Proceedings of the 2013 Winter Simulation Conference
R. Pasupathy, S.-H. Kim, A. Tolk, R. Hill, and M. E. Kuhl, eds

A DSM-BASED MULTI-PARADIGM SIMULATION MODELING APPROACH FOR COMPLEX SYSTEMS

Xiaobo Li
Yonglin Lei
Weiping Wang
Wenguang Wang
Yifan Zhu

College of Information System and Management
National University of Defense Technology
Changsha, Hunan 41007 P.R.CHINA

ABSTRACT

Complex systems contain hierarchical heterogeneous subsystems and diverse domain behavior patterns, which bring a grand challenge for simulation modeling. To cope with this challenge, the M&S community extends their existing modeling paradigms to promote reusability, interoperability and composability of simulation models and systems; however, these efforts are relatively isolated and limited to their own technical space. In this paper, we propose a domain specific modeling (DSM)-based multi-paradigm modeling approach which utilizes model driven engineering techniques to integrate current M&S paradigms and promote formal and automated model development. This approach constructs a simulation model framework to architect the structure of the overall simulation system and combines multiple M&S formalisms to describe the diverse domain behaviors; moreover, it provides domain specific language and environment support for conceptual modeling based on the model framework and formalisms. An application example on combat system effectiveness simulation illustrates the applicability of the approach.

1 INTRODUCTION

In the last decades, complex systems have become a research focus of academia and industry. Complex systems usually have hierarchical structure and heterogeneous subsystems, and involve a diversity of application domains. When interacting with the environment and other systems, the complicated structures of complex systems can generate non-linear behaviors which can not be acquired by linearly adding subsystems together (e.g., emergent behavior). Thus, complex system research brings grand challenges for existing scientific research and engineering practices. Enabled by the development of computer hardware and software, modeling and simulation (M&S) is being widely used in all research fields and exhibits promising potential in complex system research. M&S significantly reduces the time and cost of complex system development and it is especially useful in the overall design and demonstration phase. Moreover, M&S has an irreplaceable role when physical experiments are unavailable or highly expensive.

Complex systems are structurally heterogeneous and large-scaled, thus they are difficult to model appropriately and simulate efficiently. Currently, M&S research in each domain is relatively isolated; therefore, the comprehensive utilization of domain knowledge and M&S paradigms in multiple domains needs to be systematically studied. Moreover, many M&S researchers try to deal with complex system by improving existing methodologies and techniques based on the same technical space as that of M&S research on simple systems, but this improvement is limited to satisfy the multi-subsystem, multi-domain characteristics of complex systems.

To cope with the problem difficulty and technique complexity in simulation model development for complex systems, the M&S community has launched research on model reuse, simulation interoperability and model composability recently. This research concentrates on developing simulation models and systems based on certain M&S paradigms, such as simulation protocols, model specifications, and formalisms, to promote the reuse potentials and extend the application ranges of simulation resources.

The requirements of reusability, interoperability and composability on simulation models and systems indicate that the simulation model development process should be formalized and automated to enable efficient and effective development of complex simulation systems. Problem analysis and conceptual modeling should be formalized to provide a formal conceptual background for the following M&S activities in the problem domain, and formalisms should be used to build formal simulation models and support formal analysis of source systems. Moreover, generative techniques (e.g. model transformation, code generation) should be used to automatically generate lower level implementation or models from higher level models. Formal and automated model development process can separate problem domain from technical domain, raise the modeling abstraction level, promote efficiency, and thus support reuse, interoperation and composition of simulation resources at higher abstraction levels.

Multi-paradigm modeling (MPM) addresses complex system design and implementation issues combining the following orthogonal dimensions: model abstraction, multi-formalism modeling and metamodelling; and it highlights the importance of model transformation to link models of different formalisms at different abstraction levels (Mosterman and Vangheluwe 2004). MPM promotes formal modeling by multi-formalism modeling and enables automated implementation of models via model transformation. Thus, MPM can utilize the comprehensive fruits of current M&S paradigms in various application domains and tackle the multi-domain and multi-subsystems characteristics of complex systems. However, MPM does not provide enough support for conceptual modeling and simulation model integration.

In this paper, we introduce a DSM-based multi-paradigm modeling approach which raises the modeling abstraction level and supports the integration of domain specific models using a simulation model framework. This approach specifies a model development cycle with domain specific conceptual modeling, formal model design, automated model implementation and seamless model integration, and thus supports efficient and rapid simulation model development for complex systems.

The rest of the paper is organized as follows. Section 2 overviews related work and Section 3 presents the proposed approach. Section 4 applies this approach to combat system effectiveness simulation, and section 5 concludes this research and proposes future work.

2 RELATED WORKS

State-of-the-art simulation modeling methods try to cope with the difficulties in complex system simulation modeling from different aspects, each of which aims to achieve one or more of the following goals: (1) construct problem domain oriented language or environment to provide domain specific modeling concepts for domain modelers; (2) separate the conceptual model, the design model and the model implementation to raise the abstraction level of formal model representation; (3) multi-formalism modeling and simulation to support formal analysis and early verification and validation; (4) automatically generate executable simulation models to promote development efficiency and maintenance; (5) provide a common specification for simulation model and simulator to integrate the simulation models and promote reuse, interoperability and composability of simulation resources. According to the goals and underlying principles, state-of-the-art simulation modeling methods can be categorized into three groups as follows.

2.1 Unified Modeling Methods

These methods support unified modeling of complex systems from three aspects: the first is unified modeling using multi-domain concepts, e.g., Modelica; the second is unified modeling based on multiple formalisms, e.g., UML, DEVS; the third is simulation modeling based on a unified simulation model framework or architecture, e.g., the Department of Defense Architecture Framework (DoDAF).

