (resistance mechanism). These changes (or resistances to change) may be to prevent a value-disrupting perturbation (or facilitate a value-enhancing one) or mitigate and recover

[27] It would be interesting to test the limits of this assumption in a future research effort.

the effects of a value-disrupting perturbation (or exploit those of a value-enhancing one). The most important difference between perturbations and change or resistance mechanisms is that the former is unintentional, while the latter is intentional (from the perspective of designers and stakeholders).

If the current operating design instance is d_i, a change mechanism δ can be defined as:

$$\delta\colon \begin{cases} d_i \mapsto d_j \mid d_i, d_j \in \boldsymbol{D} \\ d_i \mapsto d_i' \mid d_i \in \boldsymbol{D}, d_i' \in \boldsymbol{D}' \end{cases}$$

Which is, an operator that transitions the current design instance d_i within the space $\boldsymbol{D}$ into either another design instance d_j within $\boldsymbol{D}$, or a design instance d_i'. The two types of change mechanism are illustrated in Figure 5-3. The new instance d_i'belongs to $\boldsymbol{D}'$, a slight variant of the initially enumerated design space $\boldsymbol{D}$ that includes d_i'(such that $\boldsymbol{D} \subseteq \boldsymbol{D}'$). It is important to reiterate that a change mechanism is being modeled similarly to a perturbation in the design space, and that it differs from it insofar as it is related to changes that are intentional.

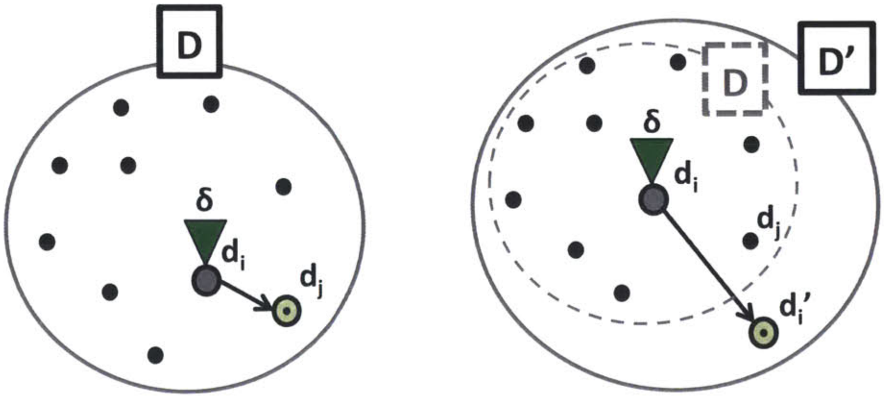


Figure 5-3: Change mechanisms are operators on the design space. The green design instance (with dot inside) indicates that it is a desired one.

If the current operating design instance is d_i, a resistance mechanism ω can be defined as:

$$\omega\colon \begin{cases} d_i \nrightarrow d_j \mid d_i, d_j \in \boldsymbol{D} \\ d_i \nrightarrow d_i' \mid d_i \in \boldsymbol{D}, d_i' \in \boldsymbol{D}' \end{cases}$$

Which is, an operator that resists a forced transition of the current design instance d_i within the space $\boldsymbol{D}$ into either another design instance d_j within $\boldsymbol{D}$, or a design instance d_i' that does not exist in the current design space $\boldsymbol{D}$. The resistance mechanism occurs either by preemptively reducing the likelihood of the imposed transition (i.e., perturbation) or by opposing the change upon occurrence of the perturbation (what is shown in Figure 5-4).

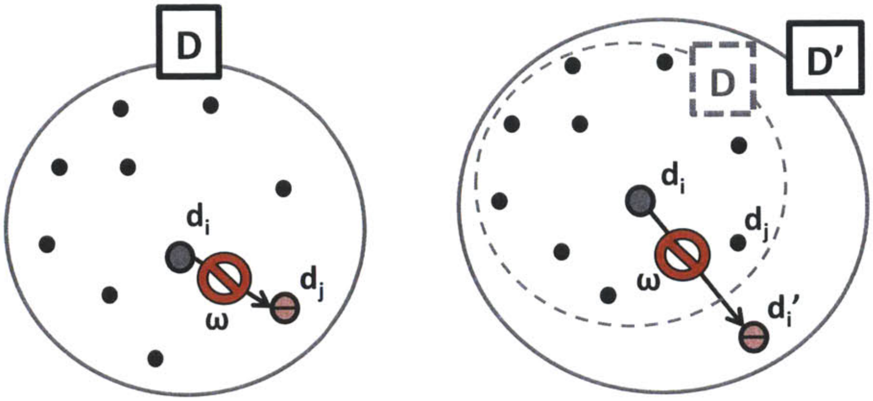


Figure 5-4: Resistance mechanisms are operators on the design space. The red design instance (with dash inside) indicates that it is an undesired one.

It is important to point out here that, unlike change mechanisms, resistance mechanisms are not assumed to operate only on the design space. Resistance mechanisms can in fact also operate on the performance space or on the value space directly, in order to impede changes in performance or value caused by perturbations in the context or needs space. For example, considering a military vehicle, a resistance mechanism in the design space is 'absorbing a physical hit' due to an 'enemy attack' (perturbation in the design space). On the other hand, a resistance mechanism on the performance space can be enabled by multi-terrain tires: if the terrain changes from asphalt to dirt (perturbation in context), the performance of the system does not change. A resistance mechanism in the value space relates to the concept of versatility: if needs shift from carrying a cargo to attacking an enemy, a vehicle that is capable of doing both (for example, by having weapons mounted on it) will resist loss of value due to the inability of attacking enemies. Hence, there exist two other types of resistance mechanisms (Figure 5-5):

$$\omega\colon \begin{cases} p_i \nrightarrow p_j \mid p_i, p_j \in \boldsymbol{P} \\ p_i \nrightarrow p_i' \mid p_i \in \boldsymbol{P}, p_i' \in \boldsymbol{P'} \end{cases}$$

$$\omega\colon \begin{cases} v_i \nrightarrow v_j \mid v_i, v_j \in \boldsymbol{V} \\ v_i \nrightarrow v_i' \mid v_i \in \boldsymbol{V}, v_i' \in \boldsymbol{V'} \end{cases}$$

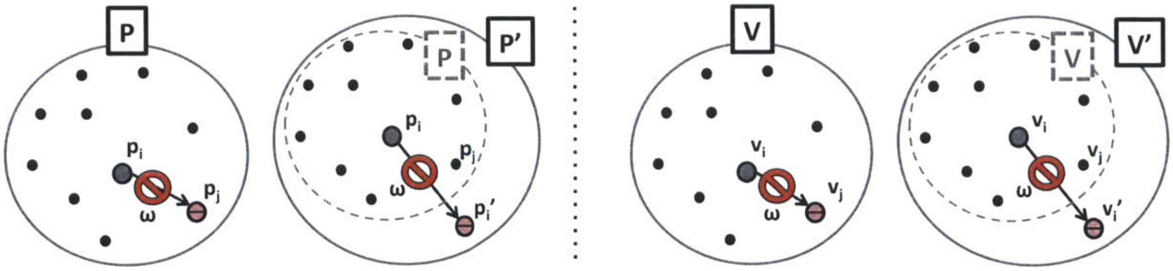


Figure 5-5: Resistance mechanism in the performance space (left) and value space (right) impede an instance from transitioning into a new one.

5.4.3 Change Options and Resistance Properties

Path enablers and path inhibitors are what enables or inhibits (respectively) a desired or undesired (respectively) change event. Change mechanisms and resistance mechanisms are the methods through which the change event is enabled or inhibited (respectively). In previous sections, it has been pointed out that a change option is the union of a (series of) path enabler(s) and a change mechanism. Similarly, a resistance property is the union of a (series of) path inhibitor(s) and a resistance mechanism.

It is possible, then, to formally define a change option:

$$CO : \bigwedge_{i=1}^{N} \varepsilon_i \Rightarrow Poss(\delta)$$

That is, a change option is a logical implication, whereby the existence of a set of N path enablers in conjunction implies the *executability* of a change mechanism (i.e., the option to execute it). The notation "$Poss$" is borrowed from situation calculus (McCarthy, 2002; Levesque et al., 1998), a logic formalism designed for representing and reasoning about dynamical domains (situations). " $Poss$ " is a special binary predicate denoting executability of actions. For example, the existence in conjunction of a modular payload bay on a UAV and a scientific payload on the ground implies the possibility of changing the current payload to the one on the ground by swapping them – i.e., the executability of the change mechanism 'swapping payload.' In the limit $N \rightarrow 1$, only one path enabler is required for the existence of a change mechanism.

Similarly, It is possible to define a resistance property:

$$RP : \bigwedge_{i=1}^{N} \iota_i \Rightarrow \omega$$

That is, a resistance property is a logical implication, whereby the existence of a set of N path inhibitors in conjunction implies the *existence* of a resistance mechanism. For example, the existence of armor on a UAV implies the absorption of a physical attack (resistance mechanism). In the limit $N \rightarrow 1$, only one path inhibitor is required for the existence of a resistance mechanism.

5.5 Dynamic Impact of Ility-Driving Elements

Section 4.3.3 discusses the dynamic impact of perturbations on the design effort. In order to respond to perturbations, the ility-driving elements described above are introduced in the system. The dynamic impact of ility-driving elements – either for preventing or responding to perturbations – in the design effort is discussed here.

5.5.1 Dynamic Impact of a Change Option

The execution of a change option implies a change in the design of the system (at the design- or enterprise-level). Such a change happens through a change mechanism – an operator on the design space. In general, the intent behind the execution of a change option stems from two main reasons: either (1) affecting the likelihood of a change in value delivered by the system, or (2) increasing the value delivered by the system. For perturbations negatively affecting value delivery (risk), intent (1) can be further subcategorized into (a) preventing a perturbation in design, (b) preventing the impacts of perturbations in context or needs,[28] and intent (2) occurs in an attempt to mitigate the effects or recover from the results of a negative perturbation in design, context or needs. For perturbations that can have positive impacts on value delivery (opportunity), intent (1) is mainly related to increasing the likelihood of the impacts from those perturbations in context or needs, and intent (2) occurs in an attempt to seize the opportunity such perturbations bring about. This information is summarized in Table 5-1.

Table 5-1: Summary of different intents for executing changes.

		Intent behind change	
		(1) Affecting the likelihood of a change in value delivery	(2) Increasing the value delivered by the system
Effect of perturbation	Risk	(a) Preventing a perturbation in design (b) Preventing the impacts of perturbations in context or needs	In an attempt to mitigate or recover from the results of a negative perturbation in any of the spaces
	Opportunity	Mainly related to increasing the likelihood of the impacts from perturbations in context or needs	Occurs in an attempt to seize the opportunity perturbations bring about

For the case of perturbations introducing risk of value loss, Figure 5-6 through Figure 5-11 illustrates the possible cases. In these figures, the red dots (with dash inside) indicate an instance in the space that is undesired. The green dots (with point inside) indicate an instance in the space that is desired. Furthermore, in the preventive case (Figure 5-6 through Figure 5-8) the dashed items never occur.

Figure 5-6 illustrates the possibility of changing in order to prevent a perturbation in design that would ultimately result in a value loss (*preventive change*). For example, in the case of the MarSec SoS, increasing the flying altitude of the swarm of UAVs (i.e., a change in CONOPs) reduces the chances of being hit by a hostile missile attack. Figure

[28] Intents (a) and (b) here are a result of assuming that the designer does not have any control over the context space or the needs space. Assuming otherwise (perfectly valid assumption and an interesting one for future research) would lead to a different hierarchy of intents.

5-7 illustrates the possibility of changing in order to prevent the impacts of a perturbation in context (the case of perturbations in needs is analogous) that would ultimately result in a value loss (*preventive change*). For example, in the case of the MarSec SoS, decreasing the flying altitude of the swarm of UAVs (i.e., a change in CONOPs) reduces the chances of being subject to adverse weather conditions (causing bad communication and target identification) forecast for higher altitudes. Lastly, Figure 5-9 through Figure 5-11 illustrate the possibility of changing in an attempt to recover from the negative impacts on value of a perturbation in any of the spaces (*reactive change*). For example, for the MarSec SoS, increasing the number of flying UAVs, thanks to the availability of a spare one on the ground, enables the SoS to recover from the loss of a UAV due to engine failure (this example is an instantiation of the flow in Figure 5-9).

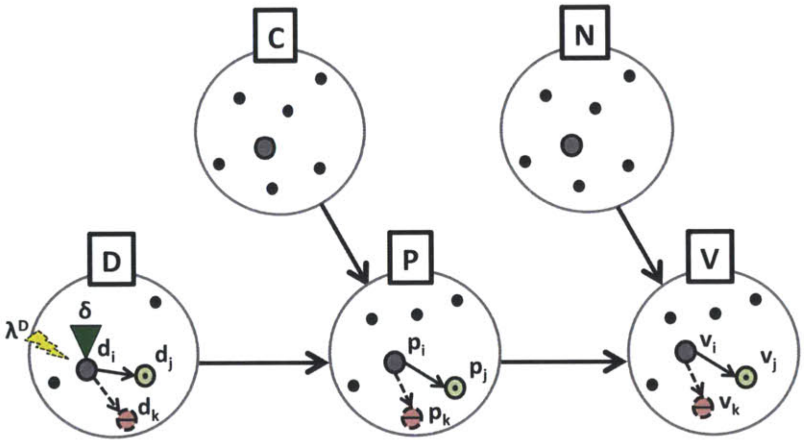


Figure 5-6: Preventive change. The perturbation in design never occurs because the design was changed preemptively.

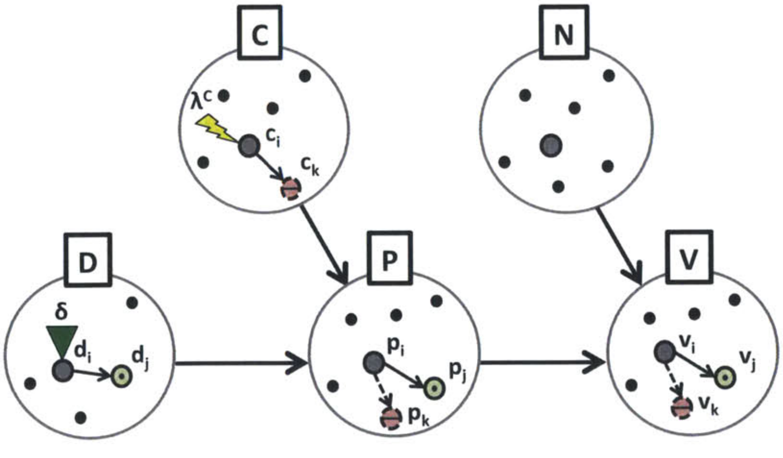


Figure 5-7: Preventive change. A change in context does occur, which would have led to an undesired design-context pair, but the design is changed preemptively so to never experience the undesired design-context pair.

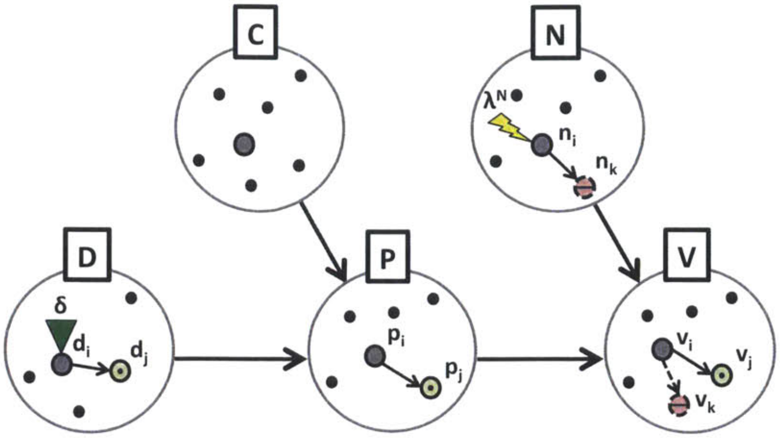


Figure 5-8: Preventive change. A change in needs does occur, which would have led to an undesired design-epoch pair, but the design is changed preemptively so to never experience the undesired design-epoch pair.

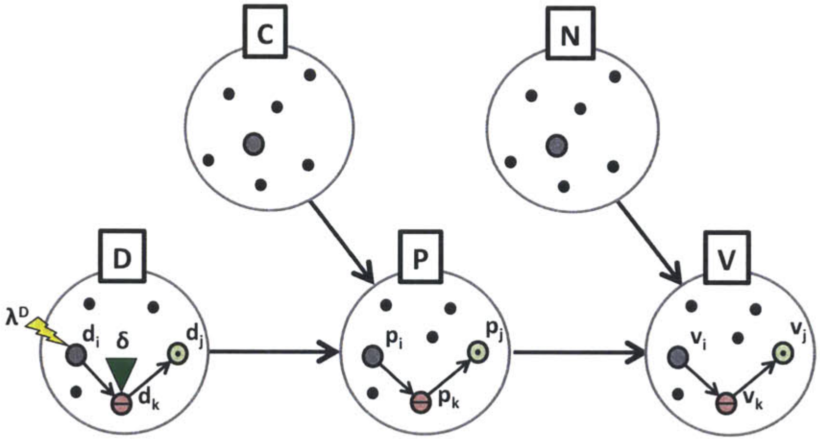


Figure 5-9: Reactive change. The perturbation pushes the design to an undesired instance. Then, the design instance is changed in order to recover from such change.

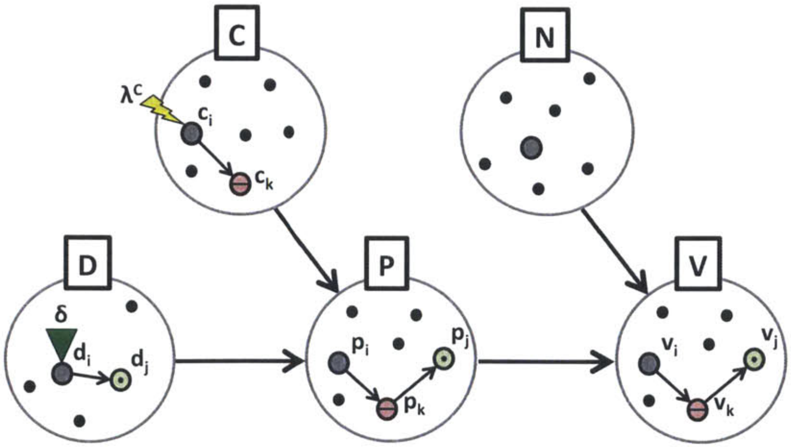


Figure 5-10: Reactive change. The perturbation pushes the context to a new instance, which results into an undesired performance instance. Then, the design instance is changed in order to recover from such change.

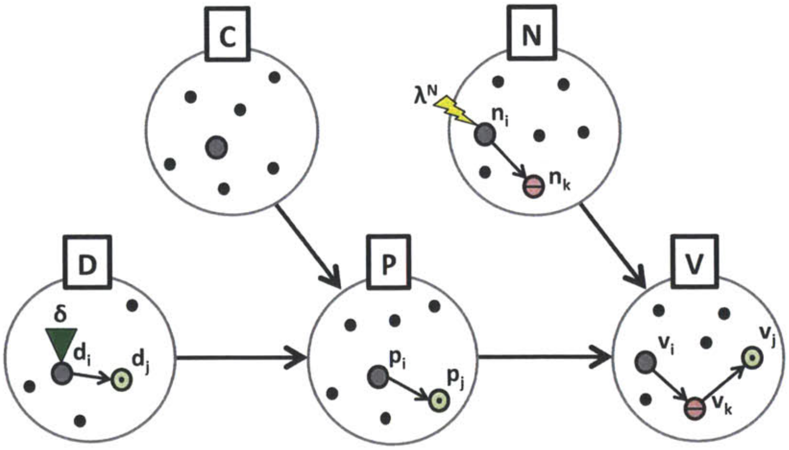


Figure 5-11: Reactive change. The perturbation pushes the needs to a new instance, which results into an undesired value instance. Then, the design instance is changed in order to recover from such changes.

5.5.2 Dynamic Impact of a Resistance Property

Resistance properties (as defined in the previous section), unlike change options, do not necessarily require execution agents.[29] As such, they are linked primarily to risk reduction, and not opportunity seizing: i.e., to perturbations negatively impacting value delivery. Despite this, they can still be included in the system for preventive or reactive reasons. They can be preventive in the sense that they reduce the likelihood of occurrence of a perturbation, without needing the execution of an action (e.g., a camouflage paint for a military vehicle reduces the likelihood of being identified and attacked by enemies). They can be reactive in the sense that they mitigate the adverse effects of a perturbation (e.g., armor on the military vehicle mitigates the adverse effects of a hostile attack). Resistance properties are assumed to be able to prevent the occurrence of a perturbation in design only (Figure 5-12) – e.g., camouflage. For perturbations in context or needs, one can resort to resistance properties that reactively mitigate the effects on performance and value (respectively). For example, for a military vehicle, if the terrain changes from asphalt to dirt (perturbation in context), multi-terrain tires resist changes in performance. Similarly, if needs shift from carrying a cargo to attacking an enemy, a vehicle that is capable of doing both (by having weapons mounted on it) will resist loss of value (due to the inability of attacking enemies). Figure 5-13 through Figure 5-15 show (reactive) resistance properties that mitigate the effects of perturbations in design, context and needs.

[29] Again, this is a modeling assumption made in the context of this research. As such, it may be revisited in other contexts. In fact, one may argue that some resistance properties can be activated. For example, windshield wipers are activated so to resist the undesired effects of rain on visibility. Within the context of this thesis, any activation (or execution, more generally) is taken to imply a change in the form or operations of a current design instance. As such, it is modeled as a change in design instance (a combination of forms and CONOPs), and hence a change option.

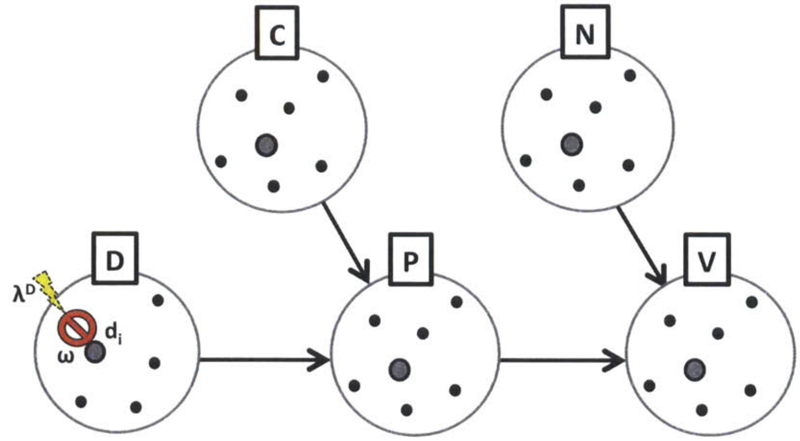


Figure 5-12: Preventive resistance property preventing occurrence of perturbation in design.

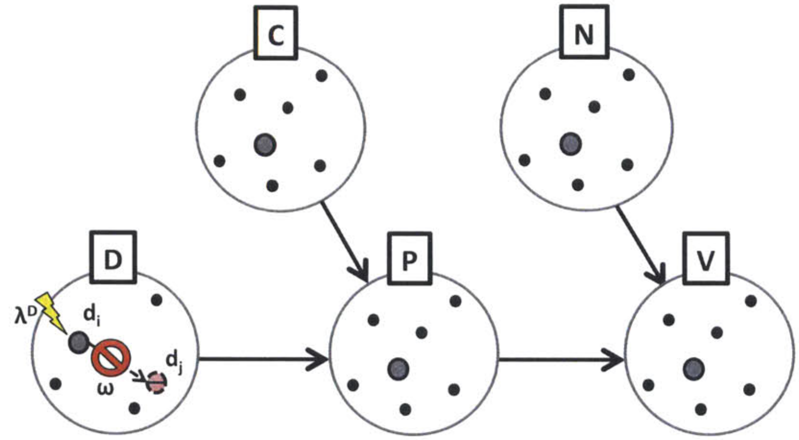


Figure 5-13: Reactive resistance property mitigating the impacts of perturbations in design.

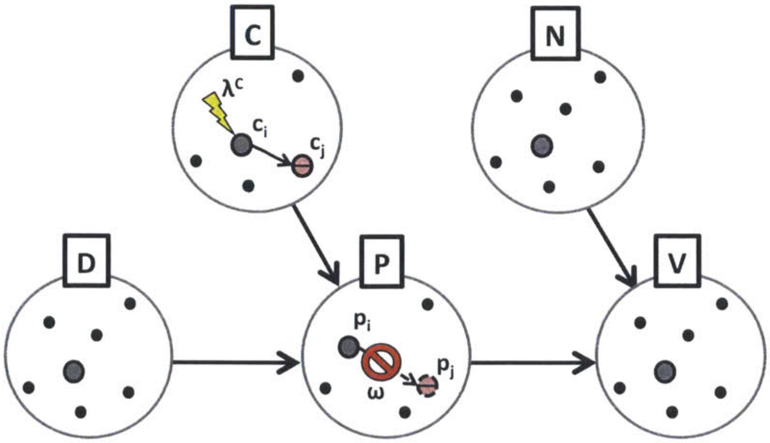


Figure 5-14: Reactive resistance properties mitigating the impacts of perturbations in context.

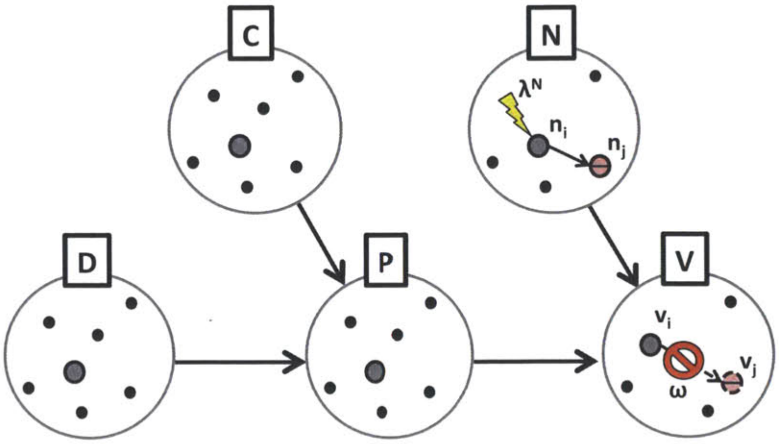


Figure 5-15: Reactive resistance properties mitigating the impacts of perturbations in needs.

5.6 Descriptive Fields for Ility-Driving Elements

This section introduces some descriptive fields for the categorization of ility-driving elements. As for perturbations, these descriptive fields can aid in the process of comparing ility-driving elements, and generating a holistic view on a set of ility-driving elements. This way, a system designer can gather a larger understanding of the ways in which the system can change and resist change over time. In the following paragraphs, some salient descriptive fields for IDEs are discussed.

Decision type. An important descriptor for an IDE is the type of decision it is linked to. Two very important decisions are that of acquiring the path enablers or path inhibitors for inclusion in the system and that of executing the mechanism. For resistance properties, it is usually the case that decisions only regard inclusion in the system (as defined in 5.4.3). For change options, on the other hand, decisions may be related to both inclusion and execution. While the execution decision is necessary, the inclusion one may not be needed for those change options (and resistance properties) enabled by latent path enablers (inhibitors) only. Another important decision is that of disposal of the IDE (i.e., of its path variables). This decision type may not be available for some IDEs: i.e., once the IDE is in the system, it will stay there until end of lifecycle.

Execution agent. This descriptive field assesses what type of agent is associated with a given mechanism. In the case of resistance properties, resistance mechanisms have no execution, and hence no agent associated with them. As for change option, an execution agent always exists. An execution agent implies a sensing unit (e.g., a thermometer) and a decision rule (e.g., if temperature is above 20 degrees Celsius, turn on Air Conditioner). The implementer of a decision rule is the execution agent. The location of the agent can be either internal to the system (e.g., embedded in the form of algorithms in an aircraft control system), or external to the system (e.g., the UAV operator). Ross et al. (2008) discuss how location of the agent can be a useful taxonomic distinction for classifying change. In this taxonomy, if the change agent is external to the system, the change under consideration is a flexible-type change (related to flexibility). If internal to the system, the change is an adaptable-type one (and related to adaptability). Depending on the particular change being considered, a single system can be both flexible and adaptable. It is important to understand that this taxonomy depends on where the boundaries of a system lie. In order for it to be useful and unambiguous, the system boundary must be explicitly defined.

Epoch-dependent unavailability. As discussed in chapter 4, an epoch is a fixed time period in which context and expectations of the system don't change. The epoch space is obtained from a list of epoch variables (in turn, the union of a list of context variables and needs variables). Epoch-dependent unavailability refers to the fact that, in certain epochs, a mechanism may not be executable. For example, if a mechanism of change for a military system is to decentralize its assets by deploying them in a friendly country, upon variation in the geo-political scenario that worsens the relationship between the two countries, such a mechanism is no longer executable.

Reusability. This is the ability to reuse an ility-driving element across the lifecycle of a system: from just once to as many times as needed. For example, a decoy for missile attack can be used only once; the ability of a satellite to adjust its orbit may be used a finite amount of times, until fuel runs out; lastly, the ability to reorganize the geographic segmentation of UAV coverage for the case of the MarSec SoS may be used as many times as needed (given no epoch-dependent unavailability). In general, resistance properties are always present and available, as long as the path inhibitors are (e.g., armor, multi-terrain tires, distributed communications network). However, it is important to note that – similar to path enablers – path inhibitors can be used up as well (e.g., ablative shielding on spacecraft burn up on use). Reusability is very important when analyzing the dynamic usage of ility-driving elements, and especially of change option, as they imply execution decisions.

Lifecycle. An ility-driving element has a lifecycle: from the decision to include the IDE in the system to the moment in which it is no longer available. Two important dates in the IDE lifecycle are start date and expiration date (if any). The start date corresponds to the moment in which all path enablers (or inhibitors) necessary for a change (or resistance) mechanism become executable (or exist). This may be at the inception of operations, or later in time. The expiration date is when the mechanism is no longer available (analogous to the expiration date of financial options discussed in chapter 2). In general, expiration date is more common for change options than resistance properties (but not always: e.g., multi-terrain tires may have an expected lifetime of 3 years before they deteriorate). Another important date in the lifecycle of the IDE is execution date. If the IDE can be used only once, then execution date also indicates the end of the IDE lifecycle. If the IDE is no longer needed and the system is still in operations, it can be disposed of. Hence, disposal date – as for expiration and execution date – can also indicate the end of an IDE lifecycle. Furthermore, if the whole system hits the termination of lifecycle, then the IDE does as well (unless it can be reused for other systems or sold).

Target. This field describes what the ultimate goal of an ility-driving element is. As discussed in 5.5.1, ility-driving elements can be aimed at affecting the likelihood of the occurrence of a perturbation, or the overall value delivery of the system post-occurrence. Some IDEs may be able to do both things in different circumstances. For example, the execution of a change option that increases the flying altitude of a UAV may be targeted at preventing the occurrence of being hit by a missile. However, the same option can be used for exploiting the opportunity of enlarging a UAV's field of view, upon the acquisition of new, higher definition camera that just entered the market.

Cost. For an IDE, there are four main types of (monetary) cost to be incurred: (1) acquisition, (2) carrying, (3) execution, and (4) disposal. Acquisition is the cost of purchasing the set of path enablers or path inhibitors needed for a mechanism. Carrying is the cost associated with maintaining the path variable in a condition that enables the execution of a change mechanism or the availability of a resistance property. For example, a workforce buffer (enabling a change in number of operators per UAV) has a periodic cost associated with the salary of each extra worker. Execution cost is related to any expense incurred upon the execution of a change mechanism (or the emergence of a resistance property) – e.g., expanding the production output of a good. Execution costs may vary depending on the epoch a mechanism is executed in. Finally, cost of disposal is that of any process associated with the termination of an IDE (e.g., demolishing a vehicle). It is important to note that IDEs may not have all four cost types. For example, latent path variables have no acquisition cost; similarly, in general, the (passive) execution cost of resistance properties is often zero. Lastly, in a dynamic analysis of carrying a set of IDEs with the system over time, considerations with regard to switching cost (i.e., cost associated with switching the portfolio of IDEs) may become relevant.

Execution time. This is related to another type of expenditure: time. Execution time is the time it takes from the moment the decision to execute a change option is made to the moment in which the new design instance is operational (as long as the path inhibitor is available, there is no execution time for a resistance property). This may take years in the case of an architectural overhaul of a major SoS (e.g., DISA JIE or the BMDS, both discussed in chapter 6). It may also take a negligible amount of time: e.g., turning the windshield wipers on in a car. The perception of execution time (like all time), however, is relative. The same change mechanism may seem to take too long in some circumstances and too short in others. For example, the mechanism 'increasing UAV altitude' in the context of the MarSec SoS may be perceived to be long if executed to counter a turbulence or storm, but short if executed to increase the field of view of a camera for a long-term mission.

Optionability. Mikaelian (2009) defines optionability as the "ability to enable types of real options." The latter, as pointed out in chapter 2, correspond to change mechanisms in the lexicon of Ross (2006) and the present thesis. Hence, optionability is related to the ability of a path variable to enable more than one mechanism. This concept (as well as a means of quantifying it) is described in chapter 7.

