Created on 7/14/2003 3:47 PM

White Paper on System of Systems Measurement

Executive Overview:

In recent years the DoD has leveraged the rapid evolution of information technology to provide overmatching combat capability to the warfighter. Continuing this trend the DoD is investing in large-scale initiatives such as Future Combat System, National Missile Defense, Army Battlefield Digitization, Navy Cooperative Engagement Capability, and the Global Command and Control System. These efforts, like many service-specific and Joint acquisition efforts, represent a new commodity class in the acquisition domain: they are “systems of systems” (SoS). The emerging importance of these aggregated systems, both in terms of the investment resources allocated to them, and the operational value of the capability they provide, has prompted DoD leadership to seek improvement in the management and oversight of such endeavors. However, established acquisition management concepts, tools, and methods, may not adequately address or account for important differences between management at the system versus system-of-systems level. For example, project size, as expressed, traditionally in terms of source lines of code or similar measures, has long been validated as a key predictor of project cost and schedule for software-intensive systems. No corresponding measure has proven to be similarly reliable at the system-of-system level. For this reason and others, there is no reliable and validated mechanism to allow DoD decision makers to assess the likely cost, schedule or technical success of these large, aggregated systems of systems.

The development of Systems of Systems requires investment in the development of the individual entities within the SoS. Traditional management methodologies are generally well suited to this part of the problem. But SoS implementation also requires investment in the integration of the individual entities within the SoS context. Furthermore, attention is required to the development of enterprise rules that govern the interaction of all the system elements. Each of these areas: system elements, system integration, and enterprise rules, require the expenditure of effort for the implementation and maintenance of the SoS, but the effort expended often falls outside or in-between conventional project-oriented management structures. The ability to predict the amount of effort required to achieve and sustain the desired SoS functionality is critical to the success of the SoS endeavor. In order to predict the effort, relevant attributes of the system elements, system interfaces, and enterprise rules must be defined, measured, and related to the effort necessary to achieve the desired outcomes.

However, estimating the magnitude of the undertaking is only part of the problem. The efficiency and effectiveness of the process by which the SoS is implemented also has an effect on the outcomes of cost and schedule. Process efficiency and effectiveness can be influenced by decisions made by the SoS implementer, but are also influenced by the environment within which the SoS is implemented.

Consequently, in order to predict the likely outcomes of SoS implementations, and to effectively manage the implementation process, new measurement constructs must be defined that address the element, interface, enterprise rules, process and environmental attributes which drive or influence SoS implementation outcomes.

1 Measuring Systems of Systems

1.1 Introduction

Although the notion of systems of systems is relatively recent, there is already accumulated experience to suggest that such undertakings often take significantly longer, cost significantly more, and present significantly greater challenges than is typically anticipated. In examining the literature, it appears that the issue stems from a general inability to predict the magnitude of the effort necessary to implement the SoS construct, and an inability to impose effective and efficient management processes to implement them. The theme running through these observations is that the acquisition community has had difficulty adapting the measurement and management constructs that have been established in the project-centric domain to the SoS domain. Though the information needs are still relevant, measurement constructs in the SoS domain are sufficiently different that new constructs and measures need to be developed, validated, and put into practice.

The implementation of Systems of Systems requires investment in the development and/or acquisition of the individual entities within the SoS. It also requires investment in the integration of these entities within the SoS context. Furthermore, enterprise rules must be developed and imposed to govern the interaction of the system elements, and the elements must be adapted to abide by these rules. Each of these areas: system elements, system integration, and enterprise rules, require the expenditure of effort on the part of the SoS architect and development collaborators. The magnitude of the effort required is driven primarily by the nature of the SoS to be implemented. But in addition to the inherent magnitude of the SoS, also important is the efficiency of the processes used to implement the SoS, as well as environmental conditions that affect the efficiency of the implementation process. Even with an understanding of the magnitude of the SoS, without efficient and effective management and implementation processes, the cost and schedule of the undertaking will be larger than anticipated. Efficient processes and environmental conditions which are conducive to permitting those processes to operate efficiently should have attributes that distinguish them from inefficient processes and unfavorable environmental conditions. These represent important measurement constructs that should be incorporated into the SoS measurement plan.

Understanding how element, interface, enterprise, process and environmental attributes influence outcomes of cost and schedule is necessary to predict these outcomes, to properly allocate resources, and to establish appropriate expectations. The ability to predict the amount of effort required to achieve the desired SoS functionality is critical to the success of the SoS endeavor. In order to predict the effort, relevant attributes of the system elements, system interfaces, and enterprise rules must be defined, measured, and related to the effort necessary to achieve the desired outcomes.

1.1.1 What is a System of Systems?

While “system of systems” is an appealing term that conveys an intuitive meaning, that meaning, if not explicitly defined, can vary from individual to individual, and can be poorly distinguished from other related concepts. To communicate effectively, the

meaning of SoS must be clearly defined as to what SoS is, and what it is not. This is necessary to define clear and unambiguous measurement constructs.
First, we must define the term “system”. ISO/IEC Standard 15288 “Systems Engineering-System Life Cycle Processes”(**insert date***), defines a system as “a combination of interacting elements organized to achieve one or more stated purposes”[1]. This definition alone is insufficient to convey clearly and unambiguously, since a “system” thus defined can apply at any level of aggregation, from the smallest subcomponent, to the largest aggregation of major systems.

Thus, the term “system of interest” is important. ISO/IEC Standard 15288 defines the “system of interest” simply as that system whose lifecycle is under consideration[2]. The “system of interest” is a means of specifying the level of aggregation or detail at which the discussion or analysis of a system is taking place.

