Flexible and Intelligent Learning Architectures for SoS (FILA-SoS)

Volume 6 – Non-Cooperative System Negotiation Model

Technical Report SERC-2015-TR-021-4

February 28, 2015

Principal Investigators

Dr. Cihan H. Dagli, Missouri University of Science and Technology

Research Team

Co-PI: Dr. David Enke, Missouri S&T

Co-PI: Dr. Nil Ergin, Penn State University

Co-PI: Dr. Dincer Konur, Missouri S&T

Co-PI: Dr. Ruwen Qin, Missouri S&T

Co-PI: Dr. Abhijit Gosavi, Missouri S&T

Dr. Renzhong Wang

Missouri S&T Graduate Students

Louis Pape II, Siddhartha Agarwal and Ram Deepak Gottapu

This material is based upon work supported, in whole or in part, by the U.S. Department of Defense through the Systems Engineering Research Center (SERC) under Contract H98230-08-D-0171 (Task Order 033, RT 48). SERC is a federally funded University Affiliated Research Center managed by Stevens Institute of Technology

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

EXECUTIVE SUMMARY

Multi-faceted systems of the future will entail complex logic and reasoning with many levels of reasoning in intricate arrangement. The organization of these systems involves a web of connections and demonstrates self-driven adaptability. They are designed for autonomy and may exhibit emergent behavior that can be visualized. Our quest continues to handle complexities, design and operate these systems. The challenge in Complex Adaptive Systems design is to design an organized complexity that will allow a system to achieve its goals. This report attempts to push the boundaries of research in complexity, by identifying challenges and opportunities. Complex adaptive system-of-systems (CASoS) approach is developed to handle this huge uncertainty in socio-technical systems.

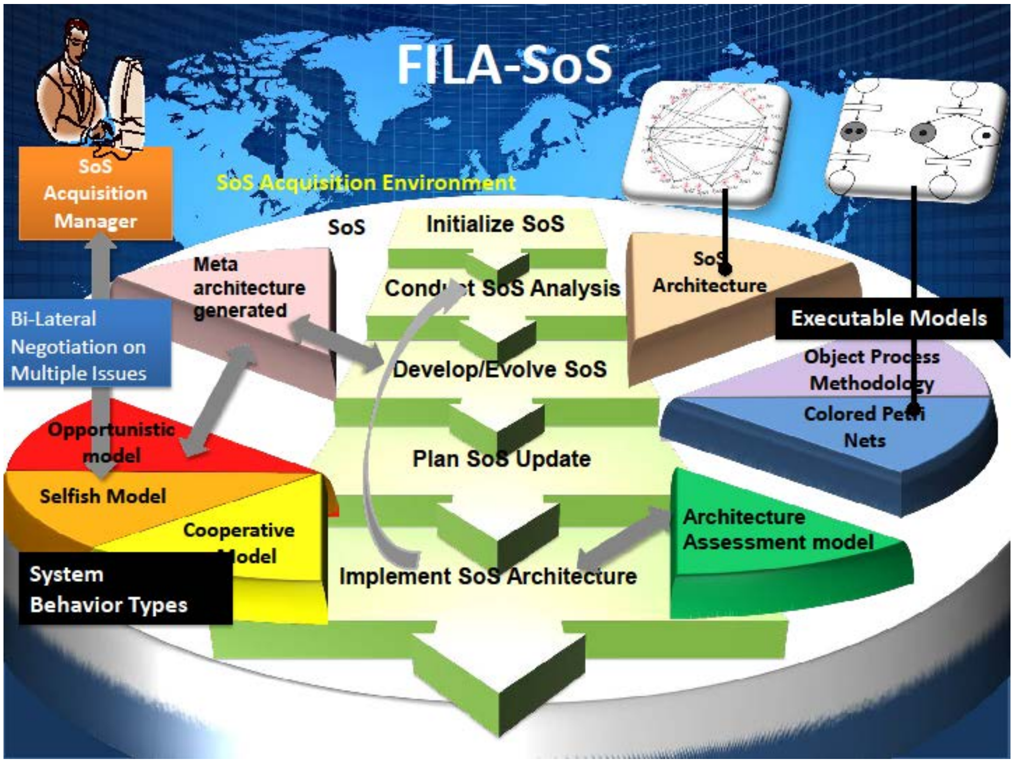
Although classically (Dahmann, Rebovich, Lowry, Lane, & Baldwin, 2011) four categories of SoS are described in literature namely; Directed, Collaborated, Acknowledged and Virtual. However, there exist infinitely many SoS on the edges of these categories thus making it a continuum. Many SoS with different configurations can fill this gap. These four types of SoS vary based on their degree of managerial control over the participating systems and their structural complexity. The spectrum of SoS ranges from Directed SoS that represents complicated systems to Virtual SoS that are complex systems.

Acknowledged SoS lie in between this spectrum. This particular SoS is the focal point of our research endeavor. Acknowledged SoS and Directed SoS share some similarities such as both have (Dahman & Baldwin, 2011) SoS objectives, management, funding and authority. Nevertheless, unlike Directed SoS, Acknowledged SoS systems are not subordinated to SoS. However, Acknowledged SoS systems retain their own management, funding and authority in parallel with the SoS. Collaborative SoS are similar to Acknowledged SoS systems in the fact that systems voluntarily work together to address shared or common interest.

Flexible and Intelligent Learning Architectures for SoS (FILA-SoS) integrated model is developed in this research task provides a decision making aid for SoS manager based on the wave model. The model developed called the FILA-SoS does so using straightforward system definitions methodology and an efficient analysis framework that supports the exploration and understanding of the key trade-offs and requirements by a wide range system-of-system stakeholders and decision makers in a short time. FILA-SoS and the Wave Process address four of the most challenging aspects of system-of-system architecting:

1. Dealing with the uncertainty and variability of the capabilities and availability of potential component systems
2. Providing for the evolution of the system-of-system needs, resources and environment over time
3. Accounting for the differing approaches and motivations of the autonomous component system managers
4. Optimizing system-of-systems characteristics in an uncertain and dynamic environment with fixed budget and resources

Some of the highlights of FILA-SoS are listed in terms of its capabilities, value added to systems engineering, ability to perform "What-if Analysis", modularity of integrated models, its potential applications in the real world and future additions to the current version.



FILA-SoS has a number of unique capabilities such as integrated model for modeling and simulating SoS systems with evolution for multiple waves. It also has modularity in the structure where the models can be run independently and in conjunction with each other. Besides there are a couple of different models for both architecture generation and SoS behavior and various individual system behavior negotiation models between SoS and individual systems. In terms of value added FILA-SoS aids the SoS manager in future decision making. It also helps in understanding the emergent behavior of systems in the acquisition environment and impact on SoS architecture quality. FILA-SoS serves as an artifact to study the dynamic behavior of different type of systems (non-cooperative, semi-cooperative, cooperative). It enables us to identify intra and interdependencies among SoS elements and the acquisition environment. FILA-SoS can provide a "What-if" Analysis depending on variables such as SoS funding and capability priority that can be changed as the acquisition progresses through wave cycles. It has the ability to simulate any architecture through colored petri nets. In addition, it can simulate rules of engagement & behavior settings: all systems are non-cooperative, all systems are semi-cooperative, and all systems are cooperative or a combination. Some of the potential applications include modeling a wide variety of complex systems models such as logistics, and cyber-physical systems. It also acts as a test-bed for decision makers to evaluate operational guidelines and principles for managing various acquisition environment scenarios. Future Capabilities that are currently in progress are extending the model to include multiple interface alternatives among systems and incorporation of risk models into environmental scenarios.

Integrated Model Structure for FILA-SoS Version 1.0 is described. It provides a short description of all independent models that make up the FILA-SoS integrated model and reports the workings of the model with three notional System-of-Systems namely; Toy Problem for aircraft carrier performance assessment, ISR (intelligence surveillance and reconnaissance) and SAR (search and rescue).

The project reports span 17 volumes. Each report describes the various aspects of the FILA-SOS integrated model:

Volume 1: Integrated Model Structure

Volume 1 is the Integrated Model Structure report for FILA-SoS Version 1.0. It provides a short description of all independent models that make up the FILA-SoS integrated model. Integrated FILA-SoS developed is tested in three notional System-of-Systems namely; Toy Problem for Aircraft Carrier Performance Assessment, ISR (intelligence surveillance and reconnaissance) and SAR (search and rescue). FILA-SoS integrated model is currently being validated with a real life data from a medium sized SoS. The results of this validation are given in volume 17.

Volume 2: Meta-Architecture Generation Multi-Level Model

Volume 2 describes Meta-Architecture Generation Multi-Level Model. The multi-level meta-architecture generation model considers constructing an SoS architecture such that each capability is provided by at least one system in the SoS and the systems in the SoS are able to communicate with each other. Secondly, it has multiple objectives for generating a set of SoS architectures namely; maximum total performance, minimum total costs and minimum deadline. Finally, the model establishes initial contracts with systems to improve performances.

Volume 3: Fuzzy-Genetic Optimization Model

Volume 3 illustrates the second meta-architecture generation model known as the Fuzzy-Genetic optimization model. This model is based on evolutionary multi-objective optimization for SoS architecting using genetic algorithms and four key performance attributes (KPA) as the objective functions. It also has a type-1 fuzzy assessor for dynamic assessment of domain inputs and that forms the fitness function for the genetic algorithm. It returns the best architecture (meta-architecture) consisting of systems and their interfaces. It is a generalized method with application to multiple domains such as Gulf War Intelligence/Surveillance/Reconnaissance Case, Aircraft Carrier Performance Assessment Case and Alaskan Maritime Search and Rescue Case.

Volume 4: Architecture Assessment Model

Volume 4 describes an Architecture Assessment Mode that can capture the non-linearity in key performance attribute (KPA) tradeoffs, is able to accommodate any number of attributes for a selected SoS capability, and incorporate multiple stakeholder's understanding of KPA's. Assessment is based on a given meta-architecture alternative. This is done using type-1 fuzzy sets and fuzzy inference engine. The model provides numerical values for meta-architecture quality.

Volume 5: Cooperative System Negotiation Model

Volume 5 specifically describes the Cooperative System Negotiation Model. The systems following this model behave cooperatively while negotiating with the SoS manager. The model of cooperative behavior is based on agent preferences and the negotiation length. Each system agent has two inherent behaviors of cooperativeness: Purposive (normal behavior) and Contingent (behavior driven by unforeseen circumstances). The approach models the tradeoff between the two behaviors for the systems. A fuzzy weighted average approach is used to arrive at the final proposed value.

Volume 6: Non-Cooperative System Negotiation Model

Volume 6 goes on to describe the Non-Cooperative System Negotiation Model in which systems behave in their self-interest while negotiating with the SoS coordinator. A mathematical model of individual system's participation capability and self-interest negotiation behavior is created. This methodology is an optimization-based generator of alternatives for strategically negotiating multiple items with multiple criteria. Besides, a conflict evaluation function that estimates prospective outcome for identified alternative is proposed.

Volume 7: Semi-Cooperative System Negotiation Model

Volume 7 describes the third and last system negotiation model, which illustrates the Semi-Cooperative System Negotiation Model. It exhibits the capability of being flexible or opportunistic: i.e., extremely cooperative or uncooperative based on different parameter values settings. A Markov-chain based model designed for handling uncertainty in negotiation modeling in an SoS. A model based on Markov chains is used for estimating the outputs. The work assigned by the SoS to the system is assumed to be a ``project'' that takes a random amount of time and a random amount of resources (funding) to complete.

Volume 8: Incentive based Negotiation Model for System of Systems

Volume 8 explains the SoS negotiation model also called the Incentive Based Negotiation Model for System of Systems. This model is based on two key assumptions that are to design a contract to convince the individual systems to join the SoS development and motivate individual systems to do their tasks well. Game theory and incentive based contracts are used in the negotiation model that will maximize the welfare for parties involved in the negotiation. SoS utility function takes into account local objectives for the individual systems as well as global SoS objective whereas the incentive contract design persuades uncooperative systems to join the SoS development.

Volume 9: Model for Building Executable Architecture

Volume 9 illustrates the process of building Executable Architectures for SoS. The operations of the SoS is a dynamic process with participating system interacting with each other and exchange various kinds of resources, which can be abstract information or physical objects. This is done through a hybrid structure of OPM (Object process methodology) and CPN (Colored petri nets) modeling languages. The OPM model is intuitive and easy to understand. However, it does not support simulation, which is required for accessing the behavior related performance. This is achieved by mapping OPM to CPN, which is an executable simulation language. The proposed method can model the interactions between components of a system or subsystems in SoS. In addition, it can capture the dynamic aspect of the SoS and simulate the behavior of the SoS. Finally, it can access various behavior related performance of the SoS and access different constitutions or configurations of the SoS which cannot be incorporated into the meta-architecture generation models of Volume 2 & 3.

Volume 10: Integrated Model Software Architecture and Demonstration FILA-SoS Version 1.0
Volume 10 elucidates the Integrated Model Software Architecture and Demonstration based on the models described above. Volume 11 and thereon the reports are aimed at the upcoming newer version 2.0 of FILA-SoS.

Volume 11: Integrated Model Structure FILA-SoS Version 2.0
Volume 11 provides Integrated Model Structure for FILA-SoS Version 2.0 that could be implemented in a new software environment.

Volume 12: Complex Adaptive System-of-System Architecture Evolution Strategy Model for FILA-SoS Version 2.0
Volume 12 provides a model to answer the first research question "What is the impact of different constituent system perspectives regarding participating in the SoS on the overall mission effectiveness of the SoS?" It is named the Complex Adaptive System-of-System Architecture Evolution Strategy Model and is incorporated in FILA-SoS Version 2.0. This volume describes a computational intelligence based strategy involving meta-architecture generation through evolutionary algorithms, meta-architecture assessment through type-2 fuzzy nets and finally its implementation through an adaptive negotiation strategy.

Volume 13: On the Flexibility of Systems in System of Systems Architecting: A new Meta-Architecture Generation Model for FILA-SoS Version 2.0
Volume 13 is termed the Flexibility of Systems in System of Systems Architecting: A new Meta-Architecture Generation Model for FILA-SoS Version 2.0. The research question is answered through an alternative technique to meta-architecture generation besides the one described in Volume 2.

Volume 14: Assessing the Impact on SoS Architecture Different Level of Cooperativeness: A new Model for FILA-SoS Version 2.0
Volume 14 proposes a new method for Assessing the Impact on SoS Architecture Different Level of Cooperativeness. Second research question is answered through a model that allows different levels of cooperativeness of individual systems.

Volume 15: Incentivizing Systems to Participate in SoS and Assess the Impacts of Incentives: A new Model for FILA-SoS Version 2.0
Volume 15 is an extension of previous systems negotiation models based on incentivizing and is aptly called Incentivizing Systems to Participate in SoS and Assess the Impacts of Incentives: A new Model for FILA-SoS Version 2.0. It also provides an approach to answer the third research question "How should decision-makers incentivize systems to participate in SoS, and better understand the impact of these incentives during SoS development and effectiveness?". This model is based on the fact that providing incentives only depending on the outcome may not be enough to attract the attention of the constituent systems to participate in SoS mission. Therefore, this model extends the approach as described in Volume 8 while considering the

uncertainty in the acquisition environment. The incentive contract is designed based on the objectives of the SoS and the individual systems. Individual system's objective is to secure highest incentives with minimal effort while the SoS manager's goal is to convince individual systems to join the SoS development while maximizing its own utility.

Volume 16: Integrated Model Software Architecture for FILA-SoS Version 2.0
Volume 16 gives an overview of the integrated model architecture in version 2.0 of the software. It includes all old and new models previously mentioned.

Volume 17: FILA-SoS Version 1.0 Validation with Real Data
Volume 17 describes the validation of the FILA-SoS Version 1.0 with a real life data provided by MITRE Corporation by from a moderately sized SoS.

INTRODUCTION

MOTIVATION FOR RESEARCH

In the real world, systems are complex, non-deterministic, evolving, and have human centric capabilities. The connections of all complex systems are non-linear, globally distributed, and evolve both in space and in time. Because of non-linear properties, system connections create an emergent behavior. It is imperative to develop an approach to deal with such complex large-scale systems. The approach and goal is not to try and control the system, but design the system such that it controls and adapts itself to the environment quickly, robustly, and dynamically. These complex entities include both socioeconomic and physical systems, which undergo dynamic and rapid changes. Some of the examples include transportation, health, energy, cyber physical systems, economic institutions and communication infrastructures.

In addition, the idea of "System-of-Systems" is an emerging and important multidisciplinary area. An SoS is defined as a set or arrangement of systems that results when independent and useful systems are integrated into a larger system that delivers unique capabilities greater than the sum of the capabilities of the constituent parts. Either of the systems alone cannot independently achieve the overall goal. System-of- Systems (SoS) consists of multiple complex adaptive systems that behave autonomously but cooperatively (Dahman, Lane, Rebovich, & Baldwin, 2008). The continuous interaction between them and the interdependencies produces emergent properties that cannot be fully accounted for by the "normal" systems engineering practices and tools. System of Systems Engineering (SoSE), an emerging discipline in systems engineering is attempting to form an original methodology for SoS problems (Luzeaux, 2013).

