System of Systems Architecture Feasibility Analysis to Support Tradespace Exploration

Stephen E. Gillespie
Department of Systems Engineering
U.S. Military Academy
West Point, NY, 10996
stephen.gillespie@usma.edu

Ronald E. Giachetti, Alejandro Hernandez, Paul T. Beery, and Eugene P. Paulo
Department of Systems Engineering
Naval Postgraduate School
Monterey, CA, 93940
regiache@nps.edu, ahernand@nps.edu, ptbeery@nps.edu, eppaulo@nps.edu

***Abstract*—The exploration of a system of systems (SoS) tradespace is made much more efficient and effective with a method to first automatically screen a large number of SoS designs for feasibility. This is because not every combination of constituent systems is capable of forming a viable SoS, much less form a SoS that exhibits the desired emergent behavior(s). The SoS Architecture Feasibility Assessment Model (SoS-AFAM) assesses the feasibility of the physical communications, process, and organizational architectures of a SoS. The model applies algorithms based on the minimum requirements for viability relevant to all SoS such as connectivity and completeness. We present a case study to demonstrate how the algorithm can greatly prune the SoS tradespace of infeasible SoS design points, which can increase the efficiency of design exploration.**

Keywords—systems-of-systems, tradespace exploration, feasibility analysis, model-based systems engineering

I. Introduction

Systems of systems engineering (SoSE) is a growing field that reflects the growing societal desire to meets its needs through the use of systems of systems (SoS) [8] [13] [14] [20]. This is due both to the increasingly networked nature of modern society and the growing recognition that, in many cases, monolithic systems cannot efficiently meet these societal needs. Coincident with the growth of SoSE has been the trend in engineering on defining the systems design space very broadly using a variety of statistical and high-power computing techniques [12] [19]; these may be broadly called methods of Tradespace Exploration (TSE). A tradespace is the set of all design points and, traditionally, the systems engineer searches the tradespace for the best design.

In SoSE, the tradespace can be very large, entailing tens of thousands of potential designs. Yet, many of these potential design points are infeasible due to physical, operational, or organizational constraints. The removal of infeasible designs from the tradespace can greatly increase the efficiency of subsequent engineering tasks in exploring the tradespace. In fact, this is the basis of the set-based design paradigm [10] [26].

This paper contributes a method to quickly assess the feasibility of SoS design points so as to greatly prune the tradespace. The paper first describes the tradespace exploration paradigm and feasibility. The paper then describes our SoS architecture feasibility assessment method. The paper briefly discusses a case study demonstrating the feasibility assessment method on a SoS architecture.

II. Tradespace Exploration for Systems of Systems

To define a system's tradespace, one must define at least four things. The first is a set of design parameters that define the architecture for the system (or SoS) in question. The second is the set of operational measures for which stakeholders are concerned (i.e., the value measures that inform how well the system meets the organization's requirements). Third, one must define a function, or set of functions, that define the relationship between the design parameters and the operational measures. Fourth, one must define a function that assesses the feasibility of realizing a system with a given set of design parameters. In doing this, one may develop a dynamic dashboard that may be explored that considers the relationship between the operational requirements and the system design—"illuminating the tradespace" [19].

Defining the set of design points and desired operational measures, while not insignificant, is highly manageable and falls largely in the domain of problem definition. The challenge of defining the tradespace primarily falls on the latter two requirements—of first defining how the values of design parameters change operational measures, and second, determining whether a combination of design parameters is feasible.

In many cases a simulation model shows how design parameters affect operational measures. SoS are typically assessed using agent based models [1] [23], though other simulations such as Petri Nets [24] or Markov models [9] have been used. The inherent challenge, however, is that for a large design space, the number of requisite experiments quickly becomes untenable, even for the fastest computers.

