System of Systems Complexity Identification and Control

Joseph J. Simpson
Systems Concepts
6400 32nd Ave NW #9
Seattle, WA 98107
jjs-sbw@eskimo.com

Mary J. Simpson
Systems Concepts
6400 32nd Ave NW #9
Seattle, WA 98107
mjs-sbw@eskimo.com

Abstract *– System of systems deployed in dynamic, adaptable environments represent the potential for the generation of great value, power and functionality. The great potential associated with system of systems deployment is offset by the increased complexity associated with the system of systems design, development, deployment, operation and disposal. This paper explores the connection between classical systems engineering techniques of N Squared Charts and Design Structured Matrices coupled with evolutionary algorithms to address both the cognitive complexity and computational complexity associated with system of systems life-cycle events. The key aspect explored in this paper is the enhancement of human perception using computational techniques. The global organizing system relationships, associated with the system of interest, are addressed by the computational component while the local value and interface relationships are addressed by the human system designers. The proper balance of global and local system relations coupled with advanced computational techniques provides the ability to reduce system complexity and increase the effectiveness of deployed systems and systems of systems operations.*

Keywords – System, System of Systems, Complex Systems, Complexity, Abstract Relation Types, N Squared Chart, Design Structure Matrix, Evolutionary Algorithms.

1 Introduction

The increasing rate of systems and system of systems design, development and deployment makes the task of system of systems evaluation, risk and value assessment an increasingly complex task. The potential great value and capability leverage that can be gained by creating a system of systems from a large set of existing systems coupled with a small set of new and/or modified systems drives organizations and enterprises to pursue the construction and interconnection of ever larger aggregations of systems.

Definitions of the primary concepts in this paper are presented next to facilitate the clear communication of the fundamental ideas, concepts and system insights that are presented in the text. A system can be defined in two basic ways: a functional definition, or a "construction rule" definition. Using the functional definition, a system is defined as "a constraint on variation" [1]. The construction rule definition of a system is "a relationship mapped over a set of objects" [2]. A system of systems is an assemblage of components that individually may be regarded as systems and possess the following two properties: 1) the component systems are able to operate independently, and 2) component systems are separately acquired and integrated as well as maintaining a continual operating existence independent of the system of systems.

Complexity is a measure of the difficulty and/or effort and/or resources required for one system to effectively observe, communicate and/or interoperate with another system. In general, there is a scale of complexity types ranging from cognitive complexity to computational complexity (see Figure 1).

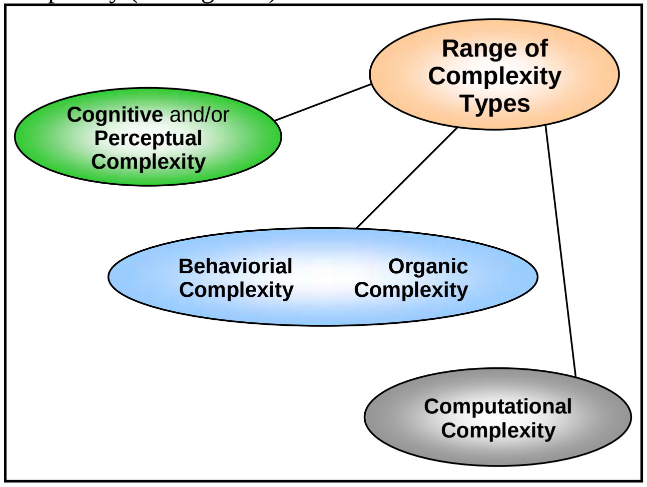


Fig. 1. Complexity Types

Cognitive and/or perceptual complexity is associated with how difficult it is for a human to clearly understand the situation. Computational complexity is associated with well-defined problems that efficiently use computational hardware, memory and time. Populating the range between computational complexity and cognitive complexity are a number of complexity types including behavioral complexity that is focused on concurrent, reactive and distributed system behavior and organic complexity that is associated with natural organic systems that have multiple areas of knowledge and information that drive very complex interactions.

The N Squared Chart is an implementation tool and methodology for the tabulation, definition, analysis and description of functional interactions and/or interfaces. A design structure matrix (DSM) is a graphical system modeling tool with associated processes and methodology that is used to communicate the system design structure.

1.1 Abstract Relation Types

Abstract relation types were developed to support the description, evaluation and communication of information associated with systems and system of systems [3]. Figure 2 shows the primary matrix components of an abstract relation type; structure space, value space and outcome space. The abstract relation type separates the encoding of system structure from the encoding of system value using the structure space and value spaces. The outcome space is used as a value integration space and/or transformation space depending on the system model and system problem.

