Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 00 (2013) 000–000

Procedia Computer Science

www.elsevier.com/locate/procedia

Conference on Systems Engineering Research (CSER'13)
Eds.: C.J.J. Paredis, C. Bishop, D. Bodner, Georgia Institute of Technology, Atlanta, GA, March 19-22, 2013.

A fuzzy evaluation method for system of systems meta-architectures

Louis Pape[a]*, Kristin Giammarco[b], John Colombi[c], Cihan Dagli[a], Nil Kilicay-Ergin [d], George Rebovich[e]

[a]*Missouri University of Science and Technoloy, Rolla, MO 65409-0370 USA*
[b]*Naval Postgraduate School, Monterey, CA 93943 USA*
[c]*Air Force Institute of Technology, Dayton, OH 45433-7765 USA*
[d]*Penn State University, Malvern, PA 19355 USA*
[e]*MITRE, Bedford, MA 01730-1420 USA*

Abstract

A method is proposed for evaluating a range of System of Systems (SoS) meta-architecture alternatives. SoS are composed through combination of existing, fully functioning Systems, possibly with minor functional changes, but certainly by using the combined Systems to achieve a new capability, not available from the Systems alone. The meta-architecture describes how all possible subsets of Systems can be combined to create an SoS. The fitness of a realizable SoS architecture may be characterized by terms such as unacceptable, marginal, above average, or excellent. While these terms provide little information about the SoS when used alone and informally, they readily fit into fuzzy membership sets that overlap at their boundaries. More descriptive attributes such as "ease of use," which might depend on individual user and a set of conditions, "mission effectiveness" over a particular suite of missions, and "affordability," which may change over time with changing business climate, etc., lend themselves readily to fuzzy evaluation as well. An approach to defining the fuzzy concepts and establishing rule sets to provide an overall SoS evaluation for many sets of participating individual Systems represented by the meta-architecture is discussed. An application of the method is discussed within the framework of developing and evaluating a hypothetical Intelligence, Surveillance and Reconnaissance (ISR) SoS capability.

Selection and/or peer-review under responsibility of Georgia Institute of Technology.

Keywords: architecture, system of systems, fuzzy, evaluation

1. SoS Background

Policies on architecting in the DoD continue to evolve, although perhaps at a slower pace than in the past. As a result, the modeling of architecture development and evolution is not yet practiced consistently throughout the community. This is particularly true in SoS settings [1]. Existing analysis methodologies and tools narrow the scope of the SoS problem space by invoking the assumption that there is a limited set of solutions that are solely or primarily driven by performance considerations that are technical in nature. However, the SoS problem boundary includes integration of technical systems as well as cognitive and social processes, which alter system behavior [2]. Most system architects design with the assumption that SoS participants exhibit nominal behavior (utopian behavior), but deviation from nominal motivation leads to complications and disturbances in systems behavior. It is

necessary to capture the behavioral dimension of SoS architecture to be able to represent the full problem space to guide the SoS architecting and analysis phase [2]. Agent based models (ABMs) can help explore the dimensions of the social aspects of SoS development by abstracting to simpler interaction models. Since one of the current problems with SoS composition is a limited, pre-conceived notion of a solution, a genetic algorithm (GA) approach may help to explore the potential architecture space more fully [3]. In both the ABM and GA approaches, however, evaluation of the intermediate and final architecture results is necessary. Evaluation of architectures lends itself to a fuzzy approach because the criteria are frequently non-quantitative, or subjective [4], or based on unknowable future conditions, such as "robustness."

In an "Acknowledged SoS," an initial SoS mission is determined and funds are allocated to the mission with a responsible organizational entity employing the SoS Manager. SoS component Systems are independent and have their own functionality, development processes, funding and operational missions. Component Systems are frequently in different phases of their own life cycles [5]. The SoS Manager has some, but not complete, authority over component Systems when they work within the SoS. He may solicit, offer funding, cajole, use the bully pulpit, beg, log roll, match make, and use his own influence to gain cooperation from the component Systems' personnel. Component Systems may have their own problems, interests and desires, goals, external influences, restrictions, image to protect or enhance, fears, and opportunities that make them cooperate more or less with the SoS. The ABM and GA approaches help model, explore, and analyze the influences of opportunity and social interaction.

The analysis question addressed by the method proposed in this paper is as follows: Given a set of participation / non-participation decisions made for each system in the proposed SoS, how well will a resulting candidate SoS architecture selection perform?

