A DEVS-based M&S Method for Large-scale Multi-agent Systems

Mingxin Zhang[1,2], Mamadou Seck[1] and Alexander Verbraeck[1]
[1] System Engineering Section, Delft University of Technology, 2600 GA Delft, The Netherlands
[2]System Simulation Laboratory, National University of Defense Technology, Changsha, China
{m.zhang-1, M.D.Seck, a.verbraeck}@tudelft.nl

Keywords: cognitive modelling, agent organization, DEVS-based framework, L-systems

Abstract:

ABMS offers various simulation systems, tools, toolkits and languages for multi-agent system research. However, there is a need for a M&S method for L-systems research as current ABMS method has some degree of difficulty in dealing with the scale and heterogeneity issues of L-systems. This paper focused on the modelling aspect of the method by combining cognitive modelling, agent organization theory and DEVS-based framework together. First of all, we explained our research initiative by giving the reasons of our research. Further literature review of our choice is also present. Then, we present a design for constructing a DEVS-based system model. We choose PRS as our preferred cognitive architecture, and constructed a DEVS-based simulation framework together with the guidance of agent organization theory. Finally we summarize the benefits of our research compared to other methods.

1 INTRODUCTION

Agent-based modelling and simulation(ABMS) offers a relatively new method for multi-agent system research[1], and there are various ABMS simulation systems, tools, toolkits and languages(e.g. swarm, Repast and NetLogo) in their respective research areas. However, there is a particular kind of multi-agent system, defined as L-systems(large-scale multi-agent systems), which is difficult to be studied using ABMS method. Typical L-systems are such as highway system, crowd evacuation, stock market and migratory birds. The problems causing the difficulty mainly come from: 1) the system scale. ABMS can only support simulation of relatively small scale multi-agent system(usually the number of agents is less than 10 thousand). When a L-system scales to a larger number, it often outstrips the processing capability of a ABMS tool or toolkit[2][3][4]; 2) the system heterogeneity. A L-system can consist of diverse sub-systems which result diverse behaviours. General ABMS tool or toolkit cannot support all the heterogeneous sub-systems interoperate in one architecture.

Our research objective is trying to solve the problems by designing a new method. In this paper, we mainly focused on the modelling aspect of the method by combining cognitive modelling, agent organization theory and DEVS-based framework together. In general, we are making efforts on two aspects. One is developing cognitive architecture based intelligent high-fidelity agents which can reason and act as teammates or competitors in complex non-deterministic environment using DEVS framework, another is constructing a DEVS-based simulation model which clearly specifies the agent independencies.

Why cognitive modelling? Most of ABMS tools and toolkits, adopt similar reactive architectures for individual modelling, which are suitable for simulating simple agents based on rules, like ant foraging model and simple stock market model. But in some cases of L-systems(e.g. highway system, and crowd evacuation), individual agent behaviours, which include BDI(belief-desire-intention), emotions, personality and personal values, are required for individual modelling, and most of ABMS methods cannot support modelling rich cognitive agents. Moreover, a multi-level reasoning ability is also required for individual modelling to solve cognitive problems in different layers sometimes.

Why DEVS-based framework? We believe that ABMS has difficulty in dealing with system scalability and interoperability due to the lack of a formal operational specification to formalize the model behaviour, model structure, and support multi-resolution modeling, hierarchical modeling and model reuse[5].

In PADS(parallel and distributed simulation) community, DEVS is widely spread as a modeling specification as it supports hierarchical, modular model representation. It also supports valid simplification, abstraction, and aggregation. Furthermore, extensions of the DEVS framework have been developed to handle variable structure, probabilistic, cellular, and logic-based representations[6]. Moreover, the DEVS modeling framework promises to be a sound basis for distributed simulation. Numerous techniques have emerged such as optimistic synchronization, dynamic load balancing, and Global Virtual Time to improve the simulation performance. However, the most significant advantage afforded by the DEVS specification is that it can support the transformation of other modeling specification[7]. For the above reasons, system scalability and interoperability is guaranteed in PADS community.

