2015 International Conference on Computational Intelligence and Communication Networks

SoS Architecture Alternatives Tradespace Modeling and Computable Experimentation: A Framework with System Engineering Thinking

test2

可以

Zhifei LI
College of Information System and Management
National University of Defense Technology
Changsha, Hunan, P. R. China 410073
lee.nudt@gmail.com

Yifan Zhu
College of Information System and Management
National University of Defense Technology
Changsha, Hunan, P. R. China 410073
yfzhu@nudt.edu.cn

Feng Yang
College of Information System and Management
National University of Defense Technology
Changsha, Hunan, P. R. China 410073
yf.nudt@gmail.com

***Abstract*—The paper presented a data-centric, capability-focused and system engineering approach to facilitate the architecture modeling and computable experimentation of challenging SoS. Firstly, modeling the SoS architecture based on a set of architecture products and Activity-Based Methodology (ABM) using Object Process Methodology (OPM), which shows a system engineering thinking. Secondly, based on the OPM/DoDAF Meta-model (DM2), from the view of Top to Down, we planned to characterize useful and insightful relationships between the SoS architecture model and its Design Structure Matrix (DSM), then studied the SoS architecture alternative tradespace exploration problems with bi-layer mapping. Thirdly, based on the OPM/DM2, from the view of Down to Top, a method of mapping OPM/DM2 to Multi-Agent Simulation (MAS) was proposed. This will encourage a closer collaboration between the system architect and the analyst.**

Keywords—system-of-systems architecture; tradespace; system engineering; OPM; DSM; MAS

I. INTRODUCTION AND BACKGROUND

Recently, capability based analysis, design, and acquisition have had a significant impact in defense related programs. This capability-based mentality shares a natural link with architecting, in that capabilities are achieved through a series of activities. These activities can be represented as an operational architecture. Through the architecting process, they can be mapped to candidate solutions, which can then be evaluated and compared. These solutions provide the ways and means by which a capability is achieved. This kind of approach has been suggested to help address high level capability needs and help to avoid the stove piping that has often plagued defense acquisition [1].

The challenge presented by the sheer number of possible alternatives is compounded in SoS problems. First, it must be able to capture and define the large number of architectural alternatives available for consideration during the early phases of acquisition and systems engineering. Next, it must provide a way to filter through the tradespace and find only the promising alternatives for evaluation, while eliminating those that are either unrealistic or are not expected to meet mission goals. Finally, because even the filtering processes will still leave large numbers of alternatives to be evaluated, there must be a way to quickly and accurately evaluate the remaining alternatives. The traditional systems engineering approach have significant limitations in dealing with the large and complex SoS. Architecture is a meta-structure for integration of subsystem, and it focus on how to select the right combination of subsystems to deliver the performance and capability. Architecture-driven development is proposed to meet the needs of the SoS engineering [2]. Architecture-driven development is both a top-down design and development process, but also a bottom-up process of assessment and validation. The remainder of this paper is organized as follows. Related work on architecture modeling and analysis is re-viewed in Section 2. Section 3 presents our approach. An illustrative implementation is offered in Section 4. Appropriate conclusions and insights are finally provided in Section 5.

II. RELATED WORK

Architectures are a means to an end, not an end in them. Many models of established architecture frameworks can be utilized to describe both structural and behavioral aspects of a system, but these models are static since they only model the static structure of the architectural elements and their relationships. Thus, they fail to support detailed dynamic analysis of how the components interact with each other, and whether the architecture as modeled exhibits the desired behavior. An executable model is strongly needed to perform time-based dynamic simulation to allow a more complete examination and early exploration to address the performance and effectiveness of the capabilities as modeled in the architecture against the capability requirements. The main method of executable models of DoDAF is based on the work of Levis from Architecture Laboratory in the United States George Mason University [3][4]. Griendling and Mavris developed an executable environment as a key