In common usage, “systems” are comprised of “subsystems” which are then comprised of “components” which are further comprised of “units”. A “unit” is typically the smallest element which is independently tracked and managed. This ideal construct serves a useful purpose from a theoretical perspective. However, as systems have grown to incorporate multiple systems of even greater scope, terms like, “sub-system”, “component”, and “unit” become ambiguous. To make the notion of the “system” more generic, and less restricted in scope, the ISO/IEC-STD-15288 uses the term “system element” to describe the elements from which the system of interest is constructed. Figure 1 depicts the concept that depending upon the definition of “system of interest”, system elements can be systems in their own right, sub-systems, components, or units.

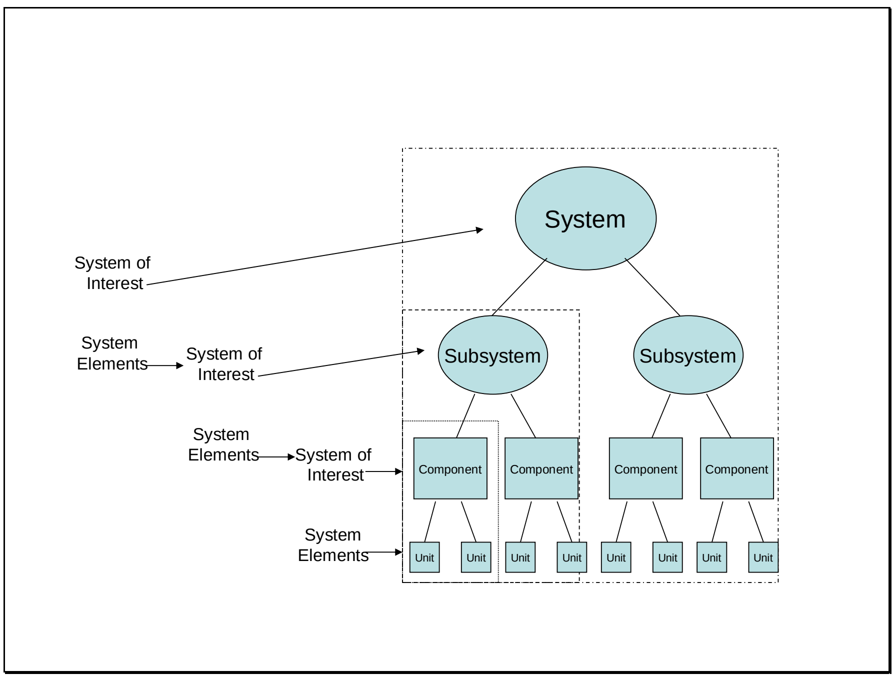

Figure 1: System of Interest Construct

[1] ISO/IEC Standard 15288 “Systems Engineering-System Life Cycle Processes”(**insert date***), ***insert page***
[2] Ibid. ***insert page***

The elements of a SoS are systems that provide functionality independent of the SoS, but when integrated, provide a capability not available when the system elements are disjoint. An important and defining characteristic of SoS is that the elements of a SoS are called upon to interact with each other in order to provide the user-desired capability. This notion of interaction of elements to provide a specific capability distinguishes SoS from a related concept, Family of Systems (FoS). Families of systems are systems which share common characteristics or a common inheritance, such that they can be considered a distinguishable group or "family". Such characteristics may include type of functionality, or mission area supported (e.g., command, control, communications; surveillance and reconnaissance platforms). Family designations are dependent upon the classification scheme, and the characteristics of the system under consideration. Therefore the "family" a particular system belongs to will depend upon the view of the observer. The value of the FoS view is that by grouping systems with common functions, characteristics, or "heredity", enables efficient management of these resources through functional specialization and/or economies of scale. For example, each mechanized division does not independently develop its own tracked combat vehicle. These are centrally acquired, allocated and maintained (for depot-level maintenance), because economies are realized through managing this commodity as part of a family. While members of a FoS may be required to interoperate, they are not necessarily integrated for the specific purpose of providing an objective capability, as a SoS would be. The similarities and distinctions of FOS and SoS are depicted in Figure 2.

