MIT ESD

Massachusetts Institute of Technology
Engineering Systems Division

ESD Working Paper Series

Revisiting the Question: Are Systems of Systems just (traditional) Systems or are they a new class of Systems?

Brian Mekdeci
Research Assistant
Systems Engineering Advancement
Research Initiative (SEAri)
Massachusetts Institute of Technology
Email: mekdeci@gmail.com

Donna H. Rhodes
Research Scientist
Systems Engineering Advancement
Research Initiative (SEAri)
Massachusetts Institute of Technology
Email: rhodes@mit.edu

Nirav Shah
Research Assistant
Systems Engineering Advancement
Research Initiative (SEAri)
Massachusetts Institute of Technology
Email: nbshah@mit.edu

Daniel Hastings
Professor
Systems Engineering Advancement
Research Initiative (SEAri)
Massachusetts Institute of Technology
Email: hastings@mit.edu

Adam M. Ross
Research Scientist
Systems Engineering Advancement
Research Initiative (SEAri)
Massachusetts Institute of Technology
Email: adamross@mit.edu

Paper submitted for CESUN 2014 - Fourth International Engineering Systems Symposium to be held June 8-11, 2014 at Stevens Institute of Technology.

ESD-WP-2014-01 March 2014 esd.mit.edu/wps

Revisiting the Question: Are Systems of Systems just (traditional) Systems or are they a new class of Systems?

Brian Mekdeci, Nirav Shah, Adam M. Ross, Donna H. Rhodes and Daniel Hastings
Systems Engineering Advancement Research Initiative (SEAri)
Massachusetts Institute of Technology
77 Massachusetts Avenue, Building E38-572
Cambridge, MA 02139
http://seari.mit.edu

I. Abstract

This paper revisits a question asked and debated widely over the past decade: are Systems of Systems (SoS) just traditional systems or are they a new class of systems? Many have argued that SoS are a new class of systems, but little research has been available to provide evidence of this. In this paper we share highlights of recent research to show SoS not only have a different structure than systems and thus need to be engineered differently, but also may possess different attributes for beyond first use properties (the "illities") such as flexibility and adaptability as compared to systems. By examining historical examples and by using a maritime security SoS as a research test bed, this paper shows that the "ility" called survivability had some design strategies that were directly mapped from systems and also allowed new strategies that only made sense for a SoS (e.g. vigilance). The paper also shows that some design strategies have a different implementation and meaning (e.g. margin) at the level of a system compared to SoS level. We conclude the answer to the question "Are SoS's just systems?" is both yes and no. They are manifestly systems but possess properties not found in traditional systems. This is shown to true of the meta-property of survivability as applied against a directed SoS.

II. Introduction

Systems Engineering has been developing since the development of the intercontinental ballistic missile and the Apollo program[1]. As System Engineering developed past the largely technical systems such as the ICBM and Apollo, it has to incorporate more sociotechnical issues. This was clearly documented by Hughes[1]. Some have argued that this development separates the discipline of traditional system engineering from a new discipline of engineering systems[2]. Another development that became clearer towards the end of the last century was that systems were being combined together into systems of systems. Of course, this seems like a tautology since the definition of a system[3] contains the idea of "a group of devices or artificial objects or an organization forming a network especially for distributing something or serving a common purpose". This definition would suggest that a system of systems is itself a system. While this is obviously true, continued development of large scale systems of systems (such as the Air Traffic Management System; which is a system) as compared to a coffee pot (which is also a system) suggested that there substantial differences between the classes of systems.

A. Differentiating a System of Systems from a Traditional System

In a seminal paper, Maier[4] outlined some of the distinguishing characteristics of SoS. Maier suggested that SoS were composed of systems which were loosely coupled together and where the system elements has the properties of *operational independence* and *managerial independence* from a technical and social perspective. Another way to regard this is to note that in a System of Systems, the individual elements may locally produce value for some stakeholders and also globally produce value for an extended set of stakeholders. The individual elements may at times act independently and may even leave the system of systems. This is manifestly true of a set of aircraft in an Air Transportation System but makes no sense for a coffee pot (the heating element by itself does not produce value). Boardman et al.[5] extended the definitions along which Systems of Systems can be distinguished from systems. Karcanias et al.[6] followed up on Maier's work and emphasized that the multi- agent nature of an SoS namely that agents in an SoS are at least partially autonomous, that each agent may only have a local view of the whole SoS or that the global view is too complex to be practical and that decision and information gathering may be distributed.

The US Department of Defense[7] defines systems of systems as "a set or arrangement of systems that results when independent and useful systems are integrated into a larger system that delivers unique capabilities", although there have been many different definitions of SoSs in the literature[8][9].. The independent and useful systems that make up a SoS are referred to as constituent systems. Not all of the components in a SoS are constituent systems; most systems of systems will still require some traditional

components that are not systems themselves, but are required for the overall SoS to function properly. Some may argue that most "components" within a system, even within a traditional system like a stereo, are actually systems themselves. For example, a CD player can be considered a system because it is made up of components such as a motor, laser, and digital-to-audio-converter (DAC). Components of a traditional system that are composed themselves of subcomponents that interact with each other, are referred to as subsystems. Yet, what distinguishes a system of subsystems, from a system of systems? There are a number of key system of systems properties that have been identified in the literature[4] which distinguish systems of systems from traditional systems. They include:

Operational Independence of the Elements. A system of systems is composed of constituent systems, which can independent and useful in their own right. They can exist and provide some value to stakeholders outside the SoS. In traditional systems, components are not likely to be valuable by themselves or be able to operate outside of the system.

Managerial Independence of the Elements. The constituent systems have the ability to decide their own actions and behavior.

Evolutionary Development. A system of systems goes through evolutionary development where components and constituent systems are added, removed, and modified in response to changes in context.

Emergent Behavior. The US DoD[7] defines emergent behavior as "behavior which is unexpected or cannot be predicted from knowledge of the system's constituent parts", even though it acknowledges that there is no single, universally accepted definition of emergence. Regardless, the reason why system architects construct a system of systems in the first place, is so that the SoS can perform higher-level functions that are not possible by any single constituent system.

Geographic Distribution. The span of the geographic distance between the component systems is so large, that they are usually only exchanging information and not useful quantities of mass and energy.

Connections. Constituent systems tend not to be totally independent or totally dependent when interacting within a SoS. Rather, they are inter-dependent and loosely coupled. In traditional systems, there tends to be tight coupling and strong inter-operations between components.

Multi-Functionality. The constituent systems within a system of systems tend to be able to perform multiple functions and roles within the SoS. Components within a traditional system tend to be uni-functional.

Contextual Diversity. Due to the increased geographical separation, constituent systems tend to have more contextual diversity in a SoS than components in a traditional system.

Unbounded. Systems of systems are often unbounded, meaning that components can be added, modified and removed independently of some central administrative control. Furthermore, the decision to add, remove or modify components is often done by stakeholders with a limited understanding of the entire SoS. Traditional systems tend to be bounded, where decisions about the addition, modification and removal of components are done by a central authority with complete knowledge of the system.

Many authors acknowledge the following types of SoSs.

Directed SoS. A directed SoS exists to meet a purpose given by a central authority. The component agents have their actions dictated by the central authority so as to accomplish a central purpose. An example is a joint military force composed of ships and planes under a single unified command.

Collaborative SoS. A collaborative SoS has no central authority with coercive power over the constituents. The actions of the individual agents are governed both by their own needs as well as the needs of the SoS. SoS objectives are met by collective agreement of the agents to pursue an agenda. An example is the Internet.

Virtual SoS. A Virtual SoS lacks any central authority. The behavior arises from the unplanned, un-coordinated interactions of the agents. An example is intermodal transport of freight via a train or truck network.

Acknowledged SoS[10]. An Acknowledged SoS has a central authority but it does not have coercive power over the agents. The different agents retain their own budgets, decision-making and objectives. An example would be the Army Future Combat System[11] or the DoD/NOASS National Polar Operational Environmental Satellite System (NPOESS)[12].

In this paper, we explore the idea that SoSs not only have a different structure than systems and thus need to be engineered differently[13] but also may possess different attributes for beyond first use properties such as flexibility and adaptability[14] as compared to systems. In Section III we explore some key differences between systems and systems of systems. In Section IV we look at "beyond first use properties" for systems and then for SoSs. In Section V, though a simulation of a maritime security SoS and case studies, we show that for survivability, the strategies used for SoSs and traditional systems can be different, and these differences are important. We speculate that this may be true of many of the illities. Thus the answer to the question "Are SoS's just systems?" is both yes and no. They are manifestly systems but possess properties not found in traditional systems.

III. SOME KEY DIFFERENCES BETWEEN SYSTEMS AND SYSTEMS OF SYSTEMS

A. Lifecycle Change

Systems of systems often experience both exogenous and endogenous change. Often, decision makers implement endogenous changes to mitigate the harmful effects of a contextual change, or take advantage of new opportunities. SoS go through various changes, some of which can be made immediately, some of which can be made only after an analysis is performed, while others require a significant re-architecting effort that results in an "evolution" in the system architecture.

The likelihood for change is higher for SoS that have long lifetimes, operate in dynamic contexts, or have complex behavior. Endogenous change, in either the number and type of components, or their attributes and capabilities, is particularly an issue with systems of systems, since they go through evolutionary development as components are added, removed and changed over time[4]. Even if the components themselves do not change, what they do within the overall system might change. For example, certain F-16s, acting as components of a larger military SoS, may change from air-to-air combat roles to air-to-surface attack roles.

B. Value Delivery to Stakeholders

Engineered systems are designed to provide value, which is the net benefit or utility (i.e. received benefits less costs for receiving those benefits) (Richards, Ross, Hastings, & Rhodes, 2008) an engineered system provides to its stakeholders. In traditional systems, the boundaries, as well as stakeholders, are often well-defined. Indeed when students are taught system engineering, one of the first things they are taught is to define the system boundary[15]. By contrast, the agents in a SoS may belong to multiple SoSs and thus the concept of the SoS boundary, as well as overall value delivery is less clear. To illustrate this concept, consider a maritime security SoS (this will be discussed in more detail later) that includes an Unmanned Aerial Vehicle (UAV), a Ground Control Station (GCS), and a satellite that links the two (). The UAV and GCS are owned and operated by one set of stakeholders (Γ_M), who lease the services of the satellite from a satellite communications provider (Γ_S). The satellite is part of the SoS that includes the UAV and the GCS (SoS M), but the satellite stakeholders also lease bandwidth to cell phone companies (Γ_C). Thus, the satellite is also part of cellular phone SoS as well (SoS C).

