System-of-Systems Architecture Selection: A Survey of Issues, Methods, and Opportunities

Zhemei Fang

***Abstract*—System-of-systems (SoSs) architecture concerns the structure of components, their relationships, and their principles and guidelines governing their design and evolution over time. Well-designed architecture is key to manage the SoS complexity, and thus plays a considerable role in the SoS development and evolution. Architecture selection that aims to identify the desired architectures from a large alternative design space is an initial but critical step toward successful architecture design. Decision-makers need scientific methods to identify those better, more informed but perhaps unexpected architecture solutions. However, the unique SoS characteristics such as interdependency and autonomy bring significant challenges to the architecture selection studies. This article identifies seven critical perspectives in SoS architecture selection, including the conceptual framework, evaluation criteria, interdependency, uncertainty, autonomy, dynamic evolution, and computational methods. The available methods, tools, and processes in the SoS and associated domains are reviewed thoroughly regarding each perspective. This review provides a comprehensive understanding of the current research status of SoS architecture selection. It builds a holistic picture that brings the fragmented pieces of the puzzle together. Towards the end, this article presents a series of research directions for future SoS architecture selection studies.**

***Index Terms*—Architecture selection, characteristics, critical issues, survey, system-of-systems (SoSs).**

I. Introduction

SYSTEM-of-systems (SoSs) has gradually drawn attention in the domains of military, aerospace, smart grid, transportation, supply chain, health care, among others in the recent two decades. SoS often refers to a collection of heterogeneous, distributed and independently managed systems that can achieve an overarching capability via interaction and interoperability [1]. Rather than a universally-accepted, rigorous SoS definition, the SoS community mostly agrees on several distinguishing characteristics, including managerial independence, operational independence, geographic distribution, evolutionary development, and emergent behavior [2], [3]. Boardman and Sauser [4] proposed a slightly different set of characteristics including autonomy, belonging, connectivity, diversity, and emergence. DeLaurentis [5] identified a few more traits including networks, heterogeneity, and trans-domain, in addition to the previous ones. These unique features add significant complexity to the SoS development, evolution, and management.

Take the recent distributed maritime operations (DMOs) as an example. The U.S. Navy aims to design a cost effective and resilient SoS capable of performing the DMO [6]. This SoS would involve plenty of distributed platforms, sensors, weapons, communication facilities, and manning that are able to integrate together to support operations across the air, surface, undersea, and cyber domains. The large quantity and diversity of elements, connectivity, and involved stakeholders challenge the development, not to mention the emerging technologies in the unmanned systems, communication, combat cloud, etc. that bring in additional uncertainty and complexity.

Inspired by the architecture design in civil engineering that depicts a building's purpose, form, and construction at the beginning [7], architecture was introduced to the systems engineering (SE) and SoS engineering domain to manage the complexity and increase the successfulness of system development and integration. The latest ISO/IEC/IEEE 42010:2011 (Systems and Software Engineering—Architecture Description) defines architecture as "fundamental concepts or properties of a system in its environment embodied in its elements, relationships, and in the principles of its design and evolution." Majority of architecture related literature follows this definition, including the U.S. Department of Defense Architecture Framework (DoDAF). Crawley *et al.* [8] defined system (product) architecture as the "embodiment of concept, the allocation of physical/informational function to the elements of form, and the definition of relationships among the elements and with the surrounding context." Derived from the definition of system architecture, in the context of SoS, Levis *et al.* [9], Kenley *et al.* [10] and Raz *et al.* [11] considered SoS architecture as an allocated architecture generated by allocating a particular functionality from a functional architecture to the physical resources of a physical architecture, as shown on the left of Fig. 1. Simply put, SoS architecture defines the way the systems work together to meet the user needs. To effectively support the SoS design and integration, a well-designed architecture should be precise, comprehensive, credible, adaptable, and most importantly fit-for-purpose.

The current studies understand the SoS architecture mainly from two perspectives. Researchers such as Crawley *et al.* [8], Maier [12], and DeLaurentis *et al.* [13] consider architecture as a set of decisions that are encoded as variables with a set of allowed alternatives. Many others equate architecture with

Manuscript received 20 March 2021; revised 20 August 2021; accepted 3 October 2021. Date of publication 29 October 2021; date of current version 26 August 2022. This work was supported by Independent Innovation Funding of Huazhong University of Science and Technology under Grant 2019kfyXJJS17.

The author is with the National Key Laboratory of Science and Technology on Multispectral Information Processing, Huazhong University of Science and Technology, Wuhan 430074, China (e-mail: zmfang2018@hust.edu.cn).

Digital Object Identifier 10.1109/JSYST.2021.3119294

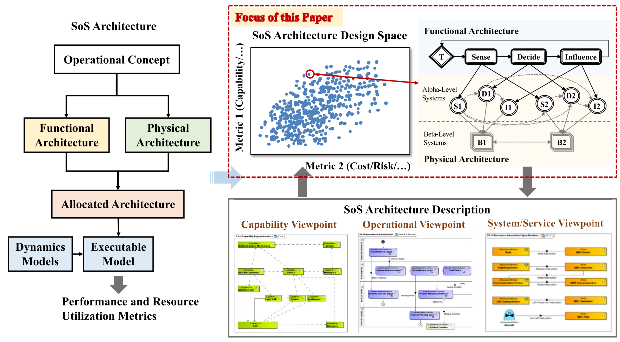

Fig. 1. Scope of SoS architecture selection in this article.

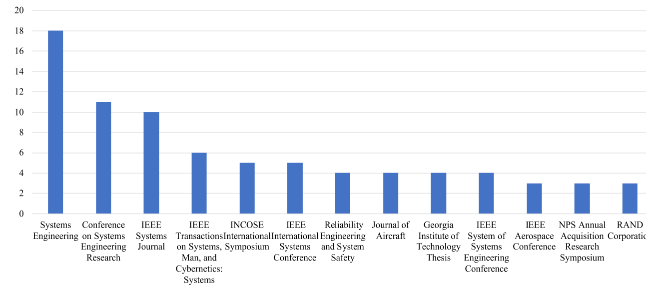

Fig. 2. Major sources of referred publications.

a set of description models that are supported by architecture frameworks. For example, a representative DoDAF-based SoS architecture is a structured package of capability mission, operational tasks, services, physical systems, functions, and the inter-relationship among these elements. As illustrated in Fig. 1, the two types of understanding are closely associated with each other. On one hand, there is a need to select an appropriate set of functions and physical entities from many alternatives prior to building more informative architecture descriptions; on the other hand, the DoDAF models with determined requirements, functions and systems provide useful guiding information to conduct optimal architecture selection. This review considers architecture as abstract-level decisions and the scope is shown on the upper-right part of Fig. 1.

An SoS architecture selection problem concerns selecting appropriate (physical) systems to fulfill functions, forming design space of architecture alternatives, and picking out alternatives that are favored by the stakeholders. The terms of SoS portfolio [14]–[16], composition [16], [17], and mix [18] are considered interchangeable with abstract-level SoS architecture (not the architecture framework supported models). This architecture selection process is severely challenged by the distinguishing SoS traits, such as interdependency, emergent effect, uncertainties, spatial-temporal dynamic changes, and system autonomy. Many researchers in the SoS community and associated domains, particularly in aerospace and defense area, have spent significant effort in developing useful methods, processes and tools to address these critical challenges.

While most studies focus on solving specific issues, such as interdependency and uncertainty, few review papers are available regarding the SoS architecture selection. Klein and Vilet [19] reviewed the status of architecture research in 2013 while Gorod *et al.* [20] reviewed the issues on SoS engineering management in 2009. There is a pressing need for a report that systematically summarizes the developed methods and techniques, and brings together the different pieces to build a panorama for SoS architecture selection research. This article reviews the current status of SoS architecture selection studies from seven perspectives, including conceptual framework, evaluation criteria, interdependency, uncertainty, autonomy, dynamic evolution, and computational methods. Based on the review, the article further suggests potential directions for future research on SoS architecture selection.

II. Research Method for the Review

This review follows the steps of collecting relevant literature, identifying critical issues, reviewing current methods, and indicating the research gaps and opportunities.

Fig. 2 demonstrates the major sources of referred publications for review, primarily are journal and conference papers, technical reports and dissertations. Due to the prevalence of SoS in the aerospace and defense area, many of the methods were initially developed for military/combat/defense SoS, although most of them are transferrable to other research areas. Many domain-specific sources brought by the multidisciplinary nature of SoS are excluded from the chart due to space limit.

To collect adequate literature, this article identifies the journals and conferences highly associated with system/SoS engineering (see Fig. 2) and search the keywords including "system-of-systems," "architecture," "meta-architecture," "architecting," "portfolio," "mix," "composition," and their combination. Additional papers are then gradually added through searching the above keywords in Web of Science and paper/dissertation/report repository of the notable SoS research groups from various universities [e.g., universities involved in the project of the U.S. DoD Systems Engineering Research Center (SERC)] and institutes (e.g., MITRE, RAND).

