Regular Paper

System of Systems Integration: Key Considerations and Challenges

Azad M. Madni[*,1] **and Michael Sievers**[†,2]

[1]*Daniel J. Epstein Department of Industrial and Systems Engineering, Viterbi School of Engineering, University of Southern California, Los Angeles, CA 90089*
[2]*Jet Propulsion Laboratory, California Institute of Technology, Pasadena, CA 91109*

Received 22 December 2012; Revised 9 February 2013; Accepted 9 February 2013, after one or more revisions
Published online 1 July 2013 in Wiley Online Library (wileyonlinelibrary.com).
DOI 10.1002/sys.21272

ABSTRACT

As systems are called on to participate on demand within system-of-systems (SoS), system-of-systems integration (SoSI) has become a key concern. This capability is especially important in defense and aerospace where systems are increasingly required to interoperate on demand to satisfy mission requirements. SoSI is also becoming increasingly important in healthcare and energy domains. SoSI involves interfacing and enabling the interactions of component systems to create the needed SoS capability to accomplish mission or business goals. SoSI, which is part of the overall SoS development life cycle, increases in complexity when there are legacy systems that need to be integrated, and when humans are tasked to perform in various capacities within the SoS. An added layer of complexity is introduced when the SoS has to exhibit certain quality attributes such as adaptability and resilience in the face of contingencies and disruptions in the operational environment. This paper addresses key considerations and challenges in SoSI. © 2013 Wiley Periodicals, Inc. Syst Eng 17: 330–347, 2014

Key words: system of systems; system of systems integration; human-systems integration; interoperability; integration ontology

1. INTRODUCTION

A *system of systems* (SoS) is a collection of systems that were originally designed as stand-alone systems for specific and different purposes but that have been brought together within the SoS umbrella to create a new capability needed for a particular mission [Mayk and Madni, 2006; Manthorpe, 1996]. An SoS may also be a system designed to accommodate varying missions, some of which cannot yet be defined with any precision. An SoS may be formed dynamically to perform a given mission and then reorganized as needed for other missions including those that have not yet been envisioned. A good SoS design might have modules that are not as good as their stand-alone counterparts that perform the same functions. As such, these modules might not be employed independently even though technically they could. An SoS is commonly characterized using terms such as *interoperable, synergistic, distributed, adaptable, trans-domain, reconfigurable,* and *heterogeneous*. As SoS complexity increases [Madni, 2012b; Vuletid et al., 2005], *system of systems integration* (SoSI) takes on a new meaning that comes with a host of new challenges. SoSI is intended to create a new mission capability through composition of component

*Author to whom all correspondence should be addressed (e-mail: azad.madni@usc.edu).

†This work was done as a private venture and not in the author's capacity as an employee of the Jet Propulsion Laboratory, California Institute of Technology.

systems that contribute to the overall capability. Once the component systems are developed, they are incrementally integrated and tested and, ultimately, deployed in the operational environment. Often, the component systems tend to have different customers and purposes and, as such, tend to employ different terminologies and concepts. Integrating such systems requires resolving semantic and syntactic differences which adds to overall SoS complexity. Furthermore, when SoSI involves integrating legacy systems, it is rarely the case that the documentation and expertise needed for smooth integration are readily available.

At the outset, it is worth noting that the conditions under which an SoS has to operate are becoming increasingly more challenging [Neches and Madni, 2012]. For example, defense and aerospace systems today have to satisfy affordability, adaptability, security, reliability, and resilience requirements [Madni and Jackson, 2009; Rehage et al., 2004; Mosterman et al., 2005]. The same is true of healthcare enterprises and energy grids.

Whether or not a given configuration is an SoS is more a matter of degree, rather than a binary decision. In fact, a system and an SoS lie on a continuum. Table I shows several key characteristics that are typically associated with an SoS. The more a system exhibits these characteristics, the more apt it is to view it as an SoS.

The criteria in Table I can be applied, for example, to determine whether or not the Curiosity rover (that landed on Mars in 2012) is an SoS. Curiosity comprises a number of interoperating subsystems designed to provide basic control, communications, power, thermal, and scientific services. However, these subsystems cannot operate in stand-alone fashion and, consequently, do not have operational independence. Therefore, the Curiosity rover does not qualify as an SoS.

Emergence is a less well-understood and more complicated concept. An SoS could exhibit new behaviors that were previously not observed because the conditions that caused them did not exist previously. In this case, the SoS is doing what it was designed to do but its behavior appears to be unanticipated because the outcome space was not fully explored. Is this emergence? It is debatable.

We can look beyond engineered systems to explore the concept of emergence. If a chemical solution exhibits precipitation at some temperature, is that emergence? Does that depend on whether or not precipitation was anticipated? Once again, it depends on how one chooses to define emergence. Emergence within complex adaptive systems (CAS) is easier to grasp with physical phenomena such as phase changes. However, biologists do not view this as a CAS phenomenon (no sensate "agents"). It is interesting to note how a phenomenon like phase change which initially appeared almost magical has become part of intent and design now that the phenomenon is understood. This observation raises further questions about emergence. Should emergence be redefined to cover the expected and designed, or should it be redefined in some other sense? Should emergence be defined along a continuum (i.e., is it a matter of degree)? These are but a few questions that need to be answered before emergence can be defined in a consistent and useful way.

There are other important considerations that come into play when a system assumes a role within an SoS. For example, it is likely that some degree of autonomy will be sacrificed when a system participates within an SoS. Also, some characteristics can potentially be both time-dependent and goal-dependent. Thus, at a given time, a system may want to control certain local resources and not make them available to the SoS to achieve its overall goal. At some other time, the system may have completed its local task, and released the previously tied up resources to the SoS. In this regard, Baldwin et al. [2012] define a game-theoretic approach that is relevant for analyzing such a collaborative SoS.

The remainder of this paper is organized as follows. Section 2 discusses the unique characteristics of SoS. Section 3 presents an SoS typology in use today. Section 4 discusses SoS integration from several different perspectives. Section 5 presents an SoSI ontology to inform and guide SoS integra-

Table I. Key Characteristics of SoS [Maier, 1998; Sage and Cuppan, 2001]

1. *Operational Independence of the Elements*: If the system-of-systems is disassembled into its component systems the component systems must be able to usefully operate independently. The system-of-systems is composed of systems which are independent and useful in their own right.
2. *Managerial Independence of the Elements*: The component systems not only *can* operate independently, they *do* operate independently. The component systems are separately acquired and integrated but maintain a continuing operational existence independent of the system-of- systems.
3. *Evolutionary Development*: The system-of-systems does not appear fully formed. Its development and existence is evolutionary with functions and purposes added, removed, and modified with experience and need.
4. *Emergent Behavior*: The system performs functions and carries out purposes that do not reside in any component system. These behaviors are emergent properties of the entire system-of-systems and cannot be localized to any component system. The principal purposes of the systems-of-systems are fulfilled by these behaviors.
5. *Geographic Distribution*: The geographic extent of the component systems is large. Large is a nebulous and relative concept as communication capabilities increase, but at a minimum it means that the components can readily exchange only information and not substantial quantities of mass or energy.

tion. Section 6 presents an SoS model for a directed SoS used for Earth seismic studies. Section 7 discusses SoS interoperability for different SoS types from a variety of perspectives. Section 8 presents human–systems integration challenges in SoS. Section 9 summarizes the contributions of this paper.

2. UNIQUE CHARACTERISTICS OF SYSTEM OF SYSTEMS

There are several integration characteristics that apply to both systems and an acknowledged SoS (Table II). These include: certification and accreditation (C&A), acquisition, structure, integration mechanisms, and verification (does the SoS conform to established requirements) and validation (does the SoS have the desired capabilities). *C&A* is a major complexity driver because of the multiple management layers, multiple funding sources, and lack of synchronization in the life cycles of the various systems within an SoS. *Acquisition* is a complexity driver due to multiple acquisition programs, multiple systems' life cycles across programs, and the need to achieve interoperability among legacy and new systems. *Structure* is a major complexity driver in that the structure of an SoS can change dynamically as systems continue to enter/exit the SoS. *Integration mechanisms* are a major complexity driver in that they need to support dynamic interoperability among constituent systems. *Verification and validation* (*V&V*) is a complexity driver because of the difficulty in synchronizing across multiple systems' life cycles, the dynamic entry/exit requirement for some of the SoS components, and the lack of a defined set of behaviors or requirements in some SoS.

In general, the boundary between a system and SoS is fuzzy with respect to these characteristics. However, in gen-

Table II. Comparing an Acknowledged SoS with System

Comparison Factors	System	Acknowledged SoS
Stakeholders	• Known at one level	• At both system and SoS levels • Include system owners with competing interests • In some cases, system owner has no vested interest in SoS • All stakeholders may not be recognized
C&A	• Program management aligned with program funding	• Added complexity due to existence of management and funding for both SoS and individual systems • SoS does not have authority over all systems
Operational Focus	• Meet operational objectives	• Meet operational objectives using systems whose objectives may not necessarily align with SoS objectives
Acquisition	• Aligned with acquisition milestones, requirements; has systems engineering plans	• Quite complex due to multiple systems lifecycles across acquisition programs involving legacy and development systems, new developments, opportunistic technology insertion • Typically have stated capability objectives upfront that may need to be translated into formal requirements
V&V	• Generally possible	• More challenging due to difficulty in synchronization across multiple systems' lifecycles, complexity of changing "parts," unspecified behaviors and requirements, and the potential for unintended consequences
Boundaries and Interfaces	• Pertain to a single system	• Pertain to systems that contribute to the SoS objective and enable the flow of data, control, and functionality across the SoS while balancing the needs of the system
Performance & Behavior	• To meet specified objectives	• Across SoS, satisfies needs of SoS users while balancing the needs of the component systems
Structure	• A-priori defined interfaces and interactions assure that assembled subsystems work together	• Dynamic structures and the potential for syntactic and semantic mismatches may lead to "lethal" configurations that could not be predicted implying that not all objectives of an SoS are achievable or that great care is needed in selecting SoS applications
Mechanisms	• Integrated and tested bottom-up following a hierarchical build-up; interfaces designed for functional components with limited use connectivity in mind; component requirement and interface verification of components as system is built preceded by functional validation	• Reusability and adaptability require flexible, generalized integration methods that provide semantic and syntactic compatibility that otherwise do not exist between systems that were not necessarily designed to be integrated. In unmanaged SoS, formal validation may not be possible and integration success is determined during SoS use.

eral, an SoS tends to be more complex than a system due to the added requirements for integration and a different emphasis and approach for design and validation. Creating detailed requirements and evaluating compliance is possible when the objectives of an SoS are clear, and the SoS structure is closely managed. For example, an SoS could provide local situational awareness from data collected and processed by field-wearable sensors and processors. Additionally, if reconnaissance aircraft data are available, then information about a neighborhood can be fused with information provided by ground sensors. Big picture situational awareness is achieved from integrating ground-based, aircraft-borne, and spacecraft sensors. Such an SoS has well-defined objectives that drive requirements for sensors, processors, and communications. These requirements can be flowed down to individual systems, and independently validated by each system engineering team.

In sharp contrast, consider the difficulty in generating requirements when the objectives and architecture of an SoS are unknown until stakeholders get together and specify a needed capability. In this case, requirements refer to communications protocols, information semantics, and usage restrictions. When each system is locally managed and maintained, assessing compliance even for general requirements may be left to local oversight with no external guarantees of compliance in terms of consistency and completeness.

There are additional differences between a system and an SoS beyond those identified in Table II. For example, the typical role of the human in a system tends to be mostly fixed with the human in the role of an operator (or an agent) with predefined interfaces. However, in an SoS, the human can orchestrate the SoS configuration, and invariably perform as an agent with changing interfaces to other agents/systems under the SoS. This dynamic behavior occurs because a system can enter and exit an SoS based on changing mission requirements that determine the mission capability package. Specifically, humans in an SoS can experience ongoing changes with respect to the agents they collaborate with as systems enter/exit the SoS. Continuous context switching associated with ephemeral SoS teams can produce excessive cognitive load that subsequently tends to show up in the form of human errors and increased human error rates [Madni, 2010, 2011]. Also, given that some participating systems in an SoS are temporary collaborators, issues of trust and lack of shared understanding can arise compromising overall SoS performance.

3. SoS TYPOLOGY

According to the Systems Engineering Guide for SoS (2008), an SoS can be classified according to the way it is managed and its openness to change and new capabilities (Table III). The form and rigor of SoSI is directly related to SoS type—an unmanaged SoS is inherently more difficult to integrate than a tightly managed SoS. Table III offers a coarse SoS classification defined in this guide. The boundaries between these types can be defined in terms of the degree of operational and managerial independence of the components.

In principle, each SoS type in Table III is open to any system that conforms with the interoperability rules of the SoS. However, an SoS may choose to restrict membership to only well-known, trusted systems. The Internet is an example of the former, while a secure banking network is an example of the latter. In an open, collaborative SoS, new, untrusted systems may enter and leave with unknowable impact on

Table III. Typology of System(s) of Systems [adapted from OUSD AT&L, 2008]

- Virtual SoS
  - no central management authority
  - no centrally agreed-upon purpose
  - systems can enter/exit dynamically based on mission requirements
  - potential for emergence of large scale system behavior (which may be desirable)
  - relies on relatively hidden mechanisms for maintenance
  - examples: Stock market, national economies
- Collaborative SoS
  - component systems interact more or less voluntarily to fulfill agreed upon central purposes
  - e.g., internet is a collaborative SoS; the Internet Engineering Task Force defines but can't enforce standards
  - central players collectively decide how to provide/deny service (some means to enforce and maintain standards)
  - examples: World-wide web, multi-robot systems
- Acknowledged SoS
  - has recognized objectives, a designated manager, and resources; however,....
  - constituent systems retain independent ownership, objectives, funding, development and sustainment
  - changes in the system are based on collaboration between the SoS and the system
  - example: DoD's Ballistic Missile Defense System
- Directed SoS
  - integrated SoS is built and managed to fulfill specific purposes
  - centrally managed during long-term operation to fulfill purposes and any new ones set by system owners
  - component systems maintain ability to operate independently, but their normal operational mode is subordinated to the central managed purpose
  - example: JPL's Deep Space Network

overall SoS integrity. In a closed SoS, the pool of systems are trusted (to some extent) and may also be required to undergo periodic inspection and validation to continue to maintain membership in the SoS.

4. SYSTEM OF SYSTEMS INTEGRATION

A virtual SoS and a collaborative SoS are not prespecified. This means that there can be no a priori expectation that information paths and/or information content are consistent in the SoS. Typically, an SoS is constituted and dynamically configured in a "loosely coupled" fashion to achieve the interoperability needed to create a mission capability package, and then reorganized as necessary to satisfy subsequent mission needs. In some cases, the systems, behaviors, and connectivity of an SoS become known only after a mission capability is specified. In this environment, traditional top-down integration and validation normally associated with systems integration are not applicable [Madni and Sievers, 2013]. Conversely, a Directed SoS and an Acknowledged SoS are prespecified making them predictable and compatible with functional verification and validation methods.

Figure 1 shows a simplified SoSI flow. The systems in this figure are linked directly or through an intermediary. While the figure shows only two SoSI configurations, there are many more possibilities as the "n" available systems are organically and dynamically configured to satisfy the needs of a mission capability package.

Figure 1 implies a well-controlled and rigorous flow. However, for an unmanaged SoS, structure and integration primarily revolve around the needs of the stakeholders, technologies involved, and legacy constraints [Madni, 2012a]. Moreover, cost and schedule constraints often conflict with the desire to thoroughly test a given level of integration before moving to the next level. At this point, there are two questions that one might ask: (1) what does it mean to dynamically integrate a collection of possibly unknown resources, and (2) what assurance is there that the SoS correctly provides the services needed by its users? The answer to both questions depends, in part, on the type of SoS and the interoperability and connectivity mechanisms used.

