A Multi-agent System Development Tool Support for Self-Organization

Maoguang Wang[1,2],Lirong Qiu[2], Fen Lin[2], Zhongzhi Shi[2]

[1]School of Computer Science, China University of Mining and Technology, China
[2]Institute of Computing Technology, Chinese Academy of Sciences, China

Summary
The theory of self-organization and adaptivity has grown out of a variety of disciplines, including thermodynamics, cybernetics and computer modeling. As modern computing environments become more open, pervasive and complex, self-organization is becoming more and more concerned in the multi-agent systems(MAS). Self-organization can be defined as the spontaneous creation of a globally coherent pattern out of local interactions. Self-organizing systems not only regulate or adapt their behaviors to the changing environment but also adjust their own software structure. It requires both connections that integrate the parts into a whole and organize different subsystems to adapt to the changes. In this paper the VAStudio multi-agent development platform is expected to create flexible and self-organization MAS software. VAStudio adopts a hierarchical structure form the fine-grain behavior level, the medium agent level to large-grain society level. Also it adopts the ontology to ensure the interaction message consistency, use policy to control the dynamic load of agents, components, protocols and plug-ins so on. And we give an experiment to illustrate the VAStudio functions.

Key words:
Multi-agent system, self-organization, multi-dimensional hierarchical structure, ontology

Introduction

Multi-agent technologies have been developed more than 20 years and the agent technology is considered to be a promising method to construct the scalable, robust, reusable high quality software system. Intelligent agent and multi-agent technologies provide a new pattern for developing the distributed, intelligent and pervasive systems. Though many relevant theories and platforms are provided by the academia and industry, it is still not easy to develop the agent system.

Due to the complexity of the network and distributed system applications, how to solve the complexity of the computer system management and deployment is becoming more and more concerned. In 2001 IBM releases a manifesto observing that the main obstacle to further progress in the IT industry is a looming software complexity crisis. Therefore IBM proposes the autonomic computing idea [1] to solve the software complexity crisis. In accordance with autonomic computing proposal, IT system should hold the capability to adjust itself without needing much manual intervention, namely, the system can execute autonomically and adapt itself to the changing environment. Also, with the multi-agent system is becoming more and more complex how to develop a self-organization system is getting more and more attention. Generally a self-organizing system not only regulates or adapts its behavior, it also creates its own organization. Organization is usually defined as structure with function. Structure means that a system's components are arranged in a particular order. It requires both connections that integrate the parts into a whole and separations that differentiate subsystems to avoid interference. Function means that this structure fulfills a purpose [2].

Despite the research done in the last years on the development of methodologies for designing MAS[3-8], there still lacks an efficient methodology suitable for the development of self-organization MAS. So far most agent development tools lack the self-organization support, which makes the user understand and develop a self-organization MAS difficult. Consequently we focus on the MAS development support for self-organization. Incorporating the AOSE(Agent Oriented Software Engineering) philosophy, we analyze the design model and agent architecture [7], a multi-dimensional and hierarchical model to support self-organization is designed and the tool VAStudion is developed to support the self-organization development. As a consequence, a self-organization behavior and a self-organization system emerge in the process of interactions between autonomous entities. This approach gives more flexibility, robustness, and is more suitable for the modeling and control of complex system. Compared with the related development tool including JADE(Java Agent DEvelopment Framework), JATlite (Java Agent Template, Lite), Zeus and so on[13,14], VAStudio provides a visual design and programming environment for the MAS. It facilitates the multi-agent development and is convenient for building the self-organization MAS system.

The rest paper is organized as follows: Section 2 introduces self-organization development support model. Section 3 depicts the experiment to illustrate how VAStudio supports the MAS development. Section 4 summaries the paper and discusses the future work.

2. Self-organization Development Support

Visual Agent Studio(VAStudio)[12] developed by our laboratory[12] is a visual MAS development platform and provides the practical agent self-organization support.

Usually abstract method is used to deal with complex systems. According to the multi-agent system characters and abstract levels, we use a multi-dimensional and hierarchical model to develop the multi-agent system. The first dimension is the development process, the second dimension is the self-organization interfaces support including the ontology, policy, protocol and plug-ins. The third dimension is the self-organization coordination model. Figure 1 illustrates the dimension of hierarchical development structure.