Why agent organization theory? In DEVS-based models, dependencies of agents is specified as a topology, such as a spatial grid or network of nodes (agents) and links (relationships). This topology describes who transfers information to whom. As large-scale multi-agent systems grow to include millions of agents, this traditional topology like network of agents is insufficient to describe the agents dependencies. However, in ABM(agent-based modelling) community, agent organization theory is proposed to specify how agents are organized and change the connection relation dynamically, and how agents interact with each other[8][9][10], which can be a guidance for modelling L-systems.

As a matter of fact, our research group is trying to combine cognitive modelling, agent organization theory and DEVS-based framework together. Literature review of our choice is present as follows.

2 LITERATURE REVIEW

2.1 Cognitive modelling

As we stated above, most of ABMS methods cannot guarantee the richness and flexibility of individual modelling, while both of the requirements can be satisfied by cognitive modelling in cognition science community.

In Cognitive science, cognitive model is proposed to help understanding and explaining the processes(e.g. perceiving, learning, remembering, planning) that the brain, especially the human brain, uses to accomplish complex tasks. Moreover, cognitive model can be used to derive new predictions for new relationships that go far beyond the original data[11]. Thus, typical applications of cognitive modeling are beginning to spill over into various fields including clinical psychology, cognitive neuroscience, agent based modeling in economics, and many more.

However, cognitive models describe human information processing at an abstract and mathematical level of analysis, for example, BDI[12]. Thus, a more computational theory, which is cognitive architecture, is proposed to focus on the structural properties of the modelled system, and help constrain the development of cognitive models within the architecture. Typical cognitive architectures include SOAR, ACT-R, and PRS[13], to name a few.

In addition to these primarily monolithic agent architectures, multi-level/layer cognitive architectures (e.g. TouringMachines, Atlantis, InteRRap, and Sloman’s theory[14]) are proposed to provide a more flexible mechanism to solve cognitive problems in different layers.

All of these architectures provide a computational level of analysis that makes it computationally feasible to derive precise predictions for complex tasks. However, there are not many works on implementing cognitive architectures for multi-agent simulation in current research[15]. Sun[16] implemented a cognitive architecture CLARION for social simulation. Mittal[17] described how ACT-R architecture can be decomposed and formalized using the DEVS formalism. Akplogan[18] implemented a formal BDI model to simulate an agent in agriculture using DEVS framework, while his research focuses on a single agent solving tasks in specified area, not for generative multi-agent system development.

Their works inspired our research. However, we believe there are several unique aspects of our method that contribute a novel design.

2.2 DEVS-based framework

To solve the scalability and interoperability problem, researchers on ABMS tried a lot of efforts. One dominate attempt is using distributed simulation architectures. Minson proved that speedup can be achieved through the integration of the Java-based lightweight agent-simulation toolkit RePast with HLA[4]. However, the performance of HLA-based architecture is limited for many factors, and one of major factors is the number of federates[19].

As a matter of fact, we believe that ABMS has difficulty in dealing with system scalability and heterogeneity due to the lack of a formal operational specification. In ABM community, to formulize the system dynamics in multi-agent system, temporal modeling specification languages have been introduced[20] in which dynamic properties are often specified in the form of a set of logical formulae. The advantage of this method is the declarative modelling of simulation models, for examples, Executable Temporal Logic[21] and the Strictly Declarative Modelling Language

SDML[22]. However, simulation of dynamics is the main purpose of this specification and it usually does not provide explicitly specified organizational structure or offer dedicated support for a specific type[23].

Another specification dealing with both structure specification and behaviour specification is the Agent/Group/Role(AGR) organization modelling approach[10], in which an organization structure consists of a set of groups, roles in each group and agents fulfilling roles. However, this usual agent-based specification is also not suitable for dealing with system scalability as they don’t account for structure change.

To guarantee system scalability and interoperability, DEVS(Discrete Event System Specification), as a modular and hierarchical formalism for modeling and analysing general systems, was first proposed by Zeigler in 1984. DEVS is easy to deal with system scale as it supports hierarchical, modular model representation, which is a sound basis for distributed simulation. Moreover, system interoperability is guaranteed as the DEVS specification can support the transformation of other modeling specification[7].

