2009-06

Interoperability risk mitigation through the application of operational capability based engineering

Lenahan, Jack; Pacetti, Don; Heller, Scott; Reed, Rebecca; Mori, Paul

14th International Command and Control Research and Technology Symposium (ICCRTS), June 15-17, 2009, Washington DC.
https://hdl.handle.net/10945/37497

14th ICCRTS
"C2 and Agility"

Title of Paper:
Interoperability Risk Mitigation through the Application of Operational Capability Based Engineering

Topics
C2 Architectures and Technologies, Networks and Networking,

Authors: Jack Lenahan Imagine One Corporation, Don Pacetti, Commander Scott Heller USN, Rebecca Reed, Paul Mori
POC: Jack Lenahan
Organization: Office of the Chief Engineer
Space and Naval Warfare Systems Center Atlantic
Charleston, S.C.
Address: P.O. Box 190022
N. Charleston, South Carolina: 29419
Phone: 843-218-6080
Email: John.Lenahan@Navy.mil

Abstract

We are interested in investigating an operational system of systems engineering approach to the resolution of interoperability issues discovered after system deployment. Operational systems of systems engineering focuses on the engineering of systems in an end to end mission thread context. Such a methodology shifts the acquisition focus from simple 'box engineering' to the behavior of systems in their operational ecosystem.

This paper proposes a Capabilities Based Engineering Framework (CBEF) to provide a methodology that will deliver operations focused enterprise requirements in addition to traditional systems requirements. Capability based approaches[1] are used to identify *and understand* interactions, patterns, structures and properties of the end to end architecture. A System of Systems (SoS) refers to an integrated package of individual solutions that interoperate to provide a required capability. In addition to interoperability requirements, an analytically based operational capability process results in the identification of *capability gaps* for a given end to end mission thread. The resulting capability gaps become expressed in terms of *functional* requirements, *interaction* requirements and *performance requirements* for the optimal "pack" of systems and distributed services.

Introduction

We begin with the assumption that formal system of systems interoperability engineering at the mission level rarely occurs in the DoD prior to acquisition. This assumption has an expensive derivative consequence. The lack of a DoD SoS interoperability engineering process leaves the procured systems exposed to expensive interoperability repair issues after system deployment. Ignoring the need to define interoperability specifications at the beginning of a system's lifecycle, it seems difficult to avoid contractual incompleteness in terms of the acquisition community. This statement can be analyzed in terms of its immediate impact. The government is currently unable to engineer for system interoperability prior to the deployment of the new systems. If we are discussing this in a classical sense, we can see that traditional systems engineering, focusing upon delivering a particular "box" which will satisfy a documented set of narrow requirements, may encounter difficulty when interoperating or even simply interfacing with other systems. If we enlarge our discussion to include so called net centric composeable applications, weaved together as a tapestry of web services and BPEL sequences to satisfy our mission needs in new and novel ways, then our interoperability problems explode exponentially. This is quantified in a study conducted by NIST (the National Institute of Standards and Technology) as depicted in figure 1 below.

Requirements Gathering and Analysis/ Architectural Design	Coding/Unit Test	Integration and Component/RAISE System Test	Early Customer Feedback/Beta Test Programs	Post-product Release
1X	5X	10X	15X	30X

Figure1 – Relative Costs to Repair Defects when Found at Different Stages of Software Development[2]

According to the study, the cost of error correction after product release is thirty times more expensive than at the requirements stage time of a system life cycle. This study validates the need for early and continued end to end mission thread interoperability engineering and testing.

Interoperability[3] is the ability to exchange and use information. The use of the data is as important as is the exchange of the data. For example, are American telephones interoperable[4]? If two English speaking people call each other, then the answer is probably yes. They can exchange voice data and understand it. If a telephone user calls a wrong number and the person who answers only speaks Russian, then they are not said to be interoperable in that case. Please note that the phones worked properly, the voice data was precisely replicated at both ends of the call, but the voice data was unusable by the participants. To summarize, without the operational context, everything works, but no one can communicate.

We can now modify our interoperability definition to state that: systems are interoperable in clearly specified contexts such that all pre-existing constraints for exchange and usage are met.