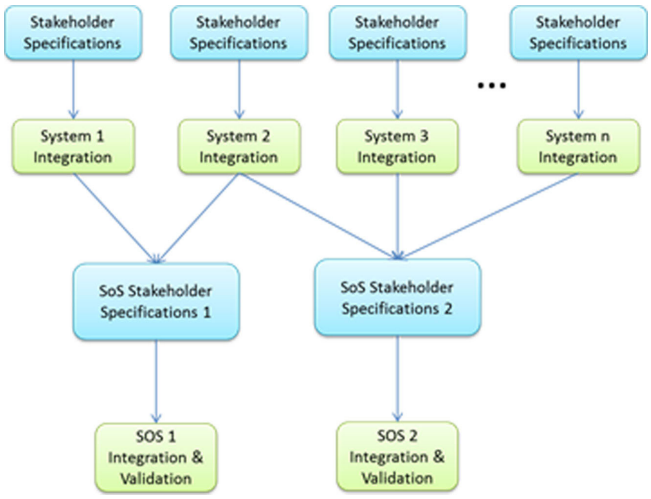

Figure 1. SoSI showing two possible integration configurations. The two SoS are configured organically and dynamically for a particular need, and subsequently reassembled to satisfy other needs. [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

Currently, there are no formal or uniformly applied mechanisms to identify and isolate systems that do not comply with the protocols and rules implicit with membership in an unmanaged SoS. Generally, SoS users stop using a "misbehaving" system and prevent that system from interoperating with other systems in the SoS. Various organizations today create and maintain lists of systems that are untrustworthy or have been found to host malicious applications (http://www.spamhaus.org).

5. SoS INTEGRATION ONTOLOGY

As described in an earlier paper [Madni and Sievers, 2013], the definition, management, and evaluation of integration approaches for systems and SoS are hampered by the lack of common terminology and clear definitions of technical and nontechnical interdependencies. To overcome this problem, we propose an SoS integration ontology shown in Figure 2. This ontology serves to:

1. Define standard terminology to describe integration features
2. Define a checklist of salient issues and influences that impact integration
3. Enable development of integration models that guide/support reasoning
4. Accommodate all stakeholders' needs and their respective viewpoints.

It is important to note that ontologies, in general, are not unique for a particular problem domain and, ultimately, all ontologies are transformed into useful representations that help answer questions and assist in reasoning in the problem space. An instantiated integration model links the properties shown in Figure 2 to each other and to properties associated with the behaviors, requirements, architectures, and allocations that are commonly held within a model of the technical characteristics of the SoS. Figure 2 also includes a "V&V" entity that captures pertinent V&V metrics and artifacts.

As shown in Figure 2, several core concepts are involved in SoSI. These concepts provide the semantic underpinnings for defining and managing SoSI. Practically speaking, the SoSI ontology offers a way of unifying the concerns in SoSI. The characteristics of the SoSI ontology and each concept in this ontology are described next.

SoSI Ontology. SoSI ontology includes both artifacts (documentation) and metrics (measurements or tests used to assess integration success). SoSI concepts are related to the SoSI domain through relationships (associations) such as: SoSI hasStakeholders, SoSI hasCA, and SoSI hasIntegration-Resources. In an instantiated SoSI model, data property assertions assign values to properties such as "documentation"

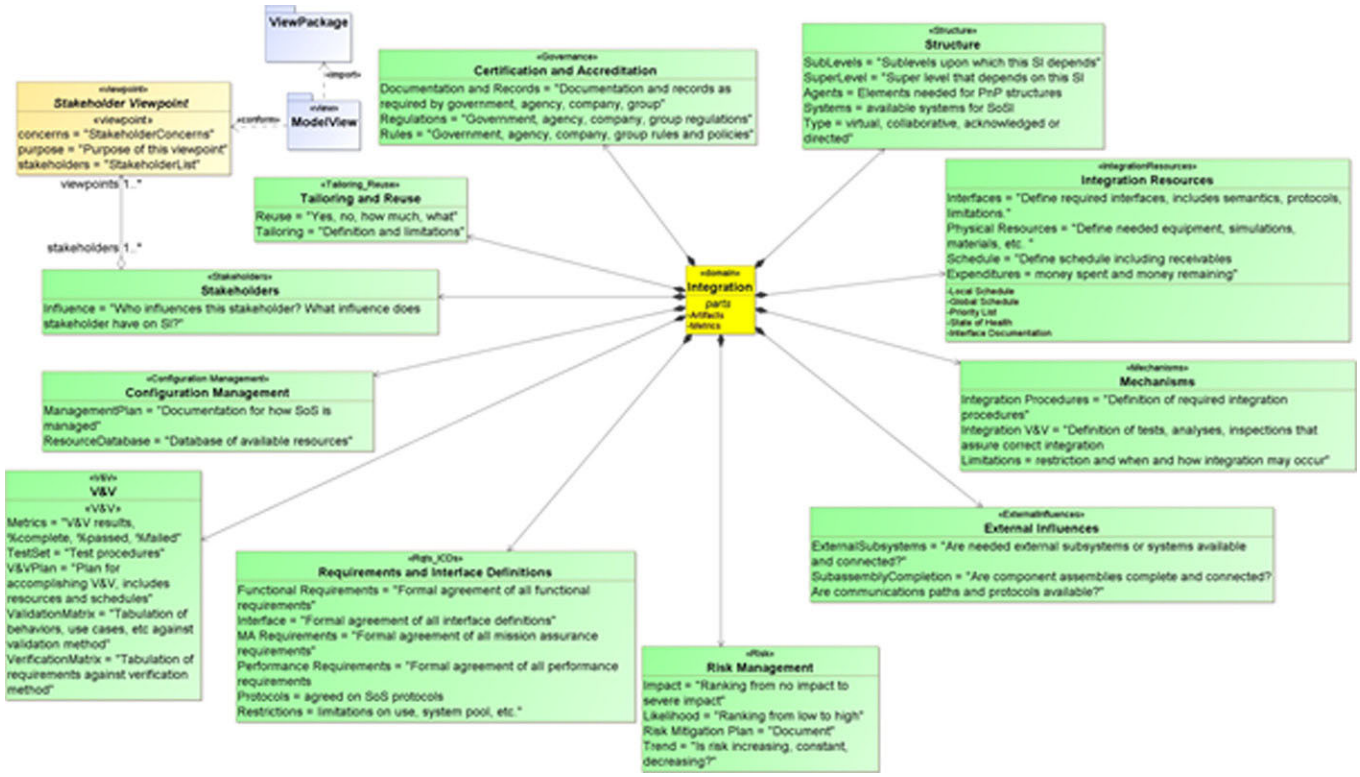

Figure 2. SoS integration ontology. [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

and "metrics." For example, a model could assign "Protocol Document D" to the documentation property meaning "**MySoSIntegration** *hasDocumentation* = **ProtocolDocumentD,**" where **MySoSIntegration** is an instance of SoS Integration. Data property assertions are not restricted to single entities. The assigned values are allowed to be a class or any element derived from a class. This characteristic enables defining, for example, a class structure for the metrics property of SoS Integration. If this class is named "**MyMetrics**," then **MySoSIntegration** *hasMetrics* = **MyMetrics** allows defining multiple metrics, and assigning values to each.

Object property assertions represent relationships between instantiations of the compositional elements in Figure 1. These assertions form the logical network that links elements of the SoSI model. As an example, a Stakeholder defined as "Manager" could have a property assertion, "hasCostReport," that is linked to the data property EarnedValue, which is contained in the concept, C&A. That is, **Stakeholder_Manager** *hasCostReport* = **EarnedValue** in which **EarnedValue** refers to an equation that involves data properties, schedule and expenditures, held in **Integration Resources**.

The network formed by object property assertions can be expected to grow quite large and complicated. Therefore, as with any complicated model, it is advantageous to define viewpoints that capture specific concerns and characteristics of stakeholders, and that are associated with views that organize the model according to the viewpoint. For example, one viewpoint for SoSI might be **ScheduleRisk**. This viewpoint captures the concerns associated with issues that could delay procuring, developing, or maintaining the SoS. This viewpoint is of interest to and could be shared by a number of stakeholders including the **ProjectManager**, the **SystemScheduler**, and the **MaintenanceSupervisor**. An example of a view that conforms to this viewpoint might be the availability of physical resources as defined in the instance of the **IntegrationResources** model element.

SoS Stakeholders. Stakeholders are individuals, organizations, or external systems with an interest in the SoS, or in a component system that could potentially become part of or participate in the SoS. Stakeholder "concerns" are those issues that are important to specific stakeholders. Some stakeholders can exert "influence" on some aspect of SoS integration and can, in turn, be influenced by other aspects. The concerns and influences of stakeholders define the "viewpoints" that characterize specific aspects and relationships (within the model) represented as views that conform with a particular viewpoint. There are typically a number of viewpoints for a given model that can be associated with one or more stakeholders. For example, a Project Manager (stakeholder) is typically concerned about schedules, costs, and risks. Figure 3 shows an instance of a viewpoint for this stakeholder that has a view defined as Earned Value Analysis. Earned Value Analysis depends on the expenditures and schedule values in the Integration Resources element. Other stakeholders (e.g., the Project Customer), who may also want

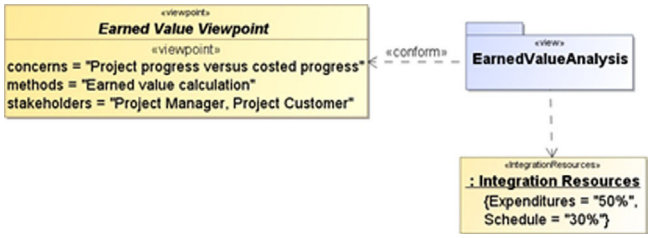

Figure 3. Viewpoint and view for two stakeholders: Project Manager and Customer. [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

to view/review an earned value report, can also share the Earned Value Viewpoint.

SoSI requires defining stakeholders, and then managing their needs and expectations. These needs and expectations can often be in conflict. Thus, managing them is as much a part of SoSI as the process of building and testing the SoS. All stakeholders are influenced by and can influence one or more stakeholders. Invariably, stakeholders monitor metrics associated with system parameters that relate to their concerns. Some stakeholders (e.g., designers, testers) also provide artifacts. The interactions, associations, and semantics that link stakeholders represent communications pathways, checks and balances, and testable assertions.

Integration Resources and Influence of External Factors. Because they are closely related, Integration Resources (or Enablers) and External Factors can be addressed together. Integration Resources define the "what," while "external factors" defines influences that affect the "what." That is, Integration Resources are the physical resources (equipment, facilities, materials, and so on), integration plans, required interfaces to external entities, and schedule and budget necessary for integration. External Influences contribute to Integration Resources but often may not be under the control of the SoS integrator. Thus, for example, suppose that an SoS plans to use a particular protocol defined by an international standards organization. Although the SoS integrator may have some influence on the decisions of the international body, it may not be able to change aspects of the protocol that are inconvenient or inefficient for the particular mechanisms planned for the SoS. In this case, an international body becomes an external influence on the design and integration of the SoS.

SoS Structure. The *structure* (or architecture) of an SoS defines the component systems and their interfaces. The dynamic nature of an SoS implies that the constituent systems and interfaces vary based on capabilities needed to satisfy mission goals. Moreover, certain functions may exist in more than one constituent system. These systems are incorporated within the SoS as needed to accommodate, for example, local maintenance schedules, availability, performance requirements, and interconnection traffic. While the specific components within the SoS structure need not be fixed, the mechanisms available to link and unlink systems need to conform to defined rules.

Figure 4 shows an exemplar SoS comprising "m" systems interconnected through two communications fabrics. In this

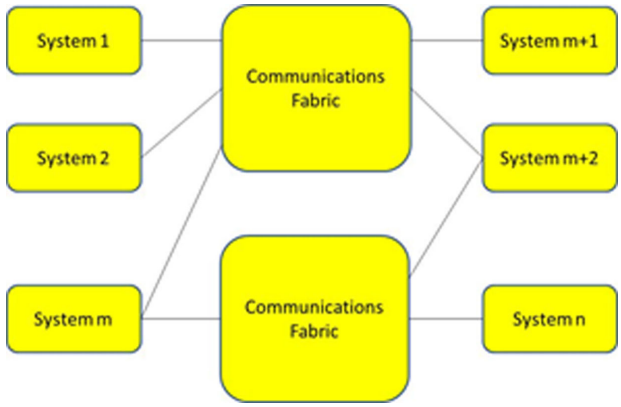

Figure 4. An example of an SoS comprising two distinct communications fabrics. [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

example, System m and System m + 2 may use either of the two fabrics. These fabrics do not necessarily have the same semantic and syntactic interfaces. This fact complicates SoS integration and validation. The example in Figure 4 also implies that connecting Systems 1 and n, for example, requires cooperation from System m.

In *virtual* and *collaborative SoS*, there is no central authority with the knowledge of available capabilities, and their ability to be integrated at any point in time. SoS integrators need an understanding of whether nondeterminism is an issue for users of the SoS to make informed decisions regarding the mechanisms used.

NASA's Deep Space Network (DSN; http://deepspace.jpl.nasa.gov/dsn) is an example of a Directed SoS structure as defined by DoD. It is important to note, however, that using the criteria in Table I, a directed SoS could not exist, because once management is centralized, an SoS becomes just another system. The DSN is an international network of radio antennas and data relays used to communicate with Earth-orbiting satellites, and interplanetary spacecraft, as well as carry out radio and radar astronomy observations. There are currently three stations in the DSN located at Goldstone California, Madrid Spain, and Canberra Australia (Fig. 5). Each station is locally managed and maintained. However, for certain NASA missions, they come together to provide communication with the spacecraft and to coordinate communication handover as the spacecraft goes out of view of one station and into the view of another. Station activities are coordinated by a centralized operations center that coincides with spacecraft visibility. Data collection and control activities are planned by the centralized operations center and automatically switched as signal strength declines at one station and increases at another. Although there are variations in when the stations assume communication related to Earth orbit and spacecraft trajectory, there are very well tested models that can predict what each station does and when. There are also a fixed set of functions that each station needs to provide while in contact and while waiting for contact. The SoS is carefully tested well in advance and simulated tracking and hand-offs are performed to assure that all resources are

Figure 5. 70M antenna at the Goldstone Tracking Station (http://deepspace.jpl.nasa.gov/dsn/gallery/goldstone3.html). [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

working as expected well before the need arises. While critical missions may use more than one station to assure uninterrupted communication, other missions might have only one station assigned. Typically, each station serves multiple missions and when not acting as a command and telemetry relay, can be repurposed for astronomy observation. The SoS is carefully controlled by a time-based activity schedule that dictates the configuration and services that a station must perform in support of flight missions as well as nonmission time that is scheduled by science teams.

The ARPANET, sponsored by the U.S. Advanced Research Projects Agency, evolved from work done at MIT Lincoln Labs, UCLA, Stanford, UC Santa Barbara, University of Utah, and Bolt, Beranek and Newman (BBN). The original ARPANET was a centrally managed, directed SoS but over time evolved into an unplanned, collaborative SoS—the Internet. The Internet itself has evolved from a chaotic collection of disparate resources available only to researchers and dedicated enthusiasts to one that is somewhat organized by web services and search engines. In the early days of the Internet, users knew how to access the resources they needed and made use of well-documented protocols for making connections and exchanging information. In many cases, users depended on personal communication with other users to find resources, learn semantic interfaces, and solve problems. The modern Internet is supported by application layers, object and semantic standards, and message transport protocols that hide many of the connectivity details and also assure communication compliance. From an SoSI perspective, the modern Internet exists by virtue of the protocols developed and stable intermediate services that provide the links between systems. Regardless of its management and determinism, a key structural theme for SoSI structures is stable and robust communications. This characteristic implies that SoS architects should exert as much control as possible over the systems and standards that form the basis of SoSI [Rechtin, 1991].

Certification and Accreditation (C&A). C&A defines the records, tests, regulations, rules, policies, and other such considerations that are binding on the SoS [Mayk and Madni, 2006]. For example, a company may write a policy that defines how components of a system must be grounded. All products from that company need to adhere to that policy to ensure that integration does not fail due to grounding-related incompatibilities.

In an SoS, C&A may take the form of international standards and protocols rather than corporate or government regulations. Compliance failure is not officially policed, but systems found lacking are kept from participating in the SoS until they become compliant or are made compliant. A centrally managed SoS may have formal requirements on a number of local operations, maintenance, testing, and scheduling. The form that C&A takes depends on the particular use of the SoS and the impact of failure. Table IV presents SoS examples within diverse sectors along with examples of representative C&A.

