An Agent-Centered Multi-Formalism Modeling Framework for SoS Behavior

Xiaobo Li, Weiping Wang, Yonglin Lei, Qun Li
Institute of Simulation Engineering, College of Information Systems and Management,
National University of Defense Technology
Changsha, PRC
{lixiaobo, wangwp, yllei, liqun}@nudt.edu.cn

***Abstract*—The system of systems (SoS) possesses essential characteristics of autonomy, evolvement and emergency, which can only be studied from the aspect of SoS behavior. Modeling and Simulation (M&S) can cope with SoS complexity and study its behavior more efficiently at lower costs. Considering the multi-subsystem and multi-domain characteristic of SoS, this paper proposes an agent-centered multi-formalism modeling framework for SoS behavior which combines different modeling formalisms to describe different aspects of SoS behavior. The framework comprises two parts: cognitive behavior which is modeled using agent based modeling, and physical behavior which is modeled using classic modeling formalisms. Combing cognitive and physical behavior modeling, this framework supports simulation modeling of SoS behavior, and thus enables early verification & validation, automatic generation of SoS behavior simulation models, and human (SoS entity)-in-the-loop simulation of SoS behavior at variable resolution levels.**

Keywords-System of Systems Engineering, Multi-formalism Modeling, Agent based Modeling, SoS Modeling and Simulation

I. INTRODUCTION

SoS is a collaborative meta-level system structure where independent complex systems are integrated to provide increased functionality and performance capabilities [1]. The fundamental elements of SoS are complex systems, while that of the system are components. Thus, compared to the system, SoS possesses essential characteristics of autonomy, evolvement and emergency, which can only be studied from the aspect of SoS behavior.

The premise of SoS behavior research is the differentiation of SoS behavior and system behavior. A system can fulfill certain function(s) independently and its components are identified according to its structural coupling relationship. While SoS integrates the subsystems according to its capability requirements, and its components are systems, which are usually heterogeneous and can perform certain functions. So SoS behavior not only includes internal behaviors of its subsystems, and more importantly, the behaviors between its subsystems. Internal system behavior belongs to systems engineering field, while inter-system behavior research belongs to SoS engineering field. SoS behavior can be conducted in physical domain, cognitive domain, information domain and society domain [2]. In this paper we call the behavior in physical and information domain "physical behavior", and behavior in cognitive domain and society domain "cognitive behavior".

According to the working progress, SoS research can be divided into three phases. T**he first phase is the formalism-based analysis and validation of SoS architecture solutions** [3], which conducts high level logic reasoning and analysis of certain aspects of SoS behaviors. This method is formal and concise; but strictly speaking, it is not SoS behavior experimentation. **The second is SoS M&S,** which builds simulation models of SoS, performs simulation experiments, and then study SoS behavior based on the data analysis of simulation results. This method can simulate the SoS behavior along the time base and unfold the interactions and behavior details, especially it can embed the simulation system into the whole SoS to conduct human-in-the-loop and SoS-entity-in-the-loop simulation; however, the development of SoS simulation system is knotty and the models are difficult to verify and validate. **The third is to conduct SoS experimentation**, which can obtain authentic data, but it is expensive and only available after the construction of SoS is finished.

Current SoS research concentrates on the first phase, but the characteristics such as autonomy, evolvement and emergency can not be researched in the first phase. This paper focuses on the second phase to provide support for the testing and evaluation of SoS architecture solutions. SoS contains multiple subsystems of different domains. Thus to build a SoS behavior simulation system from scratch is taxing, difficult, and not available when SoS is under construction. So the feasible way is to build SoS simulation systems based on SoS architecture solutions, and simulate the SoS behaviors selectively according to the requirements. For the complexity of SoS, current SoS M&S are conducted at high level with low-resolution simulation models. One typical example is System Effectiveness Analysis Simulation (SEAS) [4], which uses surrogate simulation models to simplify physical process with a probability. In this paper, we propose an agent-centered multi-formalism modeling framework for SoS behavior which combines different modeling formalisms to describe multi aspects of SoS behavior suitably. The formalisms describe the SoS behavior in detail on a formal basis, thus enables the early verification & validation of SoS behavior models, and support automatic generation of SoS behavior simulation models using model transformation techniques. Moreover, the framework can models SoS subsystems at different resolution levels and

Published by Atlantis Press, Paris, France.

support human (SoS entity)-in-the-loop simulation of SoS behavior at variable resolutions according to the requirements.