To address the aforementioned problem, significant work has been done using experimental design to statistically

approximate the relationship between design parameters and operational measures [2] [19]. This works well when the design variables are well ordered and one may reasonably assume that the interactions among the design variables are limited to second order [18]. This is not generally true in the case of a SoS. One must assume that there are significant, complex interactions among the design variables [13] [20] and, often, one has a significant number of non-ordered, categorical design variables (e.g., one such categorical design variable may be the set of constituent systems included in the SoS), which challenges the limitations of experimental design [25] [27]. Kernstine [16] outlines this challenge for SoS well and addressed it through the use of adaptive sequential experiments [17]. The other alternative is to develop very low-fidelity models that define a SoS's operational measures through analytic models that take a "best-in-class" or weighted average approach to define a set of systems' operational performance parameters through the performance of the constituent systems as done by [6].

Regardless of the approach used to define the relationship between design parameters and operational measures, to truly define a tradespace, one must assess whether or not a distinct set of design parameters has the potential to form a feasible system in the first place. This question has largely not been addressed, and, in the case of SoS, remains an open question.

III. Feasibility - General

The concept of feasibility is well understood in systems engineering [3] [5], though rarely explicitly defined [5]. One challenge of explicitly defining feasibility is that it is highly dependent upon the needs of the stakeholders and availability of resources. Regardless, there are factors that, in general, make a design feasible or infeasible. Oftentimes, these factors may be physical or integration considerations. A physical consideration assesses compliance with basic physical laws, e.g. a ship must float and be stable, so there is a relationship between its length, width, and mass that determines feasibility [19]. Integration considerations account for the needs of the sub-systems to work together, e.g., a helicopter may only be incorporated onto a ship if there is sufficient room for it to land [4] [19]. Additionally, there are factors that may contribute to the feasibility of a project such as the technological availability of required components or social and political considerations. Cost does impact feasibility, though it is more generally considered in a separate manner given its importance (i.e., cost as an independent variable).

When designing candidate solutions for a system design problem, an engineer typically conducts a manual feasibility analysis. In the case of developing a tradespace that examines thousands, tens of thousands, or more design points, however, manual feasibility analysis is clearly untenable. In this case, one must develop a function that takes design parameters as inputs and outputs whether or not such a design is feasible. The question then, is how does one consider the feasibility of a SoS that is composed of operationally and managerially independent systems, evolves over time, presents emergent behavior, and is geographically distributed.

IV. SoS Feasibility

There is relatively limited research on exploring the tradespace of SoS, primarily [6]. A challenge of this research is that it makes a tacit assumption that, given any set of potential constituent systems, one may choose any subset of systems and form a feasible SoS. However, this is not generally true because the integration of constituent systems into a SoS may not be feasible.

We posit that any SoS must be minimally feasible from at least three perspectives: the physical communications architecture, the process architecture, and the organization architecture. We generally call these, respectively, the physical, process, and organizational views. A SoS must be feasible from each of these views individually and collectively.

Physical View: The communications architecture for a SoS must form a connected network for the SoS to be feasible. If the architecture of the SoS is such that at least two subsets of the constituent system are not connected, then those two subsets cannot communicate, and thus, are not working together to produce an emergent behavior as a SoS. Figure 1 illustrates the need for a SoS to form a connected network in order to be feasible.

Process View: SoS exhibit emergent behavior [20]. If one is engineering an acknowledged or directed SoS, then there is one or more desired emergent behaviors for which one is designing. At present we can only truly engineer for "simple" or "weak" emergent behaviors per Maier's definition of emergence [21]. As such, these are behaviors that are sufficiently well understood that we can define the necessary processes—the functions and rules by which the processes are executed—to bring about the desired emergent behavior. Accordingly, we may assess whether or not a potential systems has sufficient functionality and / or will abide by the chosen rules to bring about the desired emergent behavior. If such a SoS does not have these, it is, therefore, infeasible.