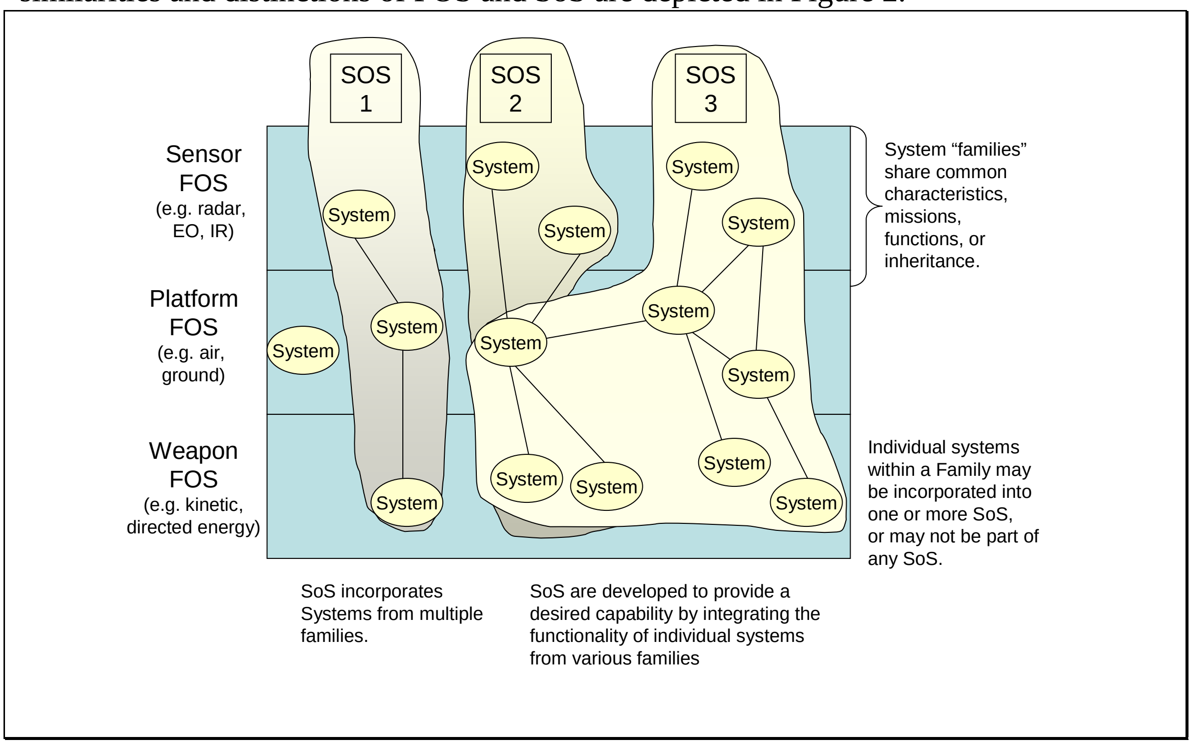

Figure 2: SoS vs FOS

SoS typically incorporate elements from different families (i.e., radar system, ISR platform, communications, C2, weapons platform, weapon) to achieve a specific operational capability (in this case, a "sensor-to-shooter" SoS). Also, individual systems may be incorporated into different SoS (for example, GPS is incorporated into many different SoS, from various sensor-to-shooter SoS (e.g., Future Combat Systems), to ISR SoS (e.g., Global Hawk), to logistic supply SoS (e.g, Global Transportation Network).

It is important to distinguish the concepts of "functionality" and "capability" when discussing SoS and FoS. "Functionality" describes the functions performed by a given system element. Common functionality is a characteristic of members of a Family of systems. "Capability" is functionality applied to some operationally-relevant purpose. The purpose of an SoS is to provide an operationally-relevant capability not achievable by a single system or family. For example tracked and wheeled combat vehicles within the family of combat vehicles, have functional characteristics of mobility and lethality. When integrated with systems from other families (e.g., infantry, artillery, airborne vehicles, sensors, C2, etc.) into a SoS, these elements provide that functionality in the operationally-relevant context (capability) of dominant maneuver. Thus, the notion of "capability-based acquisition" is synonymous with the creation of SoS, in that it integrates the concept of "functionality" realized at the system element level and organized within families, with the operationally-relevant context achieved by integrating functional (family) elements at the SoS level.

The Chairman of the Joint Chiefs of Staff Instruction 3170.01C (***insert date***) defines Family of Systems (FOS) and System of Systems (SoS) as follows:

<u>Family of Systems (FoS)</u> - A set or arrangement of independent systems that can be arranged or interconnected in various ways to provide different capabilities. The mix of systems can be tailored to provide desired capabilities, dependent on the situation.[3]

<u>System of Systems (SoS)</u> - A set or arrangement of interdependent systems that are related or connected to provide a given capability. The loss of any part of the system will degrade the performance or capabilities of the whole.[4]

Although the CJCSI 3170 is an authoritative document, the researchers feel the definitions of SoS and FOS incorporated into the 3170 are lacking for the following reasons:

1. The CJCSI makes no essential distinction between FOS and SOS in terms of their purpose. According to the CJCSI definition they both provide capabilities. This misses the opportunity to make an intuitive and helpful distinction between classifying systems according to their characteristics, functions, and inheritance, and classifying systems according to the operational capability they provide. This lack of clear distinction between the terms makes them essentially equivalent.

2. The distinction made, between independent and inter-dependent systems has no objective meaning. Any system, depending upon the assumptions used, could be considered independent or interdependent. Again, this makes FoS and SoS essentially equivalent, per the CJCSI definitions.

[3] CJCSI 3170.01C signed ****insert date****

[4] CJCSI 3170.01C signed ***insert date***

3. The notion of "tailoring" a family of system to adapt its capabilities, dependent upon the situation could apply to a flexible, responsive SoS as well.

As a result, the researchers will distinguish FOS and SoS in the manner described in the preceding section.

Elements of a SoS interact via interfaces (Figure 3). These interfaces establish the rules and pathways for communication between the interacting elements. The establishment of interfaces to permit interaction among the SoS elements is called "integration". The SoS will not function as an entity until the elements are integrated.

