ARCHITECTURE-ORIENTED COMBAT SYSTEM EFFECTIVENESS SIMULATION MODELING

Yonglin Lei [(a)], Ning Zhu [(b)], Jian Yao [(c)], Zhi Zhu[(d)], Shuai Chen[(e)]

[(a),(b),(c), (d),(e)] School of Information Systems and Management, National University of Defense Technology, 109 Yanwachi Rd, Changsha, Hunan 410073, China

[(a)] yllei@nudt.edu.cn, [(b)] zhun870525@126.com, [(c)] markovyao@163.com
[(d)] zhuzhi@nudt.edu.cn, [(e)] 1247959734@qq.com

ABSTRACT

Combat system effectiveness simulation (CESS) is a special type of complex system simulation. Their models feature multiple disciplines and are rich in domain knowledge. To develop such simulation models, model composability must play a central role where legacy models can be systematically reused and efficiently developed. Domain-friendly modeling is also a requisite for facilitating model development. Traditional modeling methodologies for CESS are either domain-neutral (lack of domain related consideration) or domain-oriented (lack of openness and evolvability) and cannot well fulfill these requirements together. Inspired by the concept of **architecture** in systems and software engineering fields, we extend it into a concept of **model architecture** for complex simulation systems, and propose a model architecture-oriented modeling methodology in which model architecture plays a central role. In this methodology, toward achieving model composability, domain-neutral M&S technologies are used to describe model architecture to improve model evolvability, and domain-specific modeling (DSM) is employed to aid domain experts. Two layers of model architecture, i.e. domain model architecture (DMA) and application model architecture (AMA), are differentiated and explicitly represented in a concrete model architecture-oriented CESS modeling framework.

Keywords: domain-neutral, domain-oriented, architecture, model architecture, model architecture-oriented

1. INTRODUCTION

Combat systems are typical complex systems. Combat effectiveness is one of the most important metrics in acquisition and overall design of various combat systems like aircrafts, warships, submarines, air defense systems, etc. Recently, simulation, instead of theoretical analysis and real experimentation, has become the most prominent approach to evaluate the effectiveness of combat systems (Zimmerman 2014). Combat system effectiveness simulation (CESS) is therefore a special type of complex system simulation of great importance. CESS simulation models are featured with multiple disciplines and intensive knowledge. Developing a useful CESS simulation is obviously non-trivial. The corresponding modeling methodologies should not only support the discipline-specific and user-friendly heterogeneous modeling, but also provide the capabilities to compose a new simulation from existing simulation models so as to both reuse the intensive knowledge within different simulation models, and provide rapid response to the simulation acquisition needs.

Traditional methodologies used in CESS can be roughly divided into two categories. One is domain-neutral and application-specific. It uses some unified modeling and simulation mechanisms with a powerful infrastructure and a model library containing domain-specific model components. The simulation applications are supposed to be assembled in a way of infrastructure plus components. Examples include standardized simulation protocol like HLA (Hemingway et al. 2012; Seo et al. 2014), model specification standard-based like SMP2 (Lei et al. 2009), and universal modeling formalism-based like DEVS (Zeigler, Hall, and Sarjoughian 1999; Seo et al. 2014). The second category can be called domain-oriented by providing a CESS-oriented simulation system, within which different simulation applications can be composed from built-in components and configured with application specific parameters. Prominent examples include EADSim (Azar 2003), SEAS (Trevisani and Sisti 2000; Miller and Honabarger 2006), FLAMES (Niland 2006), etc.

Each category by itself lacks some capabilities significant to CESS modeling. For the domain-neutral methodologies, the domain friendliness is limited since the universal property across different domains is the design focus of various protocols, specifications, and formalisms. In addition, the relationships among different model components are missing from both technologies and components themselves. Therefore, it's not easy to reuse the behavioral patterns formed by several related components. This shortcoming, however, is well treated in domain-oriented methodologies. Each CESS simulation system focuses on one or several

application domains and provides a consistent strong model architecture, which describes possible model components and their relationships in the CESS application domains and provides some extension mechanisms to support new components development and integration. These domain-oriented simulation systems still have some limitations worth mentioning, including: 1) they are developed mostly from a software engineering viewpoint. Each has a software architecture in which the model architecture is embedded. There is no explicit, formal, or platform-independent representation for the model architecture and lack open modeling methods and specifications. Therefore, it is not easy to extend and evolve the model architecture. 2) The behavioral models are mostly black-box implementations. Users have to resort to the model documents, if there are any, for understanding their dynamics. The round-trip between the model documents and implementation codes is too difficult to carry out by simulationists. To modify or extend the model behaviors, which occurs frequently in CESS applications, the users must ask system vendors for help. 3) Each is only applicable to a certain domain, ad hoc in essence, and is not scalable to other domains.