Since SoS grow in complexity and scale with the passage of time it requires architectures that will be necessary for understanding and governance and for proper management and control. Systems architecting can be defined as specifying the structure and behavior of an envisioned system. Classical system architecting deals with static systems whereas the processes of System of Systems (SoS) architecting has to be first done at a meta-level. The architecture achieved at a meta-level is known as the meta-architecture. The meta-architecture sets the tone of the architectural focus (Malan & Bredemeyer, 2001). It narrows the scope of the fairly large domain space and boundary. Although the architecture is still not fixed but meta-architecture provides multiple alternatives for the final architecture. Thus architecting can be referred to as filtering the meta-architectures to finally arrive at the architecture. The SoS architecting involves multiple systems architectures to be integrated to produce an overall large scale system meta-architecture for a specifically designated mission (Dagli & Ergin, 2008). SoS achieves the required goal by introducing collaboration between existing system capabilities that are required in creating a larger capability based on the meta-architecture selected for SoS. The level of the degree of influence on individual systems architecture through the guidance of SoS manager in implementing SoS meta-architecture can be classified as directed, acknowledged, collaborative and virtual. Acknowledged SoS have documented objectives, an elected manager and defined resources for the SoS. Nonetheless, the constituent systems retain their independent ownership, objectives, capital, development, and sustainment approaches. Acknowledged SoS shares some similarities with directed SoS and collaborative SoS. There are four types of SoS that are described below:

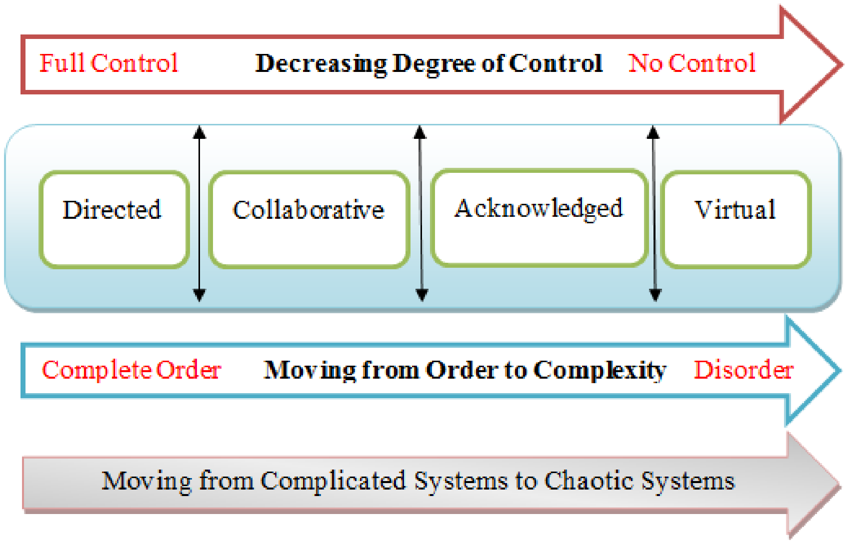


Figure 1 Schematic Drawing of Four Classical Types of SoS Based on Degree of Control and Degree of Complexity

Virtual
- Virtual SoS lack a central management authority and a centrally agreed upon purpose for the system-of-systems.
- Large-scale behavior emerges—and may be desirable—but this type of SoS must rely upon relatively invisible mechanisms to maintain it.

Collaborative
- In collaborative SoS the component systems interact more or less voluntarily to fulfill agreed upon central purposes.

Acknowledged (*FILA-SoS integrated model is based on Acknowledged SoS*)
- Acknowledged SoS have recognized objectives, a designated manager, and resources for the SoS; however, the constituent systems retain their independent ownership, objectives, funding, and development and sustainment approaches.
- Changes in the systems are based on collaboration between the SoS and the system.

Directed
- Directed SoS's are those in which the integrated system-of-systems is built and managed to fulfill specific purposes.
- It is centrally managed during long-term operation to continue to fulfill those purposes as well as any new ones the system owners might wish to address.
- The component systems maintain an ability to operate independently, but their normal operational mode is subordinated to the central managed purpose.

This research is based on Acknowledged SoS. The major objectives of the reasearch are:

- To develop a simulation for Acknowledged SoS architecture selection and evolution.
- To have a structured, repeatable approach for planning and modeling.
- To study and evaluate the impact of individual system behavior on SoS capability and architecture evolution process.

The dynamic planning for a SoS is a challenging endeavor. Department of Defense (DoD) programs constantly face challenges to incorporate new systems and upgrade existing systems over a period of time under threats, constrained budget, and uncertainty. It is therefore necessary for the DoD to be able to look at the future scenarios and critically assess the impact of technology and stakeholder changes. The DoD currently is looking for options that signify affordable acquisition selections and lessen the cycle time for early acquisition and new technology addition. FILA-SoS provides a decision aid in answering some of the questions.

This volume gives an overview of a novel methodology known as the Flexible Intelligent & Learning Architectures in System-of-Systems (FILA-SoS). Some the challenges that are prevalent in SoS architecting and how FILA-SoS attempts to address them is explained in the next section.

SYSTEM OF SYSTEM CHALLENGES

All these recent developments are helping us to understand Complex Adaptive Systems. They are at the edge of chaos as they maintain dynamic stability through constant self-adjustment and evolution. Chaos and order are two complementary states of our world. A dynamic balance exists between these two states.

Order and structure are vital to life. Order ensures consistency and predictability and makes the creation of systems possible. However, too much order leads to rigidity and suppresses creativity. Chaos constantly changes the environment creating disorder and instability but can also lead to emergent behavior and allows novelty and creativity. Thus, sufficient order is necessary for a system to maintain an ongoing identity, along with enough chaos to ensure growth and development. The challenge in Complex Adaptive Systems design is to design an organized complexity that will allow a system to achieve its goals. SoS is a complex systems by its nature due to the following characteristics that are component systems are operationally independent elements and also managerially independent of each other. This means that component systems preserve existing operations independent of the SoS. SoS has an evolutionary development and due to the large scale complex structure shows an emergent behavior. Emergence means the SoS performs functions that do not reside in any one component system.

2012 INCOSE SoS working group survey identified seven ‘pain points’ raising a set of questions for systems engineering of SoS which are listed in Table 1 (Dahman, 2012).

Table 1 System of Systems and Enterprise Architecture Activity

Pain Points	Question
Lack of SoS Authorities & Funding	What are effective collaboration patterns in systems of systems?
Leadership	What are the roles and characteristics of effective SoS leadership?
Constituent Systems	What are effective approaches to integrating constituent systems into a SoS?
Capabilities & Requirements	How can SE address SoS capabilities and requirements?
Autonomy, Interdependencies & Emergence	How can SE provide methods and tools for addressing the complexities of SoS interdependencies and emergent behaviors?
Testing, Validation & Learning	How can SE approach the challenges of SoS testing, including incremental validation and continuous learning in SoS?
SoS Principles	What are the key SoS thinking principles, skills and supporting examples?

The importance and impact on systems engineering of each pain point is illustrated below:

- **Lack of SoS Authorities & Funding and Leadership** pose several and severe governance and management issues for SoS. This conditions has a large impact on the ability to implement systems engineering (SE) in the classical sense to SoS. In addition, this problem affects the modeling & simulation activities.
- **Constituent Systems** play a very important role in the SoS. As explained earlier usually they have different interests and ambitions to achieve, which may or may not be aligned with the SoS.. Similarly models, simulations and data for these systems will naturally have to be attuned to the specific needs of the systems, and may not lend themselves easily to supporting SoS analysis or engineering
- **Autonomy, Interdependencies & Emergence** is ramifications of the varied behaviors and interdependencies of the constituent systems making it complex adaptive systems. Emergence comes naturally in such a state, which is often unpredictable. While modeling & simulation can aid in representing and measuring these complexities, it is often hard to achieve real life emergence. This is due to limited understanding of the issues that can bring up serious consequences during validation.
- **Capability of the SoS** and the individual systems capability needs may be high level and need definition in order to align them with the requirements of the SoS mission. The SoS mission is supported by constituent systems, which may not be able (or willing) to address them.
- **Testing, Validation & Learning** becomes difficult since the constituent systems continuously keep evolving, adapting, as does the SoS environment which includes stakeholders, governments, etc. Therefore creating a practical test-bed for simulating the large dynamic SoS is a challenge in itself. Again modeling & simulation can solve part of the problem such as enhancing live test and addressing risk in SoS when testing is not feasible; however, this requires a crystal clear representation of the SoS which can be difficult as discussed in earlier points.

- **SoS Principles** are still being understood and implemented. Therefore, the rate of success is yet to be addressed formally. This poses some pressure on the progress of SoS engineering. Similarly, there is an absence of a well-established agreeable space of SoS principles to drive development and knowledge. This constricts the effective use of potentially powerful tools.

The DoD 5000.2 is currently used as the acquisition process for complex systems. Schwartz (2010) described this process as an extremely complex systemic process that cannot always constantly produce systems with expected either cost or performance potentials. The acquisition in DoD is an SoS problem that involves architecting, placement, evolution, sustainment, and discarding of systems obtained from a supplier or producer. Numerous attempts undertaken to modify and reform the acquisition process have found this problem difficult to tackle because the models have failed to keep pace with actual operational scenarios. Dombkins (1996) offered a novel approach to model complex projects as waves. He suggested that there exists a major difference in managing and modeling traditional projects versus complex projects. He further illustrated his idea through a wave planning model that exhibits a linear trend on a time scale; on a spatial scale, it tries to capture the non-linearity and recursiveness of the processes. In general, the wave model is a developmental approach that is similar to periodic waves. A period, or multiple periods, can span a strategic planning time. The instances within the periods represent the process updates. A recently proposed idea (Dahman, Lane, Rebovich, & Baldwin, 2008) that SoS architecture development for the DoD acquisition process can be anticipated to follow a wave model process. According to Dahman DoD 5000.2 may not be applicable to the SoS acquisition process. Acheson (2013) proposed that Acknowledged SoS be modeled with an Object-Oriented Systems Approach (OOSA). Acheson also proposes that for the development of SoS, the objects should be expressed in the form of a agent based model.

The environment and the systems are continuously changing. Let there be an initial environment model, which represents the SoS acquisition environment. As the SoS acquisition progresses through, these variables are updated by the SoS Acquisition Manager to reflect current acquisition environment. Thus, the new environment model at a new time has different demands. To fulfill the demands of the mission a methodology is needed to assess the overall performance of the SoS in this dynamic situation. The motivation of evolution are the changes in the SoS environment (Chattopadhyay, Ross, & Rhodes, 2008). The environmental changes consist of:

- SoS Stakeholder Preferences for key performance attributes
- Interoperability conditions between new and legacy systems
- Additional mission responsibilities to be accommodated
- Evolution of individual systems within the SoS

Evaluation of architectures is another SoS challenge area as it lends itself to a fuzzy approach because the criteria are frequently non-quantitative, or subjective (Pape & Dagli, 2013), or based on difficult to define or even unpredictable future conditions, such as “robustness.” Individual attributes may not have a clearly defined, mathematically precise, linear functional form from worst to best. The goodness of one attribute may or may not offset the badness of another attribute. Several moderately good attributes coupled with one very poor attribute may be better than an architecture with all marginally good attributes, or vice-versa. A fuzzy approach allows many of these considerations to be handled using a reasonably simple set of rules, as well as having the ability to include non-linear characteristics in the fitness measure. The simple rule set allows small adjustments to be made to the model to see how seemingly small changes affect the outcome. The methodology outlined in this research and technical report falls under a multi-level plug-and-play type of modeling approach to address various aspects of SoS acquisition environment: SoS architecture evaluation, SoS architecture evolution, and SoS acquisition process dynamics including behavioral aspects of constituent systems.

How Does FILA-SoS Address SoS Pain Points

The first pain point is Lack of SoS Authorities & Funding which begs a question "What are effective collaboration patterns in systems of systems?"

Since there is lack of SoS Authority but more so persuasion involved in the workings of a SoS, systems are allowed to negotiate with the SoS manager. Deadline for preparation, funding and performance required to complete the mission are some of the issues that form the negotiation protocol. Besides different combination of behavior types assigned to the systems can help us gauge the best effective collaboration patterns in systems of systems after the end of negotiations.

The leadership issues pose the question, "What are the roles and characteristics of effective SoS leadership?" This is addressed by incorporating views from multiple stakeholders while assessing the architecture's quality. In addition, we maintain that the characteristics are similar to what an Acknowledged SoS manager would have while distributing funds and resources among systems for a joint operation. The SoS manager also has the opportunity to form his decision based on most likely future scenarios, thus imparting him an edge as compared to other models. This will improve the process of acquisition in terms of overall effectiveness, less cycle time and integrating legacy systems. Overall, the role of the leadership is presented a guide than someone who would foist his authority.

The third pain point question, "What are effective approaches to integrating constituent systems into a SoS? is addressed below. A balance has to be maintained during acquisition between amount of resources used and the degree of control exercised by the SoS manager on the constituent systems. The meta-architecture generation is posed as a multi-objective optimization problem to address this pain point. The constituent systems and the interfaces between them are selected while optimizing the resources such as operations cost, interfacing cost, performance levels etc. The optimization approach also evaluates the solutions based on views of multiple stakeholders integrated together using a fuzzy inference engine.

How can SE address capabilities and requirements? is the fourth pain point and is answered in this paragraph. Organizations that acquire large-scale systems have transformed their attitude to acquisition. Hence, these organizations now want solutions to provide a set of capabilities, not a single specific system to meet an exact set of specifications. During the selection process of systems it is ensured that, a single capability is provided by more than one system. The idea is to choose at least one systems having unique capability to form the overall capability of the SoS.

The fifth pain point on autonomies, emergence and interdependencies is one of the most important objectives of this research. This objective can be described as “How can SE provide methods and tools for addressing the complexities of SoS interdependencies and emergent behaviors?”. Each system has an autonomous behavior maintained through pre-assigned negotiation behaviors, differ operations cost, interfacing cost and performance levels while providing the same required capability. The interfacing among systems is encouraged to have net-centric architecture. The systems communicate to each other through several communication systems. This ensures proper communication channels. Together the behavior and net-centricity make it complex systems thus bringing out the emergence needed to address the mission.

FILA-SoS is an excellent integrated model for addressing the complexities of SoS interdependencies and emergent behaviors as explained in the above paragraphs.

As for the sixth pain point on testing, validation and learning goes, FILA-SoS has been tested on three notional examples so far the ISR, Search and Rescue (SAR) and the Toy problem for Aircraft Carrier Performance Assessment. For ISR (refer to Figure 2) a guiding physical example is taken from history. During the 1991 Gulf War, Iraqi forces used mobile SCUD missile launchers called Transporter Erector Launchers (TELS) to strike at Israel and Coalition forces with ballistic missiles. Existing intelligence, surveillance, and reconnaissance (ISR) assets were inadequate to find the TELs during their vulnerable setup and knock down time. The “uninhabited and flat” terrain of the western desert was in fact neither of those things, with numerous Bedouin goat herders and their families, significant traffic, and thousands of wadis with culverts and bridges to conceal the TELs and obscure their movement.

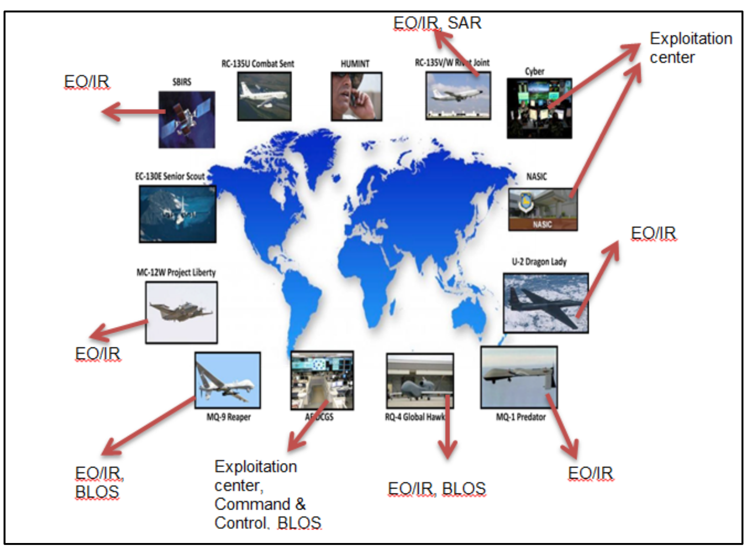


Figure 2 ISR System-of-Systems for Testing FILA-SoS

A Coast Guard Search and Rescue (SAR) (Figure 3) SoS engineering and development problem is selected for serving the Alaskan coast. Detailed information about this case study can be found in Dagli et al (2013). There is increasing use of the Bering Sea and the Arctic by commercial fisheries, oil exploration and science, which increases the likelihood of occurrence of possible SAR scenarios.