Modelica supports unified descriptions of domain concepts of different physical systems based on mathematical equations. Modelica provides domain modeling support via extended model library tailored for typical domains. Modelica also support hybrid modeling by integrating formalisms such as finite state machine and DEVS. Recently, ModelicaML is proposed to combine SysML and Modelica to support the whole model development cycle (Schamai et al. 2009).

DEVS is a hierarchical and modular formalism which supports unified description of various formalisms, and it is widely accepted by M&S community as a common theoretical foundation of M&S framework. Many DEVS-based M&S frameworks are proposed to facilitate complex system research, e.g., DEVS unified process framework (Mittal 2007). Though DEVS itself provides no domain extension mechanism, continuous efforts are made to empower DEVS platforms with domain customization capability (Ferayorni and Sarjoughian 2007).

UML provides an integrated modeling environment with a series of modeling formalisms to model different aspects of the software. It is also used for complex system simulation modeling since the simulation models are technically a kind of software. UML uses profile mechanism for domain customization and it is combined with DEVS to be more executable (Risco-Martín et al. 2009). However, UML is software-oriented and lacks simulation modeling support.

A unified simulation model framework (or architecture) describes the main components in its subdomains and their fundamental relationship, guides the simulation model development, and provides a foundation for sub-model integration. A model framework is built based on domain knowledge and simulation model expertise which is application independent, and different simulation applications are instantiated from the model framework. One of the most successful examples of this method is DoDAF in the defense domain which specifies a model framework for complex defense systems. More examples can be found in other complex system research fields, such as supply chain (Chatfield, Hayya, and Harrison 2007). Though model framework-based modeling method can efficiently support rapid development of new simulation applications; it is domain-oriented and implementation-focused, thus model frameworks themselves lack generality and thus are difficult to be extended.

2.2 Combined Modeling Methods

Combined modeling methods compose models of different formalisms and simulate them integratively. Combined modeling can be performed at two levels: the first is the interoperation of simulation systems via simulation data at the data level (a.k.a. co-simulation); the second is model composition of different formalisms at the semantics level (a.k.a. multi-modeling).

HLA is a representative of the first kind which links simulators of different simulation systems. HLA concentrates on simulation interoperability but lacks model specification, thus it is not suited to use HLA to perform simulation modeling for complex systems. However, it can be used to integrate heterogeneous models of different formalisms. There are two directions to enable HLA for complex M&S: first, basic object model (BOM) is proposed to enhance its ability for model description; second, RTI is extended using SOA techniques to enable interoperability at a broader scope.

The tool Simulink is a typical example of multi-formalism modeling which supports the composition of models of casual block diagram and stateflow. Simulink could be considered as a DSM platform since it provides DSM supports for digital signal processing and simulations (Kelly and Tolvanen 2008). However, more formalisms should be integrated to provide DSM support for more application domains as far as complex systems are concerned.

Ptolemy II is another typical example which supports multi-formalism modeling via the Director/Actor mechanism (Eker et al. 2003), and a diversity of common formalisms, such as synchronous data flow, continuous time, discrete event, and communication sequence process, have been implemented in this tool. Ptolemy II can implement domain modeling concepts as model libraries and support composable modeling based on the model libraries.

2.3 MDE-based Modeling Methods

The software engineering community proposed a new software development paradigm called model driven engineering (MDE) which uses models as the first-class artifacts in the software development process to promote development efficiency. MDE aims to construct formal, implementation-independent models using modeling languages (not programming language) and generate code from them. MDE has attracted much attention of M&S community recently, and there are several discussion panels (e.g., annual Mod4Sim workshop in spring simulation conference) and methodologies (e.g., MDD4MS (Cetinkaya, Verbraeck, and Seck 2011)) which aim to promote the application of MDE in M&S. There are two branches of MDE: model driven architecture (MDA) and model integrated computing (MIC).

2.3.1 MDA-Enabled Simulation Modeling

MDA is proposed by the Object Management Group to solve the software architecture problem and the integration of different tools using architecture models. MDA specifies a series of technical standards, such as meta-object facility for metamodelling, XML metadata interchange for model storage and interchange, UML for modeling, query/view/transformation language for model transformation and so on. MDA promotes the usage of platform independent model and platform specific model to separate model design and model implementation, and uses model transformation to generate platform specific model (PSM) from platform independent model (PIM). M&S community use UML to construct PIM of simulation models and generate object oriented code for model implementation. However, UML is not designed for simulation modeling and thus lacks description of essential simulation concepts. MDA is also applied to HLA system development and many HLA tools have realized the support for MDA.

Simulation model portability standard 2 (SMP2) is a successful application of MDA in simulation, which was proposed in 2004 by the Europe Space Agency to promote model reuse and portability (European Space Agency 2005). It uses simulation model definition language (SMDL) to define the common representation of PIM for simulation models, and exploits code generation to generate C++ code from PIM. SMP2 has been widely used in aeronautical simulation system and combat simulation systems (Li et al. 2010). Though SMP2 conforms to MDA principle and promote development efficiency, it has weaknesses which hamper its application in complex system M&S: SMP2 mainly specifies the static structure of simulation systems and fails to support behavioral modeling, thus modelers need to manually fill the behavioral code; it is a low level model specification which provides only technical modeling concepts, such as class, container, interface, and lacks DSM support for application domain.