For purposes of this paper, the clearly specified context is the mission thread. Thus, for our telephone example above, the system is interoperable for any two users who can effectively communicate given a functional technology.

Capability[5] is defined as ability to perform actions. A requirement[6] is a singular documented need of what a particular product or service should be or do. A mission defines a specific goal to be achieved through a sequence of well orchestrated actions. For example, in order to carry out a mission to find and destroy enemy submarines, the mission participants would need the capability to detect, identify, and prosecute sub surface targets, (the action of detection, the action of identification, and the action of target destruction). In order to accomplish this mission, system of systems engineers will need to derive requirements for each activity to be successful. The sequencing of these activities in order to be successful constitutes a mission thread.

Combining the total number of systems needed to satisfy the mission capabilities into a successful cohesive whole, is systems of systems engineering in a mission thread context.

The ability of each system to provide useable data throughout the mission thread from an end to end perspective is known as systems of system interoperability engineering in an end to end mission thread context.

The mission thread is the tool with which we weave composite fabrics that we call C4ISR[7] solutions. There can be time critical strike fabrics, surface warfare fabrics, interdiction fabrics, anti-submarine fabrics, etc. In geology, the term fabric describes the spatial and geometric configuration of all the elements that make up a particular rock[8]. In mission thread centric, systems of systems architectures, the multiple layers of: interfaces; systems; composeable data consuming services; fusing services; applications; systems; platforms; communications and networks capabilities constitute the spatial and geometric configurations of the elements of the architectural fabric. In simpler terms, we are using the term fabric to identify a set of architectures used to construct a system of systems architecture, or a SoS fabric if you will. The set of systems required to deliver an operational capability is also known as an end to end architecture. We chose the term fabric because the authors find it confusing to use the term architecture to simultaneously describe anything from a simple software system, to PC internals (the CPU architecture for example), or an entire set of communications architectures, network architectures or DODAF SV-6 architectures, etc. The set of capabilities delivered by multiple, integrated end to end architectures are operational fabrics. All the pieces must harmonize operationally to create a functional and interoperable fabric. The failure of any of the key pieces in any portion of the composite fabric prevents the desired capabilities from emerging. The most commonly identified failure in composing end to end mission thread operational fabrics is interoperability.

Interoperability[9] "would seem to be a straightforward concept. Put simply, interoperability is a measure of the degree to which various organizations or individuals are able to operate together

to achieve a common goal. From this top-level perspective, interoperability is a good thing, with overtones of standardization, integration, cooperation, and even synergy. Interoperability specifics, however, are not well defined. They are often situation-dependent, come in various forms and degrees, and can occur at various levels—strategic, operational, and tactical as well as technological. They are also far more likely to be recognized when interoperability problems emerge and taken for granted when such problems do not". Remember the telephone example mentioned above.

The authors of this paper believe that operational issues can best be addressed by a capability based engineering framework or CBEF. This framework is designed to enhance the acquisition life cycle. We are hoping that operational fabric analysis or system of systems architecture analysis will occur prior to specification development. Our team believes that this constitutes a professionalization of interoperability requirements engineering since operational and interoperability needs will be procured rather than 'fixed in the field'. A discussion of the CBEF model follows.

CBEF Discussion

Please remember the goal of CBEF is to reduce the risk of discovering expensive interoperability issues after the deployment of the newly developed system(s) on military platforms. Our CBEF process provides an environment which serves two specific interoperability related purposes:

1. Engineering interoperability requirements into initial specifications
2. Reverse interoperability engineering after post deployment issues are identified.