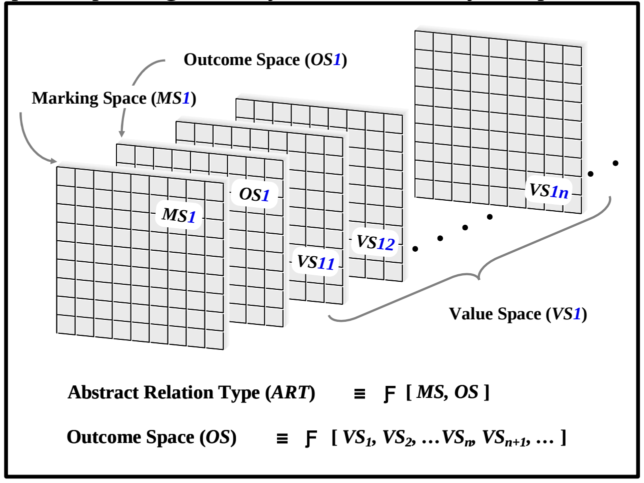

Outcome Space (OS1)
Marking Space (MS1)
MS1
OS1
VS11
VS12
VS1n
Value Space (VS1)
Abstract Relation Type (ART) ≡ F [MS, OS]
Outcome Space (OS) ≡ F [VS1, VS2, …VSn, VSn+1, …]

Fig. 2. Abstract Relation Types

The specific set of computational methods used in each instance of an abstract relation type provides the unique behavior and operations that are valid for that specific abstract relation type. Many mathematical techniques and systems engineering tools that contain a binary matrix representation of the system of interest can be readily translated into the abstract relation type form. In its most basic form an abstract relation type is a pattern used to formally define a system and/or system of systems. In this paper N Squared Charts and Design Structure Matrices will be addressed in terms of the standard abstract relation type form and analyzed for their ability to identify and control complexity of different types.

1.2 N-Squared Charts

N Squared Charts are a well defined methodology and implementation tool used to facilitate the identification, communication and documentation of system and system of systems interfaces, activities, interactions and behaviors. Using the classical N Squared tool and fixed-format methodology large amounts of detailed system information can be quickly communicated to a wide range of design and development engineers and managers from a range of mixed technical disciplines. This communication ability is based on the graphical display of the bidirectional interrelationships between the physical components and functions that reside in any given system structure. [4 - rjlano]. Manual functional analysis and restructuring techniques are used in the classical N Squared Chart approach to group physical components and interface functions and behaviors into a system with the same functions and behaviors but with fewer components and functions. The reduction in the number of system components and interface functions creates a measurable decrease in cognitive and perceptual complexity associated with the system. Figure 3 shows an example of a reduced complexity representation of the same system.

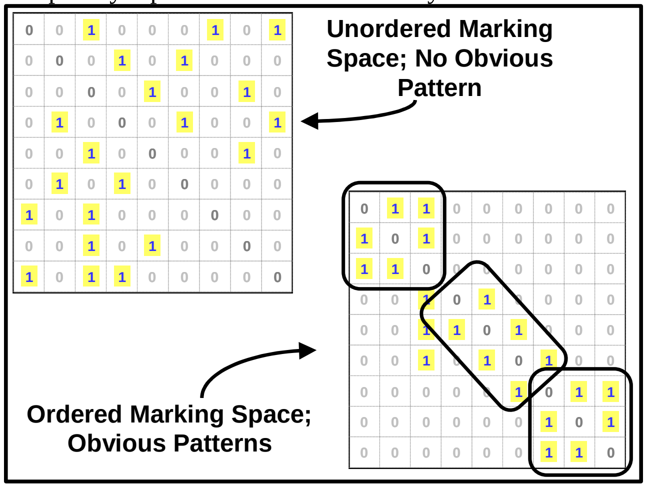
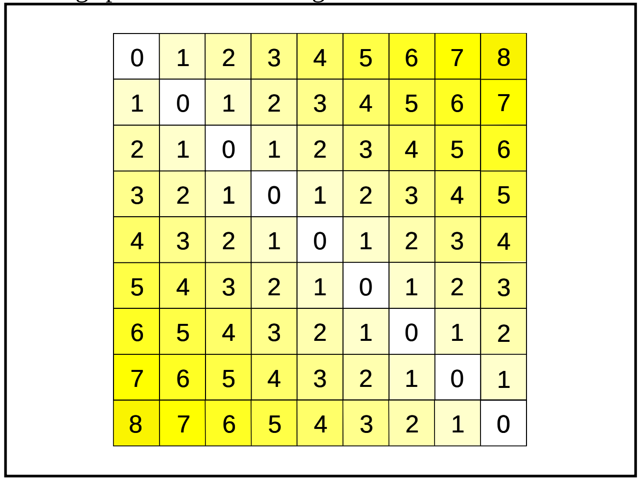