Realizability. Similarly, Mikaelian (2009) defines realizability as the "the ability to implement a given type of real option." In the lexicon of Ross (2006) and the present thesis, this corresponds to the ability of a change mechanism to be enabled by more than one path enabler. As for optionability, it is possible to expand the concept of realizability to both types of ility-driving elements. This concept is also further discussed in chapter 7.

Risk attenuation. As discussed in chapter 4, depending on the consequence (one of the descriptive fields discussed) of a perturbation – negative or positive – risk or opportunity (respectively) may be introduced. Risk attenuation concerns how well an (or a set of) ility-driving element addresses a given risk space. Two major assessments must be made here in order to obtain a description of an IDE's risk attenuation: (1) a characterization of the risk space, as to what perturbations compose it and what their likelihood and impacts are; and (2) the extent to which the IDE addresses (i.e., is able to prevent or respond to) any of the perturbation in the space. Chapter 4 (building on chapters 2 and 3) has discussed a way of identifying a relevant set of perturbations. However, the characterization of their risk profile is a different (and highly subjective) process. The view on risk characterization taken in this thesis reflects to a great extent that put forth by the National Research Council in the *red book* (National Research Council, 1996, pp. 3):

> Risk characterization is the outcome of an analytic-deliberative process. Its success depends critically on systematic analysis that is appropriate to the problem, responds to the needs of the interested and affected parties, and treats uncertainties of importance to the decision problem in a comprehensible way. Success also depends on deliberations that formulate the decision problem, guide analysis to improve decision participants understanding, seek the meaning of analytic findings and uncertainties, and improve the ability of interested and affected parties to participate effectively in the risk decision process. The process must have an appropriately diverse participation or representation of the spectrum of interested and affected parties, of decision makers, and of specialists in risk analysis.

Oftentimes, and in various disciplines, risk is associated with two main descriptors of an event: its likelihood of occurrence and its impact.[30] For example, the risk of the next earthquake hitting Southern Italy (a highly seismic area) is a combination of its probability of occurrence and its impact – often measured in terms of energy released by the earthquake (e.g., on a Richter scale). Such thinking is also present in the field of systems engineering. For example, in the *NASA Systems Engineering Handbook*, the

[30] It is opportune to point out here that such conceptualization of risk is often present in domains for which characterization of uncertainty is very hard and there is not much data on past events. In other fields that have more information at their disposals, a different approach to quantifying risk may be taken. For example, in finance risk is often seen as the variance around a mean return on investment (Markowitz, 1952). However, even in these cases, confining the concept of risk within a mathematical formalization becomes reductive. Over the years, in fact, different characterizations of risk have arisen in finance – e.g., PMPT (Swisher and Kasten, 2005) – that take into account other information (e.g., skewedness and kurtosis of distribution).

National Aeronautics and Space Administration (NASA) divides risk into four types (cost, schedule, technical and programmatic), and defines it as follows (National Aeronautics and Space Administration, 2012, pp. 139):

> Risk is a measure of the inability to achieve overall program objectives within defined cost, schedule, and technical constraints and has two components: (1) the probability of failing to achieve a particular outcome and (2) the consequences/impacts of failing to achieve that outcome.

Chapter 4 has discussed perturbation likelihood and impact in the context of this thesis, as well as ways to assess them (e.g., the Delphi Method). It is important to note that such assessments are subject to a variety of assumptions and limitations (e.g., ordinal scales of measurement, experiential biases, personal ideologies, etc.), which are further discussed in chapter 7. As such, and in the absence of more reliable information, they must be treated as "fallible indicators" (Hammond, 1996) – representing a degree of belief (de Finetti, 1937) about states that are not directly observable. The willingness to accept the potential error and pitfalls introduced by using such indicators is a policy decision for the stakeholders' discretion.

A useful tool that provides assistance in the assessment, management and communication of risk is a risk matrix. A risk matrix is not an assessment tool per se, but it can provide guidance and facilitate discussion (National Aeronautics and Space Administration, 2012). An example of a 5x5 risk matrix is shown in Figure 5-16. As discussed above, risk is relative to the person or organization in question, as well as the problem of concern. Hence, for a given organization, a common methodology for interpreting a risk matrix must be defined. For example, in (National Aeronautics and Space Administration, 2012, pp. 145), a definition widely used by NASA, other government organizations, and industry is provided:

> **Low (Green) Risk**: Has little or no potential for increase in cost, disruption of schedule, or degradation of performance. Actions within the scope of the planned program and normal management attention should result in controlling acceptable risk.
>
> **Moderate (Yellow) Risk**: May cause some increase in cost, disruption of schedule, or degradation of performance. Special action and management attention may be required to handle risk.
>
> **High (Red) Risk**: Likely to cause significant increase in cost, disruption of schedule, or degradation of performance. Significant additional action and high-priority management attention will be required to handle risk.

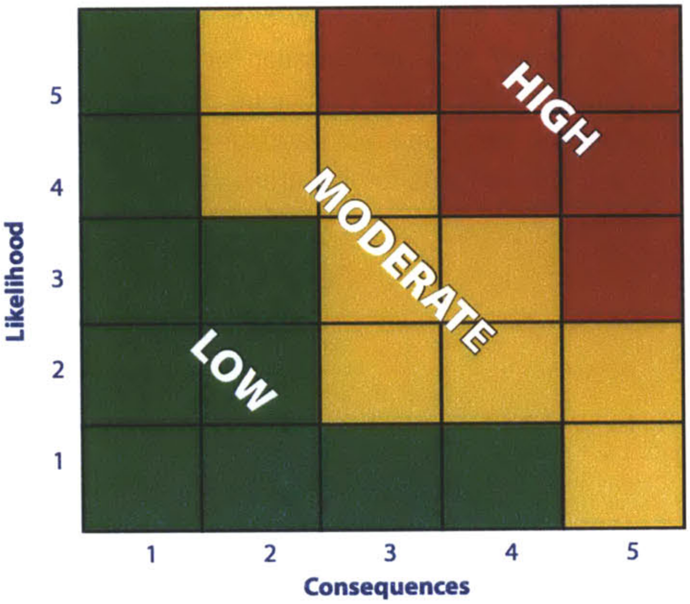


Figure 5-16: Risk matrix (National Aeronautics and Space Administration, 2012).[31]

Risk attenuation is related to how well (relative to whatever metrics or tools are used for the assessment) a given IDE addresses the risk space (an assessment that comes after that of risk). Chapter 7 discusses some proxy metrics for risk attenuation within the context of comparing different IDEs (and portfolios thereof).

Opportunity Exploitation. This is concerned with the perturbations that may have positive consequences (as discussed in chapter 4). Opportunity exploitation describes how capable a (or a set of) change option(s) is of addressing a given opportunity space. Since the concept of opportunity implies the possibility of *acting* to gain some extra value, within the context of this thesis, it is assumed that only change options (and not resistance properties) are linked to opportunity exploitation.[32]

5.7 Summary

In this chapter, a formal description of the elements that can drive the emergence of ilities over time was introduced. Such ility-driving elements have been divided into two main types: change options and resistance properties. The former, composed of path enablers and change mechanisms, imply active change in the current design instance by the hands of an executive agent. The latter resist changes in value delivery (by resisting changes in either design, performance, or value instance), and do not imply

[31] Risk matrices have been the subject of a long debate over its usefulness in the literature. For example, (Cox, 2008) points out many possible pitfalls associated with the use of risk matrices. (Talbot, 2011), on the other hand, responds to Cox's article by pointing out the upsides of using risk matrices.

[32] Testing the limits of this assumption may be an interesting *opportunity* (never was word choice more appropriate) for future research.

executability. It has been further discussed how both types of IDE can be targeted at either preventing the occurrence of a perturbation or reacting to it (by mitigation or recovery). The dynamic impact of such IDEs on value delivery has also been discussed, and all possible scenarios for value sustainment/enhancement have been described. Lastly, several relevant descriptive fields for ility-driving elements were discussed.

6 An Empirical Investigation of Change Options

"Well, randomness sounds promising, but conceptualization is beyond my range."

– Lawrence E. McCray (2014)

This chapter presents the results of an empirical investigation of change options. The first part (and the majority) of the chapter presents some of the results from an investigation on evolvability in military SoS. For a given evolution increment, the change options that enabled it were investigated. In the last part, the idea of planned adaptation in the policymaking realm is discussed and linked to the concept of change option. The link between the two is explained through the lenses of some policy cases (the most prominent of them being the regulation of particulate matter).

6.1 Empirical Examples of Change Options in Military SoS

In this section, results of case investigations of evolvable Systems of Systems (SoS) in the military domain are presented. The selected SoS cases illustrate examples of specific change options implemented by architects in designing SoS that resulted in evolvable SoS. The larger aims of this research are to further validate the usefulness of design principles for architecting systems that possess desirable lifecycle properties such as evolvability, and contribute real-world examples that architects may use to inspire specific design options for their system of interest.

6.1.1 Evolvability

The concept of evolvability has long been explored in the field of biology, where it has been defined as "the ability of a population to both generate and use genetic variation to respond to natural selection" (Colagrave, 2008). In the field of systems engineering, analogies between artificial systems and biological organisms have been drawn, and the concept of biological evolvability has often inspired the concept of systems evolvability. For example, Sussman (2007) describes evolvable systems as being able to accommodate adaptive variations in some locales without changing the behavior of subsystems in other locales. In the field of SoS engineering, evolvability has been defined as the "ability of the architecture to handle future upgrades" (Butterfield et al., 2008), with specific references to how an evolvable SoS is architected with an eye for technologies and features that are required to support possible future capabilities. Finally, in an attempt to define evolvability broadly enough so that it would be unambiguously applicable to different domains, Beesemyer (2012) proposes the following definition for evolvability: "the ability of an architecture to be inherited and changed across generations [over time]." This definition is the culmination of a larger collaborative research effort aimed at the development of methods to design for evolvability (Beesemyer et al., 2011). Beesemyer's definition of evolvability rests upon a key distinction among systems' architecture, design and instance – where the architecture is the highest level of abstraction among the three. While changeability can take place at all three levels of abstraction, evolvability is only linked to architecture changes, and can be thought of as "architecture-level changeability." Beesemyer and Fulcoly (2011) collect different concepts and ideas regarding evolvability from various fields (from biology, to technology innovation, to systems engineering) and synthesize them in a set of design principles for evolvability, listed in Table 6-1.

Table 6-1: Design principles for evolvability.

Design Principle	Description
Leverage Ancestry	Employing successful design choices of assets, capabilities and/or operations from all prior generations of the system
Disruptive Architectural Overhaul	Re-architecting significant portions of the existing system or program at the same time in order to reduce the negative impact that making many smaller changes would have
Mimicry	Imitating or duplicating successful design choices of assets, capabilities and/or operations from other systems/domains for a similar purpose
Resourceful Exaptation	Repurposing assets or design choices from prior generations or other systems/domain in order to provide capabilities for which they were not originally selected
Decentralization	Distributing assets, capabilities and/or operations to appropriate multiple locations, rather than having them located in a single location

Design Principle	Description
Targeted Modularity	Isolating parts of the system to reduce interdependencies in order to limit undesirable effects caused by either uncertainties or intentional changes
Integrability	Designing interfaces for compatibility and commonality to enable effective and efficient integration of upgraded/new system components and constituents
Reconfigurability	Creating intentional similarities in form and/or function of various system assets, capabilities, and/or operations to facilitate reuse or reallocation
Redundancy	Intentional duplication of selected assets, capabilities and/or operations to enable their future redistribution without compromising existing requirements
Scalability	Making design choices that allow scaling of resources and/or assets up or down in order to accommodate uncertainties and emergent needs
Margin	Architecting for intentional excess capacity in specific capabilities and/or operations to meet emergent needs without compromising existing requirements (i.e. meet or exceed future requirements)
Slack	Intentionally under-allocating or over-allocating specific available assets and/or resources in order to reserve excess capacity for accommodating uncertainties (i.e. prevent violation of constraints)

The first four in the set – leverage ancestry, disruptive architectural overhaul, mimicry, and resourceful exaptation – can be thought of as *strategies* design principles. Such design principles are not directly related to the architecture of the SoS, but rather suggest strategies for facilitating the achievement of evolvability properties. The remaining design principles in this set are *structural*, as they can influence the architecture of the SoS directly (e.g., a modular design implies a different architectural structure than a monolithic one).

6.1.2 Framework: from Design Principles to Change Options

Design principles are "guiding thoughts [for design] based on empirical deduction of observed behavior or practices that prove to be true under most conditions over time." (Wasson, 2006) In practice, design principles serve to help intentionally create desirable properties in a system (e.g., survivability, evolvability, flexibility, etc.). In the context of this study on evolvability, design principles are linked to the inclusion of change options in the design (or redesign) of the SoS.[33] They are used to inspire change options, as they can guide system designers through the process of brainstorming and formulating possible ways to include IDEs in systems.

The general framework through which evolvability has been studied in the military SoS is shown in Figure 6-1. In this diagram, design principles are depicted as what underlies (inspires) the insertion of a change option in the SoS. The execution of change options is then related to an evolution increment. In general, two primary types of options for every evolution event (i.e. transition from architecture A to architecture B) were investigated: those that have been used to implement the evolution, and those that were introduced into the SoS by the evolution (if any) – and that can potentially enable future evolution increments of the architecture.

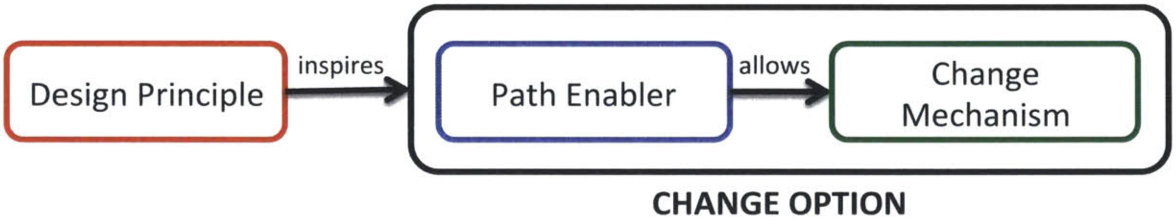


Figure 6-1: Framework for analysis of how evolvability has been instilled in SoS.

6.1.3 Military SoS

The evolvability study looked at various SoS – all conceived for military purposes. They span across three different SoS domains (Dahmann, 2012): mission, platform and IT. These SoS are very large in scope, and benefit from large budgets to consider new or different SoS implementations. The possibility of considering different SoS evolution increments is vital in order to sustain value delivery in fast-changing operational environments (e.g., US Quadrennial Review may disrupt the budgets and scopes of the above-listed SoS, which must be able to continue to deliver value). In this chapter, the cases of the Ballistic Missile Defense System (BMDS) – a mission SoS – and that of the Defense Information Systems Agency (DISA) Joint Information Enterprise (JIE) – IT SoS – are discussed more in depth.[34]

[33] Within the larger thesis context, the more general role of design principles in the generation of any IDE is discussed in chapter 7.

[34] It is important to note here that the research effort performed on these SoS is based on literature that is publicly available and unclassified.

6.1.3.1 Ballistic Missile Defense System

The Ballistic Missile Defense System (BMDS) is a mission SoS composed of several constituent systems and supporting efforts, as shown in Figure 6-2. Its main goal is to enable a robust, layered defense against hostile missiles in all phases of flight – boost, intermediate, and terminal. United States ballistic missile defense efforts can be traced to the World War II years, in an attempt to respond to the German V-2 missile, which was a threat for US' European allies. Initially, it turned out to be an arduous endeavor, and in fact it wasn't until 1960 that a US guided missile test resulted in a success. Through the years, the program has received different names and alternate attention, experiencing low peaks during the years of the Antiballistic Missile Treaty (1972-2002). The current version of the BMDS was deployed in 2004 with limited interception and detection capabilities, which have been increased since that time. The BMDS SoS architecture includes: networked sensors (including space-based) as well as ground-based and sea-based radars for target detection and tracking; ground-based and sea-based interceptor missiles for destroying a ballistic missile using either direct collision ("hit-to-kill") technology, or an explosive warhead; command, control, battle management, and communications network providing the operational commanders with the needed links between the sensors and interceptor missiles.

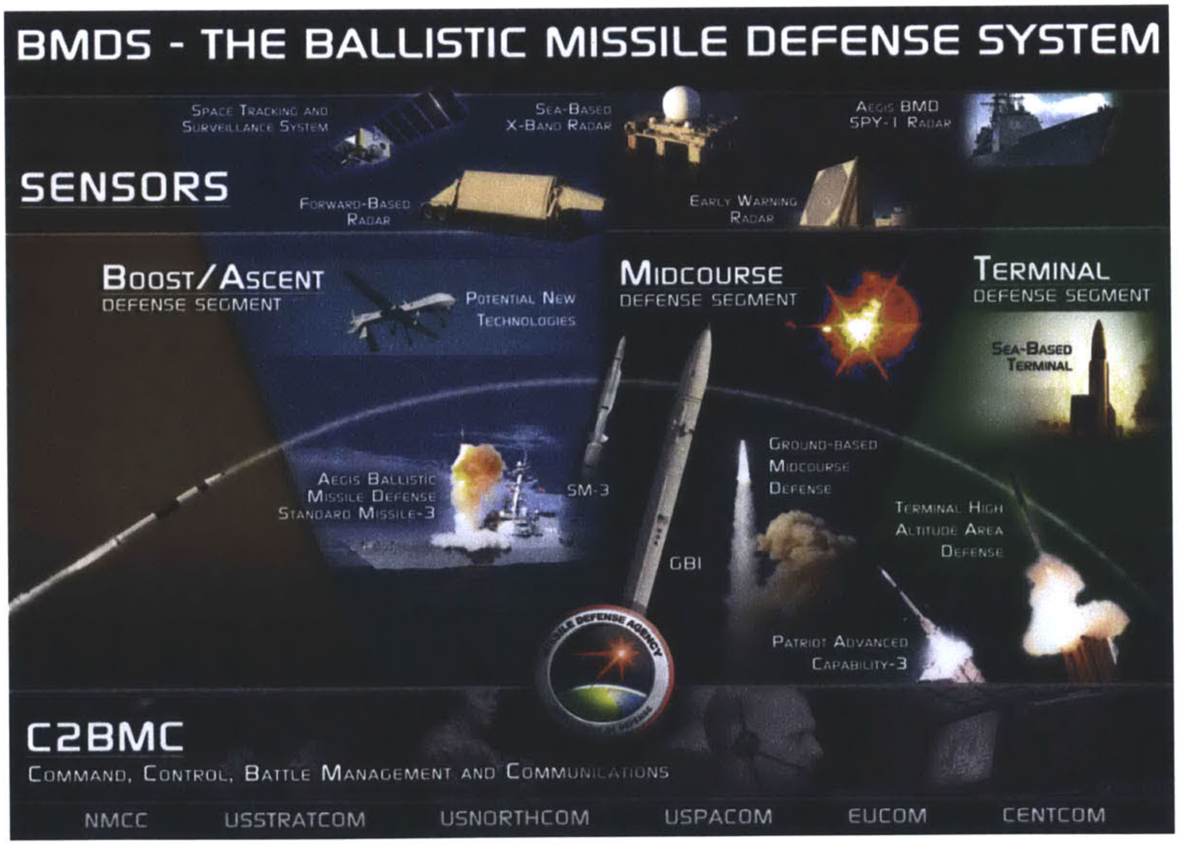
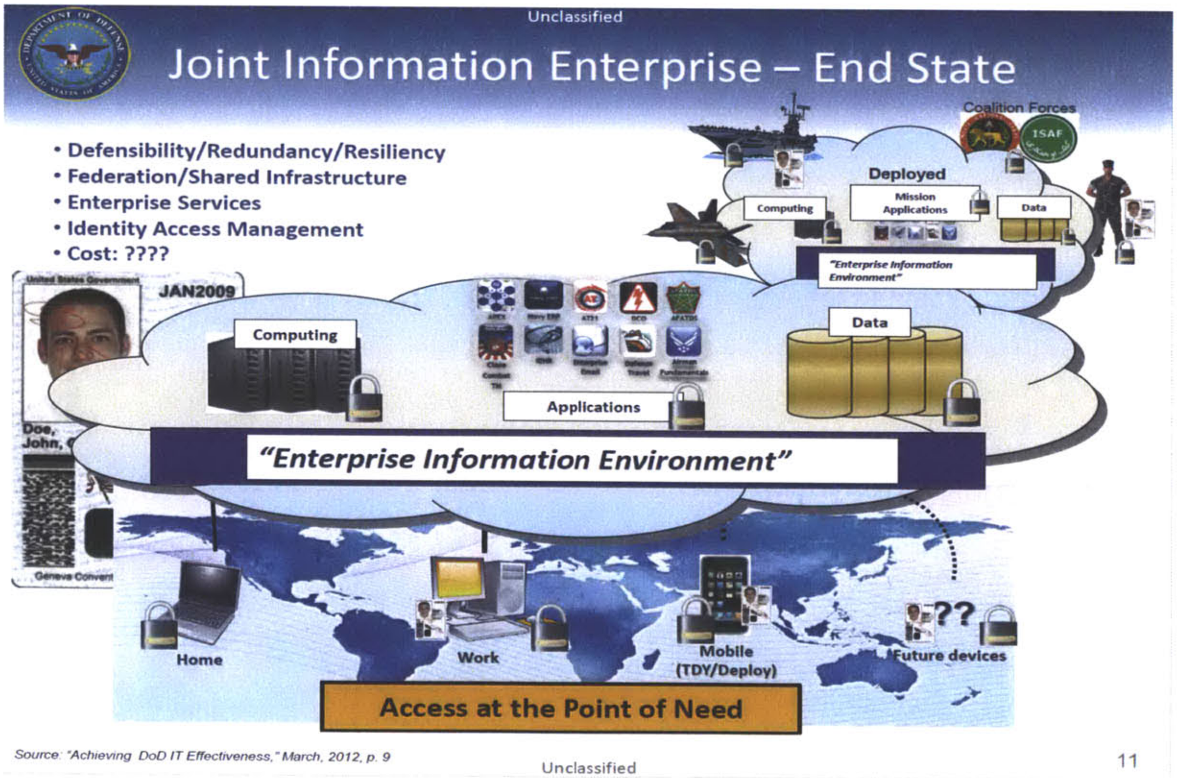

BMDS - THE BALLISTIC MISSILE DEFENSE SYSTEM
SPACE TRACKING AND SURVEILLANCE SYSTEM
SEA-BASED X-BAND RADAR
AEGIS BMD SPY-1 RADAR
SENSORS
FORWARD-BASED RADAR
EARLY WARNING RADAR
BOOST/ASCENT DEFENSE SEGMENT
POTENTIAL NEW TECHNOLOGIES
MIDCOURSE DEFENSE SEGMENT
TERMINAL DEFENSE SEGMENT
SEA-BASED TERMINAL
AEGIS BALLISTIC MISSILE DEFENSE STANDARD MISSILE-3
SM-3
GBI
GROUND-BASED MIDCOURSE DEFENSE
TERMINAL HIGH ALTITUDE AREA DEFENSE
PATRIOT ADVANCED CAPABILITY-3
MISSILE DEFENSE AGENCY
C2BMC
COMMAND, CONTROL, BATTLE MANAGEMENT AND COMMUNICATIONS
NMCC
USSTRATCOM
USNORTHCOM
USPACOM
EUCOM
CENTCOM

Figure 6-2: Main constituent systems in BMDS (http://www.mda.mil/system/system.html)

6.1.3.2 DISA Joint Information Enterprise

The Defense Information Systems Agency (DISA) is an agency within the Department of Defense that provides information technology and communication support to all entities contributing to the defense of the United States – from the president, to military services, to soldiers. The mission of DISA is to "provide, operate, and assure command and control, information sharing capabilities, and a globally accessible enterprise information infrastructure in direct support to joint Warfighters, National level leaders, and other mission and coalition partners across the full spectrum of operations." (DISA, 2014)

The agency is putting forth an IT SoS, the Joint Information Environment (JIE) (see Figure 6-3), intended to be a central information sharing solution to improve DoD's ability to share information – not just between the services, but also with its industry partners and other government agencies. Currently, due to the fact that there are so many separate networks, information sharing isn't as efficient as it could be. JIE is designed to take the separate networks and collect these into a shared architecture, to be fully realized between 2016 and 2020. The research presented here focuses specifically on Mobile Solutions.


Unclassified

Joint Information Enterprise – End State

- Defensibility/Redundancy/Resiliency
- Federation/Shared Infrastructure
- Enterprise Services
- Identity Access Management
- Cost: ????

Source: "Achieving DoD IT Effectiveness," March, 2012, p. 9

Unclassified

11

Figure 6-3: The Joint Information Environment (JIE) is an IT SoS, intended to be a central information sharing solution for the improvement of DoD's ability to share information – not just between services, but also with industry partners and other government agencies.

6.1.4 Change Options in the BMDS

The first evolution increment considered is the inclusion of the Terminal High Altitude Area Defense (THAAD) interceptor battery in the BMDS in 2008. The BMDS program was officially launched in 2004, with the deployment of five long-range Ground-based Midcourse Defense (GMD) interceptors at Fort Greely, Alaska. At that point in time, the operational capabilities of the SoS were limited. In terms of intercepting capabilities, the SoS featured GMD interceptors, accompanied by the deployment of PAC-3 interceptors for short-range defense and the Aegis SM-3 interceptors for medium-range defense of the terminal segment (near US territory). In 2008, the first THAAD (Terminal High Altitude Area Defense) interceptor battery (land-based) was added to the BMDS. This new interceptor provided the BMDS with additional capability to intercept and destroy ballistic missiles inside or outside the atmosphere during their terminal phase of flight. One of the key enablers for the inclusion of this constituent system was the Command and Control Battle Management System (C2BMC), which provided the communication and data-management backbone for easily *integrating* THAAD within the current command and control nodes of the SoS.

In this evolution increment, the C2BMC system is the key path enabler that allowed for the "including new interceptor" change mechanism, as shown in Figure 6-4. Behind the design of the C2BMC is the design principle of *integrability*. This system in fact includes many technical features that ensure an easy and secure integration of new constituent systems in the BMDS. For instance, Tactical Datalink 16, a military data exchange network also used by NATO, allows for exchange of tactical pictures in near real time, as well as text messages and voice messages. It is an important tool for ensuring interoperability and it is critical for a layered defense. In addition to Datalink 16, Extremely High Frequency (EHF) satellite communications is a key element that enables the creation of an integrated network, thereby facilitating successful engagements via timely and accurate data sharing. Global Engagement Manager (GEM) allows for the integration and coordination of information, ready to be used by decision makers. It can calculate a common threat track from multiple sensors through data fusion, with sufficient data accuracy and timeliness for successful engagement. Finally, multiple and diverse paths in the Communication Network combat sensor outage or jamming, while encryption devices, routers and switches (each with specific access control lists – ACLs) further protect the internal systems and allow only identified and approved users and systems to access the C2BMC data.

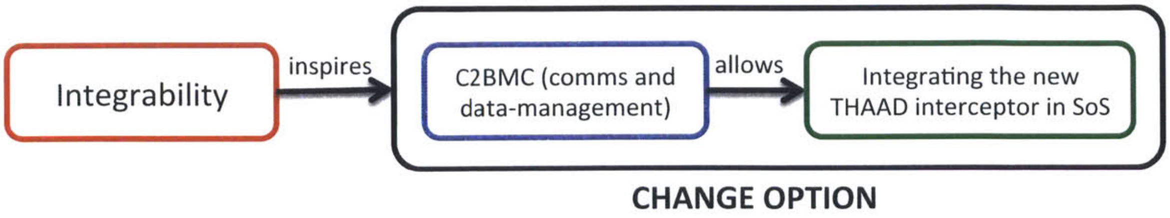

Integrability
inspires
C2BMC (comms and data-management)
allows
Integrating the new THAAD interceptor in SoS
CHANGE OPTION

Figure 6-4: Change option for the inclusion of THAAD system is linked to integrability.

The second evolution increment of the BMDS discussed here is the Phase Adaptive Approach (PAA) initiative, announced by President Obama in 2009 (Collina, 2013). PAA was conceived to enable the BMDS to extend its capabilities in European territory in order to deal with the threats posed by Iranian short-range and medium-range ballistic missiles to U.S. assets, personnel and allies. The first deployment of a ballistic missile defense asset in Europe was in March 2011, when the USS Monterey ship was placed in the Mediterranean Sea. This ship is a guided-missile cruiser of the Ticonderoga class, equipped with a sophisticated Aegis radar system designed to detect ballistic missiles. In terms of detection and tracking, the SoS used sea-based sensors mounted on the ship, as well as a forward-based X-band radar on European land (the first PAA radar was deployed in Turkey in late 2011). This strategy resonates with the design principle of *decentralization*: the presence of highly mobile Aegis BMD ships and other globally transportable systems (such as X-band Radar or THAAD) allowed for reallocating capabilities (intercepting, tracking, etc.) and resources to European bases. This flow is shown in Figure 6-5.

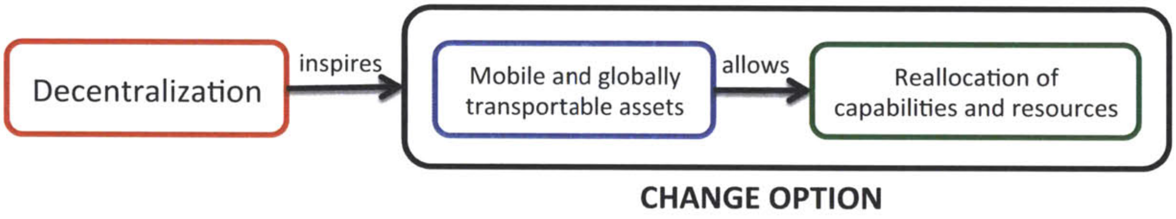

Decentralization
inspires
Mobile and globally transportable assets
allows
Reallocation of capabilities and resources
CHANGE OPTION

Figure 6-5: Change option for PAA evolution increment linked to the design principle of decentralization.

The PAA, as agreed upon by the U.S. and its allies, is a phased approach characterized by more than one increment. Although possible to deploy assets across various geographic locations in Europe from the start, it was decided to allow for some *slack* by fielding ships solely in the Mediterranean Sea during the first phase of the approach. By agreement, though, the U.S. would be able to deploy more assets "ashore" in the European countries of Romania and Poland (as it is currently planned to happen by 2018). This potential option expanding the system in terms of geographic coverage – but also of capabilities – is depicted in Figure 6-6. When, and if, executed, it will enable another significant evolution increment for the SoS.

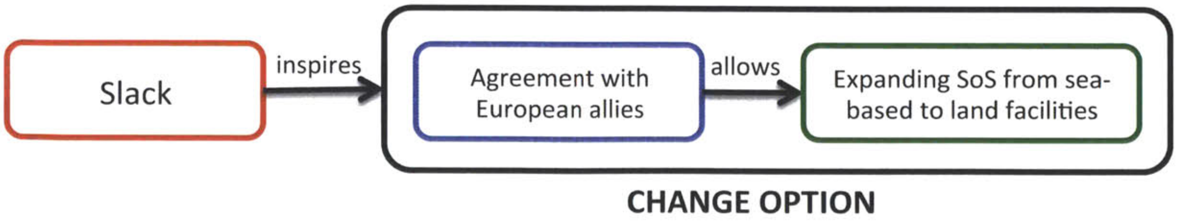

Slack
inspires
Agreement with European allies
allows
Expanding SoS from sea-based to land facilities
CHANGE OPTION

Figure 6-6: An example of instantiation of the design principle of slack for the Phase Adaptive Approach case.

6.1.5 Change Options in the DISA JIE

In fiscal 2014, DISA will begin offering mobile services as a subscription-based service, taking advantage of commercial-carrier infrastructure and providing entry points for classified services. Mobility at the enterprise level is being architected from the start, with consideration of joint information environments and providing efficiencies early on to create interoperability. The overall goal is to ensure that mobile devices, apps, email and other functions – as well as wireless networks that support them – can operate securely regardless of the environment, and can adapt to rapidly changing technology and scale to accommodate increasing numbers of users. In order to achieve this goal, it is important that new devices are easily integrated into the network, which is why SoS architects designed system interfaces for commonality and compatibility, allowing a wide variety of current and future devices to be added as the SoS evolves. These decisions embody the design principle of *integrability*, leveraging commonality and compatibility to allow a diversity of devices to be integrated into the system, even as technology changes and new devices emerge.

Furthermore, the design principle of *decentralization* can be associated to the choice of two components: the Mobile Device Management (MDM) system and the Mobile Application Store (MAS). The MDM provides for decentralized capability for policy enforcement and permissions, by distributing this capability at multiple DISA enterprise computing centers rather than a single center. The MAS, operating with MDM, can deliver, update, and delete applications on mobile devices without the end user returning the device to a centralized location for service. This means that every user is endowed with the change option of evolving his or her device as needed.

Finally, the design principle of *scalability* was also used in the DISA JIE Mobile Solution evolution increment. The mobility services used – both unclassified and classified – are scalable to accommodate increasing (or decreasing) numbers of users. The SoS architecture uses commercial carriers for providing subscription-based services. Commercial carriers and other unclassified access networks provide controlled connectivity between users and mobile enterprise. Subscription-based services are scalable; both up and down as needs change. As such they represent a path enabler in the change option illustrated in Figure 6-7. Operational capability is expected to grow with subscription-based services up to 100,000 devices – from 1,500 initial devices.

