SYSTEM EVALUATION AND DESCRIPTION USING ABSTRACT RELATION TYPES (ART)

Joseph J. Simpson
System Concepts
6400 32nd Ave. NW #9
Seattle, WA 98107
206-781-7089
jjs-sbw@eskimo.com

Dr. Cihan H. Dagli
UMR-Rolla
229 EMSE Dept.
Rolla, MO 65409-0370
573-341-4374
dagli@umr.edu

Dr. Ann Miller
UMR- Rolla
Dept. of E&CE
Rolla, MO 65409-0040
573-341-6339
annmiller@ieee.org

Abstract - Two abstract relation types (ART) are developed to represent, describe and establish a computational framework for a system. An abstract relation type is closely related to and builds upon two fundamental ideas. The first idea is the binary relation and structural modeling techniques developed by John N. Warfield. The second idea is the concept of abstract data types. These two ideas are combined to create an abstract relation type that provides a structured representation and computational method for systems and system components. The complete system description approach is based on six abstract relation types: context, concept, functions, requirements, architecture, and test (CCFRAT). When combined with digraphs and other graphical representations of the matrix form, ART provides a powerful tool for the communication of complex system interactions to large system design teams.

INTRODUCTION

System evaluation is a task that is becoming more difficult due to the increasing number of existing systems, interfaces and operations. At a basic level, a system can be defined in one of two general ways: a functional definition, or a "construction rule" definition. The functional definition of a system is "a constraint on variation." The construction rule definition of a system is "a relationship mapped over a set of objects." Abstract relation types assist in evaluation of systems that are defined or described in either of these two forms [5]. Binary relations, a primary component of the ART construct, have a long history of application in the systems science and engineering fields [4]. Binary relations can be represented as matrices, directed graphs or lattices [2]. The strong tie between the matrix form and graphical form of binary relations provide an excellent tool for system evaluation computation and communication. Structured group techniques that are based on binary relations and graph theory have been successfully used in the evaluation of large systems [10].

ART are an important tool used to group contextual system relations with a set of visual and computational techniques that are used to reason and communicate about systems. A well defined set of ART will provide a common system technical evaluation language, approach and computational set.

ABSTRACT RELATION TYPES

Two basic ideas are combined to create the ART. The first basic idea of structuring complex systems was developed by John N. Warfield [11]. This basic concept included the use of binary matrices, binary relationships and contextual relationships. A matrix **N** is represented by four sets; **N** = {***Is, It, Is x It, EIsxIt***}: where

Is is an ordered vertical index set
It is an ordered horizontal index set
Is x It is the set of all ordered pairs of ***Is*** and ***It***
EIsxIt is the entry set (or content) of the matrix.

A binary relation, as defined by Warfield, is represented by a binary matrix A that defines a two-block partition {R; ~R}on ***Is x It*****,** such that all ordered pairs in ***Is x It*** for which the entry is 1 are in the first block, and all other ordered pairs are in the second block. The first block is the binary relation R on ***Is x It*** and the second block is the

complementary relation ~R **on *Is x It***. A binary relation R on ***Is x It*** defines a binary matrix A indexed by *Is* and *It*. Some contextual relations developed by Warfield are: “is included in”, “is antecedent to”, “is subordinate to” and “is adjacent to.” These relationships are mapped in a binary matrix and are used in the analysis of the structure of complex interactions and systems [12].

The second basic idea is the abstract data type. An abstract data type is used to organize the structure and specify the operations of a specific data type. The abstract nature of the data type ensures that the data type is independent of the underlying information types [7] Abstract relation types are not independent of the underlying relations. However, they use a predefined set of operations for each specific ART. An ART is composed of two primary components: a binary relation matrix and operations on the binary relation matrix. The operations are grouped into three basic types: those operations associated with the semantics of the relation, the binary matrix construction and evaluation, and the organization and display of the matrix information. Abstract relation types (ART) extend the detail used in describing and defining the contextual relationship as developed by Warfield [12]. The basic set of system abstract relation types has six components. Those components are a relation type for context, concept, functional hierarchy, requirement, architecture hierarchy, and test. However, only two relation types, functional hierarchy (HART) and physical hierarchy (PART), are discussed in this paper. Each specific ART is identified by the relation type and the node type. For example, a HART would have functions as the nodes and “is included in” or “is composed of” as the binary relation. The methods associated with the HART support the operations of functional decomposition, abstraction level determination and other semantically legal relation operations. A PART would have physical nodes (system, segment, subsystem….) and “is part of” or “is composed of” binary relation. The methods associated with the PART would support the operations of physical decomposition, abstraction level determination, and other semantically legal relation operations.