978-1-5090-0076-0/15 $31.00 © 2015 IEEE
DOI 10.1109/CICN.2015.139

component of the ARCHITECT methodology, which uses a standard set of DoDAF views and the associated data to automatically generate executable models of four different types [5]. Despite this, there are still some limitations need to be pay more attention. *Data-centric.* Existing SoS architecture are product-centric to build the architecture model, the results are a variety of view product. The traditional executable modeling method is a conversion process from view product to executable model. DoDAF2.0 is a modeling framework of data-centric; the view product is extracted from the architecture data meta-model. If the executable model were conversed from view product, data integrity and consistency will suffer. Therefore, it is a very meaningful work to study how to implement the conversion from the DM2 to the executable model in the data-centric architecture framework. *Ideal Holistic Modeling Language.* Based on the needs of the SoS architecture alternatives, an ideal holistic modeling language should have the following characters [6]: It must be universal and support generic object-oriented concepts; It must be capable of modeling the structural, behavioral, and dynamic aspects of a system; It should support both graphical and textual syntax; It must be precise, mathematically rigorous, and executable; It must capture design space or constraints; It must support hierarchical abstractions; It should consist of a relative small set of modeling constructs and notations. It should be easy to understand and use. i.e., the modeling constructs and notations should be intuitive to architect; It should facilitate data exchange for sharing models and communicating with other computer programs and database; It should facilitate the communication between stakeholders and architects from different knowledge domains; It must be easy to implement using programming language. *Constructive combat simulations.* Relatively little work has been put into taking data from the architectures and exercising them in a constructive combat simulation environment. While the CPN is a valuable tool for understanding the dynamic behavior of a system, it fails to model a combat environment where the rules of engagement are changing and the enemy model is learning and evolving. To evaluate the military value of a future system, one must do so in a simulated combat environment in which the enemy has his own architecture. Andreas Tolk proposed the grand challenge of merging of C4ISR and M&S components into heterogeneous systems supporting the Warfighter [7]. These and other efforts left largely unaddressed is the possibility of building constructive combat simulations based on architectural data, a capability which would have significant impact on both communities.

III. PROPOSED APPROACH

As described in the previous section there are some limitations in the area of SoS architecture alternatives tradespace modeling and analysis. On the other hand, the stakeholders not only care for the SoS-level performance space such as cost, risk, P-success, but also pay more attention to the systems-level tradespace, such as the Sensor Angle Sensor Range, Sensor width, Radar Power of a UAV under specific SoS environment. The general framework of the SoS architecture alternatives tradespace modeling and computable experimentation is proposed in Figure1.

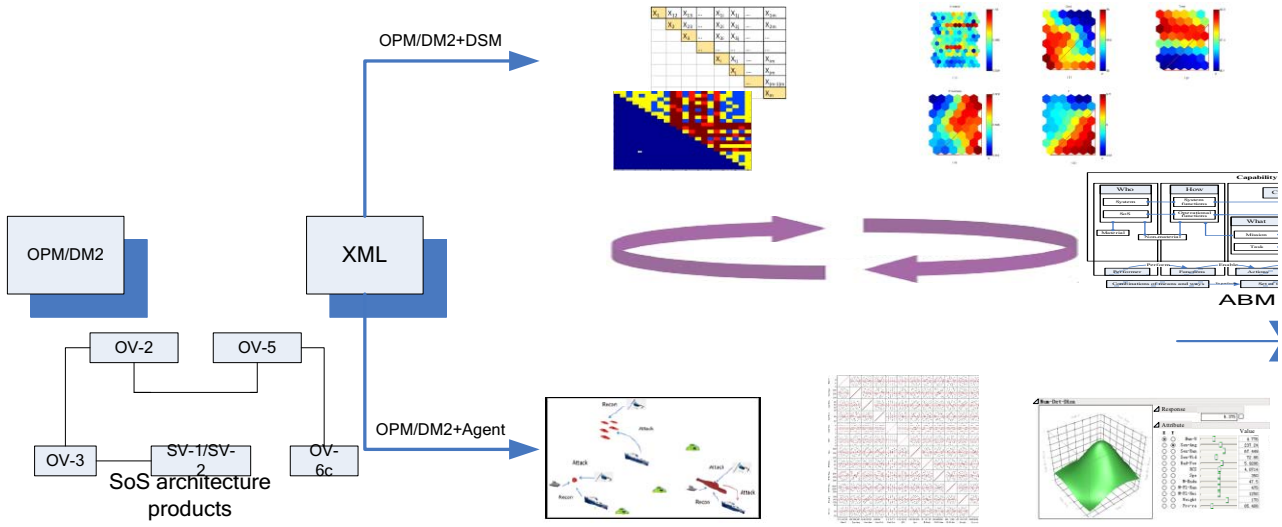

Figure 1 The general framework of the method

Object-Process Methodology (OPM) is an approach for designing information systems that combines a minimal set of building blocks (i.e., objects, states and processes that transform them) with a dual graphic-textual representation in a single model [8].Firstly, modeling the SoS architecture based on a DM2 and ABM using OPM, which shows a data-centric idea and system engineering thinking. Secondly, based on the OPM/DM2, from the view of Top to Down, characterizing useful and insightful relationships between the SoS model and its DSM, then study SoS architecture alternative tradespace exploration problems with bi-layer mapping. Thirdly, based on the OPM/DM2, from the view of Down to Top, mapping OPM/DM2 to MAS software, this will encourage a closer collaboration between the

system architect and the analyst. The use of executable architecting and constructive combat simulation as a key component of an early phase acquisition process can help to alleviate several of the challenges associated with architecture evaluation.