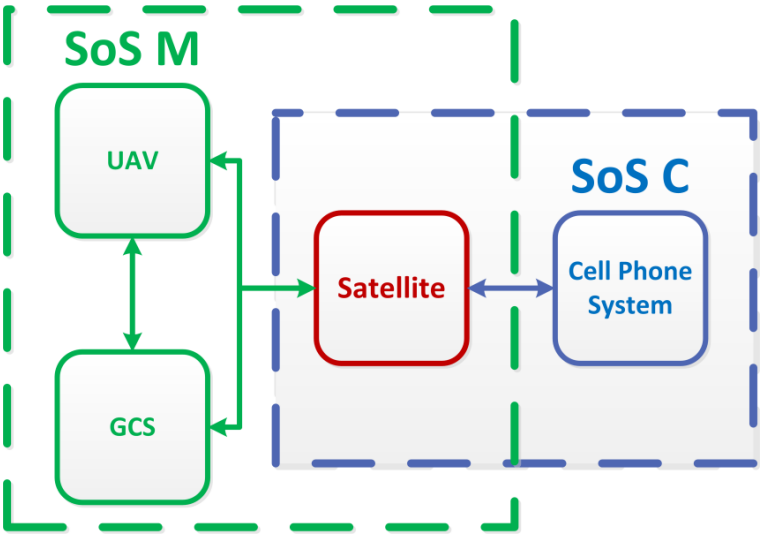

Figure 1: Example of a System Belonging to Multiple Systems of Systems

Let $V_{S|\Gamma_S}$ be the value that the satellite delivers to its stakeholders, the satellite communications provider (Γ_S). This value is different, but related to the value that the satellite provides to the cell phone operators ($V_{S|\Gamma_C}$) and the value that the satellite provides to the maritime security stakeholders ($V_{S|\Gamma_M}$). The viability of the satellite to its stakeholders is determined by whether

or not $V_{S|\Gamma_S}$ exceeds some minimum threshold set by its stakeholders ($T_{S|\Gamma_S}$). $V_{S|\Gamma_S}$ will be some combination of the value the satellite brings to its stakeholders by participating in both the maritime security SoS ($V_{M|\Gamma_S}$) and the cellular SoS ($V_{C|\Gamma_S}$). Suppose the government requires more bandwidth for the maritime security SoS and give an appropriate increase in compensation. This means less bandwidth for the cell phone companies, reducing the value of the satellite in their SoS ($V_{C|\Gamma_S}$), even though the value of the satellite increases for the satellite and maritime security stakeholders.

C. Influences

Another difference between SoS and traditional systems is the nature of the influence between the component agents. Systems can exchange mass, energy, momentum and information between their component parts[2]. However since some kinds of SoS allow operational and managerial independence of the component agents, there is also the possibility of social or economic influences between the constituents.

A variety of frameworks have been proposed to describe the structure, operation and management of an SoS[16].Of particular importance is that each constituent is trying to satisfy a locally specified value proposition, i.e., they are free to make decision that ensure their local needs are met. The extent to which these decision support a broader SoS agenda depends upon the alignment of these local needs with the SoS goals mediated by whatever influences that the SoS authority brings to bear upon the constituents. As described by Bjelkemyr:

"Each system within a SoS is a self-interested node in a network. These system nodes try to maximize their own utility under the influences of and in competition with the other nodes. The global SoS behavior thus emerges as a result of the actions at the lower levels of the SoS, down to the system element level." [16].

One can observe this challenge in real world SoSs. For example, peering disputes among the Internet service providers is an issue of choosing with which other systems one wishes to connect, i.e., with whom to collaborate. As an example, in October of 2005, Level 3 communications a Boston based Tier 1 Internet service provider decided to terminate its peering agreement with Cogent communications, another Tier 1 provider. By refusing to peer with Cogent, Level 3 cut-off direct traffic flow between their respective networks. This forced routing via third-party network increasing congestion on those links. For some customers whose only connection was via Level 3, they were disconnected from those hosts whose only connection was via a Cogent network. The same was true in the other direction. After a few days, cooler heads prevailed and the peered connection was reestablished[17]. The underlying cause of the dispute was an imbalance in traffic flow between the two networks. Level 3 felt that Cogent was in violation of their contract when Cogent tried to make inroads into Level 3's market of selling access to Tier 2 providers. If a given Tier 2 provider, directly connected to Cogent instead of going through Level 3, this might create a traffic imbalance to Cogent's benefit.

The essential difference between the decision structure in traditional system engineering versus SoS engineering is one of alignment. The SoS architect may need to influence the constituent decision makers to behave in a manner that is not necessarily locally optimal for them but does serve the interest of the SoS. This relationship between the SoS architect and the constituent decision makers is a principal-agent problem[18]. In the SoS case, the principal is the central authority/SoS architect who wishes to effect some SoS behavior that they value via the actions of the agents, i.e., constituents. Given this framing, the central authority is referred to as a SoS principal. Note that constituents may be interacting with multiple such authorities at a given time (e.g. if they are participating in multiple SoS) and may also act as such an authority themselves with respect to other constituents such as in a collaborative SoS.

D. Systems Affecting Other Systems within a SoS

One of the largest problems in a SoS is that the environment in which a constituent system interacts often forms part of the context of one or more other constituent systems. That is, the outputs of a constituent system, whether voluntary or involuntary, often become the inputs of another system within the SoS. This is illustrated in Figure 3, and was the case with the North American power grid, which suffered a large failure in 2003 as a result of systems negatively impacting other systems within a SoS.

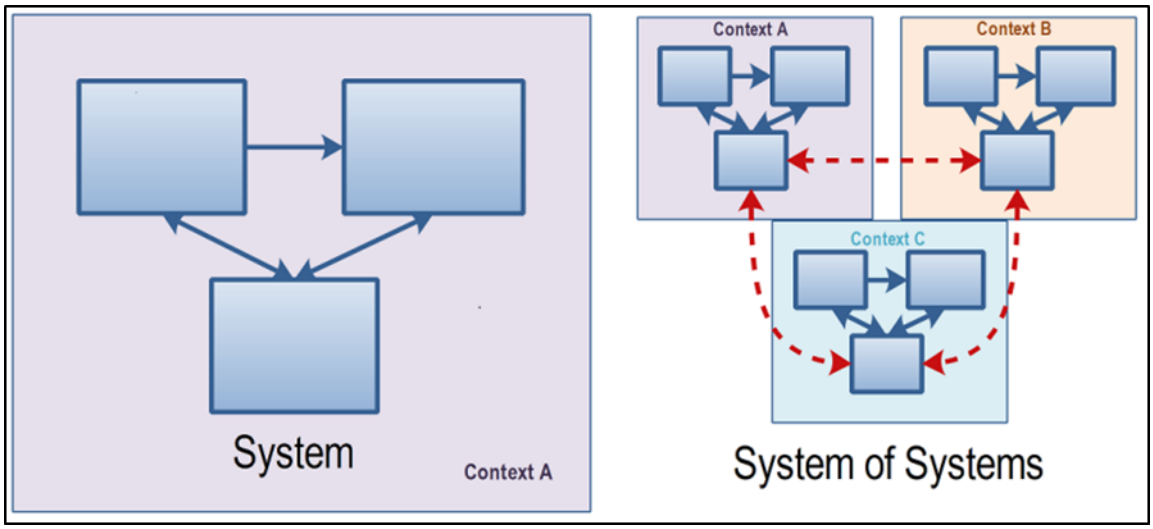

Figure 2: Contextual diversity for a System of Systems

1) SoS Case Study: North American Power Grid

Many experts believe that the North American power grid is one of the largest and most complex systems of the technological age, with approximately 15,000 nodes (power stations), and 20,000 transmission lines (edges) [23] (). The primary purpose of the power grid is to ensure that power is safely and effectively transferred from generators to end users, in a complex network of interconnected nodes that span a large geographical area and include many different independent power operators. The North American power grid is divided in the Eastern, Western and Texas regions, which are tightly coupled within each region, and loosely coupled between.

On August 14, 2003 at approximately 4:10pm EDT, the second largest blackout ever affected large parts of the Northeastern United States and parts of Canada. By the end of the day, more than 508 generating units at 264 power plants shut down, after a 3.5 GW power surge affected the transmission grid. The blackout lasted between 7 and 16 hours, and resulted in a 20% increase in fatalities during that period. Although many critical systems had backup generators, some critical systems failed because their generators failed or they ran out of fuel, including certain telephone services, water supplies, gas stations, border crossings, hospitals and television/radio stations

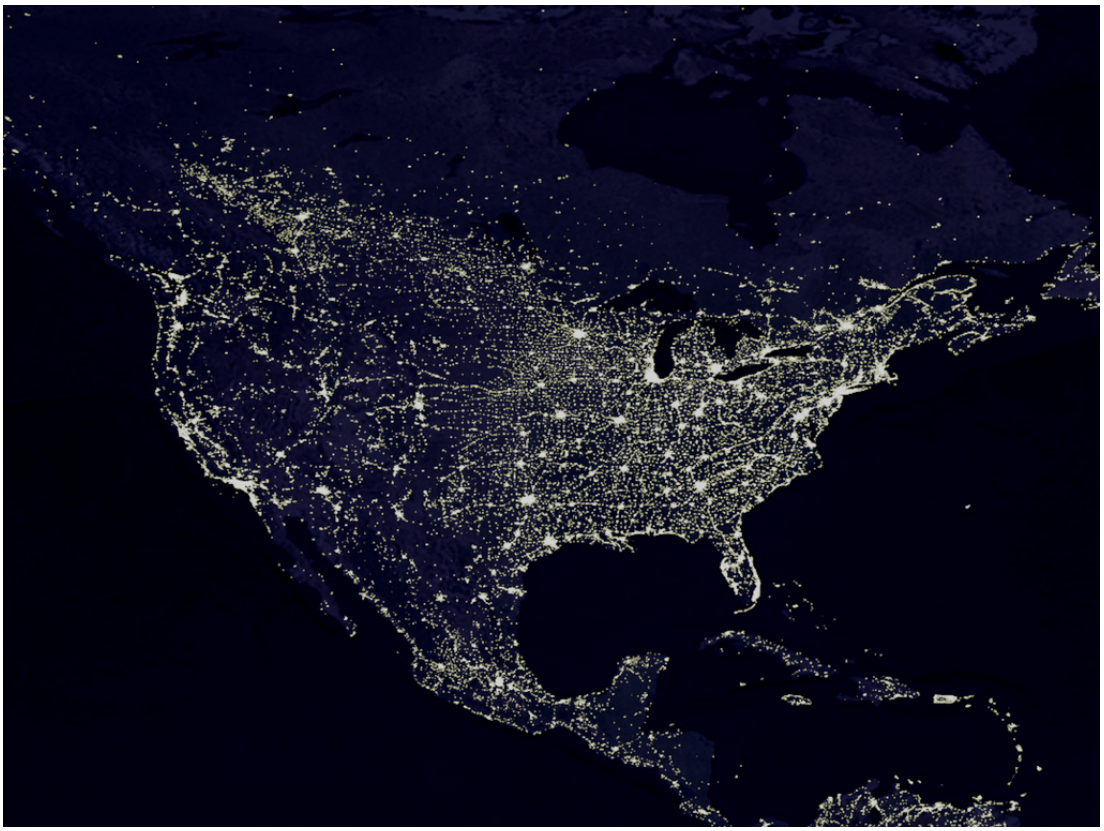

Figure 3: The Power Grid Lights Up North America as Viewed From Space

Disconnections affect the nodes that are no longer connected to the network in obvious ways. However, in networks such as the North American power grid, disconnections also impact nearby nodes that are still connected to the network. Since large amounts of electrical power are not easily stored, the power grid must carefully balance the demand of electricity to its supply. In power grids, failure of an edge transfers the load being carried by the edge to other edges in the network. If those other edges have the additional capacity to carry the load (i.e. margin), then there is not a problem. However, there are limits to how much load an edge can carry and if those limits are approached, then the new edge may fail itself either involuntarily, or by being shut down to prevent damage. When this new edge subsequently fails, then other edges must take both its load as well as the load of the original failing edge as well. This can cause a chain reaction, where failures lead to further failures that propagate throughout the system. Cascading failures are a special type of chain reaction where the small failures eventually lead to major events, e.g. a nuclear power operator spilling coffee on a command console causing a chain reaction that ends in a Chernobyl-level accident. Preventing chain reactions and in particular, cascading failures, should be a priority for system architects. illustrates how chain reaction of small events can grew into the cascading failure that resulted in the US Northeast Blackout of 2003. The initial incident was an untrimmed tree in Ohio that severed a power line. This link failure burdened other lines with an extra load, causing some of them to fail as well. Each failing power line placed additional load on the remaining lines until they succumbed as well, leading to a chain reaction of power line failures. Eventually, enough power line failures caused entire power plant generators to shut down as a precaution, in a cascading failure that eventually lead to a complete grid failure covering the Northeastern United States and some parts of Canada. There were multiple incidents and circumstances along the way that contributed to this unfortunate outcome. One such perturbation was the failure to recognize and mitigate the failing power lines in Ohio, due to a software bug affecting the control rooms of one of the Ohio power plants[24]. Cascading failures are system failures that result from component failures that cause other components to fail, which cause further components to fail, and so forth. Cascading failures are a major concern for connected networks like a power grid, but also occur in other systems as well.