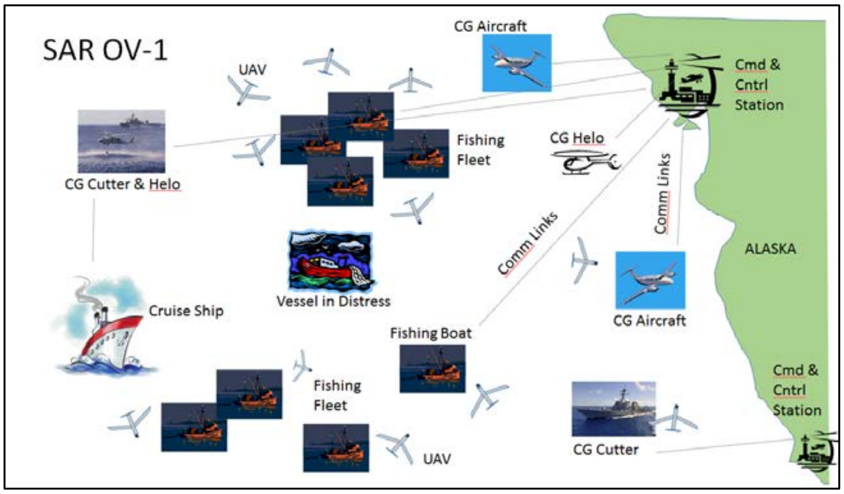


Figure 3 SAR System-of-Systems for Testing FILA-SoS

The toy problem for assessing the performance of the aircraft carrier involves multiple systems such as satellites, uav's and ground station that support the aircraft carrier to fulfill the mission (refer to Figure 4). The results have been obtained for multiple waves of the evolution process for all the examples.

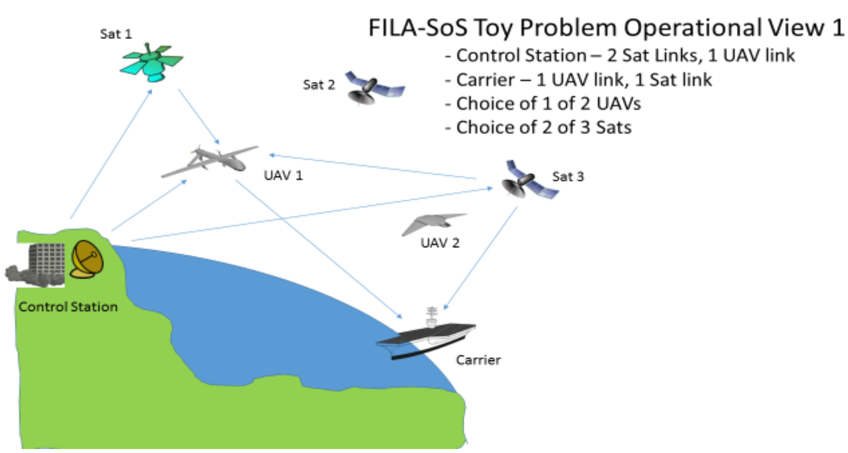


Figure 4 Aircraft Carrier Performance Assessment for Testing FILA-SoS

These example discussed above clearly show the domain independence of FILA-SoS.

FILA-SoS is a novel method of making sequential decisions over a period for SoS development. The goal is to apply the integrated model to dynamically evolve SoS architecture and optimize SoS architecture, design and validate through simulation tools. The integrated model structure can be applied to various application areas including development of dynamic water treatment SoS architecture, development of dynamic Air Traffic Management SoS, and development of autonomous ground transport SoS. FILA-SoS has a number of abilities that make it unique such as:

- Aiding the SoS manager in future decision making
- To assist in understanding the emergent behavior of systems in the acquisition environment and impact on SoS architecture quality
- To facilitate the learning of dynamic behavior of different type of systems (cooperative, semi-cooperative , non-cooperative)
- Identifying intra and interdependencies among SoS elements and the acquisition environment
- Modeling and application to a wide variety of complex systems models such as logistics, cyber-physical systems and similar systems
- Acting as a Test-bed for decision makers to evaluate operational guidelines and principles for managing various acquisition environment scenarios
- Appropriate to model SoS that evolve over a period of time under uncertainties by multiple wave simulation capability.

OVERVIEW OF THE FILA-SOS INTEGRATED MODEL

In this section an overview of FILA-SoS is described. The model developed called the FILA-SoS is using straightforward system definitions methodology and an efficient analysis framework that supports the exploration and understanding of the key trade-offs and requirements by a wide range system-of-system stakeholders and decision makers in a short time. FILA-SoS and the Wave Process address four of the most challenging aspects of system-of-system architecting:

- Dealing with the uncertainty and variability of the capabilities and availability of potential component systems.
- Providing for the evolution of the system-of-system needs, resources and environment over time.
- Accounting for the differing approaches and motivations of the autonomous component system managers.
- Optimizing system-of-systems characteristics in an uncertain and dynamic environment with fixed budget and resources

DEFINITION OF VARIABLES FOR SOS

This list comprises of the notation for variables used to solve the Acknowledged SoS architectural evolution problem:

C: Overall capability (the overall goal to be achieved by combining sub-capabilities)
c_j: $j \in \boldsymbol{J}$, $\boldsymbol{J}= \{1, 2,\dots, M\}$:
Constituent system capabilities required
s_i: $i \in \boldsymbol{I}$, $\boldsymbol{I}= \{1, 2,\dots, N\}$:
Total number of systems present in the SoS problem
Let $\boldsymbol{A}$ be a N x $M - matrix\ of\ a_{ij}\ where$
$a_{ij} = 1\ if$ capability j is possessed by system i
$a_{ij} = 0\ otherwise$
P_i: Performance of system i for delivering all capabilities $\sum_j a_{ij}$
F_i: Funding of system i for delivering all capabilities $\sum_j a_{ij}$
D_i: Deadline to participate in this round of mission development for system i
IF_{ik} Interface between systems $i\ and\ k\ s.t.\ s \neq k,\ k \in I$
IC_i: The cost for development of interface for system i
OC_i: The cost of operations for system i
KP_r : $r \in \boldsymbol{R}$, $R= \{1, 2,\dots, Z\}$:
The key performance attributes of the SoS
FA: Funding allocated to SoS Manager
p= {1, 2,…, P}:
Number of negotiation attributes for bilateral negotiation
t_{max}: Total round of negotiations possible
t : Current round of negotiation (epochs)

t_{max}: Total round of negotiations possible
$V_{pi}^{SoS}(t)$: The value of the attribute p for SoS manager at time t for system i
$V_{pi}^{S}(t)$: The value of the attribute p for system i owner at time t
TQ: Threshold architecture quality

The model involves a list of stakeholders such as the Acknowledged SoS manager, system owners/managers, SoS environment etc.

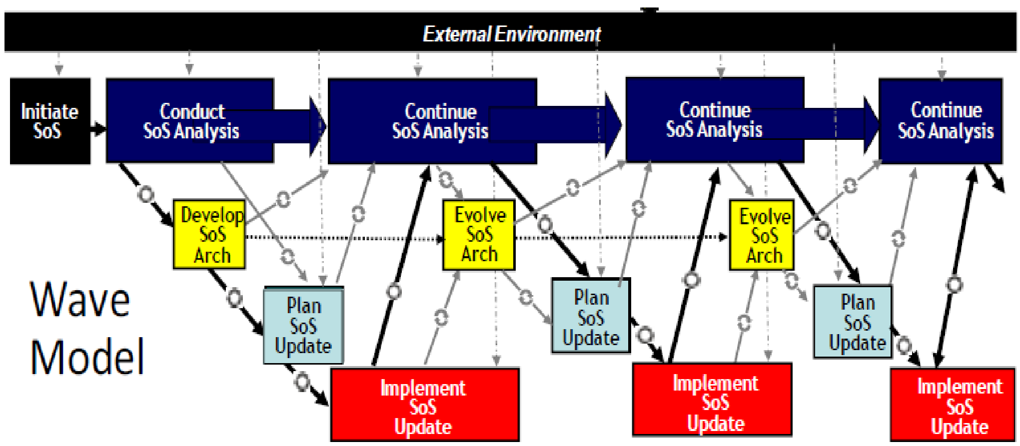


Figure 5 The Wave Model of SoS initiation, Engineering, and Evolution

FILA-SoS follows the Dahmann's proposed SoS Wave Model process for architecture development of the DoD acquisition process as depicted in Figure 5. FILA-SoS addresses the most important challenges of SoS architecting in regards to dealing with the uncertainty and variability of the capabilities and availability of potential component systems. The methodology also provides for the evolution of the system-of-system needs, resources and environment over time while accounting for the differing approaches and motivations of the autonomous component system managers. FILA-SoS assumes to have an uncertain and dynamic environment with fixed budget and resources for architecting SoS. The overall idea being to select a set of systems and interfaces based on the needs of the architecture in a full cycle called the wave. Within the wave, there may be many negotiation rounds, which are referred to as epochs. After each wave, the systems selected during negotiation in the previous wave remain as part of the meta-architecture whilst new systems are given a chance to replace those left out as a result.

Processes involved in the wave model and their analog in FILA-SoS can be explained through the first stage of Initializing the SoS. In terms of initializing, wave process requires to understand the SoS objectives and operational concept (CONOPS), gather information on core systems to support desired capabilities. This starts with the overarching capability C desired by Acknowledged SoS manager and defining the c_j or sub-capabilities required to produce capability C and FA, funding allocated to SoS Manager. These also form the input to the FILA-SoS for the participating systems s_i. FILA-SoS requires t_{max} the number of negotiation cycles, selection of the meta-architecture modelling procedure and system negotiation models assigned to participating systems.

The second stage is called the Conduct_SoS_Analysis. For the Wave process, it represents starting an initial SoS baseline architecture for SoS engineering based on SoS requirements space, performance measures, and relevant planning elements. For FILA-SoS the baseline architecture is called as the meta-architecture. Meta-architecture is basically picking up the systems s_i and their respective capabilities a_{ij}. Meta-architecture modelling requires the values for KP_t , the key performance attributes of the SoS, P_i (Performance of system i) , F_i (Funding of system i), and D_i deadline to participate in this round of mission development for system i which is assumed to be the total for all capabilities possessed by system i. The cost for development of a single interface for system i, IC_i and OC_i the cost of operations for system i is also needed at this stage of the model. The next step is the Develop/ Evolve SoS. In this case in terms of the Wave process essential changes in contributing systems in terms of interfaces and functionality in order to implement the SoS architecture are identified. Within FILA-SoS this signals the command to send connectivity request to individual systems and starting the negotiation between SoS and individual systems. This stage requires the number of negotiation attributes P for a bilateral negotiation between Acknowledged SoS manager and each systems i selected in the meta-architecture and t_{max} which denotes the total round of negotiations possible.

The next phase is Plan SoS Update in Wave process. In this, phase the architect plans for the next SoS upgrade cycle based on the changes in external environment, SoS priorities, options and backlogs. There is an external stimulus from the environment, which affects the SoS architecture. To reflect that in FILA-SoS determines which systems to include based on the negotiation outcomes and form a new SoS architecture. Finally, the last stage in Wave process is Implement SoS Architecture which establishes a new SoS baseline based on SoS level testing and system level implementation. In the FILA-SoS the negotiated architecture quality is evaluated based on KP_r, key performance attributes of the SoS. If the architecture quality is not up to a predefined quality or TQ the threshold architecture quality the Acknowledged SoS manager and systems i selected in the meta-architecture go for renegotiations. Finally the process moves on to the next acquisition wave. The evolution of SoS should take into account availability of legacy systems and the new systems willing to join, adapting to changes in mission and requirement, and sustainability of the overall operation. FILA-SoS also has the proficiency to convert the meta-architecture into an executable architecture using the Object Process Model (OPM) and Colored Petri Nets (CPN) for overall functionality and capability of the meta-architecture. These executable architectures are useful in providing the much-needed information to the SoS coordinator for assessing the architecture quality and help him in negotiating better.

Some of the highlights of FILA-SoS are described in terms of its capabilities, value added to systems engineering, ability to perform “What-if Analysis”, modularity of integrated models, its potential applications in the real world and future additions to the current version. The most important capability of FILA-SoS is it being an integrated model for modeling and simulating SoS systems with evolution for multiple waves. Secondly, all models within FILA-SoS can be run independently and in conjunction with each other. Thirdly, there are two model types that represent SoS behavior and various individual system behaviors. Finally, it has the capacity to study negotiation dynamics between SoS and individual systems.

The value added by FILA-SoS to systems engineering is it aids the SoS manager in future decision making, can help in understanding the emergent behavior of systems in the acquisition environment and its impact on SoS architecture quality. Besides, it has three independent systems behavior models, which are referred to as cooperative, semi-cooperative and non-cooperative. These behavior models are used to Study the dynamic behavior of different type of systems while they are negotiating with SoS manager. In addition, FILA-SoS assists in identifying intra and interdependencies among SoS elements and the acquisition environment.

FILA-SoS also can facilitate a “What-if” Analysis using variables such as SoS funding and capability priority that can be changed as the acquisition progresses though wave cycles. The parameter setting for all negotiation models can be changed and rules of engagement can be simulated for different combinations of systems behaviors.

Potential Application of FILA-SoS include complex systems models such as logistics, cyber-physical systems. In addition, it can act as test-bed for decision makers to evaluate operational guidelines and principles for managing various acquisition environment scenarios. While the future capabilities that we would like to be included are extending the model to include multiple interface alternatives among systems and incorporation of risk models into environmental scenarios.

INDEPENDENT MODULES OF FILA-SOS

The FILA-SoS has a number of independent modules that are integrated together for meta-architecture generation, architecture assessment, meta-architecture executable model, and meta-architecture implementation through negotiation. An overall view is presented in Figure 6.

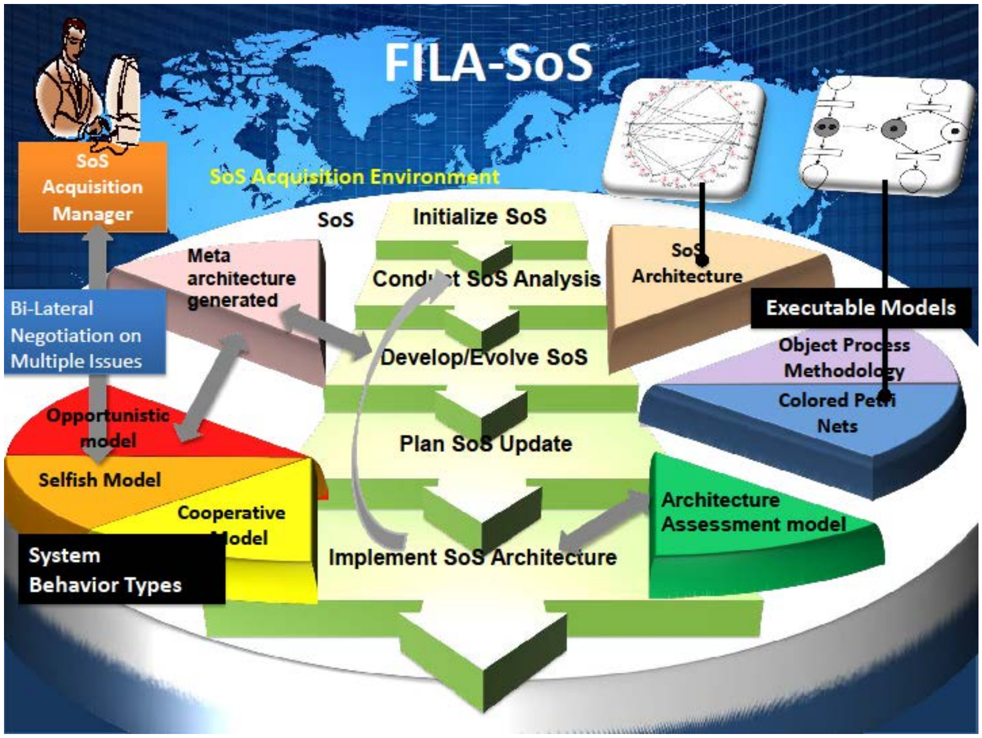


Figure 6 Integrated modules within FILA- SoS

All the independent models are listed below for reference:

- Meta-Architecture Generation Model
- Architecture Assessment Model
- SoS Negotiation Model
- System Negotiation Model: Non-Cooperative
- System Negotiation Model: Cooperative
- System Negotiation Model: Semi-Cooperative
- Executable Architecting Model: OPM & CPN
- Overall Negotiation Framework

The first meta-architecture generation method is fuzzy-genetic optimization model (Pape, Agarwal, Giammarco & Dagli, 2014). This model is based on evolutionary multi-objective optimization for SoS architecting with many key performance attributes (KPA). It also has a type-1 fuzzy assessor for dynamic assessment of domain inputs and that forms the fitness function for the genetic algorithm. It returns the best architecture (meta-architecture) consisting of systems and their interfaces. It is a generalized method with application to multiple domains such as Gulf War Intelligence/Surveillance/Reconnaissance Case and Alaskan Maritime Search and Rescue Case.