2. Evaluation of Typical Attributes

Any number of architecture attributes may be used for evaluation, but peoples' cognitive processing power, as well as the number of independent attributes, typically cannot sustain large numbers [6][7]. A program's (System's) current attribute status is often presented as a color code, as in the Contractor Performance Assessment Report [8] for example, or plotted on stop light charts, or displayed on Kiviat charts to allow reviewers to compare evaluation of alternative architectures, or a program status from one quarterly review to the next. An overall evaluation is still largely a gestalt of component attribute values, especially when not all areas are weighted equally. Attribute evaluations themselves are ideally suited to fuzzy logic approaches because of the difficult nature of boundaries between evaluation regions. A particular SoS architecture chromosome may fall partially into an Acceptable, and partly into a Marginal set, as any point on the goodness scale between 2.3 and 2.8 does, shown in Figure 1.

.

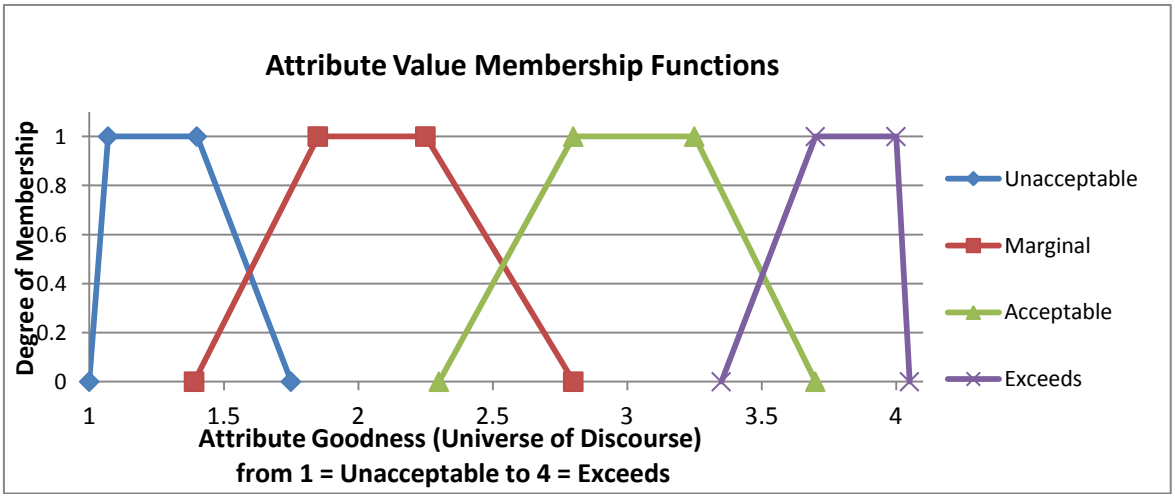

Figure 1. Attribute value membership functions

Four attributes are selected to demonstrate the evaluation method in this example: **Performance**, **Affordability**, Developmental **Flexibility**, and Operational **Robustness**. Other attributes are possible, but for simplicity, only the four are discussed. It is unlikely that any set of SoS attributes could be completely orthogonal. The attributes themselves are fuzzy, and they overlap somewhat in that they frequently evaluate in a correlated way; i.e., good programs are frequently good in many areas, etc. The gradations of the attributes are also "fuzzy," in that an exact boundary between any described gradations from bad to good is difficult to define. Uncertainty about what is meant (at the boundaries) by any of the attributes themselves (or the gradations within them) implies that some observers might evaluate an SoS as Affordable, and some observers might interpret that same SoS as *not* Affordable – based on a perceived but non-quantifiable difference in risk, margins, value, recent expenditure rate, gut feel, sizing up of program personnel personalities, etc. Discussion of the several types of uncertainty possibilities bring in different types of fuzzy inference systems and is beyond the scope of this paper. This paper is merely to introduce fuzzy evaluation to architectures. In fact, these differences of opinion are always there and managers are employed to evaluate them and manage with them. The range of uncertainty can be represented by an overlap of each gradation of the attribute evaluation; the demarcation between the evaluation levels is what is fuzzy. A fuzzy inference system allows operation on the fuzzy boundaries in a mathematically precise way. Fuzzy membership functions for the grades in each attribute include the imprecision of the language. Rules about how to combine the component attribute evaluations into an overall evaluation of the SoS consider the importance of individual attributes to the whole. The membership functions need not be compact, nor single peaked; conclusions can be drawn with mathematical precision regardless of the shape of the membership function. This is one of the features that allow fuzzy logic to cover non-linear situations. The rules can be simple or complex, as well. It is simply easier to think and learn about the fuzzy logic model approach when the functions are compact with small overlaps and the rules are few. [7] The purpose of the model is not to predict all SoS development, but to learn about how the structure of this modeling approach can help us understand SoS developments.

