Complexity Reduction: A Pragmatic Approach

Joseph Simpson, Mary Simpson
System Concepts, LLC

Abstract

An increasing need for shared communication in disparate domains as well as the production of increasingly dynamic, large-scale systems has always been at the heart of the practice of systems engineering. As a branch of general systems theory, systems engineering was developed to address practical considerations posed by diverse organizations, environments and cultures within which systems are designed, developed and operated. Types and categories of complexity are used in this paper to focus the discussion on complexity and the reduction of complexity. Formal and theoretical foundations of systems science and systems engineering provided the basis upon which many effective systems engineering tools were built. This paper identifies some of the classical tools of systems science and systems engineering that manage complexity. Based on these classical tool components and principles, abstract relation types (ART) were developed to enhance the understanding and application of these tools. A pragmatic approach that is designed to reduce complexity as well as compare relative complexity reduction between and among methods is also presented. The direct value of systems engineering techniques as they are applied in any context is rooted in the ability of systems engineering techniques and systems engineering practitioners to reduce the cognitive complexity associated with the systems problem of interest.

Introduction

Systems engineering was developed to specifically address problems of complexity found in the environment, systems design, deployment, and operations. Systems engineering uses engineering technology teams and team leaders to effectively design, develop and deploy large-scale systems. Formal concepts and theoretical foundations of systems engineering were developed by many thought leaders in systems science and engineering. Key fundamentals for complex systems engineering analysis, design, development and operation include the Language for a Generic Design Science; the Law of Triadic Compatibility with its Principle of Division by Threes, and the Law of Gradation and its three corollaries: the Corollary of Congruence, the Corollary of Diminishing Returns and the Corollary of Restricted Virtual Worlds [Warfield, 1990, 1994]. Though there are other fundamentals, these particular formal and theoretical aspects of complex systems, systems science and engineering form the bases for a number of practical systems engineering tools and analysis approaches, and will be explored in this paper. Specific classical systems engineering techniques and approaches addressed in this paper are N Squared Charts (N2C) [Lano, 1979], Automated N Squared Charts (AN2C) [Hitchins, 1992], Design Structure Matrix (DSM) methods [Steward, 1981], and the Interpretative Structural Modeling (ISM) approach [Warfield and Cardenas, 1993, 2002]. Each of these recognized techniques will also be discussed and addressed as cognitive and/or computational complexity reduction mechanisms. The ART approach is used in this paper as a framework to discuss and compare the complexity reduction and complexity management aspects of these techniques. In addition, the reduction of cognitive complexity will also be discussed through the use of a structured analysis methodology for security. These techniques are presented as fundamental systems engineering tools that can be adapted and generalized to support the application of evolutionary algorithms and abstract relation types to the reduction of complexity associated with the realization of large-scale, dynamic systems [Simpson and Simpson, June, 2009].

Definition of Terms

Complexity is defined as the measure of the difficulty, effort and/or resources required for one system to effectively observe, communicate, and/or interoperate with another system [Simpson and Simpson,

2009, p. 2]. The preceding definition of complexity was developed to include a range of complexity types that are identified and described in systems literature. The difficulty encountered by a human in the process of clearly understanding a given problem or situation is strongly associated with cognitive and perceptual complexity. Considerations associated with the efficient or feasible use of computer memory, hardware and time resources as applied to specific classes of computational problems, are referred to as computational complexity. Systems engineering addresses many types of complexity using a variety of well-defined processes and mechanisms. Cognitive and perceptual complexities are based on two definitions of complexity given by Warfield. An early definition offered: "Complexity is a property of a system, occasioned by a variety of factors germane to the system, which complicate human perception of the system and consequently make its understanding difficult" [Warfield, 1974, p. 1-1]. Warfield's evolved definition provided "Complexity is that sensation experienced in the human mind when, in observing or considering a system, frustration arises from lack of comprehension of what is being explored" [Warfield, 2002, p. 20]. Bar-Yam defined complexity as a measure of the inherent difficulty to achieve a desired understanding, and indicated: "Loosely speaking, the complexity of a system is the amount of information necessary to describe it" [Bar-Yam, 1997, p. 12]. A relativistic view of complexity was given by Casti as, "… the complexity of a given system is *always* determined by some other system with which the given system operates" [Casti, 1989, pp. 17-18]. Organic complexity is associated with living systems which are complex by definition. Behavioral complexity is associated with the dynamic behavior of systems over a period of time. Computational complexity is generally associated with well defined algorithmic problems and the efficient solutions for these stated algorithmic problems [Sipser, 1997; Harel, 1987]. General categories of complexity have been depicted within a range indicating cognitive and perceptual complexity on one end, computational complexity on the other end, and other types of complexity populating the space between these two endpoints (see Fig. 1).

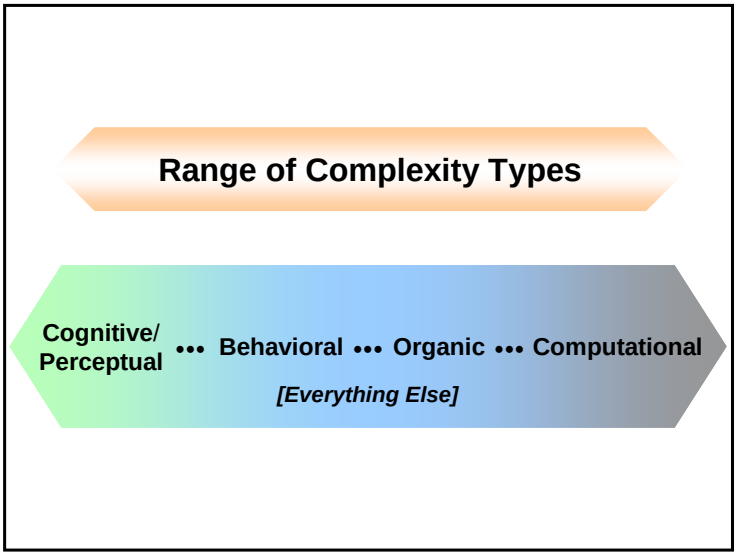

Figure 1. Complexity Categories

Complex systems are defined by carefully identifying a system, and then addressing system configurations or operations that make the system complex. A system is a set of objects with relationships between the objects and between the attributes of those objects. System objects are the components, parts, and/or segments of a given system that are connected in some manner to create the structure of the system. The term relationship is a natural language interpretive term that is used to describe the organizing or system structuring concept that forms a set of objects into a system. The term relation refers to a formal mathematical concept. Evolutionary computation is a field of