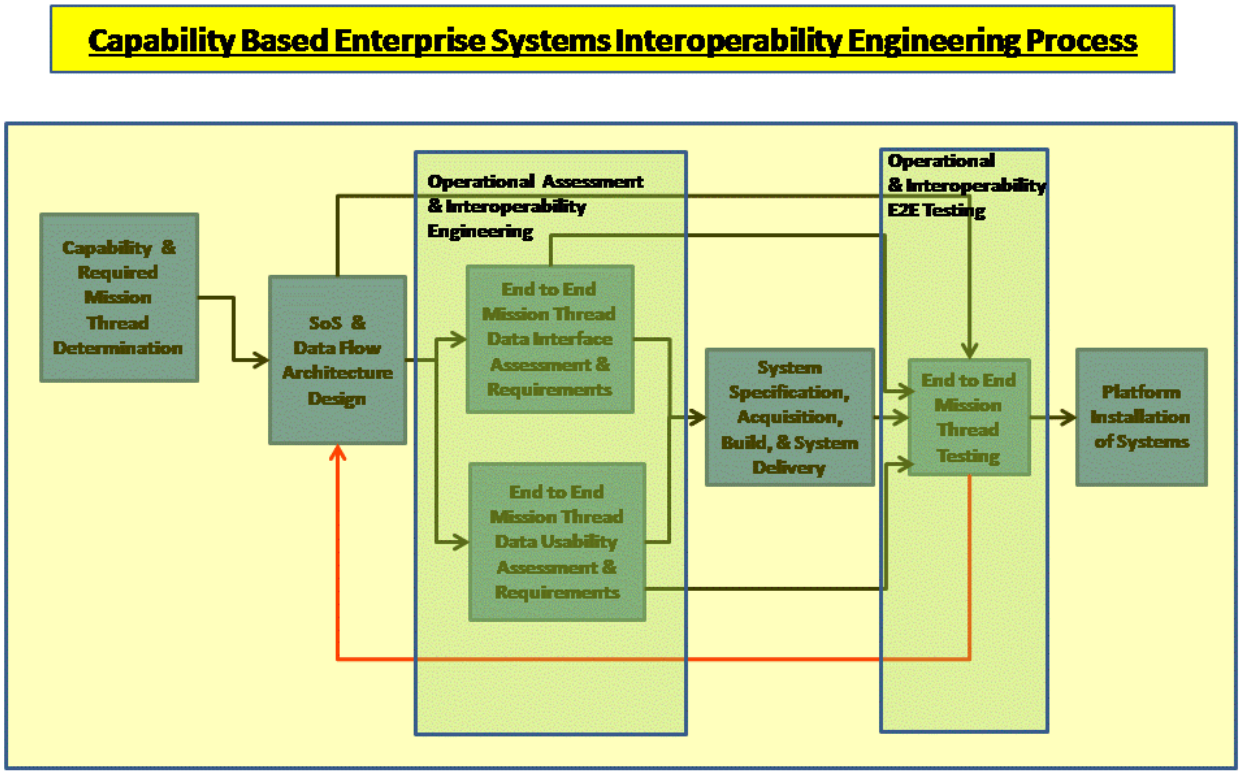

Figure 2 - The CBEF Process Model for System of System Interoperability Engineering

Figure 2 depicts a simplified acquisition model. It emphasizes several key features of the CBEF process. First, we introduce a system of systems engineering activity prior to system acquisition. The SoS activity is followed by a detailed analysis of data flows and their corresponding interfaces along with a data usability assessment associated with each downstream mission thread consuming system or human activity. This would permit interoperability data to appear in the system specification prior to contract award. This activity includes two specific interoperability functions: end to end mission thread data flow modeling which will produce data interface requirements as it outputs (at a SoS level); and an end to end mission thread data usage model (to satisfy the formal interoperability definition requirements of data exchange and usability).

We believe that mapping capabilities to mission threads, followed by a process of identifying the required individual systems, services, system collections, and statistically relevant data flows, can lead to impressive results in terms of reducing interoperability risk. By focusing on the capability and the associated mission threads needed to provide that capability, interoperability becomes manageable at least at the data interface level. However, this still leaves open the

questions surrounding data usage. Here we believe that an important step has been missing from most SoS and other System Engineering protocols: How is the data actually used in an operational environment? For example, suppose that sensor data is processed by several composed service oriented architecture (SOA) functions, each function using different fusion algorithms then presenting that data to track processors for use by a commander. Can the commander actually have enough confidence in the fused data such that he could authorize weapons launch? If the publishers of the sensor data understood its 'downstream usage', pedigreed meta data could be added to facilitate C2 decisions based upon that data. The CBEF methodology provides for a mechanism to permit the capture of system and data usage such that ***data flow patterns*** are ***understood*** in terms of ***data usage patterns***. The discussion that follows provides an example of the CBEF process as it constructs an end to end mission thread needed to support interoperability verification.

