

Design Support Functions for Developing Multiagent System on Repository-based Multiagent Framework

Takahiro Uchiya¹ Tetsuo Kinoshita²

1 Nagoya Institute of Technology 2 Tohoku University

¹ Gokiso-chou, Syowa-ku, Nagoya, 466-8555 JAPAN,

*² 2-1-1 Katahira, Aoba-ku, Sendai, 980-8577 JAPAN,
t-uchiya@nitech.ac.jp, kino@riec.tohoku.ac.jp*

Abstract

Agent systems have been designed and developed using recent agent technologies. However, design and debugging of these systems remain as difficult tasks of designers because agents have situational and nondeterministic characteristics and because no useful design support facilities have been provided for designers. To raise agent system-design efficiency, we propose an interactive design environment for agent system (IDEA) founded on an agent-repository-based multiagent framework. We specifically examine the design support functions for developing multiagent systems and emphasize the effectiveness of these functions.

1. INTRODUCTION

Software with new characteristics such as autonomy and sociality is called agent software. It follows that an information system using agents as its components is called an agent system. Agent systems of many kinds have been designed and developed for internet auctions, network management systems, video conferences, smart grids, etc. using recent agent technologies.

However, the design and debugging of agent systems persists as a difficult problem not only because of the agents' situational and nondeterministic behavioral properties, but also because design-support technologies and effective design methods are lacking. In earlier studies, we examined an agent-repository-based multiagent framework called ADIPS, which accumulates the developed agents and agent systems in an agent repository and which enables the dynamic composition and re-composition of agent systems based on this repository [1][2]. Applying the ADIPS/DASH framework, a recent implementation of the ADIPS framework with an effective agent repository [3][4][5], we developed various agent systems in our previous work [6][7][8][9][10][11][12][13]. Through the experience of developing many agent systems, we paused to realize the

importance of the design support tools to raise the efficiency of the agent system design process.

Therefore, in our research, we propose an Interactive Design Environment of Agent system called "IDEA" founded on an agent-repository-based multiagent framework. In this paper, we specifically examine the design support functions for developing multiagent systems and emphasize the effectiveness of these functions.

2. PROBLEMS IN AGENT SYSTEM DESIGN

2.1. Design Task of Agent Systems

We can discuss design support methods based on the following two approaches from a viewpoint of implementation of the agent system: (i) a Programming Approach and (ii) a Framework Approach.

(i) Programming Approach

In this approach, an agent is designed and implemented in a top-down manner using existing programming languages such as Java. Design flexibility can be retained in the design process. Therefore, agents with dedicated architecture such as basic-type and reactive-type agents can be implemented easily. However, for deliberative-type and composite-type agents, the burden of designers is expected to increase because of the design and implementation of the internal mechanisms embedded in all agents.

(ii) Framework Approach

By providing an agent design support environment based on the specific agent architecture for the designers, the agent design and implementation can be done systematically and efficiently. Such an environment, a 'framework', provides facilities for the designers such as a knowledge representation scheme, a problem-solving function, and an agent communication function. In recent years, many frameworks such as ADIPS [1][2], JADE [14], SAGE [15], AgentBuilder [16], JATLite [17], ZEUS [18], and JACK [19] have been proposed and used. These frameworks might provide many functions to design and

implement various agents for designers. These frameworks typically provide user-oriented support facilities for design, implementation, debugging, and testing.

2.2. Problems of Agent System Design

From the perspective of a state-of-the art of agent system design, we emphasize two problems.

(P1) The designer's burden might increase in direct relation to the size of the target agent system to be designed. For that reason, systematic bottom-up design processing, by which the developed agents are reused, should be supported.

(P2) The costs of both testing and debugging of the agent system become greater than those of non-agent systems. Design support functions by which designers can interact with the target agents flexibly should be required.

To overcome these problems, we first use the repository-based multiagent framework to solve (P1). Moreover, we propose a prototype of the design support environment called IDEA to solve (P2).

3. REPOSITORY-BASED MULTIAGENT FRAMEWORK

3.1. Basic Concept

The ADIPS/DASH framework is the latest repository-based multiagent framework developed by our research group (Fig. 1). The essential functions of this framework can be summarized as described below.