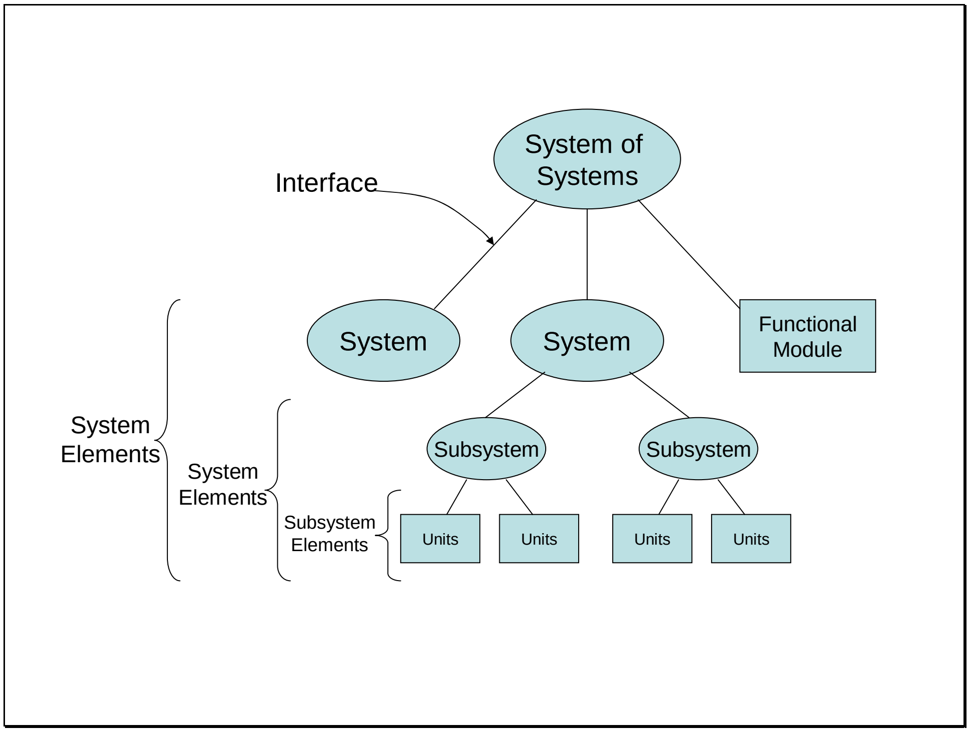

Figure 3: System of Systems

1.1.2 Terms of Reference

This research project establishes terms of reference to be used within the context of the project as described in the following section. Although it is the desire of the researchers to adopt standard terminology whenever possible, it may be necessary to apply terminology which is not standardized across all domains. In these cases, the researchers will describe the terms and their usage as they apply to this project.

Key terms of reference are as follows:

- **System: a** combination of interacting elements organized to achieve one or more stated purposes (from 15288)
- **System-of-Interest:** the system whose life cycle is under consideration (adopted from 15288)

- **System Element:** a member of a set of elements that constitutes a system
  - NOTE: A system element is a discrete part of a system that can be implemented to fulfill specified requirements (from Valerdi)
- **Enterprise rules:** rules that govern the interaction of system elements; serve the role of "local government" within the SoS
- **System of Systems:** A combination of interacting functional system elements, which are themselves systems, organized to achieve a stated operational capability
- **Family of Systems:** A classification of functional system elements based upon common system characteristics, which may include function or inheritance.
- **Capability:** The capacity to be used, treated, or developed for a particular purpose. Integration and application of functional elements to some operationally-relevant purpose.
- **Functionality:** The ability to perform the activities for which the system is specifically fitted or employed.
- **Interface:** A point at which independent systems or diverse groups interact. The device or system by which interaction at an interface is effected.[5]
- **Integration:** The establishment and maintenance of a functional interface between any two system elements/components.
- **Outcomes:** Objective output or consequences of the SoS implementation process in terms of cost, schedule, performance or other.
- **Attribute:** Observable characteristic of an entity which describes an essential quality of that entity, such as size, complexity, identity, heredity, etc.

1.2 Current SoS Measurement Capabilities & Shortfalls

The disciplines of system engineering and project management have historically focused upon the "system" as the entity of interest. This systems focus pervades the Department of Defense (DoD) and federal government acquisition processes. For example, the DoD's method for planning and programming resources for investment in new technologies is based upon the "program element" structure; the acquisition regulations typically address "programs" as the unit of interest; reporting to oversight bodies and the Congress are typically at the program level. Accordingly, the measurement constructs and measures commonly used are at the system, project, and program levels.

Although this system, project, and program-level focus has served well up to this point, these acquisition processes may not be ideally suited for system-of-systems analysis and management. Problems occur when the scope of the SoS extends beyond established management boundaries. In this case conflicting lines of authority can make the management of SoS particularly difficult. With systems whose elements have their origins in distinct functional disciplines (such as distinct military services in Joint projects), there is often no single entity whose responsibility it is to see that the interfaces between elements are properly designed and implemented, or that resources are properly allocated and balanced among the constituent elements. More subtle, but no less critical, there may be no central authority to adjudicate differences in enterprise rules from one

[5] Webster's II New Riverside Dictionary, Copyright 1988 by Houghton Mifflin Company.

functional discipline to another, which can lead to data incompatibility, semantic and syntactic disconnects. Another manifestation is the lack of management attention or resources allocated to interface management, or to the process of validating and testing interoperability across the elements of the SoS. Even the relatively straightforward process of schedule and progress measurement breaks down if inconsistent methods of work unit progress tracking are applied across the constituent elements of an SoS, thereby precluding meaningful aggregation of data at the SoS level.

1.3 Information Needs

At the simplest level, the information needs at the SoS are the same as those at the project level. Managers need to understand the status of the SoS implementation in terms of schedule and progress, resources and cost, product size and stability, product quality, process performance, technology effectiveness, and customer satisfaction. However, the manner in which these information needs map to measurable concepts and measures is somewhat different in the SoS context.