computation that is based on random variation of a population and the selection of the best individuals using evolutionary algorithms. A system is organized and structured by the relationships that bind a given set of objects into a useful, functional construct. System objects have attributes that are properties of the objects [Hall 1962; Simpson, 2006]. A group of objects is not a system when there is no relationship among or between the objects that can be used to define the system structure. The concept of a complex system is not well defined. However, many aspects of a system that would make it complex have been identified and addressed in systems engineering literature [Hall, 1989]. One aspect of system complexity is the number of objects that comprise a given system. A second aspect of a complex system is the number of different types and/or classes of objects used in the complex system. A third aspect of system complexity is the type and number of relationships associated with any given complex system. System complexity is defined by Steward as "Given the parts and their behaviors, complexity is the difficulty involved in using the relations among the parts to infer the behavior of the whole. Or phrased another way, complexity is how much more the whole is than just the sum of its parts." [Steward, page 2] In addition, system relationships can be static or dynamic. In general systems theory, Boulding [1956] used a hierarchy of complexity, or complex system types, to create a theoretical framework to organize and define a system of systems. This theoretical framework was used to explore basic aspects of complex system and general systems theory. His proposed hierarchy of complexity had nine levels as follows: 1) frameworks with static structure; 2) clockworks with predictable actions; 3) cybernetic systems with control within limits; 4) self-maintaining systems with the ability to reproduce the system; 5) plants; 6) animals; 7) human beings; 8) social organizations; and 9) transcendental systems.

Key Fundamentals

The Language for a Generic Design Science (LGDS) is specifically designed to provide direct support for the limited information processing capability of the human mind [Warfield, 1994]. Three foundational language types (prose, structural graphs, and mathematics) are used individually or in combination to support system design activities using the LGDS. Prose statements are used to provide incremental meaning in a group of statements. Structural graphics provide a mechanism to provide metrics for decision making, document knowledge organization and assess the compatibility of knowledge organization types. Mathematics provides a compact information representation and notation as well as clearly communicating transformations and manipulation activities. The combination of prose and structured graphics makes complex relationships highly visible and provides overview, meaning, perspective and integration communication capability. The combination of prose, structural graphics and mathematics provides maximum flexibility in communication as well as facilitating the optimization of communication and reasoning about complex systems. Each of these combinations of language types was defined to provide different functions [Warfield, 1994, p. 48]. Figure 2 presents a mapping between a set of classical systems engineering techniques, and the fundamental language types. The set of classical systems engineering techniques are further addressed later in the paper.

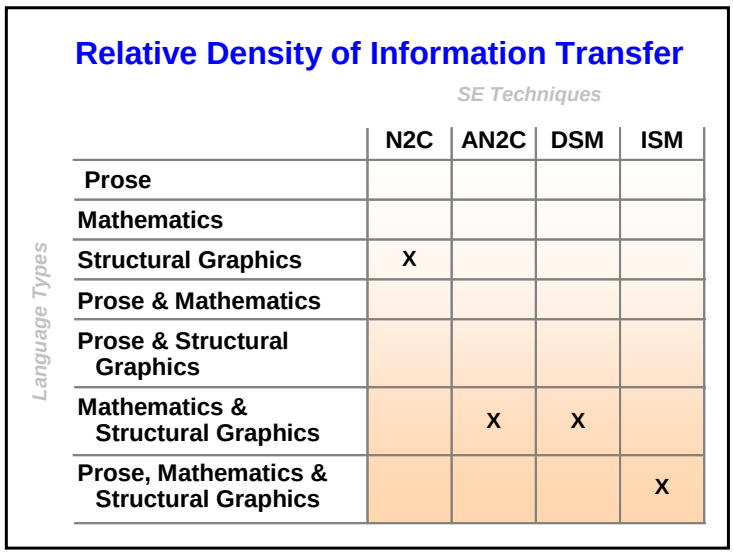

Figure 2. SE techniques mapped to language types.

The Law of Gradation (LOG) "asserts that any conceptual body of knowledge can be graded in stages, such that there is one simplest stage, one most comprehensive stage (reflecting the total state of relevant knowledge), and intermediate stages whose content lies between the two extremes" [Warfield, 1994 p. 178]. The first Corollary to the LOG is the Corollary of Congruence which "asserts that the class of situations to which a conceptual body of knowledge may apply, in whole or in part, likewise may be graded according to the demands that individual situations can reasonably make upon the body of knowledge" [Warfield, 1994 p. 179]. The second Corollary to the LOG is the Corollary of Diminishing Returns which "highlights a major responsibility of the designer to make judgments about when this point is reached" [Warfield, 1994 p. 179]. The third Corollary of the LOG is the Corollary of Restricted Virtual Worlds which "states that the identification of the stage at which diminishing returns to the situation is reached normally requires the integration of the Virtual Worlds of the affected parties … " [Warfield, 1994 p. 180]. This law and its corollaries speak to the need to avoid mindless application of an entire litany of theory and methodologies to every design situation and set of solutions. Instead, it depends on the human being to think about, and apply good judgment to the given situation and its intended outcomes.

The Law of Triadic Compatibility (LTC) is focused on quantifying and further defining the processing limit of the human mind. The LTC states: "The human mind is compatible with the demand to explore interactions among a set of three elements, because it can recall and operate with seven concepts, these being the three elements and their four combinations; but capacity cannot be presumed for a set that both has four members and for which these members interact" [Warfield, 1994, p. 45]. The Principle of Division by Threes (PDT) is a Corollary of the LTC. The PDT asserts: "Iterative division of a concept as a means of analysis is mind compatible if each division produces at most three components, thereby creating a tree with one element at the top, at most three elements at the second level, at most nine at the third level, and so on" [Warfield, 1994, p. 45].

An Example Application of the LTC and PDT: System Security Analysis

Government communications, computer and information systems have experienced substantial changes beginning in the late 1980's, and continuing on until the present time. In 1991, a three dimensional construct was developed to represent a theoretical model for the information security application domain. Even with substantial changes in communications, computer and information technology, and practices, this theoretical model has proven effective over the ensuing years. This graphical model is

clearly a structured graphic model that aligns directly to the LTC and PDT aspects at the foundation of formal systems engineering practices. This structured graphic is used to encode domain knowledge associated with the protection of information assets, and is organized around the attributes and application of an organization's information. Each of the three dimensions in this cube is further divided into three layers. The intersection of these sides and layers generates 27 smaller component cubes that can be viewed in groups of three or nine to locate and identify a range of specific characteristics associated with information security practice [McCumber, 1991].