Based on DEVS formalism, a variety of M&S frameworks and environments are developed(e.g. ADEVS, PCD++, DEVSJAVA, DEVS-Suite, JAMES[24], VLE, and Theatre). Among these, some are designed for large-scale systems based on different mechanisms. Hu[25] proposed a simulation engine oneDCoord implemented in DEVSJAVA for large-scale cellular DEVS models. Liu[26] proposed a protocol called as Lightweight Time Warp and realized in PCD++ for Large-Scale DEVS and Cell-DEVS Models. However, one of the most successful frameworks is JAMES.

JAMES[24] is a Java-Based agent modeling framework which origins from PDEVS for the parallelization of multi-agent systems. In JAMES, DSDEVS and M-DEVS as DEVS extensions are also realized, by which mechanisms like mobile agents are used in order to manage the structural changes of multi-agent systems.

Just like ABMS, most of the models realized in DEVS-based framework are simple without rich cognition. Take JAMES for example, JAMES considers an agent as an atomic model when modelling multi-agent systems. The autonomy of the agent is realized through internal transition function of DEVS atomic model, whereas perception and action of the agent are realized through external transition function and output function.

2.3 Agent organization theory

The DEVS formalism has well-defined mechanisms supporting scalability and interoperability of DEVS-based modeling and simulation systems, while it doesn’t clearly specify the model dependencies, which are essential for modelling large-scale multi-agent systems.

In ABM community, agent dependencies are modelled as agent organizations which can be seen as a set of agents regulated by rules and mechanisms of order with which autonomous agents can achieve common goals under an institutional control. In agent organizations, roles are played by agents and goals are achieved through communication of agents.

In recent years, various organizations are modelled by the researchers, such as institution, group, firm, and community. A hot research topic in modeling organizations is the modeling and simulation of organizational structure since organizational structure plays a critical role in the development of agent-based modeling[27]. The organizational structure usually involves two fundamental concepts: agent roles and their relations in terms of which the overall behavior of the multi-agent system is determined[8]. When modeling agent organizations, several organizational styles are introduced by Kolp[9]. Grossi[8] defined a formal relation between institutions and organizational structures.

3 A PRELIMINARY DESIGN

To realize our idea that combines cognitive modelling, agent organization theory and DEVS-based framework, we made a preliminary design for cognitive individual modelling and system construction. In detail, at first we identify a clear separation of concerns between various components of PRS cognitive architecture; then we illustrated how this architecture is decomposed and ultimately formalized in DEVS formalism; at last we give out how system model is constructed.

3.1 Individual agent model design

In this design, we adopted procedural reasoning system (PRS)architecture as the guidance. The choice of PRS is not the point as we treat cognitive individual modelling in a separate conceptual modelling layer, by which design any cognitive architecture can be adopted. A PRS is a typical framework for constructing real-time reasoning agents based on BDI paradigm that can perform complex tasks in non-deterministic environments. According to the PRS agent architecture by Georgeff and Ingrand[28], we constructed a DEVS-based PRS agent model framework, which is present below