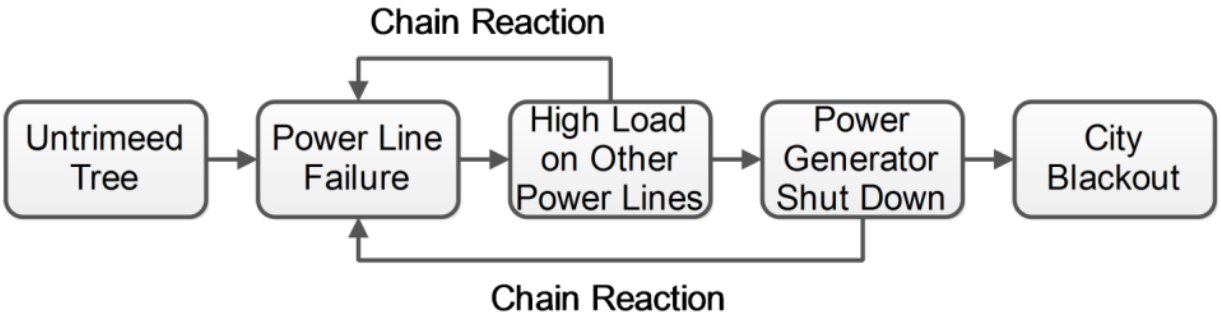

Figure 4: Chain Reaction Leading to the Northeast Blackout of 2003

IV. BEYOND FIRST USE PROPERTIES

As engineering systems have developed, attention has turned to what have been called "beyond first use properties"[2]. Since many of these systems last a long time and through possible changes in context, it is important to understand properties such as flexibility, adaptability, survivability, evolvability of the system design as much as (first) performance, cost and schedule[14]. These have been called the "ilities" and appropriate attention to them can make for fundamental changes in the initial design of the system[20]. Recent work has looked at extensively at one at the "ilities" namely survivability[21] and shown how the ideas and metrics behind this "ility" can be embedded in the initial tradespace analysis for a system even across multiple eras and epochs. Previous work has shown how the same kind of analysis can be done on embedding flexibility in the design of a system. In this paper we shall argue that for Systems of Systems, the "ilities" may take on additional meanings that have no analog for systems.

Flexibility and the other 'ilities" have been defined rigorously for systems[22]. Table 1 has a set of definitions:

"Ility"	Description
Adaptability	Ability to be changed by a system-internal change agent with intent
Agility	Ability to change in a timely fashion
Changeability	Ability to alter its operations or form, and consequently possibly its function, at an acceptable level of resources
Evolvability	Ability to be inherited and changed across generations (over time)
Extensibility	Ability to accommodate new features after design
Flexibility	Ability to be changed by a system-external change agent with intent
Interoperability	Ability to effectively interact with other systems
Modifiability	Ability to change the current set of specified system parameters
Modularity	The degree to which a system is composed of modules (not an ability-type ility)
Reconfigurability	Ability to change its component arrangement and links reversibly
Robustness	Ability to maintain its level and/or set of specified parameters in the context of changing system external and internal forces
Scalability	Ability to change the current level of a specified system parameter
Survivability	Ability to minimize the impact of a finite duration disturbance on value delivery
Value Robustness	Ability to maintain value delivery in spite of changes in needs or context
Versatility	Ability to satisfy diverse needs for the system without having to change form (measure of latent value)

Table 1: "-Ilities" Relating to Change

As systems, SoS may have similar "ility" properties. However, there may be additional ones that arise out of some of the distinguishing characteristics of systems of systems highlighted in Section II.

A. Survivability Strategies for a System

We will illustrate the possibility of additional dimensions to the "ility" properties by contrasting the options available for a system to be survivable and for a SoS to be survivable. Richards defined survivability for a system as the *ability of a system to minimize the impact of finite-duration disturbances on value delivery* through (I) the reduction of the likelihood or magnitude of a disturbance, (II) the satisfaction of a minimally acceptable level of value delivery during and after a disturbance, and/or (III) a timely recovery. Following Ross[14], he defined an epoch as a time period with a fixed context; characterized by static constraints, design concepts, available technologies, and articulated attributes. Then the value delivery for a survivable system can be plotted as

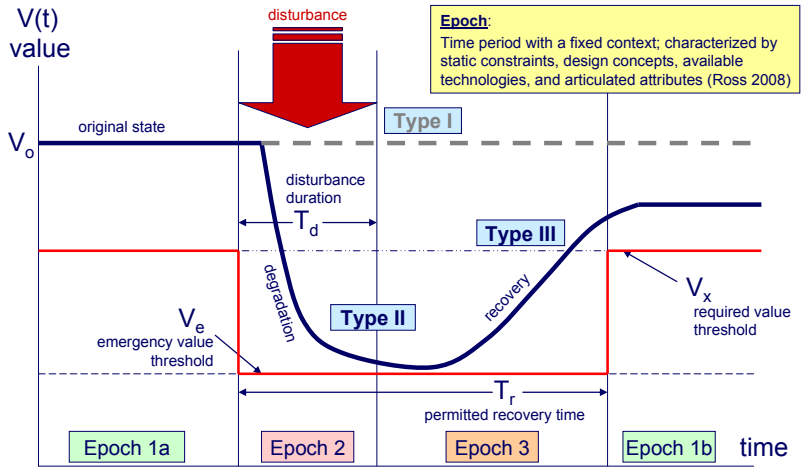

Figure 5: The Three Types of Survivability (Richards, 2009)

On the basis of a review of the empirical principles used in four systems that are known to be survivable (Blackhawk helicopter, A-10 "Warthog" attack aircraft, F-16C Falcon and the Iridium global cellular satellite constellation), he was able to derive seventeen survivability design principles. These are shown in Table 2.

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

Table 2: Survivaiblity Strategies for Systems (Richards, 2009)

When these design principles are placed on the value delivery for a survivable system, the following diagram results (Figure 6)

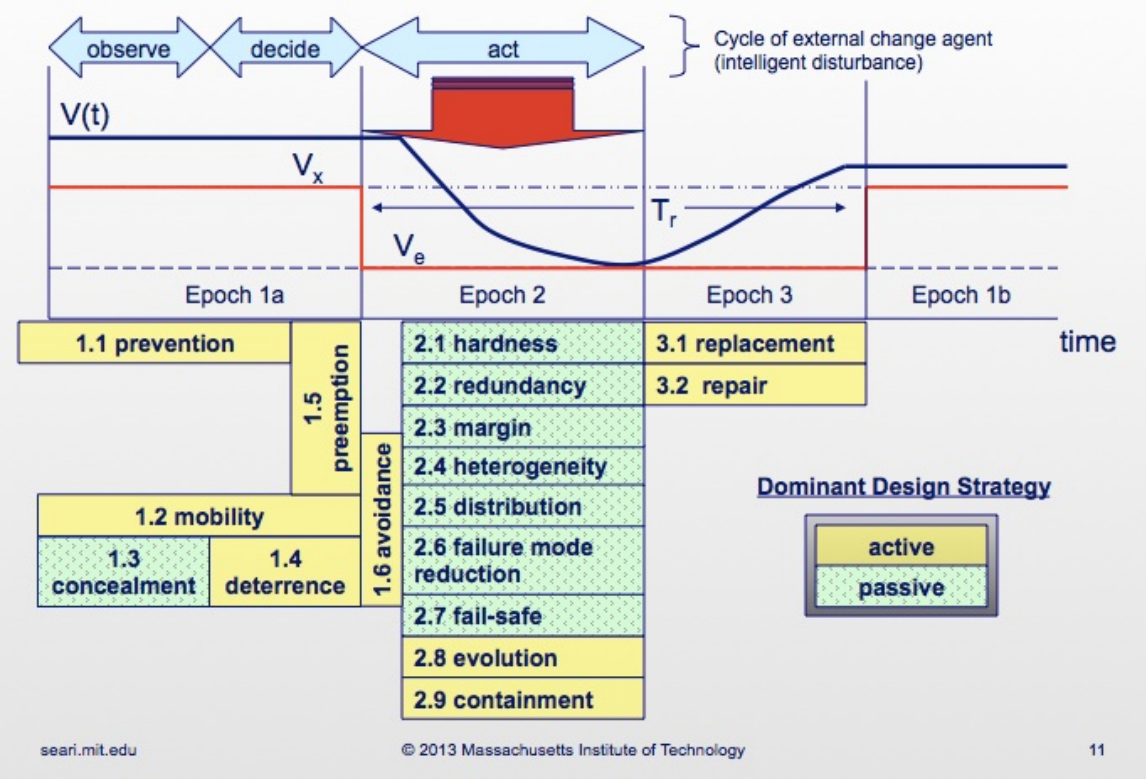

Figure 6: Applying Survivability Strategies Before, During and After Involuntary Epoch Changes

This shows the set of strategies that a survivable system may use. As long as these strategies have meaning they can be applied to SoSs. For example, the strategy of margin can be applied to a system (have more capability than necessary) as well as to a SoS (both at the level of the component systems and at the level of the whole architecture). This will be shown below in the maritime security SoS simulation.

V. System of Systems Strategies

For a SoS, the question is if there are additional strategies that enable survivability that would have no meaning for a system or possibly a different meaning than for a system? To answer that question, in this paper we examine one of the largest SoS failures ever, the Northeast Blackout of 2003 as well as develop and exercise a discrete event simulation of a SoS.

A. A Maritime Security System of Systems

In order to explore the question of SoS survivability, a discrete event simulation was developed of a maritime security SoS. Maritime security SoS (MarSoS) are relatively easy to model and possess many of the distinguishing characteristics of systems of systems. In the language defined previous, a MarSoS is a directed SoS.