Table 1. Fuzzy Rule Set defining SoS evaluation from the attribute values

Plain Language Rule	**Fuzzy Rule Definitions from MATLAB Fuzzy Toolbox**
If ANY attribute is Unacceptable, then SoS is Unacceptable	If (Performance is Unacceptable) or (Affordability is Unacceptable) or (Developmental_Flexibility is Unacceptable) or (Robustness is Unacceptable) then (SoS_Arch_Fitness is Unacceptable)
If ALL the attributes are Exceeds, then the SoS is Exceeds	If (Performance is Exceeds) and (Affordability is Exceeds) and (Developmental_Flexibility is Exceeds) and (Robustness is Exceeds) then (SoS_Arch_Fitness is Exceeds)
If ALL the attributes are Marginal, then the SoS is Unacceptable	If (Performance is Marginal) and (Affordability is Marginal) and (Developmental_Flexibility is Marginal) and (Robustness is Marginal) then (SoS_Arch_Fitness is Unacceptable)
If ALL the attributes are Acceptable, then the SoS is Exceeds	If (Performance is Acceptable) and (Affordability is Acceptable) and (Developmental_Flexibility is Acceptable) and (Robustness is Acceptable) then (SoS_Arch_Fitness is Exceeds)
If (Performance AND Affordability) are Exceeds, but (Dev. Flexibility and Robustness) are Marginal, then the SoS is Acceptable	If (Performance is Exceeds) and (Affordability is Exceeds) and (Developmental_Flexibility is Marginal) and (Robustness is Marginal) then (SoS_Arch_Fitness is Acceptable)
If ALL attributes EXCEPT ONE are Marginal, then the SoS is still Marginal	If (Performance is Marginal) and (Affordability is Marginal) and (Developmental_Flexibility is Marginal) and (Robustness is Acceptable) then (SoS_Arch_Fitness is Marginal)
	If (Performance is Marginal) and (Affordability is Marginal) and (Developmental_Flexibility is Acceptable) and (Robustness is Marginal) then (SoS_Arch_Fitness is Marginal)
	If (Performance is Marginal) and (Affordability is Acceptable) and (Developmental_Flexibility is Marginal) and (Robustness is Marginal) then (SoS_Arch_Fitness is Marginal)
	If (Performance is Acceptable) and (Affordability is Marginal) and (Developmental_Flexibility is Marginal) and (Robustness is Marginal) then (SoS_Arch_Fitness is Marginal)

Each of the attributes in this example are normalized to a range of goodness from Unacceptable through Marginal and Acceptable to Exceeds, shown in Figure 1, but each could have individualized gradations. One way to look at the membership functions is that some evaluators might refer to a program in the center of the range of "goodness" from worst to best as Acceptable, some might choose to call the same program Marginal. An equally valid way to consider the fuzziness is that *today* the program might be Marginal, but *next week* it will be acceptable with only miniscule changes in the various subcomponents that make up its evaluation. It is difficult to define precisely the line separating the gradations, *therefore one does not try*. One simply defines the boundary regions in a fuzzy way, with partial membership in more than one attribute allowed at the overlap.

The membership functions shown in Figure 1 have overlap between each adjacent value, with slightly more overlap between Marginal and Acceptable than for the other boundaries. Slightly more membership opportunity exists in the Marginal and Acceptable categories than in combined Unacceptable and Exceeds regions. In this example, there is no overlap between categories that are not adjacent, but this is not a requirement.

The selection of membership functions must represent a consensus among the stakeholders and architects. It is key to the process that a series of guided discussions among that group, by an architect with experience on numerous Programs and SoS over most of the typical life cycle, lead to selection of the shapes such as those shown in Figure 1, and rules or guidelines shown in Table 1, to translate from the individual attribute evaluations to an overall SoS evaluation. This overall evaluation can then be used as the fitness function for the SoS GA.

All the SoS attributes were simplified for this task to have the same membership function shapes, labeled with the same four gradations. However, the membership functions can be as complex as it makes sense. Individual attributes may be scored through fuzzy methods as well, but how the fuzzy process works is shown only at the top level. The next step is to derive an overall SoS evaluation through combinating the individual attribute values. The Fuzzy Logic Toolbox™ in MATLAB© allows for multiple evaluation scoring regions and various membership function shapes quite simply, but it is easier to explain the concepts with all of attribute gradations shaped the same. One can normalize the different areas with individual weights, as well. All weights are assumed equal here.

The above rules were incorporated into the Fuzzy Logic Toolbox™, yielding the displays of Figures 2-3.

