


# A network perspective on assessing system architectures: Foundations and challenges

Matthew W. Potts<sup>1</sup>  | Pia Sartor<sup>1</sup> | Angus Johnson<sup>2</sup> | Seth Bullock<sup>3</sup>

<sup>1</sup>Aerospace Department, The University of Bristol, Bristol, UK

<sup>2</sup>Thales Research, Technology and Innovation, Paris, France

<sup>3</sup>Department of Computer Science, The University of Bristol, Bristol, UK

## Correspondence

Matthew W. Potts, Aerospace Department, The University of Bristol, Avonmouth, Bristol BS8 1TB, UK.

Email: matt.potts@bristol.ac.uk

All authors contributed equally.

## Funding information

Engineering and Physical Sciences Research Council, Grant/Award Number: 16000139

## Abstract

Organizations are increasingly faced with the challenge of architecting complex systems that must operate within a System of Systems context. While network science has offered usefully clear insights into product and system architectures, we seek to extend these approaches to evaluate enterprise system architectures. Here, we explore the application of graph-theoretic methods to the analysis of two real-world enterprise architectures (a military communications system and a search and rescue system) and to assess the relative importance of different architecture components. For both architectures, different topological measures of component significance identify differing network vertices as important. From this, we identify several significant challenges a system architect needs to be cognisant of when employing graph-theoretic approaches to evaluate architectures; finding suitable abstractions of heterogeneous architectural elements and distinguishing between network-structural properties and system-functional properties. These challenges are summarized as five guiding principles for utilizing network science concepts for enterprise architecture evaluation.

## KEYWORDS

networks science, system architecture, systems of systems

## 1 | INTRODUCTION

Organizations are increasingly faced with the challenge of architecting and realising complex systems that must operate in the context of a System of Systems (SoS), or architecting and realising an entire SoS itself.<sup>1</sup> The Engineering Systems community has championed the importance of the *architecture* of an engineered system, in terms of its influence on functional behavior and desirable properties, such as robustness, flexibility, and resilience.<sup>2,3</sup> Traditionally, architecting approaches are *reductionist* in nature; seeking to decompose systems into smaller, more manageable chunks that can be reassembled into a whole with no adverse effects.<sup>4–9</sup> When architecting a complex system or complex SoS, such approaches are less likely to yield useful insights into the System of Interest (Sol; the system whose life cycle is under consideration<sup>10</sup>) due to a combination of significant system scale and heterogeneity, a high degree of interconnectedness or

interdependency within and between the component systems, and the nonlinearity of these interdependencies.<sup>11</sup> These properties can combine to frustrate a divide-and-conquer approach that is typically relied upon to manage and design systems.

Recent efforts have sought to bring complexity science to bear on systems engineering problems, for example, utilizing graph theory to explore and characterize the complex network structure of interconnections between Sol entities,<sup>12–16</sup> and considering how perturbations to such networks may provide insights into architectural robustness and flexibility.<sup>17</sup> The contribution of this paper is in constructing graphical models of two real-world SoS architectures directly from Architecture Views (AVs) created in accordance with an Enterprise Architecture Framework, using graph-theoretic analysis of these architectures to highlight challenges inherent to applying network science models and methods to architectures of this kind, and deriving a set of guiding principles to address these challenges. In doing so, it seeks to

Abbreviations: SoS, System of Systems; Sol, System of Interest.

This is an open access article under the terms of the Creative Commons Attribution License, which permits use, distribution and reproduction in any medium, provided the original work is properly cited.

© 2019 The Authors. *Systems Engineering* published by Wiley Periodicals, Inc.

contribute to a research agenda for the engineering of complex systems laid down nearly a decade ago; “How can we better model, visualize and understand networks of interdependencies, to achieve insights on the likely consequences of variations and perturbations (e.g., impact of changes to schedule or changes in performance of one component on other parts of the enterprise)?”<sup>18</sup>

The paper is structured as follows. For the remainder of this section, in order to avoid issues created by the ambiguity of key terms (eg, systems architecting, complex system, and SoS), they are defined for the purposes of this paper. Furthermore, in order to contextualize the systems architecting challenge, typical questions that an organization would ask when architecting systems are discussed. Two real-world use cases are then introduced; a generic Search and Rescue architecture (SAR) and a tactical military communications enterprise architecture (MComms). The theory supporting the assessments of architectures using a network perspective is presented. The following section presents a literature review of current graph-theoretic approaches to assessing system architectures. Architecture topology is then assessed for each of the two use cases. The discussion turns to the challenges of adopting a network perspective to assess a complex SoS architecture, highlighting the areas where care must be taken. Finally, further work is detailed, five guiding principles are suggested, and conclusions are drawn.

## 1.1 | Definitions

Some of the terminology used thus far can be prone to ambiguity. The purpose of this section is to make clear what each term means for the purposes of this paper.

### 1.1.1 | Complex system

The term complex system is poorly defined partly as a consequence of the variety of uses to which it is put. Various publications examine what complexity means to a systems engineer.<sup>19–23</sup> Complex systems can be described in terms of *dynamic complexity*, that is, how a system evolves or changes over time and the difficulty associated with predicting behavior,<sup>24,25</sup> or *socio-political complexity*, which is the complexity arising from human cognitive limits or from multiple stakeholders exerting different effects on a system.<sup>9,22</sup> Alternatively, complex systems can be described in terms of their *structural complexity*, which arises from a combination of system scale, connectivity, and heterogeneity.<sup>21,22</sup>

The starting point for this manuscript is that a systems engineer endeavoring to architect or design a complex system will initially be concerned with more objective characterizations of structural complexity and its relationship to system behavior, function, or performance. Thus, a complex system, for the purposes of this manuscript, is defined as a system formed of many interdependent components which, as a consequence of the nonlinear character of these interdependencies, has the capacity to exhibit emergence, implying that overall system behavior cannot simply be inferred from the behavior of system components.<sup>2,11,24,26–28</sup>

### 1.1.2 | System of systems

The term SoS is also loaded with ambiguity. A recent review of pertinent publications on SoS Engineering<sup>29</sup> identified the increasingly rich descriptions and classifications used, and noted that authoritative guidance and robust methodologies are still lacking for SoS Engineering. Early thinking by Maier<sup>30</sup> defined an SoS as “an assemblage of components with operational and managerial independence.”<sup>30</sup> Maier also postulated that the geographical distribution of a system's parts could be a contributory characteristic, but argued that geographical distribution alone is not sufficient for a system to qualify as an SoS. International standards do not provide much clarity. The draft annex to ISO/IEC/IEEE 15288, System Life Cycle Processes 2015,<sup>10</sup> defines an SoS as a “set of systems for a task that none of the systems can accomplish on its own,” echoing Maier's description but adding the notion of some emerging functionality that can only arise through the bringing together of the constituent systems. This point is further reinforced by Boardman and Sauser who set out five key characteristics of an SoS: autonomy, belonging, connectivity, diversity, and emergence.<sup>31,32</sup> The defining separation between a system and an SoS is equivalent to the separation between a component and a system of components—the notion of emergent properties at the aggregate level that cannot be attributed to their parts. Boardman and Sauser say it best: “The distinction lies in the meaning and significance of ‘gathering together’ – an SoS is much more because its parts, acting as autonomous systems, forming their own connections and rejoicing in their diversity, lead to enhanced emergence, something that fulfills capability demands that set an SoS apart.”<sup>31</sup> The draft ISO 21839, Systems of Systems Considerations in Engineering of Systems 2017,<sup>33</sup> similarly places emphasis on emergence as a defining characteristic of an SoS; the emergence of some new capability that none of the constituent systems have. Alternative definitions assert that an SoS is deliberately brought together for the purpose of a goal,<sup>34</sup> but this is problematic because evidence for this may be limited (eg, the City of London is arguably an SoS without a designer, purpose, or goal). The *a priori* existence of a common goal also excludes an SoS for which multiple conflicting goals can be identified, or for which a shared goal changes over time, or only arises after the systems have been established. Other definitions make explicit the evolving nature of the SoS in terms of new systems or services<sup>35</sup> or in terms of a lack of overall control of constituent system development lifecycles.<sup>36</sup> There are also disagreements concerning where SoS engineering should focus, with some arguing that it is more concerned with acquisition activity<sup>22,29,30,34,37–39</sup> and others arguing that it concerns technical engineering issues.<sup>32,36</sup> The most widely used definition remains that of Boardman and Sauser, which is used in this paper alongside a stance that SoS engineering is concerned with more than purely technical engineering issues, including political, economic, social, and acquisition considerations.

### 1.1.3 | System architecture

System architecture can be defined as “the embodiment of concept, the allocation of physical/informational function to the elements of form, and the definition of relationships among the [system] elements

and [between a system and its] surrounding context.”<sup>40</sup> Other views of architecting suggest its role is as a precursor to engineering design and that architecting largely defines the subsequent functionality of the system.<sup>7,8</sup> International standards and architecture frameworks instead suggest a broader remit for systems architecting activity, implying that it is conducted for a number of purposes; including to promote stakeholder understanding, to make design or investment decisions, or to conduct capability analysis.<sup>41–47</sup> Within traditional lifecycle models, it is the concept development stage that corresponds to most systems architecting activity<sup>48</sup> although clearly architectures may be updated and serve a myriad of purposes, throughout a system’s lifecycle. There are several architecture frameworks currently in use: “conventions, principles and practices for the description of architectures established within a specific domain of application.”<sup>45</sup> Architecture frameworks can be used to guide an architect by creating several views, or perspectives, on the Sol for this purpose, such as NATO Architecture Framework (NAF)<sup>49</sup> or Ministry of Defence Architecture Framework (MODAF).<sup>41</sup> This manuscript adopts a view of systems architecting as activity conducted intensively toward the start of an Sol lifecycle in order to support stakeholder understanding, investment decisions, and design decisions that largely determine the subsequent functionality of the Sol. An architecture thus captures the concept, functions, form, and relationships of the Sol in a format that can be used to enable a shared understanding and support decisions about the Sol.

## 1.2 | Context

With working definitions presented, our focus can turn to contextualizing the problem of assessing complex SoS architectures. The case studies analyzed below are used to exemplify the exploration of complex SoS architecture evaluation. To contextualize the problem, consider an organization faced with an Invite To Tender for the upgrade of a system that operates as part of an SoS. The upgraded system, the Sol, may be complex and the SoS of which it is part may also be complex. Several challenges are inherent to such a scenario; whether to bid for delivery of such a system; how to characterize design challenges for the Sol (eg, boundary definitions); how to make good design decisions that will determine the subsequent functionality of the Sol; and, potentially, how to evaluate and select one of several candidate architectures. If an organization utilizes systems thinking to assist in the characterization of such challenges, they are likely to maintain a broad and diverse perspective during the early phase design process.<sup>50–52</sup> This work aims to explore the extent to which a graphical representation of an architecture can be useful in such a situation. The next section introduces the real-world use cases considered in the remainder of the paper.