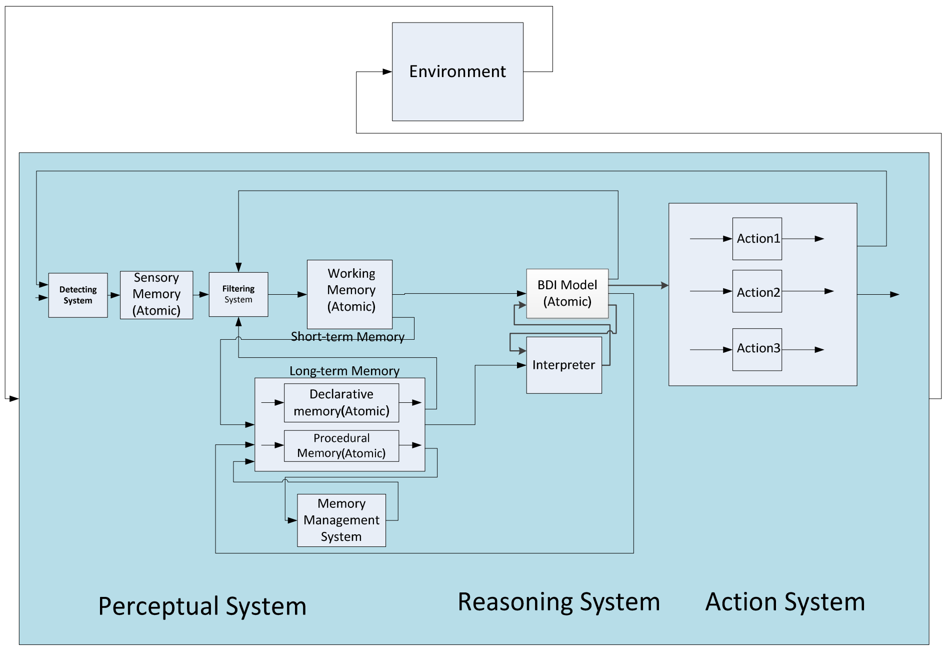

Figure 1 DEVS-based PRS Agent Model framework

This DEVS-based PRS agent model is constituted by three main parts: (1)perceptual system, (2) reasoning system, and (3) action system. Each part of the framework consists of a set of interconnected sub-systems and modules.

Perceptual System is modelled as an information processing model[29] to form belief which is based on the idea that humans process the information they receive, rather than merely responding to stimuli. In this model, the mind's machinery includes sensory memory with a great capacity for bringing information in, working memory(also called short-term memory with a capacity of 7 ± 2) for actively manipulating information, and long term memory with unlimited capacity for passively holding information so that it can be used in the future.

The sensory memory has a detecting system(sensory receptor), which receives and holds all external and internal stimuli based on attention mechanisms. The sensory memory is modelled as an atomic model which determines whether the input should be brought into the working memory, or discarded. The input ports of sensory atomic model are modelled as sensory organs (biological or artificial) used to capture information.

The working memory is also modelled as an atomic model, where information from long-term memory and the sensory memory is combined to form belief which help solve problems. However, the working memory has a small capacity which limits the abilities of agents to solve problems. Therefore, a filtering system is modelled to be a secondary mechanism to determine what information would be useful for problem solving. This mechanism can be modified after reasoning process.

Long-term memory is modelled as a coupled DEVS model which manages knowledge of all an agent knows. Long-term memory can be classified as declarative memory and procedural memory. Declarative memory is modelled as an atomic model which manages factual information that can be retrieved and acted upon. Procedural memory is also modelled as an atomic model which manages the steps of central cognitive processing. The items stored in long-term memory are organized and managed by memory management system.

Reasoning system is modelled as an atomic model and an interpreter based on BDI paradigm. In this model, beliefs representing what the agent believes about itself, other agents and the environment are modelled as belief sets, which are updated by working memory. Desires representing the motivational state of agent are modelled as a run-time stack. KAs including a set of plans are stored in long-term memory. They lie dormant until they are called back into the working memory and thus put to use. Intention structure is modelled as a run-time stack of hierarchically related KAs and maintained by the interpreter.

We modelled an interpreter to evaluate the optional KAs, which selects the most appropriate

KAs based on system beliefs and goals, places selected KAs in the intention structure, selects a task from the root of the intention structure and finally executes one step of that task. The most difficult part of the process is how to evaluate the KAs and select the most appropriate ones. It is worth noting that we are inspired for the precondition and evaluation of a KA by Dennett's three levels of abstraction[30]. However, the process of interpreting can be achieved by introducing Jason[31] interpreter. Related works are such as MADeM[32] and MOISE+[33].

Action system is the execution part of an agent. As action is modelled as a message in DEVS-based PRS agent model, action system can be modelled as a message distribution system.

An action can be either a primitive action which is a behaviour or activity that can be executed directly, or a new goal, or a new belief.

3.2 Design of formalization of components

As stated before, the DEVS-based cognitive agent model consists of a set of interconnected sub-systems and modules. Besides these, there are other supporting data structure which construct the cognitive agent model together. All the components of the agent model are formalized and instructed as below in the table using parallel DEVS (PDEVS)[34] extension of DEVS formalism.