In this research, to tackle the above problems found in CESS practice with traditional modeling methodologies, we propose a model architecture-oriented modeling methodology by employing several technologies provided by model driven engineering paradigm to meet the modeling requirements of CESS. In Section 2, the background and related work are described. In Section 3, the model architecture-oriented CESS modeling methodology and a concrete modeling framework are proposed. In Section 4, conclusions and future work are briefly discussed.

2. BACKGROUND

2.1. Defining CESS

Effectiveness is the ability of a system to accomplish a certain mission. To evaluate the effectiveness of a certain combat system, a CESS needs to construct a mission environment. A typical mission of a combat system is to attack some enemy combat systems or defend some assets from attack by enemy combat systems. A combat system's model usually consists of two kinds of components. The first one is within physical or information domain, including models of platforms, sensors, weapons, measures, communicators, etc. We call these physical models since their functionalities and behavior patterns are relatively stable across applications. The second one is within cognition or social domain, including course-of-actions, formations, situation awareness, combat planning, combat rules, etc. These models are called cognitive models since they are largely determined by the commanders and combatants within the simulation and highly variable in different applications.

A typical experiment based on CESS is like the following. Some performance parameters of the combat system under investigation, e.g., RCS (Radar Cross Section) of the combat platform or range of the sensor on board, are changed while keeping others constant. By monitoring the changes occurring in the combat outcome, the effectiveness of the combat system can be evaluated. The typical purpose is to provide quantitative support for choosing the optimal design alternatives for the combat system in question.

CESS falls to a special and large category of military simulations. The opposing forces on each side are limited to a few. Each side only accomplishes one mission in an experiment so as to make the influence of the combat system on the overall combat outcome more prominent. In this regard, CESS is an engagement-level simulation, compared to the mission-level and above where there are many missions within each side (Sjoberg 2009). In CESS, human is not in the loop. That means the relevant behaviors of combatants, i.e. cognitive behaviors, have to be completely represented in the simulation.

2.2. Modeling requirements of CESS

Models are always crucial to simulation, especially to complex simulations like CESS. It is better to solve the most difficult part of each simulation problem at model level rather than implementation level. From both essential and practical points of view, CESS shows three kinds of important modeling requirements.

1. Model composability.

Model composability is an extensive explored topic in defense simulation area (Davis and Anderson 2004). In CESS domain, there is great benefit to realize model composability. Generally speaking, any combat system can be used to accomplish many missions. To comprehensively evaluate the effectiveness of a combat system, a couple of simulation applications are supposed to be constructed. There are many overlaps across these applications that give rise to lots of opportunities to reuse and compose models. For example, the possible missions for a fighter aircraft include air combat, ground attack, surface attack, air-defense breakthrough, antisubmarine, etc., while a warship is probably meant to fulfill missions like surface combat, air defense, sub defense, and so on. To evaluate their effectiveness, different simulation applications should be created. The effectiveness used to trade off the design alternatives shall come from a synthesis of all possible combat outcomes. The chance to reuse and compose models exists not only among simulation applications created for a certain combat system, but among applications for different combat systems. For instance, a warship model created for surface attack simulation of a fighter aircraft can be reused in an air defense simulation of a warship although the conditions and focus change.

2. Domain specific modeling

CESS as a complex simulation consists of many different kinds of model components. Each may belong

to a distinct domain. To clearly represent the behaviors of each component and their interactions to facilitate understanding and validation of the models by the analysts, different formalisms ought to be exploited to represent behaviors of each model component in a natural way. The modeling methodology should be able to couple them together. Among these, a clear separated modeling between physical domain and cognitive domain is necessary. For the physical domain, the emphasis is mainly on choosing appropriate formalisms for each kind of components and coupling different formalism-based models together semantically. In the cognitive domain, different combat platforms have different cognitive behavior patterns. Furthermore, the modelers would be the analysts or users since the cognitive behaviors change across applications. In this regard, the methodology shall provide combat platform specific and user friendly modeling capabilities.

3. Model evolvability

Since modeling and simulation has been applied into CESS for decades, model evolvability issue becomes more and more significant. In many organizations, including some academic institutes dedicated to applied modeling and simulation research, it has become necessary to both develop and evolve simulation models. Although the concept of model evolvability in general encompasses a lot of connotations, the requirements emerged from CESS gives more focus on the simulation models' potentials to be evolved in a convenient way. That is, the aim is to make simulation models more understandable and modifiable in order to meet the new demands not considered at the outset of a simulation development project. In most cases, CESS simulation models are developed in a document-plus-code way, where documents are used to capture the conceptual and mathematic models; codes are the implementation of these models. The link between conceptual models and codes are often very weak. It is not easy to keep both consistent. When the first inconsistency unavoidably occurs, the models begin to 'die'. Because of this, the simulation models are supposed to be represented abstractly, formally, and platform-independently, and model transformation and code generation should be applied where possible.