1) Objectives

Not all maritime security SoSs are the same, but they typically share the goal of ensuring that ships entering a particular body of water are complying with appropriate rules and regulations. At a minimum, a MarSoS is responsible for maintaining a certain degree of situational awareness over a particular body of water, known as the Area of Interest (AOI). Usually, the MarSoS is required to know the location, identity and heading information of any ship within the Area of Interest (AOI). In many cases, the MarSoS is responsible for enforcing the laws and regulations governing the AOI as well, and this typically means tracking and intercepting any possible ships that may be violating those rules.

2) Components

The primary effectors of the MarSoS are manned and unmanned vehicles. All of the unmanned vehicles are aerial (i.e. UAVs) while the manned vehicles include both aerial and naval. These vehicles are not merely components of a larger system, but systems in their own right. In addition to the vehicles, there are supporting systems, such as command center, radar towers and ground control stations. The selection of components that constitute the MarSoS is variable, and one of the ways in which strategies can be tested and induced, but design options for the MarSoS in the quantitative model included the following:

Operational UAVs (OUAV). UAVs used for operational functions, such as detection of targets.

Tactical UAVs (TUAV). UAVs used for identification and observation of targets.

Manned Patrol Boats (MPB). Manned patrol boats used for detection, identification, observation and interception of targets.

Manned Helicopters (MHel). Manned helicopters used for identification and observation of targets.

Manned Patrol Aircraft (MPA). Manned patrol aircraft used for identification and observation of targets.

Command Center. Center used for coordination of tasks between the vehicles of the MarSoS.

Airbases. The location where manned patrol aircraft reside when not in use as well as the location for the ground control stations controlling the UAVs. The distance between the UAVs and the ground control station impacts the quality of the communications.

Docks. The location where manned patrol boats reside when not in use.

3) System of Systems Characteristics of a MarSoS

A MarSoS exhibits some of the key properties that distinguish systems of systems from traditional systems[25], including the following:

Component Independence. The vehicles themselves are capable of operating independently and providing value to their own stakeholders outside of the SoS. For example, a UAV that is part of the SoS, can be used outside of the SoS, for wildlife tracking or to fight forest fires, for example.

Distributed Authority. Managerial independence requires that the constituent systems have the ability to make decisions on their own, or through collaboration with other constituent systems. A MarSoS can be configured such that decision making, such as task assignment, is done via a central authority, or done collaboratively by the vehicles themselves.

Geographical Separation. Maritime security systems of systems are typically made up of several vehicles, along with supporting components (e.g. radar towers, ground control stations, etc.). Unlike traditional, monolithic systems, most of the components within a MarSoS are not co-located, but rather are distributed around a relatively large AOI.

Multi-Functionality. A simple, traditional system is more likely to have a single function or purpose, whereas a system of system is more likely to be multi-functional. In a MarSoS, there are several tasks that need to be performed, i.e. detect, identify, and if necessary, intercept targets.

Increased Contextual Diversity. Since components in systems of systems are more likely to be physically separated than those in traditional systems, it follows that they will be more likely to be operating under different environmental conditions. While a UAV in one part of the AOI may be experiencing a relatively high load of including ships, other UAVs in other parts of the AOI may be idle. Certain vehicles in the MarSoS, for example, may be operating on water, others on land, while others are in the air.

Decreased System Awareness. Since components in the MarSoS are often operating under different contexts, they must share the contextual information with each other in a timely manner, for all of the components to have the same system awareness at any given time. In order for that to happen, three things must occur; (1) the important differences in context must be apparent, (2) stakeholders must be willing to share this information (not always the case, particularly if the contextual differences are the stakeholder preferences and policies), and (3) mechanisms must exist for this information to be shared in a timely manner. For these reasons, components within systems of systems that operate under different contexts may be operating under incorrect or incomplete information about the system itself, than components within traditional systems operating under the same context.

Evolutionary Development. Traditional systems are typically assembled during implementation, before the system is operated. Since components of systems of systems are often added or removed during the operation of the SoS, what the SoS is composed of may be continually evolving. Such is the case with a MarSoS, which may change its configuration at any time based on evolving stakeholder needs and contextual considerations. An example of a change might be to replace larger UAVs coordinated by a central authority with swarms of smaller UAVs that coordinate their tasks collectively.

Abstruse Emergence. In traditional systems, emergent behavior is often part of the design (or at least expected), and as such, is usually a benefit overall. In systems of systems, particularly those with evolutionary development, emergent behaviors are more difficult to predict and often end up being problematic. Such is the case with a MarSoS, particularly when disruptions occur and there is a disconnect in the information sharing between the various components. This may lead to one component acting and re-acting based on a different set of information than another, leading to emergent behavior that is difficult to predict.

4) Dynamic Context

For this quantitative model, the area of interest is a littoral environment, consisting of narrow water passageway located between two landmasses. Typically, all ships above a certain size are required to have international transponders that pass the relevant identity, cargo and destination information to local port authorities, although additional confirmation by the MarSoS is sometimes required. The MarSoS normally has to identify smaller ships that typically do not have transponders. Methods of identification include marine radar, visual inspection and/or direct communication. Although there are many different types of ships that pass through this AOI, they can be divided into three broad categories; (1) compliant ships, (2) smugglers, (3) terrorists. Compliant ships are a general category for ships that are complying with the appropriate laws and regulations, and are not hostile towards the MarSoS. Compliant ships include civilian sailboats, cargo ships, cruise ships, fishing vessels, police boats and friendly military vessels. Smugglers are non-compliant ships carrying contraband. A vessel of any size may contain contraband, but larger vessels have more capacity for smuggling. However, cargo size is not an indicator of cargo importance, as smugglers may hide some of the most dangerous contraband, such as weapons-grade uranium, aboard smaller vessels. Terrorists are non-compliant ships that wish to detonate high-impact explosives at the port. For the quantitative model, terrorists have a Modus Operandi (MO) based on a previous study on a low-probability, high-impact terrorist attack on a maritime port conducted by the Naval Postgraduate School[26] in which terrorists appear to be compliant civilian vessels until they are close to the port, at which point they head at full speed and detonate a bomb.

5) Model description

Researchers to give insight into various designs and contexts when it is impossible or impractical to use real systems and environments use models. Although the real systems and environments may be very complex, all models make some assumptions that may simplify the underlying relationships enough so that they may tractable. If simple, parametric models can be made that describe the real phenomenon with enough fidelity that they are useful to decision makers, then it is often possible to use numerical methods and computing solutions to explore a full tradespace of millions of designs and contexts. However, if it is necessary to capture emergent behavior that is not easily simplified, as in the case with many systems of systems particularly those that involve socio-technical aspects, then simulation is often necessary.

a) Simulation of Complex Systems

In choosing the type of simulation to be used, several considerations were made. First, since time is a critical property of a MarSoS, dynamic simulations are required. This means static simulations, such as traditional Monte Carlo methods, are inappropriate. The next consideration to make is whether to use discrete or continuous models. Continuous models are more appropriate for modeling continuous phenomenon, such as electricity flow through a power grid or climate change caused by CO2 emissions. However, endogenous and exogenous changes in the MarSoS are typically discrete. A ship randomly arriving in the AOI, a UAV being shot down, or a smuggler being identified as hostile entity are all examples of discrete-events that occur during the operation of a MarSoS. It is at these discrete-events that the state of the system and/or context changes, the effects of which must be determined through simulation. The last consideration to be made is whether or not the simulation is deterministic or stochastic. As with many complex systems of systems, there is a large amount of uncertainty both in context and within the system. Inputs to the system are often random in both time and space. The arrival of boats in the AOI, for example, can be modeled as a Poisson process with a mean time between boat arrivals of $1/\lambda$, and a random location uniformly distributed along the border of the AOI. Endogenous events can be stochastic as well; such as the exact time it takes a human operator to identify a ship once he has a visual from a particular UAV. Since there is uncertainty in both exogenous and endogenous events, a stochastic simulation, particularly one that models the extreme conditions, is necessary particularly if survivability and robustness of the MarSoS is a concern.

Discrete-event simulation is a popular technique that has been used to model large-scale systems of systems like maritime security systems of systems. Discrete-event simulations are particularly well suited to models that have stochastic, dynamic and discrete-event properties, because they simulate discrete-events in a step-by-step manner, updating the state of the system and its context at specified time intervals, and allow random variables as inputs.

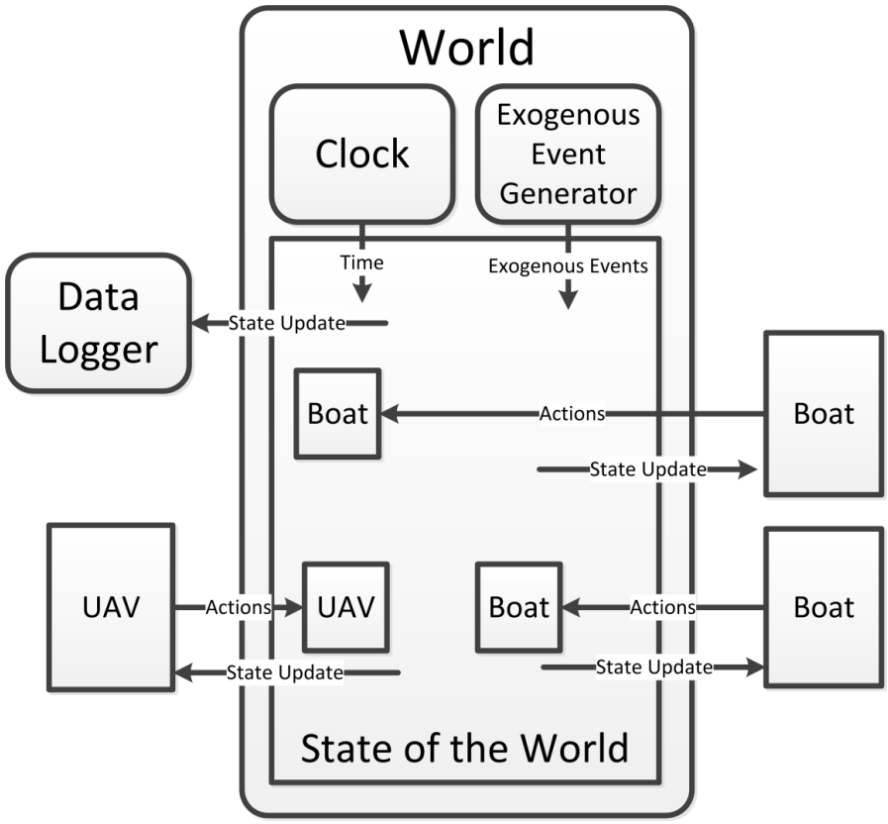

Figure 7 Information Flow through the DES

b) Agent Based Modeling

Systems of systems have many properties that make assessing value delivery non-trivial. Unlike many simple systems, equations alone do not adequately describe the relationship between components. Constituent systems have managerial and operational independence, which makes modeling of individual behavior and decision making necessary. Action and reaction to environmental changes is not simply a matter of physics. Instead, constituent systems perform actions based on the information they have at the time and their own priorities. If humans are operating some of the constituent systems, then there is typically going to be a stochastic behavioral element to be considered as well. When the individual behaviors of the constituent systems interact with each other, the behavior of the overall SoS may not function as expected.