The second meta-architecture generation model is based on multi-level optimization (Konur & Dagli, 2014). In this model, architecting is done in two rounds: the first being the initiating the SoS by selecting the systems to be included in the SoS and then improving the SoS's performance by allocating funds to participating systems. The model is generic based on multiple attributes such as maximum performance, minimum cost and minimum deadline. It based on a Stackelberg game theoretical approach between the SoS architect and the individual systems.

The particle swarm optimization (Agarwal, Pape, & Dagli, 2014) technique for meta-architecture generation is similar to fuzzy-genetic model. Except for the fact that evolutionary optimization technique in this case is based on swarm intelligence. In addition, there are some new key performance attributes used to calculate the architectures quality. Cuckoo search optimization (Agarwal, Wang, & Dagli, 2014) based meta-architecture is again anew biologically inspired method of optimization. It has been shown that it in certain cases it performs better than PSO.

The first architecture assessment method is based on type-1 fuzzy logic systems (FLS) (Pape et al., 2013). The Key Performance Parameters (KPP) chosen are performance, affordability, flexibility, and robustness. It can capture the viewpoints of multiple stakeholders'. It can also accommodate any number of KPPs.

Another architecture assessment method is based on type-2 fuzzy modular nets (Agarwal, Pape & Dagli, 2014). The attributes used for evaluation were Performance, Affordability, Developmental Modularity, Net-Centricity and Operational Robustness. Type-1 fuzzy sets are able to model the ambiguity in the input and output variables. However, type-1 fuzzy sets are insufficient in characterizing the uncertainty present in the data. Type-2 fuzzy sets proposed by Zadeh (1975) can model uncertainty and minimize its effects in FLS (Mendel & John, 2002).

It is not possible to implement such meta-architecture without persuading the systems to participate, hence to address the issue a negotiation model is proposed based on game theory (Ergin, 2104). It is an incentive based negotiation model to increase participation of individual systems into Search and Rescue SoS. The model provides a strategy for SoS management to determine the appropriate amount of incentives necessary to persuade individual systems while achieving its own goal. The incentive contract is designed based on the objectives of the SoS and the individual systems. Individual system's objective is to secure highest incentives with minimal effort while the SoS manager's goal is to convince individual systems to join the SoS development while maximizing its own utility. Determining the incentives for individual systems can be formulated as a multi-constraint problem where SoS manager selects a reward for the individual system such that the reward will maximize SoS manager's expected utility while satisfying the constraints of the individual systems.

Another negotiation model based on clustering and neural networks is developed (Agarwal, Saferpour & Dagli, 2014). This model involves adapting the negotiation policy based on individual systems behavior that is not known to the SoS manager. The behavior is predicted by clustering the difference of multi-issue offers. Later the clustered data is trained using supervised learning techniques for future prediction.

Individual systems providing required capabilities can use three kinds of negotiation models based on their negotiation strategies non-cooperative Linear Optimization model, cooperative fuzzy negotiation model, and Semi-cooperative Markov chain model (Dagli et al., 2013).

Executable architectures are generated using a hybrid of Object Process Methodology (OPM) and Colored Petri Nets (CPN) (Agarwal, Wang, & Dagli, 2014), (Wang, Agarwal, & Dagli, 2014), and (Wang & Dagli, 2011). To facilitate analysis of interactions between the participating systems in achieving the overall SoS capabilities, an executable architecture model is imperative. In this research, a modeling approach that combines the capabilities of OPM and CPN is proposed. Specifically, OPM is used to specify the formal system model as it can capture both the structure and behavior aspects of a system in a single model. CPN supplements OPM by providing simulation and behavior analysis capabilities. Consequently, a mapping between OPM and CPN is needed. OPM modeling supports both object-oriented and process-oriented paradigm. CPN supports state-transition-based execution semantics with discrete-event system simulation capability, which can be used to conduct extensive behavior analyses and to derive many performance metrics.

NON-COOPERATIVE SYSTEM NEGOTIATION MODEL

Resource constrained scheduling problems (RCSPs) concern with allocating scarce resources to activities over time in order to meet the requirements on the quality, quantity, completion time, and others in the delivery of projects, products, or services (Brucker, Drexl, Möhring, Neumann, & Pesch, 1999). The problems are often formulated as optimization models where objective functions can be the minimization of project duration, minimization of the project cost given performance payments and penalties, and minimization of the consumption of critical resources (Talbot, 1982). Resource constraints can be on a period-to-period basis such as the amount of skilled labor available for each day; they can also be over the life of the project such as the total funding for the project. The solution to RCSPs specified when and how a job is processed. Various methods have been implemented to solve RCSPs, including integer programming (Talbot, 1982), heuristics (Herroelen, De Reyck, & Demeulemeester, 1998), genetic algorithm (Wall, 1996), and simulation (Bouleimen & Lecocq, 2003).

RCSPs can be a candidate model of individual systems because they are relevant to the SoS problem in this research. Participation requests from the SoS can be seen as projects with multiple jobs. Each job has specified completion time (e.g., deadline), performance requirement (e.g., capacity requirement), and performance payments (e.g., funding). A project's jobs usually consume the same resources that are provided either by the project execution entity (i.e., the individual system) or the project provider (i.e., the SoS). For example, performing a project consumes a portion or all of the performance payment provided by the project provider; it may also need skilled workers belonging to the project execution entity. The project execution entity schedules all jobs over time by specifying when and how each job is performed. The best schedule is often determined using optimization techniques. That is, the project execution entity formulates an optimization problem for the project by specifying the objective of executing the project. Constraints must be considered include those specified by the project provider (e.g., project completion time), determined by the environment or context (e.g., the expected return from projects with similar risks in the market), and specified by the project execution entity (e.g., the minimum rate of return from the project). An assessment of the project based on the optimal project schedule provides useful information for the negotiation between the project provider and the project execution entity. For example, if the project execution entity finds out that the performance payment provided by the SoS is not sufficient to cover project expenses or the project deadline is too short to be met, she will ask for additional payments or additional time with a justification.

MODEL OVERVIEW

A model of an individual system k is built, which is capable of providing both capabilities to a system of systems (SoS) and interfaces with other individual systems in the SoS. The request for participation is sent from SoS to the individual system, including:

- Requested capabilities, C_i
- Requested interfaces between the system k and other individual system j on capability i, χ_{ij}

- Deadline of delivery, $SoS.d_i$
- Funding for providing the requested capabilities, $SoS.f_i$
- The performance requirement on each of requested capabilities, $SoS.p_i$

This model can be termed a "non-cooperative" model in that the necessary condition for the individual system k to collaborate with the SoS is that the incremental profit from the participation is nonnegative. Therefore, a resource allocation problem is formulated to model the decision behavior of the individual system. The optimization problem is solved with considerations of the capabilities, resources and efficiency of the system. Moreover, the market condition is modeled so that the system agent has a rational assessment of the incremental profit provided by the SoS.

The outputs sent from the system k to the SoS include:

- Provided capabilities, C_i
- Provided interfaces between the system *k* and other individual system *j* on capability *i*, χ_{ij}
- Delivery time deviation (additional time needed), $SoS.\Delta d_i$
- Funding deviation (additional fund needed), $SoS.\Delta f_i$
- Performance deviation (under performance is any), $SoS.\Delta p_i$

The outputs listed above are saved in an n by (m+4) matrix shown below.

	Architecture (the portion related to system j)																							deadline	funding	performance
	S_j	I_1	I_2	I_3	I_4	I_5	I_6	I_7	I_8	I_9	I_{10}	I_{11}	I_{12}	I_{13}	I_{14}	I_{15}	I_{16}	I_{17}	I_{18}	I_{19}	I_{20}	I_{21}	I_{22}	Systemj.Δd	Sustemj.Δf	systemj.Δp
C_1																										
C_2																										
C_3																										
C_4																										
C_5																										

Figure 7: Outputs from System to SoS

Model Description

Two linear programming (LP) models are built to provide two scenarios of negotiations with the SoS, which are solved using the optimization tool in Matlab. In the first scenario, the SoS is informed possible performance deviation (if any) at the provided funding and time. In the second scenario, the SoS may be provided the capabilities and interfaces as it desires, yet it may be asked to provide additional funding and/or time. Model setting

The following is the setting of the individual system *k*.

i: the index of capabilities, and $i = 1, 2, \ldots, n$.
I: The set of capability indices, and $I = \{1, 2, \ldots, n\}$.
j: the index of individual systems, and $j = 1, 2, \ldots, m$.
J: The set of system indices and $J = \{1, 2, \ldots, m\}$.
C_i: binary variable indicating whether the capability *i* is requested, $\forall\ i$.
I_{req}: the set of requested capability indices; $I_{req} \subseteq I$.
p_i: performance requirement on capability *i*, $\forall\ i \in I_{req}$.

d_i: deadline to deliver capability i, $\forall\, i \in I_{req}$.
f_i: funding provided to capability i, $\forall\, i \in I_{req}$.
χ_{ji}: binary variable indicating the requested interface between system j and k ($k \neq j$) in forming capability i, $\forall\, i$ and j.
n_i: the number of interfaces that system k is requested to provide on capability i, $\forall\, i \in I_{req}$.
y_i: the system k's throughput of capability i in a unit of time, $\forall\, i \in I$.
I_y: the set of indices of capabilities that system k is able to build, $I_y = \{i \mid y_i > 0\}$.
I_{reqf}: the set of indices of capabilities that is requested by the SoS and the system k is able to build, and $I_{reqfsbl} = I_y \cap I_{req}$.
$I_{reqifsbl}$: the set of indices of capabilities that is requested by the SoS but the system k is not able to build, and $I_{reqifsbl} = \bar{I}_y \cap I_{req}$.
Z_{avg}: the system k's average available resource per unit of time.
Z_{range}: the range of the system k's resource per unit of time.
T: planning time horizon. $T = \max\{\text{int}\lceil \sum_{i\in Ireq} p_i/(y_i \times Z_{\min}) \rceil, \max_{i\in Ireq} (d_i)+1\}$. $\text{int}\lceil \sum_{i\in Ireq} p_i/(y_i \times Z_{\min}) \rceil$ is the time needed (in integer) if the system k has only the minimum resource and build the requested capabilities in sequence. $[\max_{i\in Ireq} (d_i)]$ is the relaxed upper bound of deadlines. The model assumes the planning time horizon is no shorter than these two.
t: the index of time; $t = 1, 2, \ldots, T$.
Z_t: the system k's resource available at time t, and $Z_{\min} = Z_{\text{avg}}\text{-}0.5Z_{\text{range}} \leq Z_t \leq Z_{\max} = Z_{\text{avg}}+0.5Z_{\text{range}}$,$\forall\, t$.
Z_{ti}: the resource allocated to build capability i at time t. $z_{it} \geq 0\ \forall\ i \in I_{reqfsbl}$ and $\forall\ t$. $z_{it\ is}$ are the decision variables.
c_{ci}: the cost of consuming one unit of resource in providing capability i at time zero, $\forall\, i \in I_y$.
c_{Ii}: the cost of consuming one unit of resource in providing interfaces associated with capability i, as a percentage of c_{ci}.
g: the inflation rate of unit cost over time.
c'_{ti}: the cost of consuming one unit of resource in providing capability i and the associated interfaces at time t, for $t = 0, 1, \ldots, .d_i$ and $i \in I_y$. $c_{ti} = c_{ci}(1+c_{Ii}n_i)\exp(g(t\text{-}0))$.
pm: profit margin.
c_{ti}: the market price of one unit of resource for providing capability i at time t. $c_{ti} = (1+\text{pm})c'_{ti}$.
c''_{ti} (>0): virtual penalty on exceeding the deadlines, for $d_i < t \leq T$ and $i \in I_y$.
cp_{ti} ($= c'_{ti} + (c_{ti} - c'_{ti})^+ + c''_{ti}$): the cost of consuming one unit of resource for providing capability i and the associated interfaces at time t, after the adjustment for the opportunity cost, $\forall\, t$.
w_i : the weight that system k put on capability i, $\forall\, i \in I_{reqfsbl}$; and $w_i = [f_i/d_i] / (\sum_{i\in Ireq} f_i/d_i)$.

LINEAR PROGRAMMING (LP) MODELS FOR DECISION SUPPORT

The first problem (P1) helps identify the best performance of system k on the requested capabilities and interfaces with the given funding and deadlines. First, $z_{ti} = 0$ for $i \notin (I_{reqfsbl})$; and $\forall i \in I_{reqfsbl}$, z_{ti}'s are determined by the following problem:

$$\min \sum_{i \in I_{reqfsbl}} \left(-w_i \sum_{t=1}^{d_i} y_i z_{ti} \right)$$

$$\begin{aligned}
&\textbf{subject to:}\\
&\sum_{t=1}^{d_i} y_i z_{ti} \le p_i, \forall i \in I_{reqfsbl};\\
&\sum_{i \in I_{reqfsbl}} z_{ti} \le Z_t, \text{ for } t = 1,2,\cdots, \max_{i \in I_{reqfsbl}} \{d_i\};\\
&\sum_{i \in I_{reqfsbl}} \sum_{t=1}^{d_i} c_{ti} z_{ti} \le \sum_{i=1}^{n} f_i;\\
&z_{ti} \ge 0, \forall i \in I_{reqfsbl} \text{ and for } t = 1,2,\cdots,d_i.
\end{aligned} \tag{1}$$

The objective of (P1) is to minimize the weighted sum of under performances. The first constraint means the performance on a requested capability does not exceed the required performance. The second constraint means the resource consumed at time *t* should not exceed the resource available at then. The third constraint means the total costs should not exceed the total funding provided. The fourth constraint indicates that the resource relocated to building a requested capability and the associated interfaces is nonnegative.

Denote by $\{z_{ti}^* \mid t = 1, 2, \ldots, d_i$, and $i \in I_{reqfsbl}\}$ an solution to (P1). Let $z_{ti}^*=0$ at $t = 1, 2, \ldots, d_i$ and for $i \notin I_{reqfsbl}$ so that z_{it}^* is defined at any *i*. The minimized performance deviation is calculated by

$$\Delta p_i^* = y_i \sum_{t=1}^{d_i} z_{ti}^* - p_i \tag{2}$$

The minimized performance deviation is calculated based on the assumptions of no extension of deadlines and/or no additional funding:

$$\Delta d_i^* = 0 \tag{3}$$

$$\Delta f_i^* = 0 \tag{4}$$

The second problem (P2) helps determine the minimum additional funding and/or minimum additional time to meet the goal of forming all the capabilities and interfaces that system *k* is capable of providing. First, $z_{ti} = 0$ for $I \notin I_{reqfsbl}$; and $\forall\ I \in I_{reqfsbl}$, z_{ti}'s are determined by the following problem:

(P2)

$$\min\left\{\sum_{i\in I_{reqfsbl}}\sum_{t=1}^{T}cp_{ti}z_{ti}\right\}$$

subject to:

$$y_i\sum_{t=1}^{T}z_{ti}=p_i,\ \forall i\in I_{reqfsbl}$$

$$\sum_{i\in I_{reqfsbl}}z_{ti}\le Z_t,\ \forall t$$

$$z_{ti}\ge 0,\ \forall t \text{ and } i\in I_{reqfsbl} \tag{5}$$

The objective of (P2) is to minimize the total costs, including the additional fund used and the virtual penalty on additional time used. The first constraint means the performance on a requested capability is equal to the required performance. The second constraint means the resource consumed at time *t* should not exceed the resource available at then. The third constraint indicates that the resource relocated to building a capability and the associated interfaces is nonnegative.

In (P2), the funding constraint is relaxed and the deadline for finishing any capability *i* is "extended" to *T*. We choose *T* to be

$$T=\max\left\{\operatorname{int}\left\{\sum_{i\in I_{reqfsbl}}\left\lceil\frac{p_i}{Z_{\min}y_i}\right\rceil\right\},\ \max_{i\in I_{reqfsbl}}(d_i)\right\} \tag{6}$$

$\operatorname{int}\left\{\sum_{i\in I_{reqfsbl}}\left\lceil\frac{p_i}{Z_{\min}y_i}\right\rceil\right\}$ is the time needed if building capabilities in a sequential manner and the system *k* has just the minimum resource. The SoS may provide a very long deadline that is not needed by the system k. Considering this possibility, we choose such a value of *T* so that a feasible solution to (P2) is assured.