## 2 | USE CASES

This work makes the use of two real-world enterprise architectures originally created and validated by Thales, and chosen as representative of real-world SoS architectures featuring a diversity of entities and relationships. In both of the use cases described below, the architecture is modeled as a directed, unweighted graph, where

architecture entities are represented as nodes (termed vertices) connected by links (termed edges) representing of interdependencies or relationships.<sup>53,54</sup>

### 2.1 | Search and rescue generic architecture

Our first use case is an SAR NAF-based architecture, developed by Thales in order to inform systems architecture training and help the development of NAF v4.<sup>55</sup> The SAR Use Case can be considered an SoS; the overall ability to rapidly locate and recover distressed vessels in changing and adverse environmental conditions depends on the ability of the diverse constituent systems, acting as autonomous agents, to co-ordinate in order to fulfill a capability demand that none can achieve alone. It is only in bringing together a diverse collection of systems that a timely and effective SAR operation can be conducted over a large geographic area covering both maritime, coastal, and in-land deployments.

A directed graph was constructed using several AVs from the SAR architecture; capabilities, services, and logical nodes (from logical views in NAF, not to be confused with nodes from graph theory) were modeled as vertices with relationships between them modeled as unweighted directed edges. Logical nodes include rotary and fixed wing assets for recovery and search, SAR asset control, and places of safety (eg, hospitals), and the relationships modeled include communication channels, taskings, and command and control. Services include messaging services, recovery services, and situational awareness services along with how the services contribute to each other and depend on logical nodes. The capabilities include the capability to detect, locate, recover, and communicate effectively and relationships include how capabilities depend on each other, contribute to services, and depend on logical nodes.

A table was created listing the ID and type of each architectural entity involved in a set of AVs, and the source and target entities involved in every relationship between these entities; the AVs employed were: Architecture Concept Diagram, C1 Capability Taxonomy, C3 Capability Dependencies, C1-S1 Capability to Service Mapping, S1 Service Taxonomy, L1 Node Types, L2 Logical Scenario, and L3 Node Interactions. It should be noted that the terms modeled are not independent of each other. In NAF, services contribute to capabilities, and capabilities are contributed to by nodes. SAR presents significant challenges to an architect, for example, the heterogeneity of entities (air, land, and maritime assets) and acquisition cycles, which strain interoperability and intervention, compounded by geographical and environmental challenges. The table of relationships, known as an edge table, was imported into Gephi, which is an open source software for exploring and manipulating networks and was used to visualize the resulting network.<sup>56</sup> Analysis of the network was undertaken by importing the same edge table into Python and using the NetworkX package to calculate various network metrics.<sup>57</sup> Edge weights could be added to model the relative importance of relationships and interactions, with values taken from NAF AVs themselves (if explicitly represented there) or from expert opinion, for example, using some estimate of system importance.<sup>58</sup> However, in the absence of data on the

relative significance of the relationships between architectural entities represented by network edges, they were treated here as unweighted.

## 2.2 | Tactical military communications enterprise architecture

Our second use case is a tactical MComms, created in accordance with the MODAF, to enable Thales and the customer to have a shared understanding of the complex environment within which a tactical military communications solution would have to interoperate. The tactical military communications architecture describes the challenge of enabling effective tactical communications between soldiers. The tactical military communications enterprise architecture can be considered to be an SoS; the overall ability for a soldier to be able to communicate effectively (via voice and data, on a network with sufficient security, availability, and integrity) with a range of other actors in adverse environmental conditions (including adversarial electronic counter measures) over a large and contested geographical region depends on the ability of the diverse systems and agents that make up the SoS, which form their own connections in order to fulfill the capability demand that none can achieve alone. The MComms SoS was brought together to enable secure and timely end-to-end communications and information services between deployed soldiers and other actors in the land environment on a geographically distributed battle-field. Again, while several communication systems, services, and software can fulfill parts of this purpose in isolation, it is only in the joining together of several of these diverse elements that overall effective communications can be delivered to soldiers. Further, the joining together of these systems is nontrivial (different technologies, processes, agents, and personnel are challenging to integrate) and requires an additional layer of management to form and sustain the overall SoS.

Several entities from the enterprise architecture were modeled; Systems, Services, Functions, Artifacts (components), Software, and Capability Configurations, which were modeled as vertices. Specifically, from the following AVs, an edge table of relationships was created and interrogated in the same way as for SAR use case; AV-1 Overview and Summary Information, StV-2 Capability Taxonomy, StV-4 Capability Dependencies, SOV-1 Service Taxonomy, and SOV-5 Service Functionality.<sup>56,57</sup> The relationships between these entities were modeled as unweighted directed network edges.

## 3 | THEORY

This section introduces and defines the graph-theoretic topological properties used to explore the SoS architectures. This section aims to make these concepts and terms less opaque for systems engineers who may not be familiar with graph theory or network science concepts.

### 3.1 | Vertex-level topological properties

A graph,  $G$ , is made up of vertices,  $V$ , some of which are connected by edges,  $E$ , where  $G = (V, E)$ . Graphs can either be *directed*, where an edge  $ij$  connects vertex  $i$  to vertex  $j$ , but not vice versa, or *undirected* where

edges are considered to be bi-directional connections. The degree of a vertex is the number of edges incident to that vertex. A graph's average degree characterizes the connectivity of a typical vertex and hence the average connectivity of the graph as a whole. The average shortest path length (or characteristic path length),  $l_G$ , of a graph  $G$  is given below, where  $N = |V|$  is the number of vertices in  $G$  and  $d(i, j)$  is the geodesic distance between nodes  $i$  and  $j$ , that is, the number of edges in the shortest path between vertex  $i$  and vertex  $j$ , assuming  $d(i, i) = 0$  and  $d(i, j) = 0$  if  $j$  cannot be reached from  $i$ :<sup>59</sup>

$$l_G = \frac{1}{N \cdot (N - 1)} \cdot \sum_{i \neq j} d(i, j). \quad (1)$$

The characteristic path length of a graph can be used to characterize efficient resource flow in the graph where a shorter average path length indicates that resources, information, energy, and so on, can flow through the network relatively easily.

The Clustering Coefficient of a graph,  $C$ , quantifies the probability that vertices  $i$  and  $j$  are connected, given that  $i$  and  $j$  are both directly connected to a shared neighbor vertex  $k$ . The Clustering Coefficient can be calculated globally (for the whole graph) as the proportion of connected triples in the graph that form two sides of a connected triangle:

$$C = \frac{3 \times \text{no. triangles}}{\text{no. connected triples}}. \quad (2)$$

There are several approaches to identifying which entities in a graph are the most "important," with different approaches defining the term "important" in different ways. This paper concentrates on a few commonly used approaches. For a more detailed review of topological measures of significance, the interested reader is directed to Newman,<sup>59</sup> Guzman et al.,<sup>60</sup> and Scott.<sup>61</sup>

A simple approach to characterizing the importance of networked entities is to simply consider the entity's degree. Vertex degree could be considered an indication of vertex importance in the sense that highly connected entities have greater influence than those entities that are less well connected. The in-degree of a vertex is defined as the number of incoming connections incident on it, while out-degree is defined as its number of outgoing edges. The In-Degree Centrality of a vertex,  $i$ , is defined as the fraction of the graph's vertices from which edges arrive at  $i$ , whereas the Out-Degree Centrality of  $i$  is defined as the fraction of the graph's vertices to which edges depart from  $i$ , both normalized with respect to the maximum possible degree in a simple graph,  $N - 1$ .

The Closeness Centrality of a vertex represents its average distance to every other node in the network, measured in terms of shortest paths. The Closeness Centrality,  $CC(i)$ , of a vertex  $i$  is the reciprocal of the sum of the shortest path distances from  $i$  to all other  $N - 1$  vertices in the graph (3). A normalized version is used here to enable the comparison between the use cases, which have a different number of vertices, multiplying the Closeness Centrality by the sum of the

minimum possible distances,  $N - 1$ :

$$CC(i) = \frac{N-1}{\sum_j d(i,j)}. \quad (3)$$

Closeness Centrality is thus a measure of importance in terms of which entities are the most central in the architecture, and consequently closest to the other vertices. For a system architect, Closeness Centrality could be used to identify entities in the architecture that, if removed, would have a large impact on the average Closeness Centrality of the architecture, as a proxy for cohesiveness of the architecture. As other authors have noted<sup>62</sup>, there are practical problems with using Closeness Centrality as a measure of vertex importance. Where no path exists between two vertices,  $i$  and  $j$ ,  $d(i,j)$  is undefined. Treating the path length in such a case as zero artificially inflates the centrality of  $i$  and  $j$ . Treating such a path length as effectively infinite prevents the calculation of a defined value for Closeness Centrality for  $i$  and  $j$ . For both of the architectures considered here, the directed nature of network edges and the tree-like structure of (parts of) the networks ensure that paths do not exist between many pairs of vertices. The impact on the Closeness Centrality metric means that care must be taken when using it as a proxy for vertex importance.

An alternative approach, recommended by Boldi and Vigna, but rarely used within the systems community, is to use Harmonic Closeness Centrality, which flips the sum and reciprocal terms from Closeness Centrality, removing the impact of undefined path lengths between vertices. The Harmonic Closeness Centrality,  $H(i)$ , of a vertex  $i$  is the sum of the reciprocal of the shortest path distances,  $d(i,.)$  from  $i$  to all other  $N - 1$  vertices in the graph (where  $1/d(i,j) = 0$  if there is no path from  $i$  to  $j$ ):

$$H(i) = \sum_{j \neq i} \frac{1}{d(j,i)}. \quad (4)$$

A similar approach to Closeness Centrality is to examine importance in terms of the *change* in the sum of distances between vertex pairs, if the vertex in question were removed.<sup>63</sup> The Closeness Vitality of vertex  $i$ ,  $CV(i)$ , is calculated as the difference between the Wiener Index (the sum of the lengths of the shortest paths between all pairs of vertices, in effect the total shortest distance for graph  $G$ ) of the graph with vertex  $i$  removed,  $I_W(G \setminus \{i\})$ , and the Wiener Index of the graph without the vertex removed,  $I_W(G)$ . Important entities can thus be identified as those whose removal would increase the geodesic distance between vertices:

$$I_W(G) = \sum_{i=1}^N \sum_{j=1}^N d(i,j), \quad (5)$$

$$CV(i) = I_W(G) - I_W(G \setminus \{i\}).$$

An alternative approach to considering importance as geometric closeness is to consider importance to be associated with enabling communication between many entities. The Betweenness Centrality,  $BC(i)$ , of a vertex  $i$  is the number of shortest paths in the graph that pass through vertex  $i$  and is given below, where  $N$  is the set of vertices in

graph  $G$ ,  $\sigma(s, t)$  is the number of shortest paths between vertices  $s$  and  $t$ , and  $\sigma(s, t|i)$  is the number of those shortest paths that pass through some vertex  $i$ :

$$BC(i) = \sum_{s,t \in N} \frac{\sigma(s, t|i)}{\sigma(s, t)}. \quad (6)$$

A further approach to characterizing the importance of a vertex,  $i$ , is to consider the well connectedness of its neighbors, which involves considering the well connectedness of the neighbors of these neighbors, and so on. How to calculate this recursive definition of importance? Consider a population of random “walkers” traversing the network. At each step, each walker picks a random edge leaving their current vertex and follows that edge to a new vertex. Over time, the initially randomly distributed walkers will come to be distributed across the network in a way that favors well-connected nodes, that is, those that are adjacent to many well-connected nodes. The Eigenvector Centrality,  $x_i$ , of a vertex  $i$  captures this intuition, and is defined below:

$$x_i = \frac{1}{\lambda} \sum_{j \in G} A_{ij} x_j, \quad (7)$$

where  $A$  is the adjacency matrix of the graph  $G$  (the adjacency matrix is a square matrix representing the graph  $G$ , where  $a_{ij} = 1$  if vertex  $j$  is connected to vertex  $i$  and  $a_{ij} = 0$  otherwise) and  $\lambda \neq 0$  is a constant.