Unordered Marking Space; No Obvious Pattern
Ordered Marking Space; Obvious Patterns

Fig. 3. Reordered N Squared Chart

The N Squared Chart techniques were coupled with evolutionary computational approaches in work published by Hitchins to create a “automated N Squared Chart” technique. [5]. The addition of the evolutionary computation addressed major cost issues with the analysis of N Squared Charts buy eliminating many of the most difficult analysis tasks that were performed by human analysts. Specifically, these difficult tasks are generating valid alternative system grouping from the initial detailed system description as well as the identification and clustering valid functional aggregations. For these tasks the evolutionary computational approach creates clusters and/or groupings that must be validated by the human design team. Now the perceptual complexity associated with the task of system clustering and aggregation has been greatly reduced and/or eliminated. In many practical cases the most difficult step is the generation of the initial system configurations.

While the classical N Squared Chart techniques could be represented by the structural marking space component of an abstract relation type, the automated N Squared Chart approach would need to be used to address the value spaces and computational methods components of the abstract relation type. The value space that is coupled with the marking space is shown in Figure 4.



0	1	2	3	4	5	6	7	8
1	0	1	2	3	4	5	6	7
2	1	0	1	2	3	4	5	6
3	2	1	0	1	2	3	4	5
4	3	2	1	0	1	2	3	4
5	4	3	2	1	0	1	2	3
6	5	4	3	2	1	0	1	2
7	6	5	4	3	2	1	0	1
8	7	6	5	4	3	2	1	0

Fig. 4. N Squared Chart Value Space

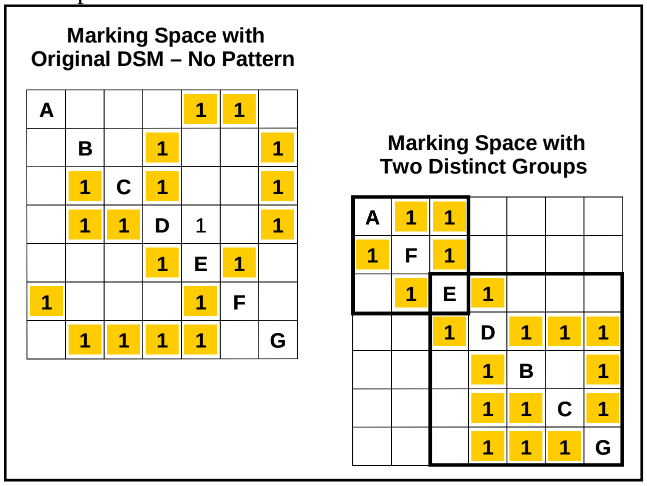
1.3 Design Structure Matrices

Similar to the N Squared Chart technique, Design Structure Matrix techniques are a classical systems analysis and management technique developed from work focused on the reduction of the computational complexity associated with the execution and solution of set of linear equations [6]. Starting from the basis of computational complexity reduction in systems of linear equations Design Structure Matrix techniques were developed to facilitate the solution of large-scale systems problems using graphical communication techniques coupled with a set of computational rules and techniques that are designed to identify system clusters and the highest value system configurations in a manner similar to the automated N Squared Chart techniques presented by Hitchins. It is interesting to note that the Design Structure Matrix techniques were developed and reported about 25 years before the automated N Squared Chart techniques.

Design Structure Matrix techniques appear to have a varied and non-standard set of applications in the literature. While the matrix diagonal is always identifiable from the definition of a matrix there are multiple different conventions and implementations that assign meaning to the upper and lower triangular areas of a Design Structure Matrix. One of the perceived values of a using the abstract relation type pattern is to encode and document the specific process steps as well as the syntax and semantics associated with a specific Design Structure Matrix approach and application area.

One kind of Design Structure Matrix called a dependency structure matrix is used in the development of new products and the analysis of schedules associated with product development. An example of system partitioning and system clustering from the literature will be examined here to illustrate the application of abstract relation types to the various forms of the Design Structure Matrix technique [7].