To organize the review for SoS architecture selection methods, this article identifies several critical issues that are grouped into seven categories: conceptual framework; evaluation criteria; interdependency; uncertainty; autonomy; dynamic evolution; and computational methods. The identification of critical issues is enlightened by three perspectives: the SoS traits, the SoS pain points identified by a survey conducted by the International Council on Systems Engineering (INCOSE) SoS working group in 2012 [21], [22], and the three categories of architecture design variables (i.e., composition, topology, and control) [13]. The identified issues mainly concern the questions of how to collect and organize the required information, how to proceed, how to evaluate the architecture, how to incorporate the unique features, and how to optimize the architecture.

Take the naval warfare SoS towards DMO as an example. A conceptual framework can provide a formalized and structured procedure to collect and organize information such as operational environment, mission tasks, in-service systems, etc. The architecture evaluation criteria could be resilience and cost-effective, as mentioned in [6]. The interdependency

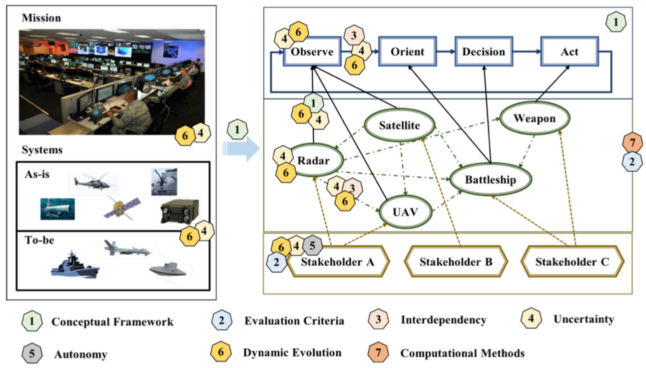

Fig. 3. Critical issues in SoS architecture selection problems.

(e.g., functional dependency between engagement and detection, interoperability between F-35 and E-2D), system autonomy (e.g., independent operation of satellites), uncertainty (e.g., adversary capability, new technology), and dynamic evolution (e.g., mission change, system aging) increase both the importance and difficulty of architecture design. These elements further aggravate the computational burden of selecting a desired architecture. Fig. 3 demonstrates the general locations where these issues occur in the SoS architecture selection process using a notional naval combat SoS example.

III. Review of Methods Regarding Critical Issues

A. Conceptual Framework

A conceptual framework provides high-level goals, basic structures or step-by-step procedures to understand and solve a given problem. Capability-based planning (CBP) is a guiding principle to establish conceptual frameworks for SoS architecture development. CBP refers to planning under uncertainty to provide capabilities suitable for a wide range of challenges and circumstances within economic limit [23]. The U.S. Joint Chiefs of Staff [24] defined capability as the ability to achieve a desired effect under specified standards and conditions through combinations of means and ways to perform a set of tasks. As the primary initiative for supporting SoS development, even though the term "CBP" itself has gradually phased out, the CBP thinking has already pervaded the major U.S. defense acquisition decision support systems such as the Joint Capabilities Integration and Development System. For example, the new SoS concept, mission engineering, combines SE process and operational planning to deliver SoS "capability" [25]. The CBP thinking has been a major guideline over years for selecting and designing SoS architectures.

Davis *et al.* [26], [27] in the RAND Corporation initiated the investigation of capability-based SoS architecture selection. The authors developed a seven-iterative-step framework for assessing the capability options and employed a portfolio-analysis-tool (PAT) that involves the techniques of hierarchical analysis, scorecard evaluation, and nonlinear aggregation for option evaluation. Many of the past studies on SoS architecture selection are directly influenced by or share similarity with the RAND's study. For example, the Aerospace Systems Design Laboratory [28], [29] has developed several procedural frameworks for capability-based SoS alternative analysis and evaluation with different focuses. Mokhtarpour and Stracener [30] proposed a typical six-step conceptual framework: describing SoS mission; identifying candidate systems; selecting feasible candidate systems; determining SoS alternatives; evaluating SoS alternatives in terms of decision factors; and selecting the preferred SoS depending on the situation. Under the CBP-supported frameworks, component systems are selected and linked together for delivering capability directly on missions.

The "wave" model for SoS development [31]–[33] also serves as a higher-level temporal or logical-perspective framework [14]. It consists of four iterative decision parts including "conduct SoS analysis," "develop/evolve SoS architecture," "plan SoS update," and "implement SoS update." Architectural frameworks such as DoDAF models [34], [35] and some other forms of hierarchical representation [11], [13], [14] provide semi-finished architectures that cover tactical products (e.g., system viewpoint, physical architecture) and strategic products (e.g., capability viewpoint, functional architecture) to guide further architecture selection.

B. Evaluation Criteria

Capability is one of the most important criteria for SoS evaluation due to the prevalence of various CBP-based frameworks. Measure of effectiveness (MoE) is a common capability-level metric that provides a measurable way of evaluating how well a proposed solution provides capabilities to achieve a desired result [36]. MoEs are often converted from measure of performances (MoPs) that tie to a specific physical implementation. The Expected Utility Theorem can help to build the relationship between MoPs and MoEs. That is, the effectiveness of an SoS is defined by utility or value. Multiattribute utility theory (MAUT) and additive value model (AVM) enable the aggregation of different MoEs with proper models and weights. Davendralingam *et al.* [14], Fang *et al.* [37], Vascik *et al.* [38], Dieffenbacher [39] among many others employed MAUT and AVM to evaluate the effectiveness of SoS architecture alternatives. Regarding "value," Collopy and a number of researchers [40], [41], [42] in the aerospace domain are promoting the concept of value-driven design in which system attributes are converted to economic values so that system design becomes budgeting system attributes among many design teams.

Risk is also a frequently used criterion that evaluates the impact of uncertainty on SoS capability and helps to form a trade-off design space with capability. Identified SoS risks include constituent system abandonment, exile, sociotechnical risks, risks related to dependability, evolutionary risks, and hazards [43]. The idea of capability-risk trade-off analysis in the SoS context originated from the financial modern portfolio theory (MPT). The classical risk quantification methods in finance, including variance, semi-variance, value-at-risk (VaR), and conditional value-at-risk (CVaR), illustrate great potential in managing the portfolio risks for SoS architecture. Davendralingam *and* DeLaurentis [14] and Zhang *et al.* [44] employed co-variance

TABLE I
INTERDEPENDENCY ANALYSIS METHODS IN SOS ARCHITECTURE SELECTION

Category	Specific Methods	References	Interdependency Explanations
Simulation-based methods	Agent-based modeling (ABM)	[64], [65]	Message passing between different agents and agent interaction rules
	System dynamics	[66]	Causal influence among variables
	Petri nets	[67], [68]	Token flows from places to transitions
Probability-based methods	Bayesian network	[69], [70]	Conditional probability
	Influence net	[71]	Conditional probability for positive and negative influence respectively
	Markov chain	[72]	Transition probability from one state to another
	Co-variance	[14]	Co-variance between system development time
Network-based methods	Network theory	[73], [74]	Links between nodes and weights on links
	Super-network	[75], [76]	Links between nodes in a super-network combined by two or more networks
Parametric-based methods	Input-output model	[77], [78]	Linear relationship between input and output
	Functional dependency network analysis (FDNA)	[79], [80]	Strength of dependency (SOD) and criticality of dependency (COD)
	Systems operational dependency analysis (SODA)	[81], [82]	SOD, COD and impact of dependency (IOD)
	Data-driven parametric models	[83], [84]	Data fitted model reveals interdependency
Other methods	Mathematical constraints	[14]	Compatibility, prerequisite, resource competition, information relay, etc.
	Share states	[85]	Share states, exchanged information, etc.

and variance as risk indicators for system development delay and development stability. Davendralingam and DeLaurentis [45] and Shah *et al.* [46] initiated the application of CVaR to SoS portfolio optimization in order to capture the downside risk of capability loss. User-defined risk functions are also common. Wei *et al.* [47] defined risk as remaining useful life that can be predicted by time series analysis of system operational data. Zhang *et al.* [48] considered risk as the probability of a project failing to provide certain benefit in a specified period.

Nonfunctional attributes, such as flexibility, robustness, resilience, changeability, survivability, agility, and affordability (especially resilience [49]) pervade the areas of complex systems and SoSs in the recent decade. Often referred to "ilities" [50], these features clearly demonstrate the special needs differentiating SoS architecture from project portfolio. A thorough explanation and clarification of these "ilities" can refer to de Weck's paper [51]. While many papers concentrate in the quantification of nonfunctional attributes to support monolithic complex system design [52]–[54], most claimed that the proposed methods are transferrable to SoS problems. There are also papers [55]–[60] focusing on defining and quantifying flexibility, robustness, resilience and so on for SoS problems and meanwhile employing these attributes to evaluate and improve SoS architecture alternatives [14], [56], [59]–[63].

C. Interdependency

The INCOSE SoS working group identified interdependency and its resultant cascading effect as one of the seven SoS pain points [21], [22]. As the root cause of cascading failures that often fail the entire SoS, the criticality of interdependency analysis raises concerns in both the SE/SoS community and specific disciplines, such as aerospace and civil engineering. This review categorizes the current methods for describing, analyzing, and assessing the impact of interdependencies for SoS architecture selection into five types: simulation-based methods; probability-based methods; network-based methods; parametric-based methods; and other methods. Table I gives a summary of the five categories of methods used for architecture selection.