2.3.2 MIC-Enabled Simulation Modeling

MIC aims to provide domain specific modeling solutions for problem domains instead of a unified modeling language and a suite of common technical standards. MIC contains the following steps: construct domain specific modeling languages for problem domains based on metamodelling, generate domain specific modeling environments for domain modelers, and automatically implement domain specific models by semantic anchoring. M&S activities are essentially domain problem oriented, thus DSM for simulation has been widely studied.

DSM research in M&S can be divided into two categories: the first is to use DSM method in simulation system development from the aspect of software; the second is domain specific simulation modeling of source system. The first category belongs to the software engineering field, e.g., the research presented in (Hemingway et al. 2012) uses DSM to automatically generate engine configuration files and glue code from a integration model to integrate heterogeneous HLA systems. In this paper we concentrate on the second category, which has the following three aspects of concern (Li et al. 2011): (a) from the application aspect, researchers construct domain specific modeling language (DSML) and environment (DSME) from scratch or based on M&S formalisms to build problem-oriented solution,; (b) from the tool aspect, generic M&S tools and platforms are extended to provide domain support; (c) from the M&S development cycle aspect, M&S community aims to incorporate DSM into the whole M&S cycle via framework support for domain knowledge integration or adaptation of software engineering techniques to enable DSM for M&S system development. Though there is plenty of research on DSM for simulation, how to compose domain specific models for complex system M&S still needs to be explored since DSML composition in software engineering community can not solve this problem completely (Li et al. 2011).

Another important branch in MIC is MPM, which uses state-of-the-art M&S and software engineering techniques to deal with the heterogeneity of the complex systems (C. Hardebolle and Boulanger 2009). Successful applications of MPM includes OsMoSys (Vittorini et al. 2004), ModHel'X (Hardebolle and Boulanger 2007), Maya (Zhou et al. 2003). Although DSM has attracted the attention of MPM researchers and been applied in certain domains (Lara, Levendovszky, and Mosterman 2009), how to incorporate DSM into a MPM-based modeling framework for complex systems remains an open issue.

Summary. The categorization of three groups is neither absolute nor complete, and there is a trend to model the complex systems using combinations of these methods, e.g., DEVS/HLA. These methods have their own advantages and disadvantages as far as the five goals are concerned. Generally speaking, each of unified methods is based on a powerful unified modeling formalism with its corresponding technical space, thus this kind of methods supports model reuse, model composition and simulation interoperability. However, the heterogeneity of complex systems is not appropriately addressed since no single formalism can describe all aspects of complex systems. Combined methods exploit multiple formalisms and combine heterogeneous models, but they lack the support for modeling at higher abstraction levels, and model reuse and composition issues arise. MDE methods have the supports of modeling paradigms, technical standards and code generation tools, thus they facilitate the model development process of complex systems and provide high level modeling support. Actually, MPM can be either categorized into the combined methods or MDE methods since its technical essence is using MDE techniques to support multi-formalism modeling. So MPM has a promising potential in simulation modeling and we explicitly study how to incorporate DSM into MPM in this paper.

3 A DSM-BASED MULTI-PARADIGM SIMULATION MODELING APPROACH

3.1 Model Driven Engineering for Simulation Model Development

As discussed in Section 2.3, model driven development of simulation models and systems is becoming prevalent since it promotes development efficiency and model quality by using a formal and automated development cycle. MDE has promising potential in simulation modeling for complex systems for the following reasons. (1) Simulation application domain usually has relatively explicit structure framework and behavioral patterns to support PIM design for simulation model, thus it provides a sound foundation for MDA utilization. (2) Many simulation modeling languages and environments have been developed for specific application domains; though many of them are based on isolated technical spaces and at implementation level, they still pave the way for MIC-based simulation modeling since they have plentiful knowledge of domain engineering and language engineering. (3) Wide application of SMP2 indicates that MDA-based simulation modeling promote development efficiency and model reuse, but DSML and M&S formalisms should be integrated into SMP2 to raise the abstraction level and strengthen behavioral modeling capability. (4) MDA and MIC can offset mutual disadvantages: MIC provides DSM support and formalisms unification support (via metamodelling and model transformation) to enhance MDA (especially SMP2); while MDA provides common model specification and technical standards. Thus MDA and MIC used in combination will significantly improve the efficiency of simulation model development.

Simulation model development usually involves a series of activities including problem formulation, requirement analysis, conceptual modeling, model design, model implementation, and model integration (Balci 2012). MDE can be used for simulation model development since the technical essence of computer simulation models is a special kind of software. MDE develops formal models (instead of documents) as the products in the model development cycle, and uses generative techniques (such as model transformation and code generation) which can be executed by the computer to automatically achieve the transformation between model products of different phases or implement the higher level models. MDE can achieve higher-level interoperability in the LCIM since different aspects and development phases are formally described, and the technical standards specified by MDA provide a technical foundation for re-use, interoperation, composition of simulation resources from the aspect of software implementation.

3.2 A DSM-based MPM Approach to Enable Formal and Automated Model Development Cycle

As shown in Figure 1, we propose a DSM-based MPM approach for simulation modeling of complex systems, which comprehensively utilizes the following state-of-the-art modeling methods: MDE employs formal models to represent products of different phases, and utilizes model transformation and code generation to automate the development process; DSM is evolving as the centerpiece of MDE which provides domain specific solutions for problem domains, thus it is especially useful to model complex systems of multiple domains; M&S formalisms are important research fruits and irreplaceable foundations of M&S endeavor; A model framework is the abstraction of the knowledge based on M&S experience on the system under study (SUS), which provides the system architecture to guide the development process and sub-model integration. This approach combines multiple M&S paradigms based on MDE techniques and consists of the following four key steps for simulation model development.