For an architect, the Eigenvector Centrality may be a more suitable measure of importance of an entity in an architecture than simply looking at degree since it highlights influential vertices in terms of their location within the network. The directed acyclic nature of the networks used to represent the two use cases presents challenges in the use of Eigenvector Centrality. Namely, vertices that have no in-coming edges have a null Eigenvector Centrality score, as do vertices whose only in-coming edges are from null scoring vertices. Katz Centrality is a similar measure but that deals with this issue by allocating each vertex an initial positive value of centrality. The Katz Centrality for vertex  $i$  is

$$x_i = \alpha \sum_{j \in G} A_{ij} x_j + \beta, \quad (8)$$

where  $A$  is again the adjacency matrix of the graph  $G$  with eigenvalues  $\lambda$ , with parameter  $\beta$  controlling the initial centrality.

There are several other measures related to Eigenvector Centrality, such as Google's PageRank measure, which identifies important web pages and Hyperlink-Induced Topic Search (HITS) hub and authority score, but these are not discussed in detail here. Instead, an interested reader is directed to Newman.<sup>64</sup>

### 3.2 | Graph-level topological properties

Further topological measures of significance can be explored at the graph level (corresponding to the level of the SoS as a whole). The first is the density,  $D$ , of the directed graph,

$$D = \frac{|E|}{N(N-1)}, \quad (9)$$

where  $|E|$  is the number of edges in the graph and  $N = |V|$  is the number of vertices in the graph.



Density is a simple measure of how densely connected the directed graph is, where a value of  $D = 1$  corresponds to a graph for which every possible connection between vertices is present. Density considerations could be useful in evaluating two competing complex SoS architectures as a higher density solution may have a greater integration challenge, or a greater dependency management challenge, but could, perhaps, enjoy greater resiliency.

Another SoS-level analysis involves considering a directed graph's strongly connected components. A strongly connected component is a set of a graph's vertices where, for every pair of vertices,  $i$  and  $j$ , in the component, there exists a path from  $i$  to  $j$  and a path from  $j$  to  $i$ .<sup>65</sup> In other words, if every vertex in a component is reachable from every other vertex, it is strongly connected. Determining a graph's strongly connected components can reveal a core and periphery structure in a directed graph. This may assist the overall understanding of an SoS, helping identify how subsystems interact with one another. The identification of strongly connected components may also help identify where to apportion responsibility, boundaries, or internal and external dependencies. This approach has the potential to identify a set of nodes as making up a network's core or periphery. It contrasts with node-level analyses that attempt to identify core nodes or peripheral nodes based on their individual properties (eg, some measure of centrality).<sup>66</sup>

A similar idea to the identification of strongly connected components is the identification of communities within a graph. A community may be described as a set of vertices that are more strongly or more frequently connected to each other than to other vertices in the network. There are numerous algorithms and approaches to community detection and the interested reader is directed to Fortunato<sup>67</sup>. For the complex SoS architectures explored here, it could be important for an architect to identify communities in order to partition the SoS into subsystems. The "modularity of the partition,"  $Q$ , has been suggested as a measure of how good a particular division of a network into communities is.<sup>68–70</sup>  $Q$  is defined as the fraction of edges that fall within the identified communities, minus the expected value of the same quantity if the edges were randomly assigned. Thus, a particular identified community structure can be evaluated as significant if there are more within-community edges than would be expected by random chance, corresponding to a nonzero  $Q$ . A value of  $Q = 0$  would correspond to a partitioning with no more significant community structure than would be expected at random. To calculate  $Q$ , first calculate the fraction of the edges in the network that each connects a pair of vertices from the same community,

$$\text{Fraction of Edges} = \frac{\sum_{ij} A_{ij} \delta(c_i, c_j)}{E}, \quad (10)$$

where  $\delta(c_i, c_j)$  is 1 if  $c_i = c_j$  and 0 otherwise and the number of edges in the network is  $E$ . By randomly rewiring the network, but preserving the degrees of vertices, the probability of an edge existing from  $i$  to  $j$  is  $k_i^{\text{in}} k_j^{\text{out}} / E$ , where  $k_i^{\text{in}}$  is the in-degree of  $i$ ,  $k_j^{\text{out}}$  is the out-degree of  $j$ , and  $E$  is again the total number of edges in the network.  $Q$  is then given by

$$Q = \frac{1}{E} \sum_{ij} \left[ A_{ij} - \frac{k_i^{\text{in}} k_j^{\text{out}}}{E} \right] \delta(c_i, c_j). \quad (11)$$

We have presented several vertex-level and graph-level properties that can be used to quantify the importance of an entity or set of entities in a complex SoS architecture. However, each focuses on different interpretations of how importance manifests itself. Clearly, some of these properties will be more suited to some architectures than others, but the point here is that care must be taken to understand what the property means in the context of the system or SoS when it is modeled as a graph. A brief summary of the concepts discussed in this section is shown in Table 1.

## 4 | APPLYING NETWORK SCIENCE TO SYSTEMS ENGINEERING

The notion of applying network science to enterprise architectures is not new, with several authors noting that pertinent aspects of an enterprise can be represented as a complex network of relationships and dependencies across multiple components and thus can be analyzed using network science approaches. A literature survey is provided by Santana et al.<sup>71</sup> Network analysis on enterprise architectures has used centrality measures, such as Betweenness Centrality and Eigenvector Centrality to identify which components are most important to the overall architecture in terms of their relationships.<sup>71–74</sup> There appears to be no agreement in the literature about which aspects of an enterprise should be captured within an enterprise architecture, nor is there agreement on how such an architecture can be modeled from a network perspective—indeed, this is identified as an open research question.<sup>71</sup>

Similarly, graphical models have also been suggested as an approach to assess SoS architectures in a quantitative manner<sup>75</sup>, where architectures could be evaluated in the early phase of their design for their ability to meet mission goals. The work has resulted in a recommended methodology that uses mission goals to represent resources, operations, stakeholders, and policies relevant to an SoS and assigns these mission goals to mission threads (suggested to be represented as sequence diagrams), which can be used to evaluate the ability of different architectures to meet the mission goals. The methodology recommends developing executable architectures that can be tested for their ability to meet the mission goals before a commitment is made to the engineering design of the SoS architecture. The authors stop short of detailing how the graphical models can be constructed for architectures; however, and the focus is instead on a framework to be fleshed out later with further use cases.

Such work often seeks to build on the success of network approaches to representing large-scale design products, engineering products, and engineering projects.<sup>76–80</sup> A range of network science techniques have been used, including inter alia, explorations of edge density, clustering coefficients, and assortativity. Similarly, Design Structure Matrices (DSMs) are frequently used in the product and engineering design community and also provide a means of representing a system as a network, allowing network science concepts, including measures of system complexity or evaluations of system modularity to be applied.<sup>81–83</sup>

**TABLE 1** Summary of network concepts

Network concept	Description
Characteristic Path Length	The length of the shortest path between two nodes, or average of all such lengths for a graph. Characterizes efficient resource flow in a graph where a shorter average path length indicates that resources can flow through the network relatively easily.
Clustering Coefficient	Quantifies the probability that two vertices are connected, given that they are both directly connected to a shared neighbor.
Degree Centrality	The degree centrality for a vertex is the fraction of a network's nodes that it is connected to.
Closeness Centrality	Closeness Centrality of a vertex represents its average distance to every other node in the network, measured in terms of shortest paths. Measure of importance in terms of which entities are the most central in the architecture, and consequently closest to the other vertices.
Harmonic Closeness Centrality	Harmonic Closeness Centrality removes the impact of undefined path lengths between vertices that can hinder the use of Closeness Centrality by taking the sum of the reciprocal of the shortest path distances.
Closeness Vitality	Impact of removing a node on a network's characteristic path length.
Betweenness Centrality	The Betweenness Centrality of a vertex is the proportion of shortest paths in the graph that pass through the given vertex. Important entities are those that enable connectivity between many entities.
Eigenvector Centrality	Eigenvector Centrality suggests influential vertices in terms of their location within the network, where influential vertices are those that are adjacent to many well-connected nodes.
Katz Centrality	Similar to Eigenvector Centrality but by assigning each vertex an initial value of centrality, the issue of vertices that have no in-coming edges have a null Eigenvector Centrality score, as do vertices whose only in-coming edges are from null scoring vertices, is removed.
Density	Proportion of pairs of network nodes that are connected. Could assist in evaluating competing architectures as a higher density solution may have a greater integration challenge, or a greater dependency management challenge, but conversely could enjoy greater resiliency.
Strongly Connected Components	If every vertex in a set of vertices is reachable from every other vertex in the set, the set is a strongly connected component. Strongly connected components may reveal a core and periphery structure in a directed graph.
Community Structure	A community can be described as a set of vertices that are more strongly, or more frequently, connected to each other than to other vertices in the network. There are several algorithms for community detection.

Alternative frameworks exist within systems engineering to model a complex system as a graph and are centered on extensions of DSM and Domain Mapping Matrices (DMM), which are used together to create an Engineering System Matrix (ESM).<sup>13,84</sup> In an ESM, a graphical model of a system is created by modeling system elements (which include system drivers, stakeholders, objectives, functions, objects, and activities) as vertices and the relationships between them as edges. The ESM approach also includes modeling of the temporal domain through the inclusion of which entities and relationships “exist” at various time steps. The ESM approach explores several topological metrics of the resulting graph in an effort to characterize a complex system in order to support decisions, such as where to exert influence in the system to make the most improvements or to determine which parts of the system are more important than others. The author highlights challenges in the collection and the mapping of the data in order to construct a graphical model of the project (eg, relying on qualitative interviews to establish relationships between some entities). Several topological assessments were made from the resulting graph at different times; number of vertices, number of edges, average degree, average path length, and clustering coefficient. What the number of vertices and edges tell an observer about a systems engineering project is not explored. The work uses average shortest path length to suggest a means to select candidate system configurations, considering a system with a shorter average path length to be potentially more desirable.