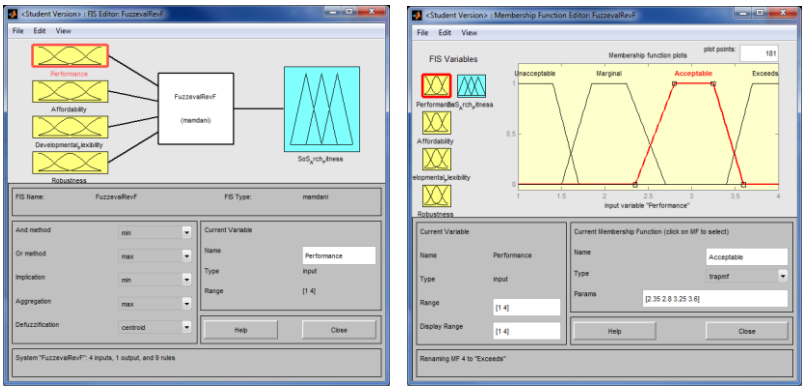

Figure 2, (a) MATLAB Fuzzy Logic Toolbox, (b) input membership functions for the Attributes

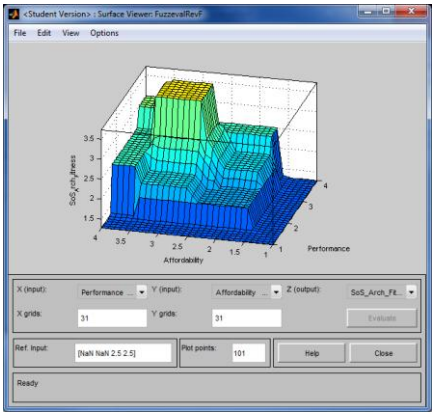

Figure 3. 3-D output example of fuzzy SoS evaluation showing the highly non-linear effect of the simple rules in Table 1

In this research, the concept of a chromosome to store information about a proposed SoS architecture, herein referred to as a *meta-architecture* is used. The chromosome consists of binary bits representing the participation of individual systems and their interfaces. A fuzzy inference system (FIS) is used to combine attribute values that

contribute to the higher-level SoS evaluation as shown in Figure 2 The evaluation process for the attributes is not defined yet. The FIS allows one to describe this evaluation in a simple, general way. This concept is similar to the color coding used in Contract Performance Assessment Reports [7], with red, yellow, green, gold and blue representations across a number of performance areas. One could use fuzzy criteria to reach the evaluation for each of the attributes in a real system, but that is an easy extension of this approach.

3. A genetic algorithm (GA) approach of looking at SoS composition

For a proposed SoS with **n** Systems, the SoS manager invites the Systems into the SoS with some offer of resources, usually funding, to provide an extension of their current capability. This extension, whether of basic capability or a changed interface with other System(s) within the proposed SoS, enables the desired new SoS capability. The SoS manager proposes an organization and resources to create the new SoS capability.

The Systems contract to accept the resources and deliver their capability increment as participating Systems within the SoS, *or* they choose not to participate. This simple binary representation of the SoS constitutes the makeup of a chromosome upon which a genetic algorithm may operate. The intent is to be able to search the potential architecture space, as represented by the different possible combinations of participating systems and their interfaces, and then evaluate each candidate SoS within the GA as it creates mutations of the chromosome in each generation. The fuzzy evaluation is used by the GA to approve chromosomes for propagation to the next generation. When the SoS manager is satisfied with a chromosome, it may be passed on to the ABM to see how the Systems respond (through their agents). If a system chooses not to participate, there may be further rounds of negotiation and development based on the capabilities that the participating Systems can deliver to the SoS.

The simplest model comes down to an initial probability **p** that each individual System will agree (or be able) to participate **($0 \leq p \leq 1$)**. In the case of a refusal, the System might be having internal problems (technical, managerial, staffing, schedule, etc) that prevent it from being able to take on any additional tasks; they might think the proposed SoS won't work, and therefore choose not to divert any attention to it; the stakeholders for their primary mission may not allow such diversion of resources; or they may desire to participate, but not have the capability to do so. For whatever internal reasons, a System may choose *not* to participate; this is represented by a "0" in the SoS chromosome at the position representing that System, discussed further in the next section. There are **n** possible Systems. The expected number of systems to participate on average in each instantiated SoS architecture is **pn** . The decision itself can be determined through complicated, multiparty negotiations, or in this first example of an evaluation framework, by calling a random number generator (evenly distributed between zero and one), to decide if the System will participate when the random number is less than **p** .

s1	s2	s3	s4	s5	s6	s7	s8	s9	s10	1-2	1-3	1-4	1-5	1-6	1-7	1-8	1-9	1-10	2-3	2-4	2-5	2-6	2-7	2-8	2-9	2-10	3-4	3-5	
1	1	1	0	1	1	1	1	1	1	1	1	0	0	1	0	1	1	0	1	0	1	0	1	1	1	1	0	1	…

Figure 4. Example partial GA chromosome for SoS with 10 Systems

Part of the chromosome for the 10 System SoS is shown in Figure 4 (**n** = 10). In this example, System 4 chooses not to participate; so no interfaces can exist with System 4. In addition, System pairs (1,5), (1,7), (1,10), and (2, 6) do not create an interface between them for whatever internal reasons.

