NPS-AM-10-063

EXCERPT FROM THE PROCEEDINGS

OF THE

SEVENTH ANNUAL ACQUISITION RESEARCH SYMPOSIUM THURSDAY SESSIONS VOLUME II

Acquisition Research
Creating Synergy for Informed Change

May 12 - 13, 2010

Published: 30 April 2010

Prepared for: Naval Postgraduate School, Monterey, California 93943

The research presented at the symposium was supported by the Acquisition Chair of the Graduate School of Business & Public Policy at the Naval Postgraduate School.

To request Defense Acquisition Research or to become a research sponsor, please contact:

NPS Acquisition Research Program
Attn: James B. Greene, RADM, USN, (Ret.)
Acquisition Chair
Graduate School of Business and Public Policy
Naval Postgraduate School
555 Dyer Road, Room 332
Monterey, CA 93943-5103
Tel: (831) 656-2092
Fax: (831) 656-2253
E-mail: jbgreene@nps.edu

Copies of the Acquisition Sponsored Research Reports may be printed from our website www.acquisitionresearch.net

System Development and Risk Propagation in Systems-of-Systems

Muharrem Mane—Muharrem Mane is a postdoctoral researcher in the School of Aeronautics and Astronautics Engineering, Purdue University. He received his PhD from Purdue University in Aerospace Engineering in 2008. His current research interests are in risk analysis and propagation, resource allocation and design under uncertainty, and network modeling and analysis. He currently works in the System-of-Systems Laboratory led by Dr. DeLaurentis.

Muharrem Mane
Postdoctoral Researcher, School of Aeronautics and Astronautics
Purdue University
701 W. Stadium Avenue
West Lafayette, IN 47807
Phone: (765) 494 7958
Fax: (765) 494-0307
mane@purcue.edu

Daniel DeLaurentis—Daniel DeLaurentis is an Assistant Professor in the School of Aeronautics and Astronautics Engineering, Purdue University. He received his PhD from Georgia Institute of Technology in Aerospace Engineering in 1998. His current research interests are in mathematical modeling and object-oriented frameworks for the design of system-of systems, especially those for which air vehicles are a main element, and approaches for robust design, including robust control analogies and uncertainty modeling/management in multidisciplinary design.

Daniel DeLaurentis
Assistant Professor, School of Aeronautics and Astronautics
Purdue University
701 W. Stadium Avenue
West Lafayette, IN 47807
Phone: (765) 494 0694
Fax: (765) 494-0307
ddelaure@purcue.edu

Abstract

The emphasis of the Department of Defense on capability-based acquisition has led to the simultaneous development of systems that must eventually interact within a system-of-systems. Thus, system development and acquisition processes encounter interdependencies that generate complexity and risk. The authors' prior work has developed a Computational Exploratory Model to simulate the development processes of these complex networks of systems intended for a system-of-systems capability. The model's goal is to understand the impact of system-specific risk and system interdependencies on development time. The progress documented in this paper focuses on the quantification of risk propagation and the impact of network topologies on the propagation of disruptions. The improved model enables trade studies that differentiate the effectiveness of alternate configurations of constituent systems and that quantify the impact of varying levels of interdependencies on the timely completion of a project that aims to achieve a desired capability level.

Introduction

The purpose of capabilities-based acquisition, as described by Charles and Turner (2004), is to acquire a set of capabilities instead of acquiring a family of threat-based, service-specific systems. The Missile Defense Agency (MDA), for example, uses capability-based acquisition to evaluate the success of a program based on its ability to provide a new capability for a given cost, and not on its ability to meet specific performance requirements (Spacy, 2004). The Joint Mission Capability Package (JMCP) concept is another example that aims to create a joint interdependency between systems to combine capabilities in order to maximize reinforcing effects and minimize vulnerabilities (Durkac, 2005). The goal is a more efficient utilization of both human and machine-based assets and, in turn, improved combat power.

To accomplish the desired capability, systems are increasingly required to interoperate along several dimensions that characterizes them as systems-of-systems (SoS) (Maier, 1998). Systems-of-systems most often consist of multiple, heterogeneous, distributed systems that can (and do) operate independently but can also collaborate in networks to achieve a goal. Examples of systems-of-systems include: civil air transportation (DeLaurentis, Han & Kotegawa, 2008), battlefield ISR (Butler, 2001), missile defense (Francis, 2007), etc. According to Maier (1998), the distinctive traits of operational and managerial independence are the keys to making the collaboration work. The network structure behind the collaboration, however, can contribute both negatively and positively to the successful achievement of SoS capabilities and, even earlier, to the developmental success. Collaboration via interdependence may increase capability potentials, but it also contains concealed risk in the development and acquisition phases. Brown and Flowe (2005), for instance, have investigated the implications of the development of SoS to understand the drivers that influence cost, schedule, and performance of SoS efforts. Results of their study indicate that the major drivers—as indicated by subject-matter-experts—include systems standards and requirements, funding, knowledge, skills and ability, system interdependencies, conflict management, information access, and environmental demands.

Disruptions in the development of one system can have unforeseen consequences on the development of others if the network dependencies are not accounted. The goal of a single system’s program manager is the mitigation of risk, leading to successful development of that specific system. While direct or immediate consequences of decisions are nearly always considered, the cascading second-and-third order effects that result from the complex interdependencies between constituent systems in an SoS are often not, which make success all the more difficult. It falls on acquisition managers and systems engineers (or systems-of-systems engineers) to understand and manage the successful development of a system, or family of systems, to produce the targeted capability in this challenging setting.