Each type of the methods given in Table I has pros and cons. Simulation-based methods can incorporate sufficient details in high-fidelity models but they could be too costly when evaluating a large number of potential SoS architecture alternatives. Probability-based methods can easily capture the cascading effect of component system failures in a relatively simple way, but they are short of capturing the effect of partial dependency. Network-based methods that highly abstract an SoS problem are able to demonstrate the SoS features from a topological and statistical perspective (e.g., degree centrality, betweenness centrality). However, node-link abstraction might miss information of specific component systems and interactions that are critical to an SoS. Parametric-based methods strike a compromise between high modeling complexity as in simulation-based methods and coarse granularity as in network-based methods. The key challenge is to determine suitable parametric structure and appropriate parameters with actual physical explanations to represent the interdependencies. Mathematical constraints that are often incorporated in portfolio optimization problems are favored by their simplicity but might lose too much detailed information of an SoS problem. Share states often appear in nonlinear dynamic control systems and require identifying specific state-space functions. Method choice to deal with the interdependency hinges on the mission objective, problem size, known information, and computational resources among many others.

D. Uncertainty

The essence of CBP is joint capability planning under "uncertainty" that is ubiquitous in the SoS problems. A framework proposed by Walker *et al.* [86], [87] is employed to organize the uncertainty-related methods. Walker *et al.* proposed the framework to manage uncertainties for strategic (e.g., environmental, climate) decision making from three dimensions—location, level, and nature. I adopt the "location" dimension that indicates the source and location of uncertainty and select the context, model, and outcome as representative aspects. The "level" dimension is also used, which ranks from one to four based on the availability and quality of information (high availability and

TABLE II
METHODS DEALING WITH UNCERTAINTY IN SOS ARCHITECTURE SELECTION

Level	Location	Specific Location of Uncertainty	Solution Approaches	Refs
Level 2 (Alternate futures with probability)	Context (i.e., external forces)	Future terrorism risk (lognormal distribution)	Trinomial decision tree & Real options analysis	[88]
	Models (i.e., systems and their interactions)	System development schedule (normal distribution) & Dependency	Covariance as risk & Modern portfolio theory	[14]
		System development rate & Dependency	Covariance as risk & Multi-objective formulation	[44]
		System operability (beta distribution)	Systems operational dependency analysis	[81]
		System dependency (conditional probability)	Bayesian network	[69]
		System parameters	Stochastic agent-based simulation	[65]
	Outcomes (i.e., preferences)	System preferences on negotiation with SoS	Probability based negotiation process	[89]
Level 3 (A few plausible futures: set theory or fuzzy theory)	Context	Technology, policy, adversary, etc.	Scenario analysis & Epoch descriptor impact matrix & tradespace exploration	[38], [39]
		Consumed resources	Triangular fuzzy number	[48]
	Models	Weights on system attributes	Fuzzy linguistic variable value	[44]
	Outcomes	Preferences on SoS attributes (e.g., performance, flexibility)	Fuzzy evaluation	[62]
Level 4a (Many plausible futures: interval estimate)	Context	Adversary capability	Robust decision making (RDM) (including scenario analysis, vulnerability analysis and robust trade-off)	[90]
	Models	System capability index	Robust optimization & semi-definite programming	[14]
		System values on specific performance criteria	EVIKOR method to obtain system score	[44], [91]
	Outcomes	Not found		

quality: 1; low availability and quality: 4). Many architecture selection methods addressed one entry in the uncertainty matrix formed by “location” and “level.” Some of the entries in the uncertainty matrix might have not been explored yet. Table II summarizes the methods coping with uncertainties in the SoS architecture selection problems. The table does not include level 1 that is equivalent to deterministic situation with sensitivity analysis, and level 4b uncertainty that represents completely unknown futures.

According to the summary in Table II, uncertainties that are located in external environment, systems, structure, and stakeholder preferences are often represented by stochastic theory, fuzzy theory, set theory, and interval theory. Resultant problems are often addressed by simulation-based or optimization-based methods towards the objectives of optimization, satisfaction and prioritization, based on criteria such as performance, risk, cost, and robustness. Scenario analysis is a basic method that can be widely combined with other approaches, regardless of the uncertainty level. When it comes to the model uncertainties, they are often associated with constituent system attributes, weights of these attributes, connections between systems and so on. Different representations of uncertainties and evaluation criteria call for different solution approaches. For example, the pursuit of robustness in cases where input data are uncertain without known probability distribution resorts to robust optimization. Additionally, a variety of scoring techniques in decision theory, such as VIKOR, EVIKOR (extension of VIKOR), analytical hierarchy process, TOPSIS, etc., are applicable to evaluating and ranking attribute weights (referring to performance attribute) and criteria weights. Based on our observation, studies on preference uncertainties of SoS outcomes have not received much attention yet.

E. Autonomy

Autonomy or independence of constituent systems is a significant challenge for selecting desired SoS architecture. Because interests of systems and SoS might not be compatible with each other. A constituent system can choose to participate in an SoS depending on its own willingness. Among the four typical SoS types—directed SoS, acknowledged SoS, collaborative SoS and virtual SoS (in order from most to least centrally directed), the constituent systems in the latter three types of SoS have to make a balance between autonomy and cooperation (i.e., individual interest and SoS interest).

Three types of methods are used to deal with the autonomy issue—individual priority guaranteed, game theory, and negotiation. “Individual priority guaranteed” means a component system has the priority to satisfy its own requirements. Konur *et al.* [92] used a fixed binary flexibility indicator to represent a system’s willingness to join an SoS and incorporates the indicator in portfolio optimization. Davendralingam and DeLaurentis [14] set requirement constraint that a constituent system has to satisfy. The “individual priority guaranteed” methods simplifies the problem caused by system autonomy, but it may miss the complex relationship between component systems and overall SoS.

Game theory is a recognized method to address the autonomy, cooperation, and competition issues in an SoS [93]. Baldwin *et al.* [94] developed a game theoretic simulation for exploring equilibrium of autonomy and belonging in a simple hypothetical SoS. Konur and Dagli [95] employed Stackelberg games to model the relationship between systems and SoS and balance the local and global requirements through budget allocation and negotiation. Grogan *et al.* [42] adopted Stag Hunt game to model an SoS where players can choose between independent (noncooperative) and federated (cooperative) strategies. Axelsson [96] reviewed current applications of game theory to SoS. He found that Nash equilibrium as the basis of game theory was actually rarely discussed in the SoS applications. Consistent with his findings, I also observe that application of game theory to SoS architecture selection problems often stays at a very theoretical level or uses the concept of game rather than mathematical analysis.

Negotiation, also in the realm of generalized game theoretical methods, is another research idea for balancing independence and collaboration. Incentive mechanism design (i.e., reverse game theory) is often incorporated in the negotiation process. Qin *et al.* [89] developed a negotiation framework with fast evaluation of given proposals and generation of alternative solutions. Kilicay-Ergin and Dagli [97] modeled acknowledged SoS as a principal-agent problem and designed budget-based mechanism and multiround negotiation process to enhance system participation. Fang *et al.* [37], [98] employed transfer contract mechanism and approximate dynamic programming (ADP) to approach SoS capability optimization through self-negotiation and value transferring between constituent systems.

F. Dynamic Evolution

In theory, an SoS is almost never completely formed. Evolutionary development often reflects a process of progressive changes in the involved entities. The concept of SoS evolution can be understood in both general sense and narrow sense.

In general sense, SoS evolution covers all the aspects in the SoS development and design. Official documents, such as Systems Engineering Guide for SoS [32] and U.S. Defense Acquisition Guidebook [99], as well as DoDAF and model-based systems engineering relevant methodologies have been used to formalize the development and evolution process to improve efficiency and reduce risks. Lehman [100], a pioneer in software evolution research, developed eight important laws that motivate software evolution, including continuing change, increasing complexity, self-regulation, conservation of organizational stability, conservation of familiarity, continuing growth, declining quality, and feedback system. These laws also guide the studies of SoS evolution. Carney *et al.* [101] discussed the motivation, locality and outcome of SoS evolution by drawing upon the software evolution experience. Additionally, Lock [102] took a risk analysis perspective to evaluate possible changes and suggest proper actions to guide SoS evolution.

In narrow sense, dynamic evolution process of SoS architecting can be considered as a multistage decision-making problem in which new systems are added, current systems are replaced or removed, and the interdependency network can switch to a more efficient structure over time [103]. An important line of research that is related to the quantitative support for SoS evolution is dynamic strategic planning (DSP) [104]. Multiple methods have been developed to support the concept of DSP, including real options analysis (ROA) [88], epoch-era analysis (EEA) [105], time-expanded decision network (TDNs) [106], graph theory [107], etc. Intelligent algorithms are often combined with or used by these methods to support the computation.