FUNCTIONAL HIERARCHY RELATION TYPE

The functional hierarchy relation type is based on an “is included in” relation. Functional decomposition and functional flow block diagrams are classical system engineering methods that represent a system using this contextual relation. An example of a functional decomposition is shown in Table 1.

Num	Function Name
0.0	Execute Task
1.0	Prepare for Task
1.1	Conduct Pre-task Briefing
1.2	Prepare Personnel for Task
1.3	Prepare Equipment for Task
2.0	Transit to Task Execution Area
2.1	Prepare for Transit
2.2	Load Unit on Transport
2.3	Move to Task Area
3.0	Perform Task
3.1	Evaluate Task Area
3.2	Execute Task Objectives
3.3	Evaluate Task Effectiveness
4.0	Return from Task Area
4.1	Transit to Initial Area
4.2	Evaluate Unit Status
4.3	Record Lessons Learned

Num	Component Name
0.0	Wagon
1.0	Wagon Body
1.1	Metal Floor
1.2	Metal Sides
1.3	Metal Fasteners
2.0	Wooden Sides
2.1	Wood Uprights
2.2	Wood Planks
2.3	Wood Fasteners
3.0	Front Wheel Assembly
3.1	Front Axel
3.2	Front Wheels
3.3	Front Handle Assembly
4.0	Rear Wheel Assembly
4.1	Rear Axel
4.2	Rear Wheels
4.3	Rear Axel Attachment Assembly

Table 1 – Examples of Functional versus Physical Hierarchies

The 'Execute Task' function is represented as the main function at level 0.0, or highest level of functional abstraction. At the next level of functional abstraction, four functions are presented to represent the higher level function: 'Prepare for Task,' 'Transit to Task Execution Area,' 'Perform Task,' and 'Return from Task Area.' Then each function is further elaborated by functions at the next lower level of functional abstraction. At the core of the HART is a binary matrix that represents the structure of this functional relation. Similar to abstract data types, ART have operations that are performed on the existing relation structure. These computational operations provide a mechanism to calculate, compute and reason about functional hierarchical relations.

Given the matrix structure of the HART, key characteristics and attributes of the structure can be precisely described, computed and communicated. Some of the characteristics and attributes are: level of abstraction, functional completeness at any level of abstraction, functional timing, and functional sequence. Figure 1 shows the translation of the execute task function into a binary "is included in" relation matrix. The binary relation matrix is read down the columns as "is included in." As seen in the second column, a function is completely included in itself so the intersection of a function on the matrix diagonal contains a 1 that indicates the function is contained in itself. Column 2 also shows that the top level function, 'Execute Task,' is composed of all of the other functions listed in the matrix. For a binary relation matrix, the level of abstraction can be calculated from the number of cells in each row that contain the number 1. In the second row we see that there is only one cell with the number 1 in the row, so this is the highest level function. The third, fourth, fifth and sixth rows all contain two cells that contain the number 1; these are second level functions. The rest of the rows all contain three cells that contain the number 1 indicating that these are level three functions, the lowest level in this function set.

Once the binary matrix structure has been established for the HART, other operations can be performed on the matrix. For example, the level two functions shown in Figure 1 are sequential in nature. Therefore, sequential logic and constraints can be applied and evaluated by HART methods. The top level function may have specific time, energy and other constraints. Because the second level functions are sequential, the time constraint can be allocated across these level-two functions. The identification of a sequential function context relation allows the computation of total time for the top level function. The time for each lower level function may be added for a total time, or the total time required by the highest level function may be budgeted as a time constraint to each lower level function.