2.3. Model driven software and system modeling

Simulation models are in essence a special kind of software. Most of the above CESS modeling requirements can be handled well in a representation level using software-modeling methods. Model-driven engineering (MDE) is a recent outstanding advancement in the software engineering domain, and has been successfully used in systems and simulation modeling fields. By raising the abstraction of the computerized model representation to platform-independent and domain specific level, MDE can contribute much to solving CESS modeling problems. Currently, there are mainly two approaches for realizing MDE. One is model driven architecture (MDA); the other is domain specific modeling (DSM).

1. MDA and SMP

MDA is the OMG (Object Management Group)'s solution for MDE. For modeling, MDA adopts a bottom-up strategy. The UML (Unified Modeling Language) is prescribed as the modeling language for all kinds of platform-independent models (PIMs). MDA also provides several related technologies to support the transformation from PIM to platform-specific model (PSM) and code generation. For domain specific modeling requirements, MDA provides two kinds of metamodeling mechanisms, one is light-weighted UML profile-based metamodeling, and the other is directly metamodeling based on MOF (Meta-Object Facility). In the former, all the modeling capabilities provided by UML can be inherited. In some cases such as domain specific software, this is a big advantage. For other cases like system simulation, however, this may become a downside due to the significant differences between simulation models and common software. For this reason, in system simulation areas, there are cases where the MOF-based metamodeling approach is chosen. Simulation modeling platform (SMP) is one representative instance.

SMP is a simulation model specification standard maintained by European Cooperation for Space Standardization (ECSS) (Sebastiao and Nisio 2008). The main objective of SMP is to provide a platform-independent simulation model specification to facilitate simulation model portability and reuse it across different simulation applications within European Space Agency (ESA) and ECSS. SMP mainly consists of three parts (ECSS 2011). The first one is a simulation model definition language (SMDL), which itself is a metamodel based on MOF. The modeling capability provided by SMDL is actually very similar to the UML's structural modeling features, like class-based, component-based, and interface-based, etc. What has been missed in UML is partly supported by the SMDL, like instance modeling, assembly modeling, and schedule modeling. The rest is supported by the simulation component model (SCM), which is the second part of SMP. SCM views models, simulator, and simulation services all as components. The simulation services include time keeper, event manager, link manager, and many others necessary for simulation modeling which are not included in UML. The interfaces of these components are defined in detail by SCM. The third part of SMP is a C++ mapping specification, which maps the platform-independent SMDL and SCM into C++. The semantics of SMP modeling language is articulated in such a translational way. With SMP, design-based integration and composition of simulation models can be easily fulfilled. The behavioral modeling, however, is not well supported, since it is not within the design objectives. Nonetheless, SMP is very helpful for solving the model composability and model evolvability problems.

2. Domain specific modeling

MDA is in essence technology-oriented, whereas domain-specific modeling (DSM) (Kelly and Tolvanen 2008) is problem-oriented and follows a top-down strategy. The objective is to support the modeler in modeling the problem using domain specific and human-friendly languages. Different from MDA which generally prescribes UML as the 'official' model language for every domain (via profile-based metamodel extension mechanism), for DSM, there is no pre-existent modeling language at the beginning. The languages are designed from scratch for a domain scope. The first step is to analyze the domain requirements and concepts. The second step is to use a certain metamodeling language to describe the domain concepts and their rules to get a domain-specific metamodel, which actually defines a domain-specific modeling language. The third step is to create the visual representation or concrete syntax of the language. The forth step is to define the code generators for model checking, codes, documentation, etc. To make the code generator development easier, it is necessary to provide a domain-specific framework or component library. The model representation based on DSM tends to be a higher-level abstraction. In this way, the user can conveniently specify the models and have a good understanding of the behavior described within the model. DSM is mostly appropriate to be applied in a certain well-defined domain, e.g., within a company or organization.

DSM is actually a special approach to domain engineering. As stated earlier, CESS is a knowledge intensive domain. Following the idea of domain engineering (Harsu 2002), the knowledge in CESS can be divided into domain invariant knowledge (DIK) and application variable knowledge (AVK). DIK refers to the knowledge common to the domain, but not specific to a particular application, whereas AVK is the knowledge specific to one or several applications but not qualified to the domain level. The main interest of applying DSM is in cognitive domain since there is much more AVK in it than in physical domain. In this regard, even the entire physical domain knowledge can be viewed as DIK, which can be pre-implemented separately. Specifically, for a set of physical domain models, there can be many possible combat course-of-actions and command decision choices in cognitive domain. In addition, for different combat systems, there may be different cognitive behavior patterns. Most importantly, the analysts have to create cognitive models themselves since these models are mostly mission related. The CESS modeling methodology should ease the cognitive modeling task as much as possible. DSM does provide a good solution.