Intelligent algorithms, primarily including evolutionary algorithm (EA) [108] and genetic algorithm (GA) [92], [97], [109] are also directly adopted as a computational framework to formulate and resolve the SoS architecture development process. Compared to intelligent algorithms, ADP, also known as reinforcement learning (RL) in other domains, illustrates great potential in representing the evolutionary process intuitively and straightforwardly. ADP is a method combining dynamic programming with stochastic approximations. Approximation allows us to alleviate the computational complexity resulted from the large number of systems, interactions, and uncertainties in an SoS. Maier [110] has proposed dynamic programming as a promising method to formulate the SoS management problem. Fang *et al.* [37], [111] and Davendralingam and DeLaurentis [112] leveraged the advantages of ADP to solve the SoS architecture evolution problem using a multistage portfolio selection formulation. Again, intelligent algorithms can support the optimization inside an ADP formulation.

The pervasion of big data, machine learning (ML), and artificial intelligence (AI) today brings new opportunities to the research on SoS architecture evolution. For example, in air transportation systems where historical data are available and easy to obtain, Kotegawa [113] employed ML techniques such as logistic regression and random forests to forecast route addition and removal.

G. Computational Methods

Most optimization methods and techniques are applicable to the SoS architecture optimal selection problem with objective functions and constraints. This section describes the computational methods from the aspects of mathematical modeling approaches and solution approaches.

1) Modeling Approaches: Formulating an SoS architecture selection problem requires to identify the objectives, decision variables, constraints and uncertain factors. Most SoS architecture selection problems can be formulated as multiobjective (e.g., performance, risk, schedule, safety) optimization models. Two common methods used to deal with the multiobjective issue are: picking one of the objectives as objective function while using others as constraints [14], [37], [46]; and using weighted average method to include all objectives [44], [114]. Meanwhile, researchers employed tradespace exploration method [38], [39] to provide decision-makers a visualized platform to make informed and balanced decisions.

The structures of objective functions vary with problem characteristics, and result in different types of mathematical models. Integer programming (IP) [37], [44], [114], mixed integer linear programming (MILP) [98], and nonlinear quadratic programming (QP) [14] are commonly adopted models. The uncertain factors add another level of complexity to both the objective functions and constraints. The added uncertainty requires stochastic optimization models and associated techniques listed in Table II. For example, Davendralingam and DeLaurentis [14] employed Bertsimas–Sim's robust linear formulation to address parametric data uncertainty without excessively penalizing the objective function.

Multiple decision stages could also complicate the models. Markov decision process (MDP) and MDP-based dynamic programming or ADP have gradually received attention for formulating multistage SoS architecture selection process [37], [111], [115], [116]. Each architecture alternative is assimilated to an MDP state that respects the Markov property as they contain the description of the full SoS architecture.

2) Solution Approaches: Exact and heuristic algorithms are both commonly used to solve the optimization problems formulated as IP, MILP, and QP models for SoS architecture selection. Exact algorithms (see [14], [37], [46], [98]), including benders decomposition, cutting-plane approach, branch-and-bound algorithm and more, aim to provide optimal solutions exactly. These methods are more suitable for small size problems and might encounter computational difficulty when the problem size increases. Optimization tools such as CPLEX, GUROBI, and YALMIP provide solver packages for conducting the exact algorithms.

Heuristic algorithms (including metaheuristic and hyper-heuristic algorithms as well, see [44], [47], [114], [117], [118]) are applied more frequently than exact algorithms to the SoS architecture selection problems due to their less strictness on assumptions. Typical heuristic algorithms include EA, ant colony optimization, particle swarm optimization among many others. SoS architecture selection related optimization problems often employ GA and differential evolution algorithm that are both under the realm of EA. Multiobjective problems primarily adopt nondominated sorting genetic algorithm II (NSGA II), NSGA III, and nondominated sorting differential evolution algorithm. As a multiobjective EA that uses nondominated sorting and crowding distance method to maintain the diversity of Pareto frontiers, NSGA II has been applied the most to the SoS architecture selection problems compared to other methods.

ADP models that straightforwardly illustrate the evolutionary decision-making process need to design appropriate and effective policy strategies (e.g., approximation, learning), in addition to the above-mentioned optimization algorithms. Powell [119], [120] categorized the ADP strategies into policy function approximations, cost function approximations, value function approximations, and direct lookahead approximations in his comprehensive review paper [119] of stochastic optimization that involves uncertainty, approximation and learning.

IV. Future Opportunities

A. Exploring Architecture Solutions for Encompassing Different SoS Paradigms

The review above reveals that current studies often focus on addressing a particular aspect for architecture design. The SoS community has been seeking for a viable design method that can encompass different SoS paradigms that vary in control degree, topology type, mission objectives, etc. This is a big vision of SoS engineering but is quite a challenging mission. A fit-for-all solution is not quite possible. The recent digital twin technologies might offer a great opportunity to collect adequate high-quality architecture data that could be used to be analyzed with different methods for addressing different issues during SoS architecting. The ideas have been explored preliminary in the SERC project [121] and the DANSE project [122]. However, how to formalize the process, generate credible data, and make good use of the data is still a question that needs to be further investigated.

B. Conceptual Framework Focused Opportunities

1) Integrated Conceptual Framework: "Integrated" here has twofold meanings. First, "integrated" refers to the integration of different conceptual frameworks. Many of the proposed frameworks help frame the goals, procedures, modelling standards and analysis techniques for the architecture selection problem. CBP-based procedural frameworks, wave style process models, architecture descriptions and hierarchical representations are not in a separate or opposing relationship. On the contrary, they overlap and can complement each other towards a better and more useful framework. For example, capability-oriented procedural framework can incorporate architecture descriptions (e.g., DoDAF models) into some of its procedures to conduct more informed analysis. While the importance of developing an integrated and practical framework is recognized [22], [28], there is still a lack of a rigorous, comprehensive, and ready-to-use integrated conceptual framework. Such a framework requires a deeper understanding of the current architecting implementation pain points, architecture frameworks and their fundamental ontology, and the purpose of architectures. Java frameworks, such as Spring and Hibernate provide a template for the final look of the conceived integrated framework for SoS architecture selection. The key point is to simplify rather than complicate the architecture selection process.

Second, "integrated" refers to the integration of conceptual framework and analytical methods. A read-to-use framework requires many plug-in modules to support different types of analysis (e.g., simulation-based analysis, optimization-based analysis, statistics-based analysis). Hence the available information organized as partial architectural models (e.g., DoDAF models) should be transferred to the analysis modules automatically and precisely. Past work has conducted sketchy mapping between DoDAF models and simulation models [67], [123], and between DoDAF models and optimization models [34], [35]. However, additional efforts are required to generate a smoother mapping between conceptual elements and analytical elements. It needs a thorough understanding of both the procedures and structures of an SoS problem and the factors involved in various analysis.

C. Evaluation Criteria Focused Opportunities

1) Architecture Level Risk Characterization and Quantification: SoS risk management itself is a separate and broad subject that involves many elements such as risk planning, assessing, monitoring, etc. Lopes *et al.* [43] summarized past studies on SoS risk management. With regard to SoS architecture selection, risk is often used for conducting architecture tradeoff analysis. These studies tend to develop or employ a risk index without thoroughly understanding, identifying and defining the risks. Although multiple types of risk metrics are used by different researchers, many questions are still unclear. For example, is one single selected type of risk capable of representing SoS risks? Is a probability-based risk metric able to manifest the characteristics of risks? Is a quantified risk metric able to identify and evaluate the true weakness of an SoS architecture? Much work remains for establishing a mature SoS architecture risk assessment system.

2) Quantification and Tradeoff of Nonfunctional Attributes: Taking nonfunctional attributes into consideration for the SoS architecture evaluation is an inevitable trend. However, enhanced flexibility, adaptability, robustness, resilience, etc., often require increased complexity. A key question is how much added complexity (from modularity, redundancy, etc.) is needed for an SoS to achieve significant improvements of the aforementioned nonfunctional attributes [28]. Effective SoS complexity metrics and continuing efforts in quantifying the nonfunctional attributes are necessary to answer the question. In addition, the relationship between different "ilities" are nonlinear and complex. Alberts [124] took an initial exploration on the monotonous correlation between features, such as responsiveness, flexibility, adaptability, versatility, resilience and innovation, using agent-based simulation. The synergy effects between different nonfunctional attributes could be more complicated. For example, architecture with high flexibility (which is defined as the number of options to complete a task) could be more resilient to failures; however, excessive flexibility might reduce the resilience due to the difficulty of option selection. Hence, there is a need to develop more effective metrics, design more sophisticated experiments, and conduct deeper studies on the mechanisms behind the synergy. The further studies rely on a more profound understanding of the nonfunctional attributes.

D. Interdependency Focused Opportunities

1) Model and Data Driven Interdependency Effect Formulation: The "model" here refers to those revealing the activity, functional and physical relationship in an operational environment, such as the FDNA model and SODA model. The "data" refer to the input system parameters (e.g., radar detection range) and various output parameters (e.g., number of destroyed targets) from the simulation models or field tests. Few papers have integrated the two perspectives together. Nonetheless the integration could take advantages of both the activity/functional/physical models and the data and ML algorithms. While keeping the interdependency with real physical meanings, data-based algorithms can help to identify some missing interdependency factors.