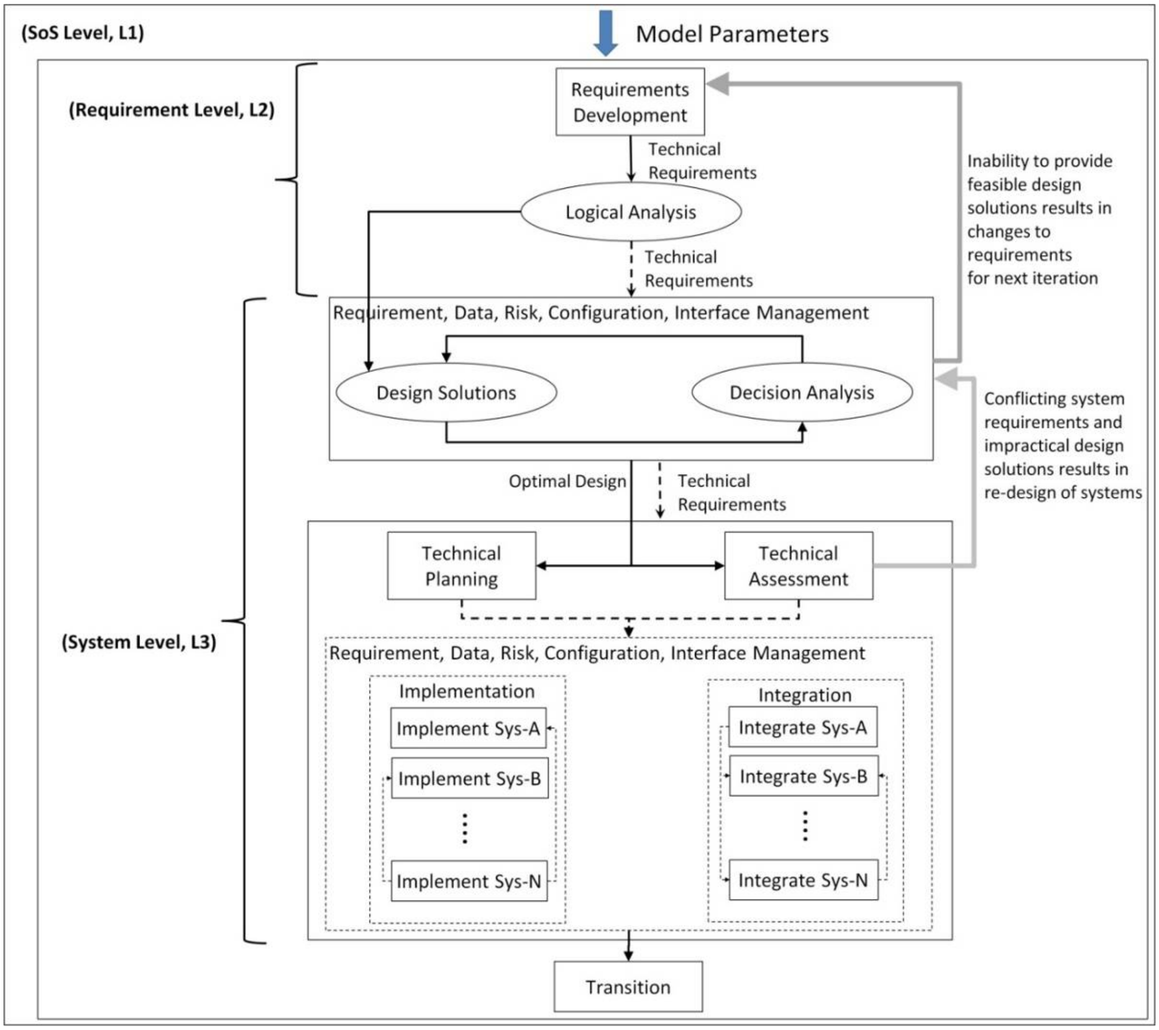
Evidence is abundant that system-of-systems-oriented endeavors have struggled to succeed amidst the development complexity. The Future Combat System is a latest example (Gilmore, 2006). Civil programs have not been spared either, e.g., Constellation Program (Committee on Systems, 2004) and NextGen (2009). Rouse (2001) summarizes the complexity of a system (or model of a system) as related to the intentions with which one addresses the systems, the characteristics of the representation that appropriately accounts for the system’s boundaries, architecture, interconnections and information flows, and the multiple representations of a system.

The work presented in this paper specifically targets complexities stemming from system development risk, the interdependencies among systems, and the span-of-control of the systems or system-of-systems managers and engineers. *The objective of the research summarized in this paper is to quantify the impact of system-specific risk and system interdependency complexities using a computational exploratory modeling approach.* The work comprises new improvements to a computational exploratory model (CEM)—a discrete event simulation model—previously introduced in prior Acquisition Symposia (Mane & DeLaurentis, 2009) that aims to provide decision makers with insights into the development process by propagating development risk in the SoS network and capturing the impact that system risk, system interdependencies, and system characteristics have on the timely completion of a program. We also briefly introduce complementary work related to an analytical approach to treat the same complexities via computations on conditional probabilities that relate the transmission of risk in network dependent systems.

Computational Exploratory Model (CEM) Overview

The CEM is based on the 16 basic technical management and technical system-engineering processes outlined in the *Defense Acquisition Guidebook* (DoD, 2008a), often referred to as the 5000-series guide. However, an SoS environment changes the way these processes are applied. The *Systems Engineering Guide for System-of-Systems (SoS-SE)* (DoD, 2008b) addresses these considerations by modifying some of the 16 processes in accord with an SoS environment. The resulting processes and respective functions consist of translating inputs from relevant stakeholders into technical requirements, developing relationships between requirements, designing and building solutions to address requirements, integrating systems into a high-level system element, and performing various managing and control activities to ensure that requirements are effectively met, risks are mitigated, and capabilities achieved.

The CEM, centered on these revised processes, is a discrete event simulation of the development and acquisition process. This process creates a hierarchy of analysis levels: SoS Level (L1), Requirement Level (L2), and System Level (L3). Component elements at each level are a network representation of the level below. The SoS Level (L1) is comprised of the numerous, possibly interdependent, requirements (L2) needed to achieve a desired capability. Similarly, satisfaction of each requirement in the Requirement Level (L2) requires a number of possibly interdependent systems (L3). **Error! Reference source not found.** presents the description of the process modeled by the CEM.



Conceptual Model of Acquisition Strategy based on SoSE Process
(described in DoD, 2008b)

At the Requirement Level (L2), *Requirements Development* contains the technical requirements of the SoS (provided externally). The technical requirements are then examined in *Logical Analysis* to check for interdependencies amongst the requirements. A check for inconsistencies amongst requirements is also performed. *Design Solution* development and *Decision Analysis* are the next processes, which belong to the System Level (L3). They produce the optimal design solution from the set of feasible solutions to meet the given requirements. The optimal design solution is based not only on the current set of requirements and solution alternatives but also takes into account all previous information available through requirements, risk, configuration, interface and data management processes. Because most acquisitions are multi-year projects involving many different parties, the overlap between the management processes, *Design Solution* and *Decision Analysis* processes, allows for greater tractability of decisions. It is at this stage that system interdependencies are identified. The optimal design solution obtained from this phase is then sent to the next stage: *Technology Planning* and *Technology Assessment*. In the event that an optimal or sub-optimal design solution to successfully implement the given requirements does not exist, the feedback loop to *Requirement Development* translates into a change in the technical requirements for the SoS. *Technology Planning* and *Technology Assessment* are System Level (L3) scheduling processes that oversee the implementation, integration, verification and validation for all the component systems in the SoS.

Systems in the SoS are often dependent on other systems for either implementation, integration, or both. Disruptions during these stages of development in one of the systems result in time-lags in the acquisition process and to delays that propagate through the network of component systems impacting seemingly independent systems. For example, if the implementation of a system A is dependent on the Implementation of a system B—as could be the case for the development of an aircraft that depends on the specifications of a radar system—funding cuts to system B can result in development delays in system B but can also impact the development of system A. If, on the other hand, a third system C depends on system A, this could also be affected by the problems caused in system C due to funding cuts.

The *Implementation* and *Integration* Phases of component systems constitute the lowest level of detail modeled in the CEM. The design decisions made at earlier stages must be implemented and integrated in these phases to generate the final product of a program. **Error! Reference source not found.** presents an abstraction of the layered networks that result from the modeling of the acquisition process: systems are grouped to satisfy a requirement, and requirements are grouped to generate a capability.

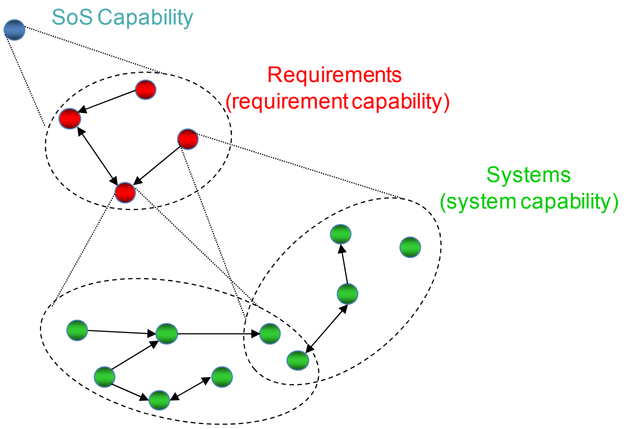


Figure 1. Layered Network Abstraction of Computational Exploratory Model

Systems can be independent, can satisfy several requirements, and can depend on other systems. The CEM simulates these layered relationships to capture the impacts that any changes—related to decision-making, policy, or development—in any of the component systems, requirements, and relationships between them have on the completion of a project. The exercise of the CEM described in this paper specifically targets complexities stemming from system risk, the interdependencies among systems, and the span-of-control of the SoS authority (if present). The next section will present the model dynamics that make possible the study of these complexities and will explore the design space of the SoS authority and tradeoffs between development risk and the number of systems and system interdependencies in an SoS.