Organizational View: A SoS is both a technological system and an organization. It is an organization as its constituent systems are decision-making entities with various levels of autonomy. Accordingly, these constituent systems have relationships defined in the design of the SoS. This

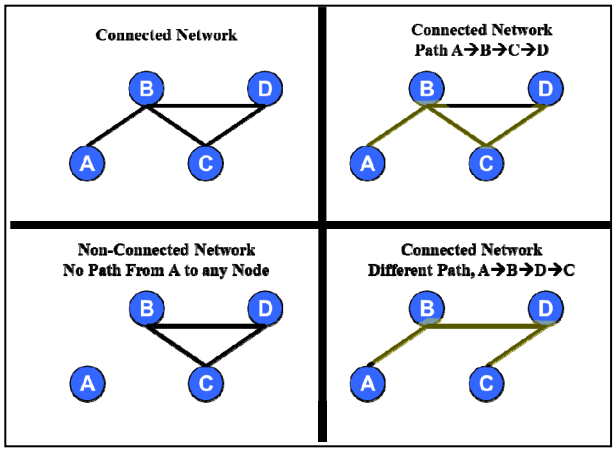

Fig. 1. The distinction between a connected and non-connected network.

organizational design is feasible if the constituent systems agree to the definition of the relationships and if the organizational design forms a connected network that is supported by the physical architecture. The former question is simple, if an organizational design dictates that two systems have a relationship that one or both of the systems disagree with, the disagreeing system will not participate in the SoS and it is not feasible. The latter question is two parts. First, the network formed by the constituent systems and relationships must be connected. If it is not, the infeasibility results for the same reason as in the physical view. Second, two systems must be able to communicate if they are to have a relationship. If they cannot, then that organizational design is infeasible.

Together these three claims compose a set of necessary measures by which to assess SoS feasibility for any SoS. As such, we may develop generic algorithms to test these questions quickly for a large number of possible SoS designs. This facilitates subsequent SoS TSE.

V. SoS Architecture Feasibility Assessment Model (SoS-AFAM)

We developed the SoS Architecture Feasibility Assessment Model (SoS-AFAM) which consists of a set of algorithms to quickly assess a large number of SoS designs for feasibility. It assesses the set of possible designs first for physical feasibility, then process and organizational feasibility, and finally the integration of those three views. If a SoS design is feasible across all three perspectives, it is judged feasible. Due to space constraints, we do not present the full algorithms here; for a more complete treatment see [11].

The scope of the SoS-AFAM is limited to considering the design of a SoS at a single point in its evolution for a given operating environment. It is further limited to considering only the development of acknowledged or directed SoS that have an agreed upon central purpose.

A. SoS-AFAM – Input

The input to the SoS-AFAM is a set of information across the three views: physical, process, and organization. This includes the set of possible constituent systems, any changes (re-factorizations) that may be made to those systems, the characteristics of those systems, the possible processes—both functions and rules, and the possible organizations, defined as the set of relationships between any two systems. Importantly, for each possible constituent system, one must also identify the characteristics of those systems—what functions they are capable of, what communications sub-systems they have, what process rules and relationships they are willing to accept.

For example, one could represent the set of all possible functions as a table that cross-references systems versus functions. In the case of Table I, one sees a table that states that System 1 can perform Functions 1 and 2.

We represent the specific SoS design as a vector whose entries correspond to the systems, processes, and organizations used for that design point. For example, if there are five possible constituent systems, ten possible ways to organize the SoS, and two possible processes defining how they interact in the SoS, then the vector $<S_1, S_2, S_3, S_4, S_5, O, P>$ represents the tradespace of the SoS S_n takes the value of 1, if the n^{th} system is included and 0 if it is not. The variable O takes a value between 1 and 10 corresponding to the ten organization types available. The variable P takes the value of 1 or 2 corresponding to the specific process used by the SoS.

TABLE I. Example System vs. Function Cross-reference

	Function 1	Function 2	Function 3
System 1	X	X	
System 2		X	X
System 3			X

B. SoS-AFAM – Physical

The first feasibility question is, can one form a connected network with the included constituent systems? To do this, an adjacency matrix represents the communications architecture of the included constituent systems. The matrix is an $n \times n$ matrix where n is the number of included systems. Each i-j entry represents whether or not the i^{th} and j^{th} system share a common communications subsystem; a 1 indicates a common communications means and a 0 otherwise. With this adjacency matrix, one may use a network science package (e.g., the IsConnected function in MATLAB) to assess for connectivity. A connected adjacency matrix establishes a minimal constraint on feasibility.