2) Interdependency Incorporated SoS Architecture Optimization: Although the interdependency analysis has been under the spotlight of SoS research, past focus was primarily on the analysis of given architecture alternatives instead of selecting architectures from a large number of choices. A few studies [14] place rule-based interdependencies (e.g., compatibility, relay) as logic constraints while some others [116] directly assigned a quantified value to the synergy effect caused by the interdependency. Few studies consider the nonlinear relationship between SoS capability and component system capabilities in the objective function or constraints. The challenge lies in both the formulation and computational complexity of a nonlinear objective function or constraint. Optimizing SoS architecture with certain degree of fidelity requires a manageable and mathematically representable formulation with appropriate amount of details on the interactions. Piece-wise linear parametric models, such as FDNA or SODA that quantify the interdependency using several parameters with proper physical meanings can generate results close to simulation models [81], as long as with appropriate parameter identification. Hence, this type of parametric models is likely to be incorporated in the objective function or constraints of an architecture optimization problem.

E. Uncertainty Focused Opportunities

1) Robust Architecture Selection Under Deep Uncertainty: Though uncertainty has always been considered as a major challenge in the SoS problems, deep uncertainty has not been explicitly researched on in the SoS community. According to Walker *et al.* [86], [87], uncertainty of level 4a in Table II as "many plausible futures" and level 4b as "unknown future or unknown-unknown" are considered as deep uncertainty that receives heated discussion recently in the domain of climate and environment policy design. Deep uncertainty is defined as the condition in which analysts do not know or the parties to a decision cannot agree upon the appropriate models to describe interactions among a system's variables, the probability distributions to represent uncertainty about key parameters in the models, and/or how to value the desirability of alternative outcomes [87]. As above-mentioned, many SoS problems are inherently with deep uncertainty. Potential methods for addressing deep uncertainty [87] include info-gap decision theory, assumption-based planning, robust decision making (RDM), adaptive policy making, dynamic adaptive policy pathways, decision scaling, ROA, and EEA. The key words for these methods are possibility exploration, mix of known and unknown information, robust to change, and adapting and learning. These key words demonstrate great potential in selecting robust SoS architectures under severe uncertainties. In fact, some of them (i.e., RDM, ROA, EEA) methods have already been applied to the SoS architecting and evolution problems.

F. Autonomy Focused Opportunities

1) Partial Autonomy and Collaboration Mechanism in Acknowledged SoSs: System autonomy and its impact on SoS architecting have not received adequate attention yet. For example, the rules to guide the independently operated constituent systems towards desired SoS capability are fundamental to a successful SoS construction, but they still highly depend on human interactions without proper technical and analytical methods. PAT is a promising approach to deal with the conflict in acknowledged SoSs. The "principal" refers to the role of SoS manager that owns partial control over the SoS while the "agent" refers to the role of SoS participant that maintains independent objectives and development. Kilicay-Ergin and Dagli [97] initially borrowed the framework of PAT and applied to acknowledged SoS negotiation. Safarkhani *et al.* [125], and Vermillion *et al.* [126] recently applied PAT to complex system design (that shares similar structure with acknowledged SoSs) for more effective requirement allocation. The initial studies demonstrate the potential of PAT as a framework for addressing acknowledged SoS conflict coordination. When PAT targets at the relationship between SoS manager and SoS participants, transfer contract coordination mechanism [37], [98] focuses on the relationship between SoS participants. Current results of these methods stay at a theoretical level, thus more efforts are needed to reach theory maturation and get ready for real applications. In addition, most of the current work has not considered the possibility of irrationality of either the system-level decision-makers or the SoS-level decision-makers.

2) Trust Between Autonomous Constituent Systems in Collaborative/Virtual SoS: Architecture selection for collaborative or virtual SoSs is quite challenging since they do not have an overarching purpose and each constituent system has high-degree autonomy. Collaborative/virtual SoS architecting depends on the self-organizing capability of constituent systems and the trust level between them. Trust and reputation models in open multiagent systems [127] provide useful reference to address the trust issue between constituent systems. Distributed ledger technologies such as blockchain [128] can help to ensure security, integrity and trust between independent systems via complex mathematical algorithms.

G. Dynamic Evolution Focused Opportunities

1) Learning-Based Architecture Selection: Decisions made in the early phases have a very significant bearing on the subsequent evolution, hence architects need to develop effective framework to manage the SoS evolution and learning process. ML techniques, especially deep RL related algorithms, are applicable to the learning and prediction of SoS architectures. For example, Raman and D'Souza [129] proposed a decision learning model to ensure that prior experiences on past architecture design decisions are systematically factored into the assessment of uncertainty of the decision alternatives. As researchers gradually delve into the subject, they have to face two major challenges: developing effective algorithms for realistic SoS evolution problems, and learning from low-quality data or small data in the information-sensitive domains such as aerospace and defense. Solutions to these challenges not only rely on the development of advanced ML algorithms, but also depends on correct analysis of the target problem.

2) Evolving Architecture Design Space: SoS evolutionary architecture selection is often formulated with fixed size of design/decision variables and objective function. However, new design/decision variables might occur in the later stages and need to be added to the problem at that moment. Moreover, the structure of an objective function may need to be changed at a certain point. Both of them lead to an evolving design space. Not much work has been performed to address this issue. Some initial attempts [130] employed an adaptive random projection (ARP) method to deal with the evolving design space issue in a multidisciplinary SoS optimization problem.

H. Computational Methods Focused Opportunities

1) Computational Methods for Multidimensional Complexity Due to SoS Characteristics: The challenges for computational methods come from the need to incorporate the unique SoS characteristics. For example, the above-mentioned robust portfolio optimization method [14], [131] and the ARP method [130] were designed specifically to deal with the uncertainty issue and evolution issue in an SoS problem. Therefore, researchers need to understand the features of an SoS problem and connect the problem elements with algorithms instead of solely digging into the computational algorithms.

V. Summary

An abstract-level SoS architecture that accounts for the interdependency, uncertainty and autonomy among other SoS features builds the skeleton of an SoS. This skeleton is equally important as the more detailed architectural descriptions guided by architectural frameworks. Initial architecture selection based on quantitative approaches can help decision-makers move out of the past experience comfort zone and identify those better, more informed but perhaps unexpected solutions. The unique SoS traits challenge the traditional SE and portfolio selection methods and tools. It prompts the development of new, innovative and more suitable methods. This article reviewed these methods, tools, processes, and frameworks from seven perspectives that are identified critical for selecting the useful SoS architecture alternatives. They are conceptual framework, evaluation criteria, interdependency, uncertainty, autonomy, dynamic evolution, and computational methods. The existing methods regrading each aspect were reviewed thoroughly to provide a clear picture of the SoS architecture selection research status.

Several future research opportunities are further identified in developing more effective approaches for SoS architecture selection. Some of them are quite challenging. For example, learning-based architecture selection is difficult to implement due to high-degree complexity and lack of high-quality data. Deep understanding of the target problem and proper leverage of AI and ML algorithms are both required to build a learning-incorporated SoS architecting process. Some of the opportunities may receive progress in the near term, such as the architecture design space exploration and optimal selection considering the interdependency.

Acknowledgment

The author would like to thank Prof. D. DeLaurentis, Prof. M. Wang, Dr. D. Wang, and the anonymous reviewers and editors for their valuable comments.

References

[1] M. Jamshidi, *System of Systems Engineering: Innovations for the 21st Century*. Hoboken, NJ, USA: Wiley, 2009.
[2] M. Maier, "Architecting principles for Systems-of-Systems," *Syst. Eng.,* vol. 1, no. 4, pp. 267–284, 1998.
[3] A. P. Sage and C. D. Cuppan, "On the systems engineering and management of systems of systems and federations of systems," *Inf., Knowl., Syst. Manage.,* vol. 2, no. 4, pp. 325–345, 2001.
[4] J. Boardman and B. Sauser, "System of systems - the Meaning of of," in *Proc. IEEE/SMC Int. Conf. Syst. Syst. Eng.*, 2006, pp. 1–6.
[5] D. DeLaurentis, "Understanding transportation as System-of-Systems design problem," in *Proc. 43rd AIAA Aerosp. Sci. Meeting Exhib.*, 2005, pp. 1–14.
[6] C. H. Popa *et al.*, "Distributed maritime operations and unmanned systems tactical employment," Naval Postgraduate School, CA, USA, Tech. Rep. AD1060065, 2018.
[7] C. E. Dickerson *et al.*, "Architecture definition in complex system design using model theory," *IEEE Syst. J.,* vol. 15, no. 2, pp. 1847–1860, Jun. 2021.

[8] E. Crawley, B. Cameron, and D. Selva, *System Architecture: Strategy and Product Development For Complex Systems*. London, U. K.: Pearson Education Ltd., 2016.

[9] A. H. Levis and L. W. Wagenhals, "C4ISR architectures: I. Developing a process for C4ISR architecture design," *Syst. Eng.,* vol. 3, no. 4, pp. 225–247, 2000.

[10] R. Kenley, D. Timothy, W. Paul, and D. DeLaurentis, "Synthesizing and specifying architectures for system of systems," in *Proc. 24th Annu. INCOSE Int. Symp*, 2014, pp. 94–107.

[11] A. K. Raz, C. R. Kenley, and D. DeLaurentis, "A System-of-Systems perspective for information fusion system design and evaluation," *Inf. Fusion,* vol. 35, pp. 148–165, 2017.