Table 1 Formal description of components

Module	Formal Description	Model type
Sensory Memory Model	$M_S =< X_S, Y_S, S_S, \delta_{\text{int}_S}, \delta_{ext_S}, \delta_{con_S}, \lambda_S, ta_S >$	DEVS atomic model
Working Memory Model	$M_w =< X_w, Y_w, S_w, \delta_{\text{int}_w}, \delta_{ext_w}, \delta_{con_w}, \lambda_w, ta_w >$	DEVS atomic model
Long-term Memory Model	$DN_L =< \text{D}, \ \{M_D, M_P\}, \ \{I_i\}, \ \{Z_{i,j}\} >$	DEVS coupled model
Declarative Memory Model	$M_D =< X_D, Y_D, S_D, \delta_{\text{int}_D}, \delta_{ext_D}, \delta_{con_D}, \lambda_D, ta_D >$	DEVS atomic model
Procedural Memory Model	$M_P =< X_P, Y_P, S_P, \delta_{\text{int}_P}, \delta_{ext_P}, \delta_{con_P}, \lambda_P, ta_P >$	DEVS atomic model
Declarative Memory item	$M =< Memory\_type, Memory\_name, parameter1 \ldots parameter >$	Data structure
Procedural Memory item	Executable programs or codes	
Memory	$MS = \{(M_1, M_2 \ldots M_n) \mid M_i \in M, i \in [1, n]\}$	A list of Memory items
Beliefs	$BS = \{(B_1, B_2 \ldots B_n) \mid B_i \in B, i \in [1, n]\}$	A list of Beliefs
Belief	$B =< Time, Source, BeliefType, Content, Possibility >$	Data structure
Desire	$D =< (g_1, g_2 \ldots g_n) \mid g \in G >$	A run-time stack of goals
Goal	$G =< Goal\_type, Goal\_name, parameter1 \ldots parameter >$	Data structure
Intention	A run-time stack of hierarchically related KAs	
KAs	$KAs = \{(KA_1, KA_2 \ldots KA_n) \mid KA_i \in KA, i \in [1, n]\}$	A list of KAs
KA(procedure)	$KA =< ID, Time, PCondition, Actions, Goal, effect >$	Data structure
Action	$< MSG >$	An event
MSG	$MSG =< ID, Source, Destination, Type, Content, Time >$	Input/output Message

As shown in the table above, knowledge in memory is represented as a list of memory items. For a declarative memory item, a data structure containing several components is adopted. However, procedural memory is stores as executable programs or codes as procedural memory involves the method of how to perform tasks. Desire is a consistent set of goals, and each goal is generated instantaneously or functionally. The last goal in the desire is the top-level goal which is persistent and initially given to the agent. The goal type is one of: Achieve, Perform, and Maintain. Intention is a run-time stack of hierarchically related KAs. A KA is a data structure. Each KA consists of a component($Actions$) which describes the steps of the procedure and an invocation condition($PCondition$) which specifies under what situations the KA is useful. A KA also has components($Goal$ and $effect$) to express the results and utility of performing certain sequences of actions under certain conditions. Actions in DEVS-based cognitive model are modelled as messages(events).

3.3 Design of system model construction

We adopted DEVS framework to reconstruct the system model. A DEVS-based system model for a L-system is typically as shown in Figure 2.

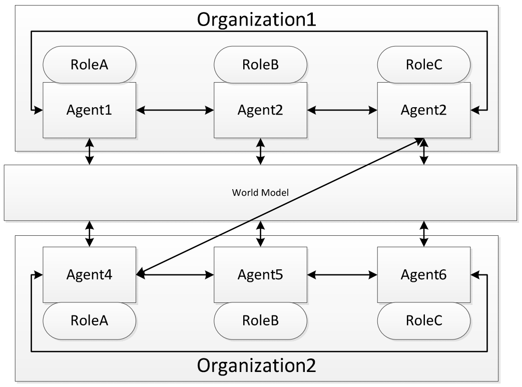

Figure 2 System model design

In general, a system model includes agent organizations and a world model. In each agent organization, each agent plays a certain role and is connected to other agents for communication and coordination. However, agents in different organizations can also communicate directly or through the communication with world model.

4 A SIMULATION IMPLEMENTATION

4.1 Simulation scenario

To test the above design, we constructed a simulation system. Our hypothesis is that there are two competitive teams of agents pursuing a common goal, which is to find a same destination in the cell-based environment, and the environment is full of barriers which could cause difficulties on rooting for agents. In each team, different agent organizations and strategies are modelled, that is, agents have different roles, and communicate directly or through accessing the local cell information. In individual modelling level, agents in each team have different individual planning abilities.