3. MODEL ARCHITECTURE-ORIENTED CESS MODELING METHODOLOGY

3.1. Architecture and model architecture

Architecture is a frequently seen term in many domains, such as building, software, and system engineering. There are numerous definitions of architecture (CMU-SEI 2014). Here we adopt the definition provided by ISO/IEC/IEEE for systems and software engineering (ISO/IEC/IEEE 2011). Architecture is the "*Fundamental concepts or properties of a system in its environment embodied in its elements, relationships, and in the principles of its design and evolution*." An architecture is what is fundamental to a system — not necessarily everything about a system, but the essentials.

A complex CESS is a system. Its architecture shall include essential elements like a simulator, simulation services, and simulation models; and essential relationships between simulator and simulation models, and those among different simulation models as schematically shown in the left part of Figure 1. Since the simulator, simulation service, and relationships between simulator and simulation models are largely determined by the M&S technology chosen, the remaining complexity will mainly lie in simulation models and their relationships. Following the definition of architecture of a system, we define model architecture of a simulation system as follows:

Definition 1 Model architecture *of a simulation system is the fundamental concepts or properties of the simulation system in its execution environment embodied in its model components, their relationships, and principles of building or evolving these models.*

For the principles of building or evolving models, we refer to the modeling methods used to support the description and representation of model architecture, including model specifications, formalisms, and languages.

In a sense, model architecture is the kernel of a complex simulation like CESS. Model architecture is also the key to resolve the modeling problems encountered by CESS and other complex simulations. As mentioned in the previous section, for a domain-oriented simulation system, the model architecture can be divided into a Domain Model Architecture (DMA) and many possible Application Model Architectures (AMAs), which is shown in the right part of Figure 1. Each AMA will reuse the DMA via customizing or extending. To achieve maximum reuse, the DMA ought to be designed in an abstract way at the application domain level. What is common to various AMAs is extracted and built into a DMA; a DMA only contains abstract knowledge to be reused in AMAs.

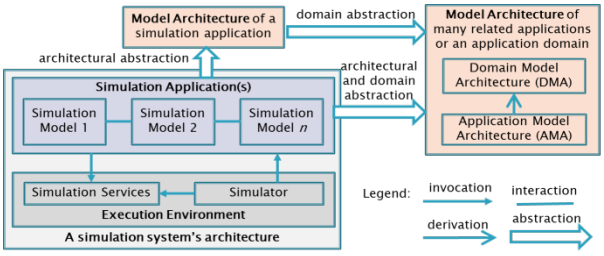


Figure 1 from simulation's architecture to its model architecture

Towards this goal, DMA and AMA can be defined similarly to model architecture:

Definition 2 Domain Model Architecture (DMA) *is with a domain-oriented simulation system. It is the fundamental concepts and properties of the simulation application domain in its execution environment embodied in its model components, their relationships, and principles of building or evolving these models.*

Definition 3 *In a domain-oriented simulation system, an* Application Model Architecture (AMA) *is the fundamental concepts or properties of a simulation application in its execution environment embodied in its model components, their relationships, and principles of building or evolving these models. The model components and their relationships are designed with reuse by either customizing or extending those of DMA. The principles are the same with those of DMA.*

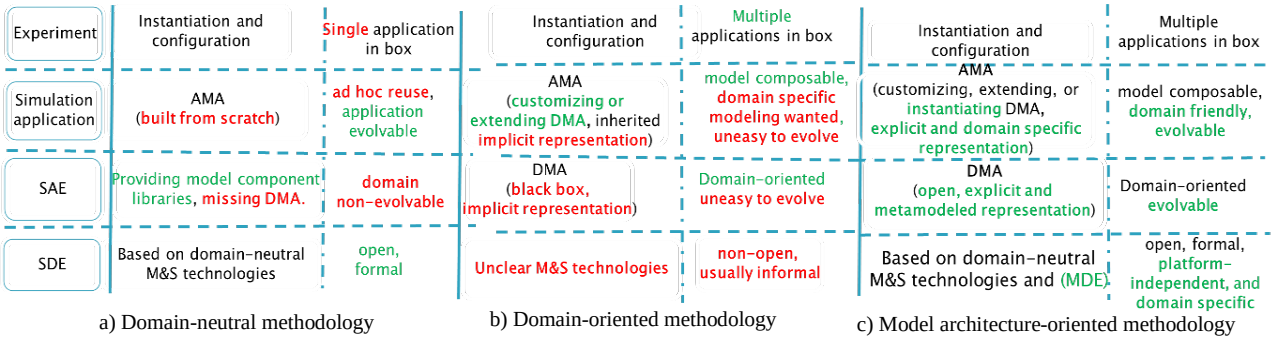
3.2. Toward a model architecture-oriented CESS modeling methodology

3.2.1. Traditional methodologies and their limitations

From the viewpoint of model architecture, traditional CESS modeling methodologies can be structured as shown in Figure 2 (a, b) with four logic layers generic to simulation modeling, including experiment, simulation application, simulation application environment (SAE), and simulation development environment (SDE) layer.