Take, for example, the “product quality” category, with the measurable concepts of functional correctness. In the development of an individual system, the systems engineering process decomposes requirements to greater and greater levels of specificity and detail, until the smallest independently manageable unit of functionality is specified. These units are then designed, tested, and integrated into ever larger, more complex subsystems, until the fully-developed system is tested against criteria developed from the original requirement specification. As is often the case, lapses in the quality attribute of functional correctness at the interface between units will manifest themselves during integration testing. These likelihood and impact of these defects can be reduced by implementing controls and measures that focus attention on the functional correctness of system/subsystem interface specification and design.

In the SoS context, however, the system elements are often already developed by some entity outside the immediate control of the SoS implementer. Instead of managing the quality of the interfaces by measuring functional correctness of the design, the implementer must first discover the design of the interface through engineering analysis. The effort becomes less one of specification and design, and becomes an exercise in reverse-engineering and design discovery. The scope and effectiveness of the reverse-engineering and discovery process can be influenced by factors that are different from those in the single-domain systems engineering context. For example, organizational relationships between the individual system element developers might influence the availability of interface specification and design information. Similarly, the compatibility of the enterprise rules under which the system element was developed might influence the effort required to implement the interface. These factors will give rise to new measurement constructs and measures.

As this example demonstrates, although SoS information needs will be largely the same, the measurement constructs and measures may be different than for an individual system.

1.4 Recommended Measures for SoS

Table 2 identifies measurement categories of schedule and progress, resources and cost, product size and stability, product quality, process performance, technology effectiveness, and customer satisfaction. As the preceding paragraph discussed, these categories reflect potential information needs which should be similar at the SoS level and the system element level, but may require different interpretation and implementation as measurement constructs and measures.

As with any complex undertaking which requires the cooperation and interaction of many elements it is important to understand how the various elements operate and influence one another, in order to make reasoned decisions regarding what to measure, what the measures mean, and how to make decisions based upon the data the measures provide. The Defense Acquisition University has initiated a study of SoS to determine the fundamental drivers of cost, schedule and performance outcomes. As part of their study, the researchers have endeavored to describe the factors that drive the observed SoS outcomes. An important component of that endeavor is to identify measurement constructs that support information needs of SoS implementers and the oversight processes of the DoD. Accordingly, it may be useful to use the DAU SoS taxonomy as a framework for discussing the measurement constructs and measures for SoS.

As depicted in Figure 4, the underlying principle of SoS measurement is that the outcomes that tend get a lot of attention in the management of things, namely cost, schedule, and performance, arise from the effort expended to implement them. This is not a revelation, but it is perhaps the only relationship that we know (at this point) is causal. This simplistic observation serves to keep us focused on the fact that anything that causes effort to be expended will drive cost.

Suppose effort is decomposed into two components: effort which is driven by the magnitude of the task, and the effort which is driven by efficiency of the process by which the task is accomplished. For brevity, we will refer to that effort driven by the magnitude of the task as “inherent effort”, and that effort driven by the efficiency of the implementation process as “induced effort”. These represent distinct measurable concepts; one associated the “product”, and the other associated with the “process”.

The notion of inherent effort derives from the observation that when comparing two tasks, certain distinguishing characteristics (attributes) may lead one to believe that whatever the circumstances of the task execution, one task may require greater effort than the other. In the product development case, these characteristics are typically related to the size and complexity of the product to be built. Larger, more complex tasks will invariably require more effort than small, simple tasks.

Induced effort (for lack of a better term) refers to the effort incurred by the implementing process above that which would have been expended under ideal circumstances, using optimal methods. Stated another way, induced effort is “wasted” effort incurred due to inefficiencies of the process, diseconomies of scale, or adverse environmental conditions.

Why is it important to distinguish between “inherent” and “induced” effort? It is a way to separate the effort related to the “product” from that related to the “process”. This is important because while both product and process attributes have an impact on total effort, and therefore total cost, the drivers of process-related effort are different from those of product-related effort, and the management controls for each are different.

Failure to distinguish between the product and process related effects will blunt a manager's ability to trace problems to their source and develop appropriate interventions.

Focusing first on the "product" side of the equation, we assert that the inherent effort of the SoS task can be inferred or derived from observable attributes of the SoS to be implemented. The observable attributes that describe the inherent magnitude of the SoS effort fall into three categories: attributes related to SoS elements, attributes related to SoS interfaces (internal and external), and attributes related to the enterprise rules that govern the behavior of the elements within the SoS. In assessing the inherent effort of a system, we note that the influence of integration needs to be specifically addressed. This is particularly the case when system elements have been developed without the expectation of their incorporation into the target SoS.

The effect of enterprise rules is particularly important in SoS, and generally not of any significance in the single-domain case. Enterprise rules cover issues like data schemas, network balancing protocols, timing standards, message formatting, etc. Within the single-domain case, where the system is developed for a particular purpose, the enterprise rules are incorporated, explicitly and implicitly in the requirements, specification and design of the system. However, when different systems, developed under their own enterprise rules are integrated, the mismatch between these enterprise rules can be problematic. In order to understand the cost of creating interoperability among the systems of SoS, an understanding of the degree of compatibility between the enterprise rules of the system elements and the SoS must be attained. This will give rise to a new class of measures.

So the information needs of product size and stability, as well as resources and cost will map to measurement constructs related to the SoS elements, interfaces, and the enterprise rules that impact both the elements and the interfaces.