The remaining part of the paper are organized as follows. Section 2 proposes the modeling framework and section 3 discusses the technical implementation of the modeling framework. Section 4 presents the related work and section 5 concludes the paper.

II. A COMPOSABLE MODELING FRAMEWORK FOR SOS BEHAVIOR

To employ SoS behavior M&S for architecture testing and evaluation, we propose a working process framework as shown in Figure 1. Architecture design is based on three kinds of diagrams: requirement, structure and behavior diagrams, which specify the main structure and behavior patterns of the SoS according to the requirements. Behavior modeling embodies the behavioral specifications in architecture solutions and add necessary details based on agent and classic formalisms. All behavior models are implemented as SMP2 models, which can be simulated with SMP2 simulation engine to conduct SoS behavior experimentation.

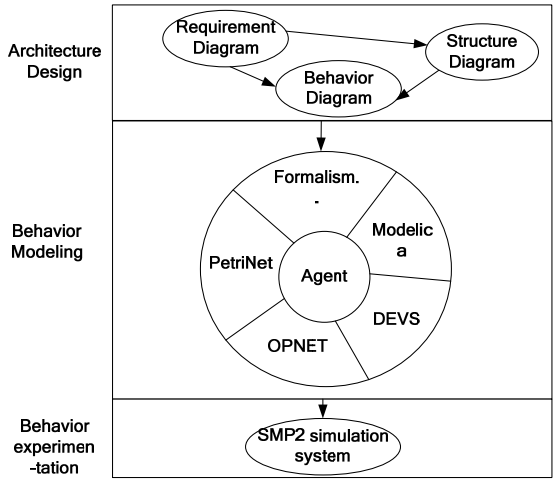

Figure 1. SoS Behavior M&S process for architecture testing and evaluation

The complexity of SoS structure determines the complexity of its behavior. When modeling complex SoS behavior, different modeling paradigms and formalisms should be used for multiple aspects or characteristics of SoS behavior. SoS behavior can be roughly divided into cognitive behavior and physical behavior. In this section we propose a composable modeling framework which includes agent-based cognitive behavior modeling, classic formalism based physical behavior modeling, and the composition of these two kinds of behavior models.

A. Agent-based Cognitive Behavior Modeling

Agent based modeling (ABM) is a new modeling paradigm and an advancement of object orientation, which reflects the modeling desire of intelligence and sociality. Although agents may have different meaning in different research areas, it is a consensus that agents possess some common characteristics, such as reactive, active, autonomous, target-guided, sensitive and so on. Cognitive behavior is characterized as intelligent, adaptive and evolving behaviors that based on certain rules or knowledge, and ABM is recognized as the prevalent and suitable way for cognitive modeling in many areas since ABM originate in cognitive and social domain.

Moreover, traditional modeling methods are not competent to depict characteristics of SoS behavior, while ABM exhibits several advantages. 1) SoS is the integration of independent subsystems to fulfill a common purpose, and one intrinsic characteristic of agent is autonomy; so ABM is suitable to model the independence and autonomy of SoS subsystems. 2) The adaptiveness and target-guided learning of agent enable ABM to model the evolvement of SoS and its constitutive elements. 3) Agent is designed to model the emergency behavior produced by large scale interactions among SoS elements.

SoS behavior modeling brings many new challenges to ABM, the significant ones of which are: 1) SoS includes numerous constituents which have complex internal behaviors, while former ABM research focus on numerous agents with simple internal behaviors; so the modeling method and simulation efficiency should improve accordingly. 2) The heterogeneous characteristic of SoS subsystems means its elements vary in intelligence and autonomy, so SoS simulation systems contains agent-based models of varied intelligence level and non-agent models; how to couple these heterogeneous models into a unified simulation system is a tough issue. 3) SoS behavior research sometimes examines the relationship between cognitive behavior and physical process inside a subsystem, so how to compose cognitive behavior part and physical behavior part into a whole subsystem simulation model needs to be studied. 4) The key point of SoS behavior modeling is to describe the coordination behavior of SoS based on subsystem agent models, and how to model the coordination behavior of heavyweight agents (agents that have complex external and internal behavior) needs to be researched.