The ESM has been used to model a malaria surveillance SoS as a graph. The subsequent graph has then been analyzed to attempt

to determine the most influential constituent systems in an effort to aid decisions around investment of effort.<sup>16,85</sup> The process and architecture domains of a healthcare SoS were modeled, using a weighted DSM for the system components, a weighted DMM for the processes, and captured dependencies between the two for an overall DSM. The influence of each constituent system is given by the product of three variables; an environmental risk factor, the sum of weights for the processes that the constituent system contributes to, and an architecture attribute. The architecture attribute could be given by a common centrality measure, such as Betweenness, Closeness, or Eigenvector Centrality taken from the graphical model created using the adjacency matrix of the overall DSM. In the health surveillance SoS, there was little agreement between the different architecture attributes (Closeness, Betweenness, and Eigenvector Centrality) and the authors stopped short of providing guidance over which attributes should be used in different scenarios. The results of the analysis on the surveillance SoS<sup>16,85</sup> suggested that intervention efforts should be centered around a small number of constituent systems as these have a significantly greater influence in the overall SoS than others.

Graph-theoretic approaches have also been used to identify potential bottlenecks in an SoS.<sup>15</sup> In this work, a fictitious, simple manufacturing SoS is modeled graphically, where suppliers, manufacturing plants, testing centers, and customers are modeled as vertices with edges connecting them representing a flow of resource. The author assigns arbitrary weights to the edges, which could represent logistic costs, and determined where bottlenecks exist and what the

maximum performance of the manufacturing SoS could be. The use of such an approach requires that the problem at hand is concerned with throughput and that the complex system or SoS architecture can be adequately modeled as a network that transports some resource from a source to a sink. Although the example used for a complex SoS architecture is a simplified manufacturing network, the justification that such a system is an SoS is not provided, nor is it determined how well a typical SoS can be modeled as a network with resources passed between sources and sinks. The work does highlight that there is potentially a wealth of theorems within graph theory that could assist the architecting of a complex SoS.

This paper starts from a different modeling perspective to the work above. Instead of using a DSM or MDM, or creating a new application-specific abstraction based on system entities, a graphical model is constructed directly from AVs created in accordance with an Enterprise Architecture Framework. Rather than analyzing synthetic architectures, this work uses two real-world architecture products. Our work explores the extent to which the metrics and methods described in the sections above can be usefully exploited in each case and highlights the challenges inherent with taking such an approach. We summarize our findings by suggesting five guiding principles for the effective mobilization of networks concepts for complex SoS architectures.

## 5 | RESULTS

### 5.1 | Vertex-level topological properties results

From the edge tables created for each use case architecture described in Section 2, we calculate a range of network metrics. First, we employ several different topological measures of significance with the aim of identifying which entities in each use case architecture are the most important or influential. The results reveal little agreement between the centrality measures regarding which entities are identified as the most important. The most commonly used measure from the literature surveyed is Closeness Centrality, and Figure 1 shows the lack of significant correlation between Closeness Centrality and other topological measures of significance, demonstrating that different measures identify different competing nodes as most influential. Within both the SAR and MComms architecture, there are several vertices that appear to be important in terms of Closeness Centrality but which score zero or very low by other measures, such as Betweenness Centrality or Eigenvector Centrality. The converse is also true, where for both architectures some of the entities identified as the most important or influential by Eigenvector Centrality or Degree Centrality have a very low or zero Closeness Centrality. These concepts do not provide a consensus on which architectural entities are the most important or influential in an architecture, as importance or influence is treated different by each measure.

To examine the level of agreement between the various concepts, every correlation between the topological measures of significance was calculated for both architectures (Table 2). Although the results show some similarity between specific measures (noting that Betweenness and Load Centrality are different implementations of the

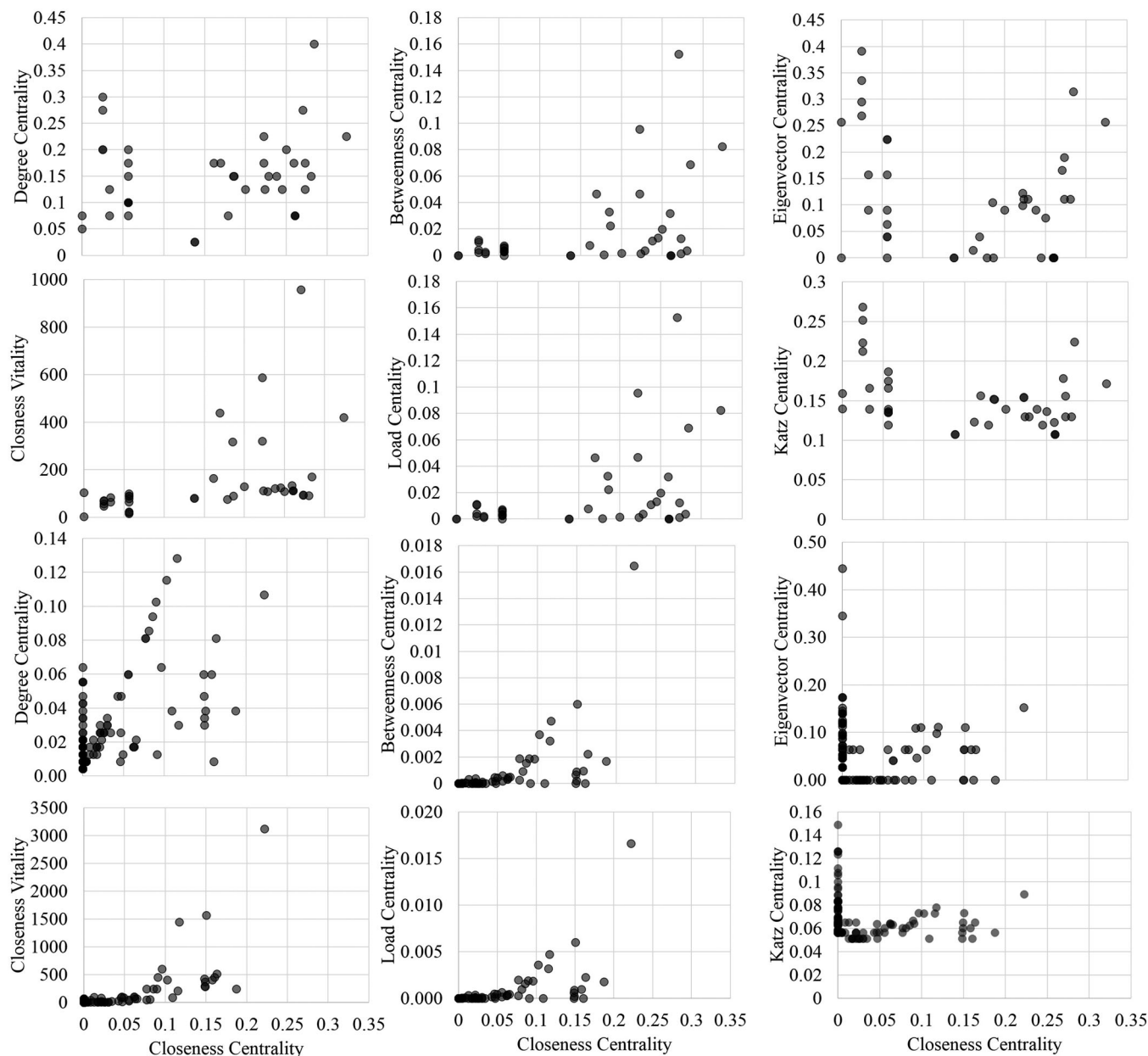
same measure, while Katz Centrality is directly related to Eigenvector Centrality), they also indicate the lack of significant correlations between others for both architectures. The only significant correlation for both architectures is between Harmonic Closeness Centrality and Eigenvector Centrality/Katz Centrality and between Closeness Vitality and Betweenness Centrality. A more detailed survey of correlations between topological measures of significance in published work has been conducted before,<sup>86</sup> but the results found here show less significant correlations, albeit with relatively small networks. None of the topological measures of significance considered here strongly correlate with Closeness Centrality and there is little agreement between topological measures of significance, which seek to determine which entities are the most important. Despite these measures employing similar concepts, such as geometric distance and connectivity, the lack of significant correlations suggests that caution must be exercised in their use to determine which architectural entities are the most important or influential. For example, Closeness Centrality, Harmonic Closeness Centrality, and Closeness Vitality all consider an entity to be important if it is geometrically close to many other entities, yet for both architectures there are no significant correlations between these concepts. Similarly, Betweenness Centrality and Eigenvector Centrality both consider an entity to be important if it contributes to connectivity across a graph, yet again for both architectures there are no significant correlations between these concepts. While both architectures exhibit a significant correlation between Closeness Vitality and Betweenness Centrality, and Harmonic Closeness Centrality and Eigenvector Centrality, the lack of correlations between other similar measures reduces confidence that either of these concepts convincingly identify the most important entities. Overall, trying to determine which architectural entities are the most important for each architecture is inconclusive, largely due to the nuanced assumptions underpinning each particular measure. This point is discussed further in the next section.

### 5.2 | Graph-level topological properties results

At a more aggregate level of consideration, the edge density for each of the two architectures is calculated and is shown in Table 3. It is possible to compare the density values of two competing architectures in order to gain an idea of the likely integration, interface, and dependency management challenges associated with an architecture solution as part of an architecture evaluation process. Here, we see that while both architectures do not exhibit a high density of connections compared with a complete graph, the SAR architecture is nearly an order of magnitude more connected than the MComms architecture, demonstrating greater interconnectivity between architectural entities for SAR.

The two use case architectures are then visualized. Figures 2 and 3 show each architecture as a directed graph, with vertices colored first by entity type (eg, whether the nodes represent capabilities, services, or operational nodes for the SAR use case, or services, functions, systems, etc, in the MComms use case) and then by community membership (discussed below; where each identified community is colored separately). The networks are visualized using Fruchterman-Reingold's Force Directed Layout (an algorithm that





**FIGURE 1** Scatter plots of Closeness Centrality (horizontal axis) against other topological measures of significance for the SAR (top six) and MComms (bottom six) use cases. Note that Load Centrality is a different algorithmic implementation for Betweenness Centrality, hence the similarity in these plots. Different measures identify different nodes as most influential. Different topological measures of significance identify different competing nodes as most influential. Similar variation was found for the following measures but are not shown; Harmonic Closeness Centrality, PageRank, HITS Hub, and HITS Authority

simulates the graph as a physical system, assigning forces to edges and nodes, where every node repels every node, but connected nodes attract each other, calculating the summed forces acting on the nodes and then moving nodes on a 2-D plane accordingly, seeking equilibrium).<sup>56,87</sup> Isolated vertices, or at least those with little connectivity are therefore pushed to the outer perimeter of the network, while vertices with high mutual connectivity are clustered together. Vertices are sized by their degree to indicate the vertices with the highest connectivity. Edges are colored by their destination vertex color. The density differences from Table 3 are apparent in Figures 2 and 3 and while a direct comparison between these two

architectures is not appropriate, one can imagine comparing two competing architectures or architectural configurations within the same use case scenario to compare their complexity (simply in terms of the interconnectedness of each architecture).