As described above, the threshold considered for connectivity between two systems is relatively low—they need only share a common communications sub-system. In reality, this may not be sufficient for two systems to actually communicate. If one desires a higher degree of fidelity, one may develop a function that reflects the relevant requirement. For example, if two systems share a common FM-radio communications sub-system, their ability to communicate is not only a function of their shared platform, but also of their geographic location, as FM-radios have limited range. In this case, one would require that the two systems must be within

range of each other to score a 1 in the relevant *i-j* cell of the adjacency matrix. Similar considerations may be given to bandwidth, latency, error-rate, or system availability.

C. SoS-AFAM – Process

The second question is whether or not a SoS design has the potential to bring about the desired emergent property(ies). This is done in three ways: 1) sufficient functionality, 2) acceptance of operating rules, and 3) lack of system interference.

Sufficient Functionality: A SoS design point has both a set of constituent systems, each with the functions it can perform, and a process with a number of requisite functions to be performed. At a minimum, for a SoS to be feasible, the set of constituent systems must be able to complete all of the functions required by the process. This is easily checked by summing the total number of functions, by type, available from the included constituent systems, and comparing these numbers to a similar sum for a given process. For example, if a military indirect fire system has the process "observe" then "shoot," and, if the constituent systems can only "shoot," that SoS would be infeasible as those systems cannot "observe."

Rule Acceptance: A SoS process may have a number of rules governing the activities of the constituent systems and the SoS. For a SoS design point to be feasible, all included constituent systems must agree to abide by the chosen process rules. The algorithm to check this is to simply compare the rules for a relevant process against a table that enumerates what each constituent system considers acceptable. If all constituent systems in a design point agree to all process rules in the design point, that point is considered feasible.

System Interference: The "sufficient functionality" analysis considered the raw numbers and types of functions required for a process versus the amount provided by the constituent systems. This, however, may not be enough. In some cases, functions must be performed simultaneously. Due to the nature by which constituent systems provide functionality, it may not be possible for them to provide two functions simultaneously. This may be checked by identifying all required simultaneous operations for a given SoS design point's process(es) and comparing this with a table that identifies interference between each constituent system and the functions it performs against every other system–function combination. For example, Table II identifies an interference between a UAV observing and an artillery system shooting simultaneously. If a process required simultaneous observation and shooting, this UAV-Artillery SoS would not be feasible.

TABLE II. SYSTEM-FUNCTIONALITY INTERFERENCE EXAMPLE

	Artillery – Shoot	**UAV – Observe**
Artillery – Shoot	-	Interference
UAV – Observe	Interference	-

D. SoS-AFAM – Organizational

The question of a SoS design point's organizational feasibility is examined by two tests: whether or not the systems agree to the organizational relationships and whether or not the organizational architecture forms a connected set. We define an organization as the set of all relationships defined between any two constituent systems in the SoS design point. We represent it as a matrix in which each row and column corresponds to a constituent system and each *i-j* entry is enumerated with the defined relationship. Each relationship is clearly defined; for example, in the U.S. Army the relationships "tactical control" or "direct support" have very distinct meanings.

Organization Acceptance: This rule mandates that every constituent system in a SoS design point agrees to the defined relationships for that SoS design point's organization matrix. This may be checked in a manner similar to "rule acceptance" for the process view.

Organizational Connectivity: At a minimum, every organization has two types of relationships, either two systems have a relationship or they do not. With this information, one may modify a design point's organizational matrix by deleting the rows and columns of the non-included constituent systems and then assigning a 1 to each remaining *i-j* entry if there is a relationships and a 0 if there is not. One may then assess this for connectivity in the same manner as physical connectivity is described above. Note that a SoS design may be physically connected, but not organizationally connected or vice versa.

E. SoS-AFAM – Integration

The SoS-AFAM considers the set of all possible SoS designs across three perspectives: physical, process, and organization. If a design is feasible across all three perspectives, it may be judged preliminarily feasible. This alone is not sufficient. The SoS architecture must be mutually supporting across the different views. We consider each of the three pairs of views.