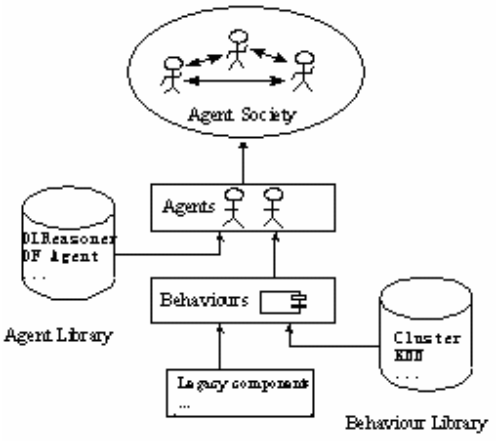

Fig. 1 Hierarchical development structure.

The bottom Layer provides the support for the legacy components. The behavior describes the agent capabilities, which can be conveniently inherited by every child behavior to implement various applications. Although behaviors are components with concrete actions, they cannot execute as independent entities. They are used and added to different autonomous, intelligent and mobile agents. In VAStudio, agents that wrap concrete behaviors can migrate to distributed sites and can communicate and collaborate with their peers. The highest layer of the structure is agent society which can collaborate to resolve the complex tasks according to their roles and goals. In this layer, we can monitor the message transmission, agent collaboration process, and so on.

During the design VAStudio support the two methods: top-down and bottom-up from the fine-grain behavior level to large-grain society level, or vice versa. Figure 2 depicts the VAStudio platform.

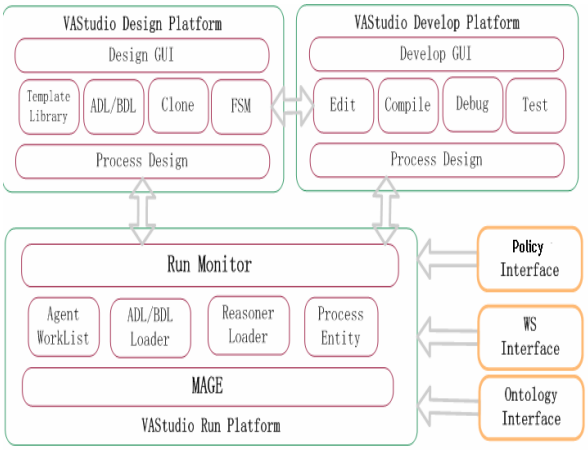

Fig. 2 VAStudio logical structure.

Using visual wizards VAStudio Design Platform supports five generating methods: FSM(Finite State Machine), Workflow Chart, Template Library, Behavior and Agent Description, Cloning, where FSM support the customization of the agent, Workflow chart generates the agent by specifying the task flows, the template supports the various agent models: deliberate, reactive, hybrid etc. The behavior and agent description methods support the simple and complex behavior composition. Agent description illustrates the agent capability, the name, the goal and the needed resource.

VAStudio Develop Platform supports the code edit, compile and debug. Moreover the runtime platform supports the deployment to the MAGE environment [12]. It supports the BDL to describe behavior and ADL to describe agent.

Behavior Generating Algorithm:

<Behaviour>::= <Definition><Capabilities><Extra Classes>
<Definition>::= <Behaviour Name><Behaviour Type><Behaviour Package><Behaviour Description>
<Behaviour Type>::= Behaviour|CyclicBehaviour | FSMBehaviour|OneShotBehaviour|ParallelBehaviour|SequentialBehaviour|SimpleBehaviour
<Capabilities>::=<Capabilities Type><Capabilities Source><Capabilities Command><Capabilities Parameters>
<Capabilities Type>::= Java Component|Executable File|DLL
<Extra Classes>::=<Extra Classes Name><Extra Classes Instance><Extra Classes Parameters>
Agent Generating Algorithm:

<Agent>::=<Definition><Local Address><Acquaintance Addresses><Extra Classes><Capabilities><Environment Variables><Sessions>
<Definition> ::= <Agent Name><Description >
<Extra Classes>::= [Extra Classes] BEGIN{<Class>}*END
<Capabilities>::= [Capabilities]BEGIN{<Capability>}*END
<Capability>::=<Capability_name> <Capability_type> <Capability_command_line> <Capability_source>
<Sessions>::= [Sessions]BEGIN{<Session>}*END
Given a society S and a scene $s \in$ scenes(S), an interaction contract is defined as a tuple, Interaction-contract=<*a*, *s*, *CC*, *P*>, where the set of agents A={ (*a*,*r*) | $a \in$ Agents, $r \in$ roles(s)},*CC* is a set of contract clauses, and *P* is the protocol to be followed. The set A in this definition represents the set of all agents enacting participating in interaction scene *s*, and *CC* is a set of contract clauses describing, that is possible conditions and deadlines concerning the results and interaction patterns of scene s. If the system satisfies the goal of self-organization, it should adjust its behaviors according to the dynamic role and scene.

As to the self-organization interfaces dimension, it supports the behavior library Plug-ins, the agent library Plug-ins, Policy interface, ontology interface, Service Interface, component library etc. Users can reuse the resources via ontologies and services. When modeling ontology VAStudio provides an ontology editor to add or delete concepts and relationships. Now we support the concept classifier, the organization is a tree structure.

As to the self-organization coordination model dimension we provides the coordination model including peer to peer, hierarchical, federation and multi-issue models so on.

Now VAStudio supports the multi-dimensional and hierarchical development. From the agent to the society, it provides rich agent template generators:

- Mage.core.Agent
- Mage.core.ReactiveAgent
- Mage.core.DeliberativeAgent
- Behaviour Library
- Agent Library

Furthermore we are designing the graphical methods to support the FSM, AUML and Flow Chart etc.

We are further enrich the support for the policy interface, ontology interface, web services interface and various coordination models.

3. VAStudio Experiment

Considering various scenes, the agents should adjust their behaviors and structures according to the changing environment. If the agent migrates from one society of Auction to the society of Contract-Net, it should self-organize the collaboration process and interaction behaviors.

To support the self-organization, VAStudio provides[11]:

- The ontology library and the database for describing the agent interaction messages and resources.
- The defined organization relations and coordination models, which mainly include the relations of superior-subordinate, co-worker, peer-peer relationship, and peer to peer model, federation model etc.
- The protocol library contains most of the used protocols, such as FIPA-Contract-Net-Protocol, Negotiation-Protocol, Auction-Protocol, the related Meta-Protocols and so on.
- The policy is used to control the agent interactions and organizations, which depicts when and how to interact with other agents [9].

Figure 3 shows one of the agent generation wizards.

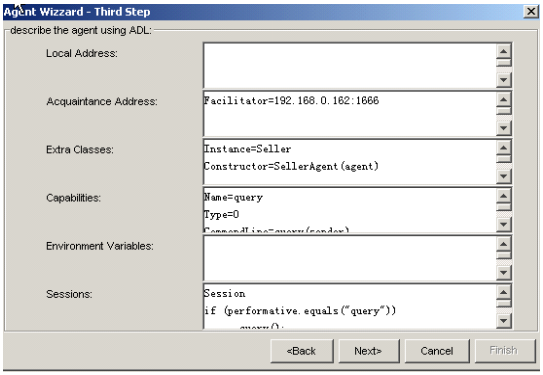

Fig. 3 The agent generation wizard.

Now we set up a FIPA-Contract-Net scene, firstly we develop 3 agents:a1,a2,a3, a1 is the initiator, a2 and a3 are the responders, then add them to the society s1. Now these agents only have basic behaviors. VAStudio just specify the role and FIPA-Contract-Net-Protocol to them respectively. Using top-down method to model the scene: firstly we customize the agent which analyzes the needed behaviors. And it generates the needed behavior code framework automatically. Then according to the function needs we add the code to the framework so as to make the behavior own the corresponding functions. After adding the behaviors to the behavior library we can generate the needed agent. Ultimately a society can be organized including the relevant agents. And the ontology can be specified during the design process. Figure 4 illustrates the agent ontology interface.

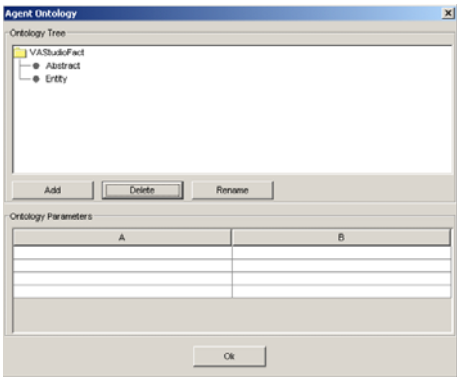

Fig. 4 The ontology interface.