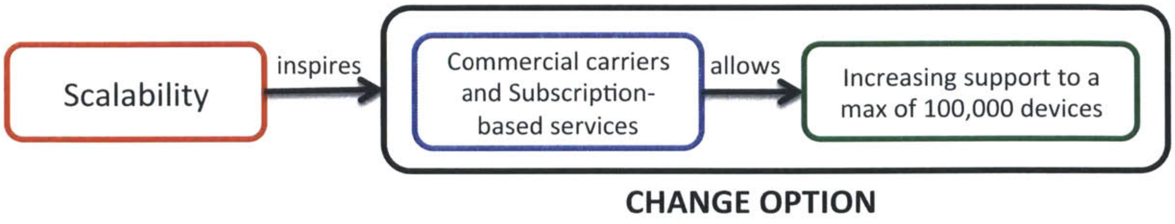

Scalability
inspires
Commercial carriers and Subscription-based services
allows
Increasing support to a max of 100,000 devices
CHANGE OPTION

Figure 6-7: Scalability-inspired change option for future evolution increments of DISA JIE Mobile Solutions program.

6.1.6 Architect's Intent

An interesting outcome from interactions with practitioners (SoS architects and analysts) is the fact that, from a practitioner's standpoint, all change options fall within certain categories of *intent* – i.e., what is the change option going to be used for? Considering the full set of options found in the investigation, and using the input of practitioners, six preliminary intents were delineated:

- Desire to *add new* constituent system
- Desire to *add more* of existing constituent system
- Desire to *replace* or *upgrade* capabilities of existing constituent system
- Desire to *add more* of existing function
- Desire to *change* the way in which a *function is performed*
- Desire to physically *relocate* resources/capabilities

The following subsections present further examples of changes in military systems, as seen through the lenses of the first two of the above architects' intents.

6.1.6.1 Desire to Add More of Existing System

Defense Support Program (DSP). The possibility of detecting and tracking ballistic missiles using the heat signature such missile generate when they are launched (as well as infrared signals from their plumes) can be dated to shortly after the Second World War. In fact, it was in such an environment that scientists for the RAND Corporation started to investigate the possibility of the development of an infrared warning satellite. This led to the ideation of the Missile Defense Alarm System (MIDAS), and of its more successful successor: the Defense Support Program (DSP). The idea behind the DSP was to achieve a constellation of three satellites that would enable coverage of the Atlantic, Pacific, and Eurasia. The first of those satellites was launched in the early 70's and it was placed in a strategic location so to monitor Soviet and Chinese missile launches. In addition to this first satellite, two more have been launched later on to form the wanted constellation over Atlantic, Pacific and Eurasia. Eventually, in addition to the three pre-conceived satellites, a fourth satellite was added, thereby creating a new European station. The initial architecture composed of satellites and ground stations (used to control the satellites and receive the data they collect) was designed such that a new constituent system could be easily integrated into the SoS (design principle of integrability).

BMDS – Aegis BMD. As discussed above, in September 2009, President Barack Obama announced that the U.S. was going to seek a Phased Adaptive Approach (PAA) to missile defense in Europe. PAA would have enabled the BMDS to extend its capabilities in European territory in order to deal with the threats posed by Iranian short-range and medium-range ballistic missiles to U.S. assets, personnel and allies. The first deployment of a ballistic missile defense asset in Europe was in March 2011, when the USS Monterey ship was placed in the Mediterranean Sea. In terms of detection and tracking, the SoS used sea-based sensors mounted on the ship, as well as a forward-based X-band radar on European land (the first EPA radar was deployed in Turkey in late 2011). Just two years later, by 2013, the number of Aegis BMD ships increased from one to twenty-nine, enabling the SoS to rapidly and significantly *scale up* its interception and detection capabilities (design principle of scalability). In future years (FY 2015-2017), the US Navy plans to have up to 32 Aegis BMD ships.

6.1.6.2 Desire to Add More of Existing Function

Defense Support Program (DSP). The possibility of detecting and tracking ballistic missiles using the heat signature such missile generate when they are launched (as well as infrared signals from their plumes) can be dated to shortly after the Second World War. Initially, scientists and engineers worked on the development of the Missile Defense Alarm System (MIDAS). Later on, a major decision was made to move to higher geosynchronous orbits (as opposed to the 2,000 miles above Earth of MIDAS satellites), which led to the development of the Defense Support Program (DSP). The original design of the DSP was a constellation of three satellites that would enable coverage of the Atlantic, Pacific, and Eurasia. After having successfully completed the launch of the third satellite, it was decided to launch an additional one, so to constitute a four-satellite operational constellation covering Pacific, Atlantic, Europe and Eurasia. Since the very first launch, the capabilities of DSP satellites have been enhanced. The first model of the satellite, which encompassed the first four flights, had 2000 detectors. Later models of the satellite (e.g., DSP-1, first orbited in 1989) added 4,000 more detectors, for a total of 6,000 detectors per satellite (design principle of scalability). The addition of more of the same detecting function provided far more accurate estimates of the coordinates associated with missile launches – an improvement intended to allow DSP to provide more precise information in the event of a nuclear exchange with the Soviet Union. The newer model was also harder to jam, as it could detect infrared radiation from two different parts of the electromagnetic spectrum.

BMDS – Boost intercept capabilities. With the current capabilities of the BMDS, hostile missiles can be mainly attacked during the intermediate and terminal phase of flight. An important goal of the Missile Defense Agency (MDA) is to add more intercepting capabilities for the boost phase of a missile. In order to provide more flexibility and targeting opportunities, the MDA plans to develop and test several new technologies designed to intercept and destroy ballistic missiles during the ascent phase of flight. An interesting route that the MDA has considered is to adapt the existing laser beam technology developed by DARPA in the 1980s (design principle of resourceful exaptation) onto an aircraft (a modified Boeing 747-400F), in order to increase (boost-phase) interception capabilities. The resulting constituent system – the Boeing YAL-1 Airborne Laser Testbed weapons system – would be integrated within the larger BMDS

SoS and be able to both visually detect and attack hostile missiles in their boost phase. Another avenue that the MDA is considering to increase the BMDS functionality with regard to intercepting missiles in their boost phase is the possibility of leveraging Unmanned Aerial Vehicles (currently considering the Predator UAV) [mimicry] so to integrate them with space assets (e.g., STSS), in order to achieve larger over-the-horizon sensor netting. This would enable the engagement zone of Standard Missile-3 interceptors (which can currently only intercept in midcourse) to be extended to the boost portion of a missile's trajectory.

6.2 Planned Adaptation in Policymaking

Planned adaptation is an example of coping with uncertainty (especially that associated with knowledge) in the realm of policymaking. It is motivated by the fact that, in principle, one would want regulatory programs to be based on the current state of the world – i.e., on current knowledge and circumstances. However, in many fields in policymaking, this is rarely the state of the art. (Eicher et al., 2012)

The concept of planned adaptation resonates well with the idea of conceptualizing systems that feature the possibility to change at future points in time (i.e., change options), so that they are able to adjust to the unfolding of uncertainty and the materialization of new knowledge. The idea behind planned adaptation is to be able to (1) revise rules when relevant new knowledge appears, and (2) take steps to produce such improved knowledge (McCray et al., 2010). Planned adaptation is composed of four major activities: prepare, discriminate, observe, and adapt. Prepare is concerned with understanding and framing the problem; discriminate with selecting best action at the present stage; observing with interacting with the real world and the policy in action so to gain more knowledge (i.e., learning); and adapting with taking action based on this newly acquired knowledge, in order to improve the state of the world. This concept is very close to a feedback loop in engineering control theory. A sensor asserts the state of the world (and any distance there may be between the current state of the world and the desired state of the world), and a controller acts to change the current state of the world to bring it closer to the desired state of the world.

In the realm of policymaking, the controller is the policy maker. In order for the policy maker to make the decision to adapt a given policy to a new state of knowledge or needs (i.e., via a *change mechanism*), the right *path enablers* must be in place. In other words, for successful planned adaptation, institutions and apparatuses must be in place so that changes can be made in a timely and efficient manner. However, if – as discussed – it is hard to design engineering systems that are able to adjust to new environments, it is even harder in most policy cases. In fact, in the policymaking realm some problems are even more exacerbated than in the systems engineering domain. For example: there are several delays between all the activities involved in mandating regulations; it is hard to understand what needs to be sensed and how to sense it; social complexities related to repercussions of policy decisions on the public are more pronounced. Hence, for these and more reasons, the tendency is to lock into certain policy situations, which oftentimes may not be favorable.

6.2.1 The Case of Particulate Matter Regulation

A great example of a successful planned adaptation undertaking is the US Air Quality regulation. Beginning in 1980, Congress has mandated regular reviews of existing standards, based on new knowledge assessments on the effects of particulate matter and other pollutants to health. Such knowledge assessments are then linked to a policy assessment, which determine whether (and in what way) old standards must be changed. The regulation of US Air Quality is ultimately in the discretion of the Environmental Protection Agency (EPA) under the Clean Air Act, which enables the agency to set National Ambient Air Quality Standards (NAAQS). The planned adaptation resides in the fact that the standards are subjected to fresh scientific review every five years. The review process is managed by a semi-independent entity, the Clean Air Scientific Advisory Committee (CASAC), which ensures the scientific integrity of the updated knowledge assessment, and it proposes adjustments to policy makers at the EPA. Hence, CASAC – along with mandatory periodic reviews, government incentives, and other – is one of the *path enablers* for bringing about changes in regulation (that better adjust to current knowledge and circumstances). The general idea of the change option set up in this case is shown in Figure 6-8.

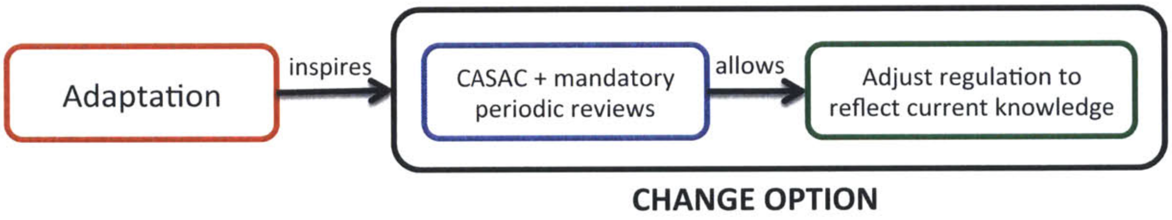

Adaptation
inspires
CASAC + mandatory periodic reviews
allows
Adjust regulation to reflect current knowledge
CHANGE OPTION

Figure 6-8: The change option set up in the case of Particular Matter regulation.

As a matter of fact, the regulation of particulate matter (PM) has been particularly successful, accounting for an approximate 90% the Clean Air Act benefits (McCray et al., 2010). In this case, the periodic reviews of the current scientific assessments have led to great progress not only in the assessment of the risks associated with the presence of the pollutant, but also in the understanding of the cause of adverse health effects. In fact, initially it was thought that the cause of the problem was the visible black smoke (measured as Total Suspended Particulate). After repeated assessments and investigations (strongly incentivized by the government), it was shown that it is mostly smaller particles that are associated with adverse health effects. As a consequence, the standard was changed to particles less than 10 µm, and ultimately 2.5 µm. Without planned adaptation, there was a significant risk to be trapped in the incorrect theory that black smoke was the sole source of adverse health effects.

6.2.2 Path Enablers for Planned Adaptation

Particulate Matter regulation is not the only successful example of planned adaptation. McCray et al. (2010) discuss a variety of them: the review of aircraft safety certification under the Department of Transportation (DOT) and the Federal Aviation Administration (FAA); the monitoring of drug safety under post-marketing surveillance and review from the Food and Drug Administration (FDA); the National Academy of Science (NAS) review of nutrition requirements for animals, under the United States Department of Agriculture (USDA). When analyzing across successful cases, some general path enablers seem to appear frequently across cases, as discussed below.

In general, it is very hard to adapt and evolve when one has to rely on self-evaluation and self-correction. Hence, an important path enabler associated with planned adaptation has turned out to be the existence of *knowledge assessment institutions* that are *independent* and *autonomous* from the regulating ones. For example, in the case of Particulate Matter, the EPA set up a semi-autonomous body (the CASAC) that would ensure to reassess the knowledge base behind the setting of Air Quality Standards every five years. Furthermore, still relevant for the regulation of Air Quality, another autonomous body (the Health Effects Institute) was appointed to reassess the knowledge base of the Harvard Six Cities study (Health Effects Institute, 2006). Another successful planned adaptation case that features a third independent and autonomous body is aircraft safety certification. In this case, the independent body is the National Transportation Safety Board (NTSB), which investigates accidents in order to ensure that previously undetected causes of safety failures are discovered and addressed. This analysis, then, informs the FAA's policymaking process. To augment the effectiveness of the abovementioned path enabler (i.e., establishing independent knowledge assessment institutions), another useful path enabler is that of having *mandatory periodic reviews* of the current state of the knowledge, like in the case of PM regulation.

Another path enabler for effective planned adaptation has demonstrated to be a sound *incentive structure* with regard to *revealing (new) knowledge*. An exemplary case in this regard is the nuclear Non-Proliferation Treaty (NPT), where the International Atomic Energy Agency (IAEA) was instituted to ensure that information be revealed about the (in)existence of nuclear weapons programs in all participating countries. Furthermore, this path enabler can be instantiated by incentivizing the act of questioning the current status quo in the field and paying attention to minority reports. Failing to do so has shown to lead to poor policy outcomes – e.g.: (U.S. Senate, 2004).

6.3 Summary

This chapter has discussed an empirical investigation of change options. It presented results from an investigation of evolvability in military SoS, as well as planned adaptation in the realm of policymaking.

The first part of the chapter focused on studying evolvability in military SoS. Some examples of change options that were used during past and current evolutionary increments were discussed for military SoS, such as the Ballistic Missile Defense System and the DISA Joint Information Enterprise. Furthermore, a link between the entirety of the change options found in the larger research effort (only an excerpt of which was presented here) and specific architect's intents was discussed. Examples of further change events in military systems as related to two of these intents were given.

Lastly, the chapter has closed with a discussion on the concept of planned adaptation in policymaking and its link to change options. The regulatory system behind some important regulations (such as Ambient Air Quality standards and aircraft safety certification) was investigated through the lens of the change option construct. This led to the observation of some path enablers that are common in the practice of effective planned adaptation.

7 Ility-Driving Element Analysis

"We are permanently trapped in the duality between what one can measure and what one wants to know... 'I cannot predict the future; nevertheless, I will try to predict the future'."

– Frank R. Field, III (2014)

This chapter introduces the Ility-Driving Element Analysis (IDE Analysis), a structured approach for the generation, evaluation and selection of ility-driving elements during the conceptual phase of system design. Using this approach, it is possible to formally think about the introduction of design- and enterprise-level elements that can drive the emergence of ilities in complex systems. Although it is impossible to control uncertainty, the existence of such elements in the design of a system can enable better management of it. IDE Analysis is a generalization of the activities performed in step 5 of the SAI method (see chapter 3), and it is important to underline that it is performed with a baseline design of the system in mind. It is scalable in effort, and, as for many tasks performed in conceptual design, its completion involves much creativity and expertise. This chapter discusses the flow and activities of the IDE Analysis from a conceptual standpoint, and it also presents applications of some of the proposed concepts to the MarSec SoS case study.[35]

7.1 Origins and General Framework

As discussed in chapter 2, the introduction of ility-driving elements is most related to the part of the conceptual phase engineering effort that deals with strategic level thinking. The foundational concepts behind IDE Analysis can be traced back to McManus and Hastings (2006), where the framework in Figure 7-1 – relating the problem of uncertainty to desired ility behaviors – was introduced. The essence of this framework is encapsulated in the following two writs: "<uncertainty> causes <risk> handled by <mitigation> [resulting] in <outcome [ility]>" or "<uncertainty> causes <opportunity> handled by <exploitation> [resulting] in <outcome [ility]>." Uncertainty, as they characterize it, has been described in chapter 2. Risks and opportunities are consequences of the uncertainties on (the value delivery of) the system. Risk is

[35] Although the applications presented here are representational in nature (since they were carried out by a team of researchers, and not practitioners), they have received face validity over the course of multiple interactions with a team of systems architects and engineers working on the design of a real MarSec SoS.

associated with downside uncertainty, while opportunity with upside uncertainty. McManus and Hastings (2006) also discuss how, in general, "risk can be quantified by considering (probability of problem)×(severity of problem) ", and opportunity by considering "(probability of an event)×(value of the event)." They also discuss general types of risks and opportunities encountered in system design (see Figure 7-1). Mitigations and exploitations are technical or programmatic strategies that one can use in order to "avoid or manage risk, and/or exploit opportunities." A list of common strategies used for aerospace systems is illustrated in Figure 7-1 (and discussed in detail in their work). The ility outcomes are consequences of implementing these strategies and can be thought of as "the desired attributes of the system that quantify or at least characterize its interaction with uncertainties."

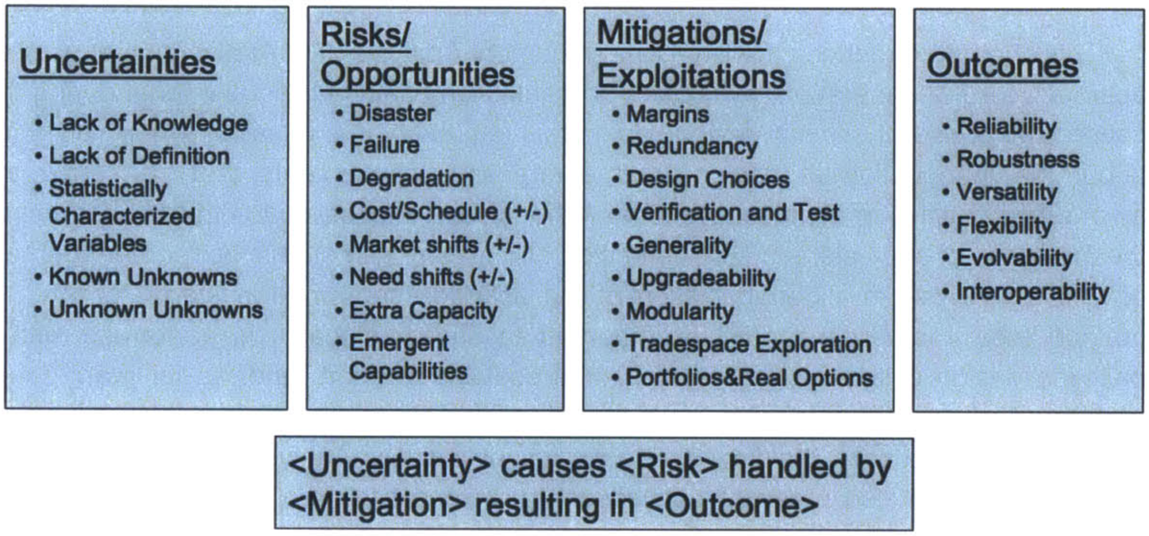


Figure 7-1: Framework that relates the existence of uncertainty to desired ility outcomes (McManus and Hastings, 2006)

The general concepts behind IDE Analysis are the same as the ones put forth in McManus and Hastings (2006). The starting point is the existence of uncertainty, which results into either risk or opportunity (or both). Such uncertainty can be parameterized into perturbations, as discussed in chapter 4. Uncertainty also drives the desire for specific ility-related behaviors in systems. In the literature, there are strategies (like the ones McManus and Hastings discuss) and heuristics associated with the emergence of different ilities. These strategies correspond to the design principles briefly discussed in chapter 5 (and further discussed in the following section). Hence, given a list of ilities of interest (see step 3 of the SAI method), it is possible to derive one of design principles. Finally, from a list of relevant design principles and parameterized uncertainty (in the form of a perturbation set), it is possible to derive appropriate ility-driving elements. These are derived as instantiations of design principles, in light of a certain perturbation. IDEs constitute the link back to the emergence of desired ility properties throughout the lifecycle of a system. This flow is illustrated in Figure 7-2.

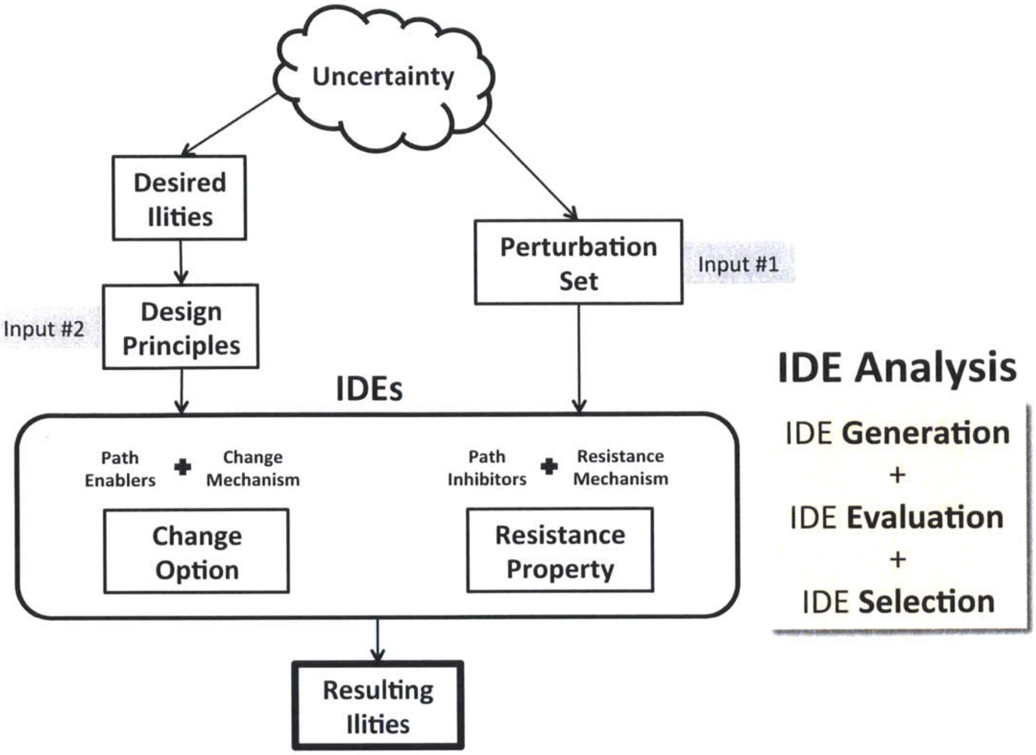


Figure 7-2: General flow from the existence of uncertainty to the emergence of counteracting ility behaviors. The IDE Analysis is aimed at identifying (i.e., generating, evaluating, and selecting) IDEs, given relevant design principles and perturbations.

7.2 Inputs to IDE Analysis

The first activity in IDE Analysis is the generation of a list of candidate IDEs for implementation in the system. There are two major inputs to such task: a list of perturbations and a list of relevant design principles. The former is an outcome of step 2 in the SAI method (see chapter 4); the latter of the desired ilities identified in step 3 of the SAI method. Both originate from the existence of uncertainty system design, as shown in Figure 7-2. Additional inputs may be, for example, a Design Structure Matrix (discussed in chapter 2), which can be useful insofar as identifying latent path variables.

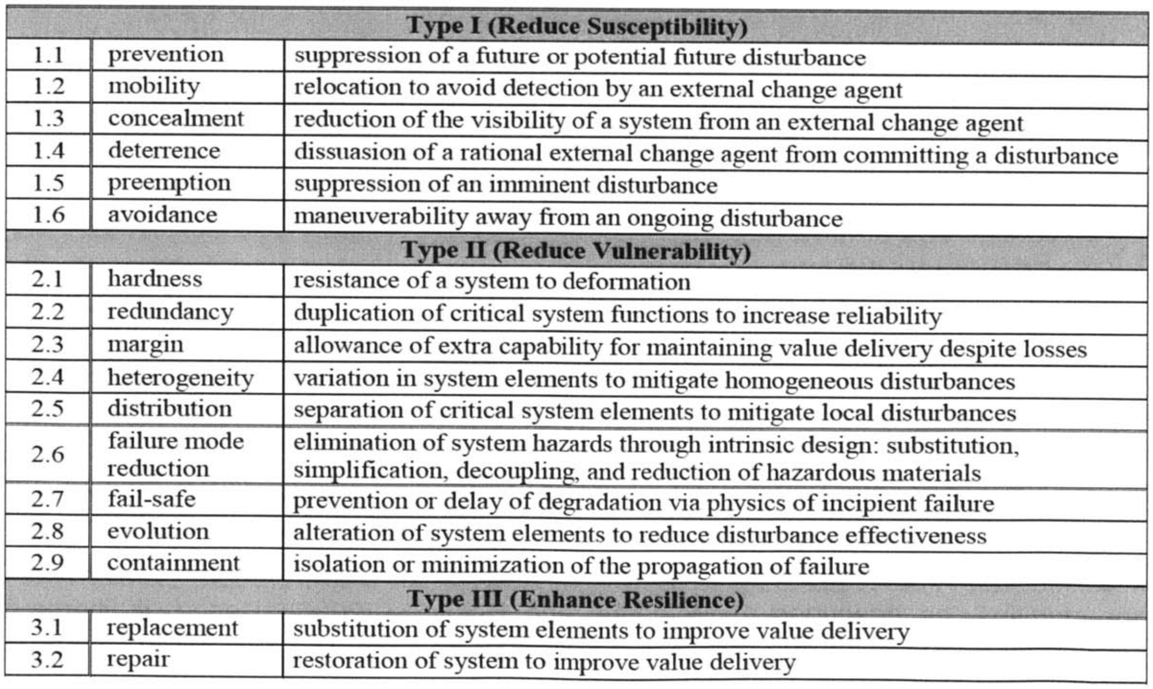
7.2.1 Perturbations

The anatomy of perturbations in the conceptual design effort has been discussed in chapter 4. Chapter 4 also described step 2 of the SAI method, i.e.: a way of producing a set of perturbations directly from identified uncertainty categories. Perturbations can be (1) spontaneous or imposed changes in design (λ^D), (2) changes in context (λ^P), or (3) changes in needs (λ^N). These eventually cause changes in value delivery. While the first type (λ^D) is usually associated with negative impacts on value delivery (e.g., failure of an engine), λ^P and λ^N can have negative or positive (or both) impacts – risks and opportunities, as described in McManus and Hastings (2006). Hence, if filtered by consequence (positive or negative), the perturbation set can be thought of as the risk space or opportunity space a designer is faced with. It is important to underline here that the delineation of such spaces (on which the rest of the analysis lie) is a function of what the designer/practitioner/decision maker envisions as most relevant uncertainty. As such, it is an inevitably subjective process, subject to cognitive biases and limited information (chapter 8 discusses these issues). The willingness to accept the potential error that may be introduced by choosing a certain list of perturbations (or of indicators for the suitability of IDEs) is a policy decision one must unavoidably make, if he or she deems such analysis relevant.

7.2.2 Design Principles

Design principles can be thought of as "guiding thoughts [for design] based on empirical deduction of observed behavior or practices that prove to be true under most conditions over time." (Wasson, 2006) They are heuristics that serve to help intentionally create desirable properties in systems, and designers often draw inspiration from them (as illustrated in chapters 5 and 6). Chapter 5 has introduced some of the salient design principles associated with evolvability. Many of these principles can be extended to similar ilities as well (e.g., flexibility, adaptability, etc.). It is important to note here that a considerable literature of design principles exists also for ilities such as robustness or survivability. For example, chapter 4 discusses how Richards (2009) proposes a definition for survivability and identifies three general design strategies for survivability: (I) susceptibility reduction, (II) vulnerability reduction, and (III) resilience enhancement. Associated with these strategies, he generates a list of seventeen design principles, listed in Table 7-1.

Table 7-1: Validated set of survivability design principles (Richards, 2009).



Type I (Reduce Susceptibility)		
1.1	prevention	suppression of a future or potential future disturbance
1.2	mobility	relocation to avoid detection by an external change agent
1.3	concealment	reduction of the visibility of a system from an external change agent
1.4	deterrence	dissuasion of a rational external change agent from committing a disturbance
1.5	preemption	suppression of an imminent disturbance
1.6	avoidance	maneuverability away from an ongoing disturbance
Type II (Reduce Vulnerability)		
2.1	hardness	resistance of a system to deformation
2.2	redundancy	duplication of critical system functions to increase reliability
2.3	margin	allowance of extra capability for maintaining value delivery despite losses
2.4	heterogeneity	variation in system elements to mitigate homogeneous disturbances
2.5	distribution	separation of critical system elements to mitigate local disturbances
2.6	failure mode reduction	elimination of system hazards through intrinsic design: substitution, simplification, decoupling, and reduction of hazardous materials
2.7	fail-safe	prevention or delay of degradation via physics of incipient failure
2.8	evolution	alteration of system elements to reduce disturbance effectiveness
2.9	containment	isolation or minimization of the propagation of failure
Type III (Enhance Resilience)		
3.1	replacement	substitution of system elements to improve value delivery
3.2	repair	restoration of system to improve value delivery

7.3 Generation of IDEs

This section describes the fist main activity in IDE Analysis: generating an extensive list of possible ility-driving elements (i.e., change options and resistance properties) to consider for the design of the system. The main brainstorming method proposed is Design Principle to Perturbation (DPP) mapping. Alternative brainstorming activities – leveraging cause-effect mapping or DSM-driven techniques – are discussed. After the brainstorming phase, path variables are matched to mechanisms to form change options or resistance properties.

7.3.1 Design Principle to Perturbation Mapping

The goal of this activity is to generate an extensive list of ility-driving elements (within the participants' time- and effort-related possibility). The process of generating IDEs starts by mapping relevant design principles to perturbations. This consists of brainstorming instantiations of design principles that can (partially) address either the value loss caused by a negative perturbation (risk) or the potential value gain associated with a positive perturbation (opportunity). An aid for the completion of this task is the Design Principle to Perturbation (DPP) matrix shown in Figure 7-3, wherein design principles are listed as rows and perturbations (both shifts and disturbances) as columns. The empty cells in the matrix are filled with instantiations of design principles in the form of both path variables (enablers or inhibitors) and mechanisms (change or resistance). For example, the design principle of *redundancy* (related to survivability – see Table 7-1) can inspire the path inhibitor of 'redundant optical sensor' in the case the perturbation 'loss of situational awareness' (due to an optical sensor malfunctioning) occurs. This path inhibitor enables the resistance mechanism of 'impeding loss of situational awareness' for a UAV. More than one entry per cell can be entered. This – together with the fact that there may be many design principles and many perturbations – can lead to very large matrices.

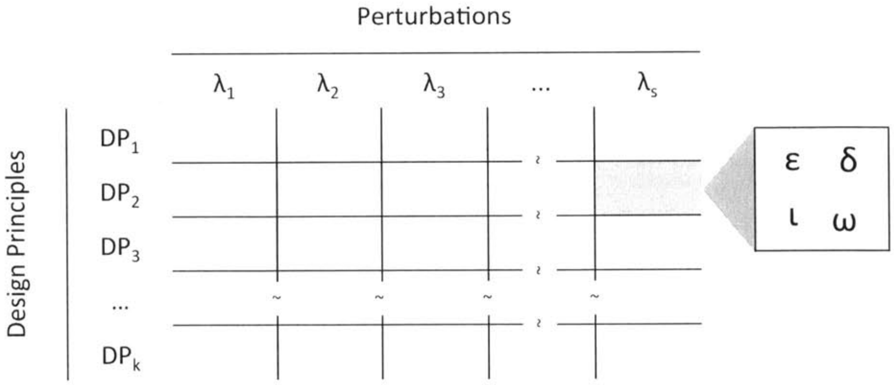


Figure 7-3: Design Principle to Perturbation mapping matrix. Cells contain instantiations of design principles in the form of path enablers (ε), path inhibitors (ι), change mechanisms (δ) or resistance mechanisms (ω).

An example application of this structured brainstorming technique to the case of the MarSec SoS is illustrated in Table 7-2. This table contains an excerpt of the actual information gathered performing this activity for a subset of five perturbations and nineteen design principles. In the larger matrix for the study, fifteen perturbations were considered, as well as more than twice as many design principles. Furthermore, the actual matrix contains more than one entry per cell, whereas only one has been reported here, given the constrained space. It is clear that a design principle does not have to map to every perturbation, although it may. The general flow, as discussed above, is to focus on one design principle-perturbation pair at the time, and think about possible instantiations of the design principle – in the form of a path variable or a mechanism – that would help addressing the perturbation. For example, for the pair 'redundancy'-'decreased situational awareness' (due to sensor failure), the path inhibitor of a 'redundant sensor' would resist a change in performance due to sensor loss.

Table 7-2: Excerpt from the DPP matrix generated for the MarSec SoS case.