Physical Support of Organization: If two constituent systems have an organizational relationship, they must have a means to communicate. This is easily checked by comparing the modified organizational connectivity matrix described in section V.D. against the physical connectivity matrix described in section V.B. If there is a pair of included constituent systems in a SoS design point that do not have a shared communications sub-system, then the design is infeasible.

Physical and Organizational Support of Process: As seen above, for a SoS design to be feasible, the physical architecture must support the organizational architecture. Therefore, we may consider the physical and organizational support of the process architecture simultaneously. In a process, between the sequential execution of any two functions by different constituent systems, information must transfer from the system that conducts the first function to the system that conducts the second function. This information follows a path along the organizational network (supported by the physical network). This transfer of information must be timely (i.e., the second system must receive the information in sufficient time to execute its function) and in the appropriate form (i.e., the information received by the second system must be usable by that system) for the SoS to be feasible. This may be checked by considering each pair of transitions between functions in the process architecture against how long it takes for the organization to send that information providing constituent system to the receiving constituent system and whether or not

it can convert the information to the required form. If the SoS is not timely or cannot provide necessary information in an appropriate manner, it is not a feasible SoS.

F. SoS-AFAM – Conclusion

The SoS-AFAM assesses the feasibility of a large number of SoS designs across measures that are common to all SoS– the physical, process, and organizational architectures. One may iterate the SoS-AFAM at more detailed levels (e.g., consider not just shared communications sub-systems for physical connectivity, but whether or not the SoS network has sufficient bandwidth as discussed in section V.B.) if it is necessary to further refine the set of feasible SoS. In doing this, one may define the tradespace for a SoS design problem.

VI. Analysis and Utility

The SoS-AFAM assesses the feasibility of a SoS across three perspectives. The most significant challenge in doing this is defining the design variables and their relevant characteristics (e.g., agreement to work as a part of a SoS with a given policy); however, this is a matter of background research, surveys, and making reasonable assumptions. The computational complexity of the SoS-AFAM is relatively low. The algorithms, while not discussed in detail in this paper, are relatively simple. Moreover, as this model is a "one-strike and you're out" type model, one need not test every single design point for feasibility across all perspectives; rather, a SoS design that has been deemed infeasible in one perspective may be excluded from further analysis in other perspectives. This reduces the number of design points one must check. For a large set of potential SoS designs, this is a useful feature.

The utility of the SoS-AFAM is that it may be used to define the tradespace of a SoS. The SoS-AFAM can quickly screen a large set of SoS designs, and then, with a sufficiently small sample, directly assess each design for its operational properties using an appropriate method (e.g., an ABM or Petri net). All of this information can then be used to develop a dynamic dashboard to facilitate decision makers' understanding of the SoS design problem and "illuminate the tradespace."

As an example, the authors developed a hypothetical military situation for developing an indirect fire SoS [11]. The initial design considered nine distinct constituent systems, one re-factorization, eight processes, and eleven organizations, resulting in 90,112 possible SoS design points. With less than 10 minutes of analysis, the SoS-AFAM reduced tradespace to 7,980 feasible design points. We next assessed these design points for their operational measures (cost, percent enemy killed, and percent collateral damage) using an analytic model for cost and an ABM for the mission effectiveness measures. We were able to examine all design points in less than a week on a personal computer. All of the data was then used to build a tradespace dashboard.