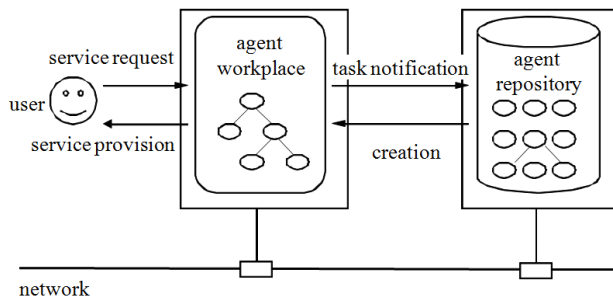


Figure 1. ADIPS/DASH: Repository-based multiagent framework.

F1. Repository-based multiagent framework for distributed problem solving

The ADIPS/DASH framework provides a multiagent platform over the distributed environment, which consists of the distributed Agent Workplaces ('workplaces') and the Agent Repository ('repository').

The repository manages various DASH agents ('agents'). It is responsible for design and realization of the multiagent systems based on the users' requests.

A workplace is an agent execution environment on a distributed computer platform. It is responsible for monitoring and controlling the behavior of agents realized by the repository.

The user can design various multiagent systems by sending a request to the repository and realizing a system on the workplace to execute the problem-solving tasks. Therefore, this framework is called the Repository-based multiagent framework.

F2. Building a multiagent system by Agent Repository

A multiagent system is constructed in the repository and delivered to the workplace. The organization procedure in the agent repository (Fig. 2) is basically the same as the Contract Net protocol [20]. However, the following functions, such as the instantiation of organization onto the workplace and the reconstruction of organization at the run-time of agent systems, are unique and important features of the ADIPS/DASH framework for the development and runtime environment of agent systems.

A user and/or an agent running on the workplace can send a request to the repository to start a design process of a multiagent system, which provides a required service for the user/agent. Receiving a request, the repository seeks suitable agents and constructs an agent organization (a multiagent system) using the Extended Contract Net Protocol (ECNP) of the ADIPS/DASH framework.

The ECNP is an agent cooperation protocol that is used to address the design processes of constructing, reconstructing, and instantiating multiagent systems.

When a required agent organization is constructed in the repository, the repository instantiates the agent organization as a multiagent system run on the specified workplaces. Consequently, the multiagent system can be realized dynamically with respect to the given request.

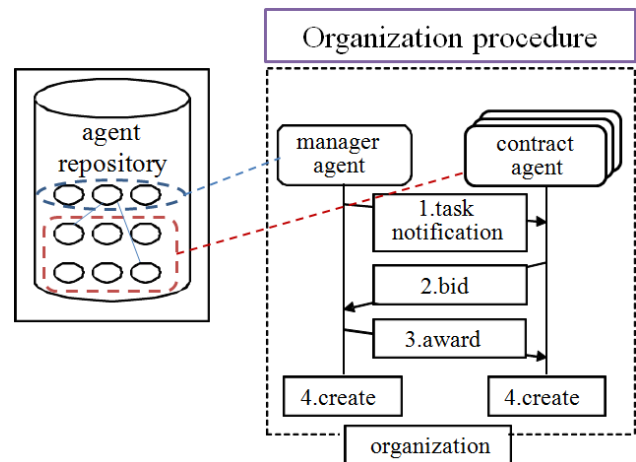


Figure 2. Organization procedure in the agent repository on the ADIPS/DASH framework.

F3. Using multiagent systems on the Agent Workplace

When a multiagent system is instantiated on a workplace by the repository, the workplace activates this system to start a problem-solving task. The workplace monitors the behavior of these agents and collects runtime information such as an execution log.

When the problem-solving task of the multiagent system is finished, the agent organization is dissolved by the workplace. All agents of the multiagent system are stopped and removed from the workplace.

However, it is possible to construct a multiagent system that consists of active agents run on the distributed workplaces using ECNP. In this case, a construction process based on ECNP is similar to the original CNP. Moreover, to use and reuse the realized multiagent system in the future, the workplace can save the multiagent system in the repository by a request for preservation.

F4. Reconstruction of multiagent system at runtime

It is possible to reconstruct the structure and functions of a multiagent system at the runtime of the system based on a request of a user or a problematic agent in the multiagent system. The ECNP provides the functions and protocols for reorganization operation.

F5. Rule-based agent programming

The design of an agent is undertaken to describe the behavior knowledge for cooperative problem solving together with the metaknowledge necessary for managing the agent in the repository.

The behavior knowledge is described as a set of rules using the rule-type knowledge representation format. However, the metaknowledge is described using a frame-type knowledge representation format. The description of the designed agent is called the agent program, which is

interpreted and executed using an inference engine (production system type engine) of the DASH agent.

F6. 'Rule Set' for reusable knowledge

General-purpose and useful behavior knowledge are definable as a pre-defined set of rules, called a 'Rule Set' and stored in the repository, to support the agent design task based on the use/reuse of the rule sets.