The main characteristic of domain-neutral methodology is its emphasis on certain domain-neutral M&S technologies, either simulation protocol standards, model specification standards, or universal modeling formalisms. Although some powerful simulation development environments provide general or domain-specific model component libraries for reuse, they do not account for DMA. Modelers have to develop one AMA and corresponding model components for each set of application requirements from scratch. Across different applications within a given domain, the AMA developed for one application is unrealistic to be reused in another application for both theoretical reasons, like parsimony principle applied to modeling, and practical reasons, like cost and design complexity. One special advantage of domain-neutral methodology is that the AMA is evolvable since there is an open and standardized representation. However, since there are few domain level considerations in designing an AMA, semantically revising the AMA according to the new application requirements is rather intractable. Domain-neutral methodology is essentially application-specific, and only one certain application can be well supported by the resulting simulation system.

Domain-oriented methodology usually takes composability as the primary objective; the DMA is designed with all the domain modeling requirements in mind. The AMA corresponding to a certain simulation application would mostly be to customize or extend the DMA. In this way, the domain model components and their relationships are reused to a greater extent. As a result, the efforts required for AMA development are greatly reduced. In practice, each specific methodology will be embodied in a powerful simulation system or simulation application environment. Their shortcomings include: 1) the model architectures, especially DMA, are not explicitly represented, which prevents the modelers from extending the model architecture systematically. 2) There is no formal, platform-independent representation of both model architecture and component behavior, which prevents the analysts from completely understanding the models underground and the system from evolving as necessary. 3) There are few domain-specific modeling supports both for physical domain and especially for cognitive domain; the cognitive modeling method provided to users is mostly either configuring rules or handwriting tactical behavior scripts, which lacks either user-friendliness or flexibility.



a) Domain-neutral methodology b) Domain-oriented methodology c) Model architecture-oriented methodology

Figure 2: Traditional CESS modeling methodologies from the viewpoint of model architecture

3.2.2. Model architecture-oriented CESS modeling methodology

From the viewpoint of CESS modeling requirements, domain-oriented methodology is generally more useful than domain-neutral methodology since the most challenging requirement, model composability, could be well supported by DMA. Each specific domain-oriented system simulation is developed mostly from the standpoint of software engineering. Domain-neutral M&S technologies are not given enough prominence as in domain-neutral methodology. This makes achieving model evolvability and composability, but leaves much

to be desired. For domain-specific modeling requirement, both methodologies show little awareness of its importance or lack systematic modeling mechanisms. Taking into account these concerns, we propose a *model architecture-oriented methodology* to overcome those limitations found in traditional ones (see the c part of Figure 2). This methodology basically follows ideas from domain-oriented methodology and incorporates powerful M&S technologies found in domain-neutral methodology and introduces several steps:

1. To center domain-oriented simulation systems on model architecture

Practices tell that simulation model architecture is the bottleneck of CESS and other knowledge intensive complex simulations, and should be oriented in the first place from the domain-level viewpoint. Model architecture is the kernel assets that should be well represented and conveniently evolved. Either viewing CESS from a standpoint of software engineering or thinking CESSs simply being another application of some powerful M&S technology will do few helpful to solve those CESS problems.

2. To divide model architecture into DMA and AMA

For knowledge-intensive simulations like CESS, it is better to have a separation between domain level knowledge and application level knowledge to simplify application modeling. Model architecture is supposed to be divided into clearly separated DMA and AMA. DMA only represents domain level knowledge, i.e., DIK; while AMA only represents application level knowledge, i.e. AVK, by reusing knowledge within the DMA.

3. To incorporate DSM into model architecture

As discussed in the earlier section, DSM is a good means to improve domain friendly CESS modeling in general and cognitive modeling in particular. Since model architecture is broadly defined as those essential to a simulation system, it is rational to view domain specific metamodels as part of DMA and domain specific models as part of AMA. This would extend the relationships between DMA and AMA to instantiation in addition to extension and customization.

4. To model DMA and AMA using domain-neutral M&S and MDE technologies

Modeling DMA and AMA using appropriate modeling formalisms and representing them explicitly with a platform-independent and open model specification will greatly ease modelers semantically understanding the concepts and relationships in model architectures, and facilitate model composability and evolvability.

Compared to the domain-oriented methodology as shown in the part b of Figure 2, model architecture, DMA and AMA in specific, is the focus. Different kinds of M&S technologies are applied to model and represent various pieces of the model architecture where appropriate. DSM is applied to improve application level cognitive modeling friendliness. To support DSM, certain metamodeling technologies should be incorporated in simulation development environment layer.