The determination of whether two Systems achieve an interface between them can be independently arrived at through a model of System interactions, but for the purposes of this research, the achievement of an interface between two systems has a probability of **q,** where **($0 \leq q \leq 1$)**.[a] If a uniformly distributed random number is less than **q** for a given pair of Systems, then the two Systems achieve an interface between them. A "1" represents the successful interface at that interface position in the SoS chromosome. Unless both Systems are already participating, **q** does not come into play, because an interface cannot exist with a non-participating System (at the very least, it does not do any good). If all Systems participate, the maximum number of interfaces is **n(n-1)/2** . If

[a] The two Systems might both be willing to participate, but still only be able to achieve the interface between them with probability **q**.

only **pn** Systems participate, and **q** is the likelihood of having an interface with another participating system, then $\mathbf{p^2qn(n-1)/2}$ is the *expected* number of contributing interfaces. Placing a "1" in the chromosome represents an interface between the ij^{th} Systems if the Systems *both* participate, *and* a random number (different from the two generated for participation probability) as described above is less than **q** when generating that chromosome bit. When a GA creates a new generation of trial chromosomes it may use internal knowledge of the systems, feasibility checking, or other techniques to select chromosomes for fitness evaluation, but for illustrating the fuzzy method proposed here, only randomly generated chromosomes were used.

4. Attribute Evaluations

The meta-architecture offers a space that is spanned by all possible SoS architecture configurations. From the meta-architecture perspective, the evaluation of the chromosome provides a fitness value to use in a GA approach. Genetic algorithms have been used for the generation and selection of system architectures in conjunction with fuzzy logic as fitness assessor previously [3]. The novel aspect of this method is that it can be used to evaluate a chromosome directly for the four attributes introduced in section 2: Performance, Affordability, Flexibility, and Robustness.

4.1. Performance

To evaluate the **Performance** attribute, let the average capability contribution of the individual System be $\mathbf{C_i = C_{req}/n}$, where *i* is the System index number and $\mathbf{C_{req}}$ as a first iteration estimate of the requested capability from the SoS. In a reference to Network Centric Systems theory [9], one may also assume that there is *some* capability gained through the interfaces between the Systems as well. Using $\mathbf{C_{int\ ij}}$ as the small but distinct capability contribution from the $\mathbf{ij^{th}}$ interface; assume the *expected* total contribution from the interfaces is about equal to the initial *requested* capability from the Systems themselves (before you lose some due to the participation probability):

$$\mathbf{C_{int\ ij} = C_{req}/(p^2qn(n-1)/2)} \qquad (1)$$

But you *expect* $\mathbf{p^2qn(n-1)/2}$ interfaces, so the sum of the contribution from the interfaces for the SoS is simply $\mathbf{C_{req}}$. Therefore, the *expected* capability of the SoS from both Systems and Interface contributions is

$$\mathbf{C_{SoS} = pnC_i + C_{req} = (1+p^2)C_{req}} \qquad (2)$$

In other words, the capability contribution if *all* systems participated with the expected probability of achieving their interfaces is $\mathbf{q\,C_{req}}$. The fact that a random selection of Systems and Interfaces with the selection parameters **p** and **q** can provide more or less than the expected numbers of systems and interfaces allows randomly generated architecture chromosomes to deliver more or less than either expected or requested capability. Simply normalize the performance by the initial requested capability $\mathbf{C_{req}}$. If one gets *exactly* the *expected* contributions, the performance attribute will be $\mathbf{(1 + p^2)C_{req}}$. If **p** and **q** were small but you got large numbers of systems participating and interfacing by chance, you could get a large number for the normalized, delivered capability.