$\{c''_{ti}\,|\,t=d_i+1,\ldots,T;\ i\in I_{reqfsbl}\}$ is the virtual penalty on exceeding the deadlines. For any capability *i*, the penalty satisfies, where □ indicates much greater sign.

$$c''_{(d_i+1)i} \gg \max_{1\le t\le d_i} c_{it} \tag{7}$$

and

$$c''_{ti}>c''_{si},\ \forall t>s>d_i \tag{8}$$

The virtual penalty like such indicates that the cost to build any capability *i* after the desired deadline, d_i, is extremely high and the marginal cost grows with the prolonged time. Consequently, the objective function of (P2) discourages the use of the resource after the deadlines unless system *k* has to. Therefore, the objective function of (P2) effectively penalizes the usages of both the extra funding and time.

Denote by $\{\hat{z}_{ti} \mid t = 1, 2, \ldots, T; i \in I_{reqfsbl}\}$ an solution to (P2). Let $\hat{z}_{ti}$=0 at any time *t* and for $i \notin I_{reqfsbl}$ to make $\hat{z}_{ti}$ be defined at any *i*. Since the system *k* guarantees the performance on the capabilities that it is able to provide, the minimum deviation of performance is either zero or $-p_i$. The minimum additional time needed is found to be

$$\Delta d_i^* = \begin{cases} \hat{d}_i - d_i & i \in I_{reqfsbl} \\ \inf & i \in I_{reqifsbl} \\ 0 & i \in \overline{I}_{req} \end{cases} \tag{9}$$

where

$$\hat{d}_i = \max_{1 \le t \le T}\{t \mid \hat{z}_{ti} > 0\}, \ \forall\, i \in I_{reqfsbl} \tag{10}$$

The total minimum additional funding is

$$\Delta f^* = \left(\sum_{i \in I_{reqfsbl}} \sum_{t=1}^{d_i} c_{ti}\hat{z}_{ti} - \sum_{i=1}^{n} f_i \right)^+ \tag{11}$$

which is further split as the additional funding needed for each requested capability:

$$\Delta f_i^* = \lambda_i \Delta f^* \tag{12}$$

Where λ_i is determined by

$$\lambda_i = \begin{cases} \left(\sum_{t=1}^{\hat{d}_i} c_{ti}^* z_{ti} - f_i \right)^+ \Big/ \sum_{i \in I_{reqfsbl}} \left(\sum_{t=1}^{\hat{d}_i} c_{ti}^* z_{ti} - f_i \right)^+ & i \in I_{reqfsbl} \\ \text{Inf} & i \in I_{reqifsbl} \\ 0 & i \in \overline{I}_{req} \end{cases} \tag{13}$$

NEGOTIATION WITH THE SOS

If the system k is not able to provide any capability requested by the SoS, that is, $I_{reqfsbl} = \emptyset$, the system k will tell the SoS that the performance deviation is zero, but the deviations of deadline and funding is extremely large:

$$System_k.\Delta p_i = 0,\ \forall i \tag{14}$$

$$System_k.\Delta d_i = \inf,\ \forall i \tag{15}$$

$$System_k.\Delta f_i = \inf,\ \forall i \tag{16}$$

Otherwise, the solutions to (P1) and (P2) generate the foundation of two negotiation scenarios for the system *k*. We assume these two scenarios occur with chances. The system *k* may not share the complete information (e.g., the best performance, the capabilities not capable of providing, cost information) with the SoS during the negotiation for some reasons (e.g., business secret, for better negotiation outcomes).

The first scenario of negotiation is derived from the solution to (P1). The agent of system *k* provides the SoS the performance deviation defined as:

$$System_k.\Delta p_i = \begin{cases} \Delta p_i^* & i \in I_{req} \\ 0 & \text{otherwise} \end{cases} \tag{17}$$

with the given funding and time:

$$System_k.\Delta d_i = \left[0, 0, \cdots, 0\right]^T \tag{18}$$

$$System_k.\Delta f_i = \left[0, 0, \cdots, 0\right]^T \tag{19}$$

Δp_i^* in Eqn. (317) has been defined in (2). In this scenario of negotiation, the system *k* has strong motivation to participate and, therefore, its shows the minimum deviation of performance to SoS, as it is indicated by Eqn. (17). Since the system *k* does not want to let the SoS know what capabilities it is not capable of providing, the system *k* does not update the C_i's and χ_{ji}'s.

The second scenario of negotiation is derived from the solution to (P2). The agent of system *k* provides the SoS the performance deviation defined as:

$$System_k.\Delta p_i = \begin{cases} 0 & i \in I_{reqfsbl} \\ -p_i & \text{otherwise} \end{cases} \tag{20}$$

that is the system k fully meets the performance requirement on the capabilities that it is capable of providing. The system agent adds a random nonnegative number above the minimum additional fund needed in order to not share the private information with the SoS. If the additional fund needed is zero, the agent will still ask for an additional funding equal to a small portion of the provided funding. On those capabilities that the system k is not capable of providing, the agent asks for a very large amount of additional funding equal to Mf_i. Therefore,

$$Systemk.\Delta f_i = \begin{cases} \Delta f_i^*(1+\theta_i) & i \in I_{reqfsbl} \text{ and if } \Delta f_i^* \neq 0 \\ \theta_i f_i & i \in I_{reqfsbl} \text{ and if } \Delta f_i^* = 0 \\ M f_i & i \in I_{reqifsbl} \\ 0 & i \in \overline{I}_{req} \end{cases} \tag{21}$$

The agent adds a random nonnegative integer above the minimum time in order to not share the private information with the SoS. If the system k is not capable of providing a capability, then the agent asks for a very long additional time,

$$\Delta d_i = \begin{cases} \Delta d_i^* + \varepsilon_i & i \in I_{reqfsbl} \\ M d_i & i \in I_{reqifsbl} \\ 0 & i \in \overline{I}_{req} \end{cases} \tag{22}$$

Again, since the system *k* does not want to let the SoS know what capabilities it is not capable of providing, the system *k* does not update the C_i's and χ_{ji}'s.

VARIATION OF THE INDIVIDUAL SYSTEM

Technically, we can produce a family of models by modifying the model parameters. An excel file (setting_systemk.xlsx) is created as an input file for the matlab code of this model, which lists the model parameters that can be modified to create a family of models with different characteristics and behaviors. In the following these parameters, and the way of selecting values for these parameters, are discussed in sequence.

- y: throughput (units of capability produced per time unit per resource unit). It is an nonnegative real value vector of *n* elements. "y(i)=0" means the individual system k is not capable of providing capability i. The greater the y(i), the more efficient the system k in building the capability i.
- Resource per time unit (assuming a symmetric distribution of resource)
    - Z_avg: a positive real number. The greater the Z_avg, the higher the average resource of the system k.
    - Z_range: X_range is a positive real number and no greater than 200% of Z_avg. The greater the Z_avg, the more volatile the resource of system k.
- Costs
    - c_c: the cost of consuming one unit of resource per time unit in producing capabilities. It is a nonnegative vector of n elements. "c_c(i)= 0" if "y(i)=0". The greater the c_c(i), the more expensive for the system k to provide capability i.
    - c_I: the cost of consuming one unit of resource per time unit in producing interfaces, as a percentage of c_c. It is a vector of n elements. "c_I(i) = 0" if "y(i)=0". The greater

the c_I(i), the more expensive for the system k to provide an interface with other systems on capability i.

  - growth_rate: the continuous growth rate per time unit. It must be positive to ensure that the additional time needed is minimized.
  - pm: the required profit margin of capabilities. pm describes the minimum required rate of return from providing capabilities. It is a vector of n elements. "pm(i)=0" if "y(i)=0". pm is nonnegative. It is the primary parameter for modeling the utility function of the system k.

- If: pm(i) = 0, the system k would like to collaborate with SoS in providing capability i without an attempt to make profit (but losing profit is not acceptable).
- Otherwise: the system k would like to collaborate with SoS in providing capability i only if the minimum required rate of return is met (i.e. making a minimum level of profit is required from the collaboration). The greater the pm, the "greedier" the system k is.
- CriticalPro: the probability of sending the P1 result to SoS, and (1-CriticalPro) is the probability of sending the P2 result to SoS. CriticalPro is within the range of [0,1], including the two boudnaries. Use CriticalPro to control the way of negotiation: (P1) or (P2). There are two extreme scenarios:
- If: CriticalPro = 0, the result of (P2) is sent to SoS;
- If: CriticalPro = 1, the result of (P1) is sent to SoS;
- Notes:
- (P1) tells the best performance of the system k at the given deadlines and funding.
- (P2) tells the additional funding needed and additional time needed in order to meet the performance requirements (if the system k is capable).
- In (P2) the additional funding needed may be greater than the minimum additional funding needed; similarly, the additional time needed may be longer than the minimum additional time needed. This is a strategy that the system k uses in the negotiation and protecting its private information. Please refer to parameters AF1, AF2, and AT for the details.
- Negotiation Parameters for (P2)
  - AF1: If the system k will use up all funding to be provided by SoS, it requests up to AF1 more than the additional funding needed. AF1 is a nonnegative real number. The greater the AF1, the greater the profit, and the failure rate as well, from the negociation with SoS.
  - AF2: if the system k will not use up all funding to be provided by SoS, it requests up to AF2 more than the funding provided by the SoS. AF2 is a nonnegative real number. The greater the AF2, the higher the failure rate and profit from the negotiation with SoS.
  - AT: the system k request up to AT units of time more than the minimum additional time needed. AT is a nonnegative integer.
- If: AT=0, only the minimum additional time needed is requested.
- else: request up to AT units of time more than the additional time needed.
- The greater the AT, the higher the failure rate and time flexibility from the negotiation with SoS.

The table below illustrates the systems selected in the meta-architecture in the first column and the corresponding capability they possess in the second column. This table helps the reader to understand the negotiations that proceed after the meta-architecture generation process. The tables below explain the offer made by the SoS agent based on the meta-architecture to the individual systems. The offer consists of three attributes namely funding, performance and deadline. These attributes are independent of each other. The funding is expressed in million dollars, performance in sq.km of area covered and deadline is measured as waves in the overall SoS architecting process. The concept of wave is similar to the concept of era in epoch-era analysis. Epoch-Era Analysis is an approach for describing systems over time as existing in a series of static contexts (epochs) that change stochastically. Many epochs constitute an era. The results of the first wave are presented as 22 tables. Each table belongs to a particular system consisting of the values of three attributes that are demanded or provided by SoS. Each system as described above can provide only a certain set of capabilities; hence, the rows of capabilities, which cannot be provided by the particular system, are empty.

Systems Selected for Negotiation (11)	Capabilities Possessed
System 1,2,7	(1 and 5)
System 8	(1) only
System 11,12, 13	(2 and 5)
System 14	(3 and 5)
System 18	(4 and 5)
System 21, 22	(5) only

Table 2 Selfish Negotiation Model Results for System 1 (Accepted by SoS)

The Offer by SoS to Sys 1	Capabilities	Deadline SoS.d_i	Funding SoS.f_i	Performance SoS.p_i	System 1 response: Accepted by SoS to provide capabilities C1 and C5	Deadline Systemj d_i	Funding Sustemj f_i	Performance Systemj p_i
	C1	1	10.2	13.5		1	0.48815514	0
	C2	0	0	0		0	0	0
	C3	0	0	0		0	0	0
	C4	0	0	0		0	0	0
	C5	1	10.55	16		0	0.25603565	0

The table above contains information regarding a bilateral negotiation between SoS and System 1. System 1 can provide Capability 1 (C1) and/or (C2) but no more. The values in the first row corresponding to C1 indicate that SoS requires System 1 to join in the first wave of the SoS architecture. SoS is providing a funding of 10.2 units to acquire C1 and demands performance level of 13.5 units for the same. The response to this offer of SoS can be read in the adjacent columns consisting of delta values of attributes.

The delta values correspond to $(delta = System\,response - \; SoS\,offer)$. Hence a positive value of "1"in deadline means that System is not ready to participate in the first wave but will be ready by the second wave. Equivalently the funding delta value is positive 0.488 meaning the System 1 has asked for an additional amount from SoS for providing C1. The Performance delta value is zero, which can be interpreted as the System 1 prepared to provide the required performance levels.

Similarly for C5, SoS asks the System 1 to join in the first wave, has provision of 10.55 units of funding to extract a performance of 16 units. The response is recorded as delta change in the adjacent columns. Based on the delta values of each attributes and the SoS negotiation thresholds, System 1 is accepted to be a part of the first wave of the architecting process.

Table 3 Selfish Negotiation Model Results for System 2 (Accepted by SoS)

The Offer by SoS to Sys 2	Capabilities	Deadline $SoS.d_i$	Funding $SoS.f_i$	Performance $SoS.p_i$	System 2 response: Accepted by SoS to provide capabilities C1 and C5	Deadline $Systemj\ d_i$	Funding $Sustemj\ f_i$	Performance $Systemj\ p_i$
	C1	**1**	**10.1**	**11**		**0**	**0**	**-2.13163E-14**
	C2	1	0	0		0	0	0
	C3	**0**	**0**	**0**		**0**	**0**	**0**
	C4	**1**	**0**	**0**		**0**	**0**	**0**
	C5	1	10.2	12		0	0	-7.10543E-15

Table 4 Selfish Negotiation Model Results for System 7 (Accepted by SoS)

The Offer by SoS to Sys 7	Capabilities	Deadline $SoS.d_i$	Funding $SoS.f_i$	Performance $SoS.p_i$	System 7 response: Accepted by SoS to provide capabilities C1 and C5	Deadline $Systemj\ d_i$	Funding $Sustemj\ f_i$	Performance $Systemj\ p_i$
	C1	**1**	**2.2**	**13.5**		**1**	**0.08714281**	**0**
	C2	1	0	1		0	0	0
	C3	**1**	**0**	**0**		**0**	**0**	**0**
	C4	**1**	**0**	**0.5**		**0**	**0**	**0**
	C5	1	2.45	16		0	0.11753782	0

Table 5 Selfish Negotiation Model Results for System 8 (Accepted by SoS)

The Offer by SoS to Sys 8	Capabilities	Deadline $SoS.d_i$	Funding $SoS.f_i$	Performance $SoS.p_i$	System 8 response: Accepted by SoS to provide capabilities C1 and C5	Deadline $Systemj\ d_i$	Funding $Sustemj\ f_i$	Performance $Systemj\ p_i$
	C1	**1**	**15**	**5**		**1**	**0.70049494**	**0**
	C2	1	0	0.5		**0**	**0**	**0**
	C3	**0**	**0**	**0.5**		**0**	**0**	**0**
	C4	**1**	**0**	**0**		**0**	**0**	**0**
	C5	1	0.25	2		**0**	**0**	**0**

Table 6 Selfish Negotiation Model Results for System 11 (Accepted by SoS)

The Offer by SoS to Sys 11	Capabilities	Deadline SoS.d_i	Funding SoS.f_i	Performance SoS.p_i	System 11 response: Accepted by SoS to provide capabilities C1 and C5	Deadline Systemj d_i	Funding Sustemj f_i	Performance Systemj p_i
	C1	**1**	**0**	**1.5**		**0**	**0**	**0**
	C2	1	10.7	19.5		1	0.39757587	0
	C3	**1**	**0**	**0.5**		**0**	**0**	**0**
	C4	**1**	**0**	**0.5**		**0**	**0**	**0**
	C5	1	11.2	22.5		0	0.21964713	0

Table 7 Selfish Negotiation Model Results for System 12 (Accepted by SoS)

The Offer by SoS to Sys 12	Capabilities	Deadline SoS.d_i	Funding SoS.f_i	Performance SoS.p_i	System 12 response: Accepted by SoS to provide capabilities C1 and C5	Deadline Systemj d_i	Funding Sustemj f_i	Performance Systemj p_i
	C1	**1**	**0**	**1**		**0**	**0**	**0**
	C2	1	10.35	18.5		0	0.0164735	0
	C3	**0**	**0**	**0**		**0**	**0**	**0**
	C4	**0**	**0**	**0.5**		**0**	**0**	**0**
	C5	1	10.6	21		0	0.14676918	0

Table 8 Selfish Negotiation Model Results for System 13 (Accepted by SoS)

The Offer by SoS to Sys 13	Capabilities	Deadline SoS.d_i	Funding SoS.f_i	Performance SoS.p_i	System 13 response: Rejected by SoS to provide capabilities C1 and C5	Deadline Systemj d_i	Funding Sustemj f_i	Performance Systemj p_i
	C1	**0**	**0**	**0**		**0**	**0**	**0**
	C2	1	18	41		**2**	**0.62534576**	**0**
	C3	**0**	**0**	**0**		**0**	**0**	**0**
	C4	**0**	**0**	**0**		**0**	**0**	**0**
	C5	1	18	42		**1**	**0.28538953**	**0**

The system13 can meet the performance requirement on the capability C2 and C5 but requests an extension of deadline for 2nd wave on C2 and an additional funding of 0.6235 units. Since the System is not ready to participate in in the current wave and needs more time (upto the 3rd wave of SoS) the SoS agent rejects the offer of negotiation.