B. Classic Formalsim based Physical Behavior Modeling

SoS physical behavior is the end effector of SoS, which has multiple characteristics and needs to be modeled according to different behavioral aspects. M&S community has developed and used a number of modeling formalisms for behavior modeling in various application domains, such as Discrete Event System Specification (DEVS), Petri Net, Statecharts, Modelica and so on. We call these formalisms classic formalisms. Classic formalisms usually describe one aspect or several aspects of system (SoS) behavior, e.g., Petri Net describes the concurrency characteristic. These formalism can be used in combination to model the SoS behavior more integrally. Take the combat SoS for example, Statecharts can be used to describe main internal states and the transition,

Modelica can be used for kinematics, OPNET can be used for communication, and DEVS can be used to model the event interface and response.

Classic formalism based physical behavior modeling has several advantages. Firstly, classic formalisms support formal analysis, early verification and validation. Secondly, A lot of modeling and simulation platforms have been developed for classic formalisms, thus ease the technical implementation and simulation of the formalism-based models. Thirdly, formalism-based models are at a high abstraction level, thus support interoperability, reuse and composability. Fourthly, there have been lots of research fruits of classic formalism based physical behavior modeling, which provides a sound base for SoS behavior modeling research.

C. An Agent-centered Multi-Formalism Modeling Framework

Although subsystem behavior of SoS looks the same as traditional system behavior, SoS global behavior is essentially different from traditional system behavior. The behavior between subsystems is function based interactions, which are autonomous, intelligent and heterogeneous. These characteristics require that SoS behavior modeling should take cognitive behavior modeling as the center and adopt a agent-centered multi-formalism composition modeling approach. This approach models the cognitive behavior of subsystems and other components using ABM, and integrate other formalism-based physical behavior models based on the cognitive agent models.

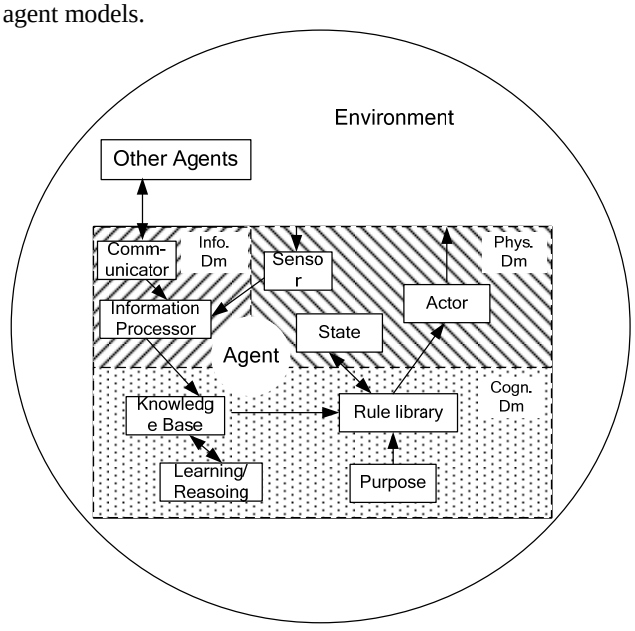

Figure 2. An agent-centered SoS behavior modeling approach

The essence of this approach is to use ABM to model the whole SoS and use classic formalisms to model the physical behavior of the agent in detail when necessary. Thus this approach can describe the SoS behavior from multiple aspects and support multi-resolution simulation of SoS behavior according to the research purpose. Inspired by SEAS, Figure 2 illustrates the proposed framework in detail. 1) In physical domain, the states and transitions of Agent are modeled by Statecharts, actions are modeled by DEVS or Petri Net, and the physical behavior of the sensor can be modeled by Simulink. 2) In information domain, OPNET can be used to model the communication behavior. 3) In cognitive domain, predicate logic can be used to model the rule library, and neural networks, genetic algorithms can be used to model the learning/reasoning mechanism. 4) formalisms can also be used to model the environment, e.g., cellular automata can be used to model the geography environment. The behavior in cognitive domain is the core of agent behavior, which control and manage behavior in all three domains. The behavior in physical and information domain is more stable and has been intensively researched in system engineering field; while cognitive behavior modeling is more flexible and hard to describe.