The information cube model (ICM) is expanded in this paper to provide an example of an application of LTC and PDT analysis. The asset protection cube (APC), shown in Figure 3, is a new theoretical model for asset protection, and has been developed and is being applied to the protection of assets. At this high level of abstraction the APC model is domain independent and provides three primary components: the system, the threat, and the target. Using the PDT, each of these Asset Cube components is assigned to another lower-level cube, and then these components are divided into specific subcomponents. The system cube is divided into the type of system, system specification and program components. The system type is then further divided into product system, process system and environment system. System specification is further divided into system function, system requirements and system architecture. The system program is divided into cost, schedule and technical categories. The threat cube is divided into exposure, action and effect. The threat exposure is divided into threat actor, threat mechanism and threat vector. The threat action is divided into pre-event, event, and post-event. The threat effect is divided into immediate effect, near-term effect and long-term effect. The target cube is divided into configuration, value and protection. As noted the target cube is domain-specific and the categories from the previously referenced ICM will be used. The target configuration is divided into transmission, storage and processing. The target value is divided into confidentiality, integrity, and availability. The target protection is divided into technology, policy and human factors. Each of these cubes is structured to support human reasoning and conform to LTC. If the system evaluation and analysis task must include the consideration of relationships and structure from more than one cube, then a mathematical relation must be developed and applied using computer programs and applications to effectively address the problem solution in a consistent and repeatable fashion.

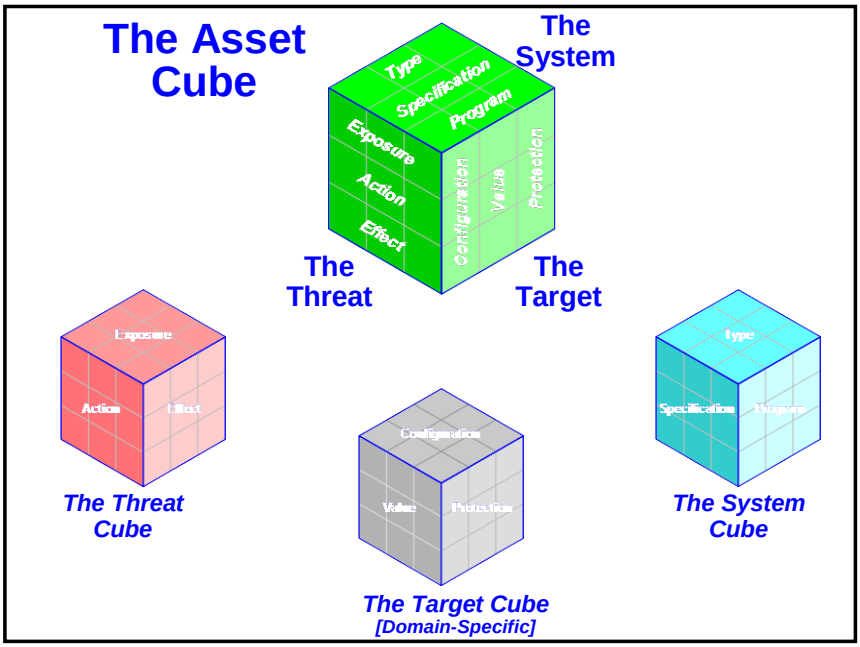

Figure 3. The Asset Protection Cube as an expanded example of the Law of Triadic Compatibility and its corollary, the Principle of Division by Threes.

Classical Systems Engineering Analysis Techniques

Three general classes of systems engineering evaluation and analysis techniques are addressed in this paper. These evaluation and analysis techniques are addressed using the two primary components of a system: a set of objects, and the system relationship mapped over the objects. The first general technique is N Squared Charts (N2C) developed by Lano to address cognitive complexity associated with large-scale system design, and includes its evolved technique of automated N Squared Charts (AN2C) developed by Hitchins. The second general technique is design structure matrices (DSM) developed by Steward to address problems associated with computational complexity. The third general technique is Interpretative Structural Modeling (ISM) developed by Warfield to address the cognitive complexity associated with the determination of an unknown system structure. A common characteristic shared by these three techniques is a graphical representation of the system structure under consideration using a type of structural graphics. While the communication power of the graphical components of the DSM and N2C techniques appear to have been independently discovered and developed, the graphical system structural component associated with the ISM methods were designed as part of the ISM technique. Further, the N2C uses only structural graphics, while the AN2C and DSM uses structural graphics combined with mathematics and the ISM technique uses prose, structural graphics and mathematics.

N Squared Charts

N2C is a well-defined methodology and systems analysis tool used to facilitate the identification, communication and documentation of a system and its interfaces, activities, interactions and behaviors. Using the classical N2C tool and fixed-format methodology, large amounts of detailed system information can be effectively communicated to a wide variety of design and development engineers and managers from a range of mixed technical disciplines. This ability to communicate is based on the graphical display of the directional relationship between the system objects that reside in any given system structure [Lano, 1979]. Manual functional analysis and restructuring techniques are used in the classical N2C methods of analyzing system objects, relationships and their interfaces. The N2C approach method is flexible, allowing functions, people, subsystems and processes to be used as objects in the N2C visual representation. One goal of these manual analysis techniques is the identification of a system with the same local interface functions and behaviors, but with fewer system objects and object interfaces. The reduction in the number of system objects and interface functions creates a measurable decrease in cognitive and perceptual complexity associated with a given system. Examples of reduced cognitive complexity using an aggregate system representation have been reported in the systems engineering literature [Simpson and Simpson, April, 2009]. Figure 4 presents an example of cognitive complexity reduction using N2C.

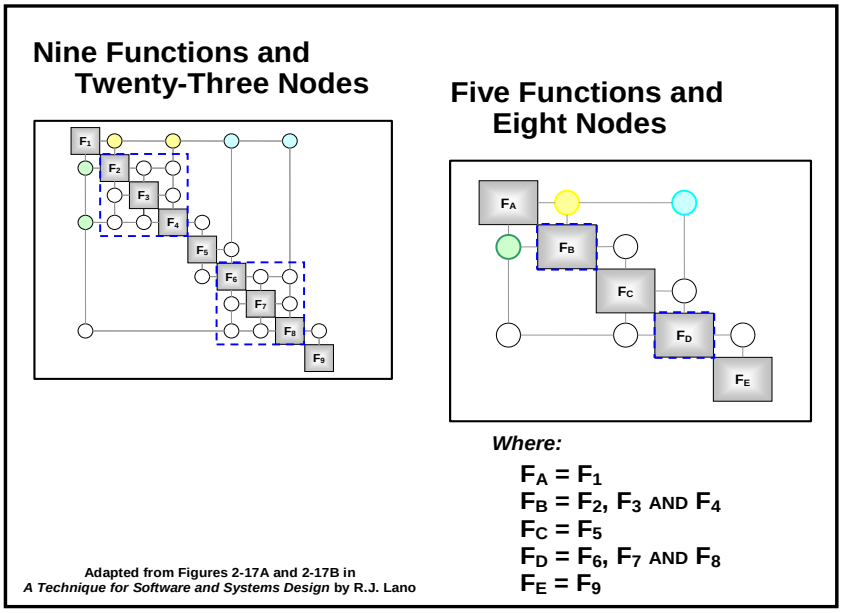

Figure 4. Example of cognitive complexity reduction using N^2 Charts.