	0.0	**1.0**	**2.0**	**3.0**	**4.0**	**1.1**	**1.2**	**1.3**	**2.1**	**2.2**	**2.3**	**3.1**	**3.2**	**3.3**	**4.1**	**4.2**	**4.3**
0.0	**1**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1.0	**1**	**1**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2.0	**1**	0	**1**	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3.0	**1**	0	0	**1**	0	0	0	0	0	0	0	0	0	0	0	0	0
4.0	**1**	0	0	0	**1**	0	0	0	0	0	0	0	0	0	0	0	0
1.1	**1**	**1**	0	0	0	**1**	0	0	0	0	0	0	0	0	0	0	0
1.2	**1**	**1**	0	0	0	0	**1**	0	0	0	0	0	0	0	0	0	0
1.3	**1**	**1**	0	0	0	0	0	**1**	0	0	0	0	0	0	0	0	0
2.1	**1**	0	**1**	0	0	0	0	0	**1**	0	0	0	0	0	0	0	0
2.2	**1**	0	**1**	0	0	0	0	0	0	**1**	0	0	0	0	0	0	0
2.3	**1**	0	**1**	0	0	0	0	0	0	0	**1**	0	0	0	0	0	0
3.1	**1**	0	0	**1**	0	0	0	0	0	0	0	**1**	0	0	0	0	0
3.2	**1**	0	0	**1**	0	0	0	0	0	0	0	0	**1**	0	0	0	0
3.3	**1**	0	0	**1**	0	0	0	0	0	0	0	0	0	**1**	0	0	0
4.1	**1**	0	0	0	**1**	0	0	0	0	0	0	0	0	0	**1**	0	0
4.2	**1**	0	0	0	**1**	0	0	0	0	0	0	0	0	0	0	**1**	0
4.3	**1**	0	0	0	**1**	0	0	0	0	0	0	0	0	0	0	0	**1**

Figure 1 – Binary "Is Included In" Matrix

The HART provides the foundation for the development of "sequential function" and "concurrent function" operators and binary relation matrix calculations. Using these techniques, large complex functional structures can be developed, analyzed and communicated across large distributed system development teams.

PHYSICAL HIERARCHY RELATION TYPE

The physical hierarchy relation type is based on an "is part of" or "is composed of" binary relation. Physical system decomposition is an example of this type of systems analysis activity. An example of a physical decomposition for a wagon is shown in Table 1. The 'Wagon' component represents the highest level of physical composition. The hierarchical structure of the physical decomposition was selected to match the hierarchical structure of the functional decomposition to highlight the fact that even though the binary structure is the same, the operations and semantics of the PART have a different meaning than the operations and semantics associated with the HART. Physical properties and attributes are important aspects used in PART methods and computations. Weight is an example of a physical property that can be allocated, from the top-down, or computed, from the bottom-up in the PART. As with the HART, each level of physical decomposition can be clearly computed and communicated using PART methods. Using the "is a part of" and "is included in" binary relation, the wagon physical decomposition would look exactly like the binary matrix in Figure 1. The second column, of Figure 1, indicates that all of the physical components are included in the wagon. In a manner similar to the HART methods, each row would be evaluated to calculate the proper level of physical decomposition. Even though column 2 of Figure 1 indicates that all of the physical components are included in the wagon, the physical level of decomposition must be taken in to account when PART methods are calculating a complete system. The semantics associated with each relation and ART will cause the methods associated with the ART to change depending on the ART type.

An example of using the PART to represent and evaluate system weight is shown in Figure 2. In this case the numbers in the cells represent the allocated weight for each physical component. The total wagon weight, 26 units, is located in cell (2, 2). Summing the weight of all level two components; rows 3, 4, 5, and 6, gives the expected weight of the total system. Summing the weight of all of the level three components; rows 7 through 18 again gives the expected weight for the complete system.