The MComms architecture (Figure 2) has several vertices, which are unconnected, several of which represent functions. Interestingly, functional entities are also some of the highest connected vertices, suggesting that some isolated vertices represent functions that may not have been fully considered in the original architecture. The MComms architecture shows which systems interact, which systems provide functionality, which software they use, and how they

**TABLE 2** Table of Pearson's correlation coefficients ( $R^2$ ) between centrality measures for two use cases: SAR (d.f.=39) (below the leading diagonal) and MComms (d.f.=233) (above the leading diagonal)

	Degree Centrality	Closeness Centrality	Harmonic Closeness Centrality	Closeness Vitality	Betweenness Centrality	Load Centrality	Eigenvector Centrality	Katz Centrality
Degree Centrality	–	0.443***	0.139***	0.205***	0.277***	0.274***	0.208***	0.167***
Closeness Centrality	0.034	–	0.000	0.454***	0.377***	0.375***	0.007	0.001
Harmonic Closeness Centrality	0.335***	0.239**	–	0.030**	0.013***	0.013***	0.802***	0.833***
Closeness Vitality	0.145*	0.181**	0.025	–	0.897***	0.897***	0.068***	0.032**
Betweenness Centrality	0.356***	0.183**	0.041	0.863***	–	1.000***	0.044**	0.020*
Load Centrality	0.356***	0.183**	0.041	0.863***	1.000***	–	0.044**	0.019*
Eigenvector Centrality	0.491***	0.066	0.631***	0.002	0.041	0.041	–	0.817***
Katz Centrality	0.607***	0.131*	0.720***	0.006	0.064	0.064	0.832***	–

Note: \* $P < .05$ , \*\* $P < .01$ , and \*\*\* $P < .001$ , otherwise  $R^2$  values are not significant.

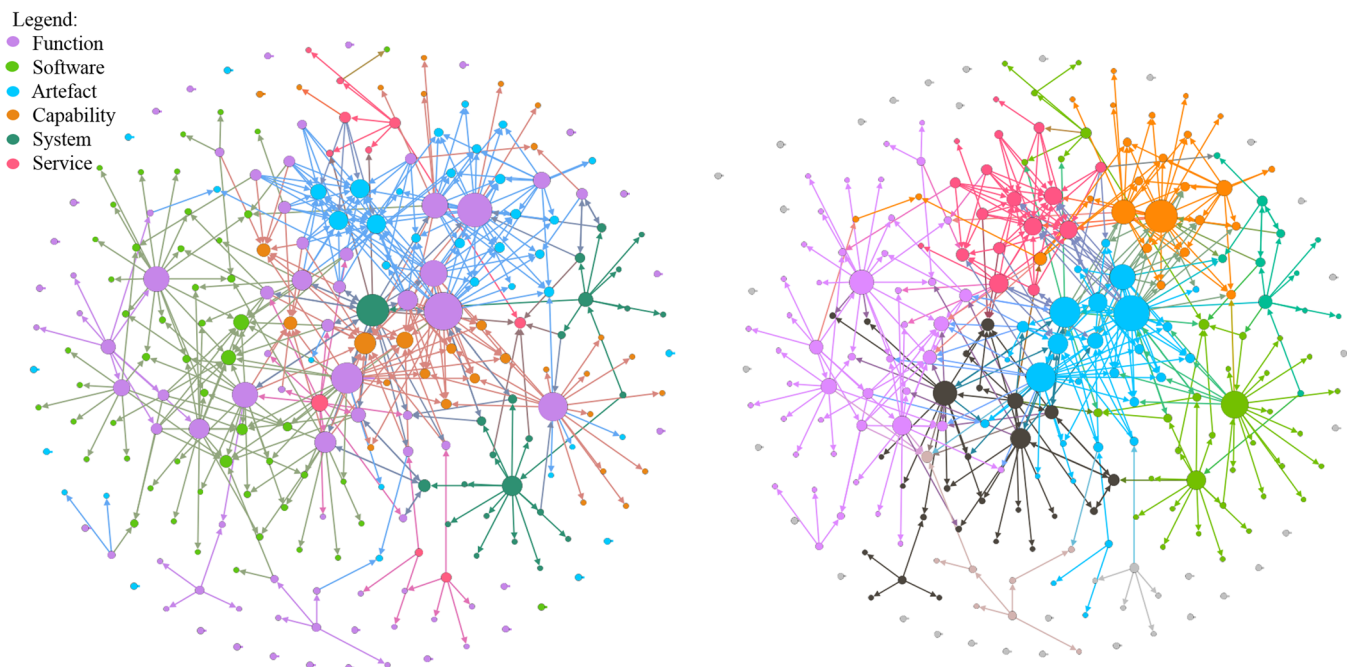
**TABLE 3** Edge density for two use case architecture networks

Property	SAR	MComms
Number of vertices, $N$	41	122
Number of edges, $E$	235	475
Density, $D = E/N(N - 1)$	0.074	0.009

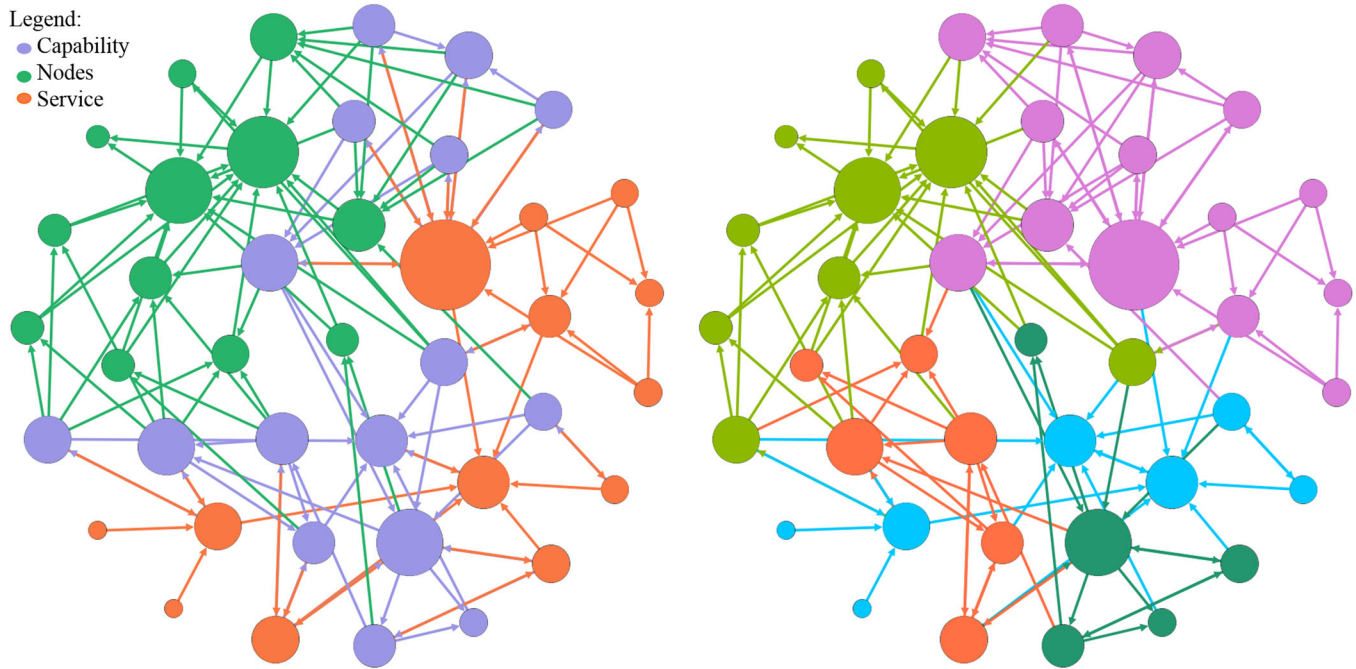
contribute to capability. For the MComms use case, an architect may be specifically interested in the three systems with the largest degree as they appear central to enabling capability through functions and services. As expected, these three systems correspond to a secure deployable broadband voice, data, and video communications system, a secure deployable tactical radio system and more interestingly, a

system for interfacing between the tactical radio system and other deployable systems, suggesting that these three systems are central to the SoS capability. The SAR architecture appears to be largely hierarchical (Figure 3) with entities tending to connect to other entities of the same type. In the SAR use case, an architect may again be specifically interested in the high degree vertices or perhaps where entities are connected to those of a different kind. We discuss challenges with the visual interpretations of networks in the next section.

A community detection algorithm is then used to identify community structure within each architecture, potentially offering an alternative perspective on its structure or modularity. Several algorithms are available to detect communities within a network, where a community is a grouping of vertices that have particularly strong



**FIGURE 2** Network diagrams representing the Military Communications (MComms) use case. A node's size is proportional to its degree. A node's color represents entity type (left, shown in legend) or community membership (right, each color represents a detected community). Networks visualized using Gephi, an open source software for exploring and manipulating networks



**FIGURE 3** Network diagrams representing the Search and Rescue (SAR) use case. A node's size is proportional to its degree. A node's color represents entity type (left, shown in legend) or community membership (right, each color represents a detected community). Networks visualized using Gephi, an open source software for exploring and manipulating networks

connectivity to each other but relatively weak connectivity to vertices outside the group. Here, Blondel et al.'s algorithm<sup>88</sup> was used to determine communities of vertices in each of the two use case networks.

Applying the community detection algorithm to a complex SoS architecture provides an approach to partitioning the architecture into functional units (although it should be noted that the MComms architecture includes 32 isolated vertices, each comprising an individual "community"). The SAR architecture was partitioned into five communities, and the MComms architecture was partitioned into nine nontrivial communities, both with an uneven distribution of node types. A visual interpretation of the MComms architecture suggests some potentially interesting groupings of architectural entities. One such relevant community is the grouping of software and functions (originally distributed among the purple and green vertices on the left hand side of Figure 2) and another is the collection of the high-degree system discussed earlier (the secure deployable broadband voice, data, and video communications system) grouped together with several functions, capabilities, artifacts, and a service (shown in the teal community in Figure 2). This teal community supports the earlier visual interpretation of this system as a potential particular area of focus for this architecture. Interestingly, this system is grouped with a range of entities more operational in nature than technical communications. For the SAR use case, consider the orange community detected (Figure 3), which suggests the Communication Service, Boat Service, Distress Monitoring Capability, Inform Capability, Communications Capability, and SAR Asset Controller Node could be considered as one grouping of related aspects. Logically, this makes sense, as the SAR Asset Controller Node is an extensive communication services consumer and is integral to monitoring and informing others of distress signals.

This appears to make sense, and may be something that is neglected or overlooked in a "divide and conquer" approach to system development. However, as we discuss later, this assessment requires some caution.

Next, we examine the strength of the communities detected within each use case, using  $Q$ , the modularity of the partition. For SAR,  $Q = 0.422$  and for MComms,  $Q = 0.544$ , suggesting that the communities detected within both architectures are more significant than would be expected at random (values over 0.3 are typically taken to be significant in the literature). One could therefore argue that the nontrivial communities detected within each architecture offer an alternative approach to partitioning or grouping entities, which as shown in Figures 2 and 3, does not correspond with the architectural entity type.