Mechanisms. Mechanisms are processes, procedures, tests, and inspections that are required for integration. They are generally motivated by safety and verification concerns. At the most basic level, mechanisms assure that appropriate safety precautions have been taken (e.g., checks performed before attaching a cable to a connector). At higher levels, procedures are put in place to ensure that enabling a given function for the first time does not adversely affect or cause the failure of other components, subsystems, or systems. It is important to note that mechanisms tend to be somewhat specific to given industries and products. Table V presents some common SoS integration mechanisms. These mecha-

Table IV. Representative C&A

System/Industry	Representative C&A
DoD Ballistic Missile Defense System (BMDS) (acknowledged SoS)	DoD approved communications protocols, local test and maintenance schedules, event reporting, backup, disaster recovery, formal documentation of failures and corrective actions
Internet (collaborative SoS)	Internet Engineering Task Force (IETF) imposed protocols and protocol implementations, working groups, communication standards, security, and so forth.
Banking (collaborative SoS)	Government regulations, international treaties and agreements, internal bank policies, backup and disaster recovery policies, central bank regulation, legal constraints, ISO standards, OCC standards, business codes, and licensing. C&A imposed by national and international banking authorities to enable the exchange of funds among banks around the world.

Table V. Key SoSI Mechanisms and SoS Categories (Table III)

SoS Type	Integration Mechanism	Example
Virtual SoS (deliberate/on-the-fly)	Integration is largely accomplished through adherence to documented standards for resource sharing, navigation, and documentation; SoS evolves with its environment; No authority to enforce compliance with governing in standards	WWW; control is exerted through published standards for resource naming, navigation and document structures; Web sites choose to abide/not abide by the standard; system is controlled by forces that assure cooperation and compliance with core standards.
Collaborative SoS	No centralized organization to enforce compliance with integration standards. Component systems voluntarily collaborate to fulfil larger purpose. Agreements enforced by acceptance/rejection of service requests among collaborators. Primary integr. mechanism is agreements documentation.	Elements are computer networks and major computer sites; some networks exchange information using documented protocols; protocol adherence is largely voluntary with no central authority with coercive power.*
Acknowledged SoS	Are collaborative with designated manager to enforce collaboration; Manager dictates accepted protocols and semantics for collaboration; can require evidence of adherence to interface specs.	Air Force's Air and Space Operations Center; Navy's Naval Integrated Fires Counter Air Capability
Directed SoS	Built for specific purposes and centrally managed; Component systems operate independently but their services are co-opted by centralized manager; Component systems are tightly coupled with centralized function through layered architectures.	JPL's Deep Space Network; comprises a number of stations with the ability to carry out science investigations; they come together when needed to support a space flight mission.

*Coercive power emerges through agreements among major sites to block traffic and sites observed to misbehave.

nisms are fundamentally applicable to any integration process. It is also worth noting that each mechanism has associated documentation that gets filed (i.e., persistently stored) in the development project database. These documents serve as evidence that certain required processes were in fact performed. As important, they become a source of information that can be useful for diagnosing anomalous behavior after an SoS has been deployed.

The banking industry, for example, is governed by a wide variety of local, federal, and international rules, regulations, treaties, and laws. There are ISO standards that define currency codes, securities identifiers, Eurobond identifiers, business codes, standard e-business ontology, licensing, and so forth. The U.S. Office of Comptroller of the Currency publishes a number of regulations and may take enforcement action when laws, rules, or regulations are violated. There are also a number of regulations published by the U.S. Consumer Financial Protection Bureau (CFPB), Federal Reserve Bank (FRB), Federal Deposit Insurance Corporation (FDIC), and regulations defined by each country the bank does business with. Also, banks may have internal policies and procedures for managing and protecting customer information and assets.

Although banks are integrated as a collaborative SoS, required C&A imposes tight restrictions on allowable integration mechanisms to ensure the safety and security of the bank and customers as well as mechanisms that conform to government regulation. The Bank Secrecy Act (BSA) of 1970, for example, requires financial institutions in the United States to detect and prevent money laundering. A financial institution must file five reports for transactions in excess of $10,000. Given that most transactions are managed electronically, each financial institution must provide a means to integrate information potentially from multiple sources and trigger a warning when any BSA condition occurs.

Risk Management. Risk management activities are an integral part of SoSI initiatives. Risk management during SoSI subsumes acceptance testing, deployment, transition from existing system to new SoS, operations, maintenance, logistics, training, and upgrades. SoSI needs integration plans which are derived from development plans, engineering plans, and test plans. NASA, for example, has issued a Risk Management Handbook (NASA-SP-2011-3422) that includes the informational flow and roles shown in Figure 6.

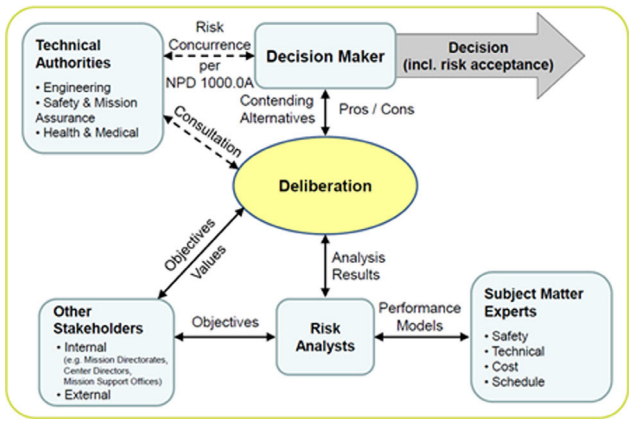

Figure 6. NASA risk information flow and functional roles (NASA/SP-2011-3422, http://www.hq.nasa.gov/office/codeq/doctree/NHBK_2011_3422.pdf). [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

Table VI. Key Elements of Configuration Management

- *CM Planning*: Defines the tools, entities, responsibilities, and relevant documentation that captures how the SoS configuration will be managed.
- *Identification*: Essentially the "parts list" covering the tracked CIs, ownership, versions, models and so forth.
- *Change Control*: Define the change process, responsibilities, and what documentation accompanies the changes
- *Configuration Status Accounting*: Metrics associated with the SoS, CI changes, and the state of its documentation
- *Configuration Verification*: Establishes the consistency between the SoS and its documentation

Risk items are typically inserted in a table that places risk likelihood on one axis and risk consequence on the other. The intersection of a likelihood and consequence defines the criticality of the risk. A color code is used in the matrix in which high-likelihood and high-consequence risks are coded red, intermediate risks are coded yellow, and low risks are coded green. Risk is assigned an action that indicates what, if anything, should be done to mitigate the risk. Additionally, the risk trend (increasing, decreasing, no change) is tracked.

The flow in Figure 6 and the risk table are applicable to a centrally managed SoS such as the DSN in which a Flight Director for a specific flight mission can articulate a set of objectives and values that are analyzed and adjudicated. Risk items are difficult to identify and even more difficult to track or mitigate in collaborative or virtual SoS. This is the case because of the dynamic nature of the SoS, the lack of specific goals, and little or no insight into the systems expected to participate in carrying out a particular capability. In this latter environment, one might almost want to assume the worst case, that is, the likelihood of the risk occurring is high, and the consequence is that the capability is not available. In noncritical applications this condition might be acceptable since the payoff when the capability does work is high. Of course, if a multi-billion-dollar spacecraft is heading to Mars, Flight Directors want very high assurance that no "red" or "yellow" risks are outstanding and that all "green" risks are under control.

Configuration Management (CM). CM typically comprises processes and documentation that control the performance, capabilities, and composition of the SoS (Table VI). CM assures that expected behaviors are present in the SoS and that these behaviors agree with their associated artifacts. Global CM is possible for an SoS that is centrally managed and, to some extent, for locally managed component systems. In general, however, systems within an SoS have no obligation to publish or maintain their configuration. Consequently, SoS configuration changes as new systems join, member systems leave, local maintenance is performed, and systems elect to temporarily opt out of the SoS entirely, or in part.

6. AN SoSI MODEL FOR A DIRECTED SoS

The SoSI ontology (Fig. 2) is used to develop an SoSI model using SysML notation for a Directed SoS used for Earth seismic studies. Figure 7 shows a package structure for this model. The top-level packages are SoSI Model and SoS Model. The SoSI Model contains the elements associated with defining and instantiation elements of the SoSI ontology. The SoS Model package contains the elements related to requirements, behaviors, and structures of the SoS.

The SoS comprises a central operations center and a number of remote data collection stations. Nominally, the stations collect science data as determined by a local scheduling function. For global studies or when a large earthquake occurs, the operations center forms an SoS with one or more

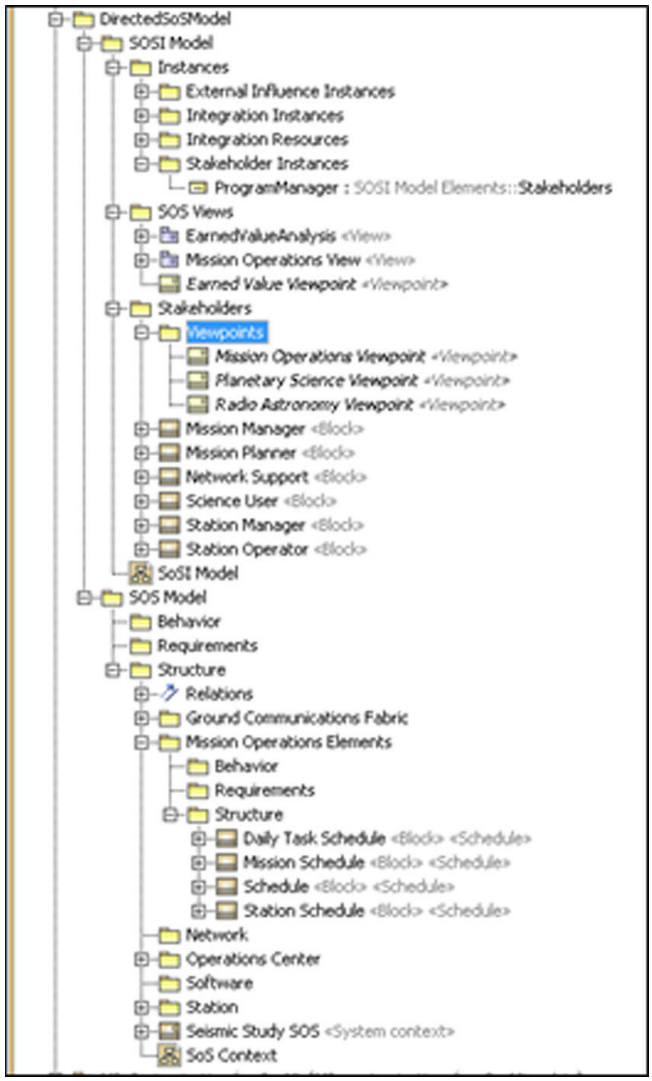

Figure 7. SoS package structure. [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

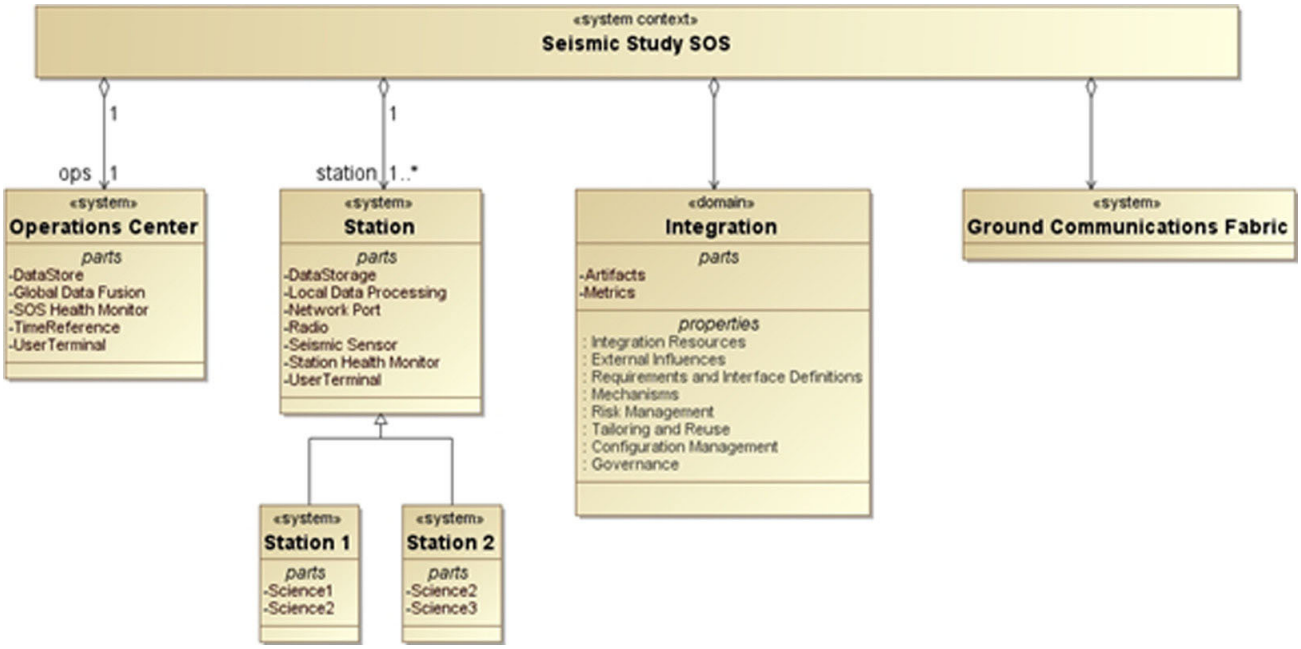

Figure 8. Simplified seismic study SoS aggregation diagram. [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

stations so that global patterns can be monitored. Figure 8 shows an aggregation diagram for the SoS. The aggregation diagram comprises an Operations Center and one or more Stations. Each Station has a number of basic properties and any number of specialized parts (data storage, local processing, network connections, sensors, and so forth) as required for the science observations which are carried out when the Station is not employed for global seismic studies. The Operations Center provides centralized coordination of the Stations for global science studies and data collection and has its own suite of subsystems (e.g., data storage, global data fusion, health and status monitoring).

Figure 8 includes the SoS Integration Domain defined in Section 5 and a Ground Communications Fabric that provides global connectivity when needed.

SoSI Stakeholders. Figure 9 shows six hypothetical stakeholders for the Seismic SoS. These stakeholders are: the mission manager, the mission planner, the science user, the station manager, the network support personnel, and the station operator. The roles, concerns, and stakeholder views for each stakeholder are presented in Table VII.

Figure 10 shows the Mission Operations Viewpoint which identifies the Mission Manager and Mission Planner as stakeholders. This viewpoint has a corresponding view that depends on properties within an instance of the Integration Resources element. That instance has data property assertions for Global Schedule, Local Schedule, Priority List, and State of Health properties as shown in Figure 10. Figure 10 also shows that the Mission Operations View "imports" Mission Operations Elements (i.e., the members of this package become members of the Mission Operations View namespace).