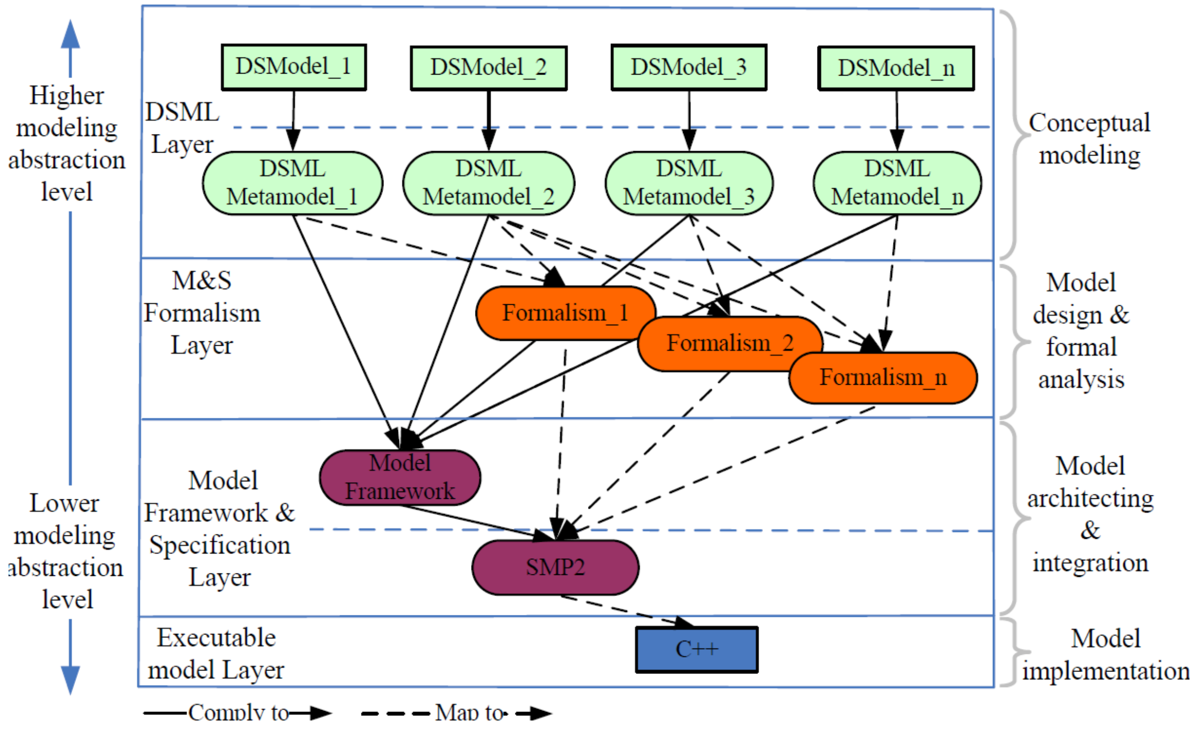

Figure 1: A DSM-based MPM approach for simulation modeling of complex systems

3.2.1 DSM-based Conceptual Modeling

Conceptual modeling remains a tough issue in simulation model development cycle since it is difficult to find a precise and formal way to represent the conceptual model which contains highest-level concepts, constraints, and assumptions (Balci 2012). It is inefficient to extract useful information from conceptual models for simulation modeling, and the consistency between simulation models and conceptual models is hard to maintain. Conceptual modeling of complex systems is extremely important for multiple subsystems and domains are involved in model development life cycle, and there are numerous efforts which focus on formalize conceptual models. A typical example is model theory (Tolk, Diallo, and Padilla 2011) which provides a mathematical foundation for conceptual models, and another example is diagramming techniques for building formal conceptual models (Heath, Ciarallo, and Hill 2012).

Recently, DSM is recognized as a new conceptual modeling paradigm (Setavoraphan and Grant 2008)(Setavoraphan 2009) since it provides DSML and DSME tailored for problem domains. The conceptual models can be built as formal models with high-level concepts and stored as computer-processable documents. As shown in the DSM layer in Figure 1, the complexity of complex systems modeling can be reduced by decomposing the complex system into a series of problem domains and building domain specific conceptual models for each domain.

The metamodel design of DSML is an essential step of DSM which involves joint efforts of domain experts, language engineers and M&S experts. To enable DSM for multi-domain and domain model integration, we propose a metamodeling approach which combines model framework-based structure modeling and formalism-based behavior modeling. The DSML metamodel imports the corresponding structural information for the specific domain from the framework, including entity hierarchy and interfaces to other part of the model framework. The behavior patterns of each domain are depicted based on classical M&S formalisms, such as DEVS for discrete-event behavior and Modelica for continuous-time behavior. These formalisms can be combined or specialized to depict the characteristics of problem domains. Domain-specific models can be either transformed to formalism-based models which guide simulation model design and support formal analysis, or implemented as executable codes for simulation by code generators.

3.2.2 Model Framework-based Model Architecting and Integration

A model framework is the architecture of the simulation system and provides a structural foundation for the integration of simulation models of sub-systems across different domains. A model framework itself is technology-independent and should be described using high level concepts. However, current model frameworks are usually implemented based on certain technical specifications or modeling paradigms. Examples include Agent-based model framework for SEAS, HLA/BOM specification for distributed simulation systems. We build our model framework based on SMP2, a MDA-compliant model specification, to provide a PIM for the model framework in this approach.

The model framework or simulation system architecture plays an irreplaceable role in complex simulation systems overall design and architecting. There is always an overall design scheme, whether it is implicit or explicit, proposed in the overall design phase and guides the whole development process. Model framework is abstracted from the main structure and fundamental relationships of the SUS, and it explicitly represents the overall architecture from the aspect of simulation modeling. Thus model framework is the first class artifact in the development of complex simulation systems, which provides structural foundations for each domain and their coupling relationships.