Detailed Model Dynamics

The CEM operates as a discrete event simulator of the development process. Several challenges arise in developing a model for purposes of simulation and learning. Disruptions occur at various stages of development and are governed by the risk associated with the project or individual systems. The CEM models risk associated with the implementation and integration of each component system as well as the risk due to the system interdependencies. Furthermore, systems and SoS engineers are often faced with the decision of using legacy assets to satisfy a given requirement or opt for the development of brand new ones. The CEM includes parameters such as *readiness-level* to differentiate between legacy assets/platforms, new systems, and partially implemented/integrated systems (i.e., systems under development) and to investigate the impact that the inclusion of such systems in the development of an SoS has on the success of a project. The next sub-sections describe the model details: parameters and inputs, Implementation and Integration dynamics, and the risk model.

Model Input Parameters

Error! Reference source not found. presents the input parameters and the remainder of this section expands and explains their role in the ECM.

Table 1. Input Parameters of Computational Exploratory Model

Parameter	Notation	Description
	Requirement Level (L2)	
Requirement dependencies	D_{req}	Adjacency matrix that indicates requirement interdependencies
Risk profile	R_{req}	Probability of disruptions in *Requirement Development Phase*
Impact of disruptions	I_{req}	Time penalty when disruptions hit *Requirement Development Phase*
	System Level (L3)	
System dependencies	D_{sys}	Adjacency matrix that indicates system interdependencies
Development pace of design	t_{des}	Increase in completion of *Design Solutions Phase*
Design risk profile	R_{des}	Probability of disruptions in *Design Solutions Phase*
Impact of design disruptions	I_{des}	Time penalty when disruptions hit *Design Solutions Phase*
Span-of-control	soc	Indicator of how *Implementation* and *Integration* are performed (sequentially or simultaneously)
System initial readiness-level	$m^0(i,r)$	Initial readiness-level of system *i* to satisfy requirement *r* (for *Implementation Phase*)
System risk profile	$R_{sys}(i,r)$	Probability of disruptions (during implementation) of system *i* when satisfying requirement *r*
Impact of disruptions	$I_{sys}(i)$	Time penalty when disruptions hit system *i* during Implementation/Integration
Implementation pace	$p_{imp}(i)$	Increase in readiness-level at each time step during implementation of system *i*
Integration pace	$p_{int}(i)$	Increase in completeness-level at each time step during integration of system *i*
Implementation start	$l_{imp}(i,j)$	Readiness-level of system *j* when *Implementation Phase* of dependent system *i* begins
Strength of dependency	$S(i,j)$	Strength of dependency of system *i* on system *j*

The requirement dependency matrix (D_{req}) indicates how the development and satisfaction of requirements depend on each other, which impacts the sequence in which requirements are developed and satisfied. For example, if Requirement A depends on Requirement B, then development of Requirement A begins when Requirement B has been satisfied. As requirements are developed, the risk profile (R_{req}) of *Requirement Development* indicates the probability of disruptions at this stage in the development process. Disruptors signify a change in requirements or addition of new requirements. When a requirement is changed after the acquisition process has begun, it affects all subsequent processes and it causes a time delay (I_{req}) that is added to the project time. Every requirement that is implemented is fed into its own *Design Solution* and *Decision Analysis* (**Error! Reference source not found.**) process. The *Design Solution* and *Decision Analysis* processes feed into each other and the risk profile (R_{des}) indicates the probability of disruptions at each time-step during the completion of the stage with a value between 0 and 1. Any disruptions at this stage indicate that the design solution provided is not feasible and a time penalty (I_{des}) that indicates a re-design of the solution is incurred. If the solution fails in multiple consecutive time-steps, then the requirement is sent back to *Requirement Development* stage, otherwise the set of component systems and their user-defined parameters are sent to the *Technical Planning* and *Technical Assessment* (**Error! Reference source not found.**) processes based on the development-pace parameter of this stage.

Implementation Phase Dynamics

Technical Planning is the stage in which *Implementation* and *Integration* of component systems is performed. The *Implementation Phase* simulates the development of each system. The nature of candidate systems may range from legacy systems to off-the-shelf, plug-and-play products to custom-built, new systems. Development of a “brand new” SoS has been and will remain a rare occurrence. In their study on SoS, the United States Air Force (USAF) Scientific Advisory Board (Saunders et al., 2005) stated that one of the challenges in building an SoS is accounting for contributions and constraints of legacy assets. Similarly, the regular utilization of off-the-shelf component systems in both defense and civil programs contribute to cost and time savings but also introduce a different type of risk to the system development process (Constantine& Solak, 2010). These legacy systems may be used “as-is” or may need re-engineering to fulfill needs of the new program.

Here, we define *legacy systems* as systems that have been developed in the past to achieve a particular requirement, and *new systems* as not-yet-developed systems envisioned to satisfy a new requirement. When considering the use of legacy systems to meet a new requirement, the capability of these systems to satisfy the new requirement is not necessarily the same as their capability to meet the original requirement for which they were designed. Additionally, the risk associated with the modification of a legacy system and the risk associated with the development of a brand new system can be quite different. Legacy systems may, however, provide cost and/or time benefits if modifications are less severe than a new development, as is the case with new systems. To delineate systems in a meaningful way, we describe the spectrum of a system’s ability to satisfy a requirement in terms of its readiness-level.

System readiness-level, a concept proposed by Sauser et al. (2006), is a metric that incorporates the maturity levels of critical components and their readiness for integration (i.e. integration requirements of technologies). This is an extension of the widely used

Technology Readiness Level (TRL), a metric that assesses the maturity level of a program's technologies before system development begins (Department of Defense Directive 5000.2, 2005). While similar in spirit to the SRL metric proposed by Sauser, Verna, Ramirez-Marquez, and Gove (2006), readiness-level in the present work is defined in a different manner and with less detail. We define system readiness level as the readiness-level of a system *i* to satisfy requirement *r*, *m*(*i,r)*, with a value between 0 and 1. A system with a readiness-level of 1 is a fully developed system that can provide a certain level of capability. The dynamic model starts the *Implementation Phase* of a system from its initial readiness-level and simulates its development/implementation until it reaches a readiness-level of 1. An initial readiness-level of 0 indicates a brand new system that must be developed from scratch, while a system with an initial readiness-level greater than 0 indicates a legacy system that is partially developed to satisfy a requirement *r*, but needs further development to reach a readiness-level of 1. In general, careful research of a candidate system *i* will determine its initial readiness-level to satisfy a requirement *r*, and, therefore, the amount of development necessary to achieve a readiness-level of 1.0.

The CEM simulates the *Implementation Phase* as a series of time steps in which a pre-determined increment of readiness ($p_{imp}(i)$) is gained at each time-step of each system *i*, or lost if a disruption occurs (according to the system risk profile of system *i* in satisfying requirement *r*, $R_{sys}(i,r)$). This is clearly a gross simplification of the actual development process for a system; however, it adequately serves the purposes of the research, which is focused on the interdependencies between systems to develop an SoS capability and aims to capture the impact of disruptions on the development process. Accurate modeling of the *Implementation Phase* would increase the accuracy of the model for a particular application, but it would not change the nature of the observed results.