Design Principle	Perturbation				
	λ_1	λ_2	λ_3	λ_4	λ_5
	Boat arrival rate variation	**Comms jamming**	**UAV loss**	**Variation in situational awareness**	**Information attack**
Prevention		Powerful transmitters	Periodic maintenance	Water repellant windshield – Waterproof cameras	
Mobility		Vary velocity to get closer to signal		Enhanced control system	
Concealment			Stealth design		Higher Altitude
Avoidance		Increase transmitter/receiver power			Vary flight paths and speeds
Defensive Posture	Divide Area of interests between assets			Bring assets back to base promptly	
Deflection		Relay info to satellite or higher altitude asset	Decoys for missile attack		
Reconfigurability	Use higher ratio of detection UAVs				
Scalability	Increase # of UAVs - # of operators	Knob controlling the gain on reception			Knob controlling the gain on reception
Hardness		Increased signal power	Armor against physical attacks	More stable structure	Double Authentication
Redundancy	Spare assets to put to use		Spare UAV	Multiple sensors	
Margin	Higher than needed number of operators per UAV	Higher than needed reception gain			
Heterogeneity	Divide area of interests between assets		Multi-role asset (e.g., detect + identify)		Satellite and Direct Links

	Perturbation				
	λ_1	λ_2	λ_3	λ_4	λ_5
	Boat arrival rate variation	**Comms jamming**	**UAV loss**	**Variation in situational awareness**	**Information attack**
Decentralization		Geographical Distribution			Distribute/decentralize authority
Containment		Knob controlling the gain on reception			
Stable Intermediate Instances		Training personnel in multiple tasks/ irregular ops	Go to pre-validated, stable design instance		Training personnel in multiple tasks/ irregular ops
Replacement	Change detection UAVs to higher coverage ones		Spare UAV	Change camera for bigger resolution and/or field of view	
Adaptation		Training personnel in multiple tasks/ irregular ops	Multi-role asset (e.g., detect + identify)	Enhanced control system	Training personnel in multiple tasks/ irregular ops
Targeted Modularity	Modular asset allocation so to increase # of assets in targeted areas			Modular payload bay for swapping camera	
Integrability	Extend SoS to include radar tower detection	Integrate satellite relay comms	Integration of new UAV		

7.3.2 Design Principle to Perturbation Cause Mapping

Perturbations, as defined in chapter 4, are operators on any of the three important spaces in the design effort – design, context, or needs. They describe the transition from one design, context, or needs instance to another, which in turn triggers a change in value delivery (i.e., value instance). As such, they are inherently concerned with the effects of an event, not the cause. However, as described in chapter 5, some ility-driving elements may be concerned with the prevention or avoidance of a perturbation. This is not explicitly captured in the DPP matrix in Figure 7-3 (although the causes of a given perturbation may well be in the mind of the practitioner), which is mostly concerned with the perturbation and its effects. For example, a 'UAV loss' perturbation can be due to both engine failure and enemy missile attack. In the former case, a 'spare UAV' can help recover from the perturbation; however, in the latter case, it is also possible to prevent the occurrence of the perturbation by implementing an 'agile control system' (that avoids the missile) or by adopting 'stealth technology'. A variant of the DPP matrix – the Design Principle to Perturbation Cause (DPPC) matrix – that takes into account potential causes of a perturbation (as brainstormed in step 3 of SAI method – see chapter 4) is illustrated in Figure 7-4. An example of how, for the same perturbation (UAV loss), there may be two different causes (physical attack and engine failure) resulting in two different instantiations of design principles ('armor' and 'spare UAV') is illustrated in Figure 7-5.

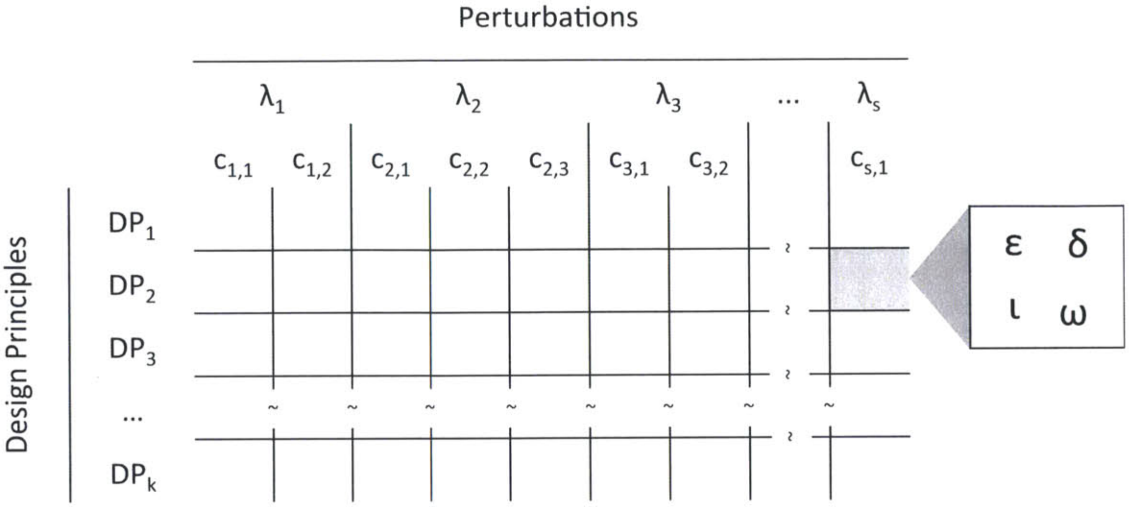


Figure 7-4: Design Principle to Perturbation Cause ($c_{i,j}$) mapping matrix.

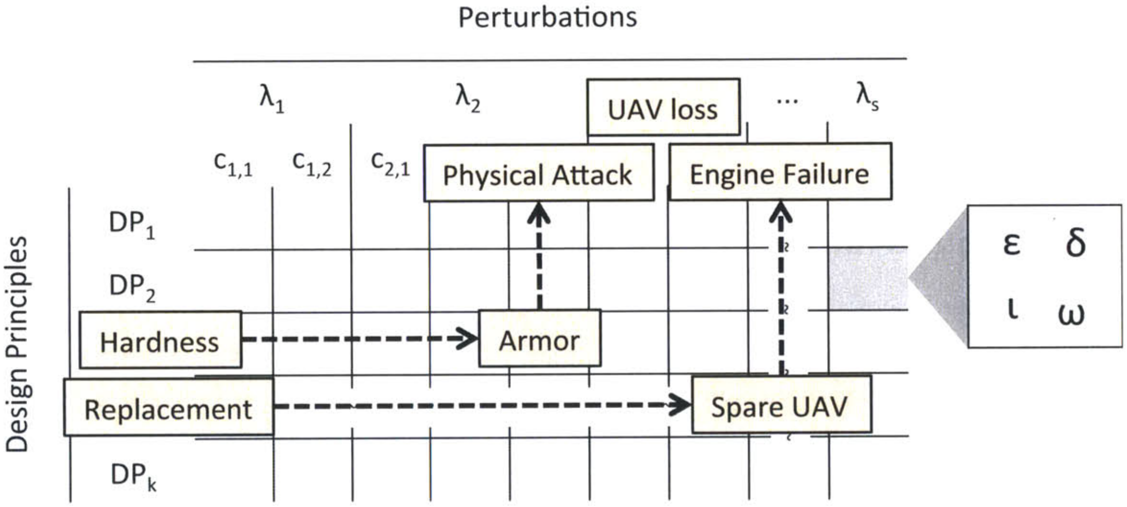


Figure 7-5: Example DPPC mapping for the case of the MarSec SoS.

7.3.3 Intervention Points in Cause-Effect Mapping

To the end of developing and applying "viability strategies" for Systems of Systems, Mekdeci (2013) discusses the cause-effect mapping: i.e., a causal diagram of perturbations' causes and effects. Such a diagram – a simple illustration of which is shown in Figure 7-6 for a perturbation in the context of a MarSec SoS – enables further development of the initial list of perturbations, and, very importantly, identification of relevant points of intervention. Cause-effect mapping can provide an additional layer of details regarding perturbations: it begins with a terminal event of particular interest (i.e., a perturbation in design/context/needs or a change in the performance/value delivered by a system), and it continues by tracing back possible causes to that event. Causal arrows link causes and effects. This activity is analogous to the generation of a causal loop diagram in System Dynamics (Sterman, 2000).

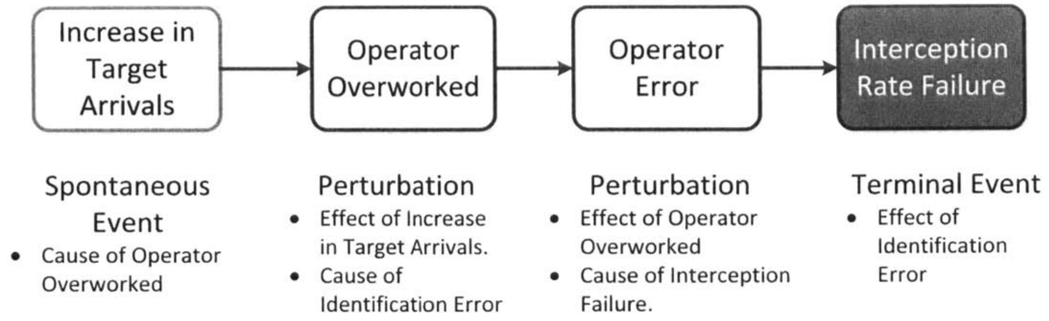

Increase in Target Arrivals
Operator Overworked
Operator Error
Interception Rate Failure
Spontaneous Event
• Cause of Operator Overworked
Perturbation
• Effect of Increase in Target Arrivals.
• Cause of Identification Error
Perturbation
• Effect of Operator Overworked
• Cause of Interception Failure.
Terminal Event
• Effect of Identification Error

Figure 7-6: Simple cause-effect mapping diagram (Mekdeci, 2013).

Cause-effect diagrams can grow large relatively rapidly. In his work, Mekdeci (2013) develops a larger cause-effect mapping diagram than the one shown in Figure 7-6, in order to identify strategies (analogous to design principles) for viable SoS. In fact, he points out how "the cause-effect mapping is useful for being able to highlight areas of intervention." These points of intervention lie between a cause and an effect. Within the context of IDE Analysis, intervention points can be used to brainstorm possible ways (i.e., IDEs) of avoiding the occurrence of an effect from a cause. For example, in the diagram in Figure 7-6, an increase in target arrivals causes operators to be overworked, which in turn causes errors; a possible way to reduce the likelihood of such occurrence would be to have a workforce buffer built in the system, or to design in such a way to incorporate human factors and enable UAV operators to easily deal with an increase in workload. In this case, reducing the likelihood of occurrence is a resistance mechanism, while the latter two are path inhibitors.

7.3.4 Latent Path Enablers and Path Inhibitors Identification

As mentioned in the beginning of this chapter, IDE Analysis is a generalization of the activities performed in step 5 of the SAI method (see chapter 3), and it is performed with a baseline design of the system in mind. It is likely that relevant path enablers or path inhibitors for certain change or resistance mechanisms of interest may already be in the baseline design of the system. Chapter 5 discusses and differentiates between path enablers/inhibitors that must be acquired, and those that exist already in the system: i.e., *latent* path enablers/inhibitors. For example, the existence of a satellite relay in the initial architecture can be one possible enabler of a change in the geographic segmentation of UAVs. Similarly, an SoS architecture with a plane with a high margin in coefficient of lift can survive the damage of a wing. Hence, higher than needed coefficient of lift is a latent path inhibitor in this case. A useful activity is to analyze the current baseline of the system in order to identify latent path variables. Some of the tasks that can be performed to this end – e.g., CPA or Sensitivity DSM – have been discussed in chapter 2.

7.3.5 IDE Formation

The activities mentioned above produce unstructured databases of path variables (enablers or inhibitors) and mechanisms (change or resistance). Upon completion of these activities, the next logical step is to discern among the four different entry types and sort them into lists: i.e., a list of path enablers, one of path inhibitors, one of change mechanisms and one of resistance mechanisms. With these four lists, it is possible to form ility-driving elements by matching compatible path enablers and change mechanisms (to form change options), or compatible path inhibitors and resistance mechanisms (resistance properties). In order to perform this activity, Change Option Formation (COF) and Resistance Property Formation (RPF) matrices can be used. These matrices – shown in Figure 7-7 – have path variables listed as rows and mechanisms as columns. If a path enabler is an enabler of a given change mechanism, the two are matched by entering the label of the option in the matrix. However, it may be the case that more than one path enabler is needed for the existence of an executable change mechanism (recall definition of a change option in chapter 5). In these cases, the same change option is entered for more than one path enabler, but always for only one change mechanism: e.g., $CO_1: \{\varepsilon_1 \wedge \varepsilon_5 \wedge \varepsilon_{23} \Rightarrow \delta_4\}$. The same logic applies for resistance properties. In the matrices, it may be useful to distinguish latent path enablers. It is important to note that the initial matrices can be augmented along the way by adding path variables that are needed by a mechanism, or by adding mechanisms that are related to path variables. The extent to which the matrices are let grow is in the discretion of the practitioner.

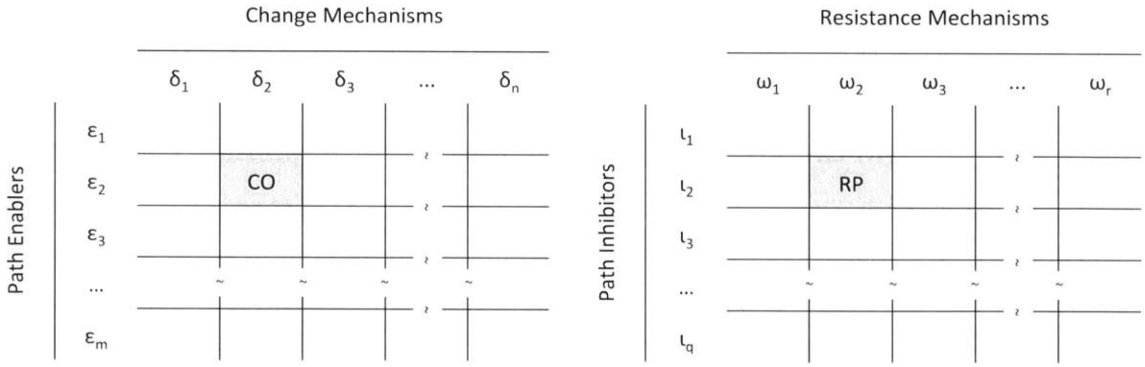


Figure 7-7: Change Option Formation (COF) matrix (left) and Resistance Property Formation (RPF) matrix (right). Matching a change (resistance) mechanism with one or more path enablers (inhibitors) can form a given option CO (property RP). In the case more than one path variable is linked to a mechanism, the same IDE is in more than one entry.

COF and RPF matrices can grow large relatively fast. This is the case for the MarSec SoS, where a long list of path variables and mechanisms was generated. In such cases, practitioners must be careful of being selective in the IDE formation process, so to end up with a list of IDEs that is a manageable space to explore (problems related with the rapid growth of the space are discussed in later sections). Table 7-3 shows an excerpt of a COF matrix for the MarSec SoS, which was selected because it shows an array of possible scenarios. First of all, there may be options whose change mechanism is linked to a single path enabler: e.g., having a 'workforce buffer' enables 'changing the number of operators per UAV.' Furthermore, the same conjunction of path enablers may be linked to more than one mechanism: e.g., having 'a modular payload bay on the asset' in conjunction with a 'different payload on the ground' can enable both 'replacing a feature an asset' and 'changing the task assignment.' Lastly, it is important to point out that some path enablers – like 'multi-role asset' (i.e., capable of detecting and intercepting) in this case – are latent in the baseline system.

Table 7-3: Excerpt of COF matrix applied to the MarSec SoS case.

			Change Mechanism					
			δ_1	δ_2	δ_3	δ_4	δ_5	δ_6
			Add asset to SoS	Change task assignment	Change geographic segmentation	Change # of operators per UAV	Change authority distribution	Add/replace feature to asset
Path Enabler	ε_1	Extra interception airplane	CO_1	CO_2				
	ε_2	Contract with aircraft supplier	CO_3					CO_4
	ε_3	Spare sensor		CO_5				CO_6
	ε_4	Long range UAV			CO_7			
	ε_5	Workforce buffer	CO_1, CO_8			CO_9, CO_{11}		
	ε_6	Dispersed comms network			CO_7			
	ε_7	Spare UAV	CO_8			CO_{10}, CO_{11}		
	ε_8	Multi-role asset		CO_{12}				
	ε_9	Central authority					CO_{13}	
	ε_{10}	Reserve budget	CO_3					CO_4
	ε_{11}	Satellite relay			CO_{14}		CO_{15}	
	ε_{12}	Common network interfaces	CO_1, CO_8	CO_2	CO_{14}			
	ε_{13}	Different payload on ground		CO_{16}				CO_{17}
	ε_{14}	Modular payload bay		CO_{16}				CO_{17}

Once the COF and RPF matrices are complete, it is possible to organize ility-driving elements into lists of change options and resistance properties, as shown in Figure 7-8. At this point, the list of IDEs may be extensive. If this is the case, the next activities are aimed at evaluating the various IDEs (as well as portfolios thereof), in order to inform a final selection of a set of ility-driving elements. If it is not, the identified IDEs can be considered directly for either (1) further analysis using modeling and simulation or (2) direct inclusion in the design of the system.

Change Options

CO_1	ε_1	$\Rightarrow$	δ_1
CO_2	ε_2	$\Rightarrow$	δ_2
CO_3	$\varepsilon_2 \wedge \varepsilon_{13}$	$\Rightarrow$	δ_3
CO_4	ε_4	$\Rightarrow$	δ_4
CO_5	$\varepsilon_1 \wedge \varepsilon_5 \wedge \varepsilon_8$	$\Rightarrow$	δ_4
...	...		...
CO_l	ε_m	$\Rightarrow$	δ_n

Resistance Properties

RP_1	ι_1	$\Rightarrow$	ω_1
RP_2	ι_2	$\Rightarrow$	ω_2
RP_3	$\iota_5 \wedge \iota_{13}$	$\Rightarrow$	ω_2
RP_4	ι_4	$\Rightarrow$	ω_3
RP_5	ι_{21}	$\Rightarrow$	ω_4
...	...		...
RP_p	ι_q	$\Rightarrow$	ω_r

Figure 7-8: Example lists of ility-driving elements.

7.4 Assessment of Ility-Driving Elements

The evaluative activities and metrics introduced here serve the purpose of providing a first order differentiation among ility-driving elements, so that a set of path enablers and path inhibitors may be selected for inclusion in the design of the system (or chosen for further – higher fidelity – analysis). Unlike ROA (see chapter 2), what is done here does not attempt to quantify the monetary value of a change option (or resistance property, for that matter). Rather, it is aimed at a "horizontal" analysis and evaluation, producing a general understanding of the value of different IDEs. Several indicators are presented, among which a decision maker can select what is most important, according to his or her preferences.

Like any other step in IDE Analysis, the evaluative one has a strong human-in-the-loop component to it, especially when no historical data exist on the object of evaluation. It is strongly advised that the assessments herein are performed with the aid of subject matter experts and decision makers. Furthermore, it is important to use structured assessment processes, such as the Delphi method (Rowe and Wright, 1999) – discussed in some detail in chapter 4.

During the assessments in the next subsections, the following spaces (delineated via the activities discussed so far) are considered:

- A set of perturbations: $\boldsymbol{\Lambda}\ s.t. |\boldsymbol{\Lambda}| = S$.
- A set of path enablers: $\boldsymbol{PE}\ s.t. |\boldsymbol{PE}\ | = M$.
- A set of change mechanisms: $\boldsymbol{CM}\ s.t. |\boldsymbol{CM}\ | = N$.
- A set of change options: $\boldsymbol{CO}\ s.t. |\boldsymbol{CO}\ | = L$.
- A set of path inhibitors: $\boldsymbol{PI}\ s.t. |\boldsymbol{PI}\ | = Q$.
- A set of resistance mechanisms: $\boldsymbol{RM}\ s.t. |\boldsymbol{RM}\ | = R$.
- A set of resistance properties: $\boldsymbol{RP}\ s.t. |\boldsymbol{RP}\ | = P$.
- A set of ility-driving elements: $\boldsymbol{IDE} = \boldsymbol{CO} \cup \boldsymbol{RP}\ s.t. |\boldsymbol{IDE}\ | = T = L + P$.

A generic subset of any of the above sets is indicated by a tilde (e.g., a subset of path enablers is: $\widetilde{\boldsymbol{PE}} \subset \boldsymbol{PE}\ s.t. |\widetilde{\boldsymbol{PE}}\ | = \widetilde{M} < M$).

7.4.1 IDE-level Assessment

There are a variety of dimensions over which IDEs can be evaluated; some of them are discussed in this section. It is very important to realize that IDEs are combinations of two things – a path variable and a mechanism, each of which with important features that may be of interest for assessment and comparison. As defined in chapter 5, an IDE is related to one mechanism, but may have more that one path variable. This implies that any metric evaluating the mechanism can be used for the IDE directly, but also that, for metrics assessing path variables, some aggregate metric must be computed for them to be useful at the IDE-level. The following paragraphs define some relevant metrics for a single ility-driving element.

7.4.1.1 Cost

This is the monetary cost of the overall inclusion of the IDE in the design of the system. As discussed in chapter 2, there can be different types of cost, associated with either the path enabler or the mechanism:

- *Acquisition cost.* For a change option, this is the cost $C_{\varepsilon_i}^A$ of acquiring a certain path enabler ε_i for inclusion in the system. For a resistance property, it is the cost $C_{\iota_i}^A$ of acquiring a path inhibitor ι_i. The acquisition cost of a latent path variable is considered to be zero (sunk cost, using economics jargon).
- *Carrying cost.* For a change option, this is the cost $C_{\varepsilon_i}^C$ of maintenance of a certain path enabler ε_i. For a resistance property, it is the cost $C_{\iota_i}^C$ of maintaining a path inhibitor ι_i. Similarly to acquisition cost, carrying cost is considered to be zero for latent path variables.

- *Execution cost.* For a change option, this is the cost C_{δ}^{E} of executing a certain change mechanism δ. For a resistance property, it is the cost C_{ω}^{E} associated with the occurrence of a resistance mechanism ω.
- *Disposal cost.* For a change option, this is the cost $C_{\varepsilon_i}^{D}$ of disposing of a certain path enabler ε_i. For a resistance property, it is the cost $C_{\iota_i}^{D}$ of disposing of a path inhibitor ι_i.

Given these four cost types, it is now possible to define the cost of change options and resistance properties. For a change option CO, composed of $\widetilde{M}$ path enablers that imply a change mechanism δ, the total cost is:

$$C^{tot}(CO) = \sum_{i=1}^{\widetilde{M}} C_{\varepsilon_i}^{A} + \sum_{i=1}^{\widetilde{M}} C_{\varepsilon_i}^{C} + k \cdot C_{\delta}^{E} + \sum_{i=1}^{\widetilde{M}} C_{\varepsilon_i}^{D}$$

Where k represents the number of time the option is executed. Similarly, for a resistance property RP, composed of $\widetilde{N}$ path inhibitors that imply a resistance mechanism ω, the total cost is:

$$C^{tot}(RP) = \sum_{i=1}^{\widetilde{N}} C_{\iota_i}^{A} + \sum_{i=1}^{\widetilde{N}} C_{\iota_i}^{C} + k \cdot C_{\omega}^{E} + \sum_{i=1}^{\widetilde{N}} C_{\iota_i}^{D}$$

Where C_{ω}^{E} is often negligible, as discussed in chapter 5. The detailed assessment of such costs may be difficult, and its success largely depends on the nature of the data available. In the absence of reliable historical data, or if only a quick, first order analysis is required, it is possible to assess the cost of ility-driving properties (path variables and mechanisms) using an ordinal scale (e.g., [no irrelevant low medium high]), relative to the entire evaluated set.

7.4.1.2 Risk Attenuation

As discussed in previous chapters, the two main reasons for including ility-driving options in the design of a system are to either shield the system from risks or exploit potential opportunities. As mentioned earlier in this chapter, if filtered by consequence (negative or positive), the perturbation set can be thought of as a representation of the risk space or of the opportunity space, respectively. In the following paragraphs, three variants of proxy metrics for risk attenuation (the main benefit a decision maker is interested in) of a given ility-driving element are discussed. It is important to note that both preventing and reacting to the occurrence of perturbations reduces risk.

The crucial task performed here is mapping the ility-driving element to the perturbation set, in order to assess whether a perturbation is covered (i.e., addressed in some way) by the IDE. For example, the change option of swapping tires in a car (made possible by the existence of common interfaces in the car and a spare tire) addresses a possible perturbation of "change in terrain" from asphalt to dirt. It is important to note that what is evaluated is the mechanism, not the path variable. To capture the information generated in this task, a Perturbation Coverage (PC) matrix like the one shown in Figure 7-9 can be used. Usually, a binary entry – [0 1], depending on whether the perturbation is covered or not by the IDE – goes in the matrix. However, it is possible to further stratify the information into more than two levels by adopting a different ordinal scale that differentiates between "how well" a perturbation is covered (e.g., [0 1 2 3], or [0 1 3 9]). For example, a "multi-terrain tire"-enabled resistance property may address the "change of terrain" perturbation better than the "switching tires" change option, as the latter requires time, stopping operations, and physical work.

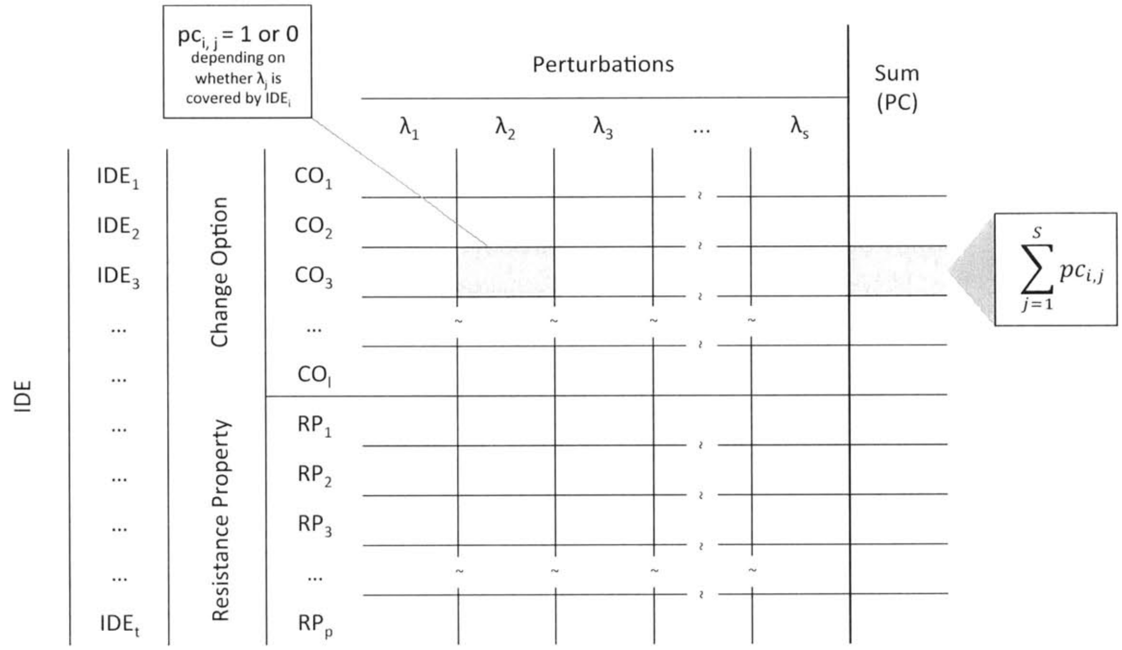


Figure 7-9: Perturbation Coverage matrix.[36]

It is possible now to introduce the first proxy metric for the risk attenuation provided by an IDE: Perturbation Coverage (PC). Given a Boolean entry $pc_{i,j}$ in the PC matrix in Figure 7-9 (assessing whether a perturbation λ_j is covered by an ility-driving element $IDE_i - CO$ or RP), it is possible to define the perturbation coverage (PC[37]) of an ility-driving element:

[36] These are essentially two matrices – one for change options and one for resistance properties – condensed in one. It is important to note that the row indicator i of $pc_{i,j}$ is matched to the ility-driving entity IDE_i, but one could choose to match it to change options or resistance properties.

[37] The scale for PC is a sum of ordinal numbers, and, as such, it does not mean much if taken independently. In fact, it should always be compared with other IDEs. A normalization factor is opportune if one wishes to use perturbation coverage as a final metric. Otherwise, PC is only used for the purposes of the following Risk Reduction and Normalized Risk Reduction metrics.

$$PC(IDE_i) = \sum_{j=1}^{S} pc_{i,j}$$

Where S is the total number of perturbations considered. Furthermore, it was noted earlier that $pc_{i,j}$ does not have to be a binary assessment. In the case it is not, the resulting PC scores would yield larger degrees of differentiation among ility-driving elements.

In chapter 5, the concept of risk and its relative nature has been discussed. The next two proxy metrics for risk attenuation are augmented variants of PC, which take into account assessed perturbation probability and impact. As discussed in chapter 4, this type of information is often elusive for perturbations considered in complex system design. If no historical data exist, one must rely on expert belief (and the multitude of biases that come with it) and use techniques like the Delphi method (also discussed in chapter 4) for assessment.[38] In some cases, and when time allows, the impact of a perturbation on the performance of a system can be estimated via modelling and simulation; impact on performance, then, directly informs the impact on value delivery. Similarly to the assessment of PC, different ordinal scales can be used here in order to assess probability and impact of perturbations. Chapter 5 also introduced how, in many different fields, the concept of risk is quantified via multiplying the probability of occurrence of an event by its impact.[39] Even in the field of systems engineering, McManus and Hastings (2006) discuss how, in general, "risk can be quantified by considering $(\text{probability of problem}) \times (\text{severity of problem})$." In the following proxy metrics, the coverage of a perturbation is weighted by the risk that the perturbation induces (as assessed). Given the assessed probability P_j and impact I_j (related to perturbation λ_j), it is possible to define the risk reduction (RR) of an ility-driving element:

$$RR(IDE_i) = \sum_{j=1}^{S} pc_{i,j}(P_j \cdot I_j)$$

The last proxy for risk attenuation introduced here takes into account the risk reduction of an IDE relative to the entirety of the risk space considered. The total risk induced by the entirety of the perturbation set considered is:

[38] The alternative, of course, as mentioned in the beginning of the chapter, would be to not do such an analysis at all. Doing such an analysis and being willing to accept the pros and cons that come with it is a policy decision.

[39] This operation is often not allowed, but nevertheless performed as a way of condensing two dimensions of information into one, risk. For example, an important problem (encountered also in the proxy metrics presented here) is that, from the rigorous standpoint of the theory of scales of measurement, it is not possible to multiply two ordinal assessments (Stevens, 1946).

$$R_{TOT} = \sum_{j=1}^{S} (P_j \cdot I_j)$$

Where S, as for above, is the total number of perturbations considered. Given R_{TOT}, it is now possible to define the normalized risk reduction (NRR[40]) of an ility-driving element:

$$NRR(IDE_i) = \frac{RR(IDE_i)}{R_{TOT}} = \frac{\sum_{j=1}^{S} pc_{i,j}(P_j \cdot I_j)}{\sum_{j=1}^{S}(P_j \cdot I_j)}$$

It is important to note here that the relative scores of such proxy metrics can vary as different scales of assessment are chosen for probability and impact (e.g., [1 2 3] vs. [1 3 9] vs. [1 2 3 4 5]). The same applies for the choice of different subject matter expert (and decision maker) groups that participate in the assessment – each individual carries a baggage of different experiential biases and ideologies. Furthermore, as mentioned earlier, the multiplication of two ordinal assessments is not mathematically rigorous (Stevens, 1946). In light of these problems and more, it is important to consider the above proxy metrics for risk not as rigorous and impeccable evaluations, but rather as "fallible indicators" (Hammond, 1996) – "fallible" because they represent a degree of belief (affine to the subjectivist view of Bayesian probability) (de Finetti, 1937) about states that are not directly observable. Frank R. Field, III eloquently summarizes this state of affairs and its problem: "We are permanently trapped in the duality between what one can measure and what one wants to know... 'I cannot predict the future; nevertheless, I will try to predict the future'." (Field, 2014)

The willingness to accept the potential error and pitfalls introduced by using such indicators is a policy decision for the stakeholders' discretion. As described by the NRC's *red book*, risk characterization is based on an analytic-deliberative process. Its aim is to describe a potential future situation in "as accurate, thorough, and *decision-relevant* [emphasis added] a manner as possible, addressing the significant concerns of the interested and affected parties." (National Research Council, 1996)

For the excerpted set of perturbations and change options shown in Table 7-2 and Table 7-3, respectively, an example application of the perturbation coverage mapping activity is presented in Table 7-4. For example, CO_8 (i.e., the possibility of adding an extra UAV given by the coexistence of workforce buffer, a spare UAV, and common communication interfaces) is a good option in terms of addressing an increase in boat arrival in the area of interest. Perturbation Coverage assessments here have been performed on a [0 1 3], where 0 indicates no coverage, 1 indicates coverage, and 3 indicates great coverage of perturbation. An example calculation of the three risk attenuation metrics proposed above is also shown in the table. In this case, R_{TOT} was

[40] If the ordinal scale used for pc is not Boolean, then the total risk considered for NRR can be multiplied by the maximum ordinal assessment of pc, as done for calculations for the example provided in Table 7-4.

taken to be the sum of all risks multiplied by the highest possible ordinal level of coverage (i.e., 3).

Table 7-4: Example of perturbation coverage matrix for an excerpt of the MarSec SoS case.