Clustering of components in the Design Structure Matrix can be done in a manner similar to that used in the automated N Squared Chart technique. In this clustering method the value space associated with the system under consideration is symmetric with the same type of values in both the upper triangular and lower triangular sections of the matrix. Figure 5 shows a clustering example from the literature. The reported result has been reproduced using the abstract relation type techniques that used different value space sets.


Marking Space with
Original DSM – No Pattern
Marking Space with
Two Distinct Groups

A				1	1	
	B		1			1
	1	C	1			1
	1	1	D	1		1
			1	E	1	
1				1	F	
	1	1	1	1		G

A	1	1				
1	F	1				
	1	E	1			
		1	D	1	1	1
			1	B		1
			1	1	C	1
			1	1	1	G

Fig. 5. Design Structure Matrix Clustering

In another example from the literature, a twelve (12) by twelve (12) matrix was used as an example to illustrate the partitioning of a dependency structure matrix. This example was interesting from a number of perspectives and provided some insight to the application of both global and local relationships relations in a system structural model.

The example presents the same system in two different structural forms, the base model and the partitioned model. Upon close examination of the base model and the partitioned model it becomes clear that there are a different number of system dependencies and/or interfaces in the two graphical representations of the system. The question then becomes are they both correct or is one correct and the other in error? If one graphical system representation has an error which representation contains the error?

The above questions are answered using the global definition of a system, “a system is represented by the relations mapped over the objects.” For the matrix graphs to represent the same system, the graphs must have the same number and type of interface relations. The partitioned system matrix that presents 30 system interconnections is taken to be the correct system structure. Now the base model correction, that has only 28 interconnections, must be accomplished using the same global system rule. The base structure model is missing the J,F connection and the F,J connection. The missing elements were defined by comparing the two matrices for consistency in connection type.

One source of complexity reduction that is accomplished by using the abstract relation types is the clear separation of the system structure representation from the system value representations using the marking space matrix and set of value space matrices.

When the system problem under consideration has a higher value assigned to either the upper or lower triangular section of the value space systems problems that address sequential operations, signal flow and/or hierarchical system design components can be effectively addressed, analyzed and evaluated. This type of analysis and system evaluation was introduced over 35 years ago by Warfield. [8]

2 Complexity Identification and Control

Both of the classical systems engineering techniques discussed in the first part of this paper reduce the cognitive complexity associated with the development of large scale systems by the introduction of clear processes and standard graphical representations of static system structure and/or behavior. In the case of a system of systems an individual system may have the power to either join an existing system of systems or disconnect from a system of systems in which it has been participating. The ability to quickly model and evaluate a large set of structural system components that represent both global and local system relations and values appears to provide an advantage in the operation of large scale system of systems.

Conceptual complexity is addressed using graphical communication techniques that are applied using standard processes. Both the standardization of the processes and the graphical representation work together to reduce the complexity of any given problem. The Design Structure Matrix example taken from the literature will be presented next using the abstract relation type pattern matric components.

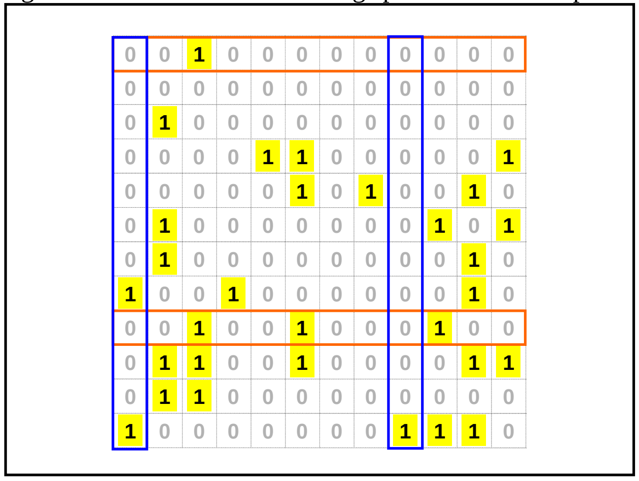
2.1 A Design Structure Matrix Abstract Relation Type

Abstract relation types are composed of a set of artifacts that document the specific abstract relation type's pattern of application as well as the computational methods, matrix components and other processes used in the effective application of this type of analysis tool. As shown in Figure 2, the main set of abstract relation type matrices are the marking space matrix, the outcome matrix and the value space matrices. The evolutionary computation method associated with abstract relation types will be used to detail the content of each of these components. There are many other computational methods that can also be used in abstract relation types that will not be considered here due to lack of space.