Using a fully instantiated model, a Mission Manager could query the model to determine when each station was planning maintenance and when each had been committed for local

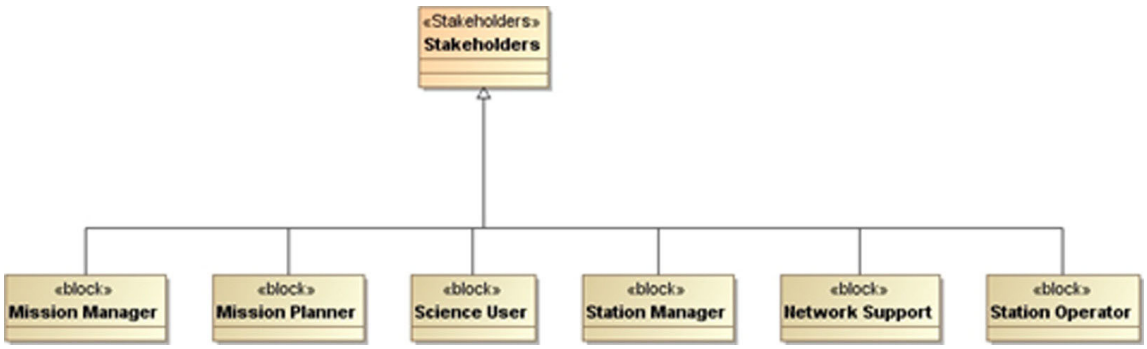

Figure 9. Seismic SoS stakeholders. [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

Table VII. Seismic SoS Stakeholder Roles, Concerns, and Views

Stakeholder	Role	Concerns	Views
Mission Manager	Responsible for coordinating global resources	Availability of resources when needed	Master schedule, priority lists, science requests station schedule
Mission Planner	Responsible for defining global mission goals and tasks with inputs from scientists	Prioritization of requests, scheduling	Science schedule, priority lists
Science User	User of science data (either from a global mission or from local experiments)	Scheduling priorities, data receipt	Science schedule, science data
Station Manager	Responsible for providing services required by missions and local science experiments	Equipment functionality, scheduling, staffing, operating funds	Station schedule, maintenance schedule, staff lists, funding
Network Support	Assures operability of communications network	Network functionality and configuration	Station schedule, network requirements, network status, network maintenance
Station Operator	Responsible for configuring and running the local station	Equipment functionality, scheduling	Station schedule, equipment status

science studies. From this information, the Mission Manager could populate a Daily Task schedule showing the times when certain SoS capabilities are available. The Daily Schedule could then be used by the Mission Planner, who is also a stakeholder of the Mission Operations Viewpoint. The Mission Planner could access the Priority List and add global science collection opportunities. Then, the Daily Task schedule can be allocated to each station and accessed by Station Managers, who have different viewpoints and views that also depends on the Integration Resources Instance (not shown). Figure 11 shows a representative schedule structure for a member of the Mission Operations Element package. It is seen that three child schedules inherit the properties and operations of the generalized schedule. One reason for defining this structure is that it aids semantic compatibility across the SoS thereby facilitating information sharing and analyses.

SoSI Structure and Mechanisms. The DSN is a directed SoS in which station systems are added and released according to a master schedule held by the Mission Manager. Much of the contents related to this element are imported from the SoS Model package shown in Figure 7. Figure 12 shows a simple instantiation of the SoS structure.

The communications ports C1 to C4 represent physical and logical interconnections between the operations center, communications fabric, and the stations. Nominally, requirements are written that define and constrain the C1–C2 and C3–C4 interfaces. Additional requirements are assigned to the communications fabric such that it implements its interface responsibilities. For example, the C3–C4 interface may require that stations produce outputs in metric units and use a particular data format that specifies the data units and data transmission using TCP/IP. On the other end, the Operations Center may expect Imperial units. In this case, the communications fabric would perform the translation between Imperial and metric units while hiding the details of the translation. We may know that stations will come and go during the life of this SoS, and we may also know that we cannot specify the internal operations of the stations. To protect against semantic errors, a protocol can be established in which units are explic-

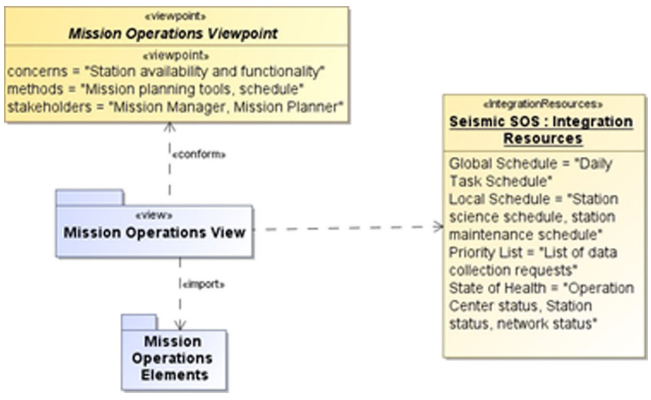

Figure 10. Mission Operations Viewpoint and View. [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

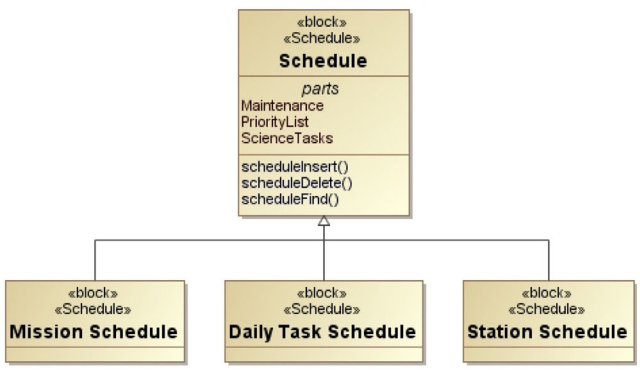

Figure 11. Example schedule structure and inheritance. [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

Figure 12. Simplified SoS structure. [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

itly stated. Then, the operations center can make a query that explicitly states the expected units, and a mechanism within the communications fabric would perform any needed transformations.

SoSI Resources and External Influences. A number of influences determine the composition and capabilities of the Earth Science SoS. For example, stations may be unavailable due to disruptions and contingencies such as power failures, earthquakes, and network outages. The state of health, an attribute of the Integration Resources element, collects current status information on which stations are available. Maintenance and other schedules, which are also contained within the Integration Resources element, enable scheduling of certain types of data collection for particular studies. For example, a station may be waiting for a replacement part which is on back-order. In this case, an external influence, that is, the vendor of the back-ordered part, interferes with integration. Also, the SoS depends on standards enforced by one or more organizations. These organizations are external influences on the SoS.

SoSI Requirements and Interface Definitions. Since the Earth Science SoS requires communication paths between the stations and the central Operations Center, we can assume that there are at least two governing organizations that oversee connectivity: the International Telecommunications Union (ITU) and the National Telecommunications and Information Administration (NTIA). The ITU (http://www.itu.int) publishes a set of handbooks and standards that cover electromagnetic effects protection, implementation guidelines, semantics, networking, quality of service, and security. NTIA (http://www.ntia.doc.gov) regulates radio spectrum sharing and the Internet. In our ontology, these documents are covered by C&A and Requirements and Interface Definitions.

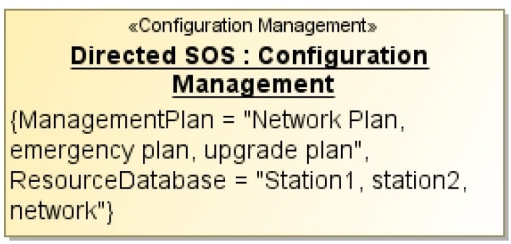

Figure 13. Configuration Management data properties. [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

SoSI Configuration Management. Each station in our SoS is responsible for maintaining local configuration documentation, and managing its change process. The Operations Center may request access to that information. However, whether or not that access is granted is not necessarily under the control of the Operations Center. In some situations, although each system is under local control, a single entity may provide blanket funding for all stations and the Operations Center. In those situations, enforcing documentation standards and configuration change processes can be influenced by the funding source's need to make the best use of funds (an external influence). For our hypothetical SoS, the Configuration Management Element has a defined set of configuration plans and a resource database (Fig. 13). The documents consist of a set of plans for managing the network and emergency configurations, and for performing upgrades. The resource database contains the stations and network components.

SoSI Risk Management. There may be a central authority (e.g., a blanket funding source) that defines the risk for the SoS. In this case, that authority can dictate a risk posture and the types of risk that are acceptable. However, if no such entity exists, then the risk is affected by how rigorously the individual stations comply with the standards and protocols defined for the SoS. Where standards or protocols do not exist, there may be a high likelihood of something not working as anticipated. However, the impact may be sufficiently small for the risk to be viewed as negligibly low. To some extent, SoS users influence risk and determine risk mitigation.

In the Earth Science example, the Mission Manager might define a risk associated with needing certain resources at a critical point in time. To mitigate that risk, the Mission Manager could require that each local station publish its schedule, that stations cannot vary from that schedule without the Mission Manager's approval, and that the Mission Manager has the ability to override the schedule if a warning is given of a certain amount of time remaining. The Mission Manager could also require a maintenance and testing schedule that assures the operability of each station's resources.

7. INTEROPERABILITY IS A CROSS-CUTTING ISSUE

Interoperability is the ability of distinct systems to share semantically compatible information and then process and manage the information in semantically compatible ways, enabling users to perform desired tasks [Madni and Sievers,

2013; Zeigler and Mittal, 2008]. The emphasis on distinct systems that were not designed to be integrated with each other is implicit in the concept of interoperability. Interoperability is intended to create a capability that serves an important human purpose and/or satisfies a mission requirement. Interoperability implies far more than getting systems to communicate with each other. It requires that the organizations involved employ some degree of compatible semantics and common interpretations of the information they exchange. The Levels of Conceptual Interoperability Model (LCIM) is a key advance in this regard [Tolk and Muguira, 2003]. A 2006 NRC Report on Defense Modeling, Simulation and Analysis lays out a strategy for meeting the interoperability challenge as well.

Interoperability is a cross-cutting concern that is beyond the scope of a single system development effort, or organization [Naudet et al., 2010]. It is also important to realize that systems are often designed and implemented before a recognized need for their interoperation exists. Such a need, when it arises, extends interoperability beyond the original scope of the system. Even so, there are internal concerns that need to be addressed by any system, if some day it is called on to interoperate with others. These include adherence to standards, choice of information processing procedures and algorithms, the validity criteria surrounding the representation and processing of information, interfaces to other systems and to their users, and the nonfunctional, quality attributes such as reliability, availability, security, privacy, and information assurance.

For results to be meaningful and valid, interoperable systems need to share common/compatible semantics. This requirement implies that interoperating systems must have compatible mechanisms for exchanging, representing, modifying, and updating semantic descriptions of information items. As important, an interoperable system needs to process information in ways that are meaningful to the other systems with which it interoperates. To the extent that the meaning and form of information is dynamic (i.e., can change over time), these systems need to be able to dynamically modify their information processing approaches and, possibly, their representations.

Since interoperability is a cross-system issue, it involves other cross-cutting concerns (e.g., security) to achieve interoperability solutions. The cross-cutting nature of interoperability has other important consequences. First, interoperability cannot be implemented piecemeal. Second, interoperability has a unique and crucial coordination role relative to the other cross-cutting concerns [Rothenberg, 2008]. Third, interoperability cannot be added as an afterthought without incurring substantial costs and diminished effectiveness.

In sum, when a system participates within an SoS, it is necessary to ensure that it understands the data, processing, and policies of the other systems in the SoS with which it interoperates, and that they, in turn, understand its data, processing, and policies. This requirement creates a need to explicitly represent and share the subset of system semantics needed to interoperate with these other systems. This need, which is rooted in the SoS construct, is rarely acknowledged or resourced within individual system design efforts, especially for preexisting systems that were designed without interoperability considerations in mind.

SoS Interoperability. It is difficult enough to implement interoperability when the system is being designed with interoperability in mind. It is even more difficult when this is not the case. Table VIII presents interoperability cases in increasing order of difficulty. The first two cases in Table VIII are demanding but straightforward, since they involve the design and implementation of a new system that is to be made interoperable with an intentionally designed SoS. The last three cases, however, are increasingly problematic, since they involve various combinations of retrofitting existing systems and achieving interoperability within ad hoc, non-intentionally designed SoS. Many real-world initiatives exemplify the last, most challenging case.

For an SoS, interoperability often serves as a surrogate for integration when independently developed, stand-alone systems are combined to provide a new capability in an SoS. In many cases, the SoS is not intentionally designed as an SoS. Rather, it is constituted and dynamically configured as needed. While an ad hoc SoS of this kind can create useful new capabilities, it can place significant stress on the component systems which, in general, were not designed to work with each other. Even for an intentionally designed SoS, it is challenging to make its component systems interoperate. Clearly, when an ad hoc SoS is created out of existing stand-alone systems, the challenge is considerably harder. Even so, interoperability is easier to achieve in such cases than integration, and in fact can offer some of the same advantages as integration while providing potentially greater flexibility and scalability. In all cases, to make systems work together effectively, it is essential to recognize that interoperability is a cross-cutting concern, which must be implemented pervasively both within and across systems.

Interoperability and Semantics. System interoperability needs to address both syntactic interoperability and semantic interoperability. *Syntactic interoperability* is the ability of two or more systems to communicate and exchange data. In this case, specified data formats and communication protocols are key. Standards such as Extensible Markup Language (XML) and Structured Query Language (SQL) offer a means to provide syntactic interoperability. Syntactic interoperability is a prerequisite to semantic interoperability. *Semantic interoperability* is the ability of two or more systems to automatically interpret the meaning of exchanged information, and to accurately produce useful results for end users (of both systems). For semantic interoperability, any two systems within an SoS have to agree on a common information exchange reference model that can be used by both. However, a standard model that has been adopted by an SoS community to facilitate interoperability among systems in the SoS community is clearly preferable. Regardless, the information exchange request content is unambiguously defined, that is, what is sent is consistent with what is understood.

Specifically, for humans to collaborate and tools to work together, it is imperative that the information exchanged is both correct and meaningful [Mayk and Madni, 2006]. This requires compatibility among concepts, terms, processes, and policies. Such compatibility is essential for semantic interoperability. The compatibility requirements for semantic in-

Table VIII. SoS Interoperability Cases

1) New system developed as part of an intentionally designed new SoS
   - overall authority should be empowered and resourced to perform SoS integration across all systems in the SoS
   - each individual system development effort should be required and resourced to adhere to interoperability guidelines mandated or recommended by this authority.
2) New systems are designed to work with an existing, intentionally designed SoS
   - required and resourced to adhere to whatever interoperability guidelines exist for that SoS
3) New system developed to be part of an existing SoS that was not intentionally designed as such
   - such a SoS may not have any interoperability guidelines
   - new guidelines may have to be developed retroactively by whatever authority controls the SoS
   - or, by collaboration with developers, maintainers, and users of other systems in the SoS.
4) An existing system, not intended to be part of any SoS, is subsequently required to interoperate with other systems in an existing or new, intentionally designed SoS
   - the existing system must be retrofitted to conform to the interoperability guidelines of the SoS.
5) An existing system that is not part of any SoS is required to interoperate with other systems in an ad hoc SoS
   - not only is the SoS not designed intentionally as a SoS, but no new system development effort is underway to incorporate interoperability as a new concern
   - instead, existing systems in the SoS must be made to interoperate, with minimal redesign/modification
   - this is demanding and risky; requires retrofitting pervasive, cross-cutting, semantic interoperability into specific aspects of multiple independent, existing systems not designed to interoperate or be part of an SoS
   - unfortunately, it is a common occurrence in the real-world to require existing systems to interoperate in ways never envisioned by their designers
   - without some degree of formal SoS integration, coupled with significant, resourced redesign and reimplementation of existing systems, such initiatives are unlikely to succeed.

teroperability encompass: terminology and controlled vocabularies; data definitions; units of expressions; computational methods and assumptions used to produce results; conceptual and functional models; key policies (e.g., access, authentication, authorization, security, transparency, accountability, privacy); business process models; atomic transactions; interface definitions and procedure invocation; state and mode definitions and compatibility; and clear definition of limits and restrictions.

If terminology and data are not aligned, there exists the likelihood of misinterpretation in exchanged information. If processes and procedures are incompatible, then it may be difficult or impossible to combine them to create new, integrated processes that implement new (composite) capabilities. If computational algorithms and information processing are not semantically compatible, the results are likely to be meaningless or, worse yet, misleading. If policies (e.g., privacy, access) are misaligned, interactions may be infeasible or in conflict with specific organizational policies. From the foregoing, it is evident that semantic interoperability is desirable (and sometimes essential) for meaningful interactions among systems, applications, and organizations. It is also the case that semantic interoperability is a substantial, potentially costly undertaking (as a design goal) because of increased validation and verification complexity.

Often, organizations or systems may have good business/technical reason to maintain distinct semantics. In such cases, to assure interoperability such distinctions need to be made explicit so that mechanisms can be developed that automatically map one set of semantics to another. It is worth noting that the mappings may not be one-to-one and may include semantics from one set that have no equivalent semantics in the other.