			Perturbation							
			λ_1	λ_2	λ_3	λ_4	λ_5	*PC*	*RR*	*NRR*
		P:	3	2	1	3	1			
		I:	2	3	5	2	3			
Change Option	CO_1	$\varepsilon_1 \wedge \varepsilon_5 \wedge \varepsilon_{12} \Rightarrow \delta_1$	1	0	1	0	0	2	11	0.14
	CO_2	$\varepsilon_1 \wedge \varepsilon_{12} \Rightarrow \delta_2$	0	0	1	1	0	2	11	0.14
	CO_3	$\varepsilon_2 \wedge \varepsilon_{10} \Rightarrow \delta_1$	1	0	3	0	0	4	21	0.27
	CO_4	$\varepsilon_2 \wedge \varepsilon_{10} \Rightarrow \delta_6$	1	0	0	1	1	3	15	0.19
	CO_5	$\varepsilon_3 \Rightarrow \delta_2$	1	0	1	0	0	2	11	0.14
	CO_6	$\varepsilon_3 \Rightarrow \delta_6$	1	0	0	1	0	2	12	0.15
	CO_7	$\varepsilon_4 \wedge \varepsilon_6 \Rightarrow \delta_3$	1	3	1	0	1	6	32	0.41
	CO_8	$\varepsilon_5 \wedge \varepsilon_7 \wedge \varepsilon_{12} \Rightarrow \delta_1$	3	0	3	1	0	7	39	0.50
	CO_9	$\varepsilon_5 \Rightarrow \delta_4$	3	1	0	1	1	6	33	0.42
	CO_{10}	$\varepsilon_7 \Rightarrow \delta_4$	1	1	1	0	0	3	17	0.22
	CO_{11}	$\varepsilon_5 \wedge \varepsilon_7 \Rightarrow \delta_4$	3	1	1	1	1	7	38	0.49
	CO_{12}	$\varepsilon_8 \Rightarrow \delta_2$	1	0	1	0	0	2	11	0.14
	CO_{13}	$\varepsilon_9 \Rightarrow \delta_5$	0	3	1	3	3	10	50	0.64
	CO_{14}	$\varepsilon_{11} \wedge \varepsilon_{12} \Rightarrow \delta_3$	1	3	0	3	1	8	45	0.58
	CO_{15}	$\varepsilon_{11} \Rightarrow \delta_5$	0	3	0	1	0	4	24	0.31
	CO_{16}	$\varepsilon_{13} \wedge \varepsilon_{14} \Rightarrow \delta_2$	1	1	1	0	0	3	17	0.22
	CO_{17}	$\varepsilon_{13} \wedge \varepsilon_{14} \Rightarrow \delta_6$	1	1	1	1	1	5	26	0.33

7.4.1.3 Opportunity Exploitation

The analysis of how well a certain ility-driving element enables the exploitation of opportunities that arise from the unfolding of uncertainty is symmetric to that for risk attenuation. The only difference is that the perturbation set is filtered by positive consequence; in this way, it becomes a representation of the opportunity space. Furthermore, the very concept of opportunity implies the possibility of acting to gain some extra value. This links it tightly to change options, and not so much to resistance properties. Hence, herein, it is assumed that opportunities can be seized by change options only, and not by resistance properties.[41] In the case the perturbation set is a representation of the opportunity space, the perturbation coverage metric PC becomes an indicator of opportunity exploitation. Similarly to risk, McManus and Hastings (2006) discuss how, in general, the ability to exploit opportunities "can be quantified by considering $(\text{probability of an event})\times(\text{value of the event})$." Consequently, if weighted by the likelihood of occurrence (P_j) and the value added (i.e., positive impact) by exploiting the opportunity (I_j), it is possible to define the opportunity exploitation (OE) of a change option as:

$$OE(CO_i) = \sum_{j=1}^{S} pc_{i,j}(P_j \cdot I_j)$$

Continuing the parallel, the total opportunity induced by the entirety of the perturbation set considered is:

$$O_{TOT} = \sum_{j=1}^{S} (P_j \cdot I_j)$$

And the normalized opportunity exploitation (NOE) of a change option is:

$$NOE(CO_i) = \frac{OE(CO_i)}{O_{TOT}} = \frac{\sum_{j=1}^{S} pc_{i,j}(P_j \cdot I_j)}{\sum_{j=1}^{S}(P_j \cdot I_j)}$$

7.4.1.4 Optionability

In chapter 2, the work of Mikaelian (2009) has been introduced. Of notable importance here is her definition of optionability as the "ability to enable types of real options." The latter, as pointed out in chapter 2, correspond to change mechanisms, as discussed in Ross (2006) and in this thesis. Mikaelian (2009) also introduces a metric for optionability, which can be summarized as the number of ways a change mechanism can be enabled.

[41] As mentioned in chapter 5, relaxing this assumption could be an interesting area for future research.

It is possible, then, to extend this idea of optionability to both change options and resistance properties[42] as a quality of their path enablers and path inhibitors, respectively. Given the set of M path enablers and the set of N mechanisms generated earlier (see section 7.3.5), the optionability of a path enabler ε_i is:

$$Opt(\varepsilon_i) = \sum_{j=1}^{N} o_{i,j}$$

Where $o_{i,j}$ is a Boolean assessing whether a path enabler ε_i is matched to a change mechanism δ_j. Similarly, given the set of Q path inhibitors and the set of R resistance mechanisms generated earlier (see section 6.3.4), the optionability of a path inhibitor is:

$$Opt(\iota_i) = \sum_{j=1}^{R} o_{i,j}$$

The Change Option Formation matrix and the Resistance Property Formation matrix in Figure 7-7 can be used to assist the assessment of the above optionability metrics. This is shown in Figure 7-10 for a change option (and is identical for a resistance property): the optionability of a path enabler (inhibitor) is the sum of the change (resistance) mechanisms it is linked to – i.e., the sum across the columns of a COF (RPF) matrix. For example, relative to the space of possible change options shown in Table 7-3, the optionability of the path enabler 'spare UAV' is 2.

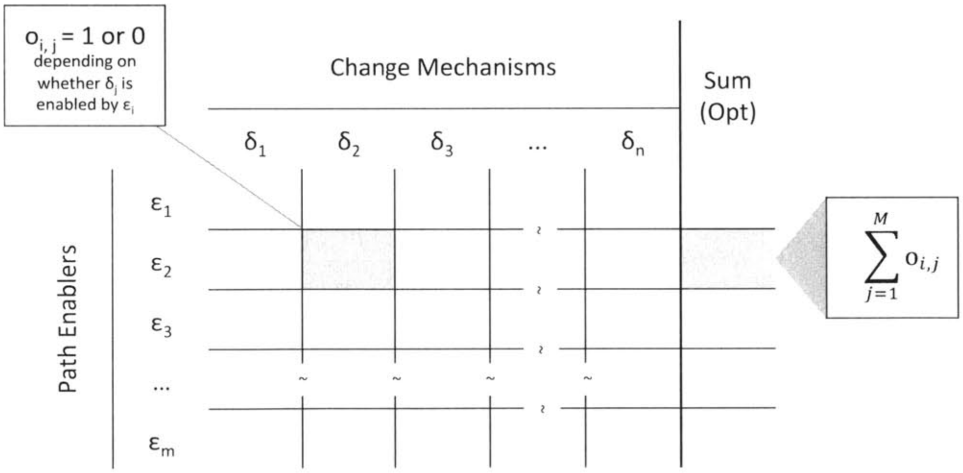


Figure 7-10: Assessment of path enabler optionability using the COF matrix.

Given these, it is possible to define a proxy metric for optionability at the IDE-level. The optionability of a change option CO, whose change mechanism is enabled by a set

[42] For which the term "optionability" is a bit of a misfit.

of path enablers $\widetilde{\boldsymbol{PE}}_{CO}$ (a subset of $\widetilde{M}$ path enablers from the entirety of the M path enablers generated earlier), is:

$$Opt(CO) = \sum_{k} Opt(\varepsilon_k) = \sum_{k}\sum_{j=1}^{N} o_{k,j} \ , \qquad k \mid \varepsilon_k \in \widetilde{\boldsymbol{PE}}_{CO}$$

Similarly, the optionability of a resistance property RP, whose resistance mechanism is enabled by a set of path inhibitors $\widetilde{\boldsymbol{PI}}_{RP}$ (a subset of $\tilde{Q}$ path inhibitors from the entirety of the Q path inhibitors generated earlier), is:

$$Opt(RP) = \sum_{k} Opt(\iota_k) = \sum_{k}\sum_{j=1}^{R} o_{k,j} \ , \qquad k \mid \varepsilon_k \in \widetilde{\boldsymbol{PI}}_{RP}$$

Figure 7-11 gives an illustrative example of the optionability of a change option. In this figure, three path enablers are linked to a change option (CO$_1$). The optionability of the change option is the sum of the optionability scores of its path enablers. For example, relative to the space of possible change options shown in Table 7-3, the optionability of CO$_{14}$ (i.e., 'satellite relay' and 'common network interfaces' imply executability of 'change [UAV] geographic segmentation') is 5. It is important to stress here that this metric, like the ones for risk attenuation, is the outcome of the assessment of particular groups of domain experts.

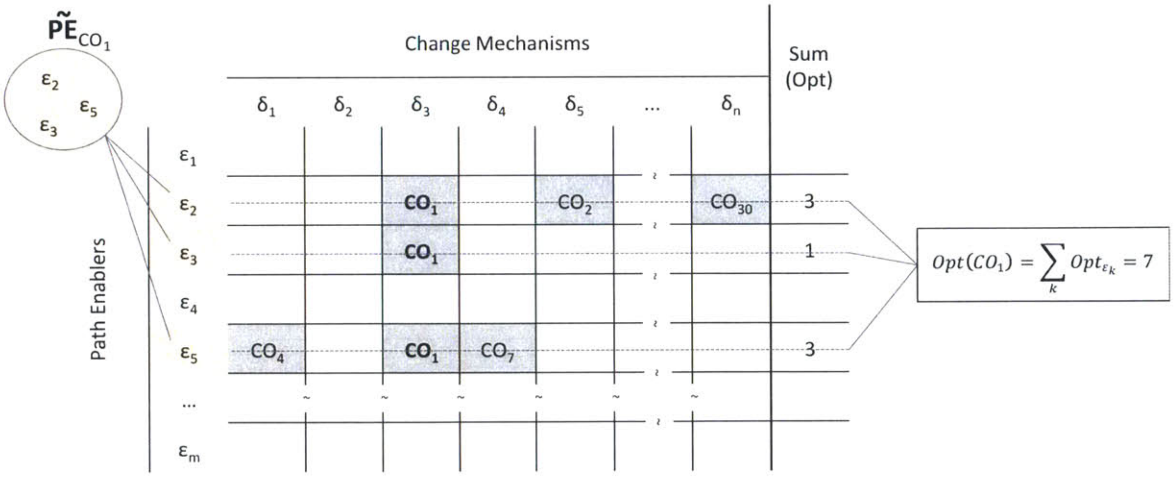


Figure 7-11: Computation of the optionability of a change option illustrated.

7.4.1.5 Realizability

In addition to the concept of optionability, Mikaelian (2009) also discusses that of realizability, which she defines as "the ability to implement a given type of real option." In the terminology of Ross (2006) and this thesis, this corresponds to the ability of a change mechanism to be enabled by more than one path enabler. As done for optionability, it is possible to expand the concept of realizability to both types of ility-driving elements. Hence, given the set of M path enablers and the set of N change mechanisms generated earlier (see section 6.3.4), the realizability of a change mechanism δ_j is:

$$Rz(\delta_j) = \sum_{i=1}^{M} o_{i,j}$$

Where $o_{i,j}$ is the same Boolean discussed above, assessing whether a path enabler ε_i matches a change mechanism δ_j. Similarly, given the set of Q path inhibitors and the set of R resistance mechanisms generated earlier (see section 6.3.4), the realizability of a resistance mechanism is:

$$Rz(\omega_j) = \sum_{i=1}^{Q} o_{i,j}$$

As for optionability, the COF and RPF matrices in Figure 7-7 can be used to assist the assessment of the above metrics. This is shown in Figure 7-12 for a change option (and is identical for a resistance property). For example, relative to the space of change options shown in Table 7-3, a highly realizable change mechanism is 'change task assignment', scoring 6. Lastly, given the fact that an ility-driving element is associated with one (and only one) mechanism, the IDE-level assessment of realizability is the same as that at the mechanism-level.

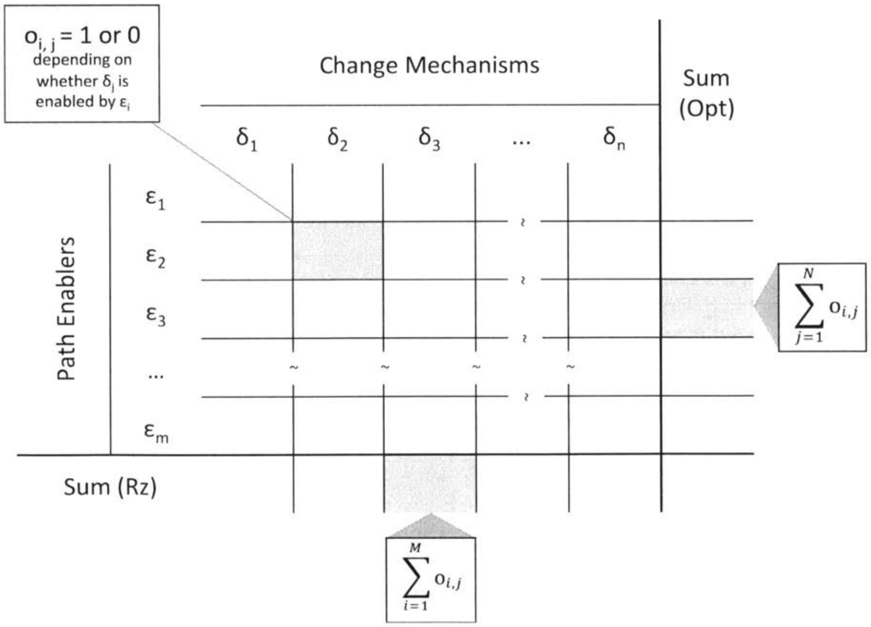


Figure 7-12: Assessment of change mechanism realizability using the COF matrix.

7.4.1.6 Dynamic Use

The ability to reuse an ility-driving element more than once across the lifecycle of a system may be of interest to decision makers. In light of this, two indicators of the dynamic use of the IDE are described here: reusability ($Reus$) and feasible epoch fraction (FEF). The former indicates the number of times an IDE can be used during the lifetime of the system. This is an ordinal assessment that can be performed using different scales (e.g., [once finite infinite]), and that is usually related to the path variable. For example, if one 'spare UAV' is available, the related change mechanism 'adding UAV to fleet' can be used only once. Similarly, having a 'workforce buffer' enables 'increasing the number of operators per UAV' only a finite number of times, while 'resisting a physical attack' with the use of 'armor' can be use for the entire existence of the system. If the IDE has a defined expiration time (that is likely going to be shorter than lifecycle), then it can't be reused an infinite amount of times. Feasible epoch fraction (FEF) relates to the epoch dependency of IDEs discussed in chapter 5. It is simply the ratio of the number of epochs in which the IDE can be used over the entirety of the epoch space considered.

The assessments of optionability, realizability and reusability for the options listed in Table 7-3 are illustrated in Table 7-5 below.

Table 7-5: Example assessment of optionability, realizability and reusability.

			Opt	*Rz*	*Reus*
Change Option	CO_1	$\varepsilon_1 \wedge \varepsilon_5 \wedge \varepsilon_{12} \Rightarrow \delta_1$	7	3	once
	CO_2	$\varepsilon_1 \wedge \varepsilon_{12} \Rightarrow \delta_2$	5	4	once
	CO_3	$\varepsilon_2 \wedge \varepsilon_{10} \Rightarrow \delta_1$	4	3	finite
	CO_4	$\varepsilon_2 \wedge \varepsilon_{10} \Rightarrow \delta_6$	4	3	finite
	CO_5	$\varepsilon_3 \Rightarrow \delta_2$	2	4	once
	CO_6	$\varepsilon_3 \Rightarrow \delta_6$	2	3	once
	CO_7	$\varepsilon_4 \wedge \varepsilon_6 \Rightarrow \delta_3$	2	2	infinite
	CO_8	$\varepsilon_5 \wedge \varepsilon_7 \wedge \varepsilon_{12} \Rightarrow \delta_1$	7	3	once
	CO_9	$\varepsilon_5 \Rightarrow \delta_4$	2	3	finite
	CO_{10}	$\varepsilon_7 \Rightarrow \delta_4$	2	3	once
	CO_{11}	$\varepsilon_5 \wedge \varepsilon_7 \Rightarrow \delta_4$	4	3	finite
	CO_{12}	$\varepsilon_8 \Rightarrow \delta_2$	1	4	infinite
	CO_{13}	$\varepsilon_9 \Rightarrow \delta_5$	1	2	infinite
	CO_{14}	$\varepsilon_{11} \wedge \varepsilon_{12} \Rightarrow \delta_3$	5	2	infinite
	CO_{15}	$\varepsilon_{11} \Rightarrow \delta_5$	2	2	infinite
	CO_{16}	$\varepsilon_{13} \wedge \varepsilon_{14} \Rightarrow \delta_2$	4	4	once
	CO_{17}	$\varepsilon_{13} \wedge \varepsilon_{14} \Rightarrow \delta_6$	4	3	once

7.4.2 Portfolio-level Meta Assessment

There are different ways one can go about choosing a set of ility-driving elements to include in the design of a system. One way is to consider the assessments of the different IDEs and pick the ones that are of interest, until a stopping criterion is reached. Another way is to compare different sets of IDEs – i.e., portfolio-level comparison. In order to perform the latter activity, there is a need for portfolio-level assessments.

Before introducing proxy metrics for portfolios of IDEs, it is important to note that a portfolio of IDEs always maps to a portfolio of mechanisms, which in turn maps to a portfolio of path variables. This is shown in Figure 7-13 for change options. The assessment of the portfolio of IDEs uses information from the latter two. Furthermore, the mapping of portfolios of IDEs to those of mechanisms is surjective – i.e., there is always at least one IDE per mechanism. The mapping of mechanisms to path variables can vary: usually the mapping of path variables to mechanisms is surjective, but it could also (rarely) be that a few number of path variables are linked to many mechanisms.

It is important to point out here that the portfolio-level assessment does not necessarily have to occur for portfolios of IDE. As mentioned in chapter 5, it can also be the case that a decision maker is interested in directly selecting a portfolio of path enablers and path inhibitors, or one of mechanisms. This may be because the analysis involves less information, or because one is interested in finding a good portfolio of mechanisms (that covers the risk space well), and build on that to find a portfolio of suitable path variables. In any case, what follows below is a discussion of portfolio-level assessment of portfolios of IDEs; the analysis for portfolios of path variables or mechanisms is very similar to (and at the basis of) that of IDEs.

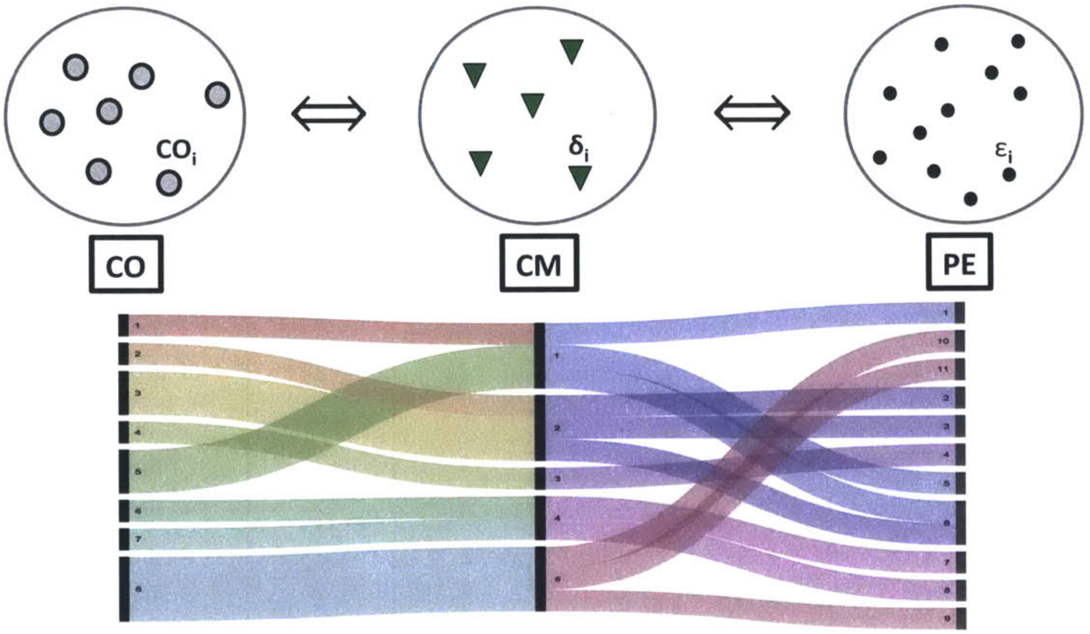


Figure 7-13: A portfolio of change options maps to one of change mechanisms, which in turn maps to one of path enablers (the same occurs for resistance properties).

The following portfolio-level assessments are described for a portfolio of ility-driving elements, $\widetilde{\boldsymbol{IDE}} = \widetilde{\boldsymbol{CO}} \cup \widetilde{\boldsymbol{RP}}$. Since $\widetilde{\boldsymbol{IDE}}$ is the union of a portfolio of change options and one of resistance properties, the assessment of portfolios of change options and resistance properties can sometimes precede that of portfolios of IDEs.

Aggregating information from the proxy metrics introduced above, the cost of a portfolio of $\tilde{L}$ change options is given by the sum of the costs associated with all the change options present in the portfolio. Mathematically, this can be summarized as:

$$C(\widetilde{\boldsymbol{CO}}) = \sum_{i=1}^{\tilde{L}} C^{tot}(CO_i), \qquad i \mid CO_i \in \widetilde{\boldsymbol{CO}}, |\widetilde{\boldsymbol{CO}}| = \tilde{L}$$

Similarly, the total cost of a portfolio of $\tilde{P}$ resistance properties is:

$$C(\widetilde{\boldsymbol{RP}}) = \sum_{i=1}^{\tilde{P}} C^{tot}(RP_i), \qquad i \mid RP_i \in \widetilde{\boldsymbol{RP}}, |\widetilde{\boldsymbol{RP}}| = \tilde{P}$$

The cost of the portfolio of IDEs, then, is given by the sum of the costs of the portfolios of change options and the portfolio of resistance properties:

$$C(\widetilde{\boldsymbol{IDE}}) = C(\widetilde{\boldsymbol{CO}}) + C(\widetilde{\boldsymbol{RP}})$$

As for risk attenuation, perturbation coverage at a portfolio-level can be approximated by the number of perturbations that are covered by the given portfolio. If that is the case, it is important to avoid counting "coverage" twice.[43] Not taking this into account may result in portfolios with good perturbation coverage, but of the same few perturbations. In order to avoid counting twice, a simple flag must be set up in the counting. Consider a portfolio $\widetilde{\boldsymbol{IDE}}$, composed of $\tilde{T}$ IDEs mapped to the set $\boldsymbol{\Lambda}$ of S perturbations using a PC matrix (e.g., Figure 7-14). Also consider a Boolean, $covered_j$, which indicates whether perturbation λ_j has already been covered (true, i.e.: 1) by one of the IDEs in the portfolio. Then, the perturbation coverage at the IDE-portfolio level is (e.g., Figure 7-14):

$$PC(\widetilde{\boldsymbol{IDE}}) = \sum_{i=1}^{\tilde{T}} \sum_{j=1}^{S} (\neg covered_j)(pc_{i,j})$$

As for the assessment at the IDE-level, perturbation coverage at the portfolio-level is a sum of ordinal assessments. Hence, it is best to normalize it relative to the maximum possible coverage, as it is done for Normalized Risk Reduction below.

[43] This is revisited in a few paragraphs.

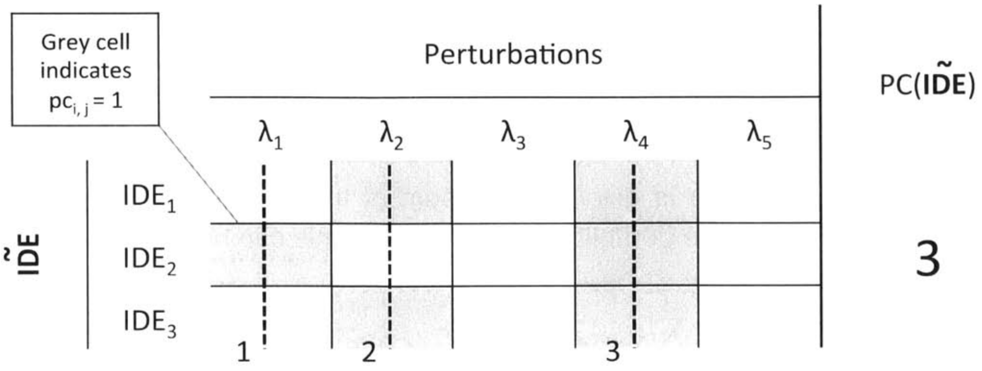


Figure 7-14: Illustration of PC computation at the IDE portfolio level using a PC matrix.

Similarly to the proxy metrics for risk attenuation at the IDE-level, using assessments of perturbation probability and impact, it is possible to weigh the perturbation coverage by the risk of a perturbation. Doing so yields a risk reduction indicator at the portfolio level:

$$RR(\widetilde{\boldsymbol{IDE}}) = \sum_{i=1}^{\tilde{T}} \sum_{j=1}^{S} (\neg covered_j)(pc_{i,j}) \left(P_j \cdot I_j\right)$$

The normalized risk reduction indicator is then given by:

$$NRR(\widetilde{\boldsymbol{IDE}}) = \frac{RR(\widetilde{\boldsymbol{IDE}})}{R_{TOT}} = \frac{\sum_{i=1}^{\tilde{T}} \sum_{j=1}^{S} (\neg covered_j)(pc_{i,j})}{\sum_{j=1}^{S} \left(P_j \cdot I_j\right)}$$

Where R_{TOT} is the total risk associated with the entirety of the risk space. For example, drawing from the perturbation coverage in Table 7-4, the NRR of the individual options CO_3, CO_4 and CO_7 is, respectively: 0.27, 0.19, and 0.41. However, when considering the portfolio of the three ($\widetilde{\boldsymbol{IDE}} = \{CO_3, CO_4, CO_7\}$), the portfolio-level NRR becomes: 0.62. Lastly, the assessment of opportunity exploitation at the portfolio level follows the same rationale discussed above and is not discussed herein.

Because of the assumption of counting coverage only once, the above risk attenuation proxies leave out an important piece of information, which may be of interest to decision makers: the redundancy of coverage for a perturbation associated with a portfolio of IDEs. In order to obviate this problem and take this information into account, it is possible to either devise new metrics or introduce a new weighting factor in the calculation of perturbation coverage. Using information from the assessments so far, a possible indicator of the degree to which a portfolio of IDEs covers perturbations repeatedly can be realizability. In fact, if IDEs' mechanisms are very "realizable," it means that they are achievable in a variety of different ways. Hence, in these cases, it is more likely that the given portfolio of IDEs covers the same perturbation in different ways. Proxies for the optionability and realizability at the portfolio level can be simple sums of their assessments at the IDE-level. Considering the same portfolio $\widetilde{\boldsymbol{IDE}}$ – which is the union of a portfolio of $\tilde{L}$ change options and $\tilde{P}$ resistance properties – optionability and realizability at the portfolio level are:

$$Opt(\widetilde{\boldsymbol{IDE}}) = \sum_{i=1}^{\tilde{L}} Opt(CO_i) + \sum_{i=1}^{\tilde{P}} Opt(RP_i)$$

$$Rz(\widetilde{\boldsymbol{IDE}}) = \sum_{i=1}^{\tilde{L}} Rz(CO_i) + \sum_{i=1}^{\tilde{P}} Rz(RP_i)$$

Aggregate metrics for the dynamic use of the portfolio of IDEs can also be derived using assessments at the IDE-level. For example, the reusability indicator ($Reus$) at the portfolio level can be a three-dimensional vector containing the percentage of IDEs in the portfolio that can be reused (1) once, (2) a finite number or (3) as many times as desired [once finite infinite] – e.g., $Reus(\widetilde{\boldsymbol{IDE}}) = [15\% \; 50\% \; 35\%]$.

7.5 Selection of IDEs

Once an extended list of ility-driving elements has been generated and evaluated, the next logical step is to select a portfolio of IDEs among them.[44] This selection process may occur for two different reasons: (1) directly including the IDEs in the design of the system, or (2) carrying the selected IDEs forward for further analysis with more detailed modeling and simulation. In any case, in order to perform a selection activity, a comparative analysis must be performed.

The following two subsections describe possible analytical techniques and visualization tools that may be used for facilitating selection. The size of the space of possibilities (i.e., possible portfolios to choose among) grows in a combinatorial fashion with the number of IDEs considered. Hence, analysis and visualization can be used here for two different ends: (1) reducing the size of the space to one that can be explored by a decision maker, and (2) exploring such space of alternatives.

7.5.1 Analytic and Exploratory Techniques

For a set of ility-driving elements, the space of possible portfolios to choose among grows large very fast. Given the limitations with regard to the amount of information that can be processed by a computer, exploring the entirety of the space becomes an intractable problem relatively quickly. For a set $\boldsymbol{IDE}$ of N IDEs among which to choose a

[44] It is important to note that, depending on time constraints, the selection may already happen as IDEs are generated. I.e., all IDEs that are considered are automatically included in the system.

portfolio, its power set – the number of all subsets in $\boldsymbol{IDE}$ (i.e., all possible different portfolios) – grows combinatorially with N:

$$|\mathcal{P}(\boldsymbol{IDE})| = \sum_{k=0}^{N} \binom{n}{k} = 2^n$$

Hence, for a space of 50 IDEs, 2^{50} evaluations must be made. Assuming ad absurdum that each portfolio evaluation takes a microsecond, evaluating all possible portfolios would take over 35 years. On the other hand, under the same circumstances, a space of 30 IDEs would result in slightly over 15 minutes of computation. It follows that a complete exploration may be possible for problems that only consider a few IDEs. However, as it can be deduced from the applications shown in this chapter, the number of IDEs to be considered can grow very rapidly for a complex system (like the MarSec SoS). In fact, in this case, a small excerpt of all the IDEs considered in the larger application resulted in 17 of them.

When faced with such a large space, the usual approach is that of using an optimization algorithm to search for the most attractive solution(s). However, in this case, even this is difficult, as the optimization of some objective function (e.g., risk reduction) for such a space of alternatives is a set covering problem (SCP). Depending on the formulation of it (e.g., whether there is a constraint on budget), this problem can be NP-complete or NP-hard (Krause and Guestrin, 2007); meaning that, as N grows large, there are no guarantees on the minimum time a solution might take to find. Krause and Guestrin (2007) discuss a similar optimization problem (optimally placing underwater sensors so to monitor environmental phenomena and water distribution networks), and leverage its "submodularity property" to "efficiently achieve near-optimal selections."[45] Although not discussed here, an application of this algorithm to the problem of selecting a portfolio of IDEs would be an interesting avenue for future work.

Hence, if the space of considered IDEs is too large (i.e., > ~30), the human brain must be interrogated in order to constrain the problem, and make it suitable for further analysis. One approach for doing this could be constraining the problem within a subsection of the space, and then move forward from there. For example, in a space with 80 IDEs to be considered, $\sim 1.27\times 10^{24}$ portfolios are possible. However, if one decides to make a first down-selection only among combinations of 4-element portfolios, the space is reduced significantly: $\binom{80}{4} = \frac{80!}{4!(80-4)!} \cong 1.5\times 10^{6}$. From that point on, then, one can decide to add other relevant IDEs to the optimal 4-element portfolio found. Furthermore, another avenue for decreasing the dimensionality of the problem can be considering only mechanisms – as opposed to IDEs – in the evaluation process. In fact, as described earlier (and shown in Figure 7-14), for a given portfolio the number of mechanisms is always smaller than that of IDEs. However, such an approach would limit

[45] In a submodular cover problem, a greedy approach that produces approximations to the solution can be implemented (Bar-Ilan et al., 1996).

the amount of info that can be used in the analysis – e.g., mechanisms don't have info related to IDEs' full cost or optionability. Lastly, another approach for constraining the problem – discussed more in depth in the in next subsection – is using visualizations. Through visualizations that allow the user to compare different IDEs, it is possible to trim the space by either selecting (before the analysis) IDEs that are considered indispensable or eliminating some.

When the problem of portfolio selection is reduced to a workable one in size, it is possible to use analytical techniques to explore it. Although one may intend to use optimization techniques, this is not advisable due to the nature of the data.[46] Going farther away from the idea of "optimal," it is possible to use iso-performance techniques (de Weck and Jones, 2006) to identify and evaluate performance-invariant sets of solutions. This way, given a threshold on attributes of interest (e.g., cost, risk reduction, realizability), it is possible to further reduce the space of possible portfolios to one that can be explored using techniques like MATE (discussed in 4.1.1). In MATE, it is possible to trade benefits versus cost on a tradespace, where the former is typically approximated with the use of a multi-attribute utility function (Keeney and Raiffa, 1976).

Lastly, it is important to reiterate that the results of any analysis performed here must be "handled with care" because of the nature of the proxy metrics – which, as discussed earlier, must be considered as "fallible indicators" (Hammond, 1993) – and that of the assessment process. The analytical techniques discussed above must be used as an augmentation for simplifying the (otherwise overwhelming) problem of selection. The space of possibilities is large, and these techniques are a way of structuring the problem or placing constraints on what a decision maker wants to explore and compare.