The key difference between the framework and SEAS approach is that physical behavior models of SEAS is a simplification of physical process based on statistical probability; while the proposed framework explicitly model the physical process and behavior in physical domain and information domain, and support simulation of SoS constituents behavior at different resolution level according to research purpose.

SoS structure is hierarchical so SoS agents are also organized hierarchically. Low level agents (atom agent) and other non-agent models compose a high level agent (we call it a composite agent). So the relationship among the agents are complicated since this hierarchical structure: the competition (rival) or cooperation relationship between atom agents, between an atom agent and an composite agent, or between composite agents. SoS structure is determined by its architecture and evolve along the time, so agents can choose to join or quit a composite agent or the SoS when situation changes.

III. TECHNICAL IMPLEMENTATION OF THE MODELING FRAMEWORK

A. Analysis of Feasible Solutions

The key for the implementation of the modeling framework is to establish the interaction or transformation relationship of the modeling formalisms (including agent). There are two feasible solutions. The first solution is metamodel mapping based model transformation, which includes the following steps: metamodeling the formalisms based on a unified meta-metamodel (e.g., Entity Relationship, Meta Object Facility), establishing the mapping relationship between metamodels of formalisms, and transformation of the models to the destination formalism according to the mapping rules. The second solution is adapter-based co-simulation. This solution constructs an adapter formalism to build a metamodel which describes the interaction relationship between two formalisms, instantiate an adapter model for the concrete

formalisms, and models of different formalisms can interact and co-simulate via the adapter.

The two solutions have their own advantages and disadvantages. The former is based on a common meta-formalism, thus it is formal and more applicable, and supports integrated modeling and unified simulation; however, the workload is heavy and it requires profound understanding of the formalisms. The second solution is easier to implement, but it is inefficient in run-time and taxing since the adapter needs to be built from scratch again when the models change. We choose solution 1 to implement all formalisms based on Simulation Model Portability 2 specification (SMP2) [5], which enables integrated modeling and unified simulation of SoS behavior.

B. SMP2-based Formalism Implementation: Petri Net as the example

SMP2 is a simulation model specification proposed by Europe Space Agency to support interoperability and reuse of simulation models, which has been widely used in complex simulation system development. We have implemented several modeling formalisms based on SMP2. We have discussed SMP2 based agent M&S in [6] and DEVS/SMP2 transformation in [7]; and Zhu proposed a SMP2-based Statecharts modeling framework in [8]. In this paper, Petri Net is taken as an example to illuminate how to implement modeling formalisms based on SMP2.

Petri Net is a concise yet prevalent formalism to model the concurrent, asynchronous system behavior in diverse application domains. We build a metamodel of basic Petri Net in Figure 3 using MetaGME language of GME [9], and from the metamodel a Petri Net modeling environment is generated using GME (in Figure 4). We establish the metamodel mapping relationship of Petri Net and SMP2 in Table 1, which is used to write a code generator to generate executable SMP2 simulation models (in C++) from Petri Net models. The code generator is convenient to implement since GME provides a Builder Object Network (BON)-based infrastructure which offers fundamental methods for model traversal and interfaces of the GME platform. We use the BON interpreter wizard [9] to automatically generate the framework code and implement the SMP2 code generation rules based on the abovementioned metamodel mapping relationship (in Table 1) in function InvokeEx() in C++ using Visual Studio. Figure 5 presents an example of code generation from Petri Net model.

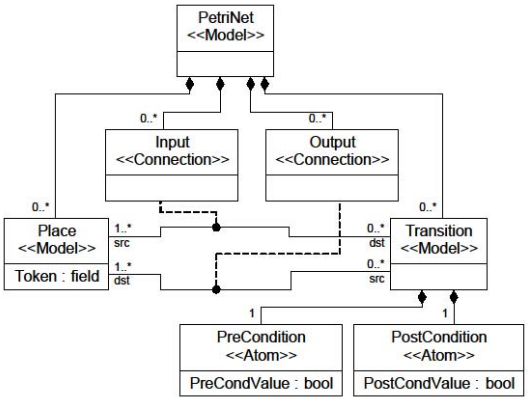

Figure 3. A metamodel of basic Petri Net based on MetaGME

Similarly, other formalisms-based models (including agent-based models) can also be implemented based on SMP2, thus the agent-centered multi-formalism behavior models are all implemented as SMP2-compliant C++ code and simulated seamlessly in SMP2 simulation environment. SMP2 is a low level model specification at implementation level, so when transforming other high level formalisms onto SMP2, no semantics will be lost during the transformation. However, for its low level representation, the SMP2 models are difficult to understand and maintain manually, thus the transformation process should be precise and automated.