N2C were designed as a tool to address software and system functional interfaces. They are not an automatic solution for the problem of interest. Only a careful, consistent and methodical application of the N2C technique will provide useful insights and benefits to the system designers. Lano details strong correspondence between the N2C and system block diagrams, program evaluation and review technique (PERT) charts as well as Gantt schedule charts. Given the power of this structural graphic form to effectively represent diverse system and program management domain information, it was successfully used to focus and communicate information across a diverse group of people. The greatest benefit of this powerful tool is the identification and aggregation of the controlling system concepts across the system interfaces. The structured definition, tabulation, analysis and documentation capabilities of the N2C approach greatly aid and support standard systems engineering tasks of design and analysis [Lano, 1979].

N2C techniques are coupled with evolutionary algorithm computational approaches in work published by Hitchins to create an "automated N Squared Chart" (AN2C) technique [Hitchins, 2003]. Combining evolutionary computational techniques with the N2C eliminates many of the most difficult analysis tasks originally performed by human analysts in the application of N2C, and simultaneously allows those critical human resources to focus on other key activities. The specific difficult tasks that are greatly reduced or eliminated include the generation of valid alternative system groupings from the initial detailed system description, as well as the identification and clustering of valid functional aggregations. Using the AN2C, the evolutionary computational approach creates clusters and/or groupings which are then validated by the human design team. The perceptual complexity associated with the task of interface clustering and aggregation has been greatly reduced and/or eliminated by applying the AN2C techniques. In many practical analytical cases, the most critical and difficult step is the generation of the initial system configurations [Simpson and Simpson, June, 2009].

While Hitchins introduced the AN2C concept, the specific solution approach considered here was developed and refined by Simpson [Simpson and Simpson, April, 2009]. Both the N2C and the AN2C techniques represent their system objects on the diagonal of a given matrix. An interface between system objects is indicated in the matrix cells that connect those two objects. In this manner, the system objects and interface connections are documented and recorded within each technique. The AN2C approach developed by Simpson uses a marking space to document and record the system object

configuration, and a value matrix to document and record the attributes associated with the system relationship. The system structure is evaluated and analyzed using the relationship represented by the value matrix, in combination with a valid object configuration, documented by the marking space.

A disciplined application of classical N2C techniques provides clear cognitive complexity reduction by reducing the number of objects and interface connections in a given system. However, the lack of standardization associated with the methods, syntax and semantics of both the N2C and the AN2C creates ambiguities that are not easily resolved by a systems engineering practitioner. This source of cognitive complexity is addressed by abstract relation types (ART) in subsequent sections of this paper. The term relation was used in the ART name to emphasize the mathematical expression of the real world organizing relationship of interest.

Design Structure Matrix Approach

Computational efficiency associated with the solution of a set of mathematical equations was one of the primary motivating aspects driving the development of the DSM technique. Initial work in the area of DSM analysis prompted the application of the technique to diverse domain areas including the process of whole system design and evaluation, project and program scheduling, economic model simulation and analysis, and chemical engineering process modeling, analysis and evaluation as well as aerospace vehicle control system analysis, simulation, and evaluation. The analytical strength and power associated with the application of the DSM approach is supported by techniques that focus on both the internal interfaces of the system as well as the total system behavior. System behaviors are associated with and assigned to the complete integrated system. Internal interfaces are associated with, and assigned to, the parts that comprise the total system. A set of well-developed, mathematical and analytical processes are also associated with, and integrated into, the DSM approach. The use of a specific DSM process and technique is determined by the system characteristics and attributes. Two of the primary attributes used in selecting the proper DSM techniques are description and prescription. System description is the process of representing the system behavior, which includes concurrent independent operations of the system parts. System description requires the use of graph circuits and feedback loops. System prescription represents how something is done; that is, a procedure that can be expressed as a tree of feed-forward connections that have no circuits.

The application of DSM techniques is grouped into four general types of system representations: task-based, parameter-based, team-based and component based [Sharman and Yassine, 2004]. The task-based system represents the input and output relationships associated with a set of tasks. The task-based approach is used to analyze project scheduling, activity sequencing and cycle time reduction using DSM partitioning, tearing, banding, simulation and Eigen value analysis techniques. The parameter-based system supports the analysis of necessary activity sequencing and parameter decision points during the analysis of activity sequencing and process construction using DSM partitioning, tearing, banding, simulation and Eigen value analysis. The team-based approach is used to represent multi-team interface attributes and arrangements during organizational design, interface management and team integration activities using DSM clustering techniques. The component-based system approach is used to analyze multi-component relationships that are key aspects of system architecting, engineering and design using DSM clustering techniques and methods. While these are the basic DSM approaches, current research is producing a range of new DSM applications, methods and techniques.

The disciplined application of classical DSM techniques provides clear cognitive complexity reduction by identifying system interface groupings that provide a system view organized around clusters of components from a given system. The lack of standardization associated with the methods, syntax and semantics of DSM creates a source of cognitive complexity that will be addressed later in the abstract relation type (ART) section of this paper.

Interpretative Structural Modeling Approach

ISM is a well developed system structuring technique used by groups to explore and create well defined system definitions. A key aspect of the ISM approach is the ability to represent a system in graphical, mathematical and prose forms. These three equivalent system forms and/or expressions were designed to identify, describe and enhance the understanding of system structure. The graphical system form provides a visual representation of the system that allows people to use their visual learning abilities to quickly understand the system form. The mathematical form provides a direct connection to computational operations that are used to analyze and reconfigure the system of interest. The prose form of the system is used to communicate the system structure in written form. These three forms are used to communicate system information to individuals who have different ways of understanding and relating to system information.