Also, organizations may not wish to expose their internal semantics to system users (e.g., human users, other organizations). Therefore, to enable system users to understand and interact with a "black box" effectively, an organization must either modify its internal semantics, match external user semantics, or map internal semantics to external user semantics. However, developing explicit semantics requires articulating the key assumptions and tacit knowledge shared by a community of practitioners with a common purpose and mindset. Invariably, these practitioners tend to be unfamiliar with formal methods for knowledge representation.

Semantic representation includes the Semantic Web 2.0, and XML-based tools such as Resource Description Framework (RDF) and Web Ontology Language (OWL). However, current semantic representation tools do not provide mechanisms to support interoperability. Therefore, an SoS should include a semantic interoperability development process model to facilitate the development of requisite semantics. Such process models can enable organizations to explicitly represent and align their processes with those of other organizations with whom they wish to interoperate.

"Designing In" versus "Retrofitting" Interoperability. Practically speaking, interoperability is achieved in one of two ways. It can be designed in (at system inception), or it can be retrofitted (after the fact). Clearly, designing interoperabil-

ity into a system is easier, more effective, and less expensive. Yet the need for interoperability is not always apparent when a system is designed, leading to the need to achieve interoperability after the fact. While retrofitting interoperability may be worthwhile, it may come at an enormous cost. The cross-cutting nature of interoperability requires it to be infused into several aspects of a system (e.g. external interfaces, user interface, use of standards and communication protocols, internal processing, policies, and procedures for data handling; data access privileges). Modifying these aspects of an existing system to make it interoperable will typically require substantial effort. Also, since interoperability depends on shared semantics, retrofitting interoperability can require major redesign of the system's representation and algorithms, as well as realignment of policies. These considerations can potentially pose serious obstacles.

System Architecture Informs Interoperability Design. Ultimately, interoperability depends on the architecture of the SoS [Chen et al., 2008]. This architecture can be explicit or implicit, formal or ad hoc. When preexisting or independently designed systems are required to interoperate in the absence of a common or reference architecture, the linkages between any two independently designed systems are invariably designed and implemented in ad hoc fashion, producing an ad hoc integration architecture.

Ad hoc architectures typically lack coherence and scalability. Therefore, it is usually desirable to develop open architectures. However, in practice, the cost and schedule of the initial development might not seem justifiable to managers and customers. So, at a minimum, interoperability should be part of the design trade space and given serious consideration in defining the system's life cycle. Ideally, by adopting a reference architecture and explicit architectural framework, and assuring that each system conforms to the framework, it becomes possible to achieve significantly greater interoperability among disparate systems. However, often times it is impractical to enforce such a framework for preexisting systems. In such cases, the result is an ad hoc SoS. In general, the architecture of an SoS, whether predesigned or ad hoc, can enhance or inhibit the ability of the component systems of the SoS to interoperate. In other words, some architectures promote interoperability, while others do not. A Service-Oriented Architecture (SOA) is an example of an architectural framework that promotes interoperability between its component services so long as they comply with the SOA paradigm and its tenets.

Interoperability and Reuse. Reuse is a key motivation for interoperability. The ability to compose an SoS out of reusable component systems has the potential to make the resulting SoS cheaper, more reliable, easier and faster to develop. It also has the potential to make the resulting SoS more consistent, easier to maintain and support, and more flexible and scalable. Reusability and interoperability converge in strategies such as the SOA, in which the behavior and interfaces of components are formally defined, thereby allowing them to interoperate with each other, and also be reused as components of multiple systems. A reusable component must provide a function that is of general value to other components. In some cases, this can be achieved by offering a single, well-defined capability that is of general use, such as looking up names in a directory or registry. The need to balance the generality of reusable components against their ability to be customized (i.e., customizability) is a key challenge. There is no general strategy to ensure that components will achieve this balance. Therefore, each component must attempt to find its own "sweet spot" that balances generality against customizability in ways that depend on its function and its relationship to other components.

Moreover, as noted above, the services in SOA are typically designed (or at least packaged) as reusable, interoperable components rather than as stand-alone systems in their own rights. These services are envisioned as components of a larger system, rather than existing systems that are to be combined into an SoS. It is, therefore, unclear that the reusability of SOA services translates into reusability of interoperable systems in an SoS. Also, interoperability achieved through SOA can produce a testing nightmare unless strict constraints are placed on the interfaces, timing, and functionality. It might also become necessary to stipulate that SOA services are either stateless or at least have strict limitations on states and rules for state transition. Also, transactions need to be atomic to prevent side-effects, if the occurrence of faults prevents their completion.

Interoperability Pros and Cons. Interoperability offers a number of advantages including: (a) *increased flexibility,* by allowing mixing and matching of systems; (b) *creation of new capabilities*, by composing new functions from existing ones; and (c) *increased cost-effectiveness*, by allowing reuse of existing systems and capabilities [Rothenberg, 2008]. The mixing and matching of systems enables performing unanticipated/unprecedented tasks from new combinations of existing functions. This can be accomplished on-the-fly (for nonrepetitive tasks), or a priori in a principled manner for repetitive or recurring tasks. Similarly, interoperability can reduce the cost of creating new capabilities by allowing existing systems to be reused in multiple ways for multiple purposes. An unheralded advantage of interoperability is that it hides overall system complexity from users by creating the illusion of an integrated system. Interoperability is usually accomplished through a common user interface, uniform semantics, and uniform policies and procedures. However, in the real world this is not always the case. In some cases, interoperability is achieved through somewhat "clumsy" means. In fact, one might question if such systems can be claimed to be truly interoperable. One example of clumsy integration is that employed on the International Space Station. The ISS employs a low-speed 1553 interface between itself and any experiment bolted on to it. While there is also a high-speed Ethernet interface for downlink from an experiment, there is no direct uplink path. Thus, if an experiment needs the high-speed link, crew members have to put the data on a thumb drive and walk it to another computer that can perform data uplink over the Ethernet. While one could argue that this is an interoperable system, it is clearly not automated, which is typically what is implied by true interoperability.

Despite obvious advantages, interoperability also has some disadvantages stemming from the increase in technical complexity and "open" system design [Rothenberg, 2008]. In particular, issues of privacy and security arise when systems are made interoperable. As important, costs can escalate

from having to make systems interoperable. Finally, interoperability adds technical complexity to system design in that new interoperability requirements are now imposed that the designer has to satisfy. Even so, the benefits of interoperability invariably outweigh the costs.

8. HUMAN–SYSTEMS INTEGRATION CHALLENGES IN SoS INTEGRATION

Humans within an SoS have to continually adapt to changes in configuration, modes, and levels of autonomy of the SoS based on changing mission requirements [Madni, 2010]. These behaviors tend to dramatically complicate the lives of humans and often result in human error. This is not surprising in that while humans exhibit adaptivity, the adaptation tends to be slow, not always complete, and occasionally not possible [Madni, 2010, 2011]. Also, humans tend to be poor at multitasking and context-switching, especially when context-switching frequency exceeds a maximum threshold. While several advances have been made in human systems integration [Booher, 2003], existing methods, processes, and tools are inadequate for integrating humans with adaptable SoS [Madni, 2010, 2011]. To fill this gap, research is needed to create new methods, processes, and tools that enable the development of dynamic HSI strategies for SoS. To this end, HSI research needs to integrate, build on, and extend the existing body of knowledge in HSI to address the needs of adaptable systems. The existing body of knowledge in HSI is quite fragmented and addresses diverse topics such as information overload, dynamic attention reallocation in multitasking and context-switching situations, distributed decision making and team coordination stress, ability to back up automation when needed, and cognitive bias in shared human–machine decision making. Integrating this fragmented body of knowledge for a variety of use cases is a promising starting point for realizing the goal of integrating humans with adaptable systems.

9. CONCLUDING REMARKS

Advances in system of systems integration (SoSI) have become critical today as SoS continues to grow in scale and complexity, and as humans continue to assume a variety of roles in SoS [Madni, 2010; Brown and Eremenko, 2009; Hively and Loebl, 2004]. In this paper, we have discussed the different definitions of SoS, key challenges in SoSI, and an underlying semantic model for SoS integration. We have identified external factors as potentially having a significant impact on SoSI. We have discussed SoSI in terms of requirements and interface definitions, emphasized the importance of C&A, and presented various forms that C&A can take based on a variety of factors. We identified the various stakeholders in SoSI along with their concerns, influences, metrics, and needed resources. We also presented various mechanisms (i.e., processes, procedures, tests, and inspections) needed for successful SoSI. We discussed interoperability in terms of advantages/disadvantages, examples, implications for SoS, and designing in versus retrofitting interoperability. In conclusion, we remind the readers that SoSI continues to be a fertile area of research and development. It is our hope that this paper will serve as a starting point for systems engineers who have the responsibility for planning and executing SoSI initiatives.

REFERENCES

R.L. Ackoff, Toward a system of systems concept, Manage Sci 17 (1971), 661–671.

W.C. Baldwin, T. Ben-Zvi, and B.J. Sauser, Formation of collaborative systems of systems through belonging choice mechanisms, IEEE Trans Syst Man Cybern Part A 42 (2012), 793–801.

S. Biffl, A. Schatten, and A. Zoitl, Integration of heterogeneous engineering environments for the automation systems lifecycle, 7th IEEE International Conference on Industrial Informatics (INDIN 2009), pp. 576–581.

M. Bonjour and G. Falquet, Concept bases: A support to information systems integration, Lect Notes Comput Sci 811 (1994), 242–255.

H.R. Booher (Editor), Handbook of human systems integration, Wiley Online Laboratory, 2003.

O.C. Brown and P. Eremenko, Value-centric design methodologies for fractionated spacecraft: Progress summary from Phase I of the DARPA System F6 Program, AIAA SPACE 2009 Conference & Exposition, 14–17 September 2009.

T.R. Browning, Applying the design structure matrix to system decomposition and integration problems: A review and new directions, IEEE Trans Eng Manage 48(3) (2001).

G. Cabri, Agent-based plug-and-play integration of role-enabled medical devices, Joint Workshop on High Confidence Medical Devices, Software, and Systems and Medical Device Plug-and-Play Interoperability, 2007, HCMDSS-MDPnP, 25–27 June 2007.

D. Chen, G. Doumeingts, and F. Vernadat, Architectures for enterprise integration and interoperability: Past, present and future, Comput Ind 59 (2008), 647–659.

M.W. Chowdhury and M.Z. Iqbal, Integration of legacy systems in software architecture, Paper presented at Specification and Verification of Component-Based Systems, 2004.

F.A. Cummins, Enterprise integration: An architecture for enterprise application and systems engineering, Wiley, New York, 2002.

J.S. Dahmann, G. Rebovich, Jr., and J. Lane, Systems engineering for capabilities, CrossTalk J Defense Software Eng 21 (2008), 4–9.

Defense Modeling, Simulation and Analysis: Meeting the Challenge, Committee on Modeling and Simulation for Defense Transformation, National Research Council, Washington, DC, 2006.

L.M. Hively and A.S. Loebl, Horizontal system-of-systems integration via commonality, Oak Ridge National Laboratory, TN, 2004.

M.O. Kahn, M. Sievers, and S. Standley, Model-based verification and validation of spacecraft avionics, Infotech@Aerospace Conference, 2012.

V. Kotov, Systems of systems as communicating structures, Hewlett Packard Company, 1997.

A.J. Krygiel, Behind the wizard's curtain: An integration environment for a system of systems, CCRP, July 1999.

A.M. Madni, Integrating humans with software and systems: Technical challenges and a research agenda, INCOSE J Syst Eng 13(3) (2010).

A.M. Madni, Integrating humans with and within software and systems: Challenges and opportunities (Invited Paper), CrossTalk J Defense Software Eng May/June 2011.

A. Madni, Adaptable platform-based engineering: Key enablers and outlook for the future, INCOSE J Syst Eng 15(2012a), 95–107.

A.M. Madni, Elegant systems design: Creative fusion of simplicity and power, INCOSE J Syst Eng 15 (2012b), 347–354.

A.M. Madni and S. Jackson, Towards a conceptual framework for resilience engineering, IEEE Syst J 3(2) (2009).

A.M. Madni and A. Moini, Viewing enterprises as systems-of-systems (SoS): Implications for SoS research, J Integrated Design Process Sci 11(2) (2007), 3–14.

A.M. Madni and M. Sievers, Systems integration: Key perspectives, experiences, and challenges, INCOSE J Syst Eng 16(4) 2013.

M.W. Maier, Architecting principles for systems-of-systems, Syst Eng 1(4) (1998), 267–284.

M.W. Maier and E. Rechtin, The art of systems architecting, 3rd edition, CRC Press, Boca Raton, FL, 2009.

W.H.J. Manthorpe, Jr., The emerging joint system of systems: A systems engineering challenge and opportunity for APL, Johns Hopkins APL Tech Dig 17 (1996), 305–313.

I. Mayk and A.M. Madni, The role of ontology in system-of-systems acquisition, Proceedings of the 2006 Command and Control Research and Technology Symposium, San Diego, CA, June 20–22, 2006.

P.J. Mosterman, J. Ghidella, and J. Friedman, Model-based design for system integration, Second CDEN International Conference on Design Education, Innovation, and Practice, Alberta, Canada, 2005.

NASA system engineering handbook, DIANE Publishing, 2010.

Y. Naudet, T. Latour, W. Guedria, and D. Chen, Toward a systemic formalization of interoperability, Comput Ind 61 (2010), 176–185.

R. Neches and A.M. Madni, Towards affordably adaptable and effective systems, INCOSE J Syst Eng 16 (2013), 224–234.

E.G. Nilsson, E.K. Nordhagen, and G. Oftedal, Aspects of systems integration, Proceedings of the First International Conference on Systems Integration, 1990, pp. 434–443.

Office of the Undersecretary of Defense for Acquisition, Technology, and Logistics (OUSD AT&L), Systems engineering guide for systems of systems, Washington, DC, August 2008.

E. Rechtin, System architecting: Creating and building complex systems, Prentice–Hall, Englewood Cliffs, NJ, 1991.

D. Rehage, U.B. Caol, M. Merkel, and A. Vahl, The effects on reliability of aircraft systems based on integrated modular avionics, Lect Notes Comput Sci 3219 (2004), 224–238.

J. Rothenberg, Interoperability as a cross-cutting concern, RAND Corporation, 2008.

A.P. Sage and C.D. Cuppan, On the systems engineering and management of systems of systems and federations of systems, Inf Knowl Syst Manage 2 (2001), 325–345.

C.E. Siemieniuch and M.A. Sinclair, Systems integration, Appl Ergon 37 (2006), 91–110.

V. Stavridou, Integration in software intensive systems, J Syst Software 48 (1999), 91–104.

A. Tolk and J.A. Muguira, The levels of conceptual interoperability model, Predings of the Fall 2003 Simulation Interoperability Workshop, Orlando, FL, 2003.

M. Vuletid, L. Pozzi, and P. Ienne, Seamless hardware-software integration in reconfigurable computing systems, Des Test Comput 22(2) (2005), 102–113.

B.P. Zeigler and S. Mittal, Towards a formal standard for interoperability in M&S/system of systems integration, Symposium on Critical Issues in C4I, 2008.

Azad Madni is a Professor in the Daniel J. Epstein Department of Industrial and Systems Engineering, and the Director of the multidisciplinary Systems Architecting and Engineering (SAE) Program in the University of Southern California's Viterbi School of Engineering. He holds joint appointments in USC's Keck School of Medicine and Rossier School of Education. He is the founder, Chairman, and Chief Scientist of Intelligent Systems Technology, Inc. He received his B.S., M.S., and Ph.D. degrees from the University of California, Los Angeles. His research has been sponsored by several major government research agencies including OSD, DARPA, DHS S&T, MDA, DTRA, ONR, AFOSR, AFRL, ARI, RDECOM, NIST, DOE, and NASA. He is the recipient of the 2011 Pioneer Award from the International Council of Systems Engineering. In 2012, he received INCOSE-LA's Exceptional Achievement Award for transformative advances in systems engineering. He is also the recipient of SBA's 1999 National Tibbetts Award for California, the 2000 Blue Chip Entrepreneurship Award from MassMutual and the U.S. Chamber of Commerce, the 2008 President's Award, and the 2006 C.V. Ramamoorthy Distinguished Scholar Award from the Society of Design and Process Science. He is also the recipient of the Developer of the Year Award from the Technology Council of Southern California in 2000 and 2004. He is a Fellow of AAAS, AIAA, INCOSE, and SDPS, and a Life Fellow of IEEE and IETE. He is listed in the major Who's Who including Marquis Who's Who in Science and Engineering, Who's Who in Industry and Finance, and Who's Who in America.