To assess the performance of a MarSoS, an agent-based DES was implemented where individual constituent systems make tactical decisions based on their own behavior characteristics and the local information they have at some particular time. This way, the individual actions and reactions can be ascertained and overall behavior of the SoS can be assessed.

c) Limitations of Agent-Based DES

As with any modeling or simulation technique, agent-based, discrete-event simulations do have limitations and drawbacks. Almost all mathematical models require simplifying assumptions, which may affect the validity of the results, and agent-based discrete-event simulations are no exception. Also, as with any model, the quality of the outputs is strongly correlated with the quality of the inputs, e.g. if there is an error in the assumption used to determine an input such as the arrival rate of ships into the AOI, then there may be a significant error in one or more outputs, such as the probability of detection.

Due to the stochastic nature of the inputs and models, the outputs of the agent-based DES are typically stochastic themselves. This means that for a given set of inputs, the simulation must be run multiple times (e.g. "trials") to gather enough samples to estimate the true attributes of the system. Depending on the level of statistical confidence required, the number of trials required could significantly increase the computational load. This is a major concern, since perhaps the largest drawback to agent-based, discrete-event simulations is the considerable computational effort required to calculate the performance of even one design under once context. Determining the potential actions and reactions of every entity, for each small increment of time, is often cumbersome even for modern computers. If the behavior of each entity depends on the behavior of all the other entities in the

system, then in many cases the behavior f(n) for the entities in the system is O(n!) meaning that the computation effort required grows in a factorial relationship with the size of the SoS and context. This can become a very serious problem if a large tradespace consisting of thousands of designs and potential contexts is to be explored, and multiple runs have to be made for each design/context pair for statistical confidence.

B. Applying the Strategy of Margin to the MarSoS

Certain system viability strategies can be applied to systems of systems either at the system-level (i.e. bottom-up) or at the SoS-level (i.e. top-down). To illustrate the difference, consider a MarSoS where UAVs are performing identification on targets that enter the AOI. The identification process can be approximated as a queue, where the targets that need to be identified arrive according to a random process specified by a mean arrival rate λ. There are k UAVs in the SoS, and each UAV identifies a target according to a random process with a mean service time of μ. As long as $k / \mu \gg \lambda$, then all the targets should get identified before they cross the AOI. Roughly speaking, k / μ represents the capacity of the system to identify targets, and λ represent the demand. Suppose there is a perturbation where a sudden, large influx of targets arrives in the AOI according to some new arrival rate λ'. For this disturbance, $k / \mu < \lambda'$ meaning that all the UAVs will likely be busy for duration of the disturbance and some targets will leave the AOI unidentified as a result. To make the SoS viable in this context, the principle of margin can be applied to either the UAVs or the SoS itself. For example, if the payloads on the UAVs were improved such that the number of targets identified per unit time (i.e. μ) increased to meet the required capacity, then the SoS would be viable in case that perturbation arose. This is an example of applying the principle of margin at the system-level, by increasing the technology level from low to high, and reaping the benefits at the SoS level. To apply the principle of margin at the SoS level, a system architect could simply increase the number of UAVs in the AOI (i.e. increase k) until there was enough capacity to satisfy the demand created by the perturbation.

The MarSoS DES can be used to demonstrate and assess how the strategy of margin can be applied at both the system and SoS-level. Suppose that the stakeholders agree that in order for a MarSoS SoS to be viable, it must able to identify at least 90% of the ships that enter the AOI, on average. A system is proposed, that has 14 vehicle assets (i.e. eight UAVs and four manned patrol boats) and the MarSoS DES simulation is run under normal contextual conditions. Since the context and system behavior is highly stochastic, the output of any particular trial is a random variable and therefore multiple trials must be run to estimate the true mean of the probability of ID. Since the simulation runs fairly quickly, a total of 10,000 runs are executed and the results are shown in .

The mean of the probability of identification (ID) was estimated to be 0.941 with a very narrow confidence interval, due to the large sample size. This system is viable, however it is important to note that there are certain times when the system may not be able to meet its viability criteria, due to some of the random effects of both the context and the system itself.

Now suppose that the traffic in the AOI is doubled, perhaps due to a temporary disturbance such as an evacuation of a nearby country due to an earthquake, or a conflict that causes major shipping lanes to be re-routed through the AOI. To see if the SoS remains viable, the MarSoS DES was run 10,000 times with the same SoS design but twice the ship traffic. As shown in , the mean probability of ID for the SoS under high traffic conditions was found to be 0.867, which is below stakeholder thresholds. Again, due to the stochastic nature of the inputs and system response, there are certain cases where the SoS meets or exceeds the threshold as shown by the right-tail of the distribution, but on average, the system does not met the requirements and would therefore not be considered viable.

To be able to handle disturbances where the traffic through the AOI is doubled, the SoS can implement the principle of margin at either the system or SoS-level to increase the throughput of the system. At the system level, the designers can use better technology (e.g. more advanced payloads) that reduce the amount of time it takes to identify a target by ½. At the SoS-level, designers can simply double the number of vehicles in the SoS. To assess the impact of both of these changes, the MarSoS SoS was run again 10,000 times for each strategy under the heavy traffic conditions. The system-level implementation of the margin strategy (i.e. increasing the technology level of the payloads) exceeded the performance threshold under heavy conditions (probability of ID = 0.960, see) and even surpassed the performance of the original system under regular traffic conditions. However, the probability of ID for the SoS that had double the number of vehicles (i.e. the SoS-level implementation of the margin strategy) only reached 0.885 (see) and failed to meet the viability threshold, although it did improve performance over the baseline SoS.

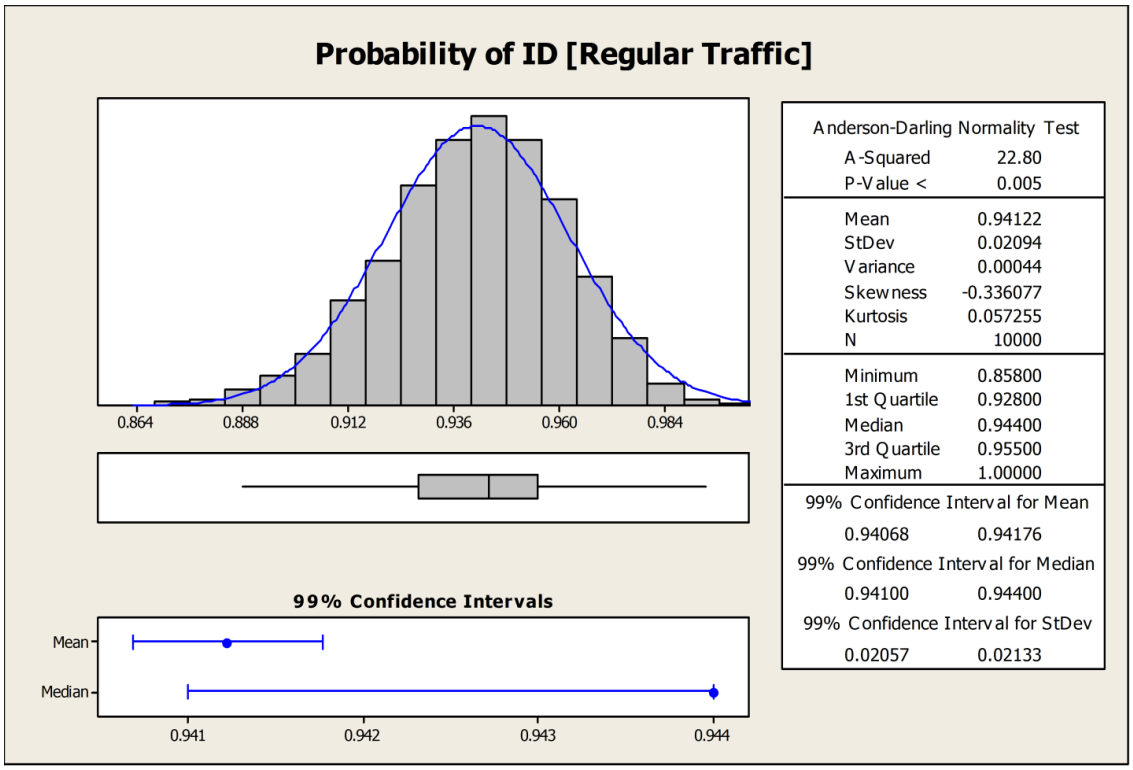

Figure 8: Probability of Identification with Baseline SoS and regular traffic

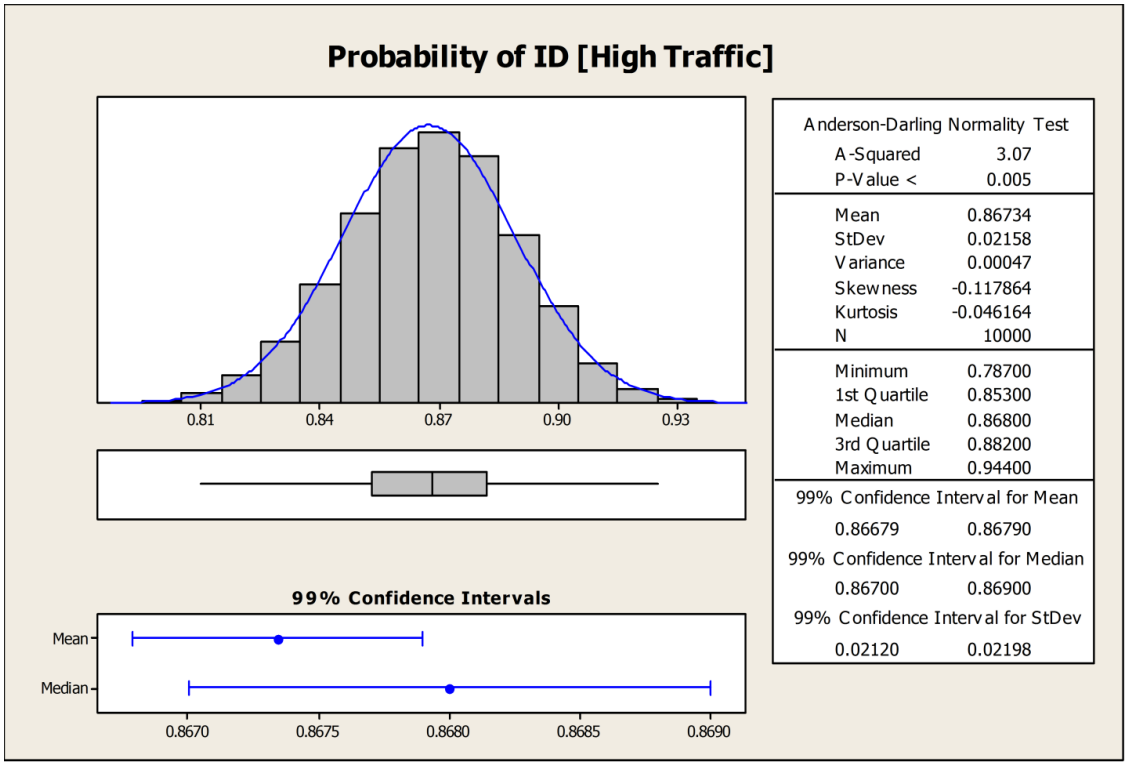

Figure 9: Probability of identification with baseline SoS experiencing high traffic