Moreover, two teams of agents are modelled as two organizations. Each organization has a unique strategy to achieve the goal. In the agent organization of our system, there are some agent roles, such as Explorer, Coordinator, Information sharer, Leader and so on. All of these roles are played by agents and they try to communicate and achieve the goal under an institutional control.

In this initial simulation test scenario, the scale of system is relatively small, as we are focusing on individual modelling and system construction. Further research will be conducted on large-scale systems by adopting parallel and distributed simulation technologies.

4.2 System model execution

The underlying simulator of DEVS-based PRS agent model and the experimental frame is DSOL[35], which is a simulation environment that supports continuous and discrete-event simulation execution and experiment. The Event-Scheduling DEVS (ESDEVS) library[36], implements the parallel DEVS formalism on top of the DSOL library. On the whole, the library specifies the meta structure of atomic and coupled DEVS models, and handles the couplings, output function, transition functions at a high level, so that ESDEVS (together with DSOL) serves as a DEVS simulator and experimental frame.

The interface of the simulation system when running is shown as follows.

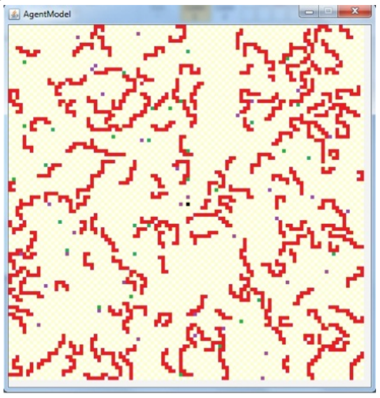

Figure 3 Simulation interface

5 DISSCUSSION

In this paper, we combined cognitive modelling, agent organization theory and DEVS-based framework in order to design a new method for M&S of L-systems. Benefits can be gained for this method.

5.1 Benefit of cognitive modelling

In psychology and cognitive science, cognition usually refers to an information processing view of an individual's psychological functions. In our design of DEVS-based PRS agent model, overall processing activity consists of a mixture of parallel and serial processing in and across the modules. In this way, basic cognitive processes(e.g., Perception, Attention, Recall, Encoding and Storing) can be modelled through parallel activities occurring in and across the modules. Besides these, reasoning, as an advanced process inside reasoning system conducted by interpreter, can also be modelled.

Sloman[14] introduced a three-layer architecture of reasoning. In our research, we modelled three mechanisms in BDI model to realize Sloman’s three-layer reasoning process for different context. A reactive reasoning mechanism is used to handle routine new information by a set of reactive procedures. A deliberative reasoning mechanism is easy to realize for tasks involve achieving new types of goals or acting in novel contexts, as our agent model is based on a PRS architecture. And when dealing with conflicting goals, or when to decide whether to change the criteria being used by the planner, a preliminary reflective reasoning mechanism is realized by meta-management of the beliefs, goals, and intentions of PRS itself. The meta-management information is stored in a meta-level KA(procedure).

With the help of reflective reasoning mechanism, a variety of advanced processes can be modelled. For example, inculcation of ethical from the agent organization, self-assessment and self-learning.

5.2 Benefit of DEVS-based framework

Most currently available cognitive architecture implementations are designed to perform complex tasks in non-deterministic environments, for example, fault detection. Therefore, they are developed using a specified language on a specified platform, which leads to poor extensibility and scalability. However, the DEVS-based cognitive agent model has inherent benefits such as extensibility, scalability and interoperability which are derived from DEVS formalism.

Another major gain from the DEVS-based cognitive agent model is its component-based design. For example, as long-term memory has a huge storage and higher access rates than other components, it may require more computing resources. As it’s rather difficult for parallelism in original cognitive model, its performance is damaged. However, as DEVS-based cognitive agent model is component-based, Declarative/Procedural Memory model can be executed on high-computing resources, like GPU.

Compared to agents in most traditional ABMS systems, DEVS-based cognitive agent model benefits not only on the rich cognition, but also on the system scalability. As this model is based on DEVS formalism, it’s much easier to construct a L-system simulation and deal with scale issue.

5.3 Benefit of agent organization theory

In PADS(parallel and distributed simulation) community, most of the theories or formalisms do not support modelling agent independencies, which indirectly results in difficulty of partitioning and load balancing of large-scale simulation system.