The evolutionary computation approach is based on the concept of randomly creating candidate system component arrangements and grouping them into a population. Each member of the population is evaluated using a “best-fit function.” The population members that score the best are then used as the basis to create the next generation population. This process is repeated until some stopping criteria are achieved. The evolutionary computational approach used here has a process associated with the marking space to generate candidate system structures and an evaluation process associated with the value spaces that is used apply the best fit function to the matrix structure. The marking space process will be detailed in the next section and then the best fit function evaluation process will be presented.

2.2 Marking Space Contents and Process

The marking space represents the structure of the organizing system relation. In the example considered here the organizing relationship is “is connected to.” The “is connected to” relationship is transitive and bidirectional. Figure 6 shows the initial marking space for the example.



0	0	1	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	1	0	0	0	0	0	0	0	0	0	0
0	0	0	0	1	1	0	0	0	0	0	1
0	0	0	0	0	1	0	1	0	0	1	0
0	1	0	0	0	0	0	0	0	1	0	1
0	1	0	0	0	0	0	0	0	0	1	0
1	0	0	1	0	0	0	0	0	0	1	0
0	0	1	0	0	1	0	0	0	1	0	0
0	1	1	0	0	1	0	0	0	0	1	1
0	1	1	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	1	1	1	0

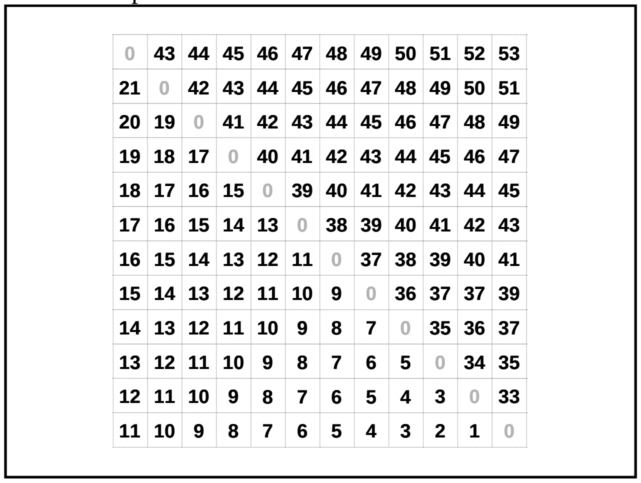
Fig. 6. Initial Marking Space

The diagonal of the marking space must always contain a zero (0) as a component is not allowed to connect to itself. The computational process randomly selects two numbers and checks to verify that the rows and columns represented by the selected numbers can be exchanged while maintaining a valid system structure. As shown in Figure 6, the numbers one (1) and nine (9) were randomly selected. Therefore, row one (1) and row nine (9) (both outlined in red) are exchanged as well as column one (1) and column nine (9) (both outlined in blue). This random mutation of the system structure is the basis for generating increasingly better system populations.

2.3 Value Space Contents and Process

A key component of the abstract relation type is the set of value spaces. An individual abstract relation type can have one or more value spaces that are either static or dynamic in nature. A static value space does not change its value arrangement during the computational cycle. A dynamic value space will change its value arrangement during the computational cycle. This example has one static value space that could be represented by the summation of three or more static values spaces. Only the one static value space is presented here to due to restricted space.

The best fit function and the value spaces are designed together in a manner that properly represents the global system relationship and the specific system problem being addressed. In this example the global system relationship is “is connected to”, while the specific system value set addresses a feed forward sequential system that encodes the forward link in the lower triangular section of the matrix. The best fit function is a minimization function, so the marking space structures that generate the lowest scores are the best structures. Figure 7 presents the static value space of this example.



0	43	44	45	46	47	48	49	50	51	52	53
21	0	42	43	44	45	46	47	48	49	50	51
20	19	0	41	42	43	44	45	46	47	48	49
19	18	17	0	40	41	42	43	44	45	46	47
18	17	16	15	0	39	40	41	42	43	44	45
17	16	15	14	13	0	38	39	40	41	42	43
16	15	14	13	12	11	0	37	38	39	40	41
15	14	13	12	11	10	9	0	36	37	37	39
14	13	12	11	10	9	8	7	0	35	36	37
13	12	11	10	9	8	7	6	5	0	34	35
12	11	10	9	8	7	6	5	4	3	0	33
11	10	9	8	7	6	5	4	3	2	1	0

Fig. 7. Static Value Space

Given a specific marking space configuration and the static value space configuration, a cell by cell multiplication of one marking space cell with one value space cell provides the set of values that are summed to create the final score for a given marking space arrangement. Therefore only the marking space cells that contain a one will produce a value greater than zero and add to the final score total.