According to the three abstraction levels of MDA, a model framework is a computation independent model of the overall structure of the whole simulation system; we use SMP2 to implement the model framework. The SMP2-compliant model framework specifies the main components, their relationships, and the simulator interfaces, which provide a technical foundation for the integration of models of different subsystems across various domains at the model integration phase.

3.2.3 M&S Formalism-based Model Design and Formal Analysis

M&S formalism includes model specification and simulator algorithm, which is the formal abstraction of M&S paradigm and experience. M&S community has developed and utilized many M&S formalisms to model different aspects of SUS. Formalisms support model design at a higher abstraction level and formal analysis since they are built based on strict mathematical foundations. Though several formalisms are able to provide M&S support for complex systems as a unified formalism (as discussed in subsection 2.1), a formalism usually can only describe only parts of characteristics of SUS. Multiple formalisms should be integrated to comprehensively describe a problem domain. Additionally, one formalism can describe more than one domain since it is abstracted from domain-independent behavioral patterns.

We use M&S formalisms to support behavioral modeling of complex systems and provide denotational semantics for DSML in the formalism layer. Complex system simulation modeling is an intricate process, thus formal analysis is important for early model verification and validation. Moreover, a single formalism supports DSM for multiple domains, so formal analysis methods (e.g., state reachability) can perform overall analysis on complex system models from various domains. There are several super formalisms, such as DEVS, Modelica, and different formalisms can be transformed to these formalisms; in this way model design and formal analysis can be performed based a unified formalism.

Normally, we only use M&S formalism for formal analysis and early verification and validation. However, in some cases formalisms are necessary to be transformed to SMP2 and thus support the integration of formalism-based models: the first is when part of the characteristics of the simulation system are the only concern and this part can be sufficiently described by the formalisms; the second is when some models of the simulation system are directly built using formalisms in their supporting environments (i.e., no domain specific models are available) and the simulation system needs to integrate these models into a integrative simulation environment. We have study the transformation from DEVS to SMP2 (Lei et al. 2009) and Statecharts to SMP2 (Zhu et al. 2012).

3.2.4 SMP2/C++ Transformation-based Model Implementation

Currently, only C++ is available to construct PSMs for PIM implementation. SMP2 specification includes a SMP2/C++ mapping (European Space Agency 2004), based on which we build a C++ code generator for SMP2 using Eclipse. Since SMP2 mainly specifies the structural aspect, a majority part of the behavioral code is generated from domain specific models using corresponding code generators for DSMLs. The two parts are integrated under the SMP2 framework and built into executable components.

3.3 A Technical Implementation Solution for the Proposed Approach

The technical implementation solution for the proposed approach is presented in Figure 2, including the following essential steps. (1) Based on the knowledge and M&S experience of the system a model framework is constructed as the backbone of the simulation system. (2) According to language engineering of DSML, it comprises abstract syntax (including constraints), concrete syntax, and semantics (Kelly and Tolvanen 2008). The metamodel of a DSML is designed based on domain specific knowledge under the guidance of the framework and concrete syntax is attached to each element of the metamodel. Based on the metamodel a DSME is generated using DSM platforms (we use GME in this approach). (3) Domain modelers build conceptual models with DSML in DSME. The semantics of DSML is achieved either by semantic anchoring which transforms domain specific models to formalism-based models based on the metamodel mapping of their abstract syntax, or by code generation which generates executable code from domain specific models. (4) Modelers who are familiar with certain formalisms can design simulation models base on the formalisms using their supporting tools. Then formal analysis and early verification and validation can be performed on these models. Formalism based models can be also transformed to behavioral code to enable composable simulation. (5) The behavioral code is combined with the framework code generated from the SMP2-compliant model framework, and these two kinds of code are built together as executable simulation model components which are stored in an executable simulation model component library. (6) Complex simulation application users select the components and compose a simulation application according to their requirements. Then they run the application in the integrative simulation environment and analyze the simulation results.