Once the scene is specified the coordination model will be applied automatically during the agent interactions.Now the agent a4 migrates from another auction society s2 to s1 society. VAStudio can load the FIPA-Contract-Net-Protocol, load ontology and assign the corresponding role to a4 agent. The process is showed in figure 5.

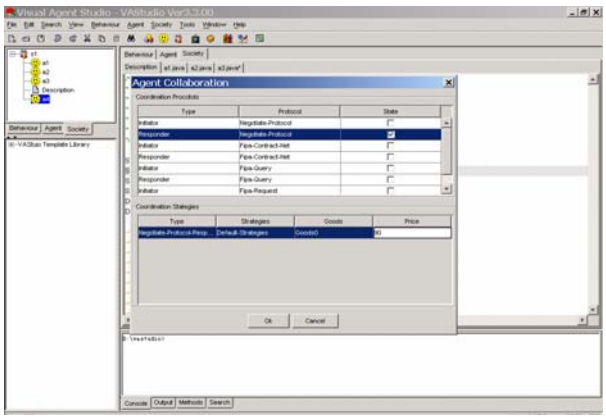

Fig. 5 The new agents role and protocol load.

After that VAStudio can deploy these agents to MAGE[10] platform and verify the organization process and task running. Figure 6 illustrates the Contract-Net-Protocol processes.

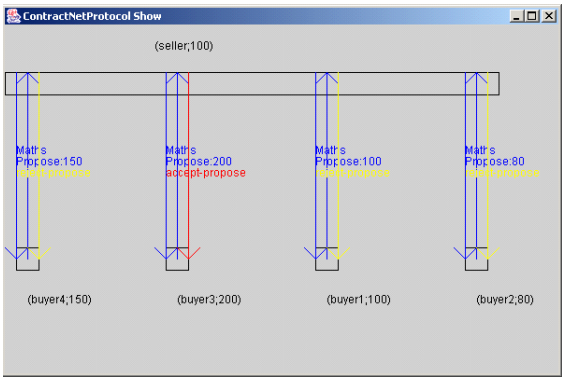

Fig. 6 Contract-Net-Protocol process.

VAStudio has proved its successful development in many projects: GEIS, Decision Support System based on agent and so on [12].

4. Related Work

There has been work on developing the multi-agent system. These works provide useful agent development tools or methodologies. But with the agent system becomes more open and complex how to support the system self-organization when design a multi-agent system is still lack efficient methodologies.

For instance, AgentBuilder[15] Pro provides graphical tools for supporting all phases of the agent construction process. Programming software agents is accomplished by specifying intuitive concepts such as the beliefs, commitments, behavioral rules and actions of the agent. AgentBuilder Pro makes it much easier to create, debug and test multi-agent systems.

JACK Intelligent Agents™ [16] is an agent oriented development environment that is built by the Agent Oriented Software Pty Ltd. It include the components such as the JACK Agent Language, the JACK Agent Compiler, the JACK Agent Kernel. The JACK™ Development Environment (JDE) allows the definition of projects, aggregate agents and teams, and their component parts under these projects. The JDE is a purpose-built toolkit that facilitates the construction of agent/team models.

The work of Vladimir Gorodetski et al [17] presents one Multi Agent System Development Kit based on and implementing of Gaia methodology. It supports the whole life cycle of multi-agent system development and maintains integrity of solutions produced at different stages of the development process. Giacomo Cabri et al [18] compared different approaches based on roles for agent development and proposed the BRAIN [19] framework. Lin Padgham et al.described a PDT

(Prometheus Design Tool) which supports the design of an intelligent agent system using the Prometheus methodology. They described how PDT supports the various stages of Prometheus through various means such as consistency checking, support for entity propagation, and hierarchical views. Fernando Alonoso, Sonia Frutos, et al [20] presented a SONIA methodology(Set of mOdels for a Natural Identification of Agents) based on a generic problem-independent analysis and a bottom-up agent identification process. SONIA naturally outputs an agent-based system.