Table 9 Selfish Negotiation Model Results for System 14 (Accepted by SoS)

The Offer by SoS to Sys 14	Capabilities	Deadline $SoS.d_i$	Funding $SoS.f_i$	Performance $SoS.p_i$	System 14 response: Accepted by SoS to provide capabilities C1 and C5	Deadline Systemj d_i	Funding Sustemj f_i	Performance Systemj p_i
	C1	**1**	**0**	**0.5**		**0**	**0**	**-0.5**
	C2	1	0	0.5		0	0	-0.5
	C3	**1**	**13**	**12**		**0**	**0**	**0**
	C4	**1**	**0**	**0**		**0**	**0**	**0**
	C5	1	13.45	13.5		0	0	0

Table 10 Selfish Negotiation Model Results for System 18 (Accepted by SoS)

The Offer by SoS to Sys 18	Capabilities	Deadline $SoS.d_i$	Funding $SoS.f_i$	Performance $SoS.p_i$	System 18 response: Accepted by SoS to provide capabilities C1 and C5	Deadline Systemj d_i	Funding Sustemj f_i	Performance Systemj p_i
	C1	**1**	**0**	**1**		**0**	**0**	**-1**
	C2	1	0	1		0	0	-1
	C3	**1**	**0**	**0**		**0**	**0**	**0**
	C4	**1**	**2.5**	**15**		**0**	**0**	**-1.95399E-14**
	C5	1	2.75	18		0	0	3.55271E-15

Table 11 Selfish Negotiation Model Results for System 21 (Accepted by SoS)

The Offer by SoS to Sys 21	Capabilities	Deadline $SoS.d_i$	Funding $SoS.f_i$	Performance $SoS.p_i$	System 21 response: Accepted by SoS to provide capabilities C1 and C5	Deadline Systemj d_i	Funding Sustemj f_i	Performance Systemj p_i
	C1	**1**	**1**	**2**		**0**	**0**	**0**
	C2	1	0.7	1.5		0	0	0
	C3	**0**	**0**	**0.5**		**0**	**0**	**0**
	C4	**1**	**1**	**0.5**		**0**	**0**	**0**
	C5	1	7.2	19.5		0	0.28627196	0

Table above implies that no changes are required for the deadlines and performance for capability 5. Overall, we also see reduced needs for funding.

Table 12 Selfish Negotiation Model Results for System 22 (Accepted by SoS)

The Offer by SoS to Sys 22	Capabilities	Deadline $SoS.d_i$	Funding $SoS.f_i$	Performance $SoS.p_i$	System 21 response: Accepted by SoS to provide capabilities C1 and C5	Deadline $Systemj\ d_i$	Funding $Sustemj\ f_i$	Performance $Systemj\ p_i$
	C1	**0**	**0**	**2**		**0**	**0**	**-2**
	C2	1	0.35	1.5		0	0	-1.5
	C3	**1**	**1**	**0.5**		**0**	**0**	**-0.5**
	C4	**0**	**0**	**0.5**		**0**	**0**	**-0.5**
	C5	1	5.45	19.5		0	0	-1.29958E-11

After the end of the first wave only System 13 is rejected due to the reasons give above. All other systems are selected and Since System 13 did not have any interface with the other participating systems, the interface matrix does not get affected as well.

Final Architecture is shown in the figure below:

S_1	S_2	S_3	S_4	S_5	S_6	S_7	S_8	S_9	S_{10}	S_{11}	S_{12}	S_{14}	S_{15}	S_{16}	S_{17}	S_{18}	S_{19}	S_{20}	S_{21}	S_{22}
1	1	0	0	0	0	1	1	0	0	1	1	1	0	0	0	1	0	0	1	1

The interface architecture remains the same as before. This model is used in two other notional SoS namely; Search and Rescue and in a toy problem for assessing the performance of the aircraft carrier. The results of these applications are given in report Volume I.

Concluding Remarks

In this research, a negotiation model is developed for non-cooperative systems in negotiating multiple issues on multiple items with the SoS manager. Optimization techniques are introduced to the negotiation decision making to help determine the best alternative of negotiation.

The current negotiation model is simplified. An immediate extension of the current model is to develop a more rigorous decision framework to support the complex contract negotiation. For example, the current model includes only limited a few alternatives for negotiation. How to identify all efficient alternatives is a research question to be answered in the future work. Also, the preferences of the counterparty should be modeled in a more precise manner. All these are subjects of future work.

APPENDIX A: LIST OF PUBLICAS RESULTED AND PAPERS SUBMITTED FROM FILA-SOS RESEARCH

Wang, R., Agarwal,S., & Dagli, C. (2014). Executable System of Systems Architecture Using OPM in Conjunction with Colored Petri Net: A Module for Flexible Intelligent & Learning Architectures for System of Systems, In *Europe Middle East & Africa Systems Engineering Conference (EMEASEC)*.

Ergin, N. K.,(2014), Improving Collaboration in Search and Rescue System of Systems, *Procedia Computer Science, Volume 36*, Pages 13-20.

Agarwal, S., & Dagli, C. H. (2013). Augmented Cognition in Human–System Interaction through Coupled Action of Body Sensor Network and Agent Based Modeling. *Procedia Computer Science*, *16*, 20-28.

Acheson, P., Dagli, C., & Kilicay-Ergin, N. (2013). Model Based Systems Engineering for System of Systems Using Agent-based Modeling. *Procedia Computer Science*, 16, 11-19.

Agarwal, S., Pape, L. E., & Dagli, C. H. (2014). A Hybrid Genetic Algorithm and Particle Swarm Optimization with Type-2 Fuzzy Sets for Generating Systems of Systems Architectures. *Procedia Computer Science, 36,* 57-64.

Agarwal, S., Pape, L. E., Kilicay-Ergin, N., & Dagli, C. H. (2014). Multi-agent Based Architecture for Acknowledged System of Systems. *Procedia Computer Science, 28*, 1-10.

Agarwal, S., Saferpour, H. R., & Dagli, C. H. (2014). Adaptive Learning Model for Predicting Negotiation Behaviors through Hybrid K-means Clustering, Linear Vector Quantization and 2-Tuple Fuzzy Linguistic Model. *Procedia Computer Science, 36,* 285-292.

Agarwal,S., Wang, R., & Dagli, C., (2015) FILA-SoS, Executable Architectures using Cuckoo Search Optimization coupled with OPM and CPN-A module: A new Meta-Architecture Model for FILA-SoS, France, Complex Systems Design & Management (CSD&M) editor, Boulanger, Frédéric, Krob, Daniel, Morel, Gérard, Roussel, Jean-Claude, P 175-192 . Springer International Publishing.

Pape, L., Agarwal, S., Giammarco, K., & Dagli, C. (2014). Fuzzy Optimization of Acknowledged System of Systems Meta-architectures for Agent based Modeling of Development. *Procedia Computer Science*, *28*, 404-411.

Pape, L., & Dagli, C. (2013). Assessing robustness in systems of systems meta-architectures. *Procedia Computer Science*, *20*, 262-269.

Pape, L., Giammarco, K., Colombi, J., Dagli, C., Kilicay-Ergin, N., & Rebovich, G. (2013). A fuzzy evaluation method for system of systems meta-architectures. *Procedia Computer Science*, *16*, 245-254.

Acheson, P., Dagli, C., & Kilicay-Ergin, N. (2013). Model Based Systems Engineering for System of Systems Using Agent-based Modeling. *Procedia Computer Science*, *16*, 11-19.

Acheson, P., Dagli, C., & Kilicay-Ergin, N. (2014). Fuzzy Decision Analysis in Negotiation between the System of Systems Agent and the System Agent in an Agent-Based Model. *arXiv preprint arXiv:1402.0029*.

Kilicay-Ergin, N. H., Acheson, P., Colombi, J. M., & Dagli, C. H. (2012). Modeling system of systems acquisition. In *SoSE* (pp. 514-518).

Acheson, P., Pape, L., Dagli, C., Kilicay-Ergin, N., Columbi, J., & Haris, K. (2012). Understanding System of Systems Development Using an Agent-Based Wave Model. *Procedia Computer Science*, *12*, 21-30.

Konur, D., & Dagli, C. (2014). Military system of systems architecting with individual system contracts*. Optimization Letters*, 1-19.

Dagli et al., 2015 Flexible and Intelligent Learning Architectures for SoS (FILA-SoS): Architectural evolution in Systems-of-Systems, 2015 Conference on Systems Engineering Research.

Ergin, D., & Dagli, C., Incentive Based Negotiation Model for System of Systems Acquisition. (Accepted by Systems Engineering Journal)

Wang, R., & Dagli, C., Search Based Systems Architecture Development Using Holistic Approach (Accepted to IEEE Systems Journal with minor revisions)

APPENDIX B: CITED AND RELATED REFERENCES

Abo-Sinna, M. A., & Baky, I. A. (2007). Interactive Balance Space Approach for Solving Multi-Level Multi-Objective Programming Problems. *Information Sciences, 177(16)*, 3397-3410.

Abraham, A., & Jain, L. (2005). *Evolutionary Multi-objective Optimization* (pp. 1-6). Springer London.

Acheson, P. (2010, April). Methodology for Object-Oriented System Architecture Development. In *Systems Conference, 2010 4th Annual IEEE* (pp. 643-646). IEEE.

Acheson, P., Dagli, C., & Kilicay-Ergin, N. (2014). Fuzzy Decision Analysis in Negotiation between the System of Systems Agent and the System Agent in an Agent-Based Model. *arXiv preprint arXiv:1402.0029*.

Acheson, P., Pape, L., Dagli, C., Kilicay-Ergin, N., Columbi, J., & Haris, K. (2012). Understanding System of Systems Development Using an Agent-Based Wave Model. *Procedia Computer Science*, *12*, 21-30.

Agarwal, S., & Dagli, C. H. (2013). Augmented Cognition in Human–System Interaction through Coupled Action of Body Sensor Network and Agent Based Modeling. *Procedia Computer Science*, *16*, 20-28.

Acheson, P., Dagli, C., & Kilicay-Ergin, N. (2013). Model Based Systems Engineering for System of Systems Using Agent-based Modeling. *Procedia Computer Science*, 16, 11-19.

Agarwal, S., Pape, L. E., & Dagli, C. H. (2014). A Hybrid Genetic Algorithm and Particle Swarm Optimization with Type-2 Fuzzy Sets for Generating Systems of Systems Architectures. *Procedia Computer Science, 36,* 57-64.

Agarwal, S., Pape, L. E., Kilicay-Ergin, N., & Dagli, C. H. (2014). Multi-Agent Based Architecture for Acknowledged System of Systems. *Procedia Computer Science, 28*, 1-10.

Agarwal, S., Saferpour, H. R., & Dagli, C. H. (2014). Adaptive Learning Model for Predicting Negotiation Behaviors through Hybrid K-means Clustering, Linear Vector Quantization and 2-Tuple Fuzzy Linguistic Model. *Procedia Computer Science, 36,* 285-292.

Agarwal,S., Wang, R., & Dagli, C., (2015) FILA-SoS, Executable Architectures using Cuckoo Search Optimization coupled with OPM and CPN-A module: A new Meta-Architecture Model for FILA-SoS, France, Complex Systems Design & Management (CSD&M) editor, Boulanger, Frédéric, Krob, Daniel, Morel, Gérard, Roussel, Jean-Claude, P 175-192 . Springer International Publishing.

Ahn, J. H., Ryu, Y., & Baik, D. K. (2012). An Archietcture Description method for Acknowledged System of Systems based on Federated Architeture. *Advanced Science and Technology Letters*, *5*.

Alberts, D. S., (2011). *The Agility Advantage: A Survival Guide for Complex Enterprises and Endeavors*. Washington DC: Center for Advanced Concepts and Technology.

Alberts, D. S., Garstka, J. J., & Stein, F. P. (1999). {Network Centric Warfare: Developing and Leveraging Information Superiority}.

Alfaris, A. A. F. (2009). *The Evolutionary Design Model (EDM) for the Design of Complex Engineered Systems: Masdar City as a Case Study* (Doctoral dissertation, Massachusetts Institute of Technology).

Alves, M. J., Dempe, S., & Júdice, J. J. (2012). Computing the Pareto Frontier of a Bi-objective Bi-level Linear Problem using a Multi-objective Mixed-integer Programming Algorithm. *Optimization*, *61*(3), 335-358.

Amberg, M. (1996, October). Modeling Adaptive Workflows in Distributed Environments. In *Proc. of the 1st Int. Conf. on Practical Aspects of Knowledge Management, Basel, 30th-31th Oct*.

Anandalingam, G., & Friesz, T. L. (1992). Hierarchical Optimization: an Introduction. *Annals of Operations Research*, *34*(1), 1-11.

ASD(NII), D. (2010). *DoD Architecture Framework Version 2.02 (DoDAF v2.02)*. Washington DC: Department of Defense.

AT&L, O. U. S. D. (2008). Systems Engineering Guide for Systems of Systems. *Washington, DC: Pentagon*.

Arnold, A., Boyer, B., & Legay, A. (2013). Contracts and Behavioral Patterns for SoS: The EU IP DANSE Approach. *arXiv preprint arXiv:1311.3631*.

Bac, M., & Raff, H. (1996). Issue-by-issue Negotiations: the Role of Information and Time Preference. *Games and Economic Behavior*, *13*(1), 125-134.

Baky, I. A. (2009). Fuzzy Goal Programming Algorithm for Solving De-centralized Bi-level Multi-objective Programming Problems. *Fuzzy Sets and Systems*, *160*(18), 2701-2713.

Baky, I. A. (2010). Solving Multi-level Multi-objective Linear Programming Problems through Fuzzy Goal Programming Approach. *Applied Mathematical Modelling*, *34*(9), 2377-2387.

Bergey, J. K., Blanchette Jr, S., Clements, P. C., Gagliardi, M. J., Klein, J., Wojcik, R., & Wood, W. (2009). US Army Workshop on Exploring Enterprise, System of Systems, System, and Software Architectures.

Blanchard, B. S., & Fabryeky, W. J. (2010). *Systems Engineering and Analysis*. Upper Saddle River, NJ: Prentice Hall.

Bonabeau, E. (2002). Agent-based modeling: Methods and Techniques for Simulating Human Systems. *Proceedings of the National Academy of Sciences of the United States of America*, *99 (Suppl 3)*, 7280-7287.

Bouleimen, K. L. E. I. N., & Lecocq, H. O. U. S. N. I. (2003). A New Efficient Simulated Annealing Algorithm for the Resource-constrained Project Scheduling Problem and Its Multiple Mode Version. *European Journal of Operational Research*, *149*(2), 268-281.

Bradley, S. P., Hax, A. C., & Magnanti, T. L. Applied Mathematical Programming. 1977.

Brucker, P., Drexl, A., Möhring, R., Neumann, K., & Pesch, E. (1999). Resource-constrained Project Scheduling: Notation, Classification, Models, and Methods. *European Journal of Operational Research*, *112*(1), 3-41.

Cara, A. B., Wagner, C., Hagras, H., Pomares, H., & Rojas, I. (2013). Multi-objective Optimization and Comparison of Non-singleton type-1 and Singleton interval type-2 Fuzzy Logic Systems. *Fuzzy Systems, IEEE Transactions on*, *21*(3), 459-476.

Chattopadhyay, D., Ross, A. M., & Rhodes, D. H. (2008, April). A Framework for Trade-space Exploration of Systems of Systems. In *6th Conference on Systems Engineering Research*, Los Angeles, CA.

Christian III, J. A. (2004). *A Quantitative Approach to Assessing System Evolvability*. Houston: NASA Johnson Space Center.

CJCSI 6212.01F. (12 Mar 2012). *Net Ready Key Performance Parameter (NR KPP).* Washington DC: US Dept of Defense.

Clouthier, R. J., Diamrio, M. J., & Polzer, H. W. (2009). Net Centricity and System of Systems. In M. Jamshidi, *System of Systems Engineering* (pp. 150-168). Hoboken NJ: John Wiley & Sons.

Clune, J., Mouret, J. B., & Lipson, H. (2013). The Evolutionary Origins of Modularity. *Proceedings of the Royal Society b: Biological sciences*, *280 (1755)*, 20122863.

Coello, C. A. C., & Lamont, G. B. (2004). *Applications of Multi-objective Evolutionary Algorithms* (Vol. 1). World Scientific.

Cohon, J. L. (1985). Multi-criteria Programming: Brief Review and Application. *Design optimization*, 163.

Coleman, J. W., Malmos, A. K., Larsen, P. G., Peleska, J., & Hains, R. (2012). COMPASS Tool Vision for a System of Systems Collaborative Development Environment. In *International Conference on System of Systems Engineering*.

Contag, G., Laing, C., Pabon, J., Rosenberg, E., Tomasino, K., & Tonello, J. (2013). Nighthawk System Search and Rescue (SAR) Unmanned Vehicle (UV) System Development. *SE4150 Design Project, Naval Postgraduate School*.