Representation of Risk

The risk associated with the development of a system is a function of its inherent characteristics (technology, funding, and complexity levels) and on risk levels of the systems on which it depends. The former may be estimated via a variety of analysis techniques that examine a system in detail, but the latter requires knowledge of system interdependencies which can be numerous, complicated, and often opaque. Developmental interdependencies of SoS create layered networks that often span among a hierarchy of levels (DeLaurentis et al., 2008; Butler, 2001; Ayyalasomayajula, DeLaurentis, Moore & Glickman, 2008; Kotegawa, DeLaurentis, Sengstacken & Han, 2008). The complexity of these networks often hides many of the otherwise explicit consequences of risk. Depending on the network topology characteristics, disruptions to one of the critical nodes or links in the network can propagate through the network and result in degradation to seemingly distant nodes (Huang, Behara & Hu, 2008).

In this study, we express risk as a density function that describes the probability of a disruption occurring at any time during the system development. We concentrate on the *Implementation* and *Integration Phase* as the development stage in which disruptions occur. Here, inherent risk is the probability of disruptions due to the development characteristics of the subject system, e.g., technology readiness-level, funding, politics, etc. Risk due to interdependencies, on the other hand, is the probability of disruptions during the *Implementation Phase* of a system due to disruption in the system on which the system of interest depends. This is essentially the conditional probability of a disruption given that another system has a disruption.

This study assumes that the inherent risk of a system *i* in satisfying requirement *r*, $R_{sys}(i,r)$, is solely a function of its readiness-level, $m(i,r)$. While a somewhat simplified definition, expressing risk as a function of a system's readiness-level is logical. Recall that readiness-level is a metric that describes the necessary development of a system to satisfy a given requirement. Therefore, risk changes as the readiness-level of a system increases. Equation 1 introduces a relationship between a system's readiness-level and risk (probability of disruption).

$$R_{sys}(i,r) = \alpha_i \left(1 - m(i,r)^{\beta_i}\right)$$

Equation 1

In this relationship, α_i (with a value between 0 and 1) is parameter that indicates the upper bound value of risk for system *i* (i.e., producing maximum probability of disruption), while β_i is a shape parameter that indicates how quickly risk changes as a function of readiness-level. This formulation implies that risk is highest at the early stages of development (e.g., low readiness-levels) and it decreases (at different rates depending on the value of the β_i parameter) as development progresses. For instance, when a system *i* has a readiness-level of 0.0—it is a brand new system—the probability of disruptions during development will be highest, and it will have a value α_i. However, when the system has a readiness-level of 1.0, the probability of disruptions will be 0. System inherent-risk is implemented in the CEM by using a uniform random distribution to select a value between 0 and 1 at each time-step of the *Implementation* or *Integration Phase* and passing it into a binary channel to see if the number is smaller or greater than the probability of disruption defined by $R_{sys}(i,j)$. This determines if a disruption occurs or not.

When all systems are independent, identification of the system with highest risk is trivial (e.g., system that, on average, will contribute more to delays in completion time). However, when systems are interdependent, systems that otherwise have a low inherent risk can be greatly impacted by disturbances because of the transmission of risk from other systems. Systems are impacted by nearest neighbors (those systems on which they directly depend; first-order dependencies) and by systems that impact those nearest neighbors (higher-order dependencies).

The CEM models risk due to interdependencies in terms of the dependency strength between two given systems. Dependency strength, $S(i,j)$, is an input parameter that takes values between 0 and 1 and is defined as the conditional probability (uniform random probability) that system *i* has a disruption given that system *j* (on which system *i* depends) has a disruption. Risk due to interdependencies is, therefore, a function of the readiness-level of the dependent-upon system as well as the strength of that dependency. A notional example of a simple SoS is utilized here to present these features of the CEM (**Error! Reference source not found.**).

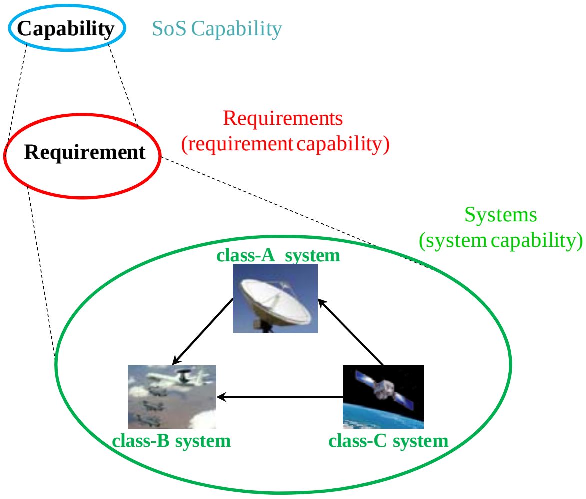
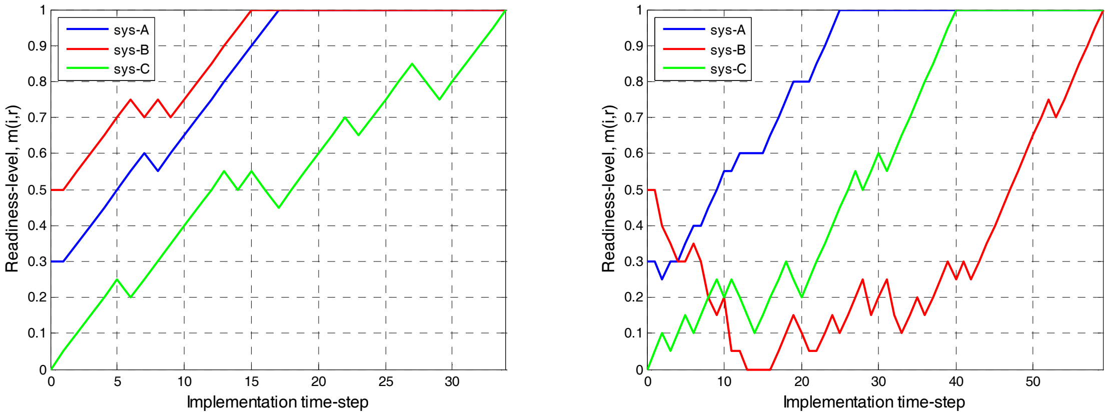


Figure 2. Layered Network Structure of Example SoS

Each system in this simple SoS network serves a role and provides a certain level of capability in order to satisfy some requirement. The links between systems indicate interdependencies among systems. The arrows indicate the directionality of dependence, including the case of mutual dependence. Mane and DeLaurentis (2009) contain more detailed information on the CEM structure. For this example, **Error! Reference source not found.** presents the implementation history of this three-system SoS with a risk profile that has α_i, and β_i, values of 0.2 and 4, respectively, and two different levels of interdependency strength, $S(i,j)$.



a) $S(i,j) = 0$ b) $S(i,j) = 1$

Figure 3. Implementation Phase History for Example Problem

Each system has a different initial readiness-level—system-A of 0.3, system-B of 0.5, and system-C of 0. Recall that an initial readiness-level greater than zero indicates a legacy system that must be further developed to achieve a readiness level of 1 to satisfy a given requirement. The model assumes that the readiness-level of a system can reduce to below initial readiness-level value. This is reasonable since inherent disruptions or disruptions due to interdependencies can result in modifications to subsystems that were not previously considered (i.e., unforeseen technology limitations of a system may require redesign of a dependent system). In **Error! Reference source not found.**a, all systems are independent (dependency strength of zero). The occasional set-backs in the readiness-level of each system are due to disruptions stemming from the inherent system risk. In **Error! Reference source not found.**b, one the other hand, dependency strength is highest (with a value of one). Recall that dependency strength indicates the probability of disruption on the dependent system given that the system on which it depends has a disruption. When the dependency strength is one, a disruption in a given system is always propagated to the dependent systems. For example, disruptions in the development of system-C propagate to system-A with probability 1 and disruptions in the development of system-A propagate to system-B with probability 1. Note, for instance, that there is a reduction in readiness-level in the development of the system-B every time that there is a reduction in readiness-level during the development of system-A or system-C (on which system-B depends). The candidate systems for a desired capability can, in general, have different levels of dependency strengths. **Error! Reference source not found.** presents a sensitivity of development time for this example problem on the value of dependency strength.