With the help of agent organization theory, agent independencies in a L-system are clearly defined, with which agents are organized under an institutional control. Moreover, agents can change the connection relation dynamically.

6 CONCLUSION

In this paper, we combined cognitive modelling, agent organization theory and DEVS-based framework together in order to realize a new M&S method for L-systems. In general, our research is novel in terms of both agent representations and agent structure all abstracted by DEVS formalism to meet the needs of L-systems simulation construction. As the agent model interface remains the same and the intelligence and dynamics are hidden inside the model, the individual agent model can be considered as a normal model in the DEVS model hierarchy which improves the system scalability and interoperability. Further research will focus on the simulation aspect of M&S for L-systems.

REFERENCES

[1] C. M. Macal and M. J. North, “Tutorial on agent-based modelling and simulation,” Journal of Simulation, vol. 4, no. 3, pp. 151–162, 2010.

[2] J. Anderson, “A generic distributed simulation system for intelligent agent design and evaluation,” in Proceedings of the Tenth Conference on AI, Simulation and Planning, 2000, Society for Computer Simulation International, pp. 36–44.

[3] L. Gasser and K. Kakugawa, “MACE3J: fast flexible distributed simulation of large, large-grain multi-agent systems,” in Proceeding of The First International Joint Conference on Autonomous Agents & Multiagent Systems, 2002, ACM, pp. 745-752.

[4] R. Minson and G. Theodoropoulos, “Distributing RePast agent-based simulations with HLA,” in Proceedings of the 2004 European Simulation Interoperability Workshop, 2004, John Wiley and Sons Ltd, pp. 1225–1256.

[5] J. P. Müller, “Towards a formal semantics of event-based multi-agent simulations,” Multi-Agent-Based Simulation IX, Springer-Verlag Berlin, pp. 110–126, 2009.

[6] H. S. Sarjoughian, B. P. Zeigler, and S. B. Hall, “A layered modeling and simulation architecture for agent-based system development,” Proceedings of the IEEE, vol. 89, no. 2, pp. 201–213, 2001.

[7] H. L. M. Vangheluwe, “DEVS as a common denominator for multi-formalism hybrid systems modelling,” in Proceedings of 2000 IEEE International Symposium on Computer-Aided Control System Design, 2000, IEEE, pp. 129–134.

[8] D. Grossi, F. Dignum, M. Dastani, and L. Royakkers, “Foundations of organizational structures in multi-agent systems,” in Proceedings of the fourth inter-

national joint conference on Autonomous agents and multiagent systems, 2005, ACM Press, p. 690-697.

[9] M. Kolp, P. Giorgini, and J. Mylopoulos, “Multi-Agent Architectures as Organizational Structures,” Autonomous Agents and Multi-Agent Systems, vol. 13, no. 1, pp. 3–25, 2006.

[10] J. Ferber and O. Gutknecht, “A meta-model for the analysis and design of organizations in multi-agent systems,” in Proceedings of the 3rd International Conference on Multi Agent Systems, 1998, IEEE Computer Society, pp. 128–135.

[11] J. R. Busemeyer and A. Diederich, Cognitive Modeling. SAGE Publications, 2010.

[12] M. Bratman, Intention, Plans, and Practical Reason. Cambridge University Press, 1999.

[13] F. F. Ingrand, M. P. George, and A. S. Rao, “An Architecture for Real-Time Reasoning and System Control,” IEEE Expert, vol. 7, no. 6, pp. 33–44, 1992.

[14] A. Sloman, B. Logan, and C. Rich, “Building Cognitively Rich Agents,” Communications of the ACM, vol. 42, no. 3, pp. 71–77, 1999.

[15] R. Sun, Cognition and Multi-Agent Interaction: From Cognitive Modeling to Social Simulation. Cambridge University Press, 2006.

[16] R. Sun, “Cognitive Architectures and Multi-Agent Social Simulation,” in Multi-Agent Systems for Society, Springer-Verlag Berlin, 2009, pp. 7–21.

[17] S. Mittal and S. A. Douglass, “Net-centric ACT-R-Based Cognitive Architecture with DEVS Unified Process,” in Proceedings of the 2011 Symposium on Theory of Modeling & Simulation: DEVS Integrative M&S Symposium, 2011, Society for Computer Simulation International, pp. 1–11.