3.3. Model architecture-oriented CESS modeling framework

Guided by the ideas of the model architecture-oriented methodology, a concrete model architecture-oriented CESS modeling framework used in practice is shown in Figure 3. The modeling task is explicitly divided into two layers: domain modeling and application modeling. Domain modeling is mainly toward pre-implementing the DIK for CESS. The product would be the DMA of CESS, which is essential to a CESS simulation system and common to the CESS domain. Application modeling is going to extend or instantiate the DMA, the product would be the AMA, in which the AVK specific to each application is embodied.

The CESS model architecture is modeled from three viewpoints, i.e., structure, physical behavior, and cognitive behavior. The main requirement for structural architecture modeling is model composability. At the domain level, the DMA defines the fundamental abstract model components common to each CESS application. Most importantly, the structural relationships among these components, including composite, aggregate, interface, and event relations are also deliberately designed and defined in an abstract manner. At the application level, the relationships among different concrete model components are largely determined by the relationships defined in DMA. The possible composite space would be greatly reduced, so as to confine the possible semantic invalidity problem to the lowest level. To accomplish this, some powerful structural modeling formalisms, e.g. object-oriented and component-based, are applied. For syntactic composability, the structural architecture is diagrammatically described by a couple of UML class diagrams and formally represented using SMP in a platform-independent way. A UML profile for SMP makes the SMP representation transformed from the UML class diagrams automatically. The SMP representation can be generated into C++ representation automatically.

Physical behavior modeling describes the behaviors within the physical domain for each concept defined in the structural architecture and the dynamic relationships among these concepts. As for the modeling methods, different kinds of behavior modeling formalisms can be employed according to their specificity, including Statecharts, discrete event, activity diagram, sequential diagram, design patterns, etc. Since the model components defined in the structural architecture are at either an abstract domain level or a concrete application level, the physical domain behaviors of these model components are divided and described in DMA and AMA levels respectively. In fact, physical domain behaviors are relatively less variable compared to cognitive ones, most of the physical domain behaviors

can be implemented into DMA. Consequently, at the application level, the main requirements for physical behavior architecture modeling are in extending or modifying the physical behaviors defined in DMA in a user friendly manner. This can be achieved by behavior inheritance and override capabilities provided by UML, and by relevant code generators specific to the behavior diagrams used. For UML behavior diagrams, UML profile is employed again to add domain-specific extensions and make the code generators more specifically designed. The target physical behavior model representation language is C++, too.