4.2. Affordability

Let **B** be the Total Budget of the SoS; assume for the moment that interfaces cost nothing. When computing affordability, it is important to consider that without *some* extra percentage offered over the budget, there would *never* be a SoS architecture chromosome that would be over budget in our simple binary participation model, resulting in an **Affordability** attribute value below Acceptable. One expects *some* invited Systems to be unable to comply for various reasons ($\mathbf{(1-p)n}$) will not participate; let $\mathbf{p'}$ be $\mathbf{(1-p)}$); those Systems would reject the budget offered, and there would be no spending of those Systems' budget allocation. However, if you use $\mathbf{Bp'}$ to adjust the offered budget, then anytime the random number generator produces even one more than the *expected* number of

participating Systems, it will already be over budget. By distributing only **Bp'/2** extra total budget, a few more chromosomes will be brought within the budget, and therefore will be worth evaluating further. If the *expected* number of systems participate, the SoS will be Affordable to a reasonable (Affordability attribute = Acceptable), but not exceptional (Affordability attribute = Exceeds), degree. Therefore, let the decision to participate cost each System **(1+p'/2)B/n.**

4.3. Flexibility

Developmental **Flexibility** can be represented simply by the number of Systems participating. If more Systems participate, there is more room for the SoS manager to maneuver among them during development with his attention, influence, oversight, budget, or whatever it takes to keep the SoS moving along.. Flexibility can be represented by the simple sum of the System components of the chromosome divided by **n** (to keep the values normalized). One could argue that more systems contribute to Robustness as well, because flexibility and robustness may be strongly correlated; that is, connotations of the words have overlap, and more so in the context of systems. However, that aspect of fuzzy linguistic analysis was not explored for the feasibility demonstration. The ideal concepts of orthogonal measures and deviations from that ideal are deferred. this approach is limited to provide a reasonable method of evaluating Flexibility from the chromosome. .

4.4. Robustness

The number of interfaces is used as the measure of Operational **Robustness**, with the maximum number of interfaces as the normalization. This measure allows chromosomes to evaluate better or worse than the *expected* number of interfaces. Mathematically, robustness is defined as the sum of the interface chromosome positions (with a value of one), divided by the normalization factor of the maximum possible number of interfaces **n(n-1)/2** .

4.5. Summary of attribute evaluations

Non fuzzy evaluation algorithms for each SoS attribute can then be summarized in terms of the chromosome itself as shown in Table 2, with a random example chromosome shown in Figure 4.

Table 2. Evaluation of Attributes from chromosome

SoS Attribute	Equation	Unaccep-table	Marginal	Accep-table	Exceeds
Performance	**(Sum (C_i) + Sum ($C_{int\ ij}$)) / C_{req}**	< .8	.8 - 1	1- 1.5	< 1.5
Affordability	**(Sum ($System_i$ * (1+(1-p)/2)B/n)) / B**	> 1	.9 - 1	.8 - .9	< .8
Developmental Flexibility	**Sum ($System_i$) / n**	< .5	.5 - .8	.8 - .9	.9 - 1
Robustness	**Sum ($Interface_{ij}$)/ (.5n(n-1)))**	< .5	.5 - .7	.7 - .9	.9 - 1

The Attribute evaluation measures in Table 2 were coded in an Excel spreadsheet implementing a random chromosome shown in Figure 4. The fuzzy evaluation process may be brought down to the level of the attribute evaluations in Table 2, but for the demonstration the fuzzy evaluation step occurs only once, at the SoS level. Typical attribute level criteria are listed in Table 4 for a hypothetical Intelligence, Surveillance and Reconnaissance (ISR) System of Systems. The ISR example is inspired by the 1991 Gulf War's "Scud Hunt," where despite thousands of sorties, no Scud launchers were destroyed. Existing systems were not capable of providing the required capability, but a potential SoS might have been able to do so. [10] Values for **p** and **q** between 0.6 and 0.85 produced a reasonable range of SoS evaluation values to demonstrate that the technique worked in less than a minute on laptop computers.

Table 3. Spreadsheet evaluation of sample chromosome using above process

		Inputs to Fuzzy Evaluation
SoS Performance	2.23	E
SoS Affordability	1.03	U
SoS Flexibility	0.9	E
SoS Robustness	0.77	A

Table 4. Example four attribute evaluation guidelines to get the input to the fuzzy SoS evaluation