The efficiency of the SoS implementation process determines how much of available resources will be consumed in the construction of an SoS of a given magnitude. This efficiency term is described as "induced" effort. In assessing the induced effort of a system, we think of "processes" and the attributes that moderate that process; attributes that induce the process to be more or less efficient in implementing the task. We note that depending upon the system of interest, some process moderators relate to choices that are within the management control of the implementer, whereas some "choices" are externally imposed. We therefore distinguish between process-related moderators (choices the implementer makes with regard to the implementation process), and environment-related moderators (externally-imposed conditions or constraints on the implementation process). We note that choices imposed by higher-level management, from the perspective of the subordinate manager, are "environmental", so care must be taken to specify the system of interest, and level of control, when attempting to distinguish "process-related" attributes from "environmental" attributes.

The attributes and the measures that describe them are discussed in detail in the following sections.

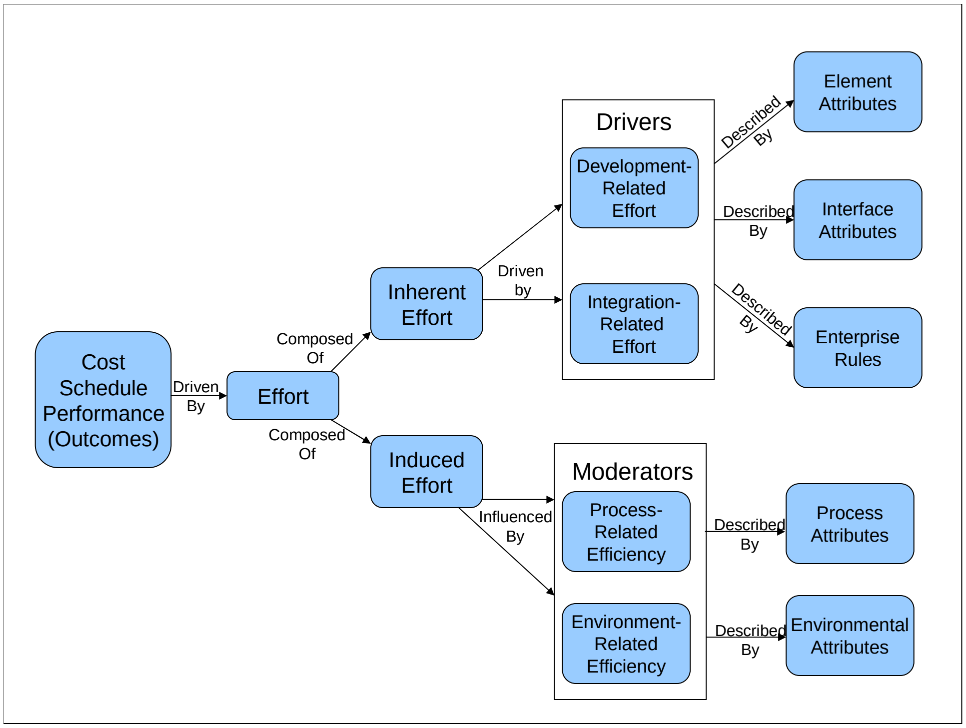

Figure 4: Research Concept Map

.

1.4.1 Element Attributes

SoS are comprised of system elements, which are predominantly systems in their own right. Attributes of these system elements will drive the effort required to integrate them into the SoS, and to maintain SoS functionality over the lifecycle. Examples include:

- Number of System Elements: The number of system elements is determined by functional analysis and allocation within the systems engineering process. This allocation process is influenced by existing systems that provide some of the required functionality. As the implementer conducts "market analysis" and becomes aware of existing products that fulfill part or all of certain requirements, a "make or buy" decision is made. For each decision, different attributes influence how much effort will be required. For the "make" decision, traditional within-domain cost analysis techniques will typically apply. For the "buy" case, techniques for estimating COTS acquisition costs are more applicable. This will include the effort for discovering the detailed functionality of the product, determination of product suitability, to determine functional fit and coverage, and then effort must be expended in the adaptation of the COTS product to the specific implementation: the development of wrappers, middleware, glue code, custom application program interfaces (APIs), etc. Also included is the effort for training the workforce to use the COTS product, and the cost to upgrade and maintain the COTS product as it continues to evolve.

- System Element Design: The homogeneity of system element design, with respect to procedures, applications, infrastructure, and data[6] will determine the effort necessary to achieve a required level of interoperability.
- Degree of Reuse: The percentage of elements to be integrated versus built will have an impact on the effort required to develop the SoS. Knowing many of the system elements are COTS or NDI will have an impact on the magnitude of the SoS development/maintenance effort.

1.4.2 SoS Integration: Interface Attributes

Integration is the establishment and maintenance of a functional interface between any two system elements/components. Interface is a generic term that covers all interactions at all levels of abstraction. For example the Open Systems Interconnection (OSI)[7] reference model describes seven "layers" which must be described and agreed-to in order for systems or components to effectively interoperate: Note that there are multiple models in addition to the OSI, and that we're using the OSI model as an example to illuminate the complexity of these interface attributes.