The agent designer can select and specify the suitable rule sets in an agent program. In the repository, the specified rule sets are included in a knowledge base of the agent. A set of rules for handling a task-oriented cooperation protocol is an example of the rule set.

F7. Wrapping external program

The ADIPS/DASH framework provides a wrapping mechanism for the agent designers to use external programs such as Java programs as the procedural knowledge of the agent.

F8. Interoperation with other agents

The interoperation mechanism can be included in the ADIPS/DASH framework. Using this mechanism, the DASH agents can communicate with agents of different types, such as FIPA-compliant agents, using the ACL messages of the DASH agent [21].

F9. Testing, debugging and validation of multiagent system

The Interactive Design Environment of Agent system (IDEA) provides an agent/multiagent system design environment. Details of IDEA are explained in the next section.

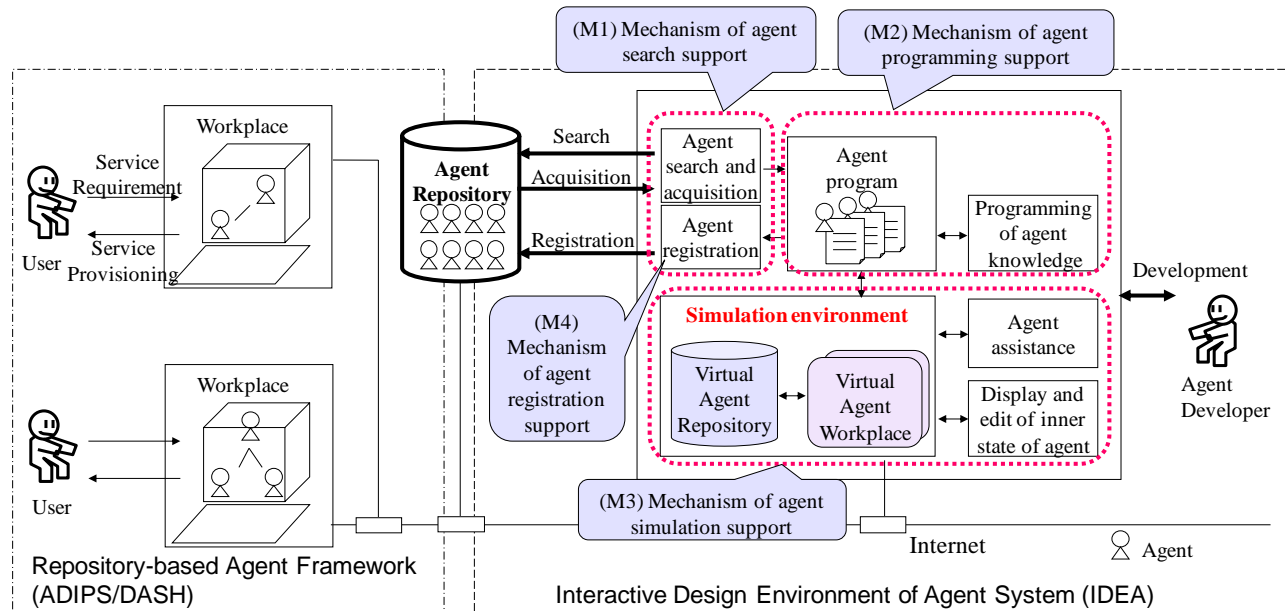


Figure 3. Overview of Interactive Design Environment of Agent system (IDEA).

4. INTERACTIVE DESIGN ENVIRONMENT OF AGENT SYSTEM

We developed a prototype of an interactive design support system called the Interactive Design Environment of Agent system (IDEA). We select and use a repository-based multiagent framework, ADIPS/DASH, for implementation of the IDEA prototype. The functional relations between the ADIPS/DASH framework and IDEA are depicted in Fig. 3.

The ADIPS/DASH is used to realize an agent execution environment equipped with the repository mechanism. Furthermore, IDEA provides an interactive design environment for agent system designers.

The following four mechanisms are introduced into IDEA to support agent design.

(M1) Mechanism of agent search support

This mechanism's three main parts are a search condition input for seeking agents from the repository, a search result display, and a preview of the agent knowledge.

In the search condition input area, a designer inputs the requirement specifications of a candidate agent, such as an agent name and a function name. The received message is displayed in the search result display area when a candidate agent, which is detected in the repository, sends a message as its response. The designer can then move an agent into the developer's environment by choosing the agent from a search result window.