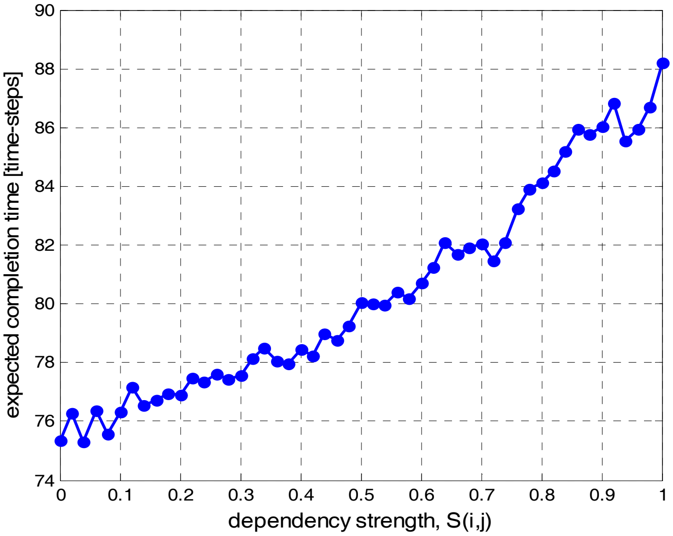


Figure 4. Impact of Dependency Strength on Completion Time for Example Problem

As expected, higher dependency strength means higher development time. In this example, the number of systems and interdependencies is invariable, and the increase in development time can be different for a different family of constituent systems. When considering the development of different families of systems that can provide a desired capability, the characteristics of interdependencies between component systems can have a large impact on the decision to pursue development of a certain alternative. Quantifying the impact that such characteristics have on the development process can aid decision-makers in selecting the most promising alternative.

Impact of Risk and System Interdependencies

Quantifying risk is a complicated function of the individual system characteristics as well as the interdependencies between systems. The combinations of systems that can achieve a given capability-level can be numerous. Depending on the selection of the constituent systems, the completion time of a project can vary greatly due to the number of constituent systems, their interdependencies, and risk profiles. As these families of systems get larger, it becomes more difficult to quantify the impact that each system and system-characteristic has on the success of a project. For instance, a three-system solution may appear to be preferable to a ten-system solution; but the interactions between the three systems can result in disruption propagation that greatly impacts the timely completion of the project. System interdependencies and their characteristics can impact the completion time of a project by affecting the way in which disruption propagate. In this section, we demonstrate the impact that system-inherent risk and the strength of interdependencies between component systems can have on the timely completion of a project. Furthermore, we show and quantify how different families of systems that can provide the same set of capabilities can have greatly differing development histories.

Interdependency Strength and Inherent Risk

For this investigation, we assume that in order to achieve some capability a family of three classes of systems has been identified; for instance, a class-A system can be a land-based radar or an airborne radar; a class-B system can be a large transport aircraft, a mid-size aircraft, or a small aircraft. Each of these classes of systems provides a certain capability that is required to achieve a global capability of the SoS. The design authority must decide which constituent system to select for each system-class. A notional example of a simple SoS is utilized here (**Error! Reference source not found.**).

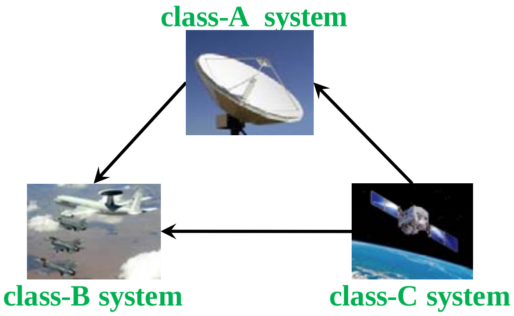
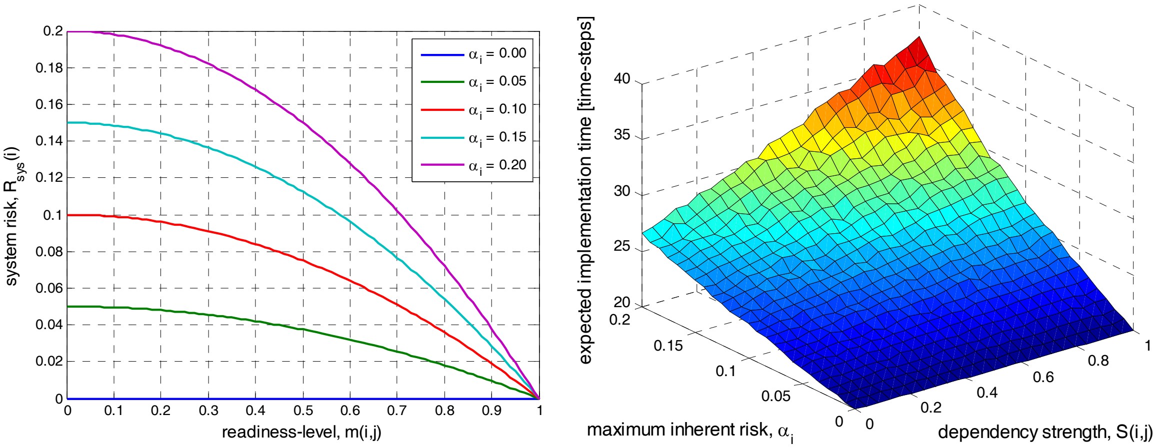


Figure 5. Interdependencies of Notional SoS

The links between systems indicate interdependencies among systems. For instance, development of a class-B system must rely on information about the development and capabilities of a class-A system in order to continue development. Similarly, development of a class-A system needs information from a class-C system. Different systems are available to designers or systems engineers for each system-class. Each candidate system can have different risk characteristics as well as different interdependency characteristics. If we assume that the systems engineer has identified these characteristics for each candidate system, then we can use the CEM to simulate the development process when different combinations of these candidate systems are considered and identify the family of systems that results in the lowest expected completion time. The strength of the CEM is in its ability to aggregate the individual system characteristics and quantify the SoS-level performance (with respect to development time) of a family of candidate systems.

Error! Reference source not found. presents results in which the expected implementation time of a family of candidate systems is measured against the inherent risk of individual systems and their interdependency strengths.



a) System-specific risk profile b) Expected implementation time

Figure 6. Impact of Risk Due to Interdependencies on Implementation Time

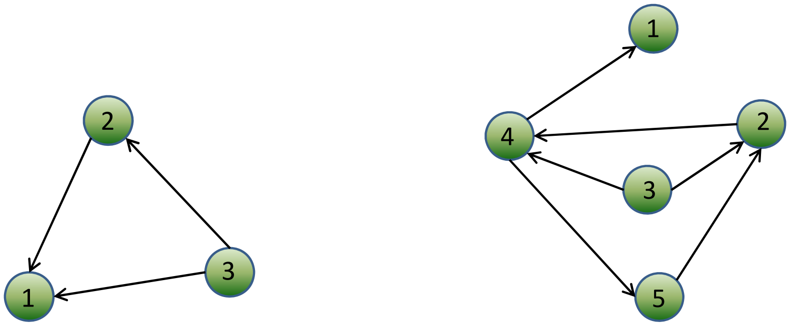
We assume here that all candidate systems will have the same risk profile and all interdependencies will have the same strength. **Error! Reference source not found.**a shows the inherent system risk, $R_{sys}(i,r)$, as a function of system readiness-level, $m(i,r)$, for five different risk profiles (five different α_i values and a fixed β_i parameter of 2). The value of α_i indicates the maximum inherent risk of a system, according to Equation 1. The assumption here is that risk is highest in the earlier stages of development and that it decreases as development progresses. The results in **Error! Reference source not found.**b present the expected implementation time when families of systems with different combinations of inherent risk profile and dependency strengths are considered. Each point on the surface indicates a family of candidate systems with a given combination of maximum inherent risk and dependency strengths. For instance, a solution that entails systems with a maximum inherent risk of zero and dependency strength of zero (e.g., independent systems with no development risk) will have an expected implementation time of 20 time units. The three systems are developed simultaneously but have no impact on each other's development. The trends in **Error! Reference source not found.**b show that the impact on implementation time of families of systems that have strong interdependencies is larger than when the systems have high inherent risk but low dependency strengths (e.g., the increase in implementation time is smaller as inherent risk increases than when the strength of dependencies increases).