SPAWAR Systems Center Atlantic has developed several toolkits designed around capability based engineering assessments. The SPAWAR toolkits also focus on data usage. This permits a greater possibility of reducing or solving interoperability problems. The set of these toolkits is collectively known as the capability based engineering framework. The toolkits consist of several knowledge bases and intelligent user assistants. Our knowledge bases have mapped the Joint Capability Areas (JCA), service specific capabilities lists (NMTLS, UJTLS, etc.) common system function lists (CSFL), and other authoritative data sources to platforms and systems.

Our process adheres to the so called SoS engineering 'Vee". This process begins by mapping Joint warfighting capabilities to capability requirements. It is these capability requirements that will form the basis of capability verification testing (end to end mission thread level interoperability testing). The capability requirements are then used to derive the dimensions of the operational fabric required to develop the necessary system of systems architecture designs (C4ISR elements design). These products are then decomposed to individual system requirements and used to design system level architectures. The systems are then synthesized into an operational fabric.

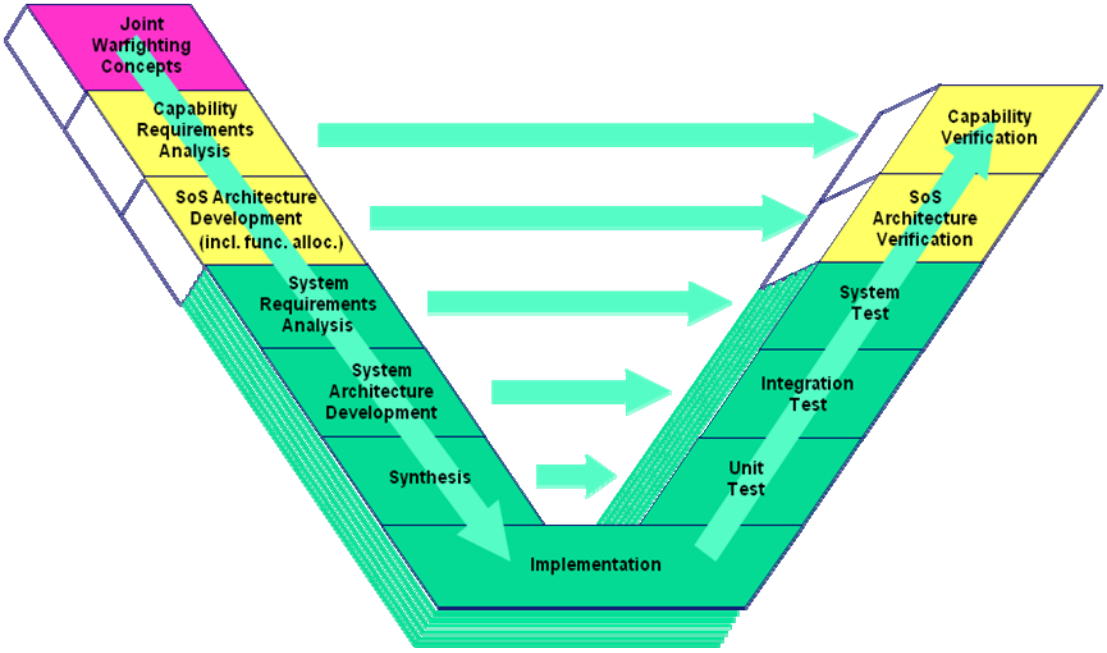

Figure3. SoS Engineering "Vee"

The arrows in figure 3 represent requirements. SoS capability requirements must be understood and properly decomposed to drive the design and development of individual solutions on the left side of the "Vee". On the right side of the "Vee", a test and evaluation process is required that can recompose the individual solutions into the SoS and validate that the overall behavior and performance satisfy end to end (E2E) requirements. If the system engineering process is properly implemented so that solutions are designed to operate within a SoS architecture from the start, the E2E test requirements are already known and testing becomes a validation of the sound enterprise system engineering.

Documentation and analysis of SoS requirements in the form of architectures is an integral part of several key acquisition milestones and artifact requirements (ICD, CDD, NR-KPP, etc). However, the usability of these products is limited as they vary in fidelity across individual solutions and also in availability based on the acquisition phase of the solution(s) under test. In addition, due to varying levels of technology maturity, test articles may not be available for solutions to test at the SoS level as desired.