Cox Jr., L. A. (2009). *Risk Analysis of Complex and Uncertain Systems (Vol. 129)*. US: Springer-Verlag.

Crossley, W. A., & Laananen, D. H. (1996). Conceptual Design of Helicopters via Genetic Algorithm. *Journal of Aircraft*, *33*(6), 1062-1070.

Dagli, C. H., & Kilicay-Ergin, N. (2008). System of Systems Architecting. Jamshidi, M. (Ed.). In *System of Systems Engineering: Innovations for the Twenty-first Century (Vol. 58). John Wiley & Sons.*, 77-100.

Dagli, C., Ergin, N., Enke, D., Gosavi, A., Qin, R., Colombi, J.,Agarwal,S., ... & Pape, L. (2013). *An Advanced Computational Approach to System of Systems Analysis & Architecting Using Agent-Based Behavioral Model* (No. SERC-2013-TR-021-2). MISSOURI UNIV OF SCIENCE AND TECHNOLOGY ROLLA.

Dagli, C. H., Singh, A., Dauby, J. P., & Wang, R. (2009, December). Smart Systems Architecting: Computational Intelligence Applied to Trade Space Exploration and System Design. In *Systems Research Forum* (Vol. 3, No. 02, pp. 101-119). World Scientific Publishing Company.

Dahmann, J., Lane, J., Rebovich, G., & Baldwin, K. (2008, April). A Model of Systems Engineering in a System of Systems Context. In *Proceedings of the Conference on Systems Engineering Research, Los Angeles, CA, USA*.

Dahmann, J., Baldwin, K. J., & Rebovich Jr, G. (2009, April). Systems of Systems and Net-Centric Enterprise Systems. In *7th Annual Conference on Systems Engineering Research, Loughborough*.

Dahmann, J., Rebovich, G., Lowry, R., Lane, J., & Baldwin, K. (2011, April). An Implementers' View of Systems Engineering for Systems of Systems. In *Systems Conference (SysCon), 2011 IEEE International* (pp. 212-217). IEEE.

Dahmann, J. S., & Baldwin, K. J. (2008, April). Understanding the Current State of US Defense Systems of Systems and the Implications for Systems Engineering. In *Systems Conference, 2008 2nd Annual IEEE* (pp. 1-7). IEEE.

Dahmann, J. (2012). INCOSE SoS Working Group Pain Points. In *Proc TTCP-JSA-TP4 Meeting*. Retrieved from http://www.ndia.org/Divisions/Divisions/SystemsEngineering/Documents/NDIA-SE-MS-SoS_2013-08-20_Dahmann.pdf

Dauby, J. P., & Dagli, C. H. (2011). The Canonical Decomposition Fuzzy Comparative Methodology for Assessing Architectures. *Systems Journal, IEEE*,*5*(2), 244-255.

Dauby, J. P., & Upholzer, S. (2011). Exploring Behavioral Dynamics in Systems of Systems. *Procedia Computer Science*, *6*, 34-39.

Deb, K. (2000). An Efficient Constraint Handling Method for Genetic Algorithms. *Computer Methods in Applied Mechanics and Engineering*, *186*(2), 311-338.

Deb, K. (2001). *Multi-objective Optimization using Evolutionary Algorithms* (Vol. 16). John Wiley & Sons.

Deb, K., & Gupta, H. (2006). Introducing Robustness in Multi-objective Optimization. *Evolutionary Computation*, *14*(4), 463-494.

Deb, K., & Sinha, A. (2009, January). Solving Bi-level Multi-objective Optimization Problems using Evolutionary Algorithms. In *Evolutionary Multi-Criterion Optimization* (pp. 110-124). Springer Berlin Heidelberg.

DeLaurentis, D., Marais, K., Davendralingam, N., Han, S. Y., Uday, P., Fang, Z., & Gurainiello, C. (2012). *Assessing the Impact of Development Disruptions and Dependencies in Analysis of Alternatives of System of Systems*. Hoboken NJ: Stevens Institute of Technology, Systems Engineering Research Center,

Department of the Navy, (1997). *Contractor Performance Assessment Reporting System (CPARS)*. Washington DC.

Díaz, E., Tuya, J., & Blanco, R. (2003, October). Automated Software Testing Using a Metaheuristic Technique based on Tabu Search. In *Automated Software Engineering, 2003. Proceedings. 18th IEEE International Conference on* (pp. 310-313). IEEE.

Díaz, E., Tuya, J., Blanco, R., & Javier Dolado, J. (2008). A Tabu Search Algorithm for Structural Software Testing. *Computers & Operations Research*,*35*(10), 3052-3072.

Director Systems and Software Engineering, OUSD (AT&L). (2008). *Systems Engineering Guide for Systems of Systems*. Available from http://acq.osd.mil/se/doc/SE-Guid-for-SoS.pdf.

Djavanshir, G. R., Alavizadeh, A., & Tarokh, M. J. (2012). From System-of-Systems to Meta-Systems: Ambiguities and Challenges. *System of Systems*.

DoD, A. S. D. "DoD Architecture Framework Version 2.0 (DoDAF V2. 0)."*Department of Defense, Washington DC* (2009).

DoD, Navy. "Contractor Performance Assessment Reporting System (CPARS)," Washington DC (1997).

Dolado, J. J. (2000). A Validation of the Component-based Method for Software Size Estimation. *Software Engineering, IEEE Transactions on*, *26*(10), 1006-1021.

Dolado, J. J. (2001). On the Problem of the Software Cost Function. *Information and Software Technology*, *43*(1), 61-72.

Dombkins, D. (1996). *Project Managed Change: The Application of Project Management Techniques to Strategic Change Programs*. Centre for Corporate Change, Australian Graduate School of Management, University of New South Wales.

Dombkins, D. (2007). Complex Project Management. *South Carolina: Booksurge Publishing*.

Dombkins, D. H. (2013).Realizing Complex Policy: Using a Systems-of-Systems Approach to Develop and Implement Policy. *Editor's Introduction*, *Volume II, Issue 5,* 22.

Dudek, G., Jenkin, M. R., Milios, E., & Wilkes, D. (1996). A Taxonomy for Multi-agent Robotics. *Autonomous Robots*, *3*(4), 375-397.

Dudenhoeffer, D. D., & Jones, M. P. (2000). A Formation Behavior for Large-scale Micro-robot Force Deployment. In *Simulation Conference, 2000. Proceedings. Winter* (Vol. 1, pp. 972-982). IEEE.

Dutta, P. K. (1999). *Strategies and Games: Theory and Practice*. MIT Press.

Eichfelder, G. (2010). Multi-objective Bi-level Optimization. *Mathematical Programming*, *123*(2), 419-449.

Epperly, T. G. W. (1995). *Global Optimization of Non-convex Non-linear Programs using Parallel Branch and Bound* (Doctoral dissertation, UNIVERSITY OF WISCONSIN–MADISON).

Ergin, N. K.,(2014), Improving Collaboration in Search and Rescue System of Systems, *Procedia Computer Science, Volume 36*, Pages 13-20.

Faratin, P., Sierra, C., & Jennings, N. R. (1998). Negotiation Decision Functions for Autonomous Agents. *Robotics and Autonomous Systems*, *24*(3), 159-182.

Farmani, R., & Wright, J. A. (2003). Self-Adaptive Fitness Formulation for Constrained Optimization. *Evolutionary Computation, IEEE Transactions on*, *7*(5), 445-455.

Fatima, S. S., Wooldridge, M., & Jennings, N. R. (2004). An Agenda-based Framework for Multi-issue Negotiation. *Artificial Intelligence*, *152*(1), 1-45.

Flanigan, D., & Brouse, P. (2012). System of Systems Requirements Capacity Allocation. *Procedia Computer Science*, *8*, 112-117.

Fogel, D. B. (2006). *Evolutionary Computation: Toward a new Philosophy of Machine Intelligence* (Vol. 1). John Wiley & Sons.

Fonseca, C. M., & Fleming, P. J. (1995). An Overview of Evolutionary Algorithms in Multi-objective Optimization. *Evolutionary Computation*, *3*(1), 1-16.

Fonseca, C. M., & Fleming, P. J. (1998). Multi-objective Optimization and Multiple Constraint Handling with Evolutionary Algorithms. I: A Unified Formulation. *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on*, *28*(1), 26-37.

Fry, D. N., & DeLaurentis, D. A. (2011, June). Measuring Net-centricity. In *System of Systems Engineering (SoSE), 2011 6th International Conference on*(pp. 264-269). IEEE.

Gao, J., Buldyrev, S. V., Stanley, H. E., & Havlin, S. (2011, January). Networks Formed from Interdependent Networks. *Nature Physics, 8,* 40-48, doi:10.1038/NPHYS2180.

Gao, Y., Zhang, G., & Lu, J. (2009). A Fuzzy Multi-objective Bi-level Decision Support System. *International Journal of Information Technology & Decision Making*, *8*(01), 93-108.

Garrett, R. K., Anderson, S., Baron, N. T., & Moreland, J. D. (2011). Managing the Interstitials, a System of Systems Framework Suited for the Ballistic Missile Defense System. *Systems Engineering*, *14*(1), 87-109.

Garvey, P., & Pinto, A. (2009, June). Introduction to Functional Dependency Network Analysis. In *The MITRE Corporation and Old Dominion, Second International Symposium on Engineering Systems, Massachusetts Institute of Technology, Cambridge, Massachusetts*.

Ge, B., Hipel, K. W., Yang, K., & Chen, Y. (2013). A Data-centric Capability-focused Approach for System-of-systems Architecture Modeling and Analysis. *Systems Engineering*, *16*(3), 363-377.

Gegov, A. (2010). *Fuzzy Networks for Complex Systems A Modular Rule Base Approach*. Springer-Verlag Berlin Heidelberg.

Giachetti, R. E. (2012). A Flexible Approach to Realize an Enterprise Architecture. *Procedia Computer Science*, *8*, 147-152.

Glover, F. (1989). Tabu Search – Part I. *ORSA Journal on Computing*, *1*(3), 190-206.

Glover, F. (1990). Tabu Search – Part II. *ORSA Journal on Computing*, *2*(1), 4-32.

Goldberg, D. E. (1990). Genetic Algorithms in Search, Optimization and Machine Learning. *Reading: Addison-Wesley*.

Gonzalez-Zugasti, J. P., Otto, K. N., & Baker, J. D. (2000). A Method for Architecting Product Platforms. *Research in Engineering Design*, *12*(2), 61-72.

Gries, M. (2004). Methods for Evaluating and Covering the Design Space during Early Design Development. *Integration, the VLSI journal*, *38*(2), 131-183.

Guariniello, C., & DeLaurentis, D. (2013). Dependency Analysis of System-of-Systems Operational and Development Networks. *Procedia Computer Science*,*16*, 265-274.

Haimes, Y. Y. (2012). Modeling Complex Systems of Systems with Phantom System Models. *Systems Engineering*, *15*(3), 333-346.

Han, S. Y., & DeLaurentis, D. (2013). Development Interdependency Modeling for System-of-Systems (SoS) using Bayesian Networks: SoS Management Strategy Planning. *Procedia Computer Science*, *16*, 698-707.

Hansen, P., Jaumard, B., & Savard, G. (1992). New Branch-and-bound Rules for Linear Bi-level Programming. *SIAM Journal on Scientific and Statistical Computing*, *13*(5), 1194-1217.

Hassan, R., & Crossley, W. (2007). Approach to Discrete Optimization Under Uncertainty: The Population-Based Sampling Genetic Algorithm. *AIAA Journal*,*45*, 2799-2809.

Hassan, R., De Weck, O., & Springmann, P. (2004, May). Architecting a Communication Satellite Product Line. In *22nd AIAA International Communications Satellite Systems Conference & Exhibit (ICSSC). Monterey, CA*.

He, Z., Yen, G. G., & Zhang, J. (2014). Fuzzy-Based Pareto Optimality for Many-Objective Evolutionary Algorithms. *Evolutionary Computation, IEEE Transactions on*, *18*(2), 269-285.

Henson, S. A., Henshaw, M. J. D., Barot, V., Siemieniuch, C. E., Sinclair, M. A., Jamshidi, M., ... & DeLaurentis, D. (2013, June). Towards a Systems of Systems Engineering EU Strategic Research Agenda. In *System of Systems Engineering (SoSE), 2013 8th International Conference on* (pp. 99-104). IEEE.

Herroelen, W., De Reyck, B., & Demeulemeester, E. (1998). Resource-constrained Project Scheduling: a Survey of Recent Developments. *Computers & Operations Research*, *25*(4), 279-302.

Hill Climbing. (n.d.). Retrieved October 24, 2013, from http://en.wikipedia.org/wiki/Hill_climbing

Holland, J. H. (1973). Genetic Algorithms and the Optimal Allocation of Trials. *SIAM Journal on Computing*, *2*(2), 88-105.

Hunt, B. R., Lipsman, R. L., Rosenberg, J. M., Coombes, K. R., Osborn, J. E., & Stuck, G. J. (2006). *A Guide to MATLAB: for Beginners and Experienced Users*. Cambridge University Press.

Hwang, C. L., Masud, A. S. M., Paidy, S. R., & Yoon, K. P. (1979). *Multiple objective Decision Making, Methods and Applications: a State-of-the-art Survey* (Vol. 164). Berlin: Springer.

INCOSE, (2011). *SYSTEMS ENGINEERING HANDBOOK v 3.2.2.* San Diego: INCOSE.

Jamshidi, M. (2008). System of Systems Engineering - New Challenges for the 21st century. *Aerospace and Electronic Systems Magazine, IEEE*, *23*(5), 4-19.

Jackson, S., & Ferris, T. L. (2013). Resilience Principles for Engineered Systems. *Systems Engineering*, *16*(2), 152-164.

Jia, L., Wang, Y., & Fan, L. (2011, December). Uniform Design Based Hybrid Genetic Algorithm for Multi-objective Bi-level Convex Programming. In *Computational Intelligence and Security (CIS), 2011 Seventh International Conference on* (pp. 159-163). IEEE.

Johnston, W., Mastran, K., Quijano, N., & Stevens, M. (2013). Unmanned Vehicle Search and Rescue Initiative. *SE4150 Design Project, Naval Postgraduate School*.

Joint Staff (2010). CJCSM 3500.04C, *UNIVERSAL JOINT TASK LIST (UJTL)*. Washing DC: Department of Defense.

Jonker, C. M., Robu, V., & Treur, J. (2007). An Agent Architecture for Multi-attribute Negotiation using Incomplete Preference Information. *Autonomous Agents and Multi-Agent Systems*, *15*(2), 221-252.

Karnik, N. N., Mendel, J. M., & Liang, Q. (1999). Type-2 Fuzzy Logic Systems. *Fuzzy Systems*, IEEE Transactions on, 7(6), 643-658.

Kilicay-Ergin, N.,(2014), Improving Collaboration in Search and Rescue System of Systems, *Procedia Computer Science, Volume 36*, Pages 13-20.

Kilicay-Ergin, N., Enke, D., & Dagli, C. (2012). Biased Trader Model and Analysis of Financial Market Dynamics. *International Journal of Knowledge-based and Intelligent Engineering Systems*, *16*(2), 99-116.

Kinnunen, M. J. (2006). *Complexity Measures for System Architecture Models* (Doctoral dissertation, Massachusetts Institute of Technology).

Kirkpatrick, S., Gelatt, C. D., & Vecchi, M. P. (1983). Optimization by Simulated Annealing. S*cience, 220*(4598), 671-680.

Koch, P. N., Evans, J. P., & Powell, D. (2002). Inter-digitation for Effective Design Space Exploration using iSIGHT. *Structural and Multidisciplinary Optimization*, *23*(2), 111-126.

Konur, D., & Dagli, C. (2014). Military System of Systems Architecting with Individual System Contracts*. Optimization Letters*, 1-19.

Konur, D., & Golias, M. M. (2013). Cost-stable Truck Scheduling at a Cross-dock Facility with Unknown Truck Arrivals: A Meta-heuristic Approach. *Transportation Research Part E: Logistics and Transportation Review*, *49*(1), 71-91.

Kraus, S. (1996). An Overview of Incentive Contracting. *Artificial Intelligence*, *83*(2), 297-346.

Kraus, S. (2001). Automated Negotiation and Decision Making in Multi-agent Environments. In *Multi-agent Systems and Applications* (pp. 150-172). Springer Berlin Heidelberg.

Krothapalli, N. K. C., & Deshmukh, A. V. (1999). Design of Negotiation Protocols for Multi-agent Manufacturing Systems. *International Journal of Production Research*, *37*(7), 1601-1624.

Lafleur, J. M. (2012). *A Markovian State-Space Framework for Integrating Flexibility into Space System Design*. Atlanta: Georgia Institute of Technology School of Aerospace Engineering Doctoral Thesis.