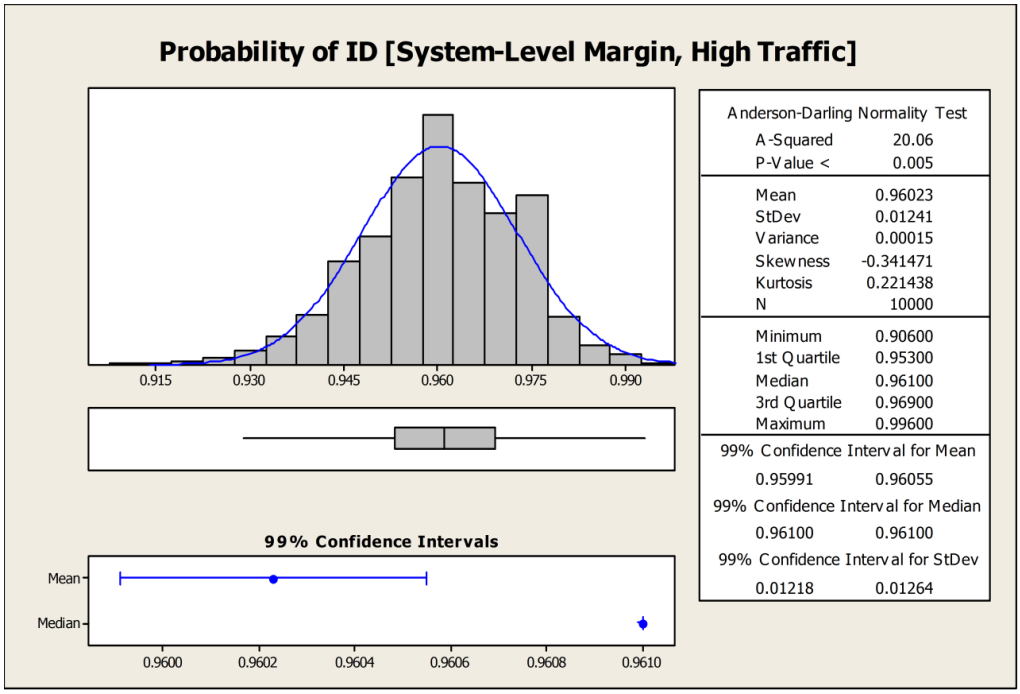

Figure 10 Probability of Identification with System Level Margin and High Traffic

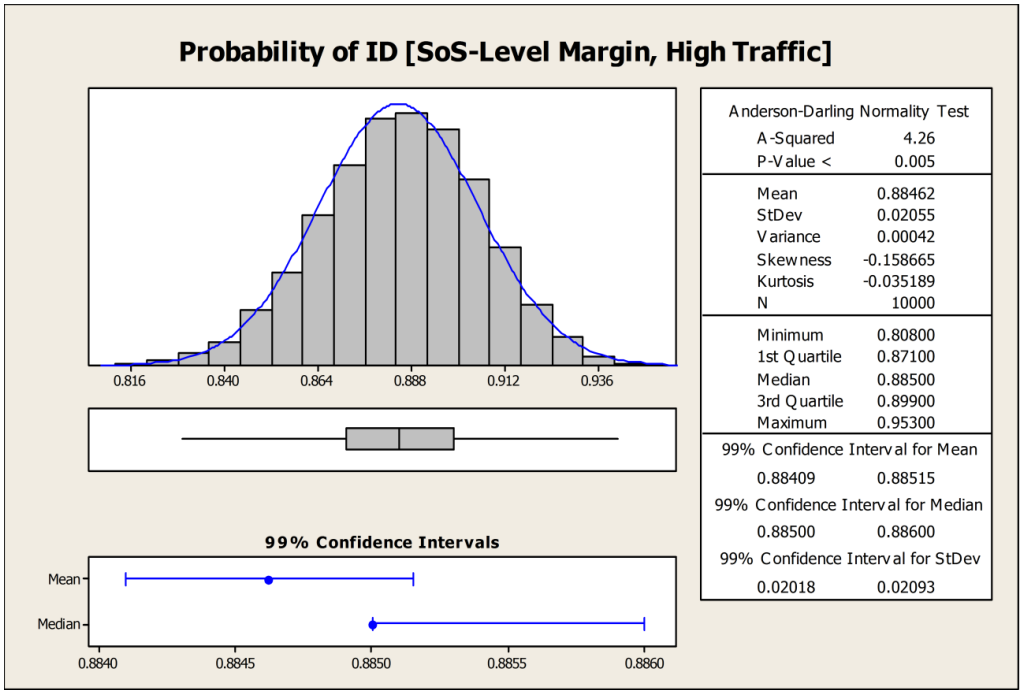

Figure 11 Probability of Identification with SoS-level margin and high traffic

The reason for the discrepancy between the system and SoS-level implementation of the margin strategy was due to the fact that there was a single command center that delegated tasks to the vehicles, and that command center operated similar to a single server queue. As more vehicles were added, the command center did not have the resources to be able to effectively communicate with the additional vehicles and communication queues formed. These queues caused the vehicles to wait longer for instructions and as a result, reduced their ability to perform their tasks in a timely manner. Thus, the additional capacity that should have been realized by having additional vehicles was negated by the additional communication overhead that did not scale accordingly. This example highlights the difficulties of making change in one area, without really understanding how the changes will impact the overall operation of the rest of the system (or SoS).

C. Other SoS Survivability Strategies

1) Redundancy

Margin is not the only survivability strategy that can be implemented at the system or SoS-level and it is not the only strategy that can have varying effectiveness a result. For example, the strategy of redundancy can be applied at the SoS level by having multiple UAVs that perform the same task. This strategy would reduce susceptibility to perturbations that reduce capacity of the system by eliminating capability, such as when components fail, or are incapacitated by hostile actions. Redundancy can be applied to the system level, for example, by having multiple payloads onboard the same UAV. By performing a cause-effect mapping, described below, system architects can see that having redundant UAVs reduces susceptibility to more perturbations than just redundant payloads. For example, while both solutions will be viable in the event that a single payload fails, only the redundant UAV solution will be viable in numerous events such as engine failure, an operator not showing up for work, UAV being shot down, etc. Furthermore, redundant UAVs also provide margin, whereas in most cases having multiple cameras of the same type onboard the same UAV will not improve capacity. The main disadvantage to applying redundancy at the SoS level in this particular case would be cost.

While redundancy may be more effective at the SoS level, the strategy of mobility, for example, may be more applicable at the system level. Mobility provides viability in the presence of location-based disturbances. A classic example is that in systems of systems that tend to have geographic distribution of their components, there is often not one SoS location. Rather, the SoS is made up of components that each may have their own location, which may be affected by local disturbances. As location-based perturbations are local and do not apply to the entire SoS, similarly the solution of mobility is local and applicable only to the specific constituent systems that may be affected.

2) Intensity Regulation

In order for a SoS to be survivable, strategies must be implemented where the constituent systems ensure that their actions do not negatively impact the contexts of other constituent systems, if those changes may not be survivable. In other words, they must not disrupt the context of other systems, if those systems are fragile to those disruptions.

In the case of an intensity-based disturbance like that involving current, the intensity regulation of the system's output is an example of a strategy that can help enable the survivability of systems. Throttling is when a system intentionally reduces its output, to increase its own survivability to a perturbation it is experiencing, or help other constituent systems avoid future perturbations by not disrupting their context. In the case of power grids, throttling is achieved in part through "shedding load", i.e. eliminating power to some areas ("rolling blackouts") and/or reducing the voltage provided ("brownouts").

There are other examples of intensity throttling used in both traditional systems and systems of systems as well. In most modern laptop computers, the CPU and GPU produce the lion share of the heat. Heat is an unintentional interaction effect that threatens the component within the laptop. If the internal temperature of the laptop rises above a critical threshold, then certain components, such as the RAM or hard drives, may fail as a result. To prevent this, CPUs and GPUs in most modern laptops will down-clock themselves, i.e. operate at a lower speed and produce less heat, for a period of time when the overall internal temperature is too high. On the other end of the SoS spectrum, throttling is specifically implemented as part of the Transfer Control Protocol (TCP) protocols that forms part of the operational backbone of the Internet itself. Specifically, TCP contains rules for end-to-end flow control that prevents individual senders from sending too much data to receivers and overstressing them to the point where they cannot respond to everything in a timely manner. Sending more data to a receiver is precisely the perturbation behind the infamous Distributed Denial Of Service (DDoS) attacks are the leading cause of Internet websites to fail, due to exogenous "hacking". In a DDoS, multiple senders are used (often because they are compromised via a malevolent computer program) to send data to a single receiver, overflowing the receivers queues with bogus requests and reducing its ability to effectively service legitimate clients.

In the language of Richards[21], intensity regulation is a failure reduction strategy to ensure survivability that applies to systems and systems of systems.

D. Cause-Effect Mapping to determine survivability strategies for the MarSoS

To develop a list of perturbations of interest, and to determine possible points of intervention where viability strategies can be applied, a cause-effect mapping is performed where the multiple causes and effects of perturbations are linked from spontaneous events to terminal conditions. Spontaneous events are those outside the system's control. In many cases, spontaneous events are exogenous (i.e. outside the system boundary), such as changes in the weather or the action of an outside entity. Spontaneous events can also be endogenous, such as a random component failure or operator error. Spontaneity is not absolute, but rather it is relative to the system.

A cause-effect mapping begins with a terminal event (Fgure 13). The causes of that event are added to the map as perturbations of interest and a single arrow from each of the causes to terminal event is drawn. If any of the causes of the terminal event are not

spontaneous events, then their causes, along with appropriate arrows, are added to the map as additional perturbations of interest. Then each perturbation of interest is examined for additional causes and effects. Arrows are drawn to any causes and effects already on the map and new perturbations are added, as necessary. This process continues until each perturbation originally started with a spontaneous event, and eventually results in a terminal event. If a perturbation cannot eventually lead to a terminal event, then it is not worth considering and should be removed from the cause-effect mapping.

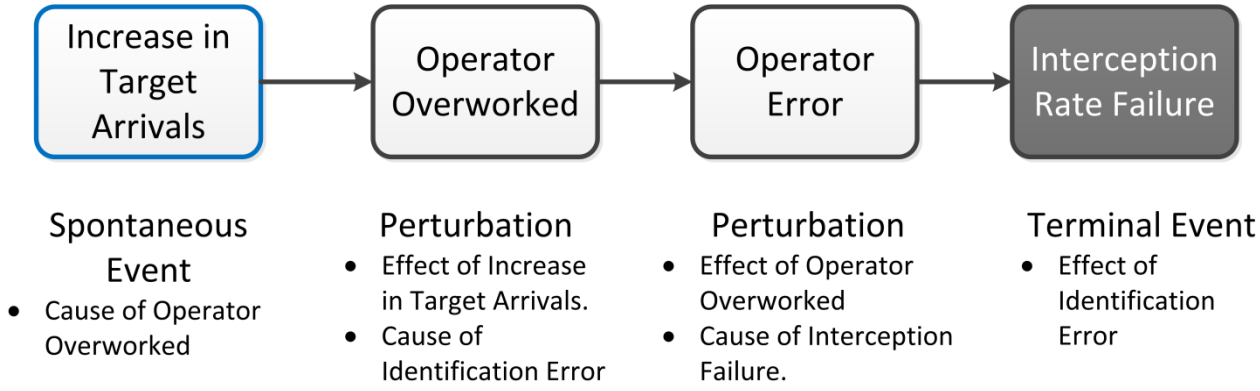

Figure 12 Simple Cause Effect Mapping

The cause-effect mapping is useful for being able to highlight areas of intervention, where strategies can be implemented to increase viability by mitigating or recovering from causes by preventing their effects from happening. For example, in Figure 14, operators tend to perform more errors when they are under a high workload (Wiener & Nagel, 1988). This perturbation can lead to a terminal event (e.g. interception rate failure), so reducing the probability of operators being overworked is one strategy for increasing the viability of the system overall. This can be achieved, for example, by incorporating automation into the identification process that double checks operator's actions for common mistakes.