Michael Sievers is a Senior Systems Engineer at Caltech's Jet Propulsion Laboratory and a Visiting Lecturer in System Engineering at the University of Southern California. He conducts research in model-based systems engineering as well as contributing to a number of JPL's flight missions. He holds a Ph.D. degree in Computer Science from the University of California, Los Angeles, and is a member of INCOSE and a Senior Member of IEEE and AIAA.

Safety Science 70 (2014) 358–366

Contents lists available at ScienceDirect

Safety Science

journal homepage: www.elsevier.com/locate/ssci

Review

Safety in System-of-Systems: Ten key challenges

Catherine Harvey [a,*], Neville A. Stanton [b]

[a] Human Factors Research Group, Faculty of Engineering, University of Nottingham, Nottingham NG7 2RD, UK
[b] Transportation Research Group, Faculty of Engineering and the Environment, University of Southampton, Southampton SO17 1BJ, UK

ARTICLE INFO

Article history:
Received 29 January 2014
Received in revised form 13 May 2014
Accepted 6 July 2014
Available online 1 August 2014

Keywords:
System-of-Systems
Socio-technical system
Military
Hawk Jet

ABSTRACT

A System-of-Systems (SoS) describes a large and distributed network of component subsystems which are themselves complex and autonomous. The goals of a SoS can only be achieved by the interactions of the subsystems and not by any subsystem in isolation. A thematic analysis of the academic literature defining SoS was conducted and the 75 most frequently used words were identified. Based on this, ten key challenges of SoS were identified. These challenges are described in this paper, along with examples from a military case study of a Hawk Jet missile simulation activity. This review is intended to extend the reader's understanding of the current state of knowledge of SoS and to exemplify key challenges in terms of a contemporary safety case study.

Contents

* Corresponding author. Tel.: +44 (0)1159514363.
E-mail address: catherine.harvey@nottingham.ac.uk (C. Harvey).

http://dx.doi.org/10.1016/j.ssci.2014.07.009
0925-7535/

1. Introduction

The concept of sociotechnical systems (STS), or Systems of Systems (SoS), is to explain, analyse, and understand how multiple and distributed system artefacts and agents engage together. Interoperability is the key to successful system integration. This brings challenges from a safety perspective as it inevitably means there are increased interfaces and interactions between multiple components. The prevalence of SoS is increasing as the communication interfaces which underlie them become more commonplace (Qureshi et al., 2007). Managing safety in SoS is no longer just about the safety-related behaviour of one group of people but about the interactions between multiple diverse groups with potentially differing cultures, training and equipment, at the present time but also in the future.

1.1. Socio-technical systems versus System-of-Systems

The early focus in organisational safety was on individual equipment failures as the cause of accidents (Reason, 1997; Salmon et al., 2012). This view evolved to encompass human errors and individual acts as part of error causation; however, since World War II the complexity of systems and their interactions has increased and this has been accompanied by a focus on failures within the wider organisational system (Salmon et al., 2012; Leveson, 2004; Trist, 1981). This reflects a shift in the unit of analysis (Robinson, 1982) towards the sociotechnical systems view, which originated in the 1950s from work conducted by the Tavistock Institute (see Trist and Bamforth, 1951). The sociotechnical systems concept highlighted the need to consider technological and social development within an organisation 'simultaneously rather than consecutively' (Jaques, 1951). This paradigm produced a shift in the view of organisations as comprising separate social and technical systems, each with a differing approach, to a balanced view of organisations as sociotechnical systems in which the 'best match', or 'joint-optimisation', between people and equipment should be sought (Qureshi et al., 2007; Robinson, 1982; Trist, 1981; Trist and Bamforth, 1951; Walker et al., 2008). Salmon et al. (2012) described recent thinking about the occurrence of accidents in a system as pertaining to the interactions between 'human and systemic factors in complex socio-technical systems'. This reinforces the view of the human and technology components of a system as highly interactive and influenced to a great extent by the complexities of the context within which they interact. Based on the sociotechnical view of organisations, we can define a STS as delivering outcomes that are a product of the collaboration between humans and technology, as opposed to a product of either the humans or technology functioning in isolation (Qureshi et al., 2007). Voirin et al. (2012) suggested that STS should be viewed as comprising three dimensions: human, technical, and organisational; with intrinsic complexity arising from the multidimensional interactions between components.

Recently, the term 'System-of-Systems' (SoS) has been adopted in reference to large-scale, complex, socio-technical networks. Like STS, SoS also has no widely accepted definition and is still under development as a concept (Jamshidi, 2011; Rendon et al., 2012). Maier (1998, p.267) described SoS as:

> 'an emergent class of systems that are built from components which are large-scale systems in their own right'

Whilst the personal computer could technically be described as a SoS (it is made up of a number of interconnected 'subsystems'), SoS is generally used to refer to a large and distributed network or 'supersystem' (Jamshidi, 2011), whose subsystems are themselves large and highly complex (Maier, 1998; Jamshidi, 2011). The key distinction described by Maier (1998) is that a SoS is made up of components which operate and are managed as independently functioning (autonomous) entities and continue to do so within a SoS (as opposed to a personal computer, the components of which do not function independently). In other words, each component has its own goals (Jamshidi, 2011) but must collaborate with the other components to achieve the goals of the SoS (Alexander and Kelly, 2013). Furthermore, these collaborations between the components of the SoS enable specific goals which could not be achieved by any of the single components in isolation (Rasmussen, 1997; Von Bertalanffy, 1950). Examples of SoS include the internet (Maier, 1998), Intelligent Transport Systems (ITS) such as Advanced Traveller Information Services (ATIS) (Maier, 1998), Air Traffic Control (Alexander and Kelly, 2013; Harris and Stanton, 2010; Mearns et al., 2013), military units with Network Enabled Capacity (NEC) (Alexander and Kelly, 2013), the oil and gas industries (Zohar, 2010), road transport networks (Salmon et al., 2012), computer security (Carayon, 2006), the nuclear industry (Reiman and Pietikäinen, 2012; WANO, 2011) and the UK National Health Service (Benn et al., 2009; Flin, 2007).

There are obvious overlaps between the definitions of STS and SoS. STS is a more mature concept than SoS, although both terms share similarities, i.e. the connectedness between a number of individually managed and operated components to produce outcomes that could only have been achieved through that connectedness and not by any one of the individual components alone. The authors suggest that whilst SoS are always examples of STS (as they are always comprised of both social and technical components), a STS is not necessarily always a SoS because it may not have all of the characteristics that have been specified for a SoS, including autonomous components and openness. From this point forward, SoS is the focus of this paper, although it is assumed that this term encompasses concepts from the STS literature.

2. Identifying the challenges of SoS

SoS create a number of unique and complex challenges which must be managed in order to ensure safety and efficiency (Salmon et al., 2012). A review of the literature on SoS was conducted to understand the particular challenges that need to be addressed. Academic papers that provided definitions of SoS were identified (Robinson, 1982; Maier, 1998; Alexander et al., 2004; Kirwan, 2001; Jones, 1995; Stanton et al., 2012; Carayon, 2006; Jamshidi, 2011; QINETIQ, 2009; Qureshi et al., 2007; Salmon et al., 2012; Boardman and Sauser, 2006; Oxenham, 2010; Walker et al., 2008) and any text referring directly to the definition of these concepts was copied into QSR nVivo for further analysis. A word frequency query was performed on the text to identify the 75 most frequently occurring words used to describe SoS, and these were grouped into five 'themes':

- Network structure.
- Complexity and external influence.
- An emergent whole.
- Information transfer across boundaries.
- Continual change and culture.

Within these themes, ten keywords were identified (see Table 1): sociotechnical/organisational (interactions), complex(ity), open(ness), emergent behaviour, unpredictab(ility), boundaries, responsib(ility), change, legacy, and cultu(re). This subdivision was based on the authors' interpretation of the academic literature in this area and the keywords are intended to characterise the key challenges of SoS, which need to be managed in order to ensure safety.

Table 1
Top 75 most frequently used words in definitions of SoS/STS.

Network structure	Complexity and external influence	An emergent whole	Information transfer across boundaries	Continual change and culture
Sociotechnical/organisational	*Complexity*	*Emergent behaviour*	*Boundaries*	*Change*
	Work		Across	Continually
Network	Actions	Whole	Area	Different
Systems	Capability	Adaptive	Geographic	Increase
Human	Control	Together	Information	Diverse
Organisation	Operate	Group	Interactions	
Technical	Properties	Joint	Connections	*Legacy*
Technology	Purposes	Large	Collaborate	Future
Components		Overall	Relationships	Development
Autonomous		Goals	Communicate	
Autonomy	*Openness*	Gathering	(Non-)linear	
Distinct	Outside	Integrated	Lines	*Culture*
Independently	Environment	Optimisation	Distributed	Behaviour
Individual	Effects		Many	People
Parts		*Unpredictability*	Multiple	Social
Elements		Design	Local	
Constituent		Predicted		
		Safety	*Responsibility*	
			Pressure	

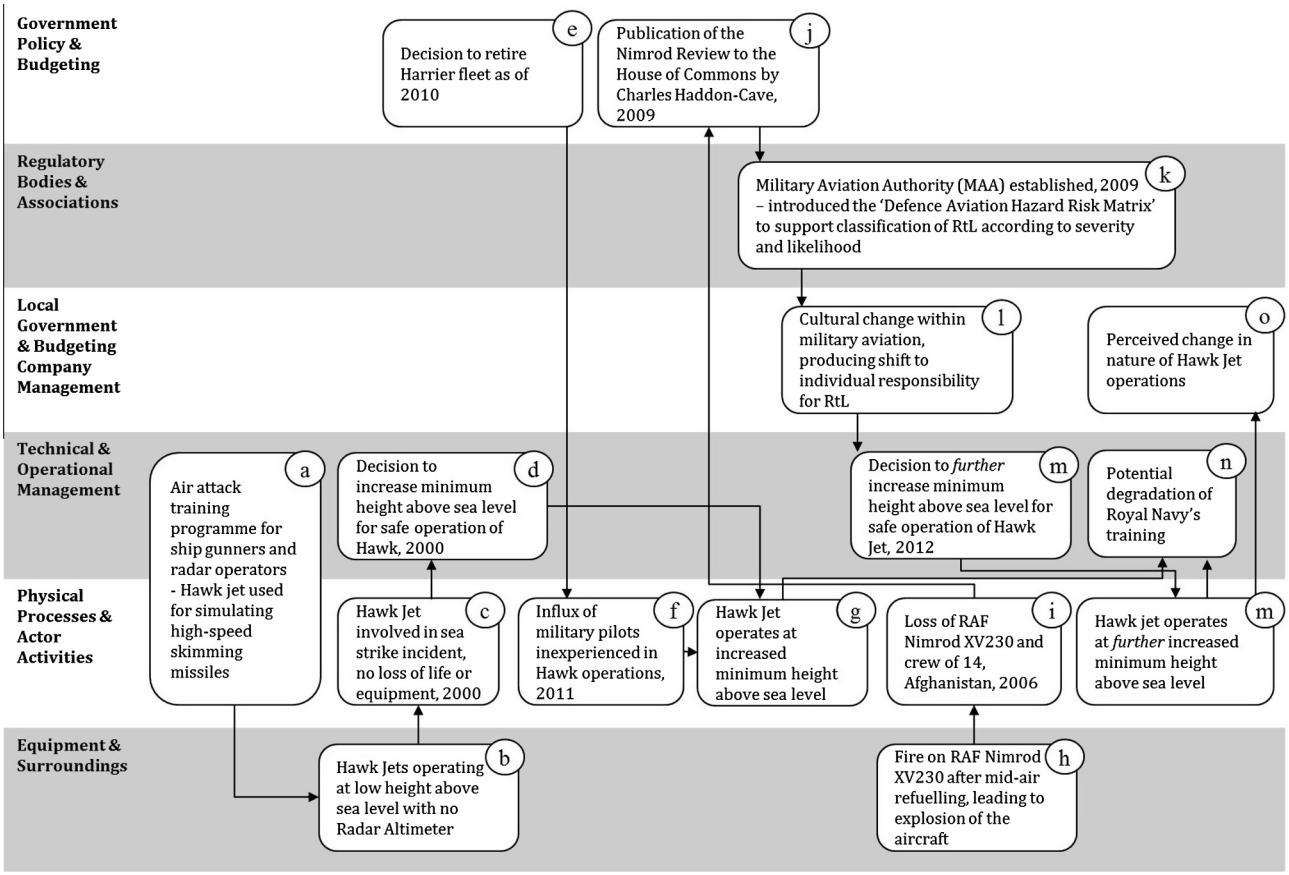

Fig. 1. AcciMap showing subsystems within the Hawk Jet SoS.

This thematic analysis provides an up-to-the-minute account of the contemporary academic thinking about SoS and can be used to characterise the key challenges that we currently face in managing these large, complex networks. To provide the reader with an in-depth understanding of these challenges, Section 4 describes each challenge in terms of current academic thinking and in relation to a case study from the military domain. The case study is intended to demonstrate how our characterisation of SoS can support our understanding of a real-world SoS, which is currently in operation and subject to on-going safety assessment. The UK MoD is a clear example of a SoS, and one in which performance efficiency and safety of the human component is absolutely crucial. The particular focus here is on the operation of the Hawk Jet by the Royal Navy and this is described in detail in the following section.

3. Case study: Hawk Jet missile simulation

A case study was identified through interviews with a subject matter expert (SME): this described the use of the Hawk Jet by

the UK Royal Navy to simulate missile attacks on surface ships. The analysts were provided with a high-level overview of the case study in an initial interview with the SME. This was followed up by a second, in-depth interview about the case study with the SME, conducted by two analysts. This resulted in a detailed account of the Hawk Jet and its use in missile simulation, which was supplemented by extra information from official documentation including Military Aviation Authority (MAA, 2010, 2011) guidelines, the official report into the Nimrod accident (Haddon-Cave, 2009) and Royal Navy safety assessment guidance (Royal Navy, 2012). Operation of the Hawk Jet to simulate missile attacks against surface ships is viewed here as a SoS because it comprises a number of interconnected subsystems, which are themselves complex. This SoS is illustrated in an AcciMap in Fig. 1. The AcciMap shows the decisions and actions taken by the subsystems of the SoS. Each node in the AcciMap is labelled (a, b, c. . .) to correspond with the description in the text in the following paragraphs. The year in which each event occurred is also included where applicable, to give an indication of timescale.

The Royal Navy uses the Hawk Jet to simulate air attacks on ships during sea training of ships' gunners and radar operators (event 'a' in Fig. 1). The Hawks are used to simulate enemy aircraft attacks and high-speed skimming missiles fired against ships (Royal Navy, 2012). In order to perform these simulation activities, the Hawk must be flown at a low height above sea level (b); however, the Hawk is not equipped with a Radar Altimeter (Rad Alt), which provides a highly accurate measure of the altitude of the aircraft above the sea. This makes flying the Hawk accurately at very low levels extremely difficult, and requires a high level of expertise to perform safely. Part of the safety management process used by the MoD involves the assessment of 'Risk-to-Life' (RtL) for personnel involved in a particular SoS. Prompted by events over a number of years, there have been some significant changes to the RtL assessment for the Hawk SoS.

1. Hawk sea strike

In 2000, a Hawk Jet was involved in a sea strike incident as a consequence of very low level flight (c). Although there was no resulting loss of life, this incident prompted a decision by the Royal Navy to increase the minimum allowable flying height above sea level for the Hawk (d).