	0.0	1.0	2.0	3.0	4.0	1.1	1.2	1.3	2.1	2.2	2.3	3.1	3.2	3.3	4.1	4.2	4.3
0.0	**26**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1.0	**10**	**10**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2.0	**6**	0	**6**	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3.0	**6**	0	0	**6**	0	0	0	0	0	0	0	0	0	0	0	0	0
4.0	**4**	0	0	0	**4**	0	0	0	0	0	0	0	0	0	0	0	0
1.1	**4**	**4**	0	0	0	**4**	0	0	0	0	0	0	0	0	0	0	0
1.2	**4**	**4**	0	0	0	0	**4**	0	0	0	0	0	0	0	0	0	0
1.3	**2**	**2**	0	0	0	0	0	**2**	0	0	0	0	0	0	0	0	0
2.1	**2**	0	**2**	0	0	0	0	0	**2**	0	0	0	0	0	0	0	0
2.2	**3**	0	**3**	0	0	0	0	0	0	**3**	0	0	0	0	0	0	0
2.3	**1**	0	**1**	0	0	0	0	0	0	0	**1**	0	0	0	0	0	0
3.1	**2**	0	0	**2**	0	0	0	0	0	0	0	**2**	0	0	0	0	0
3.2	**2**	0	0	**2**	0	0	0	0	0	0	0	0	**2**	0	0	0	0
3.3	**2**	0	0	**2**	0	0	0	0	0	0	0	0	0	**2**	0	0	0
4.1	**1**	0	0	0	**1**	0	0	0	0	0	0	0	0	0	**1**	0	0
4.2	**2**	0	0	0	**2**	0	0	0	0	0	0	0	0	0	0	**2**	0
4.3	**1**	0	0	0	**1**	0	0	0	0	0	0	0	0	0	0	0	**1**

Figure 2 – Physical Component Weight Allocation

ART UTILIZATION IN SYSTEM EVALUATION

Once the functional abstraction levels have been clearly defined using a HART, the functional interactions at each level may be evaluated using an "interfaces with" and "does not interface with" binary relation. This type of binary relationship is commonly called N Squared diagrams. Genetic algorithms are an excellent tool to assist in the search for optimal solutions in these types of structured problems [1]. Given a specific definition of the interface type and interface values, genetic algorithms may be used to find the interface arrangement that has the best or highest value configuration [3]. Two examples of the application of genetic algorithms to system evaluation are shown in Figure 3. The top two matrix representations, T1 and T2, in Figure 3 show the application of a genetic algorithm to the ordering of the level three functions from Figure 1. Figure 3, matrix T1, shows that the level three functions from Figure 1 are not strictly sequential like the level two functions in Figure 1. The "disordered" interfaces in the first matrix, T1, are ordered using a genetic algorithm and selection metric proposed by Hitchins and the resulting matrix is marked as T2 [3]. The second set of matrices, B1 and B2, were taken from a design structure matrix (DSM) clustering example [8]. The same type of genetic algorithm and selection matrix was used to evaluate the DSM example; however the top red triangular area was rated higher to indicate that the feed forward path in the lower triangular area was the preferred area for interface connection.