Evaluation Attribute	**Unacceptable**	**Marginal**	**Acceptable**	**Exceeds performance**
Performance (KPPs for ISR SoS here:) - Coverage (sq km/hr) - Resolution - # of channels - Timeliness (sensor on target, and processing, dissemination) - Adaptability (sensor type match to target)	Fails to meet multiple key performance parameters (KPPs)	Fails to meet at least one key performance parameter (KPPs)	Meets or exceeds all KPPs.	Exceeds performance in one or more KPPs by 20% or more.
Affordability A measure of the projected total ownership cost versus budget (acquisition cost plus O&M cost) and delivered capability.	Projected total ownership cost exceeds 120% of budget. Large mismatch in annual estimates.	Projected total ownership cost exceeds 100% of budget.	Projected total ownership cost is between 85% and 99% of budget.	Projected total ownership cost is less than 85% of budget.
Robustness (in the field) Ability of the SoS to continue proper functioning despite external disturbances.	More than 30% degradation in one or more KPPs due to external disturbances.	Between 11% and 30% degradation on one or more KPPs due to projected external disturbances.	Between 5% and 10% degradation in one or more KPPs due to projected external disturbances.	Not more than 5% degradation in any KPP due to estimated external disturbances.
(Developmental) **Flexibility** Ease with which the SoS can be repurposed to support other missions.	Architecture is monolithic and key SoS capability applications are tightly coupled.	Several different Architectures are possible with varying degrees of cooperation among systems.	Architecture is layered and most key SoS capability applications are loosely coupled.	Architecture is fluid and all key SoS capability applications are loosely coupled.
Ease with which individual system contributions can be traded .	0-25% of key functionality is allocated to software.	26-50% of key functionality is allocated to software.	51-75% of key functionality is allocated to software.	> 75% of key functionality is allocated to software.

By creating sample, random chromosomes in an Excel spreadsheet, one can easily check the evaluation criteria for a variety of test architecture chromosomes. The combination of the four attribute scores through the rules occurs in the fuzzy inference system in MATLAB. A reasonable number of trials with sample rules and limits allow the facilitator to tune the rules, membership functions, and category boundaries fairly quickly.

Whereas this example used all the component systems equally, the approach lends itself equally well to a SoS with constraints such as: some System to System interfaces are not feasible, the constituent Systems do not contribute equally in overall capability, or if some Systems are bottlenecks, in that, without their participation, the SoS architecture as a whole is infeasible. An example chromosome with the latter constraint is shown in Figure 5.

In this random fragment, no Comm systems participate, which is clearly unacceptable since the comms functionality is not covered by any other System. So there is another test of feasibility in the specialization case – there must be at least one participating system in each specialty area. In addition, interfaces between *like* Systems are not as likely to be useful, so one may have to discard many of the potential interfaces, arriving at a smaller, *useful* subset of interfaces that join dissimilar specialty area Systems. This is shown in Figure 6. As additional such rules and constraints are discovered, the logic for them can be codified in the SoS fuzzy evaluation framework.

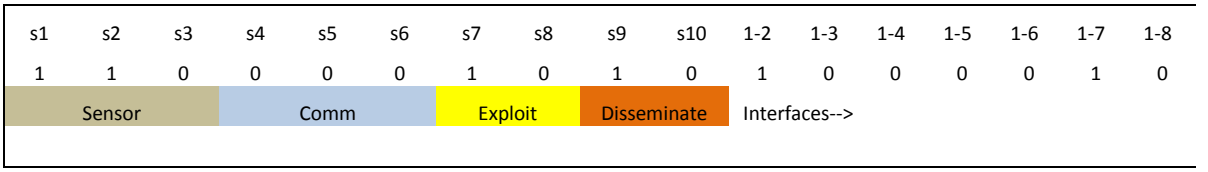

Figure 5. Chromosome fragment with specialized component Systems highlighted

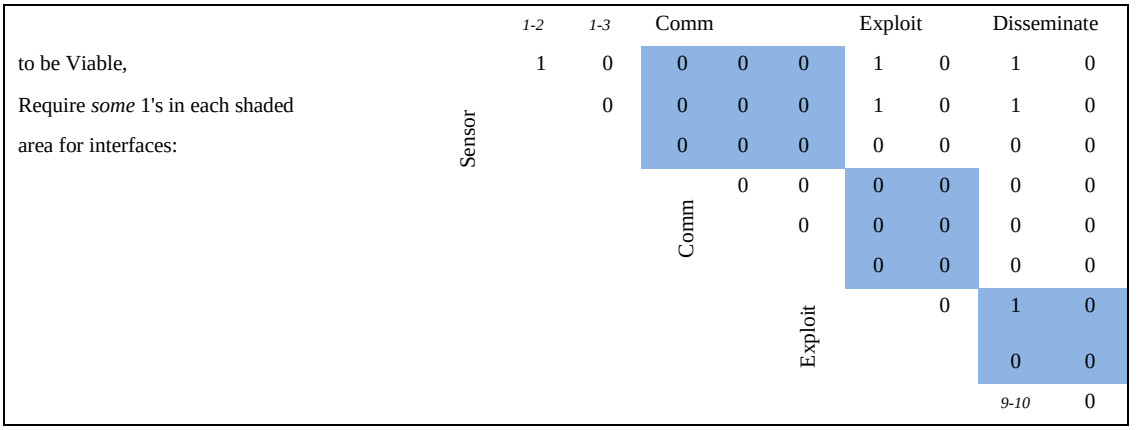

Figure 6. Interface chromosome fragment, sliced and stacked, to show interfaces that contribute to an SoS with non-similar specialization area Systems shaded. Sensor connects to Exploit through Comm, so the apparent interfaces in the upper right, unshaded portion are not useable