The use-case that this process is targeted to support is the entry of a C4ISR solution into a lab environment to validate E2E capability. This reverse engineering process assumes that E2E requirements have not been previously defined or specified and must be determined based on the capability that the individual solution provides. In order to assemble an architecture that would capture a test of the proposed solution, we apply an operational context to support a test framework. The test architecture is baselined and may be validated for future re-use and/or integration with other net centric architectures.

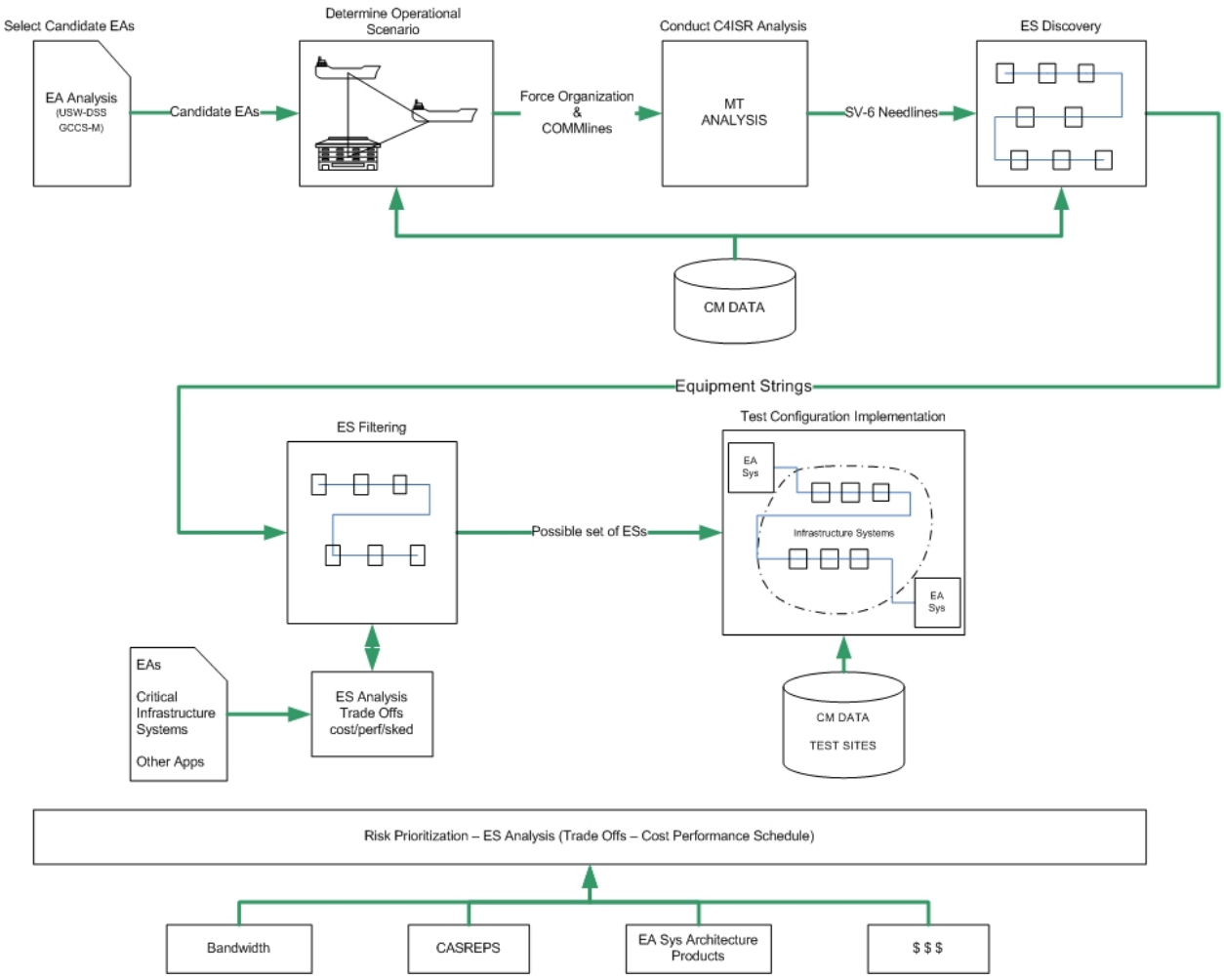

Figure 4 – System of System Engineering Process for Interoperability Testing

Figure 4 depicts the high level CBEF interoperability testing process containing the following steps:

Step 1 – Select candidate Enterprise Architectures (EA) to construct a fabric model
Step 2 – Define the Operational Scenario
Step 3 – Conduct Mission Thread and Individual Systems Analysis
Step 4 – Perform Equipment String Discovery
Step 5 – Perform Equipment String Filtering
Step 6 – Design a test configuration and execute the test

Step 1. Select Candidate Enterprise Architectures

For a given capability which needs interoperability verification, select the most appropriate architectures from program of record POR libraries. If architectures do not exist but are required for the capability to be tested for interoperability, a collaborative architecture development and design activity should be initiated to produce the desired DODAF artifacts.

Step 2. Define Operational Scenario

The input into the process is a proposed C4ISR fabric or architectural set, with one or more functions that must be tested in an E2E environment. CBEF now defines a series of operational scenarios that best describe the operational requirement(s). These scenarios must be mapped to key interoperability points in the mission threads.

Step 3: Conduct Mission Thread and Individual Systems Analysis

In this step, analysts decompose the operational scenarios defined in Step 2 into the sequence of operational activities. For each operational activity, the system function(s) required to support that activity are identified including decisions required to be supported by the activity. In addition to these functional requirements, the information elements (IEs) exchanged between operational activities in the sequence are also defined. These are referred to as interaction requirements. At this time, the operational analysts define how the data is to be used at each activity. This creates an ***understanding*** of the data flow in terms of data usage for each interaction requirement. For example, after the data is exchanged between two systems in an SoS architecture, what decisions might a watch commander make using that data as the basis of his decisions?

The output of this step is a set of SoS data interactions and data usage requirements.

Step 4: Discover Equipment Strings

In this step, the E2E strings of equipment and data flows required to fulfill the information exchange requirements are discovered and validated. This step identifies the supporting infrastructure required to exchange the information between application pairs. The equipment strings generated by this step should match the actual configuration of the operational assets.

This step may be performed manually by operational experts. But due to of the complexity and number of options that may be available, CBEF supports this step by using an automated equipment string discovery algorithm. This algorithm utilizes the required information exchanges, defined communications links and current platform configuration data to generate the communications architecture needed to satisfy the selected mission thread. This process step constructs the multi-dimensional operational fabric for the C4ISR mission.

In summary, a set of equipment strings is generated to define all applications and infrastructure required to support the functional and interaction requirements of the mission thread. The equipment strings should represent the actual configuration of the operational asset, down to the version and variant for each system

The output of this process is a set of equipment strings.

Step 5: Equipment String (ES) Filtering (Identify statistically meaningful interoperations)

In this step, the candidate equipment strings are prioritized and tailored with the goal of determining the statistically most relevant interoperations. This creates the end to end architecture which needs to be constructed and tested. The output of this step is a desired test configuration.

Step 6: Test Configuration Implementation

In this step, the prioritized equipment strings are implemented as E2E systems exhibiting the statistically meaningful data flow and interoperability. At SPAWAR Systems Center Atlantic, they are implemented by leveraging equipment in local test labs. The labs emulate the infrastructure on ships in the fleet. After test configuration construction, interoperability testing commences and various metricized reports are presented to the system designers, testers and fleet operations staff.

CBEF is still an evolving capability. Its goal is to provide an environment in which complex interoperability issues can be tested during product design or when interoperability issues arise after system deployment. We have identified several future capabilities for CBEF which are identified in the following section.