- Application Layer: Provides the user interface.
- Presentation Layer: Translates from application format to network format.
- Session Layer: Establishes, maintains, and ends sessions across the network; provides synchronization by inserting checkpoints in the data stream.
- Transport Layer: manages the flow control of data between parties across the network; divides streams of data into chunks or packets; the transport layer of the receiving computer reassembles the message from packets
- Network Layer: Translates logical network address and names to their physical address; responsible for addressing; determining routes for sending; managing network problems such as packet switching, data congestion and routing
- Data Link Layer: Turns packets into raw bits, and at the receiving end turns bits into packets; handles data frames between the Network and Physical layers; responsible for error-free transfer of frames to other computer via the Physical Layer; defines the methods used to transmit and receive data on the network.
- Physical Layer: transmits raw bit stream over physical cable; defines cables, cards, and physical aspects[8]

At each layer of abstraction, from the physical to the application layer, effort is required to establish and maintain each interface between system elements/components. Typical activities that contribute to this effort includes:

[6] Levels of Information Systems Interoperability: Report of the OASD(C3I) C4ISR Architectures Working Group, 30 March, 1998.
[7] Insert OSI reference.
[8] Networking Essentials Notes: http://www.geocities.com/SiliconValley/Monitor/3131/ne/netoc.html

  - Management of interface control documents & standards
  - Establishment of communications protocols, waveforms, message formats, semantics & syntax
  - Development of application program interfaces (APIs)
- Interface Attributes: The number and diversity of interfaces will drive effort at the SoS level. Examples of interface attributes which may have an impact on the effort required to achieve SoS functionality include:
  - Number of Interfaces
    - Internal interfaces: A point at which independent systems within the SoS context interact. The device or system by which interaction at an interface with other SoS system elements is effected.
    - External interfaces: A point at which systems external to the SoS interact with SoS elements. The device or system by which interaction at an interface with external (non-SoS) elements is effected.
  - Nature of Interfaces
    - Complexity: Timing constraints, degree of coupling, etc., will have an impact on the difficulty, and therefore the effort expended to create and maintain a functional interface.
    - Volatility: The time-dependent evolution of requirements levied upon the interface by either system element it connects will drive the magnitude of the SoS maintenance effort. Functional modules which themselves rapidly evolve, and go through frequent update cycles will require, at a minimum, an engineering assessment of the affected interfaces. If interface configurations change as a result, additional engineering effort must be expended to modify the interface and assess the impact upon other functional elements.
    - Diversity: The number of unique interfaces which must be created and maintained. In well-structured architectures, this level of diversity is managed, so standard interface rules are applied across the SoS. This requires significant initial investment, but may pay dividends over the lifecycle of the SoS.
    - Political Consequence: Description of who controls and funds the changes. Are the "owners" of the interface/entity amenable to maintaining or updating their end of the interface in order to support SoS functionality? Will they adhere to the software blocking schedule. Is there a process for addressing issues of changes so individual elements won't impact the functionality of the SoS?

1.4.3 Enterprise Attributes

Within a limited domain (e.g., within a single system element), the selection of enterprise rules can be optimized for that system. This is possible because within a particular domain of interest, the set of users and their requirements may be sufficiently well defined to allow the establishment of optimum system rules that govern the interaction of the component parts. However, once a system must interoperate with other systems, a common set of rules across the SoS must be adopted. This is analogous to the establishment of, say, a public road system, that establishes limitations and restrictions on the characteristics and performance of the vehicles that use it.

The transition from “local rules” to “enterprise rules” may require the expenditure of effort to adapt system elements, which were developed under their local set of rules optimized for single-system performance, to the enterprise rules, that were developed to optimize enterprise-level or SoS performance. A corollary to this is that in order to attain optimal SoS performance, compromises among system elements must be made, often resulting in sub-optimal system-level performance.

Certain attributes of enterprise rules may indicate the degree of difficulty in establishing effective SoS environment. Examples include:

- Synchronous vs Asynchronous timing constraints.
  - Timing constraints for near-real-time SoS, such as sensor-to-shooter applications will be technically challenging, and will drive design decisions that are inherently more costly to develop and maintain.
- Deterministic vs Stochastic
  - The criticality of event timing/sequencing will be particularly important in synchronous applications such as sensor-to-shooter, and will drive design decisions that will have an effect on inherent effort. A simplistic example in a sensor-to-shooter context would be the sequence of commands “ready, aim, fire”. In this context, the order of these commands is critically important to the operational outcome. Here, a deterministic network communications protocol should be implemented, where the order of the commands received is known, and not subject to probability. If a probabilistic protocol, such as TCP/IP were to be used, there is a potential for the commands to be sent in the proper order, but be received, due to network latency, etc., as “ready, fire, aim”, which could have disastrous safety or mission effectiveness implications.
- Functional Topology
  - Different enterprise rules apply depending upon functional topologies such as client-server, master/slave, or peer-to-peer. For each of these, elements must abide by different rules regarding common and global functions, exercise of system control, redundancy plan, etc.
- Security, privacy, reliability, maintainability, availability and safety
  - These are attributes that will be imposed at the SoS level, but will give rise to requirements imposed at the system element level. For example, if the SoS reliability requirement is 0.99, but the constituent elements are at lower levels of reliability, additional effort must be incurred to bring them up to the requisite level of reliability.

1.5 Management Measures Associated with SoS: Moderating Variables

Whereas the enterprise, element, and interface attributes tend to describe the “inherent effort” attributable to the SoS effort, moderating variables are factors which

influence the efficiency and effectiveness of the processes used in accomplishing the effort. Therefore, moderating variables will affect the cost, schedule and risk of the SoS development—perhaps dramatically so. Analysis of moderating variables will inform the identification and definition of SoS development risk factors and management best practices. Moderating variables arise from two main sources: attributes of the process itself, and attributes of the environment in which the process operates.
Examples of a moderating variable related to SoS include

- "Unity of Command". When examining this moderator, the following questions are relevant: Is there a single entity in control of the SoS development? Does this entity control resources? Does this entity have the authority to impose requirements to the constituent system elements? Anecdotal evidence suggests that the lack of Unity of Command is a risk factor in SoS development.
- Investment in architectures: When proposing the implementation of an SoS, it is important to examine the architectural implications of the SoS, as these will define many of the enterprise rules imposed upon the SoS elements. \
- Congruence of stakeholder interests: If stakeholder interests are conflicting, the SoS will be pulled in multiple directions, which will impair efficiency of implementation.