This investigation quantifies the impact that system interdependencies have on the implementation time of a project. The results presented here point out the importance of interdependencies in the development process. This type of analysis can prove useful to an SoS authority when selecting potential component systems as a part of a family of systems or SoS to satisfy a given requirement and achieve a desired capability.

This simple example considers families of systems comprised of three constituent systems. Different candidate families of systems, however, can have differing number of constituent systems that can provide different system-capabilities to achieve the desired SoS capability. Similarly, risk profile and interdependency characteristics of the constituent systems can result in different disruption propagation and different development solutions.

Comparison of Alternatives

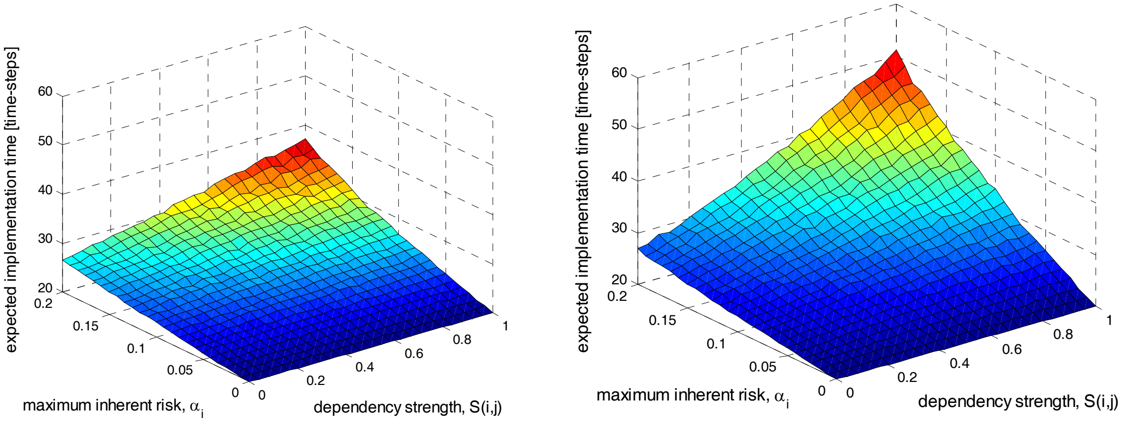
Given a set of alternative means to satisfy a requirement, an SoS authority (in conjunction with systems engineers) must determine the best network of systems to develop and acquire. The number of systems alone may not be a good indicator of the complexity of a system and the eventual developmental success. The risk profile of systems as well as the number and strength of system interdependencies play an important role that often hamper understanding of the impact of decisions. For instance, an SoS that is comprised of three constituent systems may appear more likely to succeed than an SoS comprised of five systems. However, the number and strength of interdependencies between the five systems may be such that the expected completion time of this SoS is lower than the expected completion time of the three-system SoS. The three-system example in the previous section showed that the strength of dependencies plays an important role in the timely completion of an SoS project. Here we use the CEM to investigate the impact that network characteristics (number of systems, number of dependencies, and strength of dependencies) have on the completion time of an SoS project. We compare the developmental time of two example SoSs comprised of three and five constituent systems (**Error! Reference source not found.**).



a) Three-system alternative b) Five-system alternative

Figure 7. Alternative Families of Systems

The three-system network is the same network with three interdependencies as the one presented in **Error! Reference source not found.**. The new, five-system network is clearly a larger SoS with more systems and six interdependencies. As in the previous section, different candidate systems are available to provide the required capability level. The systems engineer would like to quantify the expected implementation time of each combination of systems for the three-system and the five-systems options. Via a Monte Carlo simulation of 500 samples, we are able to compute the expected implementation time of the five-system network—we previously did the same for the three-system network. **Error! Reference source not found.** presents this result for the different combinations of inherent system risk and dependency strengths.



a) Three-system alternative

b) Five-system alternative

Figure 8. Expected Implementation Time of Alternatives

Error! Reference source not found.a presents the expected implementation time of the three-system option (the same as **Error! Reference source not found.**b), while **Error! Reference source not found.**b presents the expected implementation time of the five-system network. As in the previous analysis, these results indicate the expected implementation time of candidate component systems that have differing levels of inherent risk and interdependency strengths. The trends in the expected implementation time of the five-system option are larger than those of the three-system option. This is expected because the former has more systems as well as more interdependencies. Recall, however, that each point in these charts represents a candidate family of systems and one can see that the expected implementation times of some five-system alternatives are lower than some three-system alternatives. To show this more clearly, **Error! Reference source not found.** presents the expected completion times when the inherent system risk of all candidate systems is highest (α = 0.2).

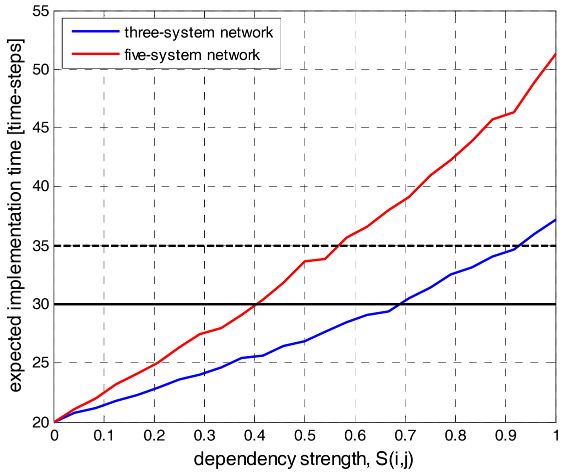


Figure 9. Expected Implementation Time of Sample Results

As previously mentioned, the implementation time of the five-system alternatives is always higher than the three-system alternatives. However, if the dependency strength between the systems in the three-system alternative has a value of 1, then the expected implementation time of this alternative will be 37 time units; if the dependency strength between the systems in the five-system is not as strong, say with a value of 0.4, then the expected implementation time will be 30 time units. Therefore, depending on the strength of the interdependencies between the constituent systems, a family of systems can be a better (lower expected implementation time) alternative. By simulating the development process of different alternatives via the CEM, it is possible to quantify the impact of system specific risk, the risk due to interdependencies, and the propagation of disruptions to compare different alternative solutions that can provide a desired level of capability.