IV. IMPLEMENTATION

A. Top-Down

Traditional SoS optimization is a process that flows from the design space to the performance space, called "Forward Mapping". However, successful experiences and experimental data are difficult to use in the design and development process. Additionally, acquisition staffs tend to pay more attention to the overall SoS performance, hoping to map the route from the performance space (the actual SoS performance and performance evaluation results) to the design space, in order to complete the design space exploration, which can help to accurately locate the design space area of concern. Limiting the SoS design optimization to a smaller space saves time spent searching in an unnecessary area, making the whole design optimization more targeted. Mapping from the "Performance Space" to the "Design Space", referred to here as "Reverse mapping", complies with the general rules of acquisition, as shown in Figure2.

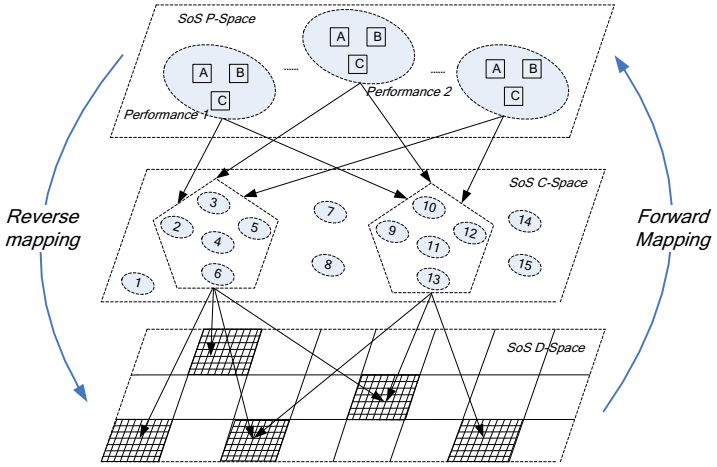

Figure 2 The general idea of the method

- From OPM to DSM.

We investigate potential benefits of employing the Design Structure Matrix in the context of Systems Engineering for the purposes of analyzing and improving the design of a SoS. OPM notation supports conceptual modeling of SoS using a single type of diagram to describe the functional, structural and behavioural aspects of a system. An OPM model consists of a set of hierarchically-organized Object-Process Diagrams that alleviate SoS' complexity. Each OPD is obtained by refining object or process in its ancestor OPD. In this, Amira Sharon made the same research in the contex of Model-Based Systems Engineering (MBSE) for the purposes of analyzing and improving the design of a product-project ensemble [9]. Using Object-Process Methodology (OPM) as the underlying modeling language, we planned to characterize useful and insightful relationships between the SoS model and its DSM. Get the semantics of and analogy between various types of relationships as they are reflected in both the OPM system model and the DSM derived from it.

- Bi-layer exploration process

Layer 1: RST-based mapping from P-space to C-space. As shown in Figure 3, this paper studied the SoS design space exploration methods of the architecture alternatives, primarily learning from previous design experience to better guide the overall design optimization with use of RST reasoning, based on the analysis of similar cases. Similar, relevant cases are first selected, according to the capability gap and required operational activities, in order to determine the initial system configuration, which provides foundational data for subsequent derivation of configuration rules. Secondly, it must be determined whether or not the parameter attributes are complete. Thirdly, if the attribute data of the configuration program is complete, then the configuration rules from the complete configuration decision table are derived, using RST. If incomplete data is included, then reasoning with corresponding use of RST in the incomplete configuration decision table is utilized. In the process of complete rule reasoning, the selected attributes are first analyzed and the continuous data is discretized, using the FCM (fuzzy C-Means) algorithm, which pre-processes data for the use of RST. Secondly, in accordance with the selected configuration, similar cases are collected from the corresponding performance estimates, along with a variety of configuration attribute data, constituting a configuration decision table. Again, the simplest related configuration rules from the configuration decision table are acquired with RST. Finally, when the performance space and the configuration space are positioned corresponding to configuration rules, the mapping from P-space to C-space can be completed.