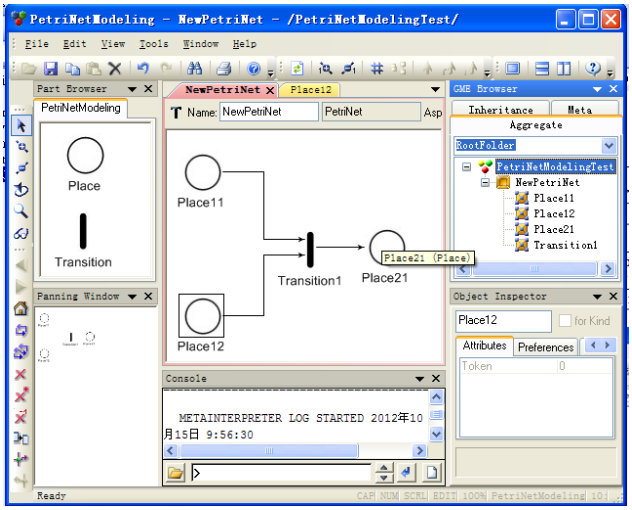

Figure 4. A Petri Net modeling environment based on GME

TABLE I. METAMODEL ELEMENTS MAPPING BETWEEN PETRI NET AND SMP2

Petri Net Elements	SMP2 Elements
PetriNet	Model (partially)
Place	Class
Transition	Class

Petri Net Elements	SMP2 Elements
Input	EventSource
Output	EventSink
Token	Field
PreCondition	Operation
PostCondition	Operation

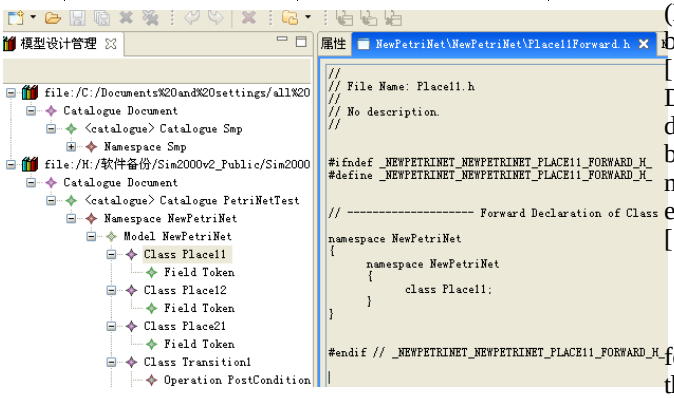

Figure 5. An example of code generation from Petri Net model

After the implementation of agent and other formalism based model based on SMP2, the agent-based multi-formalism modeling framework has been implemented as a executable system based on SMP2-compliant C++ code. The structure and interface of the subsystems in the SoS is implemented as SMP2- compliant agent models. And the internal behaviors of each agent are implemented using a set of appropriate formalisms; these formalisms-based model implementation can be composed seamlessly at C++ code level since they are all implemented based on SMP2, which provides unified function interfaces for their structural couplings and behavioral interactions.

IV. RELATED WORK

We focus on related work on composable modeling of ABM and classic modeling formalisms. In principle, ABM is an abstract modeling paradigm rather than a concrete modeling formalism and an unified agent modeling formalism for all application domains has not been proposed yet. However there are a lot for domain specific agent modeling formalism which combines ABM paradigm and domain specific problems. Currently agent-related multi-formalism modeling are performed at two levels:

(1) At the formalism (Metamodel) level, current research focus on how to incorporate other formalisms into ABM to build an **agent-based composable modeling formalism**. ABM provides a modeling framework to describe the autonomous and intelligent behavior of SoS subsystems, but it lacks the ability to model the behavior of state transition, action execution and so on. So it needs to combine other formalisms which can enhance the behavioral modeling capability to describe multiple aspects of SoS behavior. For example, Petri Net is used to describe the state transition of Agent and a new formalism called Agent Petri Net is proposed in [10]; and DEVS can be also used to model the Proactive/reactive and Concurrent behavior of agent [11].