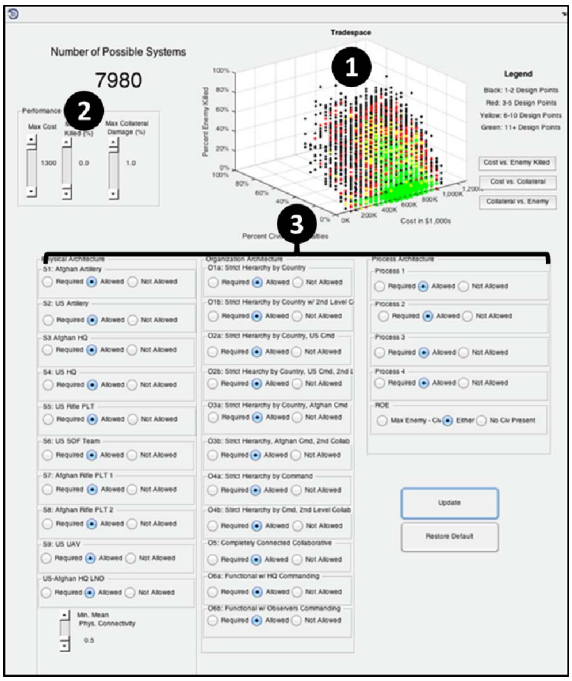

Fig. 2. Example SoS Tradespace Developed Using the SoS-AFAM

Figure 2 depicts the example tradespace with three significant points. Point 1 of the figure is a three dimensional plot with axes depicting the cost and effectiveness measures. A SoS design point is plotted if it achieves the corresponding measures on the graph. Point 2 of the figure is the user input to vary the thresholds for cost and effectiveness. As the user varies these, SoS designs that fail to achieve the thresholds are removed from the plot in Point 1. Finally, Point 3 of the figure is user input to vary the architectural requirements. For example, a user may require a certain constituent system to be included in the SoS design. This interactive visualization allows decision makers to vary cost and effectiveness thresholds and architectural requirements and thus gain an appreciation for what is possible, understand the nature of the relationship between design parameters and operational performance, and thus, make key requirements and design decisions based on this exploration and enhanced understanding.

VII. Conclusion

The SoS-AFAM contributes to the literature on SoSE by quickly identifying infeasible designs and removing them from the design space. The algorithms assess physical, process, and organizational constraints. This methodology may be applied to any SoS design problem as, while for any particular problem there are domain specific challenges, the nature of SoSE is such that the requirements for connectivity, functionality, and organizational acceptance are universal. In an example problem we show how the method can reduce the tradespace by over 90%. We believe that by pruning the tradespace, engineers can concentrate their efforts on those designs more likely to be viable with the result of thee engineering process being much more efficient and effective.

Further research can extend the SoS-AFAM to greater levels of detailed architecting and analysis. Additionally, the SoS-AFAM, as constructed, only considers the architecture of a SoS at one point in its evolutionary life-cycle for a given operational environment. Extending this concept to the development of a SoS across multiple evolutions of its life-cycle and/or to multiple environments is another area for future research. Finally, research should investigate whether the designs removed due to infeasibility are truly infeasible, or only identified as such due to the low fidelity of the models.

REFERENCES

[1] Baldwin, W. C., Sauser, B., & Cloutier, R. (2015). Simulation Approaches for System of Systems: Events-Based Versus Agent Based Modeling. *Procedia Computer Science, 44*, 363-372.

[2] Beery, P. T. (2016). A Model-Based Systems Engineering Methodology for Employing Architecture in System Analysis. Monterey, CA: Ph.D. Dissertation, Naval Postgraduate School.

[3] Blanchard, B. S., & Fabrycky, W. J. (2011). Systems Engineering and Analysis. New York, NY: Prentice Hall.

[4] Browne, D. C., Ender, T. R., Yates, W., & O'Neal, M. (2012). FACT: Enabling Systems Engineering as a Web Service. Retrieved January 31, 2017, from http://www.dtic.mil/ndia/2012system/ttrack514877.pdf

[5] Buede, D. M. (2000). The Engineering Design of Systems: Models and Methods. New York, NY: John Wiley & Sons.

[6] Chattopadhyay, D., Ross, A. M., & Rhodes, D. H. (2008). A Framework for Tradespace Exploration of Systems of Systems. 6th Conference on Systems Engineering Research. Los Angeles, CA.

[7] Davendralingam, N., & DeLaurentis, D. (2015). A Robust Portfolio Optimization Approach to System of System Architectures. Systems Engineering, 18(3), 269-283.

[8] Department of Defense. (2008). *Systems Engineering Guide for Systems of Systems.* Washington, DC: Department of Defense.