The strongly connected components within each architecture are then calculated. A network perspective can enable the identification of the core and periphery of the architecture by examining the strongly connected components of the graph. While this analysis suggests a clear core and periphery structure for the SAR architecture, this structure is not present in the MComms architecture. For the SAR use case, the largest strongly connected component includes 22% of all the vertices, with a second, smaller, strongly connected component accounting for 17% of the total entities. Thus, the SAR architecture could be partitioned into a core of two separate components that together account for over one third of the architectural entities. For the MComms use case, however, there are only three nontrivial strongly connected components, the largest of which contains just 4% of the architectural entities. Combining all three strongly connected components only accounts for 6% of the architectural entities. Note that the reason that the strongly connected components are not larger in the MComms use case is related to the "tree-like" (relatively acyclic)

nature of the network and, consequently, the many instances in which a pair of vertices are only connected in one direction.

## 6 | DISCUSSION

There is potential for the results of graph-theoretic analysis to be misinterpreted due to the nuanced nature of the modeling choices used to create a graph from an architecture, where care must be taken with network concepts. Although it may be tempting to use network measures, such as Closeness Centrality, to identify the seemingly most important or influential entities in an architecture, we have seen the lack of significant correlations between such measures. Consider an SoS represented by a directed graph for which there is no single large connected component. The theory section noted that Eigenvector Centrality can struggle to accurately reflect the importance of various architectural entities if they are in a directed acyclic graph, as with the two use case architectures here. Although in this work it is found that Katz Centrality strongly correlates with Eigenvector Centrality (Table 2), there remains potential to be misled if unsuitable network metrics are used. Further, what are the implications of choosing between two measures of centrality: Closeness Centrality treats a pair of nodes that are not connected by a valid path as equivalent to a pair of nodes connected by a path of length zero; conversely, Harmonic Closeness Centrality addresses this issue by treating a pair of nodes that are not connected by a valid path as being separated by an infinite distance. Choosing between these two measures in an informed way requires understanding the implication of network fragmentation for the functionality of the Sol. Effectively, one measure answers the question: "How central is a node within its own network component?," while the other answers a different question: "To what degree is a node central to the network as a whole?" Where a network is fragmented into different sized components, the measures behave differently.

As a further example, consider Betweenness Centrality, which might be employed to identify important vertices in a network by quantifying their role in allowing influence to flow between different parts of the network. Crucially, Betweenness Centrality assumes that edges are more important in a network when they have more potential to allow novel information to propagate through the network. Determining whether such an assumption holds for (some part of) a complex SoS architecture is not straightforward and the answer may be specific to a particular use case or even part of that use case (eg, in the case where one part of the architecture concerns physical assets organized in a communication network, while another part concerns overall management or administrative functions).

Although visualizing and examining the structure of these complex architectures, in terms of their connectivity, may uncover patterns that help architects to understand how their SoS operates, or how they may be able to leverage interventions into their SoS, it is again an area where care must be taken. Even a simple visual interpretation of networks can be misleading, with different layout algorithms telling different visual stories that can be interpreted in different ways by dif-

ferent stakeholders, a challenge further magnified by the plethora of algorithms available for graph visualization.<sup>89</sup> The analysis used here deals with a *static* view of the system based on an enterprise architecture. In taking a static representation of the Sol, it is difficult to examine if any emergent properties will be present in the Sol, and future work should investigate how a network perspective could inform or predict the presence of this property by moving toward dynamic analysis.

Network approaches are also available to offer alternative partitions of a system architecture using community detection algorithms. However, the modularity of such a partition, to evaluate the presence of this structure, does not evaluate how "correct" or "useful" the suggested communities are. To do so, a qualitative assessment of the communities detected is required. For both architectures, not all identified communities make sense from the perspective of an architect seeking a useful partitioning of functional units. While an identified community within the SAR architecture that includes all of the Command and Control (C2) functionality (the C2 capabilities and the tactical C2 operational node) along with the co-ordination service (the teal community in Figure 3) might seem sensible, an identified community within the MComms architecture that includes several external systems and seemingly unrelated artifacts and capabilities is harder to interpret, requiring considerable further effort. In line with typical networks analyses, the modeling approach used here does not distinguish between different kinds of energy, information, or material flows present in an architecture. Instead, any such flow is represented as an edge between two vertices. As a consequence, the detection of communities or strongly connected components within a network is not necessarily sensitive to variation in either the character of the nodes themselves or the relationships between them. Finally, the suggestion of a "core" and "periphery" structure within the network model is dependent upon the modeling choices used to create the network model from the architecture and also on our definition of a "core" in terms of strongly connected components, which may neglect more contextually significant indications of architecture structure.

That the two use cases considered here are each a complex SoS presents a compounding challenge arising from the diversity of stakeholders that are likely to interact with the architectures. The architectures are likely to be interpreted and perceived in different ways by a diverse stakeholder community, including potential customers. Such diverse stakeholders are unlikely to have the same consideration of what makes an architectural entity important, influential, or a valid member of some "community" or partition. Beyond the conceptual problems outlined above, a more practical challenge involves acquiring and interpreting sufficient information to build an effective network model, especially at the start of a system lifecycle where confidence in such information may be limited. Determining what features of a complex SoS architecture constitute a vertex or an edge and what determines if a vertex or edge is present requires careful consideration and may be time intensive and context dependent.

The next section proposes further work intended to progress the ability of system architects to make the use of network approaches in the face of the issues described here.



## 7 | FURTHER WORK

Edge weights and further vertex properties could be added in order to model the differences between different types of network entities and between different types of relationships between these entities or to distinguish the differing importance of interactions or entities within the architecture. If edge weights were included in the modeling, a network perspective could enable the exploration of resource flow challenges through the architecture, whether material, information, or energy, by including resource constraints as edge weights. Such an analysis may assist in identifying bottlenecks or entities that provide redundancy. Analysis of this kind may support the evaluation of architecture alternatives or different architecture configurations.

One can imagine for the SAR use case evaluating alternative architecture configurations by considering “what if” scenarios for different communication equipment deployments on distressed vessels and rescue vessels, using edge weights to represent delay, cost, or bandwidth constraints between equipment. In this way, an architect can explore if some architecture configurations have bottlenecks, not just in the communications network, but in the overall system, by considering information flow from the distress vessel to the co-ordination service operator in accordance with a relevant mission thread. However, correctly modeling the information within an architecture as edge weights underpins the utility of such an analysis, and was thus not pursued in this work that sought to bring to light the challenges in taking a network perspective on an architecture created using an architecture framework.

Similarly, for the MComms use case, one can imagine supporting a trade off analysis again examining bottlenecks in fulfilling capability requirements with current systems by including edge weights representing resource constraints, such as bandwidth and security classification between different software, services, and systems, in order to help the customer and potential suppliers understand where intervention efforts may be most useful. Edge weights may also enable more confidence to be placed in assessments of which entities in an architecture are most important, by including a greater level of fidelity in assessment of “closeness.”

Further work could explore which views from architecture frameworks are more amenable to network analysis and thereby provide a methodology for more effective modeling of system architectures as graphical models. Knowing which AVs are more amenable to network analysis would also help identify which network metrics and concepts are most suitable for identifying where interventions should be focused in an SoS. Several AVs within DoDAF (which has been widely utilized by practitioners and academia) appear amenable to network analysis, such as the capability, services, and operational viewpoints.<sup>42</sup> A further study could therefore specifically explore the extent to which a network perspective on these AVs can provide meaningful insights into the entire architecture, or at least how the analysis can be usefully bound to those AVs.

Network perspectives suggest examining whether a “core” and “periphery” structure exists and further work should seek to identify exactly what constitutes a “core” of an architecture in a real SoS. A

network perspective may also suggest asking questions that address notions of dynamic complexity, such as: “what is the effect of the removal of some of these entities, whether as a single change or as some kind of dynamic cascade effect?” It should also be noted, however, that if an architecture team wanted to model their architecture as a network and examine the effect of vertex removal or of some change cascading through the network, they would need to conduct some validation activity in order to be confident that the model captures the important characteristics and dynamics of their architecture. One pressing question concerns how such validation might be carried out at early design stage. Given that the literature defining the SoS concept makes considerable use of the notion of emergent SoS-level behavior or functionality, further work should also examine whether emergent properties of an SoS can be characterized or informed by taking a network perspective on complex SoS architectures.

Graph-theoretic approaches to SoS architectures can explore how the architectures, and their subsequent graphical models, evolve over time to explore dynamics of SoS evolution. However, this requires appropriate supporting data, for example, changes in connectivity and performance over time, which may be difficult to obtain. There are a range of tools available from the study of complex networks, which may provide further insights into complex SoS architectures, such as considerations of the homophily of the architectural elements, that is, the tendency for similar architectural elements to be connected, which may supply insights into design strategies for interfaces within a complex SoS. The application of a network perspective on other aspects of complex systems engineering, such as requirements analysis, could also be explored.

The next section summarizes the work in this paper into guidelines for the effective mobilization of network concepts for the evaluation of complex SoS architectures created in accordance with an Architecture Framework.

## 8 | CANDIDATE GUIDING PRINCIPLES FOR THE APPLICATION OF NETWORK CONCEPTS TO COMPLEX SOS ARCHITECTURE EVALUATION

1. **Partition architectures to manage heterogeneity.** The interpretation of network concepts and measures is considerably less conceptually challenging if the aspects of the architecture that are modeled are more homogenous. Graph-theoretic models can struggle to represent adequately the heterogeneity of entities and relationships present in a complex SoS. While it may be tempting to improve the fidelity of such models, perhaps by including further node properties, edge weights, or more sophisticated network types, there is a compounding challenge in establishing confidence in the data underpinning these features, given the early lifecycle stages architecting activity tends to focus on. The more sophisticated the network representation of an architecture, the more challenging meaningful interpretation of network properties



becomes, especially for more complicated aspects, such as assortativity and examining dynamic processes on the network. Instead, a more suitable approach is to consider partitioning an architecture into more homogenous AVs where network concepts may be more readily mobilized, noting that care should be taken to consider the diversity in the character of system entities (different types of network nodes) and the relationships between them (different types of network edges). However, this divide and conquer approach leaves a challenge of reintegrating the analyses of these separated components of an SoS. While it is beyond the scope of this work to recommend particular partitioning approaches, one useful contribution of a network perspective on architecture evaluation is to bring this challenge to the fore.

2. **Tension multiple metrics against each other.** As no single, widely accepted and applicable, importance or complexity metric is likely to be established for the kind of complex SoS architectures, we are interested in, a sensible approach is to use several such measures, tested and corroborated against each other. System architects should be cognisant of what each measure foregrounds and use this understanding to support their evaluations of which entities in an architecture are more important or how one architecture differs from another. For example, if resources (energy, material, and information) can be assumed to always travel via the shortest paths in an architecture, then Closeness Centrality can identify those entities that are more central in the architecture. However, even in such a case, comparison with other centrality measures will contextualize and thereby strengthen this analysis, and architects should be cognisant of the implicit modeling assumptions that underpin these metrics, so as to avoid potentially being misled.
3. **Combine quantitative and qualitative assessment.** Further to principle two, any system architect hoping to utilize a “network perspective” should temper the results of numerical analysis with a qualitative assessment. Simply because a set of enterprise architecture entities have strong connectivity between them, even taking into account inter and intraset connectivity does not guarantee that it makes logical sense to treat them as one distinct architectural “module” or “unit.” The same is true of exploring if a “core” and “periphery” structure dominates, or if one architecture is more “complex” than another using one particular measure of complexity. Thus, architects may well make use of network science analyses to examine the structure of their architecture, but they should temper these results with their own qualitative evaluation.
4. **Maintain awareness of modeling depth.** Any graph-theoretic approach to architecture evaluation should explicitly reflect on the modeling assumptions that underpin their findings. While this task is conceptually challenging, it is a necessary one to avoid misunderstanding or placing overconfidence in some numerical analysis. Simply put, the network models are not the complex SoS architectures. Further, the architectures are not the real-life SoS in question. At each level, the artifacts and models are representations, abstractions, of the higher level. Care must be taken to not lose

sight of this and test if the assumptions that underpin the network led enquiry are still valid for the architecture and for the SoS itself.