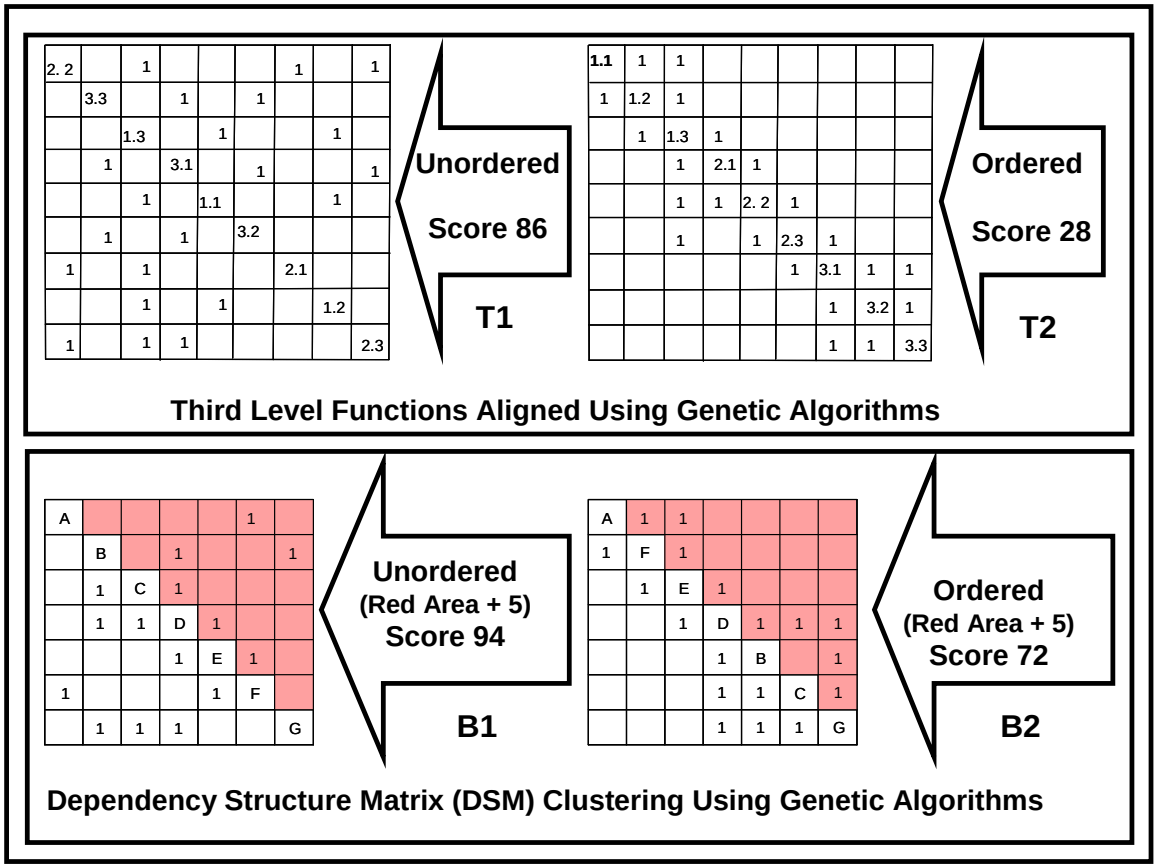

Figure 3 – Application of Genetic Algorithms to System Evaluation

The binary matrix representation of the "interfaces with" relation provides a number of benefits for genetic algorithm application. The binary matrix representation of the chromosome "building blocks" are completely specified with fixed order in

one dimension, while the other dimension provides the area of design combination. The value of the selection fitness function is provided as a function of another sequentially ordered matrix [6].

SUMMARY AND CONCULSIONS

This paper has introduced two of the six basic system ART. The twin foundations of natural language context relationships and the binary matrix form provide a solid basis upon which intelligent system computation can be based. These two ART cover most of the primary system design drivers. The functional analysis activity that creates the HART is usually one of the first steps in the system design process. The physical system design and structure that is represented by the PART is developed prior to system production. Creating the best value physical system that provides the required system functions is the primary goal of systems engineering. Incorporating an ART computational framework in the system production process will provide the foundation for the application of computational intelligence techniques as well as the clear communication of primary system design parameters. More research and development needs to be applied to the basic set of ART to create a robust system evaluation and design technique. There are many strong similarities between the genetic algorithm approach developed by Derek Hitchins [3] and the ordered messy genetic algorithm developed by Dimitri Knjazew [6]. More research is required to evaluate the value provided by these two approaches to the description, design and evaluation of systems using genetic algorithms.

References

[1] Coley, David A., "An Introduction to Genetic Algorithms for Scientists and Engineers", World Scientific, New Jersey, 2005

[2] Gratzer, George, "General Lattice Theory", Birkhauser Verlag, Boston, 2003

[3] Hitchins, Derek K., "Advanced Systems Thinking, Engineering, and Management", Artech House, London, England, 2003.

[4] Klir, George J., "Facets of Systems Science", Plenum Press, New York, 1991

[5] Klir, George J., "An Approach to General Systems Theory", Van Nostrand Reinhold Company, New York, 1969

[6] Knjazew, Dimitri, "OmeGA, A Competent Genetic Algorithm for Solving Permutation and Scheduling Problems", Kluwer Academic Publishers, Boston, MA, 2002

[7] Rosen, Kenneth H., "Handbook of Discrete and Combinatorial Mathematics", CRC Press, New York, 2000

[8] Sharman D. M. and Yassine A. A, Characterizing Complex Product Architectures, System Engineering Volume 7 Number 1, 2004

[9] Simpson, J.J, Dagli C., Grasman S., Miller A., "A Generic, Adaptive Systems Engineering Information Model", *Proceedings of the 15th Annual International Symposium of the International Council on Systems Engineering* (Rochester, NY, July, 2005)

[10] Warfield, John N. "An Introduction to Systems Science", World Scientific, New Jersey, 2006.

[11] Warfield, John N. "The Mathematics of Structure", AJAR Publishing, Palm Harbor, FL, 2003.

[12] Warfield, John N. "Structuring Complex Systems", Battle Monograph No. 4, Battelle Memorial Institute, Columbus , OH, 1974.