(M2) Mechanism of agent programming support

This mechanism has an agent-programming editor based on a rule-based knowledge representation of the ADIPS/DASH framework. Using this editor, the designer can describe and test the agent programs.

(M3) Mechanism of agent simulation support

As described in this paper, we specifically examine the design support functions for developing a multiagent system. This mechanism has some interactive simulation functions to analyze agent's behavior, such as '**virtual distributed environment**', '**exchange messages between designer and agents**', '**dynamic knowledge modification**', and '**message log analyzer**'. Fig. 4 shows an agent monitor for observing the behavior and organization of the agent in the virtual distributed environment. The agent inspector shows the inner states of an agent to monitor and modify the behavioral knowledge of the runtime agent. Furthermore, the ACL editor supports communication between the designer and the agents under development by the ACL messages of the ADIPS/DASH framework.

Using these tools, the designer can monitor and control the behavior of agents in an interactive manner during the testing and debugging of agents. Moreover, the designer can access other design stages at any time by selecting the design stage tags shown at the upper part of

screen. For instance, by selecting the 'Design' tab, the designer goes to the 'Design' stage to modify the agent program; then the designer will again view the 'Simulation' stage and resume the simulation by reloading the modified agents.

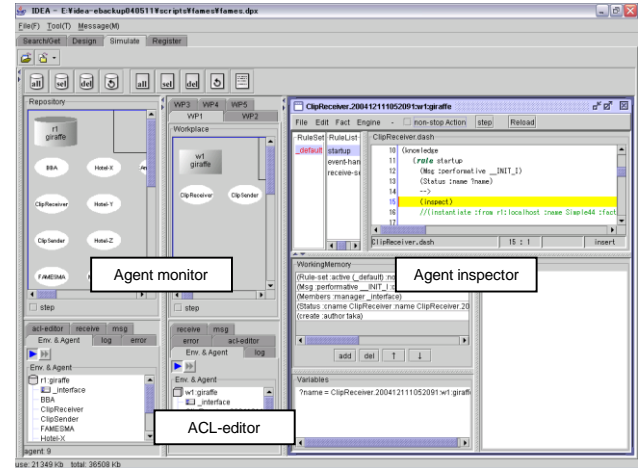


Figure 4. Overview of agent simulation support.

(M4) Mechanism of agent registration support

This mechanism provides an interface to store the completed agent system to the repository.

5. EXPERIMENTS AND EVALUATION

5.1. Experiments

To evaluate the effectiveness of IDEA, we tested the effects of design support functions: 'virtual distributed environment', 'exchange messages between designer and agents', 'dynamic knowledge modification', and 'message log analyzer'.

[Result of the function of virtual distributed environment]

The agent monitor of the simulation interface showed dynamic behavior of distributed agents in real time; the designers can observe non-deterministic behavior such as organization, communication, cooperation, and competition of agents. Therefore, we confirmed that this function is useful to develop the distributed agent system.

[Result of the function of exchange messages between designer and agents]

The ACL-editor tab (Fig. 5) enables the designer to send ACL-messages to agents under development. Consequently, the designer can select and run an agent that is a part of the agent system to test and verify that the agent and agent system function smoothly.

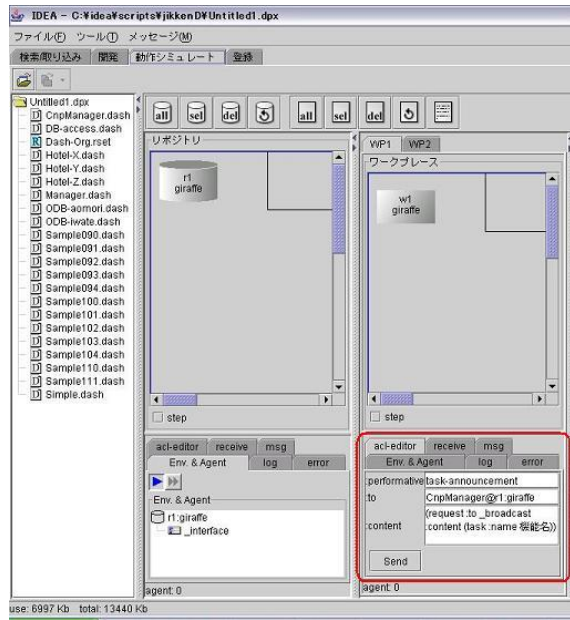


Figure 5. ACL-editor tab.