[12] M. W. Maier, "Architecting a portfolio of systems," *Syst. Eng.,* vol. 22, pp. 335–347, 2019.

[13] D. A. DeLaurentis, W. A. Crossley, and M. Mane, "Taxonomy to guide systems-of-systems decision-making in air transportation problems," *J. Aircr.,* vol. 48, no. 3, pp. 760–770, 2011.

[14] N. Davendralingam and D. DeLaurentis, "A robust portfolio optimization approach to system of system architectures," *Syst. Eng.,* vol. 18, no. 3, pp. 269–283, 2015.

[15] B. Ge, K. W. Hipel, L. Fang, K. Yang, and Y. Chen, "An interactive portfolio decision analysis approach for System-of-Systems architecting using the graph model for conflict resolution," *IEEE Trans. Syst., Man, Cybern., Syst.,* vol. 44, no. 10, pp. 1328–1346, Oct. 2014.

[16] N. Ricci and A. Ross, "Developing a dynamic portfolio-based approach for systems-of-systems composition," Massachusetts Institute of Technology, MA, USA, Tech. Rep. WP-2012-2-1, 2012.

[17] H. Derhamy, J. Eliasson, and J. Delsing, "System of system composition based on decentralized service-oriented architecture," *IEEE Syst. J.,* vol. 13, no. 4, pp. 3675–3686, Dec. 2019.

[18] Nusawardhana, "Dynamic programming methods for concurrent design and dynamic allocation of vehicles embedded in a System-of-Systems," Ph.D. dissertation, Sch. Aero. & Astro. Eng., Purdue Univ., West Lafayette, IN, USA, 2007.

[19] J. Klein and H. Vilet, "A systematic review of System-of-Systems architecture research," in *Proc. 9th Int. ACM Sigsoft Conf. Qual. Softw. Archit.*, 2013, pp. 13–22.

[20] A. Gorod, B. Sauser, and J. Boardman, "System-of-systems engineering management: A review of modern history and a path forward," *IEEE Syst. J.,* vol. 2, no. 4, pp. 484–499, 2009.

[21] J. Dahmann, "System of systems pain points," in *INCOSE Int. Symp.*, Las Vegas, NV, USA, pp. 108–121, 2014.

[22] J. Dahmann and G. Roedler, "Moving towards standardization for system of systems engineering," in *Proc. 11th System Syst. Eng. Conf.*, 2016, pp. 1–6.

[23] B. Bankston and T. Key, "White paper on capabilities based planning," Presented at Military Operations Research Society's Capabilities-based Planning II Workshop: Identifying, Classifying and Measuring Risk in a Post 9-11 World, McLean, VA, USA, 2006.

[24] "Capabilities-Based assessment (CBA) user's guide version 3 force structure, resources, and assessments directorate," Joint Chiefs of Staff J-8, Washington DC, VA, USA, 2009.

[25] G. Vesonder and D. Verma, "Mission engineering competencies technical report," *Systems Engineering Research Center*, Hoboken, NJ, USA, Tech. Rep. SERC-2018-TR-106, 2018.

[26] P. Davis, "*Analytic Architecture for Capabilities-Based Planning, Mission-System Analysis, and Transformation*. Santa Monica, CA, USA: Tech. Rep. MR-1513-OSD, RAND Corporation, 2002.

[27] P. Davis, R. Shaver, and J. Beck, *Portfolio-Analysis Methods for Assessing Capability Options*. Santa Monica, CA, USA: Tech. Rep. MG-662-OSD, RAND Corporation, 2008.

[28] J. C. Domercant, "ARC-VM: An architecture real options complexity-based valuation methodology for military systems-of-systems acquisitions," Ph.D. dissertation, Sch. Aero. Eng., Georgia Institute of Technology, Atlanta, GA, USA, 2011.

[29] J. V. Iacobucci, "Rapid architecture alternative modeling (RAAM): A framework for Capability-based analysis of system of systems architectures," Ph.D. dissertation, Sch. Aero. Eng., Georgia Institute of Technology, Atlanta, GA, USA, 2012.

[30] B. Mokhtarpour and J. Stracener, "A conceptual methodology for selecting the preferred system of systems," *IEEE Syst. J.,* vol. 11, no. 4, pp. 1928–1934, Dec. 2017.

[31] J. Dahmann, G. Rebovich, J. Lane, and K. Baldwin, "An implementer's view of systems engineering for systems of systems," in *Proc. IEEE Int. Syst. Conf.*, 2011, pp. 212–217.

[32] Systems Engineering Guide for System of Systems, Office of the Deputy under Secretary of Defense for Acquisition and Technology Systems and Software Engineering, Washington DC, VA, USA, 2008.

[33] C. Scrapper, R. Halterman, and J. Dahmann, "An implementer's view of the evolutionary systems engineering for autonomous unmanned systems," in *Proc. Annu. IEEE Syst. Conf.*, 2016, pp. 1–8.

[34] C. Guariniello, Z. Fang, N. Davendralingam, K. Marais, and D. DeLaurentis, "Tool suite to support model based systems engineering-enabled System-of-Systems analysis," in *Proc. IEEE Aerosp. Conf.*, 2018, pp. 1–16.

[35] Z. Fang, X. Zhou, and A. Song, "Architectural models enabled dynamic optimization for system-of-systems evolution," *Complexity,* vol. 2020, 2020, Art. no. 7534819.

[36] P. T. Biltgen, "A methodology for capability-based technology evaluation for system-of-systems," Ph.D. dissertation, Sch. Aero. Eng., Georgia Institute of Technology, Atlanta, GA, USA, 2007.

[37] Z. Fang, N. Davendralingam, and D. DeLaurentis, "Multistakeholder dynamic optimization for acknowledged system-of-systems architecture selection," *IEEE Syst. J.,* vol. 12, no. 4, pp. 3565–3576, Dec. 2018.

[38] P. D. Vascik, A. M. Ross, and D. H. Rhodes, "A method for exploring program and portfolio affordability tradeoffs under uncertainty using epoch-era analysis: A case application to carrier strike group design," in *Proc. 12th Annu. Acquisition Res. Symp.*, 2015, pp. 1–20.

[39] J. W. Dieffenbacher, "Managing portfolios of complex systems with the portfolio-level epoch-era analysis for affordability framework," M.S. thesis, Dept. Syst. Des. Manage., Massachusetts Institute of Technology, MA, USA, 2018.

[40] P. D. Collopy and P. M. Hollingsworth, "Value-Driven design," *J. Aircr.,* vol. 48, no. 3, pp. 749–759, 2011.

[41] R. Price and R. Malak, "A capability-based framework for supporting value-driven design," in *Proc. Conf. Syst. Eng. Res.*, Huntsville, AL, USA, 2016, pp. 1–10.

[42] P. T. Grogan, K. Ho, A. Golkar, and O. L. Weck, "Multi-actor value modeling for federated systems," *IEEE Syst. J.,* vol. 12, no. 2, pp. 1193–1202, 2018.

[43] S. S. Lopes, I. G. Vargas, A. L. Oliveira, and R. T. V. Braga, "Risk management for system of systems: A systematic mapping study," in *Proc. IEEE Int. Conf. Softw. Archit. Companion*, 2020.

[44] X. Zhang, L. Fang, K. W. Hipel, S. Ding, and Y. Tan, "A hybrid project portfolio selection procedure with historical performance consideration," *Expert Syst. With Appl.,* vol. 142, pp. 1–12, 2020.

[45] N. Davendralingam and D. DeLaurentis, "An analytic portfolio approach to system of systems evolutions," in *Proc. Conf. Syst. Eng. Res.*, 2014, pp. 711–719.

[46] P. Shah, N. Davendralingam, and D. A. DeLaurentis, "A conditional value-at-risk approach to risk management in System-of-Systems architectures," in *Proc. 10th Syst. Syst. Eng. Conf.*, 2015, pp. 457–462.

[47] H. Wei, B. Xia, Z. Yang, and Z. Zhou, "Model and data-driven system portfolio selection based on value and risk," *Appl. Sci.,* vol. 9, no. 8, pp. 1657–1675, 2019.

[48] X. Zhang, K. W. Hipel, and Y. Tan, "Project portfolio selection and scheduling under a fuzzy environment," *Memetic Comput.,* vol. 11, pp. 391–406, 2019.

[49] S. R. Goerger, A. M. Madni, and O. J. Eslinger, "Engineered resilient systems: A DoD perspective," in *Proc. Conf. Syst. Eng. Res.*, 2014, pp. 865–872.

[50] A. J. Middlebrooks, D. H. Rhodes, J. J. Cipolloni, and S. R. Goerger, "Broad utility: Architecting flexible and robust systems for a complex operational environment," in *Proc. 17th Annu. Conf. Syst. Eng. Res.*, 2019, pp. 335–342.

[51] O. L. D. Weck and A. M. Ross, and D. H. Rhodes, "Investigating relationships and semantic sets amongst system lifecycle properties (Ilities)," in *Proc. 3rd Int. Eng. Syst. Symp.*, 2012, pp. 1–13.

[52] C. F. Rehn *et al.*, "Quantification of changeability level for engineering systems," *Syst. Eng.,* vol. 22, no. 1, pp. 80–94, 2019.