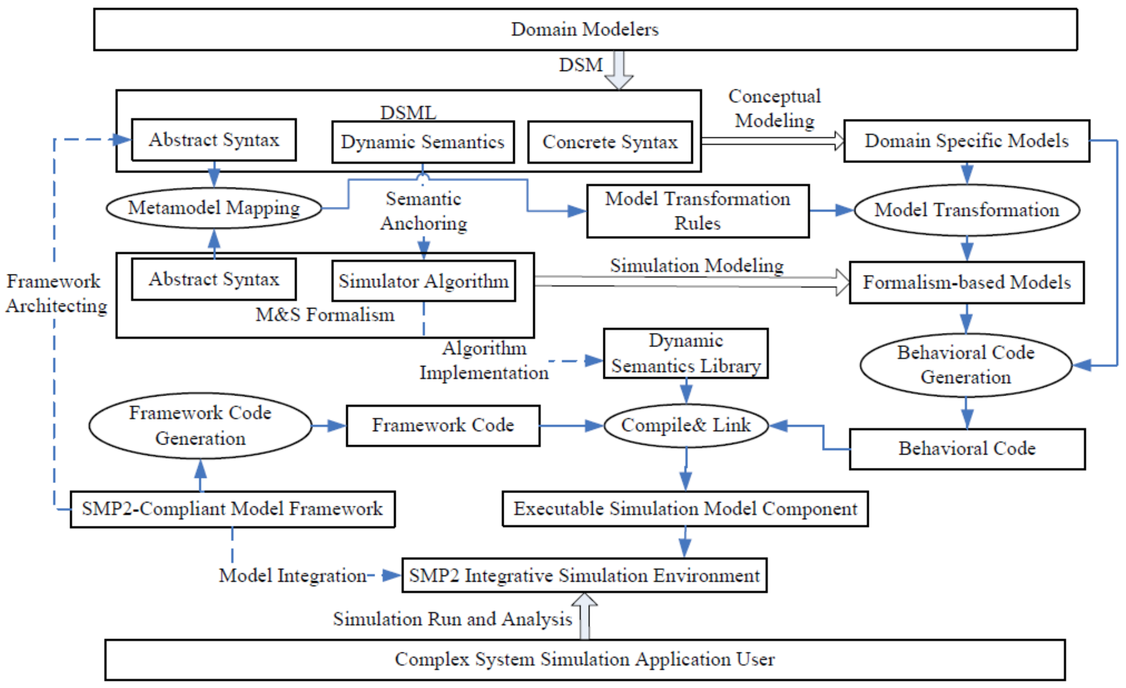

Figure 2: A technical implementation solution for the modeling approach

This technical implementation solution comprehensively utilizes MDE techniques and M&S paradigms to support formal and automated model development: DSM provides domain modeling supports and separate conceptual models and simulation models; M&S formalisms are used in model design and formal analysis; the model framework is constructed based on SMP2 to integrate simulation models and promote model reusability and composability; model transformation and code generation are employed during the whole cycle to automatically generate lower-level artifacts from higher-level products.

4 APPLICATION EXAMPLE: SIMULATION MODELING FOR COMBAT SYSTEM EFFECTIVENESS MEASUREMENT

Combat system effectiveness simulation (CoSES) is a typical example of M&S research on complex systems. Combat systems usually comprise multiple subsystems (e.g., combat platforms carry sensors, weapons and communication devices) from different domains (e.g., physical domain, information domain, and cognitive domain). Combat system modeling and simulation is researched at a series of abstraction levels: theatre/campaign level, mission/battle level, engagement level, and engineering level from top to down (Davis and Bigelow 1998). In this research, we concentrate on CoSES at the engagement level, and we do not consider social domain behaviors. Model reuse and composability is of vital importance to promote development efficiency in CoSES. As shown in Figure 3, we build a model development framework for CoSES based on the proposed MPM approach, which contains the following four modeling layers.

(1) The model framework layer comprises two parts: the first is a physical domain framework (for simplicity we incorporate information domain into physical domain) which is the backbone of the simulation system, and it specifies the affiliated combat models and their relationships; the second is a decision script framework which supports the script-based implementation of decision models. These two frameworks are connected by a cognitive decision interface framework, which is technically a set of Python/C++ transformation functions. The whole model development process is based on the model framework layer which specifies the system structure and integrates subsystem models from different domains.

(2) The behavioral representation layer, i.e., the model implementation layer, uses C++ to implement physical domain models and Python to implement cognitive domain models. The physical domain model

is implemented using SMP2/C++ since physical behaviors are relatively stable and can be hard-coded in C++. The decision script framework is implemented using Python since decision behavior is flexible and model revisions take effect immediately without rebuilding. Eventually all Python code are transformed to C++ code for integrative simulation in SMP2 environment.

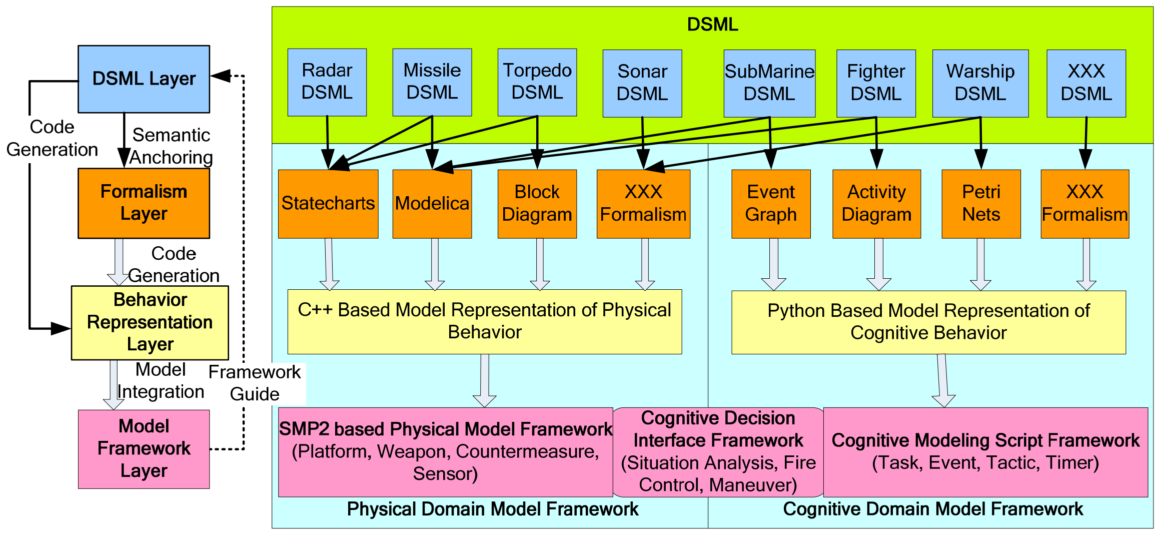
DSML Layer
Code Generation
Semantic Anchoring
Formalism Layer
Code Generation
Behavior Representation Layer
Model Integration
Framework Guide
Model Framework Layer
DSML
Radar DSML
Missile DSML
Torpedo DSML
Sonar DSML
SubMarine DSML
Fighter DSML
Warship DSML
XXX DSML
Statecharts
Modelica
Block Diagram
XXX Formalism
Event Graph
Activity Diagram
Petri Nets
XXX Formalism
C++ Based Model Representation of Physical Behavior
Python Based Model Representation of Cognitive Behavior
SMP2 based Physical Model Framework (Platform, Weapon, Countermeasure, Sensor)
Cognitive Decision Interface Framework (Situation Analysis, Fire Control, Maneuver)
Cognitive Modeling Script Framework (Task, Event, Tactic, Timer)
Physical Domain Model Framework
Cognitive Domain Model Framework