Analytical Approach to Measure Delay Propagation

Additional complexity in the model, carefully selected, will likely increase the efficacy of the CEM. However, as a simulation-based approach, it too has limitations. Therefore, in conjunction with the further development of the CEM, the authors are also developing an analytical approach that captures the characteristics of a network that results from the developmental interdependencies of systems. This is an approach that uses a network-level metric to treat the same complexities via computations on conditional probabilities that relate the transmission of risk in networks of interdependent systems. This provides means to compare networks in their ability to arrest the propagation of delays caused by random disturbances and can be used as a figure of merit when designing SoS architectures that aim to achieve some desired capability. While typical networks like the World Wide Web, social networks, and communication networks are a result of evolution, some networks of military systems created for particular purposes can be designed. The ability to quantify the performance of SoS networks enables comparison of networks, and ultimately the design of superior SoS networks that optimize that performance.

The proposed approach to measure the performance of networks in their ability to arrest the propagation of delays is based on the “lost miner problem” (Ross, 2007). In this example problem, a miner is lost in a cave inside a mine and there are four tunnels that lead out of the cave, but only one leads out of the mine (**Error! Reference source not found.**).

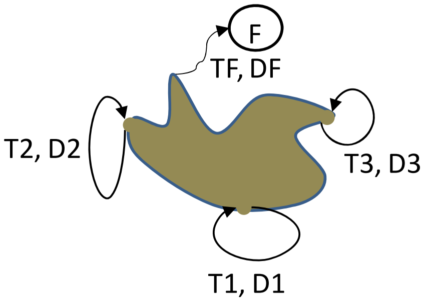


Figure 10. Lost-miner Problem

The miner can choose to enter a tunnel T_i with probability $P(T_i)$ and has no memory of his previous choice. If the miner chooses tunnel T_1, then he wanders in the tunnel for D_1 days and returns to the cave, where he must decide which tunnel to enter next. If he chooses tunnel T_2 or T_3, then he wanders in the tunnel for D_2 or D_3 days, respectively, only to return to the cave. If he chooses tunnel T_F, then he is free, instantly. The question the problem poses is: What is the expected time until the miner reaches freedom (e.g., the expected duration of the miner's stay in the mine)?

Following this reasoning, we can describe the delay propagation in the system development process in a similar manner. We describe a network of systems in terms of the number of systems (caves), the number and direction of their dependencies (tunnels), and the characteristics of the interdependencies (probability of choosing a given tunnel), e.g., probability of passing on a disruption and the impact of the disruption. The simple three-system network below is used to describe the proposed approach.

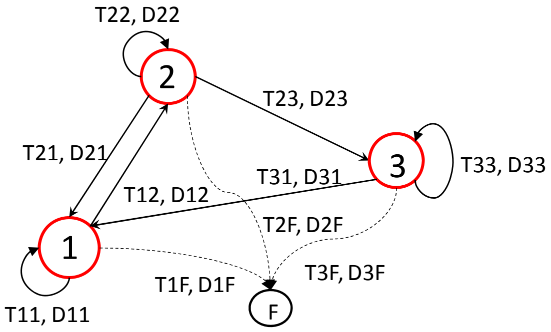


Figure 11. Example Systems Development Network

Each node represents a system that is under development (i.e., aircraft, missile, radio) to achieve some capability. The links indicate interdependencies between the systems as well as the strength of those interdependencies. For instance, system-1 depends on system-3 because information from system-3 is needed to continue development of system-1. T_{ij} represents the conditional probability that a disruption in the development of system *i* will impact development of system *j* and D_{ij} represents the impact of a disruption (delay) on system *i* that propagates to system *j*. These two quantities represent the strength of the dependency between system *i* and system *j*. Two systems can be strongly dependent if the probability of a disruption propagating from one system to the other is high, or if the delay experienced by one system because of a disruption in the development of the other is large. Node F is a sink that represent the arrest of an event and its propagation in the network. In this setting, a disruption can be seen as an event that travels from system to system causing developmental delays until it exits the system/network (via node F). This is similar to the "lost miner problem," in which the miner chooses tunnels until he reaches freedom.

In system development, disruptions can be a result of funding decisions, political environment, technological setbacks, etc. For example, system-1 can be faced with budget cuts and the program manager must reduce funding to one of the subsystems that comprise system-1. Depending on the magnitude of the reduction in funding, this can have no impact in the development of system-1 with probability T_{1F}, and nothing is affected; it can cause a delay of D_{11} days with probability T_{11} in the development of system-1 that is not large enough to impact interdependent systems; or it can result in a delay of D_{12} days with probability T_{12} that impacts development of system-2. Additionally, the delay in the development of system-2 can cause further problems that delay its development by D_{22} days with probability T_{22}; it can cause a delay of D_{23} days with probability T_{23} that creates a problem in the development of system-3; or a delay of D_{21} days with probability T_{21} that impacts system-1; or, conversely, the problem is not large enough to cause any delays with probability T_{2F}, and the propagation of delay in the network is arrested.

Depending on the strength of the dependencies between systems and the magnitude of disruptions, delays can propagate and accumulate in a network. Hence, networks with different number of systems, interdependencies, and strength of interdependencies will perform differently when faced with random disruptions. The analytical approach now under pursuit may be able to estimate the expected accumulation of delays as a function of these network characteristics. The network-level metric can enable the design of networks that minimize expected delay whenever random events hit the development process of individual systems.

Conclusions

The development of complex systems (and SoS) is beset by risk. Risk analyses of individual systems can explain the threats and opportunities of systems, but do not capture the impact that disruptions to individual systems have at the enterprise level, where multiple systems—explicitly or implicitly interdependent—collaborate to achieve various capabilities. An understanding of risk in the development and acquisition process and its cascading effects is crucial to identifying means to exploit opportunities, as well as mitigate, transfer, or avoid disruptions.

These research efforts center on the ongoing development of a Computational Exploratory Model that is based on the processes in the *SoS-SE Guidebook* and that estimates time to complete an SoS integration. The extensions to the model in this paper capture the impact of individual system risk and number and strength of system interdependencies on the propagation of developmental disruptions and, ultimately, the timely completion of a project. In particular, the present work examined changes in the systems interdependencies and system risk profiles (i.e., different inherent risk profiles for different systems and different dependency strength) when alternative systems are considered for satisfying a given requirement and providing a certain capability. Examples of alternative families of systems comprised of a different number of constituent systems showed that the number of constituent systems and their risk profiles are insufficient to quantify the development performance of SoS. The sample analyses presented here showed that these characteristics, coupled with the interdependency characteristics of a family of systems, can result in expected implementation times that are not easily foreseen.

When coupled with the theoretical basis of delay propagation and a network-level metric that describes the expected delay in a family of systems the methodology presented here can improve/facilitate the decision-making process of systems engineers and system integration as well as provide a means to design system architectures that aim to minimize delay propagation and development time.

Acknowledgments

The authors acknowledge the support for this work from a grant from the Naval Postgraduate School's Acquisition Research Program (N00244-09-1-0011).

References

Ayyalasomayajula, S., DeLaurentis, D.A., Moore, G.E., & Glickman, L.T. (2008, October). A network model of H5N1 Avian Influenza transmission dynamics in domestic cats. *Zoonoses and Public Health*, *55*(8-10), 497-506.

Brown, M., & Flowe, R. (2005). Joint capabilities and systems of systems solutions: A multidimensional approach to understanding cost drivers. *Defense Acquisition Review Journal*, *12*(2), 138-154.

Butler, J.T. (2001, April). *UAVs and ISR sensor technology (Research Report)*. Air Command and Staff College, Air University, AU/ACSC/033/2001-04. Retrieved January 26, 2010, from https://research.maxwell.af.mil/papers/ay2001/acsc /01-033.pdf

Charles, P., & Turner, P. (2004). Capabilities based acquisition…From theory to reality. *CHIPS Magazine*. Retrieved January 21, 2010, from http://www.chips.navy.mil/archives/04_summer /web_pages/GEMINII.htm

Committee on Systems Integration for Project Constellation. (2004). *Systems integration for project constellation*. Retrieved January 15, 2009, from The National Academies, http://nap.edu/html/proj-constella tion/ltr-rep.pdf

Constantine, J., & Solak, S. (2010). SysML modeling of off-the-shelf-option acquisition for risk mitigation in military programs. *Systems Engineering*, *13*(1), 80–94.