[53] K. Neema, S. Tamaskar, C. Guariniello, and D. DeLaurentis, "Complexity and flexibility enabled model based design framework for space system design," in *Proc. AIAA SPACE Astronaut. Forum Expo.*, 2018, pp. 1–15.

[54] A. M. Ross, D. B. Stein, and D. E. Hastings, "Multi-Attribute tradespace exploration for survivability," *J. Spacecraft Rockets,* vol. 51, no. 5, pp. 1735–1752, 2014.

[55] A. Gorod, S. J. Gandhi, B. Sauser, and J. Boardman, "Flexibility of system of systems," *Global J. Flexible Syst. Manage.*, vol. 9, no. 4, pp. 21–31, 2008.

[56] B. H. Poole, "A methodology for the robustness-based evaluation of systems-of-systems alternatives using regret analysis," Ph.D. dissertation, Sch. Aero. Eng., Georgia Institute of Technology, Atlanta, GA, USA, 2008.

[57] H. T. Tran, J. C. Domercant, and D. N. Mavris, "Designing resilient System-of-Systems networks," in *Proc. Annu. IEEE Int. Syst. Conf. (SysCon)*, 2017, pp. 1–6.

[58] R. Filippini and A. Silva, "A modeling framework for the resilience analysis of networked Systems-of-Systems based on functional dependencies," *Rel. Eng. Syst. Saf.,* vol. 125, pp. 82–91, 2014.

[59] P. Uday, R. Chandrahasa, and K. Marais, "System importance measures: Definitions and application to system-of-systems analysis," *Rel. Eng. Syst. Saf.,* vol. 191, 2020, Art. no. 106582.

[60] C. Guariniello and D. DeLaurentis, "Integrated analysis of functional and developmental interdependencies to quantify and trade-off ilities for system-of-systems design, architecture, and evolution," in *Proc. Conf. Syst. Eng. Res.*, 2014, pp. 728–735.

[61] T. Nelson, J. M. Borky, and R. M. Sega, "System-of-systems quality attribute-based architectural alternatives," *IEEE Syst. J.,* vol. 14, no. 3, pp. 3844–3854, Sep. 2020.

[62] L. Pape, K. Giammarco, J. Colombi, C. Dagli, N. Kilicay-Ergin, and G. Rebovich, "A fuzzy evaluation method for system of systems meta-architectures," in *Proc. 11th Conf. Syst. Eng. Res.*, 2013, pp. 245–254.

[63] N. Ricci, M. E. Fitzgerald, A. M. Ross, and D. H. Rhodes, "Architecting systems of systems with ilities: An overview of the SAI method," in *Proc. Conf. Syst. Eng. Res.*, 2014, pp. 322–331.

[64] D. N. Fry, R. Campbell, and D. A. DeLaurentis, "Modeling systems-of-systems from multiple design perspectives: Agents, interfaces, and architectures," in *Proc. AIAA SciTech Forum*, 2015, pp. 1–15.

[65] A. Mour, C. R. Kenley, N. Davendralingam, and D. DeLaurentis, "Agent-based modeling for systems of systems," in *Proc. INCOSE Int. Symp.*, 2013, pp. 973–987.

[66] K. Moolchandani, P. Govindaraju, S. Roy, and W. Crossley, "Assessing effects of aircraft and fuel technology advancement on select aviation environmental impacts," *J. Aircr.,* vol. 54, no. 3, pp. 857–869, 2017.

[67] L. W. Wagenhals and A. H. Levis, "Service oriented architectures, the DoD architecture framework 1.5, and executable architectures," *Syst. Eng.,* vol. 12, no. 4, pp. 312–343, 2009.

[68] R. Wang, S. Agarwal, and C. Dagli, "Executable system of systems architecture using OPM in conjunction with colored petri net: A module for flexible intelligent and learning architectures for system of systems," in *Proc. INCOSE Int. Symp.*, 2015, pp. 581–596.

[69] S. Han, Z. Fang, and D. DeLaurentis, "Acquisition management for System-of-Systems: Requirement evolution and acquisition strategy planning," in *Proc. 9th Annu. Acquisition Res. Symp.*, 2012, pp. 241–251.

[70] S. Y. Han and D. DeLaurentis, "Development interdependency modeling for system-of-systems (SoS) using bayesian networks: SoS management strategy planning," in *Proc. Conf. Syst. Eng. Res.*, 2013, pp. 698–707.

[71] J. Sun, B. Ge, J. Li, and K. Yang, "Operation network modeling with degenerate causal strengths for missile defense systems," *IEEE Syst. J.,* vol. 12, no. 1, pp. 274–284, Mar. 2018.

[72] M. Mane, D. DeLaurentis, and A. Frazho, "A markov perspective on development interdependencies in networks of systems," *J. Mech. Des.,* vol. 133, no. 10, 2011, Art. no. 101009.

[73] D. DeLaurentis, E.-P. Han, and T. Kotegawa, "Network-theoretic approach for analyzing connectivity in air transportation networks," *J. Aircr.,* vol. 45, no. 5, pp. 1669–1679, 2008.

[74] S. Thacker, R. Pant, and J. W. Hall, "System-of-Systems formulation and disruption analysis for multi-scale critical national infrastructure," *Rel. Eng. Syst. Saf.,* vol. 167, pp. 30–41, 2017.

[75] Q. Zhao, S. Li, Y. Dou, X. Wang, and K. Yang, "An approach for weapon system-of-systems scheme generation based on a supernetwork granular analysis," *IEEE Syst. J.,* vol. 11, no. 4, pp. 1971–1982, Dec. 2017.

[76] J. Li, D. Zhao, J. Jiang, K. Yang, and Y. Chen, "Capability oriented equipment contribution analysis in temporal combat networks," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 51, no. 2, pp. 696–704, Feb. 2018.

[77] Y. Y. Haimes, B. M. Horowitz, J. H. Lambert, J. R. Santos, C. Lian, and K. G. Growther, "Inoperability input-output model for interdependent infrastructure sectors. I: Theory and methodology," *J. Infrastruct. Syst.,* vol. 11, no. 2, pp. 67–79, 2005.

[78] Y. Y. Haimes, B. M. Horowitz, J. H. Lambert, J. Santos, K. Crowther, and C. Lian, "Inoperability input-output model for interdependent infrastructure sectors. II: Case studies," *J. Infrastruct. Syst.,* vol. 11, no. 2, pp. 80–92, 2005.

[79] P. R. Garvey and C. A. Pinto, "Introduction to functional dependency network analysis," in *Proc. 2nd Int. Symp. Eng. Syst.*, 2009, pp. 1–17.

[80] L. D. Servi and P. R. Garvey, "Deriving global criticality conditions from local dependencies using functional dependency network analysis (FDNA)," *Syst. Eng.,* vol. 20, no. 4, pp. 297–306, 2017.

[81] C. Guariniello and D. DeLaurentis, "Supporting design via the system operational dependency analysis methodology," *Res. Eng. Des.,* vol. 28, pp. 53–69, 2017.

[82] C. Guariniello *et al.*, "System-of-systems tools for the analysis of technological choices in space propulsion," in *Proc. 69th Int. Astronaut. Congr.*, 2018, pp. 1–14.

[83] M. Monsalve and J. C. Llera, "Data-Driven estimation of interdependencies and restoration of infrastructure systems," *Rel. Eng. Syst. Saf.,* vol. 181, pp. 167–180, 2019.

[84] C. Guariniello, L. Mockus, A. K. Raz, and D. DeLaurentis, "Towards intelligent architecting of aerospace system-of-systems: Part II," in *Proc. IEEE Aerosp. Conf.*, 2020, pp. 1–9.

[85] Z. Guo and Y. Y. Haimes, "Exploring systemic risks in Systems-of-Systems within a multiobjective decision framework," *IEEE Trans. Syst., Man, Cybern., Syst.,* vol. 47, no. 6, pp. 906–915, Jun. 2017.

[86] W. Walker *et al.*, "Defining uncertainty: A conceptual basis for uncertainty management in model-based decision support," *Integr. Assessment,* vol. 4, no. 1, pp. 5–17, 2003.

[87] V. A. W. J. Marchau, W. E. Walker, P. J. T. M. Bloemen, and S. W. Popper, *Decision Making under Deep Uncertainty: From Theory to Practice*. Berlin, Germany: Springer, 2019.

[88] J. Buurman, S. Zhang, and V. Babovic, "Reducing risk through real options in system design: The case of architecting a Martime domain protection system," *Risk Anal.,* vol. 29, no. 3, pp. 366–379, 2009.

[89] R. Qin, C. H. Dagli, and N. Amaeshi, "A contract negotiation model for constituent systems in the acquisition of acknowledged system of systems," *IEEE Trans. Syst., Man, Cybern.: Syst.,* vol. 47, no. 11, pp. 3050–3062, Nov. 2017.

[90] R. J. Lempert, D. Warren, R. Henry, R. W. Button, J. Klenk, and K. Giglio, *Defense Resource Planning Under Uncertainty: An Application of Robust Decision Making to Munitions Mix Planning*. Santa Monica, CA, USA: RAND Corporation, 2016.