Layer 2: SOM-based mapping from C-space to D-space. Upon completion of the preliminary configuration of SoS, relevant experimental data or the actual running information can primarily be selected from similar cases, according to the

given systems within the configuration. Secondly, the relevant Surrogate Models can be established, using relevant information and data, and then preliminary optimization can be made based on the model. Again, the design variables and the objective functions were analyzed using the SOM. A detailed study of the relationship between design variables and the objective function can then be made, using the colour changes of a two-dimensional hexagonal grid, eliminating the unimportant design variables and reducing the associated interval of design variables. Finally, the dimensions and the design variables of concern can be determined for the design space and then a new design space can be constructed with a smaller design optimization range than the original, and including local and global optimums. The smaller range of a more targeted, and relatively transparent design space optimization can improve efficiency, saving design time and cost.

B. Bottom-Up

- From OPM to Multi-Agent

Many studies have confirmed that MAS, partially due to its ability to characterize "hierarchies of interacting agents that can adapt their behavior," can successfully model the impact of Information Superiority on the battlefield. Because agent based models like MAS model systems at a high level of detail, programming them "by hand" requires extensive research and exhaustive testing. It is logical to look to DM2 as a source for the needed simulation data. The architecture descriptions were brought into the practice to help acquire C4ISR intensive systems, agent based simulations were used to help understanding the effects of C4ISR systems. It is natural that an interface between these two separate but related resources be investigated.

There are three necessary steps to accomplish the OPM mapping to MAS. In this, Zinn proposed a primarily manual process of mapping the information from a DoDAF compliant architectural description to an agent-based simulation, like the SEAS[10].

Step1. Development of OPM/DM2 models for SoS architecture.

We should development the OPM/DM2 model for the SoS architecture based on the information needed by ABS. The information required by MAS was categorized as General Attributes, Communications Information, and Orders.

Step2. Extracting data from OPM/DM2 models and converting them into MAS.

Suppose that the OPM/DM2 data models for SoS have been created and OPM/DM2 are stored in XML format, search in the database. For example, the SV-7 data models are stored in XML format; we can search the required data such as speed and altitude of mobile unit from it. The General Attributes, Communications Information, and Orders information can be got in the same way.

Step3. Analyze architecture based on ABS.

Based on the result of step1and step2, we can use MAS to analysis the architectures and evaluate the SoS alternatives.