7.5.2 Visualization Tools

As mentioned above, an effective way of choosing ility-driving elements may be through the aid of visualization tools. Tradespaces in MATE are one example of condensing multiple dimensions into a visual that is intelligible for the human brain. There, analytical transformations like the computation of a multi-attribute utility score (the most commonly used) condense multiple dimensions of benefit into one, which is then traded against cost. In this section, other ideas for comparing IDEs (and portfolios thereof) are discussed.

[46] As discussed in chapter 2, the idea of identifying optimal portfolios of (financial) assets was first introduced by Markowitz (1952), who devised a multi-objective optimization solution that yields an efficient frontier (in the mean return vs. variance space) of portfolios of investments. This idea may seem attractive for the case of portfolios of IDEs, where one may want to find efficient portfolios in the cost vs. risk reduction space. However, the optimization problems are very different in these two cases. Among other things, unlike finance, there is often no historical data to rely on for the evaluations, and obtaining one optimal solution is somewhat meaningless. Even more robust optimization techniques – see discussion of Tütüncü and Koenig (2004) in chapter 2 – would suffer from the same problems.

One approach to identify IDEs that are of relevance to stakeholders could be to actuate a "greedy" algorithm, whereby one starts by picking the best performing IDE, then the next best, and so on.[47] One way of visualizing all the IDEs generated across all dimensions of interest to stakeholders can be parallel coordinates plots. These are common for visualizations of high-dimensional geometry and analysis of multivariate data. A parallel coordinate plot is composed of as many parallel lines as the dimensions of interest. To show a set of points in this multi-dimensional space, these parallel lines are drawn next to each other, typically vertically and equally spaced. Then, a point in this multi-dimensional space (e.g., an IDE) is represented as a polyline with vertices on the parallel lines; the position of the vertex on a given line corresponds to the score of a given IDE in that dimension. Using such visualization, one may be able to discern IDEs that perform outstandingly as compared to the rest of the set (according to his or her preferences). Starting from those, it is possible to build a portfolio "greedily" in a step-by-step fashion. An example of such plot is given for the set of change options considered so far in Figure 7-15. This figure considers 5 different dimensions: *Cost* (randomly generated), *Opt*, *NRR, Rz*, and *U* (from Table 7-4 and Table 7-5). In this plot, no single change option stands out, but, for example, CO_{11} and CO_{14} both score quite well in terms of both *Opt* and *NRR*, which are arguably among the most important attributes a decision maker may be interested in when choosing an IDE over another. It is interesting to see how CO_{13}, the best in terms of *NRR*, does poorly in most of the other dimensions, and might not be the best one to consider for selection.

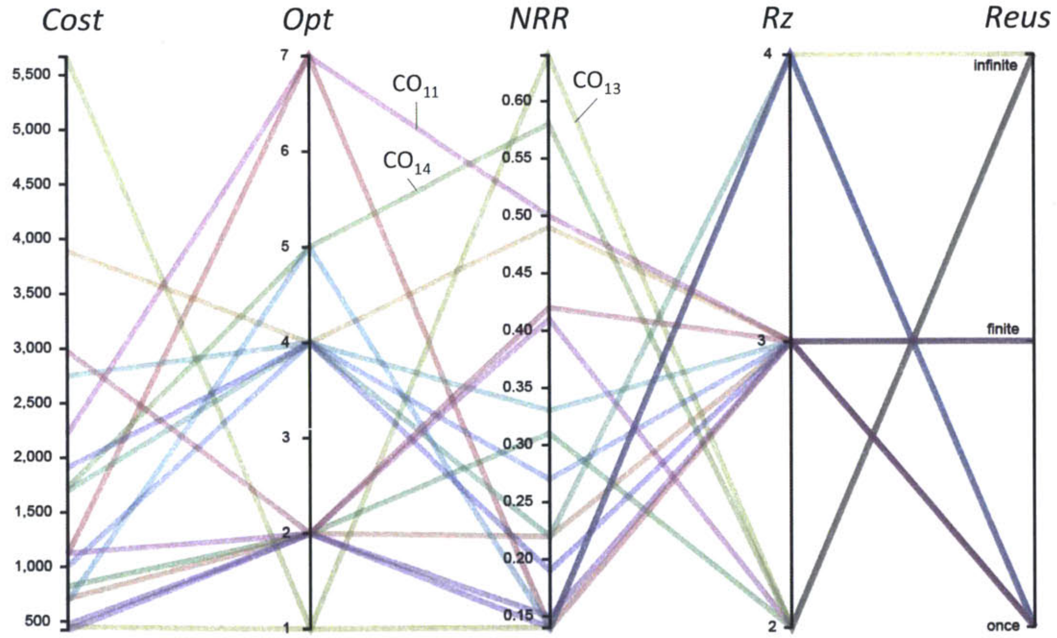


Figure 7-15: Parallel coordinates plot for the options considered thus far. Each option is colored differently and the best three in terms of NRR (CO_{11}, CO_{13} and CO_{14}) are labeled.

[47] Interestingly, such an approach is also at the basis of the submodular algorithm for analytically searching a set covering problem.

Another useful visualization technique is an alluvial diagram.[48] Alluvial diagrams are used to represent flows and to see correlations between categorical or ordinal dimensions, visually linking the number of elements sharing the same categories. It is particularly useful for exploring the nature of the space. For example, Figure 7-16 shows an alluvial diagram for the change options considered in the application to the MarSec SoS so far. The diagram shows that, for the space of options generated, few have good optionability, while about half of the set has the highest realizability.

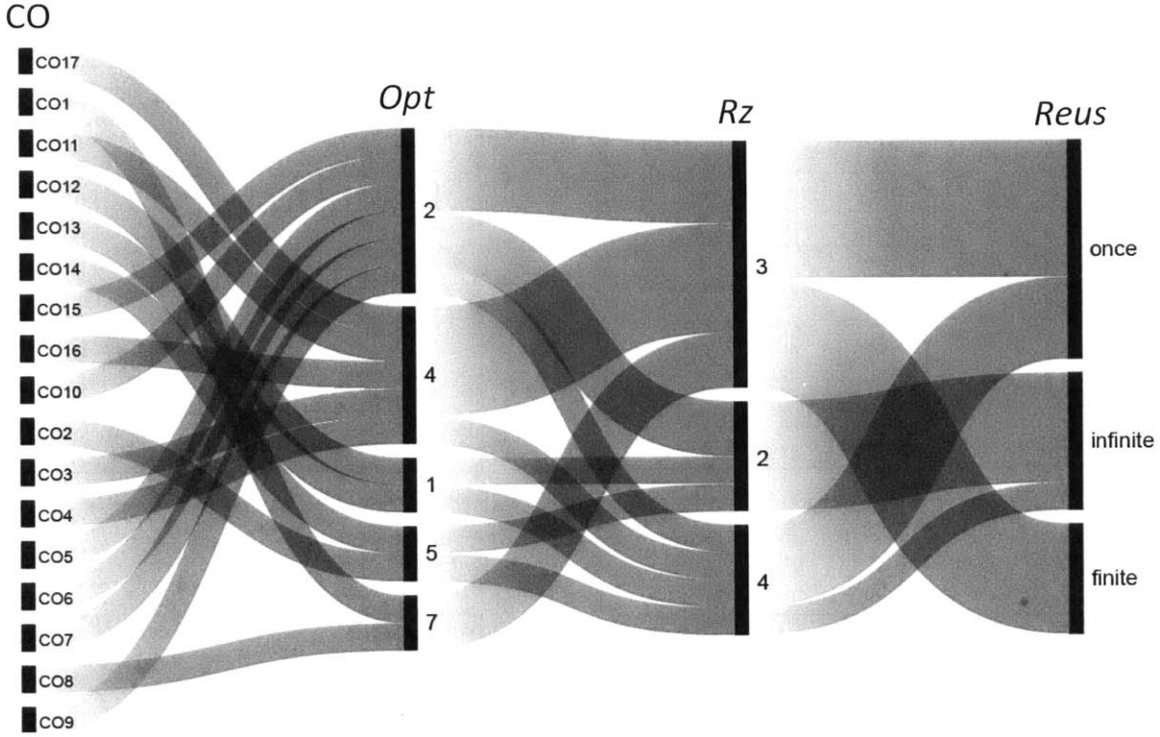


Figure 7-16: An example of alluvial diagram for the space of change options generated for the running application.

When it comes to portfolios of IDEs, as pointed out earlier, the dimensionality of the space rapidly explodes. When comparing many portfolios (to the limit that it is feasible for a computer to process), a parallel coordinates plot may be useful insofar as focusing on the portion of the possibilities that dominate the rest. When the comparison is reduced to a few portfolio alternatives and the focus is on risk, risk matrices (discussed in chapter 5) can be a useful visualization tool. In fact, these can be used to visualize and compare the risk reduction profile of a given portfolio of IDEs. For example, using the risk matrix in NASA (2012), a comparison of four small, randomly selected portfolios of change options (among the ones generated in Table 7-3) is shown in Figure 7-17. From this figure, it evinces, for example, that CO_8 and CO_{13} (related to the mechanisms of 'add asset' and 'change authority distribution') would work very well together in terms

[48] A great resource for scripts that enable the creation of alluvial diagrams, parallel coordinates plots, as well as many other effective visualizations is: http://d3js.org/.

of mitigating the risk induced by the given uncertainty space considered. Other combinations (e.g., CO_1, CO_5, CO_{10} and CO_{15}) are not as efficient and they leave some perturbations unaddressed. It is important to note that some key dimensions are missing from the risk matrices in Figure 7-17: e.g., cost and reusability. These can be incorporated through color, shape, or other visual indicators. The dimension of reusability is particularly important in this case because, if a perturbation is likely to occur many times, one may be interested in highly reusable IDEs to address it. For example, CO_8 can be used only once to address an increase in boat arrival rate (λ_1).

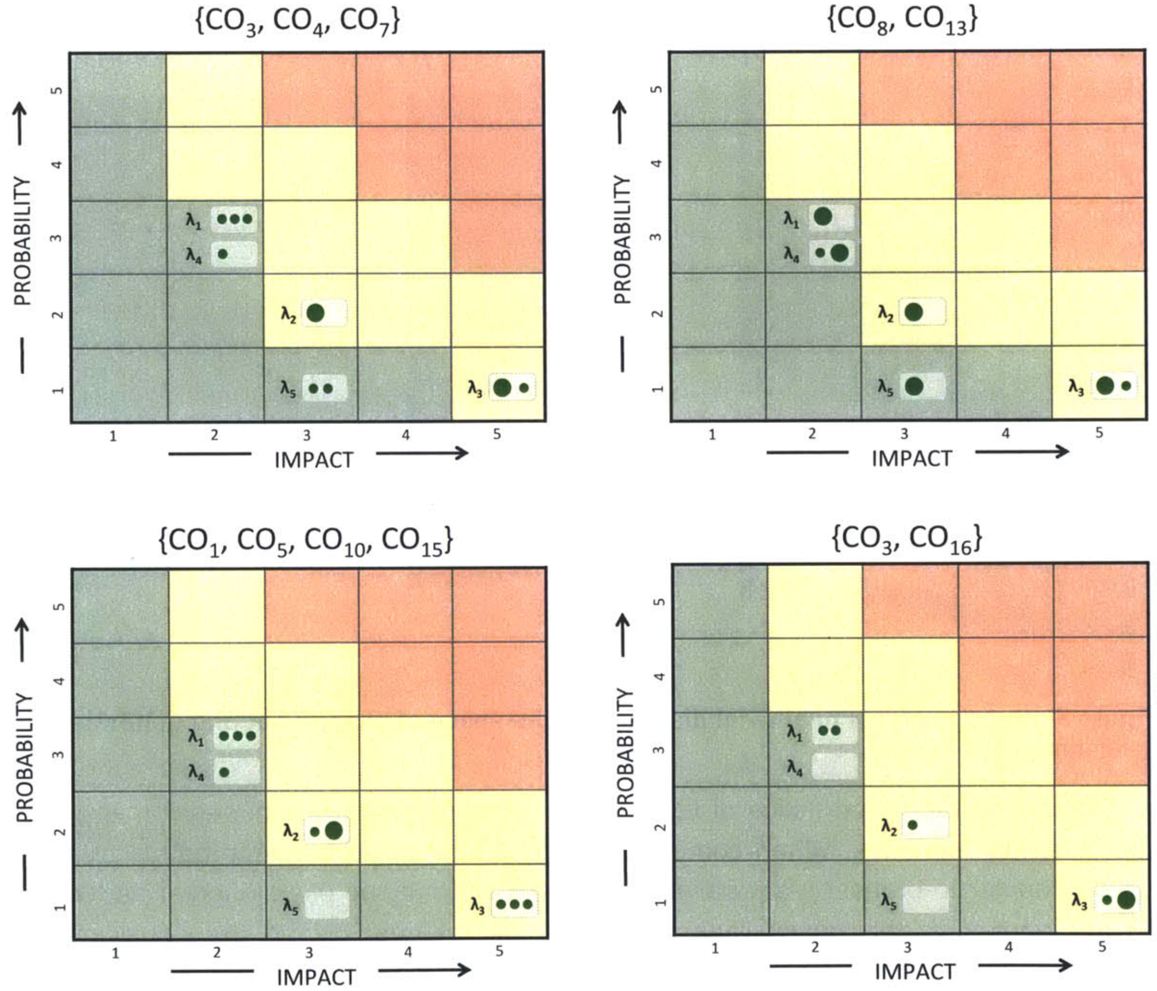


Figure 7-17: Comparison of risk reduction profiles of four portfolios of change options selected at random. Small and big circles indicate *PC=1* and *PC=3* on a [0 1 3] scale.

7.6 Summary

This chapter has introduced the IDE Analysis, a structured approach for the generation, evaluation and selection of IDEs. IDE Analysis is performed with a baseline system design in mind, and with the goal of selecting a set of relevant IDEs for either (1) further, more detailed analysis or (2) direct inclusion in the system.

The approach starts with the generation of an extensive list of ility-driving elements. In order to perform this task, different techniques were discussed (design principle to perturbation mapping, cause-effect mapping, etc.), among which the user can choose the most appropriate/affine (or perform all). Then, a number of possible proxy metrics (summarized in Table 7-6) were introduced in order to evaluate the different IDEs. Lastly, the problem of selection was discussed, for which analytical techniques are seldom effective. As an alternative, several visualization techniques (as well as some alternative usages for analytical techniques) that can facilitate the process of comparison and selection (both at the IDE- and portfolio-level) were presented.

Table 7-6: Summary of metrics introduced in chapter 7.

Metric	Acronym	Good for (at IDE- and portfolio-level):
Cost	*Cost*	Assessment of cost, including: acquisition, carrying, execution and disposal.
Perturbation Coverage	*PC*	Assessment of the extent to which perturbations are covered
Risk Reduction	*RR*	Assessment of the extent to which risk is reduced
Normalized Risk Reduction	*NRR*	Assessment of the extent to which risk is reduced, as relative to the entirety of the risk space considered
Opportunity Exploitation	*OE*	Assessment of the extent to which opportunity is seized
Normalized Opportunity Exploitation	*NOE*	Assessment of the extent to which opportunity is seized, as relative to the entirety of the opportunity space considered
Optionability	*Opt*	Assessment of the degree to which more than one mechanism is associated with the IDE (or portfolio)
Realizability	*Rz*	Assessment of the degree to which more than one path variable is associated with the IDE (or portfolio)
Feasible Epoch Fraction	*FEF*	Assessment of the epoch-dependent unavailability
Reusability	*Reus*	Assessment of the extent to which an IDE (or a portfolio) can be reused over time

Alongside the introduction of the theoretical material, a case application to the MarSec SoS was run. Although only an excerpt of the larger application to the case, the small demonstrations of the techniques (and metrics) introduced are intended to help the reader ground the abstract concepts in more concrete examples.

8 Discussion

"Que sçay-je?"

– Michel de Montaigne (1595)

This chapter presents general discussions around three main topics: applicability of research; general considerations with regard to the main research contributions; and possible areas for future research.

8.1 On the Applicability of Research

The domain applicability of this research is complex system design. As discussed in chapter 2, complex systems are those featuring high degrees of structural, social and dynamic complexity. Hence, anything from the design of an aircraft to that of the next regulation on CO_2 emissions is a valid target for the application of ideas and frameworks discussed in this thesis.

In particular, the research takes a *dynamic system perspective* for design. It first describes the inherent exploratory nature of such design efforts in the realm of complex systems. Then, it discusses ways to characterize uncertainty around such systems, as well as ways to characterize and enable responses to this uncertainty. These responses (i.e., the *ility-driving elements*) can be used to purposefully drive the emergence of ilities over time. Therefore, the content discussed in this thesis is very applicable to those domains that are specifically interested in designing systems that display ilities across their lifecycle. For example, this research may be particularly relevant for the design of complex DoD systems, which "tend to be designed to deliver optimal performance within a narrow set of initial requirements and operating conditions at the time of design." (Dashmukh et al., 2012) Other examples may be the design of large commercial enterprises (e.g., a constellation of satellites providing a service like Iridium) or of public infrastructure systems (e.g., the design of a large mass transportation system). In general, this work is relevant to those systems that require much effort and resources to be put into use, and which are expected to endure and continue to deliver value for a long time.

Furthermore, it has been discussed how the concepts put forth in this thesis can be applicable to the field of policymaking for planned adaptation efforts. Designing a regulatory system that is able to adjust to meet emerging requirements and knowledge is an attractive idea (McCray et al., 2012), and this research could perhaps help in structuring the actual process underlying the delineation of such adaptive systems. Similarly, with the right path inhibitors, it could help build regulatory systems that would resist the influence of undesired behaviors (e.g., big corporations buying off a politician).

Lastly, as mentioned in the very first lines of this thesis, design pervades every aspect of human life. With enough degrees of abstraction, the frameworks and approaches produced in this thesis are applicable to the decision-making processes of all human beings that like to think strategically about the unfolding of their future. For example, the time and effort dedicated to writing this thesis are opportunity costs associated with the inclusion of some path enablers in the life of the author (e.g., knowledge accumulated, two Masters degrees, meeting of smart and driven individuals, etc.). Hopefully, at some near or remote point in the future, these path enablers will allow for a transition from the then current to a preferred state of affairs (e.g., the obtainment of a PhD degree, a better job, the same job in a preferred city, no job and continuing to think and build on knowledge acquired while sipping on a cup of coffee and absently staring at the gorgeous landscape surrounding his hometown of Vallata, etc.).

8.2 On Research Contributions

In this section, the most important contributions of the thesis are discussed. First, a recapitulation and discussion of the characterization of uncertainty in complex sociotechnical systems is presented; then, the modeling of ility-driving elements is discussed, as well as their analysis for inclusion in systems.

8.2.1 On Uncertainty and Perturbations

In chapter 2, a literature overview of possible types of uncertainty – as well as ways to model and identify it – was presented for both the field of policymaking and that of complex system design. Starting from a discussion on the complexities that permeate such systems, it was concluded that designing in these circumstances is a very difficult task. In fact, apart from structural complexity, social and dynamic complexities make it impossible to consider such systems as closed. Hence, modeling uncertainty in the way that is done in other fields (e.g., finance or business) becomes a large oversimplification. Among other reasons, in the case of complex systems, the objective functions aren't always clear and can change over time (Rittel, 1969). Furthermore, the pronounced social component makes it impossible to draw generalizations across different cases, and "situational analysis" (Popper, 1967) becomes the only rational approach. Hence, in the remainder of the thesis, the traditional systems engineering perspective that freezes requirements early in the conceptual phase was abandoned, and a dynamic system perspective for design was embraced.

Chapter 3 has stressed the fact that modeling uncertainty in this context is an exploratory endeavor (Bankes, 1993): i.e., a way of testing possible hypotheses about the unfolding of uncertain events, thereby enabling the design of systems that are able to display ilities in the modelled uncertainty space. The modeling effort discussed starts with a formal characterization of three important spaces system designers deal with: design, context, and needs. These are mapped onto the performance space and value space through performance and value models, respectively. In this modeling approach, *design space, context space and needs space become the backbone for structuring and thinking strategically in the design effort.* The variables from which these spaces are derived (design variables, context variables, and needs variables) are the essential elements that constrain the design effort within a subspace of possibilities (i.e., of hypotheses) that is deemed relevant. Where to draw the line of such a subspace is a policy decision for the designers and the decision makers.

The space of uncertainty in this setting, then, is modelled as the possible unintentional[49] (from the perspectives of designers and stakeholders) transitions in design, context, or needs. Any trace in the diagram in Figure 8-1 is a possible state of affairs (i.e., hypothetical scenario) a system might find itself in, and corresponds to a level of value delivery (since value is a function of the design instance, context instance, and needs instance). Perturbations with negative consequence push the system into a trace that corresponds to less value delivered; perturbations with positive consequence push the system (or give designers the opportunity to push the system) to a trace that delivers more value.

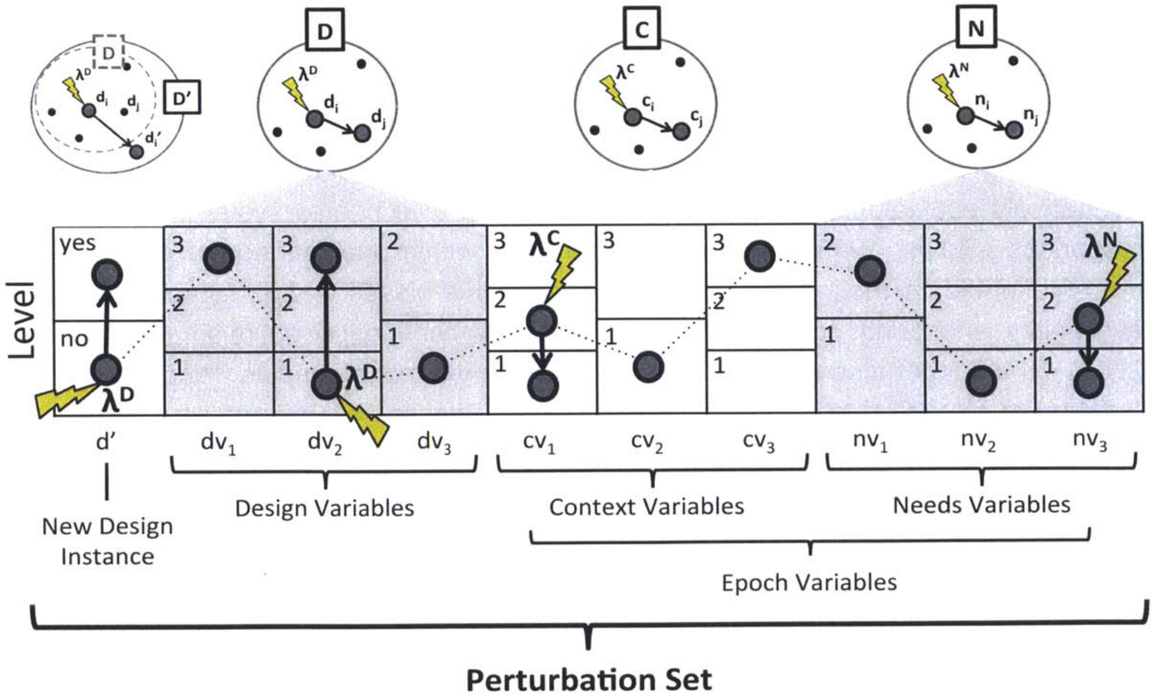


Figure 8-1: Variable-centric representation of perturbation set.

[49] The use of this word for the case of variations in needs is a bit off. However, it must be considered unintentional from the present perspective on the design effort.

The perturbation set in Figure 8-1 represents the uncertainty space a designer needs to consider, and is given by the set of variables that can be affected (by perturbations) as well as allowed transitions. It is important to note that, in the modeling of the uncertainty space proposed here, it is possible for a design instance to be pushed outside of the instantiated design space, but not for a context instance or a needs instance. There are two main modeling assumptions behind this. First, the design space is considered to be all the potential alternatives a designer may *want* to consider. Hence, a design instance with a UAV with a broken wing is conceivable, but outside the enumerated design space. Second, the context space and needs space are those conceivable at the time the conceptual design is taking place. Anything beyond these is either deemed irrelevant (or impractical) for the purposes of the design process, or is an *unknown unknown* (McManus and Hastings, 2006). If the system at hand undergoes systematic redesign efforts (e.g., large military SoS like the Ballistic Missile Defense System discussed in chapter 6), then these spaces may change over time, as new information is revealed. For example, for the Iridium satellites system briefly discussed in chapter 1, the advent of ground-based wireless telephony might have been an unknown unknown at the time of conceptual design.

An important source of uncertainty that has not been discussed or modeled in the current research is related to the modeling of performance and value delivery of a system. In fact, uncertainty can lie also in the assumptions behind the performance model and the value model. While assumptions behind the performance model can be captured to a certain extent by context variables (e.g., modeling varying technological levels), it is very difficult to conceive and represent uncertainty associated with assumptions underlying value models. As a matter of fact, the process of eliciting and constructing preferences over future prospects is plagued by unavoidable variance: normatively equivalent methods of elicitation often give rise to systematically different responses (Slovic, 1995). Even when confined within one manner of eliciting values from stakeholders (i.e., one value model), cognitive biases – e.g., representativeness, availability, anchoring, contamination, compatibility, confidence, optimism (Gilovich et al., 2002) – may undermine the trust in and veracity of the model outputs (Ricci et al., 2014).

In short, uncertainty is modeled as unintentional and exogenous operators on a set of possibilities (i.e. scenarios), given by the product of design, context and needs spaces that are deemed relevant by the designer (as mapped by performance and value models). It is important to point out, then, that perturbations enable the formation of networks (as opposed to static sets) of possibilities. Furthermore, the modeling of perturbations in this way is preparatory for the modeling and conceptualization of the elements that enable to respond to uncertainty: ility-driving elements. While perturbations are unintentional operators on the system state, IDEs are intentional operators that enable responses to the uncertainty modeled. When thinking about perturbations and IDEs from this perspective, a dynamic network-centric view on design may be taken. This is discussed in further detail in section 8.3.3.

8.2.2 On Responding to Uncertainty and IDEs

Chapters 2 and 4 also discuss how, in the context of complex system design, the presence of such irreducible uncertainty makes it unwise to design for a very specific set of objective functions and disregard the dynamic outer environment or the fact that such objective functions can change. In these cases, robust or changeable (Ross, 2006) systems are better suited than those designed using optimization strategies that work well only when finely tuned to precisely known environments (Simon, 1996). In this scenario, systems that are able to exhibit ilities across their lifecycle are desirable, which is why chapter 5 attempts to define and model ility-driving elements in system design.

As conceptualized in chapter 5, ility-driving elements can be of two types: change options and resistance properties. The former imply an ability to make a change in the system that is executed by an agent (internal or external). The latter do not imply an executable change, but rather the ability to resist undesired changes in the system. With regard to the occurrence of a perturbation, both types of IDE can be preventive (avoiding the occurrence of the perturbation) or reactive (mitigating or recovering from the impacts of the perturbation). An important subtlety with regard to this way of modeling ility-driving elements arises: resistance properties, unlike change options, are considered to be passive, as they require no execution decision by a human agent (i.e., no decision rule implemented by a human-created logic, internal or external to the system). As discussed in chapter 2, the word "option" implies the "power of deciding" (i.e., of executing, in this case), which is not present in the case of resistance properties. Hence, in this view, any type of active adjustment, however small, is considered a change event. This is very much a philosophical standpoint that can be revisited in the future. In fact, the idea of a resistance property has come to life when thinking about impulse perturbations, as opposed to graceful degradation. An alternate view might be that, in the case of graceful degradation, small adjustments that enable the system to maintain the same state are a resistance property. However, this would imply a definition of resistance property that rests on small changes by an executive agent. As briefly discussed in chapter 5, this issue relates closely to a long philosophical debate: "Theseus' paradox." This is a thought experiment that raises the question of whether an object that has had all its components replaced over a long period of time (the ship of Theseus in the original puzzle) remains fundamentally the same object. The standpoint taken in the formalization of resistance properties and change options presented in this thesis is that it doesn't remain the same object: every time an adjustment is made, an intentional change by an agent has occurred (most likely to sustain value delivery).

8.2.2.1 Dynamic System Perspective for Design

The dynamic system perspective discussed first in chapter 4 implies the awareness of the fact that, for large and long-lasting complex systems, operational contexts and expectations will inevitably change over time. Furthermore, the system itself may undergo forced transitions to new instances or spontaneous ones, due to, for example, the malfunctioning of the inner workings of the system. Hence, the need for modeling the epoch space in addition to the design space (Ross and Rhodes, 2008). From such modeling, two layers of exploration evince: (1) an exploration of the design space and the way it maps to the performance and value spaces, and (2) an exploration of the outcomes of (1) under different assumptions about the outer environment and the objectives pursued (i.e., the epoch space).

Perturbations and ility-driving elements are the essential units behind this dynamic system perspective. They are, in fact, the operators that allow (or impede, in the case of resistance properties) transitions from one state (a trace in the space in Figure 8-1) in the space to another. Figure 8-2 shows all the spaces and operators involved in this view of the design effort. With no path enablers or path inhibitors, the dynamic performance of the system would be in the hands of unintentional operators: perturbations. The inclusion of path enablers and path inhibitors in the design of the system allows for the emergence of two other types of operators: change mechanisms and resistance mechanisms. These are intentional and enable the system to address the occurrence of perturbations. In this way, they drive the emergence of ilities over time.

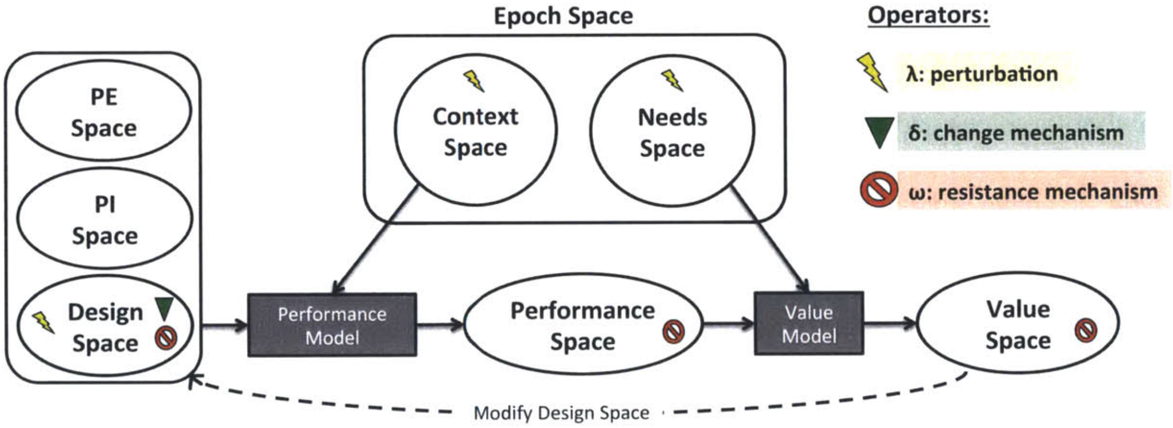


Figure 8-2: Spaces and operators in the dynamic system perspective for design. Perturbations can operate on design, context and needs spaces; change mechanisms on design space only; resistance mechanisms on design, performance and value spaces.

8.2.2.2 Static vs. Contingent Value

The identification of suitable ility-driving elements for inclusion in the design under the dynamic system perspective is not aimed at the enhancement of near-term value delivery of the system. That is, it is not geared toward the fulfillment of *static* functional requirements. Rather, ility-driving elements drive the emergence of *contingent value* – i.e., value that materializes only upon the (imminent, if prevented) occurrence of perturbations. For example, if a system component fails, having a spare component, as well as the ability to replace the faulty component, delivers contingent value to stakeholders. This value is contingent upon the failure of the component (i.e., if the component never fails, the value never materializes).

It is important to point out that there are systems for which dealing with certain critical contingencies is a primary functional requirement. This occurs especially for those systems that are built for a one-time (or few-times) mission. For example, in the building of a spacecraft, a launch abort system that enables the crew to get off the shuttle in case of an emergency (like a rocket suffering a catastrophic failure) is included in the system. This system changes the design of the spacecraft significantly, but it is yet geared toward an unlikely contingency. Within the context of the IDE Analysis proposed in chapter 7 (geared at the identification of relevant IDEs), the abort system would be a (quite important!) latent path enabler that is part of the baseline system architecture on which the analysis is performed.

8.2.2.3 Robust Systems vs. Antifragile Systems

In the best-seller book *Antifragile: Things That Gain From Disorder*, Taleb (2012, pp. 3-4) discusses the concept of antifragility as different from that of robustness:

> Some things benefit from shocks; they thrive and grow when exposed to volatility, randomness, disorder, and stressors and love adventure, risk, and uncertainty. Yet, in spite of the ubiquity of the phenomenon, there is no word for the exact opposite of fragile. Let us call it antifragile. Antifragility is beyond resilience or robustness. The resilient resists shocks and stays the same; the antifragile gets better... Antifragility makes us understand fragility better. Just as we cannot improve health without reducing disease, or increase wealth without first decreasing losses, antifragility and fragility are degrees on a spectrum.

The purpose of introducing an array of ility-driving elements in a system is not only that of sustaining value delivery, but also that of enhancing value delivery. The former is related to robustness ("value robustness" in the lexicon of SEAri), the latter with antifragility. An antifragile system is one that benefits from perturbations in the outer environment. This can happen either by actively exploiting opportunities (e.g., demand increases and flexible production line enables the increase in production – one of the classic change options in business) or passively benefitting from an exogenous shock (e.g., a taxi company that operates only electric cars benefits from a sudden increase in gasoline price – which would be a resistance property with respect to performance). Hence, including a variety of different change options and resistance properties in a system (e.g., using IDE Analysis) has the potential of designing systems that, across their lifecycle, are both robust and antifragile.