5. **Reflect on the questions a network perspective enables.** Finally, the methodology used here cannot be seen as an alternative to more traditional approaches to architecture evaluation, but instead should be understood as offering a complementary perspective on architecture evaluation. It is instead recommended as a complementary perspective, in line with a broader systems thinking approach, that enables architects to ask questions of their architecture that may otherwise go unanswered. Such questions include; “what makes an entity important in this architecture, what role does reciprocity, assortativity, or community structure play in this architecture, what makes this architecture more robust or resilient than another?,” or “where is there more merit in considering this diverse and rich architecture at a greater level of abstraction or a more fine-grained level of detail?,” or even “what is it about this architecture that led us to towards the toolbox of network science?” As graph-theoretic approaches continue to mature, it will remain important to critically assess the nature of the questions that they are capable of answering.

## 9 | CONCLUSIONS

In this paper, we have taken a networks perspective on SoS architecture analysis in order to explore the potential for this approach to inform architecture design and selection, and also to highlight a series of challenges that need to be addressed if the approach is to be useful for system architects.

While several network measures may have relevance to architectural design and evaluation decisions, we argue that issues with interpreting network-theoretic properties pose challenges that interfere with their utility. Two real-world SoS architectures (a search and rescue architecture, and a military communications architecture) were characterized as networks, where a set of architectural entities were modeled as vertices, and their dependencies and communications links were modeled as edges. The most significant challenges in taking a network perspective on real-world architectures are located at this stage; representing a complex architecture as a network, deciding what should be included and excluded from the network model, and finding meaning in network concepts, measures, and metrics.

A range of simple network metrics were applied to the two use case networks, real-world enterprise architectures, in an attempt to identify key entities and assess gross structural organization. Given the diversity of, for example, measures of vertex significance and community detection, it is challenging to either select particular measures (which may supply partial or idiosyncratic results) or, alternatively, apply a wide range of measures (which will typically disagree in ways that require careful analysis), without first considering what such measures correspond to in the real-world architecture.

Graph-theoretic approaches provide an alternative approach to exploring and understanding a complex SoS, by representing it as a

graph and examining this graph's structure, but critically this structure is the *modeled* graph-theoretic structure. It is not the architecture itself, only an idealized representation of it. In creating a graphical model, there may be aspects of the SoS that are not modeled and thus making assertions concerning the structure of the graphical model may not be equivalent to making assertions concerning the entire SoS. Analyzing SoS networks in terms of components or communities, for instance, may neglect potentially important factors, such as geographical, functional, or organizational separation between components if these are not explicitly represented in the network structure.

Taking a network perspective on SoS architectures could help inform which entities in an architecture are most important to one another, or assist architecture evaluation by helping determine whether one candidate architecture is more robust, efficient, or manageable than another. In seeking these insights, however, we argue that the tools from network science cannot straightforwardly be applied without developing a more sophisticated understanding of how they map onto the diversity, richness, and context sensitivity characteristic of complex SoS architectures. While developing a set of conceptual tools for the analysis of complex SoS architectures remains an open research challenge, by developing guiding principles for the effective mobilization of network concepts to architecture evaluation, system architects can better take the advantage of these tools. Further, in bringing to light the challenges system architects are faced in taking a network perspective on their architectures, they are better able to avoid being misled by some numerical analysis that lacks contextual awareness.

Instead of using network analysis as an off-the-shelf tool for empirical analysis and decision support in the design of SoS architectures, we advocate a more contextual approach in which network analysis is employed as one of many perspectives that can be taken on an architecture, one that may reveal insights that would otherwise be overlooked, but also one that requires cross-validation against more qualitative or systems theoretic perspectives that may do a better job of capturing the rich and heterogeneous properties of SoS architectures.

## ACKNOWLEDGMENTS

The authors would like to thank Dave Harvey and Jean-Luc Garnier at Thales who provided valuable insights during this work. The authors also wish to thank the anonymous reviewers for their detailed and insightful feedback and insightful contributions.

## ORCID

Matthew W. Potts  <https://orcid.org/0000-0002-4266-0862>

## REFERENCES

- Hartmann R, Belhoff B, Oster C, et al. *A World in Motion, Systems Engineering Vision 2025*. International Council on Systems Engineering (INCOSE); 2014.
- Crawley E, De Weck O, Magee C, et al. *The Influence of Architecture in Engineering Systems*. Massachusetts Institute of Technology. *Engineering Systems Division*; 2004.
- Urken AB, Nimz AB, Schuck TM. Designing evolvable systems in a framework of robust, resilient and sustainable engineering analysis. *Adv Eng Inform*. 2012;26(3):553-562.
- Muller G. Process and organization. In: Li-Ming L, ed. *Systems Architecting: A Business Perspective*. CRC Press; 2011:1-29.
- Muller G. Role and task of the systems architect. In: Li-Ming L, ed. *Systems Architecting: A Business Perspective*. CRC Press; 2011:31-49.
- Muller G. Systems architecting: a business perspective. In: *INCOSE International Symposium*. Vol. 21. Wiley Online Library; 2011:1845-2142.
- Rechtin E. *Systems Architecting: Creating and Building Complex Systems*. Vol. 199. Englewood Cliffs, NJ: Prentice Hall; 1991.
- Rechtin E. The art of systems architecting. *IEEE Spect*. 1992;29(10):66-69.
- Sillitto H. *Architecting Systems: Concepts, Principles and Practice*. College Publications; 2014.
- ISO/IEC/IEEE 15288. *ISO/IEC/IEEE International Standard - Systems and Software Engineering - System Life Cycle Processes*. ISO/IEC/IEEE 15288. 1st ed. 2015; 1-118.
- Bullock S, Cliff D. *Complexity and Emergent Behaviour in ICT Systems*. Technical Report HP-2004-187. Hewlett-Packard Labs; 2004.
- Aboutaleb H, Monsuez B. Measuring complexity of system/software architecture using Higraph-based model. In: *Proceedings of the International Multiconference of Engineers and Computer Scientists*. Vol. 1. Newswood Limited; 2017:92-96.
- Bartolomei JE, Hastings DE, de Neufville R, Rhodes DH. Engineering Systems Multiple-Domain Matrix: an organizing framework for modeling large-scale complex systems. *Syst Eng*. 2012;15(1):41-61.
- Davison P, Cameron B, Crawley EF. Technology portfolio planning by weighted graph analysis of system architectures. *Syst Eng*. 2015;18(1):45-58.
- Harrison WK. The role of graph theory in system of systems engineering. *IEEE Access*. 2016;4:1716-1742.
- Okami S, Kohtake N. Modeling and analysis of health-information system of systems for managing transitional complexity using engineering systems multiple-domain matrix. In: *Annual IEEE International Systems Conference (SysCon)*. IEEE Press; 2017:1-8.
- De Weck OL, Ross AM, Rhodes DH. *Investigating Relationships and Semantic Sets amongst System Lifecycle Properties (Ilities)*. Massachusetts Institute of Technology. *Engineering Systems Division*; 2012.
- DeRosa JK, Grisogono AM, Ryan AJ, Norman DO. A research agenda for the engineering of complex systems. In: *2nd Annual IEEE Systems Conference*. IEEE Press; 2008:1-8.
- Ladyman J, Lambert J, Wiesner K. What is a complex system? *Eur J Philos Sci*. 2013;3(1):33-67.
- Lloyd S. Measures of complexity: a nonexhaustive list. *IEEE Control Syst Magazine*. 2001;21(4):7-8.
- Sheard SA. 5.2.1 Systems engineering complexity in context. In: *INCOSE International Symposium*. Vol. 23. Wiley Online Library; 2013:1145-1158.
- Sheard SA, Mostashari A. 7.3.1 A complexity typology for systems engineering. In: *INCOSE International Symposium*. Vol. 20. Wiley Online Library; 2010:933-945.
- Sheard S, Cook S, Honour E, et al. A complexity primer for systems engineers. *International Council on Systems Engineering (INCOSE)*. 2015.
- Bar-Yam Y. *Dynamics of Complex Systems*. Vol. 213. Reading, MA: Addison-Wesley; 1997.
- Fischi J, Nilchiani R, Wade J. Dynamic complexity measures for use in complexity-based system design. *IEEE Syst J*. 2017;11(4):2018-2027.
- Bar-Yam Y. Unifying principles in complex systems. In: Bainbridge WS, ed. *Converging Technologies for Improving Human Performance: Nanotechnology, Biotechnology, Information Technology and Cognitive Science*. The Netherlands: Springer; 2003:380-410.
- Mitchell M. *Complexity: A Guided Tour*. Oxford University Press; 2009.