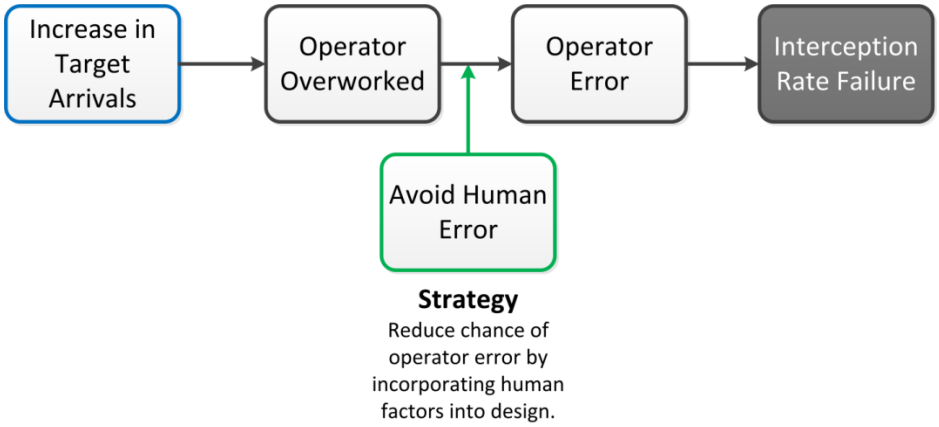

Figure 13: Applying a survivability strategy as an intervention

A cause-effect mapping was performed for the MarSoS (Figure 15). Not all perturbations are shown in the figure, rather only a very small subset is shown to highlight where new strategies for enhancing viability might be made.

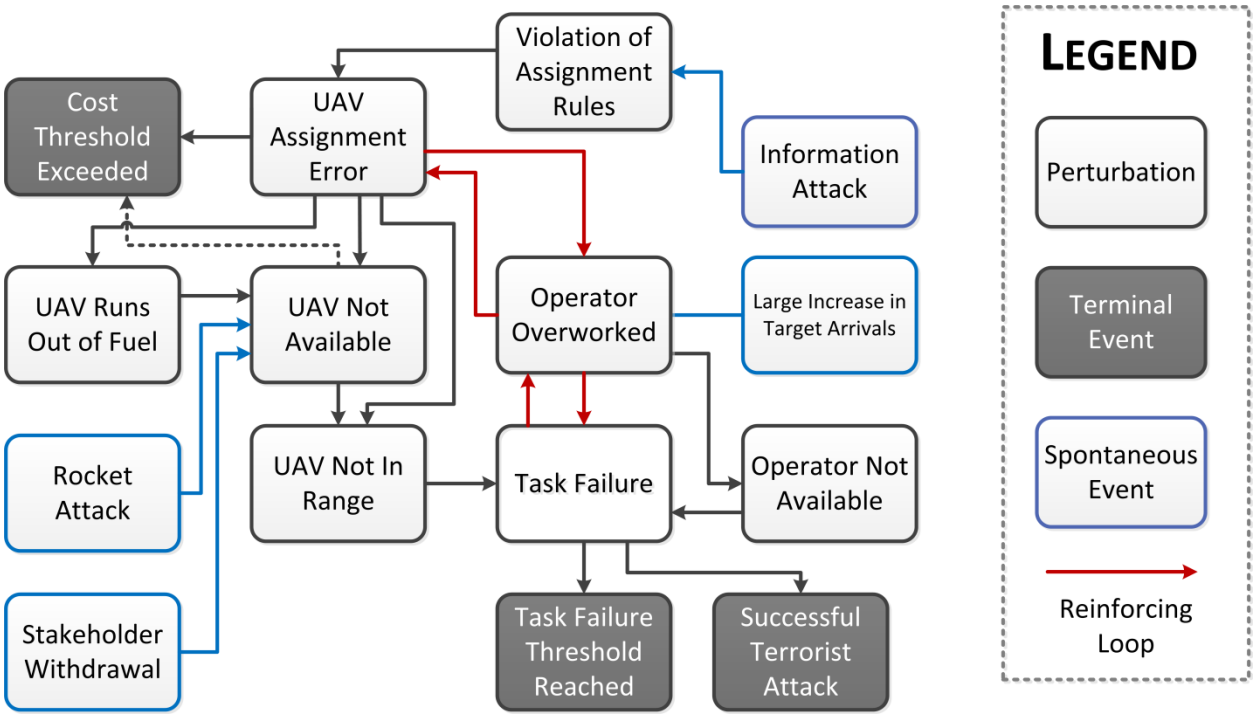

Figure 14 Cause-Effect Mapping for the MarSoS

The causal diagram highlights several interesting dynamics associated with how perturbations in this particular system propagate. System architects are constrained in how they can intervene to solve some of the problems caused by the perturbations. There is a limited amount of time, money and other resources to implement solutions to every perturbation. There are also design constraints, such as size and weight, as well as boundary constraints that limit the influence system architects have. For these reasons, system architects must be selective in the strategies they chose to implement. A cause-effect mapping enables system architects to visually identify possible points of intervention where the system or supporting enterprise can do something to avoid, mitigate and recover from certain types of perturbations. In particular, system architects should try to intervene to avoid perturbations that have multiple effects, known as common causes, and survive perturbations that have multiple causes, known as common effects. It is particularly important to intervene against perturbations that are part of a reinforcing loop, since they are both the sources of multiple effects and multiple causes simultaneously, and lead to the often devastating cascading failures seen in large-scale systems of systems. Cause-effect mappings also help system architects deal with "unknown unknowns", or perturbations that no one thought of. Sometimes the solution to a known problem is also the solution to unknown problem. Common causes and common effects are probably more likely to be causes and effects of unknown perturbations as well.

By using the cause-effect mapping to generate perturbations of interest, recognizing the differences between system and global contexts, and looking at some related historical cases, some interesting strategies for enabling survivability emerge. These, along with the points of intervention, are shown in
Figure 15.

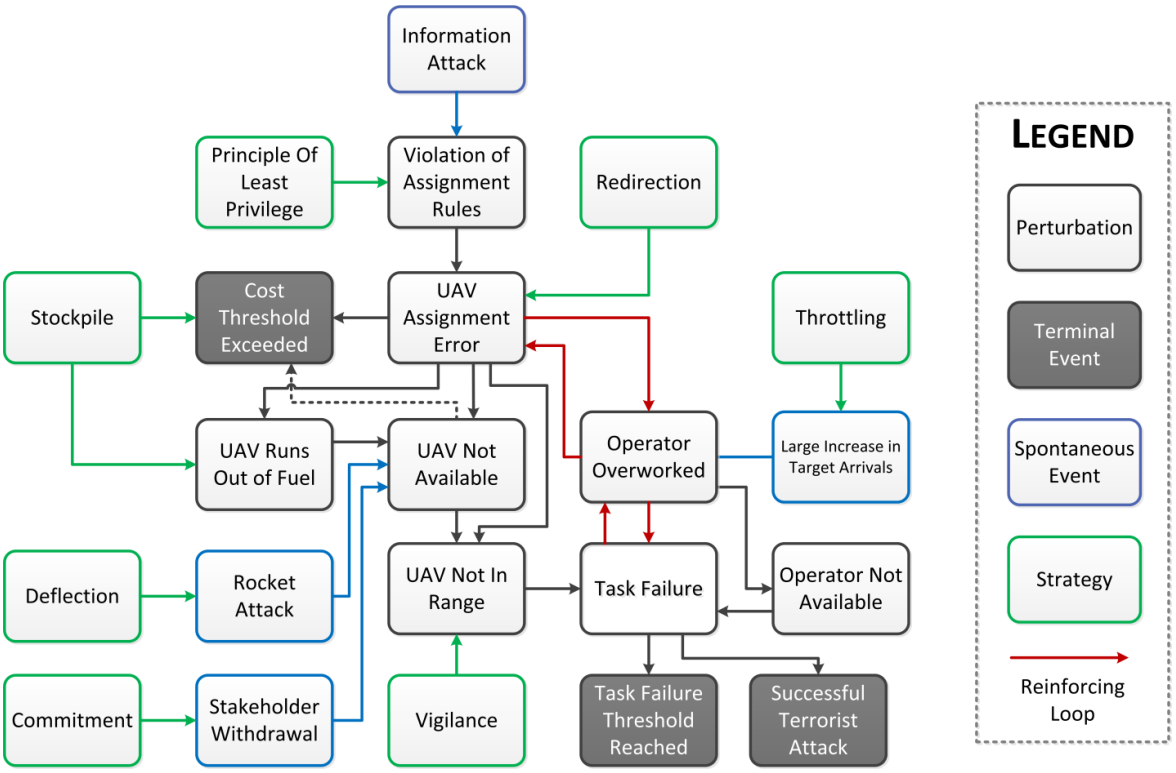

Figure 15 Cause-Effect Mapping for the MarSoS with Survivability Strategies applied

E. Illustrations from MarSoS

Using historical case studies, such as the Northeast Blackout of 2003, cause-effect mapping, and experimenting with the MarSoS DES, some strategies for avoiding and/or surviving perturbations, particularly in systems of systems, were discovered. These strategies are summarized in Table 3.

Table 3 Perturbations, Strategies and Examples for Survivability for the MarSoS

PERTURBATIONS	STRATEGY	STRATEGY DESCRIPTION	HISTORICAL CASE
Cost Threshold Exceeded UAV Runs Out of Fuel	Stockpile	Store excess consumable resources to prevent disruptions of supply.	Strategic Petroleum Reserve (SPR) by the United States.

Rocket Attack	Deflection	Divert perturbations away from the system. Deflection is diversion of exogenous perturbations away from critical components.	Chaff used during WWII to deceive radar systems.
Stakeholder Withdrawal	Commitment	Ensure that critical entities do not withdraw their support to the system.	Service Level Agreement (SLA) such as Google API SLA.
Violation of Assignment Rules	Authenticaion	Verify entities before allowing them within system boundary.	Username and password on major commercial websites, ID badges at nuclear power plants
	Strategy Of Least Privilege	Components should only be limited in their influence of other components to only what is necessary.	User account privileges on computer networks such as the Sony PlayStation Network (PSN).
PERTURBATIONS	**STRATEGY**	**STRATEGY DESCRIPTION**	**HISTORICAL CASE**
UAV Not In Range	Vigilance	Monitor for both internal and external changes.	Multiple security checkpoints at an airport.
Operator Overworked	Redirection	Divert perturbations away from critical components. Redirection is diversion of endogenous perturbations away from critical components.	Transfer Control Protocol (TCP) used in the Internet.
Large Increase in Target Arrivals	Throttling	Reduce intensity of outputs to prevent perturbations to nearby contexts.	CPU downclocking to avoid excessive heat spreading to nearby components.