5. Conclusions

In this paper, we describe the VAStudio tool for developing MAS. When considering the self-organization, VAStudio adopts the multi-dimensional hierarchical design and development approaches, which has proved to be very helpful in understanding and analyzing the self-organizing MAS. To support the self-organization, VAStudio provides the mechanism of policy control, ontology editor, protocol load, self-organization coordination models etc. We will further evaluate and verify the self-organization processes using formal methods.

Acknowledgments

This work is supported by the National High-Program of China (Grant No. 2003AA115220), the National Basic Research and Development Plan of China (Grant No. 2003CB317000) and the Youth Research Program of China University of Mining and Technology (Grant No. 0D4489).

References

[1] Jeffery O. Kephart, David M. Chess. The Vision of Autonomic Computing Outlook. IEEE Computer Society, January 2003, pp.41-47.

[2] Steffen Staab. Neurons, Viscose Fluids, Freshwater Polyp Hydra and Self-Organizing Information Systems Published by the IEEE Computer Society. 2003 IEEE IEEE INTELLIGENT SYSTEMS,pp.72-74.

[3] Marco Mamei, Franco Zambonelli. Self-Organization in Multi-agent Systems: a Middleware Approach pp.1-9

[4] Gang Li, Linyan Sun, Haiquan Li. A Study of the Self-organization Modeling of Distributed Manufacturing Network. Proceedings of the IEEE International Conference on E-Commerce Technology for Dynamic E-Business (CEC-East'04)

[5] Camazine, S, et al. Self Organization in Biological Systems. Princeton University Press, Princeton, U.S.A, 2001.

[6] Pierre Massotte, Roland Bataille. Future production systems: Influence of self-organization on approaches to quality engineering. International Journal of Production Economics 64 (2000), pp.359~377

[7] Javier Vazquerz-Salceda, Virginia Dignum, Frank Dignum. Organizing Multi-agent Systems. Technical report UU-CS-2004-015,Institute of information and computing sciences, ultrecht university.http://www.cs.uu.nl

[8] Alan Colman, Jun Han. Operational management contracts for adaptive software organization. Technical Report No: SUTIT-TR2004.06. CeCSES-TR006, Oct. 2004.

[9] Maoguang Wang, Li Zeng, Jiewen Luo, Qing Yu. A DDL Based Formal Policy Representation. PRIMA2006, LNAI4088, pp. 245-255.

[10] Zhongzhi Shi, Haijun Zhang, Yong Cheng, Yuncheng Jiang, Qiujian Sheng, Zhikung Zhao. MAGE: An Agent-Oriented Programming Environment. IEEE ICCI 2004, pp. 250-257.

[11] Maoguang Wang, Zhongzhi Shi, Jiewen Luo,et al.Dynamic Interaction Protocol Load in Mult-agent System Collaboration. PRIMA2005,pp.101-115.

[12] Intelligent Science Web site: http://www.intsci.ac.cn[OL],2006.

[13] JADE. http://jade.tilab.com/[OL],2006.

[14] Zeus. http://labs.bt.com/projects/agents/zeus/[OL],2006.

[15] AgentBuilder. http://www.agentbuilder.com/[OL],2006.

[16] JACK™ Intelligent Agents.http://www.agent-software.com/[OL],2006.

[17] Vladimir Gorodetski, Oleg Karsaev, et al. MULTI-AGENT SYSTEM DEVELOPMENT KITMAS SOFTWARE TOOL IMPLEMENTING GAIA METHODOLOGY. IIP 2004, Beijing.

[18] Giacomo Cabri, Luca Ferrari, Letizia Leonardi, et al. Role-based Approaches for Agent Development. AAMAS'04, New York, USA.

[19]G. Cabri, L. Leonardi, F. Zambonelli. "BRAIN: a Framework for Flexible Role-based Interactions in Multiagent Systems". The 2003 Conference on Cooperative Information Systems (CoopIS), Catania, Italy, November 2003.

[20] Fernando Alonso, Sonia Frutos, Loic Martinez and Cesar Montes. SONIA: A Methodology for Natural Agent Development. ESAW'2004 - 5th Intl. Workshop on Engineering in the Agents World, Toulouse, France, October 2004