The formal and theoretical foundations of the ISM approach have been carefully detailed and communicated in the system science and systems engineering literature. The process of Interactive Management (IM) was developed and refined based on the formal and theoretical aspects of system science and engineering [Warfield, 1993, 1994, 2002]. The application of ISM and IM clearly reduce cognitive complexity by providing a well-defined body of science, theory, methods and practices that are used to move from an undefined, unstructured set of problems to a structured system that can be identified and understood by the individuals and groups participating in the given problem space.

Taken as an integrated body of work, ISM focuses the science of generic design on a specific problem and/or set of problems that are distributed across a large group of individuals and/or organizations. ISM has a rich and deep history with strong theoretical connections to systems engineering and management science. It also provides a substantial contribution in practical applications; Warfield and numerous associates have produced over 200 case studies using the methodology.

Abstract Relation Type (ART) Application

Abstract relation types were developed to support the systematic reduction of cognitive complexity associated with the design, development, documentation and operation of large-scale systems. The ART construct was inspired by two fundamental data and mathematical constructs: abstract data types and binary relations. These two constructs were then mapped to Warfield's three basic language types, prose, structural graphics and mathematics, to create a framework that supports effective detailed communications of systems concepts and computational tools associated with system design activities. The ART framework has three primary components: description, structural graphics, and computer-executable methods. The ART description component is composed of a structured pattern component as well as an unstructured prose component. These prose-based components are used to describe the system of interest as well as to document and record the processes used to evaluate the system of interest. The ART structural-graphics component contains those structural graphics used to encode and communicate the system structural features associated with the system of interest. The ART computer-executable-methods component details the processing algorithm and/or the computer code required to implement the ART activities. Taken together, these three ART components are designed to integrate detailed information about the system of interest in a manner that creates a well-defined, executable system description and evaluation tool [Simpson CSER 2007].

The structured description component of the ART is based on the systems engineering pattern that has a specific name and contains five parts: problem statement, forces, context, related patterns and solution [Simpson and Simpson, July, 2006]. The pattern information is written out using prose and graphics. The name of the pattern component of the ART is chosen to highlight the specific ART content and application. The problem statement section of the ART description expresses the primary difficulties and uncertainties associated with the system and/or situation of interest. The forces component of the

ART description is where the tensions and/or key drivers that influence the choice and application of the ART are documented and described. The context component of the ART description is where the environmental, organizational and other important aspects of the ART application context are described and documented. The related patterns component of the ART description lists the patterns that are connected and associated with the current ART area of application. The solution component of the ART description expresses the methods and mechanisms that are used to resolve a specific problem within a specified context. The unstructured description component of the ART is provided as an area to record prose information about the ART application area and problem structure. The combination of the structured and unstructured description components establishes areas for a prose description of the system problem as well as candidate system solutions approaches.

The structural-graphics component of the ART provides an area within which graphical representations associated with the system problem statement and/or system solution approach are recorded and presented. The structural-graphics component associated with the application of genetic algorithms is highlighted in this paper to demonstrate one type of structural graphic application. There are many standard graphical patterns that have been developed, and can be used as structural graphics. These standard ART graphical patterns can be combined with the prose-based ART description patterns to further enhance and develop an ART-based pattern language. The AN2C and DSM structural-graphic representations of problem areas and problem solutions have been used as the foundation of an expanded structural-graphics approach that focuses on two basic aspects of the system problem. The first aspect is the representation of the objects in the system structure. The second aspect is the representation of the relationship that binds those objects as a part of the system. The combination of these two aspects provides a mechanism to represent the system in any given context.

The ART structural graphics approach uses a matrix, called a marking space, to document and communicate the system structure by encoding each interface between system objects with a value of one (1) in the marking space (see Figure 5). A second matrix, called a value matrix, is used to represent the attributes of the organizing system relationship in a mathematical representation. In the examples presented here, the organizing system relationship is identified as “is-connected-to.” This abstract, high-level, real-world relationship provides all the necessary information required to properly populate the values in the value matrix. Three properties of the real-world relationship are evaluated and compared to the properties of the mathematical relation that represents the system organizing relationship. These three properties are symmetry, transitivity, and reflexivity. The is-connected-to relationship is symmetric, transitive and irreflexive. These relationship properties are used in the design of the value matrix and the applicable evolutionary algorithm to evaluate the system of interest [Simpson and Dagli, 2008].