To choose one example, the strategy of vigilance can be demonstrated by the MarSoS DES. In the terrorist scenario, the terrorist ship appears as a fisherman when enters the AOI, typically passing through the normal detection/identification phase as any other civilian ship would. When the terrorist is close to the port, he/she speeds up and ignores coastal regulations on its way to detonate a bomb in the port. If the MarSoS is not vigilant, then by the time it notices the infraction and sends a patrol boat to intercept, the terrorist has already reached the port and detonated the bomb (a terminal event). However, the strategy of vigilance can be applied where friendly vehicles heading to port are tracked by patrol boats after they have been identified.

Performing a 2x2 design of experiments, where the vigilance strategy is applied against the control group, and looking at both a light traffic (λ = 11.25 ships / h) and heavy traffic (λ = 22.5 ships / h) context, the following two contingency tables were generated after running each treatment 10,000 times.

Graphing the results shows a strong difference in both contexts when the vigilance strategy is applied (see Figure 16). Using a Fisher exact t-test, there is a significant difference between the vigilant strategy and the control in both the light traffic and heavy traffic contexts ($p < 0.0001$).

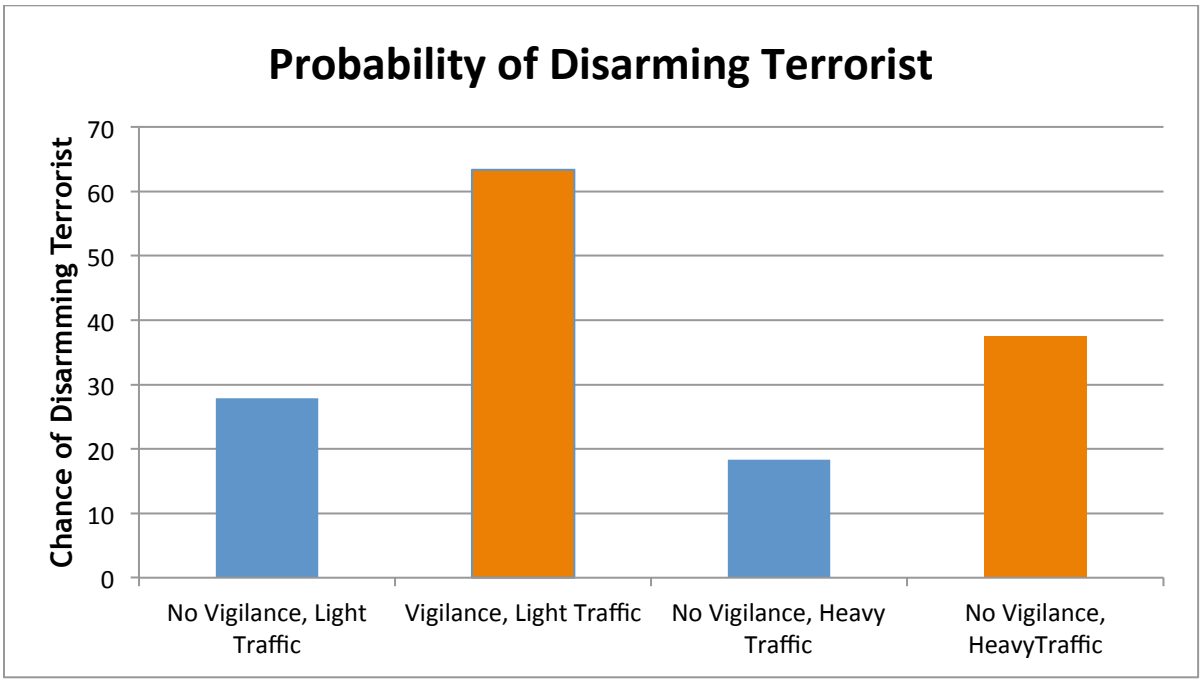

Figure 16 Comparison of Vigilance to no Vigilance Strategy for the MarSoS

The strategy of vigilance for the MarSoS is an example of a survivability strategy that has no meaning with the context of a traditional system. It is a viable strategy because the MarSoS has loosely coupled elements, geographic diversity and local contexts. These both allow the possibility of intrusion by outside elements and suggest a strategy (vigilance) as a means to deal with such hostile (and rare) outside elements.

As another example, the strategy of commitment is one that only makes sense because the SoS allows the possibility of coupling through influence (which can be withdrawn). For traditional systems, which are strongly coupled (e.g. a satellite), the concept of commitment would have no meaning.

VI. DISCUSSION

This paper has demonstrated that for one kind of system of systems (the directed SoS), the meta-property of survivability may not be understood the same way as for a system. We have demonstrated that some strategies for survivability may be the same (between a system and a system of systems), some may be implemented at the system level or system of system level with different levels of effectiveness (e.g. margin) and some have no meaning for systems (e.g. vigilance or commitment). In one sense, the loose coupling of system of systems opens up new degrees of freedom and abstract design spaces for survivability strategies relative to (traditional) systems.

So far, the paper has only demonstrated the existence of these differences for one of the "ilities" and for one kind of SoS. There has been considerable work done on some of the other "ilities" such as flexibility and adaptability[2]. We hypothesize that strategies for implementation of some of these "ilities" will show the same richness for Systems of Systems relative to Systems i.e. that they may be modalities available for flexibility in a SoS that do not exist for a system. The same may be true for evolvability, extensibility etc.

VII. CONCLUSIONS

In this paper, the question of whether Systems of Systems could be treated as (traditional) systems or were a new class of systems was examined. It was shown that SoSs as conventionally defined had properties not found in traditional systems. For example, the possibility that elements of an SoS could belong to more than one SoS at the same time. This calls into question the idea of a sharp system boundary. Another difference is found in the idea of influences. Some types of SoS allow for economic, social or regulatory influences between the component elements. The traditional definition of systems would not include this kind of coupling. However, these differences have been previously explored.

In this paper, an additional question was explored. This was whether the "illities" which meta-properties associated with systems were the same or different for systems of systems. By examining historical examples and by using a maritime security SoS as a testbed, this paper showed that the "illity" called survivability had some strategies that directly mapped from systems and also allowed new strategies that only made sense for a SoS (e, g reversion to an intermediate stable form and vigilance).

This shows that new strategies for "illities" exist in one "illity". However there are numerous of these "beyond first use properties" known to exist for complex systems. We thus hypothesize for some of the other "illities" (flexibility, adaptability et cetera) that that there might be expanded strategies associated with them being applied to SoSs. This opens up a rich new area of research for "beyond first use properties".

VIII. REFERENCES

[1] T. P. Hughes, *Rescuing Prometheus*. New York: Vintage Books, 2000.

[2] O. L. De Weck, *Engineering systems: meeting human needs in a complex technological world*. Cambridge, Mass: MIT Press, 2012.

[3] "System - Definition and More from the Free Merriam-Webster Dictionary." [Online]. Available: http://www.merriam-webster.com/dictionary/system. [Accessed: 23-Aug-2013].

[4] M. W. Maier, "Architecting principles for systems-of-systems," *Syst. Eng.*, vol. 1, no. 4, pp. 267–284, 1998.

[5] J. Boardman and B. Sauser, "System of Systems - the meaning of of," pp. 118–123.

[6] N. Karcanias and A. G. Hessami, "System of Systems and Emergence Part 1: Principles and Framework," 2011, pp. 27–32.

[7] "Systems Engineeirng Guide for Systems of Systems," Departmemt of Defense report, 2008.

[8] Y. Correa and C. Keating, "An approach to model formulation for systems of systems," vol. 4, pp. 3553–3558.

[9] A. Gorod, B. Sauser, and J. Boardman, "System-of-Systems Engineering Management: A Review of Modern History and a Path Forward," *IEEE Syst. J.*, vol. 2, no. 4, pp. 484–499, Dec. 2008.

[10] J. S. Dahmann and K. J. Baldwin, "Understanding the Current State of US Defense Systems of Systems and the Implications for Systems Engineering," 2008, pp. 1–7.

[11] "Future Combat Systems," *Wikipedia, the free encyclopedia*. 21-Aug-2013.

[12] "NPOESS," *Wikipedia, the free encyclopedia*. 24-Aug-2013.

[13] H. Eisner, "A systems engineering approach to architecting a unified system of systems," vol. 1, pp. 204–208.

[14] A. M. Ross, D. H. Rhodes, and D. E. Hastings, "Defining changeability: Reconciling flexibility, adaptability, scalability, modifiability, and robustness for maintaining system lifecycle value," *Syst. Eng.*, vol. 11, no. 3, pp. 246–262, Jun. 2008.

[15] *Space mission analysis and design*, 3rd ed. El Segundo, Calif. : Dordrecht ; Boston: Microcosm ; Kluwer, 1999.

[16] M. Bjelkemyr, D. Semere, and B. Lindberg, "An Engineering Systems Perspective on System of Systems Methodology," 2007, pp. 1–7.

[17] S. Crowley, "Level 3, Cogent resolve peering dispute, renew deal," *COMPUTERWORLD*, 2005.

[18] K. G. Binmore, *Playing for real: a text on game theory*. Oxford ; New York: Oxford University Press, 2007.

[19] J. Dahmann, G. Rebovich, J. Lane, R. Lowry, and K. Baldwin, "An implementers' view of systems engineering for systems of systems," 2011, pp. 212–217.

[20] R. De Neufville, *Flexibility in engineering design*. Cambridge, Mass: MIT Press, 2011.

[21] M. G. Richards, A. M. Ross, N. B. Shah, and D. E. Hastings, "Metrics for Evaluating Survivability in Dynamic Multi-Attribute Tradespace Exploration," *J. Spacecr. Rockets*, vol. 46, no. 5, pp. 1049–1064, Sep. 2009.

[22] O. L. De Weck, A. M. Ross, and D. H. Rhodes, "Investigating Relationships and Semantic Sets amongst System Lifecycle Properties (Ilities)," in *Proceedings of the Third International Engineering Systems Symposium*, TU Delft, 2012.

[23] R. Albert, I. Albert, and G. Nakarado, "Structural vulnerability of the North American power grid," *Phys. Rev. E*, vol. 69, no. 2, Feb. 2004.

[24] M. Zhivich and R. K. Cunningham, "The Real Cost of Software Errors," *IEEE Secur. Priv. Mag.*, vol. 7, no. 2, pp. 87–90, Mar. 2009.

[25] B. Mekdeci, A. M. Ross, D. H. Rhodes, and D. Hastings, "System architecture pliability and trading operations in tradespace exploration," 2011, pp. 387–394.

[26] J. Buschmann, T. Crider, F. Guillermo, E. . Ferraris, H. Gungor, S. Hoffmann, and M. Kelley, “Martime Domain Protection in the PACOM AOR,” Naval Postgraduate School, Monteray, CA, 2005.

IX. ACKNOWLEDGEMENTS

The authors would like to acknowledge useful discussions with the members of the SEAri group, Prof. Richard de Neufville, Prof. Joe Sussman and Prof. Oli deWeck