Future CBEF Directions

At this time CBEF has been used to evaluate the following fabric dimensions: network architectures, command and control architectures, communications architectures, intelligence architectures, surveillance architectures, and reconnaissance architectures in end to end mission thread contexts. We have primarily assessed legacy system based implementations. The following table summarizes the goals and future expansion of the CBEF tool kits in terms of assessing and providing interoperability analysis of the following:

SoS Type	Definition	Currently CBEF Supported	Future CBEF Capability
Virtual SoS	**Virtual SoS lack a central management authority and a centrally agreed upon purpose for the system-of-systems. Large-scale behavior emerges**	**Yes**	**Improve Current Analytical Tools to Include Hybrid Architectures – Legacy- SOA-ESB – Event Driven - Coalition**
Collaborative	**In collaborative SoS the component systems interact more or less voluntarily to fulfill agreed upon central purposes. The Internet is a collaborative system. The Internet Engineering Task Force works out standards but has no power to enforce them. The central players collectively decide how to provide or deny service, thereby providing some means of enforcing and maintaining standards.**	**Partially**	**Improve Current Analytical Decision Modeling Tools to Support Interoperability Data Usage Pattern Analysis at the Collaborative Level**
Acknowledged	**Acknowledged SoS have recognized objectives, a designated manager, and resources for the SoS; however, the constituent systems retain their independent ownership, objectives, funding, and development and sustainment approaches. Changes in the systems are based on collaboration between the SoS and the system.**	**Partially**	**Attempting to change the Procurement Process Model to Permit Independent Ownership to be Maintained but to increase the Specification Details at Procurement Time to Include Interoperability Requirements**
Directed	**Directed SoS are those in which the integrated system-of-systems is built and managed to fulfill specific purposes. It is centrally managed during long-term operation to continue to fulfill those purposes as well as any new ones the system owners might wish to address. The component systems maintain an ability to operate independently, but their normal operational mode is subordinated to the central managed purpose.**	**No**	**This model implies Evolutionary Capability Emergence. This would require automated assessment tools to permit faster identification of interoperability issues and possible meta data improvements or the creation of a formal interoperability markup language**

Table 1 – Definitions of Types of Systems of Systems[10]

Table one discusses the types of SoS available at this writing[11]. “Most military systems today are part of a SoS even if they are not explicitly recognized as such. Operationally, the DoD acts as an

SoS as military commanders bring together forces and systems (e.g., weapons, sensors, platforms) to achieve a military objective. However, DoD development and acquisition have focused on independent systems. Most systems are initially created and further developed without concern for explicit SoS considerations". [Maier,1998; Dahmann, 2008].

Summary

The following key points were addressed by this paper.

1. Interoperability is defined as the interfacing and usage of data. We expanded the definition of interoperability for systems as follows: systems are interoperable in clearly specified contexts such that all pre-existing constraints for data exchange and usage are met.
2. Interoperability issues are expensive to resolve after systems are deployed on platforms.
3. SPAWAR Systems Center Atlantic has developed a capability based engineering framework (CBEF) which will permit capture of interoperability requirements at system specification time during the acquisition cycle.
4. The CBEF methodology provides for a mechanism to permit the capture of system and data usage such that data flow patterns are understood in terms of data usage patterns.
5. CBEF also provides for an interoperability reverse engineering methodology by analyzing capabilities in an end to end mission thread such that interoperability issues can be resolved.

References:

1. Charles, Phil, personal correspondence, capability based engineering frameworks discussions.
2. "NIST Planning Report 02-3, "The Economic Impacts of Inadequate Software Infrastructure for Software Testing", May 2002, page 94
3. Interoperability definition from: wordnet.princeton.edu/perl/webwn
4. Campbell, Victor, personal correspondence
5. Baldwin, Kristen, "Defense View on Considerations for System of Systems SE", 25 October 2006
6. "Defense View on Considerations for System of Systems SE", 25 October 2006
7. Command, Control, Computers, Communications, Intelligence, Surveillance, and Reconnaissance
8. Source: wikipedia.org/wiki/Fabric (geology)
9. "Interoperability of US and Allied Forces: Focus on C3ISR", MR1235, Rand Corporation, 2000, chapter 2
10. Systems Engineering Guide for Systems of Systems, Version 1.0 , August 2008, Office of the Deputy Under Secretary of Defense for Acquisition and Technology, Systems and Software Engineering. Systems Engineering Guide for Systems of Systems, Version 1.0. Washington, DC: ODUSD(A&T)SSE, 2008, page 17
11. Ibid., page 16