5. Conclusion

A proof of concept agent-based model of the system interactions was developed and integrated with a genetic algorithm to explore the potential architectural design space, using a fuzzy inference system to evaluate candidate architectures for simulating acknowledged SoS creation and evolution. The participation of each System is modeled as a binary choice in a GA "chromosome" representing the possible subsets of participating systems and their interactions with other Systems within the SoS. The GA approach allows a more thorough exploration of the architectural "space" (composed of all the possible subsets of Systems and interactions) than typical, biased, preconceived, human selected subsets. The Fuzzy evaluation approach allows the introduction of non-linearities in the fitness through relatively simple rule sets and membership function determinations. The GA and Fuzzy evaluator have been integrated through the ABM for some sample chromosomes.

The model evaluates the capability of the evolving SoS architecture with respect to four attributes: performance, affordability, flexibility and robustness. The method may be easily extended to non-binary, partial cooperation, or to specialty System components of the SoS with some renormalization. The evaluation criteria can be tailored to reflect the needs of different SoS; for example, one could just as easily formulate the approach with no cost for participation but assign costs to interfaces, or change the definitions and the number of attributes. This research lays the groundwork for formalizing the evaluation and evolution of acknowledged SoS. It provides underlying logical constructs for modeling the impact of the presence or absence of systems and interfaces on an overall SoS

capability, and allows experimenting with simple rule changes in the fuzzy inference systems that could be the equivalent of policy changes in acquisition.

The next steps include extending the implementation to larger sets of systems, including more general rules for combining them through interfaces and including additional information about their contribution to the SoS capabilities, and mapping potential actual SoS components and capabilities with more realistic rule sets for the attributes' evaluations. Improvement in the GA evolution approach for small changes in later generations to examine "accessible" regions of improvement will be pursued, because you don't completely discard an existing SoS and start over in subsequent rounds of negotiations, but evolve the capability by making small changes. Finally, improved functionality in the ABM portion of the model will look at various negotiation techniques by individual systems for deciding participation and scheduling more intelligently than the demonstration models.

Acknowledgements

This material is based upon work supported, in whole or in part, by the U.S. Department of Defense through the Systems Engineering Research Center (SERC) under Contract H98230-08-D-0171. SERC is a federally funded University Affiliated Research Center managed by Stevens Institute of Technology.

Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the United States Department of Defense.

References

[1] DoD, ASD(NII), "DoD Architecture Framework Version 2.0 (DoDAF V2.0)," Department of Defense, Washington DC, 2009.

[2] J. P. Dauby and S. Upholzer, "Exploring Behavioral Dynamics in Systems of Systems," in *Complex Adaptive Systems, Procedia Computer Science, vol 6*, Elsevier, 2011, pp. 34-39.

[3] K. Haris and C. Dagli, "Architecture Trade-off Analysis and Reconfiguration," in *Proceedings Conference on Systems Engineering Research*, 2011.

[4] J. P. Dauby and C. H. Dagli, "The canonical decomposition fuzzy comparative methodology for assessing architectures," *IEEE Systems Journal, vol. 5, no. 2,* pp. 244-255, 2011.

[5] OUSD (AT&L), Dir Sys & S/W Eng, "Systems Engineering Guide for Systems of Systems," available from http://www.acq.osd.mil/se/docs/SE-Guide-for-SoS.pdf, 2008.

[6] G. A. Miller, "The magical number seven, plus or minus two: Some limits on our capacity for processing information," *Psychological Review*, vol. 63, no. 2, pp. 81-97, 1956.

[7] W. Pedrycz, P. Ekel and R. Parreiras, Fuzzy Multicriteria Decision Making; Models, Methods and Applications, West Sussex: John Wiley & Sons, 2011.

[8] Dept of the Navy, "Contractor Performance Assessment Reporting System (CPARS)," Washington DC, 1997.

[9] D. S. Alberts, J. J. Garstka and F. P. Stein, Network Centric Warfare: Developing and Leveraging Information Superiority, 2nd Edition, Washington DC: C4ISR Cooperative Research Program, 1999.

[10] M. Thompson, "Iraq: The Great Scud Hunt - TIME," 23 December 2002. [Online]. Available: http://www.time.com/time/magazine/article/0,9171,1003916,00.html. [Accessed 15 September 2012].