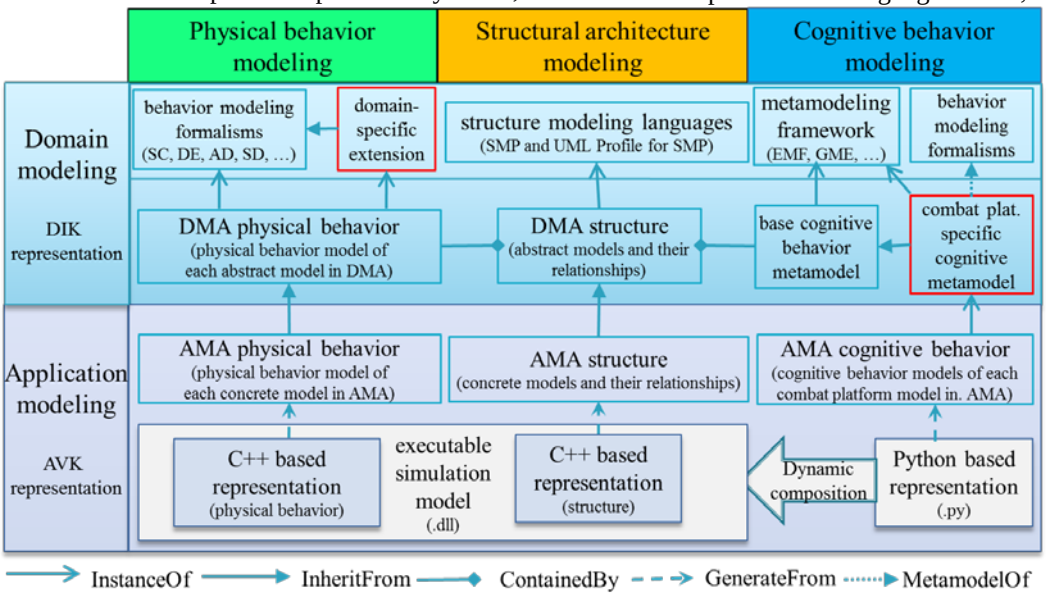


Figure 3: Model architecture-oriented CESS modeling framework

The third aspect is the cognitive behavior modeling. Both physical behaviors and cognitive behavior are behaviors performed by the model architecture concepts or objects. However, there are fundamental differences between them. The physical behaviors are largely determined by the natural laws and constraints, while the cognitive behaviors are mainly up to the free will of human beings, which means that the latter are highly variable across applications. Consequently, there are relatively few common behaviors at the domain level. What can be done at the domain level is to define a cognitive behavior modeling language for each combat platform type based on some behavior modeling formalisms with combat platform specific extensions. For instance, a fighter aircraft behavior modeling language can be designed based on Statecharts; a warship behavior modeling language can be designed based on Petri Net; etc. As abovementioned, DSM method and supportive metamodeling frameworks (e.g. EMF and GME), are employed to realize these languages. Base cognitive behavior metamodel is the common metamodel representing fundamental modeling elements and mechanisms found in various combat platform specific behavior metamodels. In fact, it refines the DMA, and is responsible to participate in the composition relationship with structural DMA. At the application level, distinctly different from in physical domain modeling, the analysts create cognitive behavior models with these combat platform specific cognitive modeling languages. In other words, the pertinent cognitive behavior metamodel is instantiated in the application modeling layer. Each combat platform specific cognitive model representation will be transformed into a Python-based representation, which is dynamically composed with the model architecture.

4. CONCLUSION AND FUTURE RESEARCH

Combat system effectiveness simulation (CESS) is one special and significant type of system simulation. The challenge confronted by CESS is both functional such as structure and behavior abstraction and non-functional such as composability, multi-domain specificity, and evolvability. The non-functional one is more difficult to overcome due to it being comprehensive, fundamental, and not directly related with a project's aim. Whatever problems would be, solving them in an abstract modeling level is almost a must. Traditional domain-neutral and domain-oriented modeling methodologies all suffer from some limitations toward the non-functional requirements of CESS. There should be a methodological shift to solve the CESS problems. The main contribution of this research is that a model architecture-oriented modeling methodology for CESS is proposed. The crucial role of model architecture is recognized and developed. By incorporating domain-neutral M&S technologies and domain-specific modeling methods into the representation of the concrete CESS model architecture, the problems encountered in traditional methodologies-based CESS practice (i.e. limitations on composability, domain-specificity, and evolvability) can be mitigated to a satisfactory extent. After more than five years research and development, most of the CESS model architecture (especially DMA) and the modeling

framework have been realized (Lei et al. 2013; Li et al. 2013). As a result, a CESS-oriented simulation system named WESS has been implemented and applied to non-trivial CESS applications.

In the next step, the CESS model architecture will evolve according to the continuous application requirements within the CESS scope. For physical behavior models described using UML diagrams, there needs to be relevant code generation tools constrained by the structural architecture to make these behaviors fully formal and evolvable. For cognitive behavior models, DSM is a rather new approach. Even though we have done some exploratory work toward this direction, there should be much more work to make it practical. Among them, one is to provide each combat platform type a domain specific modeling language and modeling tools, for instance, an EMF/GMF/OCL/Acceleo-based aircraft combating cognitive modeling language has been prototyped; the other is to borrow pertinent modeling concepts from some cognitive architectures like Soar (Zacharias, Millan, and Hemel 2008), to improve cognitive behavior modeling capabilities for more complex cognitive behaviors.

ACKNOWLEDGMENTS

The study was supported in part by the National Natural Science Foundation of China through Grant Number 61273198.

REFERENCES

Azar, M. C. 2003. “Assessing the Treatment of Airborne Tactical High Energy Lasers in Combat Simulations”. MS Thesis: Air Force Institute of Technology.

CMU-SEI. 2014. http://www.sei.cmu.edu/architecture/start/glossary/definition-form.cfm, last assessed 2014.

Davis, P.K. and R.H. Anderson. 2004. “Improving the Composability of DoD Models and Simulations”, *Journal of Defense Modeling and Simulation: Applications, Methodology, Technology*. 1:5-17.

European Cooperation for Space Standardization. 2011. “Simulation modelling platform -Volume 1: Principles and requirements”, ECSS-E-TM-40-07 Volume 1A.

Hall, S. B., B. P. Zeigler and H. S. Sarjoughian. 1999. “JointMEASURE: Distributed Simulation Issues In a Mission Effectiveness Analytic Simulator”, In *Proceedings of the 1999 Simulation Interoperability Workshop (SIW)*. 99F-SIW-159.

Harsu, M. 2002. “A Survey on Domain Engineering. Institute of Software Systems”, Tampere University of Technology. ISBN 9789521509322.

Hemingway, G., H. Neema, et al. 2012. “Rapid synthesis of high-level architecture-based heterogeneous simulation: a model-based integration approach”, *Simulation: Transactions of the Society for Modeling and Simulation International* 88:217-232.

ISO/IEC/IEEE - 42010-2011. 2011. “Systems and software engineering -- Architecture description”, http://standards.ieee.org/findstds/standard/42010-2011.html, last assessed 2014.