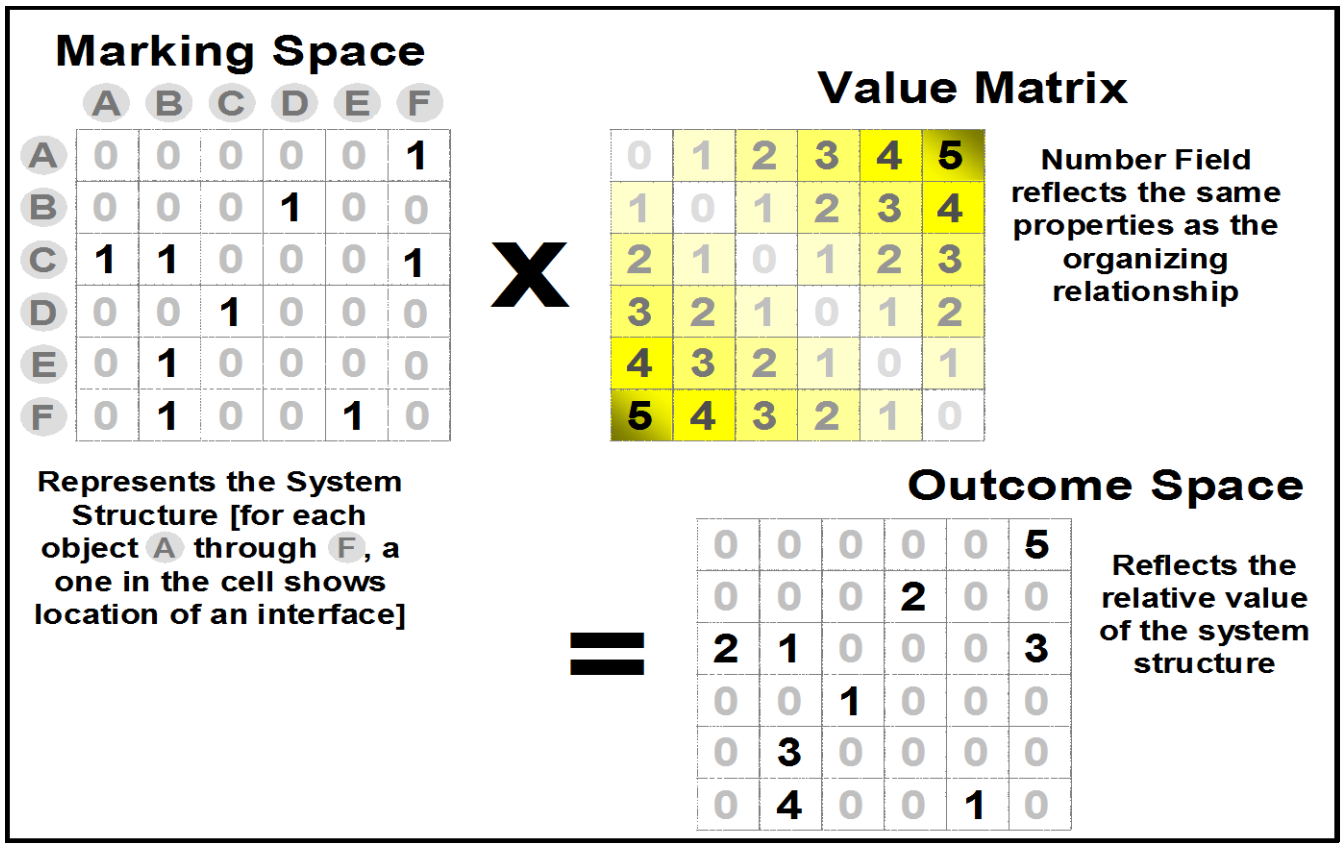

Figure 5. Notional representation of an ART marking space, value space, and its outcome space.

An ART value matrix for the AN2C is shown in Figure 6 and details the symmetric, transitive values that are organized around the value matrix diagonal. The initial and final ART marking spaces for that same AN2C example are shown in Figure 7 [Simpson and Simpson, April, 2009]. In this case, the provided value arrangement reflects the required symmetric, transitive and irreflexive properties. The initial marking space reflects the initial structural state of the system as it was first observed. The final marking space reflects the same system connections that are now grouped into a set of three clusters or subsystems. This ART structural graphic technique is valuable for discovering interconnected groups, aggregating objects into higher level system abstractions as well as visually communicating the structure of the discovered system clusters. Most real systems have hundreds of interconnections and interfaces. ART AN2C techniques provide a computer-assisted approach that reduces and/or eliminates the manual functional analysis, and consequently expands this type of analysis to systems with a large number of interconnections which would not be economically feasible and/or operationally practical to analyze using manual techniques. The cognitive complexity associated with the determination of the final system structural configuration is reduced by eliminating the need for human functional and structural analysis on the total system. The ART AN2C approach generates a small set of candidate final system structures that are reviewed, verified, prioritized and ranked by the human subject matter experts.

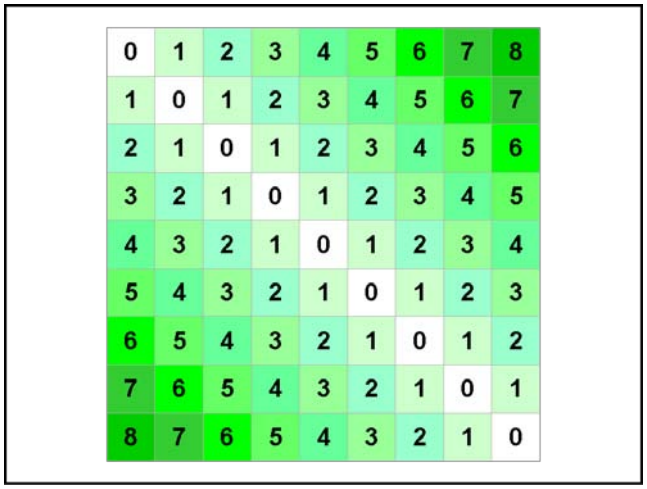

Figure 6. Abstract relation type value matrix for the automated N^2 Chart technique.

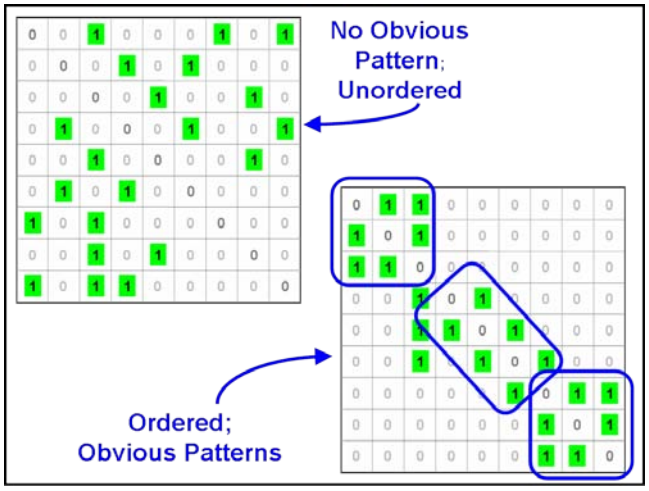

Figure 7. Initial and final marking spaces for an automated N^2 Chart technique.

The classical DSM approach represents the system structure using a marking matrix populated with a range of numbers and symbols that have specific meaning determined by the current system problem. In the DSM approach, there are two evaluation and solution concepts that impact the design of an ART DSM value space. The first DSM concept is the idea of the direction of information flow. Information feed-forward is considered more valuable than information feed-back. The second DSM concept is the inclusion of system components that have no information flow to other system components. This can result in a marking space that has no marks in the rows that represent the components with no information flow. Unlike the AN2C approach, which requires one or more marks in each row of the marking space, the DSM approach can include marking space rows that do not contain marks. An example of an initial marking space containing a row without marks is row one of Figure 8 [Simpson and Simpson, April, 2009].

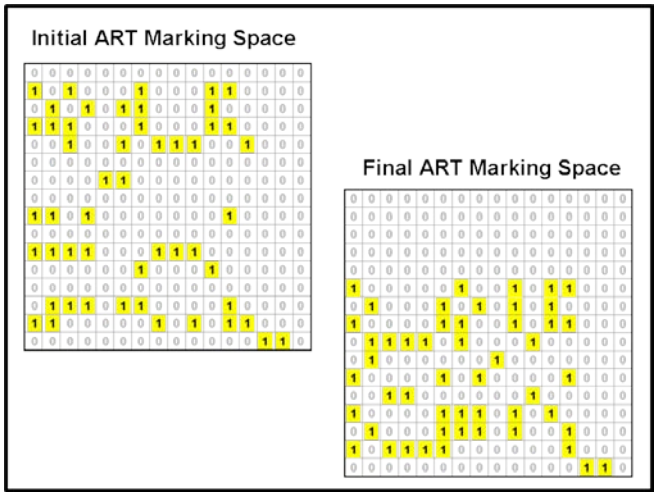

Figure 8. ART Initial and Final Marking Space of DSM Example.