(2) At the model level, current research focuses on how to **combine agent-based cognitive domain models and classic formalism-based physical domain models using adapters**. A typical example is the KIB (Knowledge Interchange Broker)-based method proposed by Arizona Center for Integrative Modeling & Simulation [12] which combines agent-based decision model and DEVS-based physical movement model. Another direction of this research is to combine agent-based behavioral model with formalism-based environment model (for example, cellular automata-based geographical environment model) via a discrete event interaction model [13].

V. CONCLUSIONS AND FUTURE WORK

This paper proposes an agent-centered multi-formalism modeling framework for SoS behavior based on the differentiation of SoS behavior and system behavior, which models the cognitive behavior of subsystems and other components using ABM and integrats other formalism-based physial behavior models based on the cognitive agent models. We choose SMP2 as the simulation model standard to implement the modeling framework based on an analysis of the feasible implementation solutions since SMP2 supports model reuse and simulation interoperation, which is of vital importance for large-scaled SoS behavior M&S. A Petri Net Example is given to illustrate how to generate SMP2-compliant C++ code from agent and classic formalism-based models.

There are plenty of future work to improve the modeling framework and enable industrial application, the most important among which are: firstly, to build a integrated modeling environment to support the approach as proposed in Figure 2; Secondly, to perform early verification and validation based on the formalism-based model; Thirdly, to show the whole M&S process of our approach by a more detailed case study.

ACKNOWLEDGMENT

The work presented in this paper is partially supported by National Natural Science Foundation of China (No.61273198, No.60974073, No.60974074 and No.71031007).

REFERENCES

[1] C. H. DAGLI and N. KILICAY-ERGIN, “System of Systems Architecting,” in System of Systems Engineering: Innovations for the 21st Century, M. Jamshidi, Ed. John Wiley & Sons, 2009.

[2] D. S. Alberts, NETWORK CENTRIC WARFARE: Developing and Leveraging Information Superiority. 2000.

[3] R. Wang and C. H. Dagli, “An executable system architecture approach to discrete events system modeling using SysML in conjunction with Colored Petri Net,” in IEEE International Systems Conference, 2008.

[4] B. Saulson and H. Beach, "SEAS ISR Architect: Mapping Department of Defense Architecture Views to Military Outcome," Journal of Defense Modeling and Simulation, vol. 3, no. 4, pp. 217–225, 2006.

[5] European Space Agency, "SMP 2.0 Handbook," 2004.

[6] W. Yu, W. Wang, Q. Li, and Y. Lei, "Model-driven and componentized development method of agent-based simulation models," System Engineering and Electronics (in Chinese), vol. 33, no. 8, pp. 221–226, 2011.

[7] Y. Lei, W. Wang, Q. Li, and Y. Zhu, "A transformation model from DEVS to SMP2 based on MDA," Simulation Modelling Practice and Theory, vol. 17, no. 10, pp. 1690–1709, Nov. 2009.

[8] N. Zhu, "A Study on Statecharts Modeling Framework Based on SMP2 (in Chinese)," National University of Defense Technology, 2011.

[9] Vanderbilt University, "GME Manual and User Guide- Generic Modeling Environment 10," 2010.

[10] B. Marzougui, "Toward a New Model of the Petri Nets: Agent Petri Nets," in UKSim Fourth European Modelling Symposium on Computer Modelling and Simulation, 2010, pp. 51–56.

[11] J. Müller, "Towards a Formal Semantics of Event-Based Multi-agent Simulations," pp. 110–126, 2009.

[12] H. Sarjoughian and D. Huang, "A Multi-Formalism Modeling Composability Framework: Agent and Discrete-Event Models," in Ninth IEEE International Symposium on Distributed Simulation and Real-Time Applications, 2005, pp. 249–256.

[13] G. R. Mayer, "Complexities of Simulating a Hybrid Agent-Landscape Model Using Multi-Formalism Composability," in Spring Simulation Multi-conference, 2007, pp. 161–168.

Published by Atlantis Press, Paris, France.