Kelly, S. and J-P. Tolvanen. 2008. “Domain-Specific Modeling: Enabling Full Code Generation”, Wiley-IEEE Society Press.

Kim, J. H., Moon I. and T. G. Kim. 2012. “New insight into doctrine via simulation interoperation of heterogeneous levels of models in battle experimentation”, *Simulation: Transactions of the Society for Modeling and Simulation International* 88:649–667.

Lei, Y., W. Zhang, X. Zhao, et al. 2009. “Research of SMP2-based Missile Countermine Simulation System”, *Journal of System Simulation* 14:4312-4316.

Lei, Y., Q. Li, F. Yang, et al. 2013. “A composable modeling framework for weapon systems effectiveness simulation”, *Systems Engineering-Theory & Practice* 33:2954-2966.

Li, X, Y. Lei, et al. 2013. “Domain-specific decision modelling and statistical analysis for combat system effectiveness simulation”. *Journal of Statistical Computation and Simulation* 2013:1-19.

Miller, J.O. and L. B. Honabarger. 2006. “Modeling and Measuring Network Centric Warfare (NCW) With the System Effectiveness Analysis Simulation (SEAS)”, In *Proceedings of the 11th ICCRTS*.

Niland, W. M. 2006. “The Migration of a Collaborative UAV TestBed into The FLAMES Simulation Environment”, *In Proceedings of the 2006 Winter Simulation Conference* edited by L. F. Perrone, F. P. Wieland, J. Liu, B. G. Lawson, D. M. Nicol, and R. M. Fujimoto, 1266-1272.

Sebastiao, N. and N. Di Nisio. 2008. “E-40-07 A New Standard for Simulation Model Portability and its implementation in SIMULUS”, http://sunset.usc.edu/GSAW/gsaw2008/s3/dinisio.pdf

Sjoberg, B. 2009. “EW M&S from Engineering to Campaign”, In Proceedings of the *AOC EW Modeling and Simulation Conference*.

Seo, K. M., C. Choi, T. G. Kim and J. H. Kim. 2014. “DEVS-based combat modeling for engagement-level simulation”, *Simulation: Transactions of the Society for Modeling and Simulation International* 90: 759-781.

Tolk, A. 2012. “Engineering Principles of Combat Modeling and Distributed Simulation”, Wiley-IEEE Society Press.

Trevisani, D. A. and A. F. Sisti. 2000. “Air Force hierarchy of models: a look inside the great pyramid”, In *Proceedings of SPIE 4026, Enabling Technology for Simulation Science IV*, 150 doi:10.1117/12.389368.

Zacharias, G. L., J. M. Millan, and S. B. V. Hemel. 2008. “Behavioral Modeling and Simulation: From Individuals to Societies”, National Academies Press.

Zeigler, B. P., S. B. Hall, and H. S. Sarjoughian. 1999. "Exploiting HLA and DEVS To Promote Interoperability and Reuse in Lockheed's Corporate Environment", *Simulation: Transactions of the Society for Modeling and Simulation International* 73: 288-295.

Zimmerman, P. 2014. "DoD Modeling and Simulation Support to Acquisition". NDIA Modeling & Simulation Committee. http://www.ndia.org/Divisions/Divisions/Systems Engineering/Documents/ NDIA-SE-MS_2014-02-11_Zimmerman.pdf

AUTHORS BIOGRAPHY

YONGLIN LEI is an Associate Professor of Simulation Engineering Institute at National University of Defense Technology, Changsha, China. His research interests include complex system simulation, model composability, model driven architecture, domain specific modeling, and their applications in defense simulations. He holds a Ph.D. degree in System Engineering from National University of Defense Technology. His email address is yllei@nudt.edu.cn.

NING ZHU is a Ph.D. student of Simulation Engineering Institute at National University of Defense Technology, Changsha, China. He holds a M.S. in system engineering from National University of Defense Technology. His research interests include model driven engineering, combating system simulation, and architecture driven simulation. His e-mail address is zhun870525@126.com.

JIAN YAO is a Ph.D. student of Simulation Engineering Institute at National University of Defense Technology, Changsha, China. He holds a M.S. in system engineering from National University of Defense Technology. His research interests include human behavior representation, model driven engineering, and combating system simulation. His e-mail address is markovyao@163.com.

ZHI ZHU is a Ph.D. student of Simulation Engineering Institute at National University of Defense Technology, Changsha, China. He holds a M.S. in system engineering from National University of Defense Technology. His research interests include domain specific modeling and combating system simulation. His e-mail address is zhuzhi@nudt.edu.cn.

SHUAI CHEN is a graduate student of Simulation Engineering Institute at National University of Defense Technology, Changsha, China. His research interests include composable modeling and combating system simulation. His e-mail address is 1247959734@qq.com.