When these two concepts are factored into the ART DSM value matrix design, the value matrix numerical values are arranged in a manner that places the largest values in the upper, right-hand corner, and the smallest values in the lower, right-hand corner (see Figure 9). The specific DSM information flow pattern and the specific DSM interconnection rules will determine the content of the ART DSM value matrix. The outcome from the DSM system analysis technique used to evaluate an electric car design was reproduced using the ART DSM approach [Simpson and Simpson, April, 2009].

0	59	60	61	62	63	64	65	66	67	68	69	70	71	72	73
29	0	58	59	60	61	62	63	64	65	66	67	68	69	70	71
28	27	0	57	58	59	60	61	62	63	64	65	66	67	68	69
27	26	25	0	56	57	58	59	60	61	62	63	64	65	66	67
26	25	24	23	0	55	56	57	58	59	60	61	62	63	64	65
25	24	23	22	21	0	54	55	56	57	58	59	60	61	62	63
24	23	22	21	20	19	0	53	54	55	56	57	58	59	60	61
23	22	21	20	19	18	17	0	52	53	54	55	56	57	58	59
22	21	20	19	18	17	16	15	0	51	52	53	54	55	56	57
21	20	19	18	17	16	15	14	13	0	50	51	52	53	54	55
20	19	18	17	16	15	14	13	12	11	0	49	50	51	52	53
19	18	17	16	15	14	13	12	11	10	9	0	48	49	50	51
18	17	16	15	14	13	12	11	10	9	8	7	0	47	48	49
17	16	15	14	13	12	11	10	9	8	7	6	5	0	46	47
16	15	14	13	12	11	10	9	8	7	6	5	4	3	0	45
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

Figure 9. ART DSM Value Matrix Example.

Pragmatic Complexity Reduction

The ART system representation form was developed to provide a standard pattern for the representation of the many classical and current systems science and systems engineering techniques that are available to assist the practicing systems professional in the identification, documentation, communication and modeling of complex systems. Almost all complex social-technical systems problems are associated with a large group of individuals from multiple technical, professional and social backgrounds. A key element in the evaluation of any complex system problem is the application of a language that communicates the system information at an abstraction level and application scope that is useful to the solution of the current system problem. Figure 10 provides an overview of the basic elements involved in the observation and identification of a system that is under development.

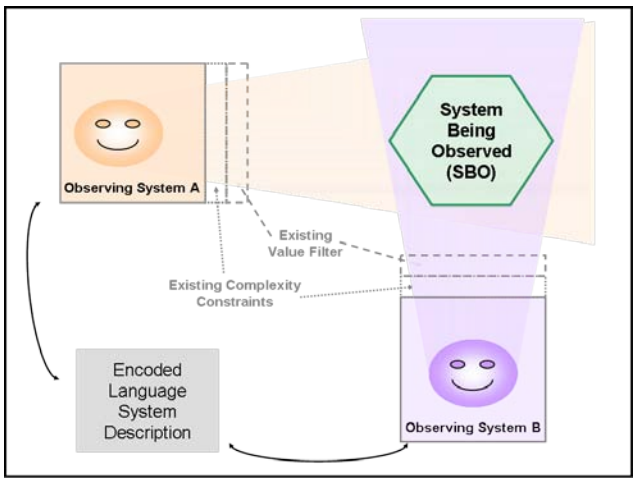

Figure 10. System Development Context.

As shown in Figure 10, pragmatic complexity reduction is achieved by careful organization and arrangement of a set of ART artifacts and patterns. A system pattern language enhances technical understanding and communication among individuals. A system pattern language that encodes system ART patterns represents both the external system behaviors and the internal system interfaces. In general, the ART system patterns act as a filter to help normalize the observers' values as well as to standardize the existing complexity constraints. Figure 11 shows a typical system situation where each observer has limited information about the scope, form and purpose of the system being observed. ART systems engineering evaluation and analysis techniques are then applied to reduce the cognitive complexity associated with the specific system analysis task.

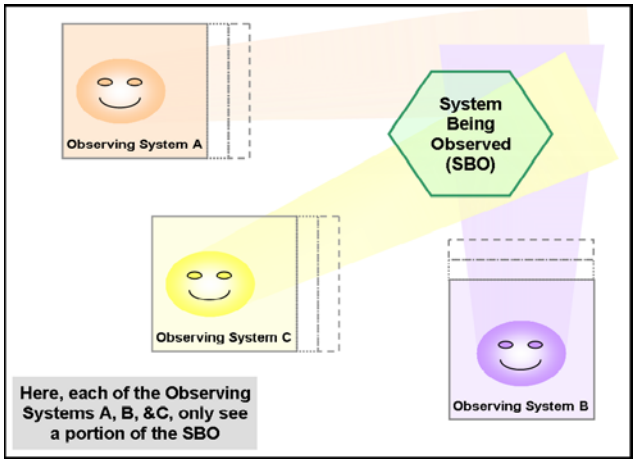

Figure 11. ART Pattern Applications.

The ART concept was designed to clearly document the combination of classical systems engineering techniques, patterns and executable algorithms that are applied in any given system context. The combination of evolutionary computation, standard systems analysis techniques, patterns and pattern languages are formed into an ART construct. The primary design goal of this ART construct is to reduce cognitive complexity and to increase the amount of clearly communicated system information. ART is designed to allow the same system to be represented by directed graphs that have different forms and semantics. This adaptable representation also applies to representing the system at different levels of abstraction and/or detail. One key to this type of adaptive system representation is the separation of the system structural representation from the system value determination and

representation.

As an enhancement of existing systems engineering techniques, the ART construct cleanly separates the structural and value aspects by creating and using a structural marking space as well a set of system value matrices. The two types of structural graphical representations, the marking space and the value matrix, provide the capability to identify a range of system structural configurations that conform to a given value set. Further, the global system organizing relationship that provides the logical basis for the construction of the ART marking space can be expressed at varying levels of aggregation and detail. The ability to vary the aggregation of structural graphics links is a primary mechanism used to reduce cognitive complexity [Simpson and Simpson, April 2009].

Key Fundamentals and the Application of the ART Framework

The three primary components of the ART framework can be completely filled in, contain no information, or have some partial degree of completeness. Applying the Law of Gradation (LOG), the information associated with a specific ART form can range from zero (0) percent complete to 100 percent complete. The ISM approach requires that 100 percent of the ART framework be completed. The N2C, AN2C and DSM techniques require that less than 100 percent of the ART framework information be completed. The LOG is used to determine when enough information has been developed to complete the current task at hand.