[18] M. Akplogan, G. Quesnel, and F. Garcia, “Towards a deliberative agent system based on DEVS formalism for application in agriculture,” in 2010 Summer Simulation Multiconference, 2010, Society for Computer Simulation International, pp. 250–257.

[19] B. Watrous, L. Granowetter, and D. Wood, “Hla federation performance: What really matters,” in Proceedings of the 2006 Fall Simulation Interoperability Workshop, 2006.

[20] A. Dardenne, “Goal-directed acquisition,” Science of Computer Programming, vol. 20, pp. 3–50, 1993.

[21]M. Amiguet, J. P. Muller, J. A. Baez-Barranco, and A. Nagy, “The MOCA platform - Simulating the dynamics of social networks,” in Proceedings of the 3rd international conference on Multi-agent-based simulation II, 2003, Springer-Verlag, pp. 70–88.

[22] B. Edmonds, “Towards an ideal social simulation language,” in Proceedings of the 3rd international conference on Multi-agent-based simulation II, 2003, Springer-Verlag, pp. 50–53.

[23] C. Jonker and J. Treur, “Relating structure and dynamics in organisation models,” in Proceedings of the 3rd international conference on Multi-agent-based simulation II, 2003, Springer-Verlag, pp. 1–21.

[24] A. M. Uhrmacher, P. Tyschler, and D. Tyschler, “Modeling and simulation of mobile agents,” Future Generation Computer Systems, vol. 17, pp. 107–118, 2000.

[25] X. Hu and B. Zeigler, “A high performance simulation engine for large-scale cellular DEVS models,” in High Performance Computing Symposium (HPC’04), Advanced Simulation Technologies Conference, 2004, pp. 3–8.

[26] Q. Liu and G. Wainer, “Lightweight Time Warp A Novel Protocol for Parallel Optimistic Simulation of Large-Scale DEVS and Cell-DEVS Models,” in 2008 12th IEEE/ACM International Symposium on Distributed Simulation and Real-Time Applications, 2008, pp. 131–138.

[27] X. Li, W. Mao, D. Zeng, and F. Wang, “Agent-based social simulation and modeling in social computing,” in Proceedings of the IEEE ISI 2008 PAISI, PACCF, and SOCO international workshops on Intelligence and Security Informatics, 2008, Springer-Verlag ,pp. 401–412.

[28] M. P. Georgeff and F. F. Ingrand, “Decision-making in an embedded reasoning system,” in Proceedings of the 11th international joint conference on Artificial intelligence - Volume 2, 1989, Morgan Kaufmann Publishers Inc, pp. 972–978

[29] E. D. Gagné, C. W. Yekovich, and F. R. Yekovich, The cognitive psychology of school learning. HarperCollins College Publishers, 1993.

[30] D. C. Dennett, The Intentional Stance. Mit Press, 1989.

[31] R. H. Bordini, J. F. Hübner, and M. J. Wooldridge, Programming multi-agent systems in AgentSpeak using Jason. John Wiley & Sons, Ltd, 2007.

[32] F. Grimaldo, M. Lozano, and F. Barber, “MADeM: a multi-modal decision making for social MAS,” in Proceedings of the 7th international Conference on Autonomous Agents and Multiagent Systems, 2008, pp. 183–190.

[33] J. Hubner, J. Sichman, and O. Boissier, “Developing organised multiagent systems using the MOISE+ model: programming issues at the system and agent levels,” International Journal of Agent-Oriented Software Engineering, vol. 1, no. 3/4, pp. 370–395, 2007.

[34] A. C. Chow and B. P. Zeigler, “Parallel DEVS: A parallel, hierarchical, modular, modeling formalism,” in Proceedings of the 26th conference on Winter simulation, 1994, Society for Computer Simulation International, pp. 716–722.

[35] P. H. Jacobs, N. A. Lang, and A. Verbraeck, “D-SOL: A Distributed JAVA based Discrete Event Simulation Architecture,” in Proceedings of the 2002 Winter Simulation Conference, 2002, pp. 793–800.

[36] M. Seck and A. Verbraeck, “DEVS in DSOL: Adding DEVS operational semantics to a generic event-scheduling simulation environment,” in Proceedings of the Summer Computer Simulation, 2009, Society for M&S International, pp. 261–266.