Figure 3: A model development framework for CoSES

(3) The formalism layer utilizes common M&S formalisms to provide behavioral semantics for domain specific models. Statecharts, Modelica and block diagram are used in physical domain for physics modeling; event graph, activity diagram and Petri Nets are used in cognitive domain for decision modeling since they are suitable to describe combat decision.

(4) The DSML layer provides DSME and DSML for typical SUS domains such as missiles, radars, warships and so on. A DSML metamodel uses the corresponding structural information in the CoSES model framework to design its structural aspect of the metamodel, and chooses appropriate M&S formalisms based on its domain behavioral patterns to design its behavioral aspect.

In this model development framework, Python scripting language is used as a supplement of C++ to provide flexible decision modeling support. The framework is still using C++ for model implementation and Python script are converted to C++ during simulation execution. We have studied decision modeling for CoSES using this model development framework in (Li et al. 2013), constructed a DSML for decision modeling, and used Petri Net to perform formal analysis. Though more works are needed to implement the whole framework, the results and insights gained in (Li et al. 2013) showed that the development efficiency was promoted and model reusability and composability was enhanced by the proposed approach.

5 CONCLUSIONS AND FUTURE WORK

Complex systems have hierarchical heterogeneous subsystems and diverse domain behavior patterns, which bring a grand challenge for simulation modeling. To enable efficient and effective M&S for complex systems, M&S community extends their existing technical space to promote reusability, interoperability and composability of simulation models and systems. Based on an overview of state-of-the-art simulation modeling methods, we insist that the model development process should be formalized and automated by a comprehensive utilization of current M&S paradigms and MDE techniques. Therefore, we propose a DSM-based MPM approach for complex systems. In this approach, a model framework is constructed to architect the structure of the overall simulation system and multiple M&S formalisms are combined to describe the diverse domain behaviors; moreover, DSMLs and DSMEs are built to provide highest-level modeling support. This approach provides an efficient collaborative framework to integrate joint efforts from domain experts, M&S users, M&S experts, software engineering experts and other M&S participants. The application example illustrates the applicability of the approach and shows that this approach can promote the development efficiency and the quality of the simulation model.

We plan to continue working on the following aspects in the future. First, more formalisms and their environments can be incorporated into the approach; second, physical domain modeling in CoSES should be studied to validate the approach with the whole combat simulation system; finally, we need to extend the application range of this approach to other typical kinds of complex systems (e.g., traffic networks).

ACKNOWLEDGMENTS

We thank the anonymous reviewers for their valuable suggestions. We are grateful to the suggestions and insights provided by Professor Hans Vangheluwe from University of Antwerp and Professor Pieter Mosterman from McGill University. This work is partly supported by the National Natural Science Foundation of China (No.61273198) and the scientific research fund of Hunan provincial education department.

REFERENCES

Balci, Osman. 2012. "A Life Cycle for Modeling and Simulation." *Simulation* 88 (7): 870–883.

Cetinkaya, Deniz, Alexander Verbraeck, and Mamadou D. Seck. 2011. "MDD4MS: a Model Driven Development Framework for Modeling and Simulation." In *Proceedings of the 2011 Summer Computer Simulation Conference*, 113–121.

Chatfield, DC, JC Hayya, and TP Harrison. 2007. "A Multi-formalism Architecture for Agent-based, Order-centric Supply Chain Simulation." *SIMUL MODEL PRACT TH* 15 (2): 153–174.

Davis, Paul K, and James H Bigelow. 1998. *Experiments in Multiresolution Modeling*. *RAND Corporation*. Santa Monica, CA: RAND Corporation.

Eker, Johan, Jörn W Janneck, Edward A Lee, J I E Liu, Xiaojun Liu, Jozsef Ludvig, Stephen Neuendorffer, Sonia Sachs, and Yuhong Xiong. 2003. "Taming Heterogeneity — The Ptolemy Approach." *Proceedings of the IEEE* 91 (1): 127–144.

European Space Agency. 2004. *SMP2.0 C++ Mapping EGOS-SIM-GEN-TN-0102 Issue 1 Revision 0*.

European Space Agency. 2005. *SMP2.0 Handbook (Issue 1 Revision 2) EGOS-SIM-GEN-TN-0099*.

Ferayorni, Andrew E, and Hessam S. Sarjoughian. 2007. "Domain Driven Simulation Modeling for Software Design." In *Proceedings of the 2007 Summer Computer Simulation Conference*, 297–304.

Hardebolle, C., and F. Boulanger. 2009. "Exploring Multi-Paradigm Modeling Techniques." *Simulation* 85 (11-12): 688–708.

Hardebolle, Cecile, and Frederic Boulanger. 2007. "ModHel'X: A Component-Oriented Approach to Multi-Formalism Modeling." In *Workshop on Multi-Paradigm Modeling: Concepts and Tools*, 1–10.