28. Waldrop MM. *Complexity: The Emerging Science at the Edge of Order and Chaos*. Simon and Schuster; 1993.
29. Dahmann J. The state of systems of systems engineering knowledge sources. In: *10th System of Systems Engineering Conference (SoSE)*. IEEE Press; 2015:475-479.
30. Maier MW. Architecting principles for system-of-system. In: *INCOSE International Symposium*. Vol. 6. Wiley Online Library; 1998: 565-573.
31. Boardman J, Sauser B. System of systems—the meaning of of. In: *IEEE/SMC International Conference on System of Systems Engineering*. IEEE Press; 2006:118-126.
32. Gorod A, Sauser B, Boardman J. System-of-systems engineering management: a review of modern history and a path forward. *IEEE Syst J*. 2008;2(4):484-499.
33. ISO/IEC/IEEE P21839. *ISO/IEC/IEEE Draft International Standard - Systems and Software Engineering - Systems of Systems Considerations in Engineering of Systems*. ISO/IEC/IEEE P21839. 2017; 1-43.
34. Jamshidi M. System of systems engineering—new challenges for the 21st century. *IEEE Aerospace Electron Syst Magazine*. 2008;23(5):4-19.
35. Abbott R. Open at the top; open at the bottom; and continually (but slowly) evolving. In: *IEEE/SMC International Conference on System of Systems Engineering*. IEEE Press; 2006:41-46.
36. Cocks D. 3.3.2 How should we use the term 'system of systems and why should we care. In: *INCOSE International Symposium*. Vol. 16. Wiley Online Library; 2006:427-438.
37. Dahmann J. *Systems of Systems Characterization and Types*. NATO Science and Technology Organization. NATO Science and Technology Organization; 2015:1-14.
38. Dahmann J, Baldwin K. Implications of systems of systems on system design and engineering. In: *6th International Conference on System of Systems Engineering (SoSE)*. IEEE Press; 2011:131-136.
39. Luzeaux D, Whippler JL. *Complex Systems and Systems of Systems Engineering*. John Wiley & Sons; 2013.
40. Crawley E, Cameron B, Selva D. *System Architecture: Strategy and Product Development for Complex Systems*. Prentice Hall Press; 2015.
41. Biggs B. *Ministry of Defence Architectural Framework (MODAF)*. Institution of Engineering and Technology; 2005:43-82.
42. Lee S. DoDAF V2.0 Overview. Washington, DC: Department of Defense, Chief Information Officer; 2010.
43. Garnier JL, Bischoff L, André M, et al. Architecture frameworks—a standard to unify terms, concepts, life-cycles and principles. In: *Proceedings of NATO Information Systems Technology Panel Symposium on Architecture Definition and Evaluation (STO-MP-IST-115)* NATO. NATO Science and Technology Organization; 2013:1-18.
44. Zachman JA. A framework for information systems architecture. *IBM Syst J*. 1987;26(3):276-292.
45. ISO/IEC/IEEE 42010. *ISO/IEC/IEEE Systems and Software Engineering - Architecture Description*. ISO/IEC/IEEE 42010:2011(E) (Revision of ISO/IEC 42010:2007 and IEEE Std 1471-2000). 2011:1-46.
46. ISO/IEC/IEEE DIS P42020. *ISO/IEC/IEEE Draft Standard for Systems and Software Engineering - Architecture Processes*. ISO/IEC/IEEE DIS P42020:201x(E). 2017:1-128.
47. ISO/IEC/IEEE FDIS P42030/D2. *IEEE/ISO/IEC Draft Standard for Systems and Software Engineering - Architecture Evaluation*. ISO/IEC/IEEE FDIS P42030/D2. 2018:1-81.
48. Kossiakoff A, Sweet WN, Seymour SJ, Biemer SM. *Systems Engineering Principles and Practice*. Vol. 83. John Wiley & Sons; 2011.
49. Architecture Capability Team, Consultation, Command & Control Board. *NATO ARCHITECTURE FRAMEWORK Version 4*. North Atlantic Treaty Organisation (NATO); 2018.
50. Boardman J, Sauser B. *Systems Thinking: Coping with 21st Century Problems*. CRC Press; 2008.
51. Buede DM, Miller WD. *The Engineering Design of Systems: Models and Methods*. John Wiley & Sons; 2016.
52. Checkland P, Scholes J. *Soft Systems Methodology: A 30-Year Retrospective*. Chichester: John Wiley; 1999.
53. Potts M, Sartor P, Johnson A, Bullock S. Hidden structures: using graph theory to explore complex system of systems architectures. In: Chapoutot A, Krob D, Roussel A, Stephan F, eds. *Complex Systems Design & Management*. CESAM Community; 2017:117-131.
54. Potts M, Sartor P, Johnson A, Bullock S. Through a glass, darkly? Taking a network perspective on system-of-systems architectures. In: Bonjour E, Krob D, Palladino L, Stephan F, eds. *Complex Systems Design & Management*. Cham: Springer International Publishing; 2019:121-132.
55. Libert P, Garnier J-L. NAFv4 Chapter 2 Example. Unpublished work. Thales Internal; 2017:1-103.
56. Bastian M, Heymann S, Jacomy M. Gephi: an open source software for exploring and manipulating networks. In: *3rd International AAAI Conference on Weblogs and Social Media*. 2009.
57. Hagberg A, Swart P, Chult SD. *Exploring Network Structure, Dynamics, and Function Using NetworkX*. Los Alamos, NM: Los Alamos National Lab.(LANL); 2008.
58. Uday P, Marais KB. Resilience-based system importance measures for system-of-systems. *Procedia Comp Sci*. 2014;28:257-264.
59. Newman ME. The structure and function of complex networks. *SIAM Rev*. 2003;45(2):167-256.
60. Guzman JD, Deckro RF, Robbins MJ, Morris JF, Ballester NA. An analytical comparison of social network measures. *IEEE Trans Comp Soc Syst*. 2014;1(1):35-45.
61. Scott J. *Social Network Analysis*. Sage; 2017.
62. Boldi P, Vigna S. Axioms for centrality. *Internet Math*. 2014;10(3-4):222-262.
63. Brandes U, Erlebach T. Fundamentals. In: Brandes U, Erlebach T, eds. *Network Analysis: Methodological Foundations*. Springer; 2005: 7-15.
64. Newman ME. *The mathematics of networks*. New Palgrave Encyclop Econ. 2008;2(2008):1-12.
65. Diestel R. *Graph Theory: Springer Graduate Text GTM 173*. Springer Graduate Texts in Mathematics. New York: Springer; 2012.
66. MacCormack A. The architecture of complex systems: do “core-periphery” structures dominate? In: *Academy of Management Proceedings*. Vol. 2010. Academy of Management; 1-6.
67. Fortunato S. Community detection in graphs. *Phys Rep*. 2010;486(3-5):75-174.
68. Newman ME, Girvan M. Finding and evaluating community structure in networks. *Phys Rev E*. 2004;69(2):026113.
69. Newman ME. Analysis of weighted networks. *Phys Rev E*. 2004;70(5):056131.
70. Leicht EA, Newman ME. Community structure in directed networks. *Phys Rev Lett*. 2008;100(11):118703.
71. Santana A, Fischbach K, De Moura H. Enterprise architecture analysis and network thinking: a literature review. In: *Proceedings of the 49th Hawaii International Conference on System Sciences (HICSS)*. IEEE Press; 2016:4566-4575.
72. Dreyfus D, Iyer B. Enterprise architecture: a social network perspective. In: *Proceedings of the 39th Annual Hawaii International Conference on System Sciences (HICSS)*. Vol. 8; 2006:2875-2885.
73. Iyer B, Dreyfus D, Gyllstrom P. A Network-Based View of Enterprise Architecture. *Handbook of Enterprise Systems Architecture in Practice*. IGI Global; 2007:306-319.
74. Santana A, Souza A, Simon D, Fischbach K, De Moura H. Network science applied to enterprise architecture analysis: towards the foundational concepts. In: *IEEE 21st International Enterprise Distributed Object Computing Conference (EDOC)*. IEEE Press; 2017:10-19.
75. Marvin JW, Garrett Jr RK. Quantitative SoS architecture modeling. *Procedia Comp Sci*. 2014;36:41-48.
76. Braha D. The complexity of design networks: structure and dynamics. In: Cash P, Stanković T, Štorga M, eds. *Experimental Design Research*. Springer; 2016:129-151.

77. Braha D, Bar-Yam Y. Information flow structure in large-scale product development organizational networks. *J Inform Technol.* 2004;19(4):244-253.
78. Braha D, Bar-Yam Y. Topology of large-scale engineering problem-solving networks. *Phys Rev E.* 2004;69(1):016113.
79. Braha D, Reich Y. Topological structures for modeling engineering design processes. *Res Eng Des.* 2003;14(4):185-199.
80. Braha D, Bar-Yam Y. The structure and dynamics of complex product design. In: Braha D, Minai Ali A, Bar-Yam Y, eds. *Complex Engineered Systems: Science Meets Technology.* Berlin, Heidelberg: Springer Berlin Heidelberg; 2006:40-71.
81. Tamaskar S, Neema K, De Laurentis D. Framework for measuring complexity of aerospace systems. *Res Eng Des.* 2014;25(2):125-137.
82. Jung S, Simpson TW. New modularity indices for modularity assessment and clustering of product architecture. *J Eng Des.* 2017;28(1):1-22.
83. Hölttä-Otto K, Chiriac NA, Lysy D, Suk Suh E. Comparative analysis of coupling modularity metrics. *J Eng Des.* 2012;23(10-11):790-806.
84. Bartolomei JE. *Qualitative Knowledge Construction for Engineering Systems: Extending the Design Structure Matrix Methodology in Scope and Procedure.* Air Force Institute of Technology Wright-Patterson AFB OH School of Engineering; 2007.
85. Okami S, Kohtake N. Transitional complexity of health information system of systems: managing by the engineering systems multiple-domain modeling approach. *IEEE Syst J.* 2017;13(1):952-963.
86. Valente TW, Coronges K, Lakon C, Costenbader E. How correlated are network centrality measures? *Connections (Toronto, Ont).* 2008;28(1):16-26.
87. Fruchterman TM, Reingold EM. Graph drawing by force-directed placement. *Software: Pract Exper.* 1991;21(11):1129-1164.
88. Blondel VD, Guillaume JL, Lambiotte R, Lefebvre E. Fast unfolding of communities in large networks. *J Stat Mech Theory Exp.* 2008;2008(10):P10008.
89. Hu Y. Efficient, high-quality force-directed graph drawing. *Math J.* 2005;10(1):37-71.

## AUTHOR BIOGRAPHIES



MATTHEW POTTS is a PhD Research Student at the University of Bristol, UK, within the Aerospace Engineering Department. He received the BEng degree from the University of Southampton, UK in Electromechanical Engineering (2008-2011). He served in various roles in the Royal Air Force (2011-2016) as an Engineering Officer and has worked as an independent consultant. Mr Potts is a registered Incorporated Engineer with the Institute of Engineering and Technology.

PIA SARTOR is senior lecturer in the Department of Aerospace Engineering, University of Bristol, UK. She received a BEng (Hons) degree in Aerospace Engineering from Ryerson University, Toronto, Canada (2006) and a PhD degree in Mechanical Engineering from the University of Sheffield, UK (2011). She worked in industry for 6 years, including as a Senior Systems Engineer at Safran Landing Systems. Dr Sartor is a registered Chartered Engineering with the Institute of Mechanical Engineers.

ANGUS JOHNSON is the Thales Head of Systems Research and the Industry lead on the Thales-Bristol Partnership in Hybrid Autonomous Systems Engineering. He has a background in radar spectrum sensing and multifunction RF systems engineering.



SETH BULLOCK is Toshiba Chair in Data Science and Simulation in the Department of Computer Science at the University of Bristol, UK. He holds a first degree in cognitive science (1993) and a doctorate in evolutionary simulation modeling (1997) from Sussex University, UK. His research expertise lies in complex systems simulation modeling. He has won over £28m in research and infrastructure funding in this area and has undertaken consultancy for the UK Government on complexity in ICT (2004) and financial systems (2012). He publishes in international peer-reviewed journals spanning health, social science, economics, biology, architecture, engineering, environmental science, computing, and physics. He was twice elected to the Board of Directors of the International Society for Artificial Life and serves on the editorial boards of the MIT Press journals *Adaptive Behavior*, *Artificial Life*, and *Frontiers in Robotics & AI*. He has given invited keynote lectures in London, Paris, Athens, Melbourne, Madrid, Granada, and Tokyo.

**How to cite this article:** Potts MW, Sartor P, Johnson A, Bullock S. A network perspective on assessing system architectures: Foundations and challenges. *Systems Engineering.* 2019;22:485-501. <https://doi.org/10.1002/sys.21519>