8.2.3 On Designing for the Emergence of Ilities and IDE Analysis

Chapter 7 has proposed IDE Analysis as a structured approach for the generation, evaluation and selection of ility-driving elements during the conceptual phase of system design. Using this approach, it is possible to formally think about the introduction of design- and enterprise-level elements that can drive the emergence of ilities in complex systems. For complex sociotechnical systems, although it is impossible to control uncertainty, the inclusion of such elements in the design can enable better management of uncertainty. IDE Analysis is performed with a baseline design of the system in mind. It is scalable in effort, and, as for many tasks performed in conceptual design, its completion involves much creativity and expertise. Lastly, it is important to point out that, rather than being a prescriptive method, IDE Analysis discusses several ways of generating, evaluating and selecting IDEs for inclusion in the system. The user, then, can judge the most relevant and efficient use in his or her case.

The goal of IDE Analysis is not to identify and focus on one or two change options (or resistance properties), as is done, for example, in Real Options Analysis (discussed in chapter 2). Rather, it is intended for generating a broad variety of possible ility-driving elements, from which to choose a set to either include in the system or carry forward for further and more detailed analysis (e.g., modeling and simulation). The generation part is aimed at creating an extensive list of possible relevant IDEs. A few techniques for generating IDEs were proposed, among which of particular relevance is the design principle to perturbation (and perturbation cause) mapping. This technique differs from some of the techniques in the literature (like Change Propagation Analysis or Sensitivity DSM, discussed in chapter 2) as it enables full creativity and IDE search outside of the current design concept. Several proxy metrics intended to inform the comparison (and ultimately selection) of IDEs have also been discussed. Given the scarcity of data for most complex systems of the future, the assessment of these metrics often occurs by the hands of domain experts. Lastly, the problem of selecting a portfolio of IDEs, given an extensive list of IDEs, has been discussed. In particular, given the nature of the space to search (and of the assessments), it is unadvisable (often impossible) to opt for full exploration or analytical optimization of possible portfolios. Rather, the better way to proceed is through the use of visualizations and human-in-the-loop decision making, which enable the interaction of the decision maker with the information about the different ility-driving elements.

8.2.3.1 The Assessment Process and Cognitive Biases

Any assessment process is inherently impacted by biases that the assessor carries with him or her. The assessments in IDE Analysis are no exception, of course. Naturally, if historical data or market data exist for the cost of certain path enablers or inhibitors, then a degree of objectivity exists in the assessment (assuming the data is still relevant to the current problem). However, the assessment of the likelihood and impact of perturbations that rarely occur and for which scarce data exist (e.g., earthquakes, next world wide war, etc.) can be considered an art. Chapters 5 and 7 have highlighted how risk characterization is a decision-relative effort, and is based on an analytic-deliberative process. Its aim is "to describe a potential future situation in "as accurate, thorough, and *decision-relevant* [emphasis added] a manner as possible, addressing the significant concerns of the interested and affected parties." (National Research Council, 1996)

These interested and affected parties that assess the risk induced by a perturbation (or the extent to which an IDE covers the perturbation, or the nature of the uncertainty space, etc.) unavoidably carry with them certain cognitive biases that originate both in their past experience and the ideologies they embrace (Layton, 1976). In particular, below are some salient cognitive biases that have been found to occur frequently in the cognitive psychology literature (Gilovich et al., 2002), and about which the practitioner of IDE Analysis must be aware:

- *Representativeness.* When making judgment using the representativeness heuristic, people estimate the likelihood of an event by comparing it to an existing prototype that already exists in their minds. This prototype is what they think is the most relevant or typical example of a particular event or object, and this may be wrong. Kahneman and Tversky (1972) define this heuristic formally as "the degree to which [an event] (i) is similar in essential characteristics to its parent population, and (ii) reflects the salient features of the process by which it is generated."
- *Availability.* When using this heuristic to judge, people rely on examples and information that comes immediately to mind, i.e.: what is promptly available to them. More precisely, they judge an event as likely or frequent if instances of it are easy to imagine or recall. Its underlying logic is that, if something can be readily recalled, it must be important or common. An outcome of using this heuristic is that people tend to use the most recent information when making judgments, or information to which is given more attention in media.
- *Anchoring.* Anchoring is one of the most well-known biases, often used in marketing strategies. Anchoring occurs when people rely too heavily on the first piece of information that is given to them in their decision making process and then subsequently interpret further information relative to that anchor.
- *Contamination.* This cognitive bias is defined as "unconscious or uncontrollable mental processing that results in unwanted judgment, emotions, or behaviors." (Wilson et al., 2002) It occurs when a judgment made rationally is in contrast with a deep ideology or set of values of the person judging, and is therefore subsequently reevaluated.
- *Overconfidence.* This bias consists in the fact that people's subjective confidence in their judgments is greater than the objective accuracy, especially when confidence is relatively high.
- *Optimism.* This is related to overconfidence, and it can be described by the fact that people tend to assign higher probabilities to their attainment of desirable outcomes than either objective criteria or logical analysis warrants. This bias is in turn linked to the planning fallacy (often observed in complex system design): people overestimate their rate of work or underestimate how long it will take them to get things done.

8.3 On Possible Future Research Efforts

Research toward a systematic definition and treatment of ilities in system design is still in its early stages. This research is built on the strong foundations coming from the work in MIT SEAri and the whole systems engineering community, and it presents a structured way of beginning to think systematically about the design of systems that display desirable lifecycle properties (relative to a space of uncertainty one can conceive and deems relevant). In the following subsections, some opportunities for future research efforts that build on this one are discussed.

8.3.1 Periodic Redesign

In the modeling of the uncertainty space, the perturbation set (Figure 8-1) does not include possible transitions into new context or needs spaces. In fact, although this thesis does acknowledge the existence of $\boldsymbol{C}'$ and $\boldsymbol{N}'$ (as representative of the fact that the modeling of the relevant context space and needs space is subject to change over time, as new information arises), it does not include them formally in the modeling of the dynamic impact of perturbations (and IDEs) over time, and in the IDE Analysis. This is because it is assumed that, at the moment of design, all that is modeled in the context and needs spaces is what is relevant to designers and stakeholders.[50] However, many systems do undergo periodic redesign (e.g., SoS or regulatory systems, as discussed in chapter 6). In fact, as discussed in chapter 2, most Systems of Systems do not appear fully formed: their development and existence is evolutionary with functions and purposes added, removed, and modified with experience, over time. For these, it would be interesting to explore the possibility of modeling changing epoch spaces over time, as new information is revealed and gathered. This would result into a dynamic IDE Analysis, wherein the carriage and switching of portfolios of IDEs is also taken into account as the plausibility of new epochs becomes worthy of consideration.

8.3.2 Relationships Among the Epoch Variables

The combination of possible contexts and possible expectations a system may face forms the epoch space. Although the two sets of context and needs variables are modeled as separate, strong couplings may exist among them. For example, the technological improvements in wireless communications have led people to change their preferences on the use of cell phones (from voice communication, to textual communication, to browsing the internet). Furthermore, the literature on cognitive psychology discusses extensively the context-dependent nature of preferences (Tversky and Simonson, 1993). These considerations lead to an interesting opportunity to model the potential coupling effects between contexts and needs, and take that into consideration in the IDE Analysis.

[50] Except for unknown unknowns that may turn out to be relevant.

Furthermore, modeling the conditional probabilities among all the conceived epoch variables may be an interesting piece of information to incorporate in the IDE Analysis. The Epoch Syncopation Framework (ESF) (Fulcoly et al., 2012) can be a starting point for considering the uncertain sequence of epochs experienced by a system. Using Monte Carlo analysis and Markov probability matrices, ESF analyzes the execution of potential change mechanisms as synced to respond to possible epoch shifts. Information coming from the application of this framework may be used for deciding the number or type of IDEs to include in a system during IDE Analysis. In order to make probabilistic information about these events more grounded in reality, it would be interesting to explore the possibility of using empirical indicators that inform the likelihood of an epoch variable to shift to a different level (as it is often done for informing risk analysis in financial companies and hedge funds).

8.3.3 Informing VASC and Other Methods

In chapter 2, some techniques for the evaluation and valuation of certain lifecycle properties have been discussed. For instance, Real Options Analysis (ROA) attempts to quantify the value of the flexibility given by an option by assigning a monetary value to it (e.g., option to increase production). Another, less prescriptive, method for the valuation of changeability in systems is Valuation Approach for Strategic Changeability (VASC) (Fitzgerald, 2012). This method is designed to "capture the multi-dimensional value of changeability while limiting the number of necessary assumptions." VASC originates under the same dynamic system perspective discussed in this research, and at its core uses a conceptualization of uncertainty akin to that provided in chapter 4. VASC is specifically targeted at the valuation of different change options for inclusion in the system. A very interesting avenue for future research and applications can be to couple IDE Analysis with VASC. In fact, IDE Analysis would provide for the brainstorming and initial down-selection of possible change options to include in the system that could then be valuated more in depth through the application of VASC. In a similar fashion, IDE Analysis could be coupled with the more recent work by Schaffner (2014) on Multi-Era Analysis, wherein a set of change options is also the starting point. More generally, any method or technique geared toward the quantification of any ility (from changeability to robustness) in systems could benefit from the coupling with IDE Analysis.

8.3.4 Interactive Visualization for IDE Selection

The last section in chapter 7 has discussed some visualization techniques for easing the task of processing a lot of information about many different alternative IDEs (or portfolios, at the portfolio-level) in support of selection. The visualization tools briefly discussed in this work are mainly static. However, interactive visualization can be an efficient and effective method for enabling stakeholders to make better decisions (Ricci et al., 2014). Hence, an interesting area for future research and application is the design of interactive visualization tools that are able to augment the cognitive experience of the decision maker when dealing with such large and diverse data.

8.3.5 Modeling in the Dynamic System Perspective for Design

Research opportunities exist for building upon and enhancing the current model of the spaces and their operators (i.e., perturbations and mechanisms), involved in the dynamic system perspective for design. For example, it is possible to imagine a fully connected network that encompasses the design space, context space and needs space – let us call it the *system space*. At every point in time, a system exists as an instance in this space, i.e.: a trace in the diagram in Figure 8-1, for example. Each instance then can be imagined to be in a potential well in the system space.[51] This way, a network of local potential wells forms, with each arc in the network having a certain escape energy[52] associated with it. In such a picture, a perturbation is an operator that provides enough energy for a system instance to escape its current potential well and land into a new one. Similarly, mechanisms are operators at the disposal of designers that can either provide the escape energy to go from one instance to another in the system space (in the case of change mechanisms), or modify the amount of escape energy required (in the case of resistance mechanisms). Some resistance mechanisms may provide more "escape energy increase" than others. In this view, depending on the presence of path enablers and path inhibitors, the energy required for transitions is either provided or increased (respectively).

[51] In physics, a potential well is the region surrounding a local minimum of potential energy. Energy may be released from a potential well if sufficient energy is added to the system such that the local maximum is surmounted (except in quantum physics, where potential energy may escape a potential well by the tunneling effect)

[52] Which could be in terms of any resource including cost, time, effort, and actual physical energy.

9 Conclusion

"Whereof one cannot speak, thereof one must be silent."

– Ludwig Wittgenstein (1922)

The starting point for this thesis and the larger research effort was the need of a method for designing systems capable of revealing desirable lifecycle properties over time – i.e., ilities. Designing for ilities is a strategic effort that involves not only technical knowledge, but also creativity. The general contributions of the research presented in this thesis are with regard to structuring the process of thinking about possible ways of embedding ility-driving elements in the design of a system. In order to do so, this thesis discussed a formal means of modeling uncertainty and its responses, based on a specific view of the design effort: the dynamic system perspective. In this perspective on design, the designer faces three key spaces of possibilities (Figure 9-1): the design space, the context space, and the needs space. Where the system exists in these spaces determines the amount of performance – and therefore value (however measured or conceived) – it delivers to stakeholders. Each space in Figure 9-1 is a collection of instances, i.e.: of possibilities that may materialize at future points in time. As such, under this perspective, any modeling activity in the design effort must be conceived as an exploratory one (Bankes, 1993). This dynamic system perspective on the design effort, then, laid the foundation for modeling uncertainty and its responses, as well as introducing ways of coming up with them.

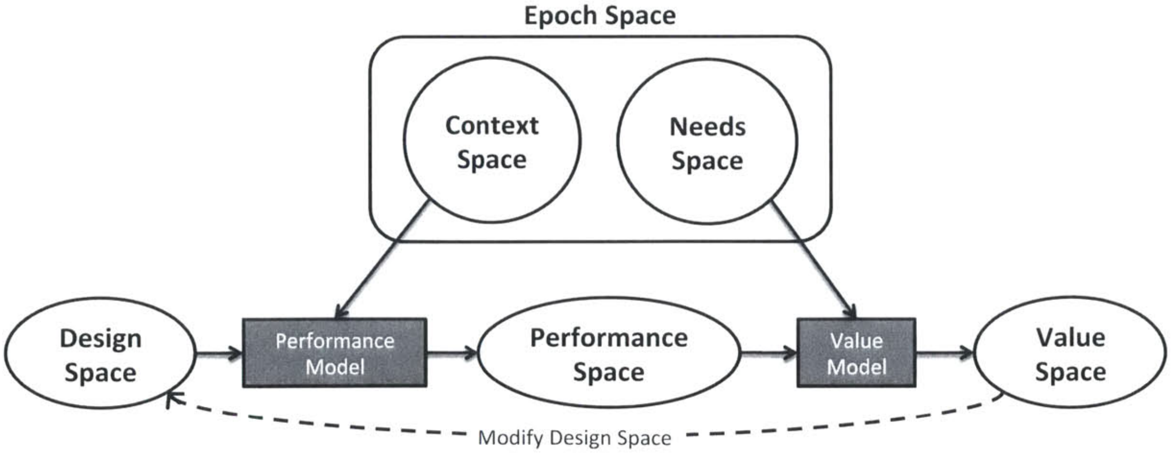


Figure 9-1: System design under the dynamic system perspective.

9.1 Key Contributions

The main domains for which this thesis has provided contributions are (1) the modeling, characterization and identification of uncertainty in system design and (2) the modeling, characterization and identification of ility-driving elements in design. The following paragraphs discuss the key contributions of this thesis, as viewed by the author. The organization and flow reflect that of the scoping questions in chapter 1:

1. On the nature of uncertainty in system design
   a. How can uncertainty be modeled effectively?
   b. What can the dynamic impact of uncertainty be on the system?
   c. In what ways can the modeling of uncertainty be useful?
2. On the nature of responses to uncertainty in system design
   a. What drives the emergence of ilities throughout the lifecycle?
   b. How can one model these ility-driving elements?
   c. What can the dynamic impact of these ility-driving elements be in response to the unfolding of uncertainty?
   d. How have ility-related behaviors emerged in systems of the past (and the present)?
3. On the identification of ility-driving elements
   a. Can there be a structured approach for the generation, evaluation and selection of such elements?

Modeling uncertainty: Based on the dynamic system perspective for design and the description of the spaces involved in it, a formal way of modeling uncertainty in system design was introduced, whereby perturbations are operators on the design, context and needs space. An example, for a perturbation in design, is shown in Figure 9-2, and described below:

$$\lambda^D : \begin{cases} d_i \mapsto d_j \mid d_i, d_j \in \boldsymbol{D} \\ d_i \mapsto d_i{}' \mid d_i \in \boldsymbol{D}, d_i{}' \in \boldsymbol{D}' \end{cases}$$

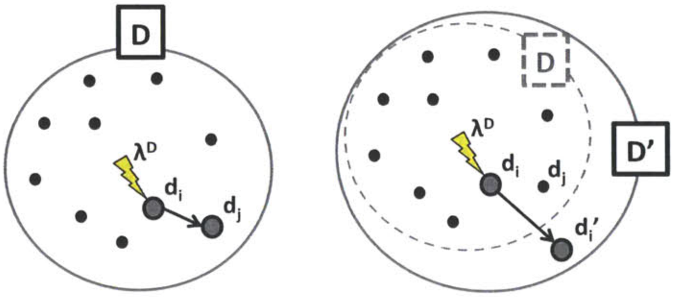


Figure 9-2: Perturbation as an operator on the design space.

Impact of perturbations: The impact of perturbations in design, context or needs on performance and value delivery has been analyzed in all the possible circumstances:

1. Unexpected or imposed changes in the design instance cause the performance (i.e., performance instance) of the system to change (e.g., car's windshield breaks), thereby changing the value delivered.
2. Changes in the context (i.e., context instance) in which the system operates cause the performance (i.e., performance instance) of the system to change (e.g., car goes from asphalt to dirt terrain), thereby changing the value delivered.
3. The expectations (i.e., needs instance) of the system change (e.g., need car to be more fuel efficient, as new job requires longer commutes), thereby changing the value delivered.

For example, for a perturbation in design, the impact of a perturbation on value delivery is shown in Figure 9-3.

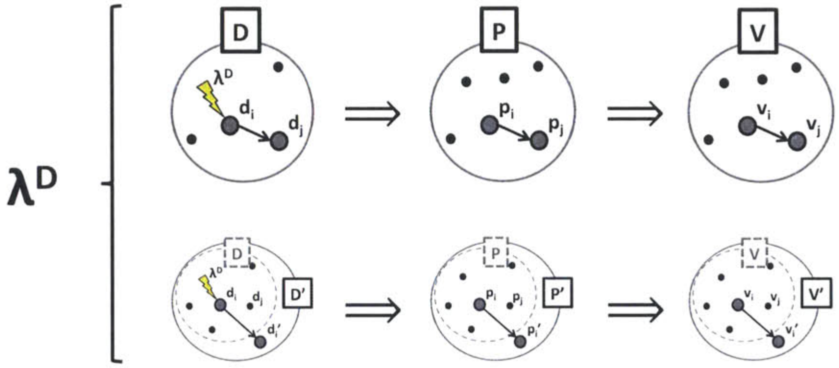


Figure 9-3: Impact on value of a perturbation in design.

Identification of perturbations: An array of techniques and heuristics that can be used for the identification of general uncertainty categories, as well as the parameterization of these into perturbations, have been introduced and applied to the MarSec SoS case. Among these are, for example, enterprise boundary analysis and structured stakeholder interviews.

The nature of ility-driving elements: The thesis research has provided an in depth discussion of the mechanisms through which a system can cope with uncertainty: (1) change mechanisms and (2) resistance mechanisms. These are possible only due to the existence of path enablers and path inhibitors in the design of the system. Ility-driving elements are the union of a path variable and a mechanism.

Modeling ility-driving elements: The modeling of ility-driving elements was three layers deep. First, a set of path enablers and one of path inhibitors was defined; then, change mechanisms and resistance mechanisms were modeled as operators on some of the spaces in the dynamic system perspective for design; lastly, change options and resistance properties were modeled as a combination of the two.

Path enablers and path inhibitors form spaces that, like the design space, are controllable by the designer. It is their inclusion that eventually results in change or resistance mechanisms at later points in time. Furthermore, they can be *in* or *on* the system, and potentially (for those *in* the system) latent in the baseline design.

A change mechanism is an operator on the design space, as shown in Figure 9-4. It is very similar to a perturbation, as it enables the transition from one instance to another. Only, in the case of change mechanisms, transitions are intentional (from the designer's perspective). A change mechanism can be used preemptively or reactively with respect to perturbation occurrence.

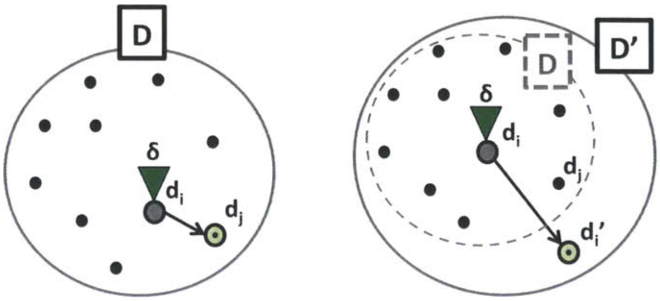


Figure 9-4: Change mechanism as an operator on the design space.

A resistance mechanism is an operator on the design, performance or value space. Its function is that of impeding an undesired transition from one instance to another. Figure 9-5 illustrates the anatomy of a resistance mechanism on the design space. Similar to change mechanisms, resistance mechanisms can be preventive (only in design) or reactive with respect to the occurrence of a perturbation.

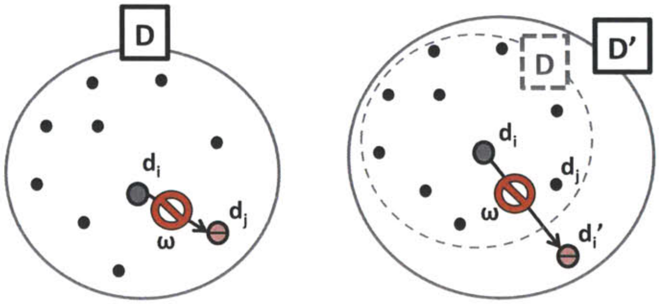


Figure 9-5: Resistance mechanism as an operator on the design space.

Change options and resistance properties, then, were defined in the form of two logical implications (the first, unlike the second, implying executability):

$$CO : \bigwedge_{i=1}^{N} \varepsilon_i \Rightarrow Poss(\delta)$$

$$RP : \bigwedge_{i=1}^{N} \iota_i \Rightarrow \omega$$

Impact of ility-driving elements: The impact of ility-driving elements on the value delivery of a system under all possible circumstances was investigated. A distinction between preventive vs. reactive use of IDEs was made. For example, Figure 9-6 shows a reactive use of a change option upon the occurrence of a perturbation in design.

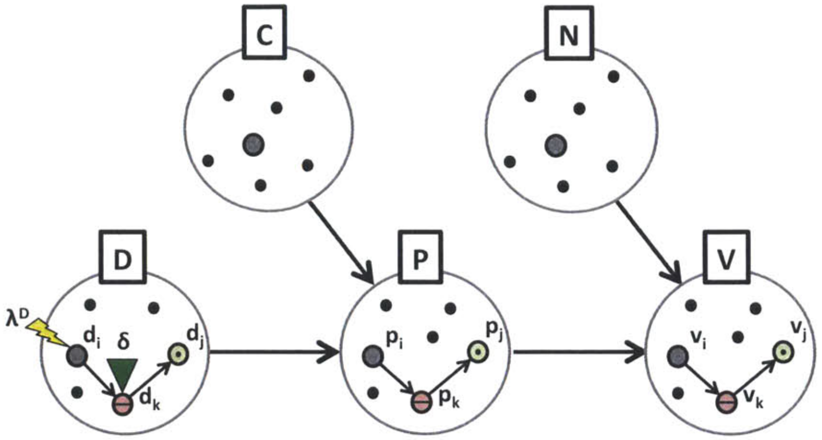


Figure 9-6:Reactive use of change option upon occurrence of perturbation in design.

Characterization of ility-driving elements: A set of descriptive fields for the characterization of ility-driving elements was proposed. The descriptive fields discussed are: decision type, execution agent, epoch-dependent unavailability, reusability, lifecycle, target, cost, execution time, optionability, realizability, risk reduction, and opportunity exploitation.

Investigation of change options: Chapter 6 discussed an empirical investigation of change options. It presented results from an investigation on evolvability in military SoS, as well as planned adaptation in the realm of policymaking (see Figure 9-7). In the case of military SoS, the array of change options found was linked to particular set of architect's intents.

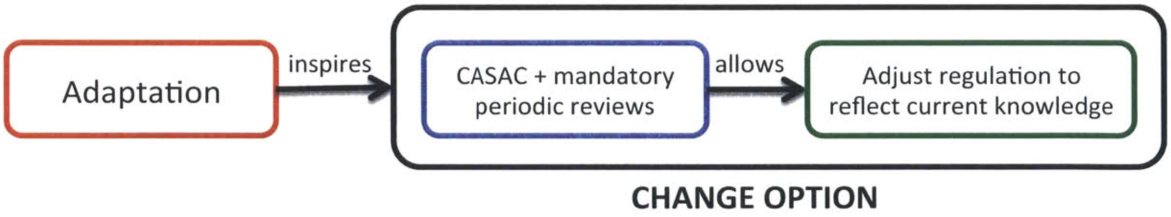

Adaptation
inspires
CASAC + mandatory periodic reviews
allows
Adjust regulation to reflect current knowledge
CHANGE OPTION

Figure 9-7: Change option related to planned adaptation in the PM regulation case.

IDE Analysis: A structured approach for the generation, evaluation and selection of ility-driving elements during the conceptual phase of system design (see Figure 9-8). Using this approach, it is possible to formally think about the introduction of design- and enterprise-level elements that can drive the emergence of ilities in complex systems. IDE Analysis, as intended in this thesis' research, is performed with a baseline design of the system in mind. It is scalable in effort, and, as for many tasks performed in conceptual design, its completion involves much creativity and expertise.

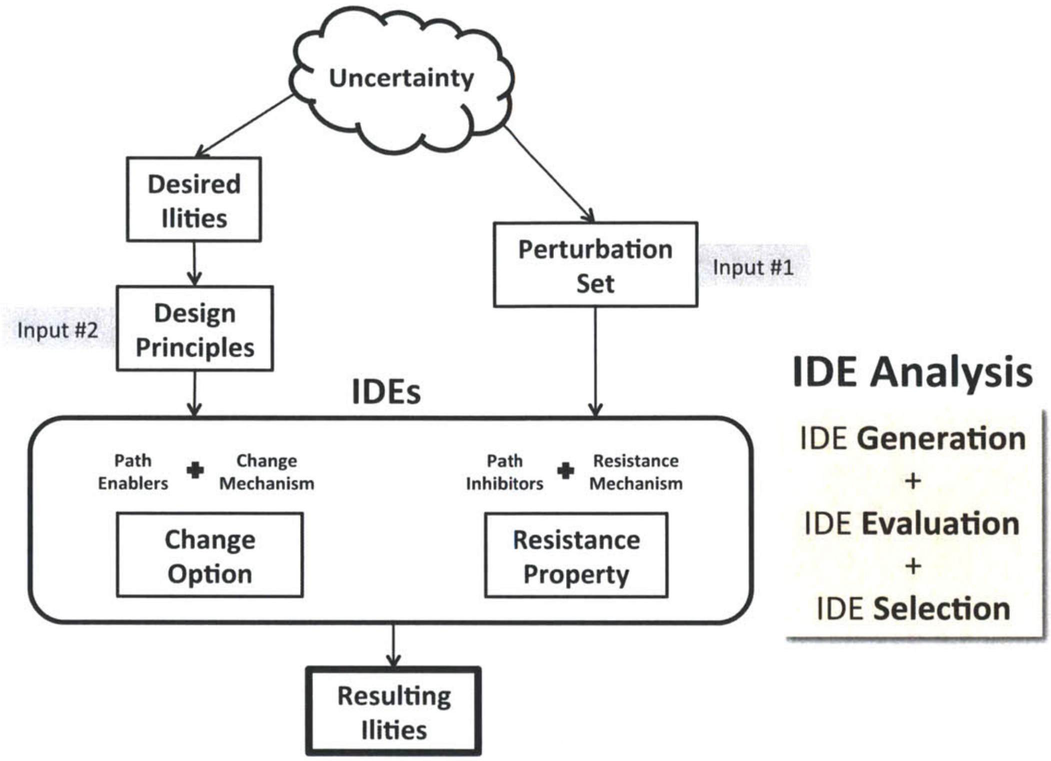


Figure 9-8: General flow from the existence of uncertainty to the emergence of ility behaviors, due to IDEs identified through IDE Analysis.

Generation of ility-driving elements: The first activity in IDE Analysis concerns the generation of an extensive list of IDEs to carry forth for comparison and selection. DPP and DPPC matrices (Figure 9-9) are introduced for facilitating the brainstorming activity. An application to the MarSec SoS was illustrated.

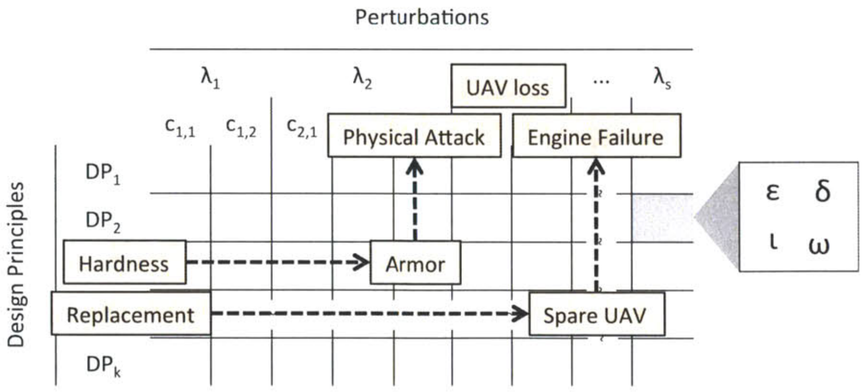


Figure 9-9: Design principles to perturbation cause (DPPC) matrix.

Evaluation of ility-driving elements: The second major activity in IDE Analysis is that of evaluating the different IDEs generated for comparison and selection. A suite of proxy metrics for the evaluation of IDE was introduced: cost, risk attenuation (*PC*, *RR*, *NRR*), optionability, realizability, reusability, and feasible epoch fraction (*FEF*). A Perturbation

Coverage (PC) matrix for the evaluation of risk reduction (or opportunity exploitation) was introduced. Similarly COF and RPF matrices were introduced for (1) the formation of IDEs and (2) the evaluation of optionability and realizability. In addition to IDE-level evaluations, portfolio-level evaluations were also discussed. For example, Figure 9-10 illustrates the evaluation of optionability at the portfolio-level.

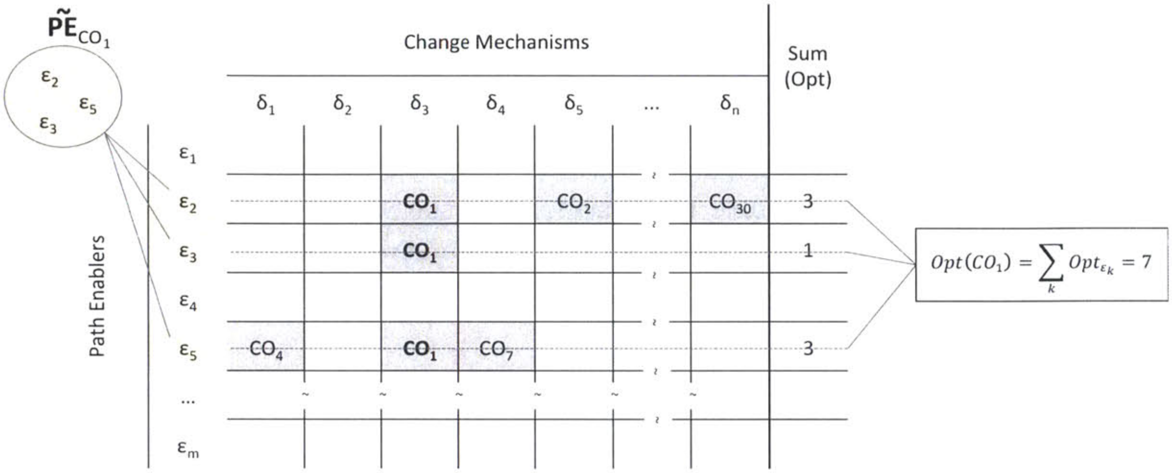


Figure 9-10: Evaluation of optionability at the portfolio-level.

<u>Selection of ility-driving elements</u>: The last activity in IDE Analysis is that of selecting a portfolio of IDEs for either (1) further analysis or (2) direct inclusion in the system. The problem of selection and the curse of dimensionality in the case of portfolios of IDEs were discussed, focusing on analytical and exploratory techniques, as well as some visualization techniques that may turn out useful in the selection process. For example, the use of risk matrices was proposed for comparing the risk reduction profiles of different portfolios of IDEs (e.g., Figure 9-11 in the case of the example application to the MarSec SoS).

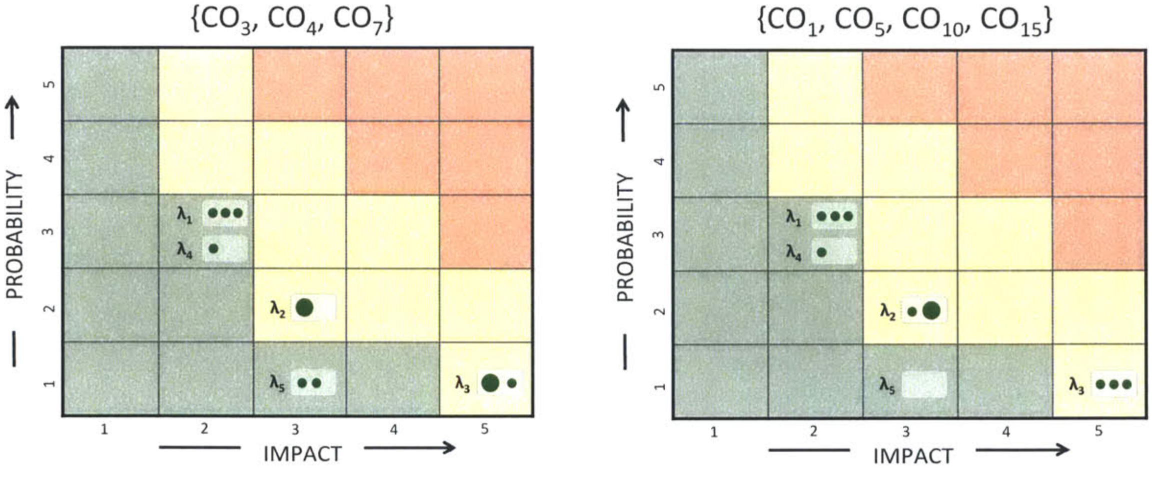


Figure 9-11: Comparison of the risk reduction profiles of two different portfolios of IDEs.