2. Shift in pilot skill levels

As part of its Strategic Defence and Security Review (H M Government, 2010) took the decision to retire the Harrier jet from service in October 2010 (e). As a result of this, a number of Royal Navy pilots who would have flown the Harrier were diverted into the Hawk program (f). Traditionally the Hawk has been flown by Civilian pilots under contract to the Royal Navy (Royal Navy, 2012): these pilots have extensive military experience in fast jets, which includes low level flight supported by a Rad Alt. This experience provided mitigation against the RtL for the Hawk air attack simulation task; however, the cohort of military pilots did not have this same level of experience and the RtL had to be reassessed in light of this (g).

3. RAF Nimrod XV230 accident

In 2006, RAF Nimrod XV230 suffered a catastrophic explosion after a mid-air refuelling procedure (h): this caused the deaths of all 12 crew members plus two mission specialists and total loss of the aircraft (i). The Government requested a comprehensive review into the airworthiness and safe operation of the Nimrod (j), which was delivered by Charles Haddon-Cave (2009). The report described the development of the safety case for the Nimrod as 'a story of incompetence, complacency, and cynicism' (p. 161) and concluded that it was undermined by the widespread assumption that the Nimrod was safe because it had been flown successfully for the preceding 30 years (Haddon-Cave, 2009). The report also identified organisational changes in the years prior to the Nimrod accident as having significant influence; these included a shift in organisational culture toward business and financial targets 'at the expense of functional values such as safety and airworthiness' (p. 355). As a consequence of the findings, (Haddon-Cave, 2009) recommended the establishment of an *independent* Military Aviation Authority (MAA) to properly assess RtL and shape future safety culture (k). Further recommendations included the need for strong *leadership*, a greater focus on *people* to deliver 'high standards of safety and airworthiness' (p. 491) and increased *simplicity* of rules and regulations. The tragic consequences of the Nimrod accident, along with the recommendations of the Haddon-Cave report, effected a culture change within military aviation: this resulted in a decision to assign *individual* accountability for RtL assessments to 'Duty Holders' (DH), where previously responsibility for risk had been held at the organisation level (l). The newly established MAA produced guidelines for the assessment of RtL, in the form of the Defence Aviation Hazard Risk Matrix (MAA, 2011) which supports the classification of single risks according to their estimated severity (catastrophic, critical, major, minor) and likelihood (frequent, occasional, remote, improbable). The resulting risk level determines at which level of DH the risk is held.

The organisational changes brought about by the events described above (i.e. influx of junior pilots) prompted reassessment of the RtL for the Hawk air attack simulation activity. The goal of safety management in the UK military is to reduce risk to a level which is As Low As Reasonably Practicable (ALARP): this is reached when 'the cost of further reduction is grossly disproportionate to the benefits of risk reduction' (Ministry of Defence (MoD), 2007). The RtL for all Hawk operations is frequently reassessed and the shift in pilot experience levels, as described above, prompted changes to the RtL for the Hawk air attack simulation activity. In order to reduce this RtL to a level which was ALARP, a decision was taken by the Royal Navy DH with SME advice to further increase the minimum height above sea level (m). A potential consequence of this decision is the degradation of Royal Navy surface fleet training against very live low level targets, as the Hawk can no longer accurately simulate sea skimming missile attacks on surface ships (n). These events have changed the nature of Hawk operations within the UK MoD (o).

4. SoS themes and key challenges

In the following sections, each key challenge is described along with a discussion of how it is manifest in the Hawk Jet missile simulation case study.

4.1. Network structure

A SoS is founded on the interactions between multiple social and technical components, within an organisational context (challenge 1). A defining feature of SoS is that its components are independently managed and operated, in other words they are autonomous. For a SoS to achieve successful operation, the components need to not only achieve their individual goals but also contribute to the goals of the SoS, which are not always pre-defined.

4.1.1. Challenge 1: socio-technical–organisational interactions

A SoS is made up of different organisational levels, which have different roles, responsibilities, expectations and strategies. Zohar (2010) warned that there may therefore be inconsistencies

between the different interacting components within the network, with higher levels within the network likely to be more concerned with policy making, compared to lower levels at which there will be an emphasis on executing procedures in a practical context. Just within the human element of SoS, there is considerable variation in the needs and expectations of the people involved with and affected by its operation. When combined with technologies within a specific organisational context, the result of this interaction is an almost infinite range of behaviours and outcomes, which need to be appropriately managed within the organisation so as not to cause unexpected and/or adverse events. In order for a network to be considered a SoS, its individual components should have local goals and individual capabilities; in other words, they should have some level of autonomy (Alexander et al., 2004; Jamshidi, 2011). This means that the components can and do operate individually, and are not just in existence for the purpose of the SoS (Boardman and Sauser, 2006). The challenge is to view the SoS as an integrated whole in order to manage SoS-level effects, but to simultaneously account for the independent operation of each individual component and to 'absorb' any changes or disturbances at the subsystem level.

In the Hawk case study, human components include aircraft pilots and Duty Holders (Delivery Duty Holder – DDH; Operating Duty Holder – ODH; Senior Duty Holder – SDH). In this case there is an issue with the increasing turnover of pilots and consequent lack of specialist training and experience of low level flight. System components include the aircraft and aircraft systems, as well as surface ships which use the Hawk for missile simulation as part of training. In this case an issue was caused by the Hawk not being equipped with Rad Alt, as this aircraft was never designed for very low level flight. Organisational components include Royal Navy (RN) and individual RN units, Royal Air Force (RAF), the Government, and the Military Aviation Authority (MAA). At the Royal Navy unit level, responsibility is for SME identification and assessment of RtL. The RAF also fly the Hawk and consequently contribute to recognition of issues and RtL assessments. In this case, Government decisions had indirect consequences for skill level of personnel, with the retirement of the Harrier fleet diverting inexperienced pilots into the Hawk programme.

4.2. Complexity and external influence

The actions of a SoS are dependent on its interacting components, which are connected via numerous links, causing events to propagate through the network in a non-linear fashion. This makes the behaviour of the SoS as a whole extremely complex (challenge 2). SoS require joint optimisation of the social and technical components, and because SoS are open systems (challenge 3), this optimisation has to occur within the context of the external environment and organisational framework.

4.2.1. Challenge 2: complexity

Hollnagel (2012) attempted to define complexity as involving many parts which are connected by relationships that are not straightforward. Within a SoS, there is a high degree of interactive complexity in the relationships between its human, technology and organisational components (Leveson, 2004; Perrow, 1999). Complexity is a product of the numerous subsystems within a SoS, each with different subgoals, and reflects the dynamic nature of information flow between these subsystems (Rasmussen, 1997), with inputs constantly being made to keep the SoS functioning (Salmon et al., 2012). SoS comprise complex interactions, characterised by unexpected and hidden event sequences, and few linear interactions, which are simple, expected, familiar and visible (Perrow, 1999).

Operation of the Hawk is inherently complex and reliant on the successful integration of a large number of simultaneous activities as diverse as reducing the risk of bird strikes, runway cleaning, and local air traffic orders. This creates technical, training, organisational, and operational complexity. There are a number of 'subsystems' which interact as part of the Hawk RtL SoS (as shown in the AcciMap in Fig. 1), each of which has inherent complexity and autonomy, and must be managed as part of the wider SoS.

4.2.2. Challenge 3: openness

As well as exhibiting intrinsic complexity, SoS also exhibit extrinsic complexity as they are 'open' (Von Bertalanffy, 1950) and so are affected by the wider environments and contexts within which they operate and, in return, have influence within these environments (Von Bertalanffy, 1950; Salmon et al., 2012; Walker et al., 2008). An open SoS has a boundary with the wider environment and other SoS, across which informational exchanges can occur; this is in contrast to a closed system which has no associated input/output of information and is, as a consequence, unresponsive to changes in the environment (Walker et al., 2008).

The Hawk SoS is defined loosely as the network of subsystems which have some impact on the operation of the Hawk, i.e. pilots, aircraft, RN, RAF, MAA, MoD and Government. There is no defined boundary, as even the highest level Government decisions (i.e. retirement of the Harrier, commissioning of the Haddon-Cave report) have had an influence on the Hawk operations. For this reason it is considered to be 'open'. This also means that the environment within which the SoS operates is subject to change, for example, the UK's engagement in a new conflict could impose different demands on the Hawk, which would in turn affect the RtL assessment in this SoS.

4.3. An emergent whole

The many and varied interactions between the individual components of a SoS produce emergent behaviour (challenge 4) which cannot be predicted (challenge 5) based on the performance of the individual subsystems in isolation.

4.3.1. Challenge 4: emergent behaviour

The phrase 'the whole is greater than the sum of its parts' has been attributed as far back as Aristotle (Sinclair, 2007) and has been used by authors when describing systems (e.g. Rasmussen, 1997; Von Bertalanffy, 1950) and more recently, SoS (e.g. Alberts, 2011). This suggests that the behaviour of a SoS will be something beyond that of the individual subsystems, groups, and individuals that function within it. In other words, the SoS will exhibit 'emergent behaviour' as a consequence of the interactions between these subsystems, groups and individuals (e.g. see Leveson, 2004; Wilson, 2012 for references to this term). This behaviour will not be planned to fulfil certain functions, rather it will evolve through the interactions and collaborations that naturally develop within the SoS (Maier, 1998). Leveson (2004) described safety as an emergent behaviour of an SoS: this is because it is not inherent in systems and cannot be predicted based on the individual components of the SoS (Dekker et al., 2011).

In the Hawk Jet missile simulation case study, turnover of aircrew increased as a result of the retirement of the Harrier: this was not planned for in the original RtL assessment activity as the Government's decision could not be anticipated by the RN. The Hawk fleet had been 'steady state' for many years, with little change to operating procedures and skill/experience of personnel. This steady state was disrupted by the increase in junior military pilots being diverted to the Hawk programme. The disruption was unexpected and the consequences unpredictable.

4.3.2. Challenge 5: unpredictability

The complexity of SoS and the non-linearity of the relationships between actions and outcomes can often lead to a situation which is virtually impossible to accurately predict (Dekker et al., 2011; Hollnagel, 2012; Kirwan, 2001). Alberts (2011) argued that we can never have enough information about initial conditions, no matter how good the information is, to make accurate predictions about future behaviours and to establish cause and effect relationships. This issue is exacerbated in SoS as each individual component cannot know fully how its actions and the interactions with other components at a local level will impact on the system at a global level (Dekker et al., 2011; Alberts, 2011; Alexander and Kelly, 2013).

The major source of unpredictability in this SoS comes from other subsystems acting on the operation of the Hawk: although the retirement of the Harrier fleet was planned at Government level, the resulting influx of inexperienced military pilots into the Hawk programme was an unpredicted consequence. In this case unpredictability occurs at an organisational level, as a result of policy changes, rather than at an individual level. Performance of the Hawk as an individual subsystem is tightly controlled and highly predictable based on historical data. In training missions, which is the primary domain of the Hawk, the behaviour of the aircraft is also highly structured and therefore predictable.

4.4. Information transfer across boundaries

Information and communications are transferred across boundaries between subsystems (challenge 6): this defines how the SoS functions. As each component of a SoS must be independently managed and operated, responsibility (challenge 7) is designated at the individual component level as well as being transferred across boundaries and upwards to a global level for the whole SoS.

4.4.1. Challenge 6: boundaries

Within a SoS, the subsystems operate and are managed as individual entities, each with their own purposes to fulfil (Maier, 1998): when these entities come together in a SoS, there needs to be a shared understanding of how information and actions are transferred and coordinated across the boundaries between them. Maier (1998) suggested that it is the boundaries between components that define the SoS, rather than the components themselves because the success of the SoS is dependent on the communications at these boundaries. Boundaries exist between all of the subsystems of a SoS and can be classified as organisational, geographical, cultural, or temporal (Carayon, 2006). Issues often arise at these boundaries due to a lack of shared understanding between subsystems (QINETIQ, 2009), creating coordination problems, ambiguity and conflicts among independently made decisions (Leveson, 2004). Furthermore, it is difficult for individuals to make decisions when they lack knowledge about how these decisions will translate across boundaries (Leveson, 2004) and also to the SoS as a whole (Salmon et al., 2012; Alexander and Kelly, 2013). As well as problems of communication, boundaries between SoS components also create problems of responsibility/ownership (QINETIQ, 2009), language consistency (or lack of) (Oxenham, 2010; Jamshidi, 2011), and training scope (Robinson, 1982; Knudsen Tveiten et al., 2012).

There are many subsystems involved in the operation of the Hawk and information needs to cross all of the boundaries between them. For example, the Hawk is operated by both the RN and RAF, so communication of issues needs to cross an organisational boundary between services.

4.4.2. Challenge 7: responsibility

With the lines between humans and technology becoming increasingly blurred in SoS, the issue of responsibility is receiving progressively more attention. This relates to the emergent behaviour phenomenon, such that the way in which one component (e.g. a technology supplier) expects its inputs (e.g. technology products) to be used is not necessarily the way in which they will be used in practice (e.g. by another SoS component) (QINETIQ, 2009; Alexander and Kelly, 2013). There arises the question who takes responsibility for the success or failure of that input? Additionally, is training sufficient to enable SoS components to take responsibility for outcomes across the SoS? Traditionally, both the scientific and legal views of accident causation focussed on individual blame (Roed-Larsen and Stoop, 2012; Dekker et al., 2011; Leveson, 2004); however, SoS theory argues for systemic causes, a shift in perspective which has important implications for the issue of responsibility. Furthermore, as SoS have greater reaching influence in the wider environment within which they operate (Leveson, 2004), their responsibility expands to other organisations, the public and the natural environment.

Responsibility for RtL has changed significantly in response to the Haddon-Cave report (Haddon-Cave 2009), moving from organisational to individual level. This means that a single person (DDH, ODH, or SDH) is accountable for RtL and in the event of a fatality or accident the responsible DH will be answerable to the Coroner. SMEs in the RN first recognised the Rad Alt issue with the Hawk – this was discussed in relation to an early version of the Risk Register. The issue was formally acknowledged and then referred to the DDH. Because the RAF had no need to operate the Hawk at very low level, it was the responsibility of the DH chain within the RN to identify and control the risk.

4.5. Continual change and culture

SoS are subject to continual change driven by technological developments, new policies and regulations, and natural 'drifts' in procedure (challenge 8), which all need to be carefully monitored, understood and managed. It is no longer adequate to manage the safety issues affecting current implementations of the SoS; there needs to be a focus on through-life management of the SoS, which accounts for the increasing longevity of SoS and the legacy from previous SoS or components (challenge 9). Finally, performance will be influenced by, and in turn will influence, the culture (or climate) of a SoS, which relates to the shared perceptions and beliefs about an organisation held by members of that organisation (challenge 10).

4.5.1. Challenge 8: change

Change can manifest in a number of ways within SoS and can have different implications depending on how quickly it occurs. For example, technical changes occur quickly, commensurate with continual improvements in technology (Leveson, 2004; Rasmussen, 1997; Roed-Larsen and Stoop, 2012). Advances in technology continually alter the nature of work (Carayon, 2006), with an increasing shift away from 'human as operator' towards 'human as monitor', with increasing automation and supervisory control (Schutte, 1999; Endsley, 1996; Sheridan, 1992). Changes can also relate to procedures and regulations: these decisions are made quickly and sometimes immediately in response to an incident or near miss; however, the effects of the change can often take time to propagate through the SoS and behaviour change is reliant on all components of a SoS understanding and adapting to the change. Change can also occur as a consequence of small adaptations and variations in practice. These changes can cause a SoS to 'drift into failure', described by Dekker (2011, p.xii) as:

'a gradual, incremental decline into disaster driven by environmental pressure, unruly technology and social processes that normalize growing risk'.

Small changes, often driven by a push toward cost-effectiveness, combine to produce a systematic migration of behaviour (Rasmussen, 1997) and often propagate through a SoS to become large changes which could not have been predicted (Alberts, 2011). With a vast number of widely distributed interacting components in an organisation, small 'drifts' in procedure or policy will not necessarily be identified as risks to the safety of the SoS (Leveson, 2004; Dekker et al., 2011), however, it is precisely this apparent lack of significance that leads to unanticipated events.