1.6 Summary of Recommended Measures

Information Category	***Measurable Concept***	***SoS Question Addressed***
Schedule and Progress	Milestone Completion	Is each system element and interface meeting scheduled milestones?
	Critical Path Performance	Are critical path delivery dates slipping? Are any schedule slippages in one element affecting the critical path of other elements?
	Work Unit Progress	How are specific activities and products progressing?
	Incremental Capability	Is capability being delivered as scheduled in incremental builds and releases (e.g. blocks)?
Resources and Cost	Personnel Effort	Is effort being expended according to plan? Is there enough staff with required skills?
	Financial Performance	Is project, element, and SoS spending meeting budget and schedule objectives?
	Environment and Support Resources	Are needed facilities, equipment, and material available to support SoS integration?
Product Size and Stability	Physical Size and Stability	How much are the products size, content, and physical characteristics changing? How much are interfaces changing?
	Functional Size and Stability	Are the requirements and associated functionality changing? Are requirements changes in one element affecting other elements, or affecting SoS functionality?
Product Quality	Functional Correctness	Are the SoS and elements good enough for delivery to the user? Are identified problems being resolved within and across elements?
	Maintainability	How much maintenance does the SoS require? How much element maintenance is driven by SoS requirement changes? How difficult is the SoS to maintain?
	Efficiency	Does the SoS and elements make efficient use of system resources? Are there sub-optimizations required at the element level imposed by SoS

		requirements?
	Portability	To what extent can the functionality be re-hosted on other platforms? How extensible are SoS functions and interfaces?
	Usability	Is the user interface adequate and appropriate for operations? Are operator errors within acceptable bounds? How common are user interfaces across system elements?
	Reliability	How often is service to users interrupted within and across system elements? Are failure rates within acceptable bounds? How many failures are due to inter-element interfaces?
Process Performance	Process Compliance	How consistently do the SoS elements, and organizations implement the defined processes? At what levels of process maturity or capability are the SoS, elements, and organizations operating?
	Process Efficiency	Are the processes efficient enough to meet current commitments and planned objectives both within the elements and across the SoS?
	Process Effectiveness	How much additional effort is being expended because of rework? How much of this rework is due to within-element factors versus across-element factors?
	Environment Effectiveness	Is the management/oversight environment conducive to effective SoS implementation?
Technology Effectiveness	Technology Suitability	Can technology meet all allocated requirements or will additional technology be needed?
	Technology Volatility	Does new technology pose a risk because of too many changes?
	Technology Impact	What amount of rework is required to add new or upgraded technology within the SoS?
Customer Satisfaction	Customer Feedback	How do our customers perceive the performance on this SoS and/or elements? Is the SoS and/or elements meeting user expectations?

	Customer Support	How quickly are customer support requests being addressed?

Information Categories	Measurable Concepts	Prospective Measures
Schedule and Progress	Milestone Completion	Milestone Dates
	Critical Path Performance	Slack Time
	Work Unit Progress	Requirements Traced Requirements Tested Problem Reports Opened Problem Reports Closed Reviews Completed Change Requests Opened Change Requests Resolved Units Designed Units Coded Units Integrated Test Cases Attempted Test Cases Passed Action Items Opened Action Items Completed
	Incremental Capability	Components Integrated Functionality Integrated
Resources and Cost	Personnel Effort	Staff Level Development Effort Experience Level Staff Turnover
	Financial Performance	BCWS, BCWP, ACWP Budget Cost
	Environmental and Support Resources	Quantity Needed Quantity Available Time Available Time Used
Product Size and Stability	Physical Size and Stability	Database Size Components Interfaces Lines of Code
	Functional Size and Stability	Number of Elements Enterprise Rule Congruence Type of Internal Interfaces Type of External Interfaces Requirements Functional Changes Function Points
Product Quality	Functional Correctness	Defects Age of Defects Source of Defects (Endogenous vs Exogenous)
	Maintainability	Technical Performance Level Time to Restore
	Efficiency	Cyclomatic Complexity Utilization Throughput
	Portability	Response Time
	Usability	Standards Compliance

	Reliability	Operator Errors Mean-Time-to-Failure
Process Performance	Process Compliance Process Efficiency Process Effectiveness	Reference Maturity Rating Process Audit Findings Productivity Cycle Time Defects Contained Defects Escaping Rework Effort Rework Components
Technology Effectiveness	Technology Suitability Technology Volatility	Requirements Coverage Baseline Changes
Customer Satisfaction	Customer Feedback Customer Support	Satisfaction Ratings Award Fee Requests for Support Support Time

2 Recommendations

Further work is needed to refine and validate the SoS taxonomy described in this paper. Intersections with other related streams of research, such as that relating to integration and interoperability, should be evaluated and if possible integrated into a common understanding.

The taxonomy, mapped to measurable constructs and measures must be validated through field trials.

Once field trials have been completed, the ICM tables should be refined based upon data gathered and observations made in the field trials.

***Recommendations for further study
***Call for further participation

3 References

4 Acronym List

5 Appendix

consider putting findings from field trials in appendix
consider putting management practice, environmental risk factor information in an appendix