DeLaurentis, D., Han, E-P., & Kotegawa, T. (2008). Network—Theoretic approach for analyzing connectivity in air transportation networks. *AIAA Journal of Aircraft*, *45*(5), 1669-1679.

DoD. (2008a). *Defense acquisition guidebook*. Retrieved April 4, 2008, from https://akss.dau.mil/ dag/

DoD. (2008b). *Systems engineering guide for system-of-systems*. Retrieved March 2, 2008, from http://www.acq.osd.mil/sse/ssa/initiat_sos-se.html\

Durkac, L.M. (2005, March). *Joint mission capability packages: The future of joint combat*. Presented at the 10th International Command and Control Research and Technology Symposium. Retrieved January 25, 2010, from online database, http://www.dodccrp.org/events/10th_ICCRTS/CD/papers/063.pdf

Francis, P. (2007, April). *Defense acquisitions: Missile Defense Agency's flexibility reduces transparency of program cost*. Testimony before the Subcommittee on Defense, Committee on Appropriations, US House of Representatives. Washington, DC: Government Accountability Office. Retrieved January 27, 2010, from www.gao.gov/new.items /d07799t.pdf

Gilmore, M.J. (2006, April). *The Army's Future Combat System Program*. CBO Testimony. Retrieved March 3, 2010, from http://www.cbo.gov/ftpdocs/71xx/doc7122/04-04-FutureCombatSystems.pdf

Huang, C.D., Behara, R.S., & Hu, Q. (2008). Managing risk propagation in extended enterprise networks. *IT Professional*, *10*(4), 14-19.

Kotegawa, T., DeLaurentis, D.A., Sengstacken, A., & Han, E-p. (2008, September). *Utilization of network theory for the enhancement of ATO air route forecast*. Anchorage, AK: 8th AIAA Aviation Technology, Integration, and Operations.

Maier, M. (1998). Architecting principles for system-of-systems. *Systems Engineering*, *1*(4), 267-284.

Mane, M., & DeLaurentis, D. (2009, May). Acquisition management for systems of systems: Exploratory model development and experimentation. In *Proceedings of the Sixth Annual Acquisition Research Symposium*. Monterey, CA: Naval Postgraduate School.

NextGen Integration and Implementation Office. (2009). *NextGen implementation plan 2009*. Washington, DC: Federal Aviation Administration. Retrieved September 25, 2009, from http://www.faa.gov/about/initiatives/nextgen/media/ NGIP_ 0130.pdf.

Ross, M.S. (2007). *Introduction to probability models* (9th ed.). Burlington, MA: Academic Press.

Rouse, W. (2001). Complex engineered, organizational and natural systems. *Systems Engineering*, *10*(3), 260-271.

Saunders, T., et al. (2005). *Report on system-of-systems engineering for Air Force capability development* (SAB-TR-05-04). Washington, DC: USAF Scientific Advisory Board.

Sauser, B., Verna, D., Ramirez-Marquez, J., & Gove, R. (2006). From TRL to SRL: The concept of systems readiness levels. In *Proceedings of the Conference on Systems Engineering Research*. Los Angeles, CA: CSER.

Spacy, W.L., II. (2004). Capability-based acquisition in the Defense Agency and implications for Department of Defense acquisition. *Journal of Contract Management*, 10-19.

USD(AT&L). (2005). *The defense acquisition system* (DoDD 5000.2). Chapter 4: Systems Engineering. Washington, DC: Author

2003 - 2010 Sponsored Research Topics

Acquisition Management

- Acquiring Combat Capability via Public-Private Partnerships (PPPs)
- BCA: Contractor vs. Organic Growth
- Defense Industry Consolidation
- EU-US Defense Industrial Relationships
- Knowledge Value Added (KVA) + Real Options (RO) Applied to Shipyard Planning Processes
- Managing the Services Supply Chain
- MOSA Contracting Implications
- Portfolio Optimization via KVA + RO
- Private Military Sector
- Software Requirements for OA
- Spiral Development
- Strategy for Defense Acquisition Research
- The Software, Hardware Asset Reuse Enterprise (SHARE) repository

Contract Management

- Commodity Sourcing Strategies
- Contracting Government Procurement Functions
- Contractors in 21st-century Combat Zone
- Joint Contingency Contracting
- Model for Optimizing Contingency Contracting, Planning and Execution
- Navy Contract Writing Guide
- Past Performance in Source Selection
- Strategic Contingency Contracting
- Transforming DoD Contract Closeout
- USAF Energy Savings Performance Contracts
- USAF IT Commodity Council
- USMC Contingency Contracting

Financial Management

- Acquisitions via Leasing: MPS case
- Budget Scoring
- Budgeting for Capabilities-based Planning

- Capital Budgeting for the DoD
- Energy Saving Contracts/DoD Mobile Assets
- Financing DoD Budget via PPPs
- Lessons from Private Sector Capital Budgeting for DoD Acquisition Budgeting Reform
- PPPs and Government Financing
- ROI of Information Warfare Systems
- Special Termination Liability in MDAPs
- Strategic Sourcing
- Transaction Cost Economics (TCE) to Improve Cost Estimates

Human Resources

- Indefinite Reenlistment
- Individual Augmentation
- Learning Management Systems
- Moral Conduct Waivers and First-tem Attrition
- Retention
- The Navy's Selective Reenlistment Bonus (SRB) Management System
- Tuition Assistance

Logistics Management

- Analysis of LAV Depot Maintenance
- Army LOG MOD
- ASDS Product Support Analysis
- Cold-chain Logistics
- Contractors Supporting Military Operations
- Diffusion/Variability on Vendor Performance Evaluation
- Evolutionary Acquisition
- Lean Six Sigma to Reduce Costs and Improve Readiness
- Naval Aviation Maintenance and Process Improvement (2)
- Optimizing CIWS Lifecycle Support (LCS)
- Outsourcing the Pearl Harbor MK-48 Intermediate Maintenance Activity
- Pallet Management System
- PBL (4)
- Privatization-NOSL/NAWCI
- RFID (6)

- Risk Analysis for Performance-based Logistics
- R-TOC AEGIS Microwave Power Tubes
- Sense-and-Respond Logistics Network
- Strategic Sourcing

Program Management

- Building Collaborative Capacity
- Business Process Reengineering (BPR) for LCS Mission Module Acquisition
- Collaborative IT Tools Leveraging Competence
- Contractor vs. Organic Support
- Knowledge, Responsibilities and Decision Rights in MDAPs
- KVA Applied to AEGIS and SSDS
- Managing the Service Supply Chain
- Measuring Uncertainty in Earned Value
- Organizational Modeling and Simulation
- Public-Private Partnership
- Terminating Your Own Program
- Utilizing Collaborative and Three-dimensional Imaging Technology

A complete listing and electronic copies of published research are available on our website:
www.acquisitionresearch.org

THIS PAGE INTENTIONALLY LEFT BLANK

ACQUISITION RESEARCH PROGRAM
GRADUATE SCHOOL OF BUSINESS & PUBLIC POLICY
NAVAL POSTGRADUATE SCHOOL
555 DYER ROAD, INGERSOLL HALL
MONTEREY, CALIFORNIA 93943

www.acquisitionresearch.org