[9] Giachetti, R. E. (2015). System of Systems Capability Needs Analysis via a Stochastic Network Model. *Naval Engineers Journal, 127*(4), 67-79.

[10] Giachetti, R.E. and Whitcomb, C.A., Rethinking the systems engineering process in light of design thinking (2016). *Proceedings of the 2016 Acquisition Research Conference*, Monterey, CA.

[11] Gillespie, S. (2016). *The System of Systems Architecture Feasibility Assessment Model.* Monterey, CA: PhD Dissertation, Naval Postgraduate School.

[12] Holland, J. P. (2013, December 2). *Engineered Resilient Systems (ERS) Overview*. Retrieved June 1, 2015, from http://www.defenseinnovationmarketplace.mil/resources/ERS_Overview_2DEC2

[13] Jamshidi, M. (2008). *Systems of Systems Engineering: Principles and Applications*. Boca Raton, FL: CRC Press.

[14] Jamshidi, M. (2009). *Systems of Systems Engineering: Innovations for the Twenty-First Century*. Hoboken, NJ: Wiley.

[15] Kenley, C. R., Dannenhoffer, T. M., Wood, P. C., & DeLaurentis, D. A. (2014). Synthesizing and Specifying Architectures for System of Systems. *24th Annual INCOSE International Symposium.* Las Vegas, NV.

[16] Kernstine, K. H. (2013). Inadecquacies of Traditional Exploration Methods in Systems-of-Systems Simulations. *IEEE Systems Journal, 7*(4), 528-536.

[17] Kertstine, K. H. (2012). *Design Space Exploration of Stochastic System-of-Systems Simulations Using Adaptive Sequential Experiments*. Atlanta, GA: Ph.D. Dissertation, Georgia Institute of Technology.

[18] MacCalman, A. D. (2013). *Flexible Space-Filling Designs for Complex System Simulations.* Monterey, CA: Ph.D. Dissertation, Naval Postgraduate School.

[19] MacCalman, A. D., Beery, P. T., & Paulo, E. P. (2016). A Systems Design Exploration Approach that Illuminates Tradespaces Using Statistical Experimental Designs. *Systems Engineering, 19*(5), 409-421.

[20] Maier, M. W. (1998). Architecting Principles for Systems-of-Systems. *Systems Engineering, 1*(4), 267-284.

[21] Maier, M. W. (2015). The Role of Modeling and Simulation in System of Systems Development. In *Modeling and Simulation Support for System of Systems Engineering Applications* (pp. 11-41). Hoboken, NJ: Wiley.

[22] Mokhtarpour, B., & Stracener, J. (2014). A Conceptual Methodology for Selecting the Preferred System of Systems. *IEEE Systems Journal, 99*, 1-7.

[23] Mour, A., Kenley, C. R., Davendralingam, N., & DeLaurentis, D. (2013). Agent-Based Modeling for Systems of Systems. *23rd Annual International Symposium of the International Council of Systems Engineering.* Philadelphia, PA.

[24] Rao, M., Ramakrishnan, S., & Dagli, C. (2008). Modeling and Simulation of Net Centric Systems of Systems Using Systems Modeling Language and Colored Petri-Nets: A Demonstration Using the Global Earth Observation System of Systems. *Systems Engineering, 11*(3), 203-220.

[25] Sanchez, S. M., & Wan, H. (2012). Work Smarter, Not Harder: A Tutorial on Designing and Conducting Simulation Experiments. *Proceedings of the 2012 Winter Simulation Conference (2012)*. Berlin, Germany.

[26] Sobek II, D. K., Ward, A. C., & Liker, J. K. (1999). Toyota's principles of set-based concurrent engineering. *MIT Sloan Management Review*, 40(2), 67.

[27] Vieira, H., Sanchez, S., Kienitz, K. H., & Belderrain, M. C. (2011). Improved Efficient, Nearly Orthogonal, Nearly Balanced, Mixed Designs. *Proceedings of the 2011 Winter Simulation Conference (WSC)*. Phoenix, AZ.