Once a set of ART form patterns are developed, it will be possible to select a candidate ART form and apply it to the problem at hand. However, these patterns must first be developed and perfected. The Law of Triadic Compatibility (LTC) and the Principle of Division by Threes (PDT) were used in the design of the ART form to restrict the number of ART components to three.

Summary and Future Work

The continuing value and effectiveness of systems engineering is tied directly to how well systems engineering techniques and practices reduce cognitive complexity associated with the development of large-scale distributed systems. A pragmatic approach has been introduced that reduces the cognitive complexity associated with large scale systems engineering tasks. This approach facilitates the reduction of cognitive complexity in at least three basic ways: 1) developing a pattern language to support the consistent application of ART patterns; 2) developing ART forms that encode typical systems engineering techniques in clearly defined processes and patterns; and 3) supporting the application of computing resources to evaluate and analyze alternative system representations. The clear separation of system structure from system value and system semantics supports the detailed communication of both the system organizing structural relationship and system interface value relationships. Further, the ART technique directly connects these relationships to mathematical relations that are the basis for the computation and analytical portions of the cognitive complexity reduction approach. As shown in Figure 12, ART contributes to a greater relative density of information transfer due to its use of a formal approach to prose, mathematics, and structural graphics.

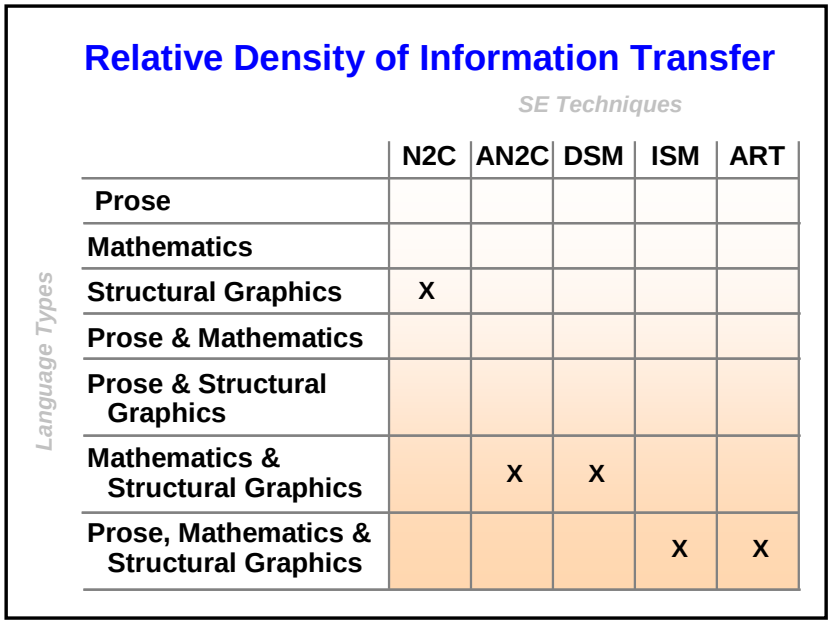

Figure 12. Relative density of information transfer including the ART forms.

Some of these techniques from the literature were motivated by an attempt to reduce the computational complexity associated with computer problem solving. There are early indications that the ART evolutionary computation technique can be used to directly reduce computational complexity associated with the solution of complex system problems. This is one area that will be explored in future work.

More research is required to develop additional detailed ART examples as well as improved methods to incorporate ART techniques into system patterns, system patterns languages and specific pragmatic complexity reduction activities associated with system science, engineering and analysis tasks.

References

Y. Bar-Yam, Dynamics of complex systems, Perseus, Reading, 1997.

K. Boulding, Kenneth, General systems theory-the skeleton of science, http://www.panarchy.org/boulding/systems.1956.html.

J.L. Casti, Alternate realities: Mathematical models of nature and man, Wiley, New York, 1989.

A.D. Hall, A methodology for systems engineering, Van Nostrand, Princeton, 1962.

A.D. Hall, Metasystems methodology, A new synthesis and unification, Pergamon, New York, 1989.

D. Harel, Algorithmics, the spirit of computing, Addison-Wesley, Wokingham, 1987.

D.K. Hitchins, Putting systems to work, Wiley, Chichester, 1992.

D.K. Hitchins, Advanced systems thinking, engineering and management, Artech House, London, 2003.

R.J. Lano, A technique for software and systems design, TRW and North-Holland, Amsterdam, 1979.

J. McCumber, Information systems security: A comprehensive model, Proceedings of the 14th National Computer Security Conference, National Institute of Standards and Technology (NIST), Baltimore, October, 1991.

D.M. Sharman and A.A. Yassine, Characterizing complex product architectures, Syst Eng 7 (2004), 35-60.

J.J. Simpson and C. Dagli, System of systems architecture generation and evaluation using evolutionary algorithms, Proceedings, 2008 IEEE 2nd International Systems Conference, Montreal, April, 2008.

J.J. Simpson and M.J. Simpson, Formal systems concepts, Proceedings, Fourth Annual Conference on Systems Engineering Research, Los Angeles, April, 2006.

J.J. Simpson and M.J. Simpson, Foundational systems engineering (SE) patterns for a SE pattern language, Sixteenth Annual International Symposium of INCOSE, "Systems Engineering: Shining List on the Tough Issues," Orlando, July, 2006.

J.J. Simpson and M.J. Simpson, A pragmatic complexity framework, Proceedings, INCOSE Spring 2009 Conference, "Systems Engineering - Affordable Success," Suffolk, April, 2009.

J.J. Simpson and M.J. Simpson, System of systems complexity identification and control, Proceedings, IEEE 4th System of Systems Engineering Conference, Albuquerque, June, 2009.

M. Sipser, Introduction to the theory of computation, PWS, Boston, 1997.

D.V. Steward, Systems analysis and management: Structure, strategy and design, PBI, A Petrocelli Book, New York, 1981.

J.N. Warfield, Structuring complex systems: Monograph no. 4, Battelle Memorial Institute, Columbus, 1974.

J.N. Warfield, A science of generic design, Iowa State University, Ames, 1990.

J.N. Warfield and A.R. Cardenas, A handbook of interactive management, AJAR, Palm Harbor, 1993, 2002.

J.N. Warfield, A science of generic design, managing complexity through system design, second edition. Iowa State University, Ames, 1994.

J.N. Warfield, *Understanding complexity: Thought and behavior*, Ajar, Palm Harbor, 2002.