Heath, B.L., F.W. Ciarallo, and R.R. Hill. 2012. "Validation in the Agent-based Modelling Paradigm: Problems and a Solution." *Int. J. Simulation and Process Modelling* 7 (4): 229–239.

Hemingway, Graham, Himanshu Neema, Harmon Nine, Janos Sztipanovits, and Gabor Karsai. 2012. "Rapid Synthesis of High-level Architecture-based Heterogeneous Simulation: a Model-based Integration Approach." *Simulation* 88 (2) : 217–232.

Kelly, Steven, and Juha-pekka Tolvanen. 2008. *Domain Specific Modeling: Enabling Full Code Generation*. Wiley-IEEE Computer Society Press.

De Lara, J., T. Levendovszky, and P. J. Mosterman. 2009. "Guest Editorial: Special Issue on Multi-paradigm Modeling." *Simulation* 85 (11/12): 685–687.

Lei, Yonglin, Weiping Wang, Qun Li, and Yifan Zhu. 2009. "A Transformation Model from DEVS to SMP2 Based on MDA." *SIMUL MODEL PRACT TH* 17 (10): 1690–1709.

Li, Qun, Yonglin Lei, Hongtao Hou, and Weiping Wang. 2010. *Simulation Model Portability Standard 2 and Its Application*. Beijing: Publishing House of Electronics Industry. (in Chinese)

Li, Xiaobo, Yonglin Lei, Hans Vangheluwe, Weiping Wang, and Qun Li. 2011. "Towards a DSM-based Framework for the Development of Complex Simulation Systems." In *Summer Computer Simulation Conference*, 210–215.

Li, Xiaobo, Yonglin Lei, Hans Vangheluwe, Weiping Wang, and Qun Li. 2013. "A Multi-paradigm Decision Modeling Framework for Combat System Effectiveness Measurement Based on Domain-specific Modeling." *Journal of Zhejiang University-SCIENCE C* 14 (5): 311–331.

Mittal, Saurabh. 2007. "DEVS Unified Process for Integrated Development and Testing of Service Oriented Architectures". Ph.D. Thesis, University of Arizona.

Mosterman, Pieter J, and Hans Vangheluwe. 2004. "Computer Automated Multi-Paradigm Modeling: An Introduction." *Simulation* 80 (9): 432–450.

Risco-Martín, José L, Jesús M. de la Cruz, Saurabh Mittal, and Bernard P. Zeigler. 2009. "eUDEVS: Executable UML with DEVS Theory of Modeling and Simulation." *Simulation* 85 (11-12): 750–777.

Schamai, Wladimir, Peter Fritzon, Chris Paredis, and Adrian Pop. 2009. "Towards Unified System Modeling and Simulation with ModelicaML." In *Proceedings of the 1st International Workshop on Equation-Based Object-Oriented Languages and Tools*, 13–24.

Setavoraphan, Kitti. 2009. "A Methodology for Development of Domain Specific Simulation Applications and Environments." Ph.D. Thesis. The University of Oklahoma, Norman, Oklahoma.

Setavoraphan, Kitti, and Floyd H Grant. 2008. "Conceptual Simulation Modeling: The Structure of Domain Specific Simulation Environment." In *Proceedings of the 2008 Winter Simulation Conference*, ed. S. J. Mason, R. R. Hill, L. Mönch, O. Rose, T. Jefferson, J. W. Fowler, 975–986.

Tolk, A, SY Diallo, and Jose J. Padilla. 2011. "Model Theoretic Implications for Agent Languages in Support of Interoperability and Composability." In *Proceedings of the Winter Simulation Conference*, ed. S. Jain, R.R. Creasey, J. Himmelspach, K.P. White, and M. Fu, 309–320.

Vittorini, V., M. Iacono, N. Mazzocca, and G. Franceschinis. 2004. "The OsMoSys Approach to Multi-formalism Modeling of Systems." *Software and Systems Modeling* 3 (1): 68–81.

Zhou, J., Z. Ji, M. Takai, and R. Bagrodia. 2003. "Maya: a Multi-paradigm Network Modeling Framework for Emulating Distributed Applications." In *Seventeenth Workshop on Parallel and Distributed Simulation,* 163–170. IEEE Comput. Soc.

Zhu, Ning, Xiaobo Li, Yonglin Lei, and Weiping Wang. 2012. "Transforming Statecharts to SMP2 for Simulation Modelling of Complex Systems." In *2nd International Conference on Computer Science and Network Technology*, 1267–1271. Changchun, China.

AUTHOR BIOGRAPHIES

XIAOBO LI is a Ph.D. Candidate in the College of Information System and Management (CISaM) at National University of Defense Technology (NUDT), and also a PhD student in the Department of Mathematics and Computer Science at University of Antwerp. His research interest includes model driven engineering techniques for M&S, simulation-based system design and demonstration, and domain specific modeling. His email address is lixiaobo.nudt@gmail.com and his website is http://msdl.cs.mcgill.ca/people/xiaobo.

YONGLIN LEI is an associate professor in CISaM at NUDT. His research interest includes simulation model engineering, and simulation-based system design and demonstration. His email address is samuelyonglin@gmail.com.

WEIPING WANG is a professor in CISaM at NUDT. His research interest focuses on system of systems engineering, and simulation-based system design and demonstration. His email address is wang.wp2010@gmail.com

WENGUANG WANG is a lecturer in CISaM at NUDT. His research interests include interoperability for system of systems, service-oriented composable simulation, and HLA. His email address is wgwangnudt@gmail.com

YIFAN ZHU is a professor in CISaM at NUDT. His research interest includes simulation-based system design and demonstration, and agent-based modeling and simulation. His email address is stephen.zhuyifan@gmail.com.