The Hawk SoS has not been significantly affected by the introduction of new technologies; rather, it is the absence of a technology (Rad Alt) that has been a contributory factor to the RtL assessment. Large changes have, however, been brought about through policy modifications as a result of significant incidents in the military aviation world and by the changing profile of Hawk pilots, as described in previous sections. Drifts are much harder to identify in SoS than in smaller-scale systems, and this is the case for the Hawk case study. It is likely that drifts will have occurred throughout the Hawk's operational lifetime, although it will have been difficult for people involved in the SoS to consistently and adequately identify the nature and consequences of these.

4.5.2. Challenge 9: legacy (and longevity)

In SoS, many of the components will have long life-spans and will need to adapt to the continuous changes within the SoS over time. A SoS is likely to look significantly different after ten/twenty/thirty years of operation compared to what it did at inception. Planning for this through-life capability must start at the very earliest stages but should continue throughout the entire lifetime of the SoS and of the individual components that it comprises (Oxenham, 2010; Jamshidi, 2011). A related characteristic of SoS is legacy, which describes a common situation in which no organisation will ever really start from scratch; there will always be 'hangovers' from previous systems and networks (see Stanton et al., 2009 for an example of this). This is particularly apparent in SoS, in which there may be numerous legacy systems from the components which come together to form the SoS (Boardman and Sauser, 2006).

The mark of Hawk used for this tasking is a 'legacy' aircraft and was never designed to fly at very low level and for this reason the Hawk was not installed with Rad Alt. This decision would have been justified at the time of design as low level flight was done by the Hunter aircraft. However, this decision did not account for future changes to RN operations, including the retirement of the Hunter in the 1990s. It is not only the design of technical components that must account for the longevity of a SoS; the human components must also be acknowledged, for example, in terms of job design. RN personnel are generally in post for two years and it is estimated that personnel will require six months at the beginning of the post for familiarisation and learning, and six months at the end for transfer of knowledge to the next post-holder. This leaves only 12 to 18 months for fully focussed work, which is arguably too short to allow the development of sufficient knowledge and experience. This work structure does not support the longevity of the SoS, which may be in operation for decades and therefore requires a high level of continuity between successive personnel, particularly at higher management levels.

4.5.3. Challenge 10: culture (and climate)

The concepts of 'safety climate' and 'safety culture' are closely related to safety management and appeared in the research literature following the 1986 Chernobyl nuclear power plant accident (Flin, 2007). Flin (2007) suggested that the concept of culture is generally more complex and manifested at a deeper level than climate; however, the distinction is beyond the scope of this paper and it is sufficient to say that the terms cover both individual and group perceptions of and attitudes towards the operation and management of an organisation or SoS (Flin, 2007; Mearns et al., 2013). This, to a large extent, determines how well safety is managed within a SoS because it is up to the people within a SoS to employ 'safe' behaviours (Dekker, 2002) and their perceptions of the organisation will determine what these behaviour are (Zohar, 2010; Fernández-Mũniz et al., 2007). It is difficult to measure how people's perceptions of a SoS's safety climate impact on the actual safety of work (Flin, 2007), for example, this will be affected by the way in which safety management is seen to be prioritised against other performance management processes within a SoS, such as those associated with efficiency or productivity (Zohar, 2010).

The Nimrod XV230 accident effected a significant culture change within the MoD. Prior to this accident the MoD accepted a certain level of risk within military operations, but the Nimrod crash was considered to have gone far beyond this. However, no individual was officially responsible for holding the Nimrod risk: rather, it was held at an organisational level. The Haddon-Cave report addressed this issue, advising the creation of the MAA, which then produced regulations to shift the official ownership of risk from organisation to individual. Other, less obvious, shifts in culture and climate are likely to take place in any SoS, prompted by individual incidents (such as a fatality) or by high level effects (such as a change in Government or engagement in a new conflict). Culture is incredibly difficult to characterise within a SoS but misunderstanding the culture of a SoS can have serious consequences for management. The military is known for having a 'can do' culture (Stanton et al., 2009), which might lead to the assumption that issues are being successfully dealt with; however, confidence in ability would be unfounded if certain SoS components are not buying into this resourceful approach. This type of culture can also lead to non-reporting of issues, if they are believed to have been successfully dealt with internally. Furthermore, reporting issues may be seen as a weakness and subgroups will be reluctant to highlight deficiencies, particularly if they perceive themselves to be at fault in some way.

5. Conclusions

The aim of this paper was to characterise the key safety challenges of a SoS, based on the current academic definition of SoS. A military SoS was studied in detail and described to exemplify the ten key challenges.

Changes in the RtL assessment for the Hawk was dependent on human (pilot skill), technical (lack of Rad-Alt) and organisational (formation of MAA) factors; furthermore, it was the compound effects of the actions of these different components which influenced RtL, signifying the complexity characteristic of SoS. However, the very nature of complexity means that it is difficult to capture in diagrammatic form (i.e. the Accimap in Fig. 1) or even in a description, and there will be some effects that have not been captured here and will only be known to certain individuals or subsystems within a SoS. Each subsystem in a SoS has a clear boundary, over which information is transferred.

Identification of potential RtL issues in the Hawk SoS is dependent on the sharing of information across multiple boundaries by SMEs. There is an inherent trust in the ability of the SMEs, which is indicative of the military's reliance on the capability of individuals. This trust is ensured to some extent by the assignment of responsibility at an individual level. There is a belief that if an

individual is held personally responsible for a risk, they will do everything within their power to mitigate the risk. The assumption is that this approach ensures more rigorous safety management than the old approach of assigning risk at an organisational level, where accountability was more difficult to ascribe. However, it could be argued that this goes against the Human Factors view and the modern Systems Approach to risk and error, which state that error is a consequence of systemic factors rather than individual shortcomings (Leveson, 2004; Carayon, 2006; Dekker, 2011; Perrow, 1999; Reason, 1997; Salmon et al., 2012; Stanton et al., 2012; Wilson, 2012). For example, Leveson (2004) suggested that the increasing complexity and interconnectedness of networks is actually shifting responsibility away from the individual, toward the organisational level. This is the model that was in place at the time of the Nimrod accident and it seems that the Military has swung back away from this approach as a consequence. There are advantages and disadvantages of both the individualistic and organisational models for risk management. At the organisational level, the complexities of the entire SoS may be more obvious and therefore managed more effectively; however, this can also breed a culture in which no-one is willing to accept responsibility for risk and blame is always shifted to a higher level in the network. This leads to assumptions by individual subsystems that they are not accountable for their own performance. Individual accountability for risk is intended to ensure that SoS components are dedicating sufficient effort to mitigation (i.e. in the Hawk example); however, a disadvantage of the individualistic view is that unsafe acts could be isolated from the wider system context and that latent conditions within this context are therefore not addressed in safety management (Reason, 2000).

Emergent behaviour, change, legacy, culture, and complexity to a certain extent, are less tangible aspects of SoS, and this makes them more difficult to define, understand, and consequently manage. For example, RtL assessments take explicit account of recent incidents and changes to policies at various levels but they are also influenced by implicit assumptions held by assessors, such as predicted changes in Government, predicted lifespan of technical components, and national/international economic climates. In many cases, assessors can only guess at the effects of these factors and opinions will vary between individuals. It is virtually impossible to set boundaries for the factors to consider in a RtL assessment, so again this comes down to trust in the skill and experience of the SMEs involved in assessment.

This paper has presented ten challenges for SoS, based on a thematic analysis of the literature. These are intended to further our understanding of what is meant by the term SoS and how we can manage the increasingly complex nature of large networks of multiple socio-technical components. A military case study was selected to exemplify these challenges and all were present within the Hawk Jet missile simulation SoS. Some challenges are obvious and explicit, whereas some are less tangible and it is the latter that will be more difficult to identify, understand and manage.

Acknowledgements

The authors would like to thank Commander Neil Bing of Air Cap SO1 Lightning, RAF High Wycombe, for his account of the Hawk Risk to Life case study and his very valuable insight into the challenges of this SoS.

References

Alberts, D.S., 2011. The agility advantage: a survival guide for complex enterprises and endeavours, first ed. Department of Defence Command and Control Research Program (CCRP), Washington, DC, USA.

Alexander, R., Hall-May, M., Kelly, T., 2004. Characterisation of systems of systems failures. In: The 22nd International System Safety Conference, Providence, RI, USA, 2–6 August, 2004. System Safety Society Publications.

Alexander, R., Kelly, T., 2013. Supporting systems of systems hazard analysis using multi-agent simulation. Saf. Sci. 51, 302–318.

Benn, J., Burnett, S., Parand, A., Pinto, A., Iskander, S., Vincent, C., 2009. Studying large-scale programmes to improve patient safety in whole care systems: challenges for research. Soc. Sci. Med. 69, 1767–1776.

Boardman, J., Sauser, B., 2006. System of systems – the meaning of of. In: IEEE/SMC International Conference on System of Systems Engineering, Los Angeles, CA, USA, April 2006. IEEE.

Carayon, P., 2006. Human factors of sociotechnical systems. Appl. Ergon. 37, 525–535.

Dekker, S., 2011. Drift into failure: from hunting broken components to understanding complex systems, first ed. Ashgate, Surrey, UK.

Dekker, S., Cilliers, P., Hofmeyr, J.-H., 2011. The complexity of failure: implications of complexity theory for safety investigations. Saf. Sci. 49, 939–945.

Dekker, S.W.A., 2002. Reconstructing human contributions to accidents: the new view on error and performance. J. Saf. Res. 33, 371–385.

Endsley, M.R., 1996. Automation and situation awareness. In: Parasuraman, R., Mouloua, M. (Eds.), Automation and Human Performance: Theory and Applications, 1st ed. Lawrence Erlbaum, Mahwah, NJ, USA.

Fernández-Mũniz, B., Montes-Peón, J.M., Vázquez-Ordáz, C.J., 2007. Safety management system: development and validation of a multidimensional scale. J. Loss Prevent. Process Ind. 20, 52–68.

Flin, R., 2007. Measuring safety culture in healthcare: a case for accurate diagnosis. Saf. Sci. 45, 653–667.

H M Government, 2010. Securing Britain in an Age of Uncertainty: the Strategic Defence and Security Review. The Stationery Office, London.

Haddon-Cave, C., 2009. The Nimrod review. An independent review into broader issues surrounding the loss of the RAF Nimrod MR2 aircraft XV230 in Afghanistan in 2006. The Stationery Office, London.

Harris, D., Stanton, N.A., 2010. Aviation as a system of systems: preface to the special issue of human factors in aviation. Ergonomics 53, 145–148.

Hollnagel, E., 2012. Coping with complexity: past, present and future. Cogn. Technol. Work 14, 199–205.

Jamshidi, M., 2011. Introduction to system of systems. In: Jamshidi, M. (Ed.), System of Systems Engineering: Innovations for the Twenty-First Century, 1st ed. Wiley and Sons, Hoboken, NJ, USA.

Jaques, E., 1951. The changing culture of a factory, first ed. Tavistock Publications, London.

Jones, P.M., 1995. Designing for operations: towards a sociotechnical systems and cognitive engineering approach to concurrent engineering. Int. J. Ind. Ergon. 16, 283–292.

Kirwan, B., 2001. Coping with accelerating socio-technical systems. Saf. Sci. 37, 77–107.

Knudsen Tveiten, C., Albrechtsen, E., Wærø, I., Wahl, A.M., 2012. Building resilience into emergency management. Saf. Sci. 50, 1960–1966.

Leveson, N., 2004. A new accident model for engineering safer systems. Saf. Sci. 42, 237–270.

MAA, 2010. RA 1210: Management of Operating Risk (Risk to Life). Military Aviation Authority, London.

MAA, 2011. Regulatory Instruction MAA RI/02/11 (DG) – Air Safety: Risk Management. MAA, London.

Maier, M.W., 1998. Architecting principles for systems-of-systems. Syst. Eng. 1, 267–284.

Mearns, K., Kirwan, B., Reader, T.W., Jackson, J., Kennedy, R., Gordon, R., 2013. Development of a methodology for understanding and enhancing safety culture in air traffic management. Saf. Sci. 53, 123–133.

Ministry of Defence (MoD), 2007. Defence standard 00-56, issue 4, Part 2. London.

Oxenham, D., 2010. The next great challenge in systems thinking: a defence perspective. Civ. Eng. Environ. Syst. 27, 231–241.

Perrow, C., 1999. Normal Accidents: Living with High-Risk Technologies, second ed. Princeton University Press, Princeton, NJ, USA.

QINETIQ, 2009. Joint Enablers for the Realisation of Network Enabled Capability (JERNEC) TIN 005. NEC systems of systems safety/risk/reliability – final report, London.

Qureshi, Z.H., Ashraf, M.A., Amer, Y., 2007. Modeling industrial safety: a sociotechnical systems perspective. In: The IEEE International Conference on Industrial Engineering and Engineering Management, Singapore, 2–5 December, 2007. IEEE.

Rasmussen, J., 1997. Risk management in a dynamic society: a modelling problem. Saf. Sci. 27, 183–213.

Reason, J., 1997. Managing the Risks of Organizational Accidents, first ed. Ashgate, Aldershot, UK.

Reason, J., 2000. Human error: models and management. Br. Med. J. 320, 768–770.

Reiman, T., Pietikäinen, E., 2012. Leading indicators of system safety – monitoring and driving the organizational safety potential. Saf. Sci. 50, 1993–2000.

Rendon, R.G., Huynh, T.V., Osmundson, J.S., 2012. Contracting processes and structures for systems-of-systems acquisition. Syst. Eng. 15, 471–482.

Robinson, G.H., 1982. Accidents and sociotechnical systems: principles for design. Accid. Anal. Prevent. 14, 121–130.

Roed-Larsen, S., Stoop, J., 2012. Modern accident investigation – four major challenges. Saf. Sci. 50, 1392–1397.

Royal Navy. 2012. *Fleet requirements air direction unit* (*FRADU*) [Online]. Royal Navy, <http://www.royalnavy.mod.uk/sitecore/content/home/the-fleet/air-stations/rnas-culdrose/fleet-requirements-air-direction-unit-fradu> 2013 (accessed 11.01.13).

Salmon, P.M., McClure, R., Stanton, N.A., 2012. Road transport in drift? Applying contemporary systems thinking to road safety. Saf. Sci. 50, 1829–1838.
Schutte, P., 1999. Complemation: an alternative to automation. J. Inform. Technol. Impact 1, 113–118.
Sheridan, T.B., 1992. Telerobotics, Automation, and Human Supervisory Control, first ed. Massachusetts Institute of Technology (MIT), Cambridge, MA, USA.
Sinclair, M.A., 2007. Ergonomics issues in future systems. Ergonomics 50, 1957–1986.
Stanton, N.A., Jenkins, D.P., Salmon, P.M., Walker, G.H., Revell, K.M.A., Rafferty, L.A., 2009. Digitising Command and Control: A Human Factors and Ergonomics Analysis of Mission Planning and Battlespace Management, first ed. Ashgate, Aldershot.
Stanton, N.A., Rafferty, L.A., Blane, A., 2012. Human factors analysis of accidents in systems of systems. J. Battlefield Technol. 15, 23–30.
Trist, E., 1981. The evolution of socio-technical systems: a conceptual framework and an action research program. In: Van de Ven, A., Joyce, W. (Eds.), Perspectives on Organizational Design and Behaviour. Wiley Interscience, New York.
Trist, E.L., Bamforth, K.W., 1951. Some social and psychological consequences of the longwall method of coal-getting: an examination of the psychological situation and defences of a work group in relation to the social structure and technological content of the work system. Hum. Relat. 4, 3–38.
Voirin, M., Pierlot, S., Llory, M., 2012. Availability organisational analysis: is it a hazard for safety? Saf. Sci. 50, 1438–1444.
Von Bertalanffy, L., 1950. An outline of general system theory. Br. J. Philos. Sci. 1, 134–165.
Walker, G.H., Stanton, N.A., Salmon, P.M., Jenkins, D.P., 2008. A review of sociotechnical systems theory: a classic concept for command and control paradigms. Theor. Issues Ergon. Sci. 9, 479–499.
WANO, 2011. Performance indicators, 2011. London, UK.
Wilson, J.R., 2012. Fundamentals of systems ergonomics. Work 14, 3861–3868.
Zohar, D., 2010. Thirty years of safety climate research: reflections and future directions. Accid. Anal. Prevent. 42, 1517–1522.