Many different types of value spaces and value space combinations are available to address system clustering, efficient process and/or event scheduling as well as assist in the analysis of large scale interface connection sets. Abstract relation types were developed to support the development, documentation and cataloging of these system patterns. While the general abstract relation type is an abstract pattern, each specific instance of an abstract relation type is a concrete system pattern that provides the basis for a pattern language.

2.4 Complexity Control

As discussed above, different types of complexity can be effectively managed in large distributed team using the classical N Squared Charts and Design Structure Matrix methods and techniques to reduce cognitive complexity. The ability to interactively control cognitive and behavioral complexity during the design, development and operation of large scale systems and system of system is enhanced by using the concepts, format and patterns associated with abstract relation types. Using automated techniques to monitor the system interconnections and encode the interconnection information into a format that can be evaluated by the evolutionary methods associated with the “is connected to” abstract relation type creates a situation where only the most probable system structural configurations are evaluated for further action.

Similar to the libraries of abstract data types that are the foundation for most programming languages, libraries of abstract relation types are viewed as the foundation for a system science and systems engineering language. The Science of Generic Design created by Warfield is the theoretical basis upon which the application of real world relationships, mathematical relations and abstract relation types is built [9]. Using these techniques many types of complexity can be effectively placed under control and reduced.

3 Summary

Many classical systems engineering techniques reduce cognitive complexity by focusing on the local relationship between the components and the associated values of the local interrelationships. The specific real-world global organizing relationship presented in this paper is the “is connected to” relationship. The abstract relation type presented here is then considered an Is-Connected-To abstract relation type. The combination of the global organizing relationship combined with local interrelationships of the system components that can be represented by the abstract relation type provides a basis for a common standard approach to the documentation, evaluation and analysis of large scale complex systems. The evolutionary computation method discussed here has the potential to greatly reduce the cost of applying the classical systems engineering methods as well as expanding the area in which these and similar types of systems science and systems engineering techniques can be applied.

As the abstract relation type concept and method is developed and refined, libraries of these system solution patterns can be developed and used as the basis for a systems science and systems engineering pattern language. The executable portions of the abstract relation types present the possibility for the construction of an executable systems science and systems engineering language based on the pattern language.

4 Conclusions

Abstract relation types offer great potential in reducing many types of complexity. In this paper the reduction of cognitive and perceptual complexity was addressed. More research is needed in the ability of abstract relation types to reduce computational complexity that is associated with any specific type of system problem and or system solution pattern.

Ordering and cataloging the wide range of disparate system analysis and evaluation methods that are based on the concept of a binary relation will further reduce cognitive complexity by presenting a well-ordered, documented and organized set of methods.

References

[1] Heylighen, Francis, “(Meta)Systems as Constraints on Variation – A classification and natural history of metasystem transformations”, Free University of Brussels Research Paper, sponsored by Belgian National Fund for Scientific Research (NFWO), Brussels, Belgium, 1994.

[2] J.J. Simpson and M.J. Simpson, "Formal Systems Concepts," *Proceedings, Fourth Annual Conference on Systems Engineering Research,* Los Angeles, California, April, 2006.

[3] J.J. Simpson, C. Dagli, A Miller, “Development and Application of Abstract Relation Types for Use in Systems and System-of-Systems Design and Evaluation.” *Proceedings, Seventeenth Annual International Symposium of INCOSE, "Systems Engineering: Key to Intelligent Enterprises,"* San Diego, California, June, 2007.

[4] Lano, R.J., *A Technique for Software and Systems Design,* TRW and North-Holland Publishing Company, Amsterdam, 1979

[5] Hitchins, Derek K., *Advanced Systems Thinking, Engineering and Management*, Artech House, Inc., London, England, 2003.

[6] Steward, Donald V., *Systems Analysis and Management: Structure, Strategy and Design,* PBI, A Petrocelli Book, New York, 1981.

[7] D.M. Sharman and A.A.Yassine, “Characterizing Complex Product Architectures,” *Systems Engineering Journal*, Vol 7, No. 1, pp. 35-60, 2004.

[8] John N Warfield, *An Assault on Complexity, Monograph No. 3,* Battelle Memorial Institute, Columbus, Ohio, 1973.

[9] John N Warfield, *A Science of Generic Design,* Iowa State University Press, Ames, Iowa, 1990,