[91] Y. Dou, Z. Zhou, X. Xu, and Y. Lu, "System portfolio selection with decision-making preference baseline value for system of systems construction," *Expert Syst. Appl.,* vol. 123, pp. 345–356, 2019.

[92] D. Konur, H. Farhangi, and C. H. Dagli, "A multi-objective military system of systems architecting problem with inflexible and flexible systems: Formulation and solution methods," *OR Spectr.,* vol. 38, pp. 967–1006, 2016.

[93] W. A. Crossley, "System of systems: An introduction of Purdue University schools of engineering's signature area," presented at the *Eng. Syst. Symp.*, 2004.

[94] W. C. Baldwin, T. Ben-Zvi, and B. J. Sauser, "Formation of collaborative system of systems through belonging choice mechanisms," *IEEE Trans. Syst., Man, Cybern. Part A, Syst. Humans,* vol. 42, no. 4, pp. 793–801, Jul. 2012.

[95] D. Konur and C. H. Dagli, "Military system of systems architecting with individual system contracts," *Optim. Lett.,* vol. 9, pp. 1749–1767, 2015.

[96] J. Axelsson, "Game theory applications in Systems-of-Systems engineering: A literature review and synthesis," in *Proc. 17th Annu. Conf. Syst. Eng. Res.*, 2019, pp. 154–165.

[97] N. Kilicay-Ergin and C. H. Dagli, "Incentive-based negotiation model for system of systems acquisition," *Syst. Eng.,* vol. 18, no. 3, pp. 310–321, 2015.

[98] Z. Fang, K. Moolchandani, H. Chao, and D. DeLaurentis, "A method for emission allowances allocation in air transportation systems from a system-of-systems perspective," *J. Cleaner Prod.,* vol. 226, pp. 419–431, 2019.

[99] *Defense Acquisition Guidebook*. Washington, DC, USA: United States Department of Defense, 2008.

[100] M. M. Lehman, J. F. Ramil, P. D. Wernick, D. E. Perry, and W. M. Turski, "Metrics and laws of software evolution - the Nineties view," in *Proc. 4th Int. Softw. Metrics Symp.*, 1997, pp. 20–32.

[101] D. Carney, D. Fisher, and P. Place, "Topics in interoperability: System-of-systems evolution," Software Engineering Institute, Carnegie Mellon Univ., Tech. Memo. CMU/SEI-2005-TN-002, 2005.

[102] R. Lock, "Developing a methodology to support the evolution of system of systems using risk analysis," *Syst. Eng.,* vol. 15, no. 1, pp. 62–73, 2012.

[103] S. A. Selberg and M. A. Austin, "Toward an evolutionary system of systems architecture," in *Proc. INCOSE Int. Symp.*, 2008, pp. 1065–1078.

[104] R. de Neufville, "Dynamic strategic planning for technology policy," *Int. J. Technol. Manage.,* vol. 19, pp. 225–245, 2000.

[105] M. E.Fitzgerald and A. M.Ross, "Sustaining lifecycle value: Valuable changeability analysis with era simulation," in *Proc. 6th Annu. IEEE Int. Syst. Conf.*, 2012, pp. 1–7.

[106] M. Silver and O. De Weck, "Time-expanded decision networks: A framework for designing evolvable complex systems," *Syst. Eng.,* vol. 10, no. 2, pp. 167–186, 2007.

[107] P. Davison, B. Cameron, and E. F. Crawley, "Technology portfolio planning by weighted graph analysis of system architectures," *Syst. Eng.,* vol. 18, no. 1, pp. 45–58, 2015.

[108] W. Tan, B. J. Sauser, J. E. Ramirez-Marquez, and R. B. Magnaye, "Multiobjective optimization in multifunction multicapability system development planning," *IEEE Trans. Syst., Man, Cybern., Syst.,* vol. 43, no. 4, pp. 785–799, Jul. 2013.

[109] P. Acheson, L. Pape, C. Dagli, N. Kilicay-Ergin, J. Columbi, and K. Haris, "Understanding system of systems development using an agent-based wave model," in *Proc. Comput. Sci.*, 2012, pp. 21–30.

[110] M. Maier, "Research challenges for systems-of-systems," in *Proc. IEEE Int. Conf. Syst., Man Cybern.*, 2005, pp. 3149–3154.

[111] Z. Fang and D. DeLaurentis, "Multi-Stakeholder dynamic planning of system of systems development and evolution," in *Proc. Conf. Syst. Eng. Res.*, 2015, pp. 95–104.

[112] N. Davendralingam and D. DeLaurentis, "Promoting affordability in defense acquisitions: A multi-period portfolio approach," in *Proc. 11th Acquisition Res. Symp.*, 2014, pp. 319–331.

[113] T. Kotegawa, "Analyzing the evolutionary mechanisms of the air transportation System-of-Systems using network theory and machine learning algorithms," Ph.D. dissertation, Sch. Aero. & Astro. Eng., Purdue Univ., West Lafayette, IN, USA, 2012.

[114] K. Shafi, S. Elsayed, R. Sarker, and M. Ryan, "Scenario-based multi-period program optimization for capability-based planning using evolutionary algorithms," *Appl. Soft Comput.,* vol. 56, pp. 717–729, 2017.

[115] Z. Fang and D. DeLaurentis, "Dynamic planning of system of systems architecture evolution," in *Proc. Conf. Syst. Eng. Res.*, 2014, pp. 449–456.

[116] I. Lluch and A. Golkar, "Architecting federations of systems: A framework for capturing synergy," *Syst. Eng.,* vol. 22, pp. 295–312, 2019.

[117] P. Liu, B. Xia, Y. Tan, and D. Zhao, "Modeling and robust optimization for system of systems problems under uncertainty," in *Proc. IEEE 4th Int. Conf. Control Sci. Syst. Eng.*, 2018, pp. 385–390.

[118] M. Li, M. Li, K. Yang, B. Xia, and C. Wan, "A Network-based portfolio optimization approach for military system of systems architecting," *IEEE Access,* vol. 6, pp. 53452–53472, 2018.

[119] W. B. Powell, "A unified framework for stochastic optimization," *Eur. J. Oper. Res.,* vol. 275, no. 3, pp. 795–821, 2019.

[120] W. B. Powell, *Approximate Dynamic Programming: Solving the Curses of Dimensionality*, 2nd ed., Hoboken, NJ, USA: Wiley, 2010.

[121] D. DeLaurentis, N. Davendralingam, K. Marais, C. Guariniello, Z. Fang, and P. Uday, "An SoS analytical workbench approach to architectural analysis and evolution," *Insight,* vol. 20, no. 3, pp. 69–73, 2017.

[122] U. Shani *et al.*, "Designing for adaptability and evolution in system of systems engineering (DANSE) prototype - Final report," DANSE Consortium Seventh Framework Programme, 2015.

[123] R. Wang and C. H. Dagli, "Executable system architecting using systems modeling language in conjunction with colored petri nets in a model-driven systems development process," *Syst. Eng.,* vol. 14, no. 4, pp. 383–409, 2011.

[124] D. S. Alberts, *The Agility Advantage: A Survival Guide for Complex Enterprise and Endeavors*, CCRP Publication, 2011, Art. no. 8.

[125] S. Safarkhani, I. Bilionis, and J. H. Panchal, "Toward a theory of systems engineering processes: A principal-agent model of a one-shot, shallow process," *IEEE Syst. J.,* vol. 14, no. 3, pp. 3277–3288, Sep. 2020.

[126] S. D. Vermillion and R. J. Malak, "An investigation on requirement and objective allocation strategies using a principal-agent model," *Syst. Eng.,* vol. 23, no. 1, pp. 100–117, 2020.

[127] M. Sievers, A. M. Madni, P. Pouya, and R. Minnichelli, "Trust and reputation in multi-agent resilient systems," in *Proc. IEEE Int. Conf. Syst., Man Cybern.*, 2019, pp. 741–747.

[128] E. Bellini, Y. Iraqi, and E. Damiani, "Blockchain-based distributed trust and reputation management systems: A survey," *IEEE Access,* vol. 8, pp. 21127–21151, 2020.

[129] R. Raman and M. D'Souza, "Decision learning framework for architecture design decisions of complex systems and system-of-systems," *Syst. Eng.,* vol. 22, pp. 538–560, 2019.

[130] S. V. Subramanian and D. DeLaurentis, "Application of multidisciplinary systems-of-systems optimization to an aircraft design problem," *Syst. Eng.,* vol. 19, no. 3, pp. 235–251, 2016.

[131] N. Davendralingam and D. DeLaurentis, "A robust optimization framework to architecting system of systems," in *Proc. Conf. Syst. Eng. Res.*, 2013, pp. 255–264.

Zhemei Fang received the Ph.D. degree in control engineering from the School of Aeronautics and Astronautics Engineering, Purdue University, West Lafayette, IN, USA, in 2017.

She is currently an Assistant Professor with the School of Artificial Intelligence and Automation, Huazhong University of Science and Technology, Wuhan, China. Her research interests center on developing theories and methods for system-of-systems architecture design and evolution. The methods include but are not limited to model-based systems engineering, agent-based simulation, approximate dynamic programming, and reinforcement learning.