Lamar, B. W., & Bedford, M. A. (2009). Min-additive Utility Functions. *MITRE Corporation*.

Lane, J. A., & Bohn, T. (2013). Using SysML Modeling to Understand and Evolve Systems of Systems. *Systems Engineering*, *16*(1), 87-98.

Li, C., & Chiang, T. W. (2013). Complex Neurofuzzy ARIMA Forecasting — A New Approach Using Complex Fuzzy Sets. *Fuzzy Systems, IEEE Transactions on*, *21*(3), 567-584.

Li, M., Lin, D., & Wang, S. (2010). Solving a Type of Bi-objective Bi-level Programming Problem Using NSGA-II. *Computers & Mathematics with Applications*, *59*(2), 706-715.

Lopes, F., Wooldridge, M., & Novais, A. Q. (2008). Negotiation among Autonomous Computational Agents: Principles, Analysis and Challenges. *Artificial Intelligence Review*, *29*(1), 1-44.

Lu, J., Zhang, G., & Dillon, T. (2008). Fuzzy Multi-objective Bi-level Decision Making by an Approximation Kth-Best Approach. *Journal of Multiple-Valued Logic & Soft Computing*, *14*.

Luzeaux, D., System-of-Systems (and Large-Scale Complex Systems) Engineering, presentation at CSDM Conference, 2013.

Maier, M. W., & Rechtin, E. (2009). *The Art of Systems Architecting, 3rd ed.* Boca Raton: CRC Press.

Malan, R., & Bredemeyer, D. (2001). Architecture Resources. *Defining Non-Functional Requirements*. Retreived from http://www.bredemeyer.com/pdf_files/ArchitectureDecisions.pdf

Maskin, E. S. (1996). Theories of the Soft Budget-constraint. *Japan and the World Economy*, *8*(2), 125-133.

Mekdeci, B., Shah, N., Ross, A. M., Rhodes, D. H., & Hastings, D. (2014). *Revisiting the Question: Are Systems of Systems just (traditional) Systems or Are they a new class of Systems?* Cambridge, MA: Systems Engineering Advancement Research Initiative (SEAri).

Mendel, J. M. (2013). On KM algorithms for Solving Type-2 Fuzzy Set Problems. *Fuzzy Systems, IEEE Transactions on*, *21*(3), 426-446.

Mendel, J. M., & John, R. B. (2002). Type-2 Fuzzy Sets Made Simple. *Fuzzy Systems, IEEE Transactions on*, *10*(2), 117-127.

Metropolis, N., Rosenbluth, A. W., Rosenbluth, M. N., Teller, A. H., & Teller, E. (1953). Equation of State Calculations by Fast Computing Machines. *The Journal of Chemical Physics*, *21*(6), 1087-1092.

Mezura Montes, E., & Coello Coello, C. A. (2005). A Simple Multi-membered Evolution Strategy to Solve Constrained Optimization Problems. *Evolutionary Computation, IEEE Transactions on*, *9*(1), 1-17.

Michalewicz, Z., & Schoenauer, M. (1996). Evolutionary Algorithms for Constrained Parameter Optimization Problems. *Evolutionary Computation*, *4*(1), 1-32.

Miettinen, K. M., & Optimization, N. L. M. O. (1999). Kluwer Academic Publisher.

Migdalas, A., Pardalos, P. M., & Värbrand, P. (Eds.). (1998). *Multilevel Optimization: Algorithms and Applications* (Vol. 20). Springer.

Min, B. K., & Chang, S. H. (1991). System Complexity Measure in the Aspect of Operational Difficulty. *Nuclear Science, IEEE Transactions on*, *38*(5), 1035-1039.

Mordecai, Y., & Dori, D. (2013). I5: A Model-Based Framework for Architecting System-of-Systems Interoperability, Interconnectivity, Interfacing, Integration, and Interaction. In *International Symposium of the International Council on Systems Engineering (INCOSE)*.

Mostafavi, A., Abraham, D., Noureldin, S., Pankow, G., Novak, J., Walker, R., ... & George, B. (2012). Risk-Based Protocol for Inspection of Transportation Construction Projects Undertaken by State Departments of Transportation. *Journal of Construction Engineering and Management*, *139*(8), 977-986.

Osman, M. S., Abo-Sinna, M. A., Amer, A. H., & Emam, O. E. (2004). A Multi-level Non-linear Multi-objective Decision-making under Fuzziness. *Applied Mathematics and Computation*, *153*(1), 239-252.

Pape, L., Agarwal, S., Giammarco, K., & Dagli, C. (2014). Fuzzy Optimization of Acknowledged System of Systems Meta-architectures for Agent-based Modeling of Development. *Procedia Computer Science*, *28*, 404-411.

Pape, L., & Dagli, C. (2013). Assessing Robustness in Systems of Systems Meta-architectures. *Procedia Computer Science*, *20*, 262-269.

Pape, L., Giammarco, K., Colombi, J., Dagli, C., Kilicay-Ergin, N., & Rebovich, G. (2013). A Fuzzy Evaluation Method for System of Systems Meta-architectures. *Procedia Computer Science*, *16*, 245-254.

Pedrycz, W., Ekel, P., & Parreiras, R. (2011). *Fuzzy Multi-criteria Decision-making: Models, Methods and Applications*. John Wiley & Sons.

Pieume, C. O., Marcotte, P., Fotso, L. P., & Siarry, P. (2011). Solving Bi-level Linear Multi-objective Programming Problems. *American Journal of Operations Research*, *1*, 214.

Pitsko, R., & Verma, D. (2012). Principles for Architecting Adaptable Command and Control Systems. *Procedia Computer Science*, *8*, 135-140.

Ravindran, A., Reklaitis, G. V., & Ragsdell, K. M. (2006). *Engineering Optimization: Methods and Applications*. John Wiley & Sons.

Räihä, O. (2008). Applying Genetic Algorithms in Software Architecture Design.

Rela, L. (2004). Evolutionary Computing in Search-based Software Engineering.

Rios, L. M., & Sahinidis, N. V. (2013). Derivative-free Optimization: a Review of Algorithms and Comparison of Software Implementations. *Journal of Global Optimization*, *56*(3), 1247-1293.

Ricci, N., Ross, A. M., Rhodes, D. H., & Fitzgerald, M. E. (2013). Considering Alternative Strategies for Value Sustainment in Systems-of-Systems (Draft). *Systems Engineering Advancement Research Initiative, Cambridge MA*.

Rosenau, W. (1991). Coalition Scud Hunting in Iraq, 1991. *RAND Corporation*.

Ross, A. M., Rhodes, D. H., & Hastings, D. E. (2008). Defining Changeability: Reconciling Flexibility, Adaptability, Scalability, Modifiability, and Robustness for Maintaining System Lifecycle Value. *Systems Engineering*, 246 – 262.

Ross, S. M. (2014). *Introduction to Probability Models, 8th Edition*. Academic Press.

Rostker, B. (200, July 25). *Iraq's Scud Ballistic Missiles*. Retrieved Sep 12, 2013, from Iraq Watch: http://www.iraqwatch.org/government/US/Pentagon/dodscud.htm

Runarsson, T. P., & Yao, X. (2000). Stochastic Ranking for Constrained Evolutionary Optimization. *Evolutionary Computation, IEEE Transactions on*, *4*(3), 284-294.

Russell, S. (2009). Artificial Intelligence: A Modern Approach Author: Stuart Russell, Peter Norvig, Publisher: Prentice Hall Pa.

Sanz, J. A., Fernández, A., Bustince, H., & Herrera, F. (2013). IVTURS: A Linguistic Fuzzy Rule-Based Classification System Based On a New Interval-Valued Fuzzy Reasoning Method With Tuning and Rule Selection. *IEEE T. Fuzzy Systems*, *21*(3), 399-411.

Schäfer, R. (2001, October). Rules for using Multi-attribute Utility Theory for Estimating a User's Interests. In *Ninth Workshop Adaptivität und Benutzenmodellierung in Interaktiven Softwaresystemen* (pp. 8-10).

Schreiner, M. W., & Wirthlin, J. R. (2012). Challenges Using Modeling and Simulation in Architecture Development. *Procedia Computer Science*, *8*, 153-158.

Schwartz, M. (2010, April). Defense Acquisitions: How DoD Acquires Weapon Systems and Recent Efforts to Reform the Process. LIBRARY OF CONGRESS WASHINGTON DC CONGRESSIONAL RESEARCH SERVICE.

Selva, D., & Crawley, E. F. (2013). VASSAR: Value Assessment of System Architectures using Rules. *IEEE Aerospace Conference* (pp. 1-21). Big Sky MT: IEEE.

Shi, X., & Xia, H. S. (2001). Model and Interactive Algorithm of Bi-level Multi-objective Decision-making with Multiple Interconnected Decision Makers. *Journal of Multi-Criteria Decision Analysis*, *10*(1), 27-34.

Shtub, A., Bard, J. F., & Globerson, S. (1994). *Project Management: Engineering, Technology, and Implementation*. Prentice-Hall, Inc..

Singer, Y. (2006, November). Dynamic Measure of Network Robustness. In *Electrical and Electronics Engineers in Israel, 2006 IEEE 24th convention of* (pp. 366-370). IEEE.

Siemieniuch, C., Sinclair, M., Lim, S. L., Henson, M. S., Jamshidi, M., & DeLaurentis, D. (2013). Project Title Trans-Atlantic Research and Education Agenda in Systems of Systems (T-AREA-SoS).

Simpson, T. W., & D'souza, B. S. (2002). Assessing Variable Levels of Platform Commonality within a Product Family using a Multi-objective Genetic Algorithm. In *Proceeding of the 9th AIAA/ISSMO Symposium on Multidisciplinary Analysis and Optimization, AIAA*.

Singh, A., & Dagli, C. H. (2009, March). Multi-objective Stochastic Heuristic Methodology for Tradespace Exploration of a Network Centric System of Systems. In *Systems Conference, 2009 3rd Annual IEEE* (pp. 218-223). IEEE.

Smartt, C., & Ferreira, S. (2012). Constructing a General Framework for Systems Engineering Strategy. *Systems Engineering*, *15*(2), 140-152.

SPEC Innovations. (2014). *Model-Based Systems Engineering Tools*. Retrieved August 20,2014, from https://www.innoslate.com/systems-engineering.

Suarez, R. (2004, Dec 9). *Troops Question Secretary of Defense Donald Rumsfeld about Armor*. Retrieved Apr 14, 2014, from PBS NewsHour: http://www.pbs.org/newshour/bb/military-july-dec04-armor_12-9/

Sumathi, S., & Paneerselvam, S. (2010). *Computational Intelligence Paradigms: Theory & Applications using MATLAB*. CRC Press.

Trans-Atlantic Research and Education Agenda in Systems of Systems (T-AREA-SOS) Project, "The Systems of Systems Engineering Strategic Research Agenda," Loughborough University, Loughborough, 2013.

Talbot, F. B. (1982). Resource-constrained Project Scheduling with Time-resource Tradeoffs: The Non-preemptive Case. *Management Science*, *28*(10), 1197-1210.

Taleb, N. N. (2004). *Fooled by Randomness*. New York: Random House Trace Paperbacks.

Thompson, M. (2002). Iraq: The Great Scud Hunt. *Time Magazine*, *23*.

Wall, M. B. (1996). *A Genetic Algorithm for Resource-constrained Scheduling* (Doctoral dissertation, Massachusetts Institute of Technology).

Wang, J. Q., & Zhang, H. Y. (2013). Multi-criteria Decision-making Approach based on Atanassov's Intuitionistic Fuzzy Sets with Incomplete Certain Information on Weights. Fuzzy Systems, IEEE Transactions on, 21(3), 510-515.

Wang, R., & Dagli, C. H. (2011). Executable System Architecting using Systems Modeling Language in Conjunction with Colored Petri Nets in a Model-driven Systems Development Process. *Systems Engineering, 14(4),* 383-409.

Wang, R., & Dagli, C. H. (2013). Developing a Holistic Modeling Approach for Search-based System Architecting. *Procedia Computer Science*, *16*, 206-215.

Wang, R., Agarwal,S., & Dagli, C. (2014). Executable System of Systems Architecture Using OPM in Conjunction with Colored Petri Net: A Module for Flexible Intelligent & Learning Architectures for System of Systems, In *Europe Middle East & Africa Systems Engineering Conference (EMEASEC)*.

Wang, Y., & Cai, Z. (2012). Combining Multi-objective Optimization with Differential Evolution to Solve Constrained Optimization Problems. *Evolutionary Computation, IEEE Transactions on*, *16*(1), 117-134.

Wang, Y., Cai, Z., Guo, G., & Zhou, Y. (2007). Multi-objective Optimization and Hybrid Evolutionary Algorithm to Solve Constrained Optimization Problems. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, *37*(3), 560-575.

Wanyama, T., & Homayoun Far, B. (2007). A Protocol for Multi-agent Negotiation in a Group-choice Decision Making Process. *Journal of Network and Computer Applications*, *30*(3), 1173-1195.

Wappler, S. (2007). *Automatic Generation of Object-oriented Unit Tests using Genetic Programming* (Doctoral dissertation, Universitätsbibliothek).

Wappler, S., & Wegener, J. (2006, July). Evolutionary Unit Testing of Object-oriented Software using Strongly-typed Genetic Programming. In *Proceedings of the 8th Annual Conference on Genetic and Evolutionary Computation* (pp. 1925-1932). ACM.

Warfield, J. N. (1973). Binary Matrices in Systems Modeling. *IEEE Transactions on Systems, Man, and Cybernetics, SMC-3* (No. 5, September), 441-449.

Weiss, W. E. (2008, December). Dynamic Security: An Agent-based Model for Airport Defense. In *Simulation Conference, 2008. WSC 2008. Winter* (pp. 1320-1325). IEEE.

Welby, S. P. Systems Engineering FY 2012 Annual Report.

Woldesenbet, Y. G., Yen, G. G., & Tessema, B. G. (2009). Constraint Handling in Multi-objective Evolutionary Optimization. *Evolutionary Computation, IEEE Transactions on*, *13*(3), 514-525.

Wooldridge, M., & Parsons, S. (2000, August). Languages for Negotiation. In *ECAI* (pp. 393-400).

Wu, D., & Mendel, J. M. (2007). Uncertainty Measures for Interval Type-2 Fuzzy Sets. *Information Sciences*, *177*(23), 5378-5393.

Yi, J. S., Kang, Y., Stasko, J. T., & Jacko, J. A. (2007), Nov/Dec). Toward a Deeper Understanding of the Role of Interaction in Information Visualization. *IEEE Transactions on Visualization and Computer Graphics, 13(6),* 1224-1231.

Yu, O. Y., Guikema, S. D., Briaud, J. L., & Burnett, D. (2012). Sensitivity Analysis for Multi-attribute System Selection Problems in Onshore Environmentally Friendly Drilling (EFD). *Systems Engineering*, *15*(2), 153-171.

Yu, T. L., Yassine, A. A., & Goldberg, D. E. (2007). An Information Theoretic Method for Developing Modular Architectures using Genetic Algorithms. *Research in Engineering Design*, *18*(2), 91-109.

Zadeh, L. A. (1975). Fuzzy Logic and Approximate Reasoning. *Synthese*, *30*(3-4), 407-428.

Zhang, T., Hu, T., Zheng, Y., & Guo, X. (2012). An Improved Particle Swarm Optimization for Solving Bi-level Multi-objective Programming Problem. *Journal of Applied Mathematics*, *2012*.

Zhang, T., Hu, T., Chen, J. W., Wan, Z., & Guo, X. (2012, November). Solving Bi-level Multi-objective Programming Problem by Elite Quantum Behaved Particle Swarm Optimization. In *Abstract and Applied Analysis* (Vol. 2012). Hindawi Publishing Corporation.

Zhang, G., Lu, J., & Gao, Y. (2008). An Algorithm for Fuzzy Multi-objective Multi-follower Partial Cooperative Bi-level Programming. *Journal of Intelligent and Fuzzy Systems*, *19*(4), 303-319.

Zhang, G., Lu, J., & Gao, Y. (2008). Fuzzy Bi-level Programming: Multi-objective and Multi-follower with Shared Variables. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, *16*(supp02), 105-133.

Zhang, G., & Lu, J. (2010). Fuzzy Bi-level Programming with Multiple Objectives and Cooperative Multiple Followers. *Journal of Global Optimization*, *47*(3), 403-419.

Zheng, Y., Wan, Z., & Wang, G. (2011). A Fuzzy Interactive Method for a Class of Bi-level Multi-objective Programming Problem. *Expert Systems with Applications*, *38*(8), 10384-10388.

Zhou, A., Qu, B. Y., Li, H., Zhao, S. Z., Suganthan, P. N., & Zhang, Q. (2011). Multi-objective Evolutionary Algorithms: A Survey of the State of the Art. *Swarm and Evolutionary Computation*, *1*(1), 32-49.