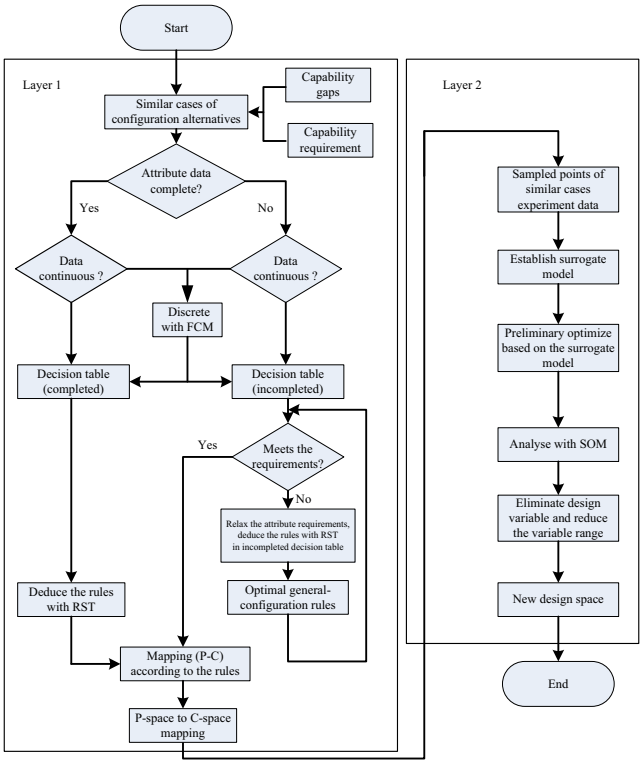

Figure 3 The Bi-layer exploration process

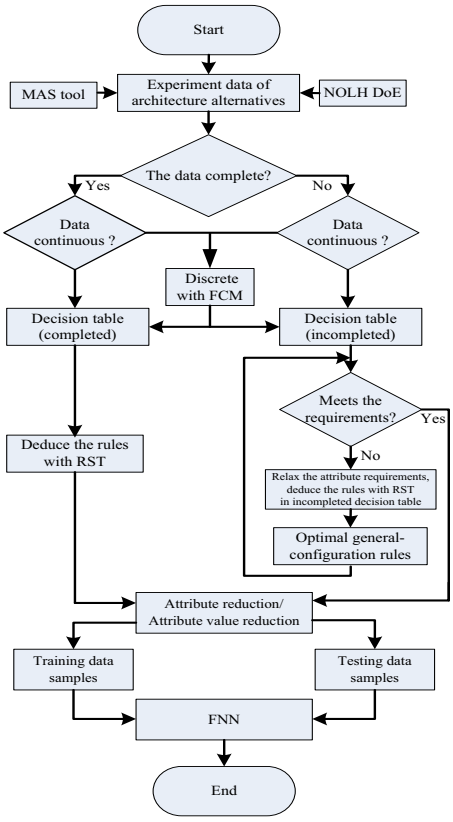

Figure 4 The RSBFNN overall flow chart

- DoE and Surrogate Model

The architecture representation is used to structure the modeling and simulation environment, which consists of agent-based models created in a MAS tool. DoE is then wrapped around the MAS model, in order to obtain a representative sampling of the multi-dimensional design space. This DoE and MAS tool provides the necessary data to create a series of decision rules for each of the key responses, with respect to the varied inputs in the actual model.

Firstly, the simulation output data stored in the database is pre-processed, and the missing data is filled in or the invalid data is removed to obtain the original data decision table. Next, the original data is discretized and normalized to obtain the minimum set of rules covering the original decision sample characteristics with the greatest degree of completeness, using rough set attribute reduction. Finally, the initial fuzzy neural network topology is determined and the network structure with the test data is trained and adjusted to get a comprehensive evaluation model with the optimal structure. The overall flow chart of the RSBFNN to system-level architecture evaluation is shown in Figure 4.

V. DISCUSSION

In this paper, a general framework of SoS architecture alternatives tradespace modeling and computable experimentation method was proposed. This framework is facilitated by a holistic modeling approach that combines the capabilities of OPM, DM2, DSM and MAS. The holistic model can not only capture the structural, behavioural, and dynamic aspects of a system, allowing simulation and strong analysis methods to be applied, it can also specify the architectural tradespace. Firstly, OPM/DM2 was developed from the view of capability. Secondly, based on the OPM/DM2, from the view of Top to Down, we planned to characterize useful and insightful relationships between the SoS model and its DSM, then studied SoS architecture alternative tradespace exploration problems with bi-layer mapping. Thirdly, based on the OPM/DM2, from the view of Down to Top, a method of mapping OPM/DM2 to MAS software was proposed. Future work could focus on how to modeling the SoS architecture alternatives tradespace based on ABM and DM2 with an smooth method from requirement to solutions, and the detailed mapping rules from OPM/DM2 to DSM and an automated XML to MAS interface. Additionally, more complex experimental design and analysis are required to enable the optimal design of the architecture.

ACKNOWLEDGMENT

This research was supported by the National Natural Science Foundation of China, and a project supported by scientific research fund of Hunan provincial education department. We are grateful to the anonymous reviewers for their valuable comments and suggestions to improve our work.

REFERENCES

[1] Chairman of the Joint Chiefs of Staff, Chairman of the Joint Chiefs of State Instruction CJCSI 3170.01 G, March 2009 .

[2] G. Raghav and S. Gopalswamy, “Architecture Driven Development for Cyber-Physical Systems,” SAE International Journal , 2010,3 (1): 95-100.

[3] Levis A H, Wagenhals L W. C4ISR Architectures: I. Developing a Process for C4ISR Architecture Design. Systems Engineering, 2000, 3 (4): 225-247.

[4] Wagenhals L W, Levis A H. Service Oriented Architectures, the DoD Architecture Framework 1.5, and Executable Architectures. Systems Engineering, 2009,12 (4): 312-343.

[5] K. Griendling and D. Mavris, An architecture-based approach to identifying SoS alternatives, 2010 5th IEEE Int Conf SoS Eng, Loughborough, Leicestershire, UK, 2010:1-6.

[6] Renzhong Wang. Search-based system architecture dvelopment using a holistic modeling approach. 2012.

[7] Tolk Andreas, Hieb, Michael. Building & Integrating M&S Components into C4ISR Systems for Suporting Future Military Operations, Position paper for the 2003 International Conference on Grand Challenges for Modeling and Simulation, August, 2002.

[8] Dov Dori. Object Process Methodology: A Holistic Systems Paradigm; with CD-ROM. Springer, 2002.

[9] Amira Sharon, Olivier L. de Weck, and Dov Dori. Improving Project-Product Lifecycle Management with Model-Based Design Structure Matrix: A Joint Project Management and Systems Engineering Approach. Systems Engineering, 2013,1 (1): 413-426.

[10] A.W. Zinn, The use of integrated architectures to support agent based simulation: An initial investigation, M.S. thesis, Systems Engineering, Air Force Institute of Technology, Wright-Patterson AirForce Base, OH, 2004.