9.2 A Final Thought

The opening quote for this chapter is the famous last proposition in Wittgenstein's *Tractatus Logico-Philosophicus*: "Whereof one cannot speak, thereof one must be silent." Wittgenstein's work was concerned with declarative logic, and specifically the relationship between language and reality to define the limits of science. Unlike science, design is concerned with imperative logic: i.e., *what should be*, not *what is*. In this context, a designer has no hard limits in speaking, thinking, and creating.[53] The content discussed in this thesis lays the foundation for thinking about the design of systems with one goal in mind: the emergence of ilities over time. The modeling effort and the IDE Analysis discussed herein enable the designer with such goals in mind to think about, communicate, and creatively generate possible alternative courses of actions, among which to choose the one that *should* be.

[53] Subject to the constraints imposed by time, resources, physics, and one's imagination.

List of Relevant Acronyms and Symbols

Acronyms

AHP: Analytic Hierarchy Process

AOI: Area Of Interest

BMDS: Ballistic Missile Defense System

C2BMC: Command and Control Battle Management System

CASAC: Clean Air Scientific Advisory Committee

COF: Change Option Formation

CONOPs: Concepts of Operations

CPA: Change Propagation Analysis

DCF: Discounted Cash Flow

DISA: Defense Information Systems Agency

DPP matrix: Design Principle to Perturbation matrix

DPPC matrix: Design Principle to Perturbation Cause matrix

DSM: Design Structure Matrix

DSP: Defense Support Program

EEA: Epoch-Era Analysis

efNPT: fuzzy effective Normalized Pareto Trace

eNPT: effective Normalized Pareto Trace

ESD: Engineering Systems Division

fNPT: fuzzy Normalized Pareto Trace

FOD: Filtered Outdegree

FPS: Fuzzy Pareto Shift

IDE: ility-driving element

JIE: Joint Information Enterprise

MarSec: Maritime Security

MATE: Multi-Attribute Tradespace Exploration

MAU: Multi-Attribute Utility

MAUT: Multi-Attribute Utility Theory

MPT: Modern Portfolio Theory

NPT: Normalized Pareto Trace

NPV: Net Present Value

PAA: Phase Adaptive Approach

PC: Perturbation Coverage

PMPT: Post-Modern Portfolio Theory

QFD: Quality Function Deployment

REM: Rule-Effects Matrix

ROA: Real Options Analysis

RPF: Resistance Property Formation

SAI: SoS Architecting with Ilities

SEAri: Systems Engineering Advancement Research Initiative

SoS: System of Systems

SoSE: Systems of Systems Engineering

SSRC: Sociotechnical Systems Research Center

TAUL: Time-weighted Average Utility Loss

TDN: Time-expanded Decision Network

THAAD: Terminal High Altitude Area Defense

UAV: Unmanned Aerial Vehicle

VASC: Valuation Approach for Strategic Changeability

Symbols

A: attribute set (wherein each element is an attribute *a*)

C: context space (wherein each element is a context instance *c*)

C: cost (metric)

CM: change mechanism set (wherein each element is a change mechanism δ)

CO: change option set (wherein each element is a change option *CO*)

CV: context variable set (wherein each element is a context variable *cv*)

D: design space (wherein each element is a design instance *d*)

DV: design variable set (wherein each element is a design variable *dv*)

E: epoch space (wherein each element is an epoch instance *e*)

EV: epoch variable set (wherein each element is an epoch variable *ev*)

FEF: feasible epoch fraction (metric)

IDE: ility-driving element set (wherein each element is an ility-driving element *IDE*)

N: needs space (wherein each element is a needs instance *n*)

NOE: normalized opportunity exploitation (metric)

NRR: normalized risk reduction (metric)

NV: needs variable set (wherein each element is a needs variable *nv*)

OE: opportunity exploitation (metric)

Opt: optionability (metric)

P: performance space (wherein each element is a performance instance *p*)

PC: perturbation coverage (metric)

PE: path enabler set (wherein each element is a path enabler ε)

$\boldsymbol{PE_{in}}$: set of path enablers in the system

$\boldsymbol{PE_{latent}}$: set of latent path enablers

$\boldsymbol{PE_{on}}$: set of path enablers on the system

PI: path inhibitor set (wherein each element is a path inhibitor ι)

$\boldsymbol{PI_{in}}$: set of path inhibitors in the system

$\boldsymbol{PI_{latent}}$: set of latent path inhibitors

$\boldsymbol{PI_{on}}$: set of path inhibitors on the system

Reus: reusability (metric)

RM: resistance mechanism set (wherein each element is a resistance mechanism ω)

RP: resistance property set (wherein each element is a resistance property RP)

RR: risk reduction (metric)

Rz: realizability (metric)

V: value space (wherein each element is a value instance *v*)

δ: change mechanism

ε: path enabler instance

ι: path inhibitor instance

$\boldsymbol{\Lambda}$: perturbation set (wherein each element is a perturbation λ)

λ^C: perturbation in context

λ^D: perturbation in design

λ^N: perturbation in needs

ω: resistance mechanism

References

[1] Abbass HA, Bender A, Dam HH, Baker S, Whitacre J, Sarker R. Computational scenario-based capability planning. GECCO 2008, July 12–16, 2008, Atlanta, Georgia, USA.

[2] Bachelier L. *Théorie de la spéculation.* Annales Scientifiques de l'École Normale Supérieure 3 (17): 21–86. (1900)

[3] Bankes S. Exploratory modeling for policy analysis. *Operations Research* 41: 435-449. (1993)

[4] Bar-Ilan J, Kortsarz G, Peleg D. Generalized submodular cover problems and applications. Proc. 4th Israel Symp. on the Theory of Computing and Systems, Jerusalem, Israel, June 1996.

[5] Bayes T. An Essay towards Solving a Problem in the Doctrine of Chances. By the Late Rev. Mr. Bayes, F. R. S. Communicated by Mr. Price, in a Letter to John Canton, A. M. F. R. S. Phil. Trans. January 1, 1763 53 370-418; doi:10.1098/rstl.1763.0053. (1763)

[6] Beesemyer JC, Fulcoly DO, Ross AM, Rhodes DH. Developing methods to design for evolvability: research approach and preliminary design principles. *9th Conference on Systems Engineering Research*, Los Angeles, CA, April 2011.

[7] Beesemyer JC, Ross AM, Rhodes DH. Case Studies of Historical Epoch Shifts: Impacts on Space Systems and their Responses*. AIAA Space 2012*, Pasadena, CA, September 2012.

[8] Beesemyer JC. Empirically characterizing evolvability and changeability in engineering systems. *Master of Science Thesis*, Aeronautics and Astronautics, MIT, June 2012.

[9] Bergey JK, Blanchette Jr S, Clements PC, Gagliardi MJ, Klein J, Wojcik R, Wood B. *U.S. Army Workshop on Exploring Enterprise, System of Systems, System, and Software Architectures.* Technical Report, CMU/SEI-2009-TR-008, ESC-TR-2009-008, SEI Administrative Agent. Copyright 2009 Carnegie Mellon University.

[10] Bertsimas D, Brown DB, Caramanis C. Theory and applications of robust optimization. *Society for Industrial and Applied Mathematics*, Vol. 53, No. 3, pp. 464-501. (2011)

[11] Black F, Scholes M. The pricing of options and corporate liabilities. *The Journal of Political Economy*, Volume 81, Issue 3 (May – June, 1973), 637-654.

[12] Blanchard BS, Fabrycky WJ. *Systems engineering and analysis.* International Series in Systems Engineering. (2006)

[13] Boardman J, Sauser B. System of Systems – the meaning of. Presented at *IEEE/SMC International Conference on System of Systems Engineering 2006*, Los Angeles, CA. (2006)

[14] Booton R, Ramo S. The Development of Systems Engineering. *IEEE Aerospace and Electronic Systems,* vol. AES-20, no.4, pp.306-310, July 1984.

[15] Boyle PP. Options: A Monte Carlo approach. *Journal of Financial Economics*, 4:323-338, 1977.

[16] Buchanan R. Wicked Problems in Design Thinking. In *Design Issues* 8.2 (Spring), pp. 5–21. (1992)

[17] Butterfield ML, Pearlman JS, Vickroy SC. A System-of-Systems Engineering GEOSS: Architectural Approach. *Systems Journal, IEEE*, vol.2, no.3, pp.321-332, September 2008.

[18] Câmara A. Valuation of event-contingent options. *The Journal of Financial Research*. Vol. XXIX, No. 4, pages 537-557. (2006)

[19] Chase WP. Management of System Engineering. R.E. Krieger. Malabar, FL. 1984.

[20] Churchman CW. Wicked problems. Guest editorial, *Management Science*, 14, No. 4, Dec. 1967, B141-142.

[21] Colagrave N, Collins S. Experimental evolution: experimental evolution and evolveability. *Heredity*, 100, pp. 464-470, 2008.

[22] Collina TZ. The European phase adaptive approach at a glance. *Arms Control Association* (2013). Retrievable at: http://www.armscontrol.org

[23] Copeland T, Antikarov V. *Real Options: A Practitioner's Guide*. Texere, 2001.

[24] Cox JC, Ross SA, Rubinstein M. Option pricing: A simplified approach. *Journal of Financial Economics*, 7:229-263, 1979.

[25] Cox JC, Rubinstein M. *Options Markets*. Prentice Hall, 1985.

[26] Dahmann J, Rebovich G, Lowry R, Lane J, Baldwin K. An implementers' view of systems engineering for systems of systems. *Systems Conference (SysCon)*, 2011 IEEE International, pages 212-217.

[27] Dahmann JS, Baldwin KJ. Understanding the current state of us defense systems of systems and the implications for systems engineering. In *IEEE Systems Conference.* (2008)

[28] Dahmann JS. Impact of systems of systems on acquisition programs. The MITRE Corporation, presented to the *System of Systems Engineering Collaborators Information Exchange* (SoSECIE), May 22nd, 2012.

[29] de Finetti B. Foresight: Its Logical Laws, Its Subjective Sources. *Annales de l'Institut Henri Poincaré*, Vol. 7. (1937) [Translated by Henry E. Kyburg, Jr.]

[30] de Marchi B. Uncertainty in Environmental Emergencies: A Diagnostic Tool. *Journal of Contingencies and Crisis Management* 3 (2), 103–112. (1995)

[31] de Neufville R et al. Uncertainty Management for Engineering Systems Planning and Design. *MIT International Engineering Systems Symposium*, Monograph, MIT, Cambridge, MA. March 2004.

[32] de Weck OL, de Neufville R, Chaize M. Staged Deployment of Communications Satellite Constellations in Low Earth Orbit. *Journal of Aerospace Computing, Information, and Communication*, Vol. 1, No.3, pp. 119-136, 2004.

[33] de Weck OL, Eckert C, Clarkson J. A Classification of Uncertainty for Early Product and System Design. In *16th International Conference on Engineering Design*, 2007.

[34] de Weck OL, Jones MB. Isoperformance: analysis and design of complex systems with desired outcomes. *Systems Engineering*, Vol. 9, No. 1, 2006.

[35] de Weck OL, Ross AM, Rhodes DH. Investigating Relationships and Semantic Sets amongst System Lifecycle Properties (Ilities). *3rd International Conference on Engineering Systems*, TU Delft, the Netherlands, June 2012.

[36] Deshmukh A, Boehm B, Jacques D, Housel T, Sullivan K, Collopy P. RT-18: Value of Flexibility. Phase I Progress Report, SERC-2010-TR-10. (2010)

[37] DISA (Defense Information Systems Agency) website. (http://www.disa.mil). Accessed on May 5, 2014.

[38] Dubois D, Prade H. A Survey of Belief Revision and Updating Rules in Various Uncertainty Models. International Journal of Intelligent Systems, Vol. 9, 61-100. (1994)

[39] Eichler HG, Oye K, Baird LG, Abadie E, Brown J, Drum CL, Ferguson J, Garner S, Honig P, Hukkelhoven M, Lim JCW, Lim R, Lumpkin MM, Neil G, O'Rourke B, Pezalla E, Shoda D, Seyfert-Margolis V, Sigal EV, Sobotka J, Tan D, Unger TF, Hirsch G. Adaptive licensing: taking the next step in the evolution of drug approval. Nature Publishing Group, Vol. 91, No. 3, March 2012.

[40] Eisner H, Marciniak J, McMillan R. Computer-aided system of systems engineering. Presented at *IEEE Conference on Systems, Man, and Cybernetics*, Charlottesville, VA. (1991)

[41] Ellison R, Woody C. Survivability Challenges for Systems of Systems. News at SEI, http://www.sei.cmu.edu/library/abstracts/news-at-sei/securitymatters200706.cfm. (2007)

[42] Feng W, Crawley EF. *Stakeholder value network analysis for large oil and gas projects.* Research Report, Engineering Systems Division. Cambridge, MA: Massachusetts Institute of Technology. (2008)

[43] Field FR, III. Knowledge and Fallible Indicators. Lecture in: ESD.101, *Concepts and Research in Technology and Policy.* Massachusetts Institute of Technology, Cambridge, MA (2014, April 23rd).

[44] Fischhoff B. Value elicitation: is there anything in there? *American Psychologist* 46.8 (1991): 835.

[45] Fitzgerald ME, Ross AM, Rhodes DH. Assessing uncertain benefits: a valuation approach for strategic changeability (VASC). *INCOSE International Symposium 2012*, Rome, Italy, July 2012.

[46] Fitzgerald ME. *Managing Uncertainty in Systems with a Valuation Approach for Strategic Changeability.* Master of Science Thesis, Aeronautics and Astronautics, MIT, June 2012.

[47] Forrester JW. System dynamics, system thinking, and soft OR. System Dynamics Review Vol. 10, no. 2-3, pp. 245-256. (Summer-Fall, 1994)

[48] Fricke E, Schulz AP. Design for Changeability (DfC): Principles to Enable Changes in Systems Throughout Their Entire Lifecycle. *Systems Engineering* - 8(4): 342-359, 2005.

[49] Fulcoly DO, Ross AM, Rhodes DH. Evaluating system change options and timing using the epoch syncopation framework. *10th Conference on Systems Engineering Research*, St. Louis, MO, March 2012.

[50] Funtowicz SO, Ravetz JR. Uncertainty and Quality in Science for Policy. *Kluwer Academic Publishers*, Dordrecht. (1990)

[51] Giffin M, Keller R, Eckert C, de Weck O, Bounova G and Clarkson PJ. Change Propagation Analysis in Complex Technical Systems. *J. Mech. Des.* 131(8), 081001 (Jul 09, 2009).

[52] Gilovich T, Griffin D, Kahneman D. *Heuristics and biases: the psychology of intuitive judgement.* Cambridge University Press, 2002.

[53] Hall AD. Metasystems methodology: a new synthesis and unification. New York: Pergamon Press, 1989.

[54] Hammond KR. *Human Judgment and Social Policy: Irreducible Uncertainty, Inevitable Error, Unavoidable Injustice.* 1st. New York, NY: Oxford University Press, Chapter 1, "Irreducible Uncertainty and the Need for Judgment," pp. 13-35; Chapter 2, "Duality of Error and Policy Formation," pp. 36–59. (1996)

[55] Health Effects Institute (HEI). Health Effects Institute, Reanalysis of the Harvard Six Cities Study and the American Cancer Society Study of Particulate Air Pollution and Mortality, pp. i-iv and 1-6, July 2006.

[56] Helton JC. Treatment of Uncertainty in Performance Assessments for Complex Systems. *Risk Analysis* 14 (4), 483–511. (1994)

[57] Henrion M, Fischhoff B. Assessing Uncertainty in Physical Constants. *Annual Journal of Physics* 54 (9), 791–797. (1986)

[58] Hoffman FO, Hammonds JS. Propagation of uncertainty in risk assessments: The need to distinguish between uncertainty due to lack of knowledge and uncertainty due to variability. *Risk Analysis* 14: 707-712. (1994)

[59] Hull JC. Options, Futures, and Other Derivatives. Prentice Hall, 2006.

[60] Jamshidi M. Systems of systems engineering: principles and applications. Boca Raton, FL: CRC Press. (2009)

[61] Jasanoff S, Wynne B. Science and Decision-Making. *In Rayner, S. and Malone, E. L. (eds.), Science and Decision-Making*, Battelle Press, Washington, D.C. (1998)

[62] Kahneman D, Tversky A. Subjective probability: A judgment of representativeness. In Kahneman, Slovic, Tversky. *Judgment under uncertainty: Heuristics and biases.* Cambridge: Cambridge University Press. (1972)

[63] Kahneman D, Tversky A. Variants of Uncertainty. In Kahneman D, Slovic P and Tversky A (eds.), *Variants of Uncertainty.* Cambridge University Press, Cambridge, U.K. (1982)

[64] Kalligeros K. *Platforms and Real Options in Large-Scale Engineering Systems.* Doctoral Dissertation in Engineering Systems, Massachusetts Institute of Technology. (2006).

[65] Keeney RL, Raiffa H. *Decisions with multiple objectives: preference and value tradeoffs.* Published by John Wiley & Sons, Inc., Ch. 2. (1976)

[66] Keeney RL. Value-focused thinking: a path to creative decisionmaking. Harvard University Press (February 1, 1996).

[67] Klir GJ. Uncertainty Theories, Measures and Principles. In Natke HG and Ben-Haim Y (eds.), *Uncertainty Theories, Measures and Principles*, Akademie Verlag, Berlin, Germany. (1996)

[68] Koopmans TC. Three Essays on the State of Economic Science. New York, U.S.A. (1957)

[69] Krause A, Guestrin C. Near-optimal observation selection using submodular functions. Association for the Advancement of Artificial Intelligence (www.aaai.org), 2007.

[70] Kuhn TS. *The Structure of Scientific Revolutions.* 1st. ed., Chicago: Univ. of Chicago Pr., 1962.

[71] Lavesque H, Pirri F, Reiter R. Foundations for the situation calculus. *Electronic Transactions of Artificial Intelligence.* 2 (3-4): 159-178. (1998)

[72] Layton ET, Jr. American Ideologies of Science and Engineering. *Technology and Culture*, Vol. 17, No. 4, Oct. 1976.

[73] Leveson N. *Safeware: system safety and computers*: ACM New York, NY, USA. (1995)

[74] Lobo MS, Boyd S. The worst-case risk of a portfolio. Working paper. (2000)

[75] Machiavelli N. *Il Principe.* Cap. XXV. (1532)

[76] Maier MW. Architecting principles for systems of systems. In *Systems Engineering, volume 1, issue 4*: pp. 267-284, 1998.

[77] Markowitz HM. Portfolio Selection. *The Journal of Finance.* 7(1): 77-91. (1952)

[78] McCarthy J. Actions and other events in situation calculus. Computer Science Department, Stanford University. http://www-formal.stanford.edu/jmc/ (2002)

[79] McCray LE, Oye KA, Petersen AC. Planed adaptation in risk regulation: an initial survey of US environmental, health and safety regulation. *Technol. Forecast. Soc. Change* (2010), doi:10.1016/j.techfore.2009.12.001.

[80] McManus H, Hastings DE. A Framework for Understanding Uncertainty and Its Mitigation and Exploitation in Complex System. *IEEE Engineering Management Review*, Vol. 34, No. 3, pp. 81-94, Third Quarter 2006.

[81] McManus HM, Richards MG, Ross AM and Hastings DE. A Framework for Incorporating "ilities" in Tradespace Studies. *AIAA Space 2007*, Long Beach, CA, September 2007.

[82] Mekdeci B, Ross AM, Rhodes DH and Hastings DE. Examining Survivability of Systems of Systems. *INCOSE International Symposium 2011*, Denver, CO, June 2011.

[83] Mekdeci B, Ross AM, Rhodes DH, and Hastings DE. A taxonomy of perturbations: determining the ways that systems lose value. *6th Annual IEEE Systems Conference*, Vancouver, Canada, March 2012.

[84] Mekdeci B, Ross AM, Rhodes DH, Hastings DE. Investigating alternative concepts of operations for a maritime security system of systems. *INCOSE International Symposium 2012*, Rome, Italy, July 2012b.

[85] Mekdeci B. Managing the Impact of Change through Survivability and Pliability to Achieve Viable Systems of Systems. Doctor of Philosophy Dissertation, Engineering Systems Division, MIT, February 2013.

[86] Mikaelian T. An Integrated Real Options Framework for Model-based Identification and Valuation of Options under Uncertainty. Doctor of Philosophy Dissertation, Aeronautics and Astronautics, MIT, June 2009.

[87] Montaigne Md. *Les essais de Michel seigneur de Montaigne.* Edition nouuelle, / trouuee après le deceds de l'autheur, reueuë & augmentée par luy d'vn tiers plus qu'aux precedentes impressions. Paris: Michel Sonnius, 1595.

[88] Myers SC. *Finance theory and financial strategy.* Interfaces, 14(1):126{137, 1984.

[89] National Aeronautics and Space Administration. *NASA Handbook of Systems Engineering.* (2012)

[90] National Research Council. *Understanding Risk: Informing Decisions in a Democratic Society.* Paul C. Stern and Harvey V. Fineberg, Editors; Committee on Risk Characterization, NRC. ISBN: 0-309-57849-3, 264 pages, 6 x 9. (1996)

[91] Natke HG, Ben-Haim Y. Uncertainty: A Discussion from Various Points of View. In Natke, H. G. and Ben-Haim, Y. (eds.), *Uncertainty: A Discussion from Various Points of View*, Akademie Verlag, Berlin, Germany. (1996)

[92] Neches R, Madni AM. Towards affordably adaptable and effective systems. *Systems Engineering Journal.* Article first published online: 19 Oct. 2012. DOI: 10.1002/sys.21234.

[93] Pareto V. *Manual of Political Economy.* 2d ed. Translated by A. S. Schwier and A. N. Page [1972]. London: Macmillan. (1906)

[94] Pavese C. *La luna e i falò.* Giulio Einaudi Editore, Torino. (1950)

[95] Pielke Jr RA. The Role of Models in Prediction for Decision. Draft prepared for *Cary Conference Discussion IX.* (2001)

[96] Popper K. The Rationality Principle. In *Popper Selections*, edited by David W. Miller [1986], pp. 357-365. (1967)

[97] Rader AA, Ross AM, Rhodes DH. A Methodological Comparison of Monte Carlo Methods and Epoch-Era Analysis for System Assessment in Uncertain Environments. *4th Annual IEEE Systems Conference*, San Diego, CA, April 2010.

[98] Rhodes DH, Hastings D. The Case for Evolving Systems Engineering as a Field within Engineering Systems*. MIT Engineering Systems Symposium*, March 2004.

[99] Rhodes DH, Ross AM. Five aspects of engineering complex systems: emerging constructs and methods. *4th Annual IEEE Systems Conference*, San Diego, CA, April 2010.

[100] Ricci N, Ross AM, Rhodes DH, Fitzgerald ME. Considering alternative strategies for value sustainment in systems-of-systems. *7th Annual IEEE Systems Conference*, Orlando, FL, April 2013.

[101] Ricci, N., Schaffner, M.A., Ross, A.M., Rhodes, D.H., and Fitzgerald, M.E., "Exploring Stakeholder Value Models Via Interactive Visualization," 12th Conference on Systems Engineering Research, Redondo Beach, CA, March 2014.

[102] Richards MG, Ross AM, Hastings DE, Rhodes DH. Multi-attribute tradespace exploration for survivability. *7th Conference on Systems Engineering Research*, Loughborough University, UK, April 2009.

[103] Rittel HWJ, Webber MM. Dilemmas in a General Theory of Planning. Panel on Policy Sciences, *American Association for the Advancement of Science*. 4 (1969): 155–169.

[104] Ross AM, Beesemyer JC, Rhodes DH. A Prescriptive Semantic Basis for System Lifecycle Properties. WP-2011-2-2, available at seari.mit.edu

[105] Ross AM, Hastings DE, Warmkessel JM, Diller NP. Multi-Attribute Tradespace Exploration as a Front-End for Effective Space System Design. *AIAA Journal of Spacecraft and Rockets*, pp. 20-28, Jan/Feb 2004.

[106] Ross AM, McManus HL, Long A, Richards MG, Rhodes DH, Hastings DE. Responsive Systems Comparison Method: Case Study in Assessing Future Designs in the Presence of Change. *AIAA Space 2008*, San Diego, CA, September 2008.

[107] Ross AM, McManus HL, Rhodes DH, Hastings DE, Long AM. Responsive systems comparison method: dynamic insights into designing a satellite radar system. *AIAA Space* 2009, Pasadena, CA, September 2009.

[108] Ross AM, O'Neill MG, Hastings DE, Rhodes DH. Aligning Perspectives and Methods for Value-Driven Design. *AIAA Space 2010*, Anaheim, CA, September 2010.

[109] Ross AM, Rhodes DH, Hastings DE. Defining Changeability: Reconciling Flexibility, Adaptability, Scalability, Modifiability, and Robustness for Maintaining Lifecycle Value. *Systems Engineering*, Vol. 11, No. 3, pp. 246-262, Fall 2008.

[110] Ross AM, Rhodes DH. Architecting Systems for Value Robustness: Research Motivations and Progress. *2nd Annual IEEE Systems Conference*, Montreal, Canada, April 2008.

[111] Ross AM, Rhodes DH. Using Natural Value-centric Time Scales for Conceptualizing System Timelines through Epoch-Era Analysis. *INCOSE International Symposium 2008*, Utrecht, the Netherlands, June 2008.

[112] Ross AM. *Managing Unarticulated Value: Changeability in Multi-Attribute Tradespace Exploration*. Doctor of Philosophy Dissertation Dissertation, Engineering Systems Division, MIT, June 2006.

[113] Ross AM. Multi-Attribute Tradespace Exploration with Concurrent Design as a Value-centric Framework for Space System Architecture and Design. Dual Master of Science Thesis, Aeronautics and Astronautics and Technology and Policy Program, MIT, June 2003.

[114] Rowe G, Wright G. The Delphi technique as a forecasting tool: issues and analysis. *International Journal of Forecasting*, Vol. 15. (1999)

[115] Saaty TL. Decision making – the analytic hierarchy and network process (AHP/ANP). *Jurnl of Systems Science and Systems Engineering*, Vol. 13, No.1, 2004, pp.1-35.

[116] Saleh JH, Hastings DE, Newman DJ. Flexibility in System Design and Implications for Aerospace Systems. *Acta Astronautica*, Vol. 53, 2003.

[117] Schaffner MA, Ross AM, Rhodes DH. A Method for Selecting Affordable System Concepts: A Case Application to Naval Ship Design. *12th Conference on Systems Engineering Research*, Redondo Beach, CA, March 2014.

[118] Schaffner MA. Designing Systems for Many Possible Futures: the RSC-based Method for Affordable Concept Selection, with Multi-Era Analysis. Master of Science Thesis, Aeronautics and Astronautics, MIT, June 2014.

[119] Schoemaker PJH. Multiple scenario development: Its conceptual and behavioural foundation. *Strategic Management Journal* 14: 193–213. (1993)

[120] Schrödinger SE. Quantisierung als Eigenwertproblem [Quantification of the eigen value problem]. *Annalen der Physik* (in German) 80 (13): 437–490. (1926)

[121] SEAri (Systems Engineering Advancement Research Initiative) website - *SEAri at MIT* (http://seari.mit.edu/). Accessed on May 2, 2014.

[122] Silver MR, de Weck OL. Time-expanded decision networks: A framework for designing evolvable complex systems. *Systems Engineering* , vol. 10, no. 2, pp. 167-188, 2007.

[123] Silver N. *The Signal and the Noise*. Published by The Penguin Press, a member of Penguin Group, Inc., pp. 20. (2012)

[124] Simon HA. Prediction and prescription in systems modeling. *Operations Research*, Vol. 38, No. 1, Jan. - Feb. (1990)

[125] Simon HA. *The sciences of the artificial.* Third Edition. Cambridge, MA. (1996)

[126] Slovic P. The construction of preference. *American psychologist* 50.5 (1995): 364.

[127] Sterman MW. Business Dynamics: Systems Thinking and Modeling for a Complex World. Irwin/McGraw-Hill. (2000)

[128] Stevens SS. On the theory of scales of measurement. Science, Vol. 103, No. 2684. Friday, June 7, 1946.

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

[130] Stewart T. Uncertainty, judgment and error in prediction. Chapter 3 in D. Sarewitz, R.A. Pielke Jr., and R. Byerly, editors. *Prediction: Science, Decision Making, and the Future of Nature.* Washington, DC: Island Press. (2000)

[131] Sussman GJ. *Building robust systems.* An essay, MIT, http://groups.csail.mit.edu/mac/users/gjs/essays/robust-systems.pdf (2007)

[132] Swisher P, Kasten GW. Post-Modern Portfolio Theory. *Journal of Financial Planning*, 26 (9): 1-11. (2005)

[133] Thunnissen DP. Uncertainty classification for the design and development of complex systems. Proceedings of the *3rd Annual Predictive Methods Conference*, Veros Software. (2003)

[134] Trigeorgis L. Real Options: Managerial Flexibility and Strategy in Resource Allocation. The MIT Press, 1998.

[135] Tütüncü RH, Koenig M. Robust asset allocation. *Annals of Operations Research*, 132:157-187, 2004.

[136] Tversky A, Kahneman D. Rational choice and the framing of decisions. *Journal of Business*, 59:4, 5251-78. Copyright 1986 by The University of Chicago.

[137] Tversky A, Simonson, I. Context-Dependent Preferences. *Management Science*. 39, 1179–1189. (1993)

[138] Ulrich W. The Metaphysics of Design: A Simon-Churchman 'Debate'. In *Interfaces* 10.2, pp. 35–40. (Apr 1980)

[139] US Senate, Select Committee on Intelligence. Report on the U.S. Intelligence Community's Prewar Assessments on Iraq: Conclusions. Overall Conclusions on Weapons of Mass Destruction. pp. 1-14, July 2004.

[140] van Asselt MBA, Rotmans J. Uncertainty in Integrated Assessment Modeling. *Climatic Change* 54: 75–105, 2002.

[141] van Witteloostuijn A. Uncertainty in Psychology: A Look beyond the Non-Differentiated Approach. Maastricht University, Maastricht. (1987)

[142] Viscito L, Ross AM. Combining pareto trace and filtered outdegree as a metric for identifying valuably flexible designs. In *7th Conference on Systems Engineering Research*, Loughborough, UK, 2009.

[143] von Schomberg R. Controversies and Political Decision Making. In von Schomberg R (eds.), *Controversies and Political Decision Making*, Kluwer Academic Publishers, Dordrecht. (1993)

[144] Walton MA. Managing Uncertainty in Space Systems Conceptual Design Using Portfolio Theory. PhD thesis, MIT, 2002.

[145] Wang T, de Neufville R. Identification of real options "in" projects. *16th Annual International Symposium of the International Council on Systems Engineering*, Orlando, FL, July 2006.

[146] Wasson CS. *Systems analysis, design and development.* Published by John Wiley and Sons, Inc., Hoboken, New Jersey, 2006.

[147] Weber J. A response to public administration's lack of a general theory of uncertainty: A theoretical vision of uncertainty. *Public Administration Review* 23: 18-45. (1999)

[148] Wilcox K. Design space Exploration. Lecture in: 16.888, *Multidisciplinary System Design Optimization.* Massachusetts Institute of Technology, Cambridge, MA (2012, February 23rd).

[149] Wilson TD, Centerbar DB, Brekke N. Mental contamination and the debiasing problem. Gilovich, Thomas (Ed); Griffin, Dale (Ed); Kahneman, Daniel (Ed). *Heuristics and biases: The psychology of intuitive judgment.* (pp. 185-200). New York, NY, US: Cambridge University Press. (2002)

[150] Wittgenstein L. Tractatus Logico-Philosophicus. (1922) Translation by Bertrand Russel. Published by Cosimo, Inc., 2007.

[151] Wu MS, Ross AM, Rhodes DH. Design for Affordability in Complex Systems and Programs Using Tradespace-Based Affordability Analysis. *12th Conference on Systems Engineering Research*, Redondo Beach, CA, March 2014.

[152] Wu MS. *Design for Affordability in Defense and Aerospace Systems using Tradespace-based Methods*. Dual Master of Science Thesis, Aeronautics and Astronautics and Technology and Policy Program, MIT, June 2014.

[153] Wynne B. Uncertainty and Environmental Learning: Reconceiving Science and Policy in the Preventive Paradigm. *Global Environ. Change* 2, 111–127. (1992)

[154] Zadeh LA. Fuzzy sets as a basis for a theory of possibility. *Fuzzy Sets and Systems* 1 3-28. North-Holland Publishing Company. (1977)

[155] Zimmermann HJ. Uncertainty Modelling and Fuzzy Sets. In Natke HG and Ben-Haim Y (eds.), *Uncertainty Modelling and Fuzzy Sets*, Akademie Verlag, Berlin, Germany. (1996)