[Result of the function of dynamic knowledge modification]

The agent inspector portrayed in Fig. 4 enables the designer to modify agent knowledge dynamically at the runtime of an agent system. Thereby, the designer can test and modify the functions of the agent system both quickly and smoothly.

[Result of the function of message log analyzer]

The message log analyzer is portrayed in Fig. 6 and Fig. 7. Figure 6 presents the agent map for analyzing message flow of the whole agent system. Figure 7 portrays the messaging monitor for analyzing the message sequence of the agent system. They are generated using behavioral logs of agents stored in IDEA.

The agent developer can detect a bottleneck that restricts message passing of agent systems using this tool. Moreover, developers can determine the causes of agents' abnormal behavior by monitoring the message sequence.

[Overall evaluation]

We confirmed that all functions are effective for developing the agent system.

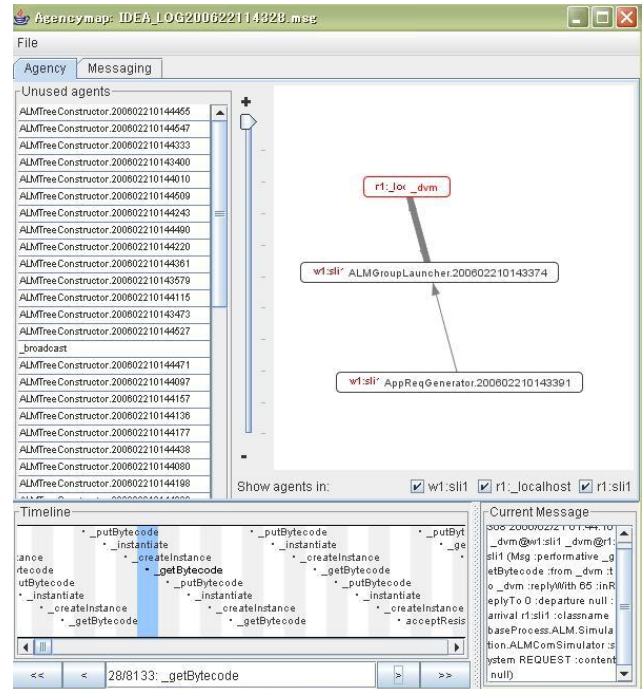


Figure 6. Agent map for analyzing the message flow of the whole agent system.

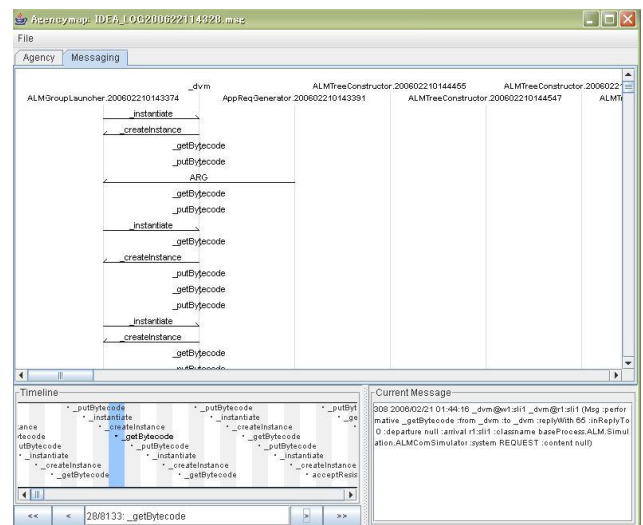


Figure 7. Messaging monitor for analyzing the message sequence of the agent system.

5.2 Comparison with related framework

JADE is an extremely well-known agent framework used to develop Java-based multiagent systems. In terms of behavior analysis of agent systems, we can compare IDEA with JADE.

[Mutual functions]

<1. Virtual distributed environment>

The JADE framework has an agent working environment called a 'container'. It can simulate and monitor the agent behavior in a virtual distributed environment.

<2. Exchange messages between designer and agents>

The JADE framework supports message exchange between a designer and agents using a sniffer function. However, the specifiable message parameter is restricted. IDEA can set the message parameter: 'performative (communicative act)', 'to' and 'content', but JADE can set the message parameter: only 'to' and 'content'.

<3. Messaging monitor for analyzing message sequence>

The JADE sniffer provides this function.

[Unique functions]

<4. Dynamic knowledge modification>

IDEA can change the agent knowledge dynamically at run-time. This function facilitates testing and debugging the unstable agents or uncompleted developed agents.

<5. Agent map for analyzing message flow>

IDEA can monitor the message flow and detect the bottleneck agent or busy agent. This function is indispensable for simulation of massive agent systems.

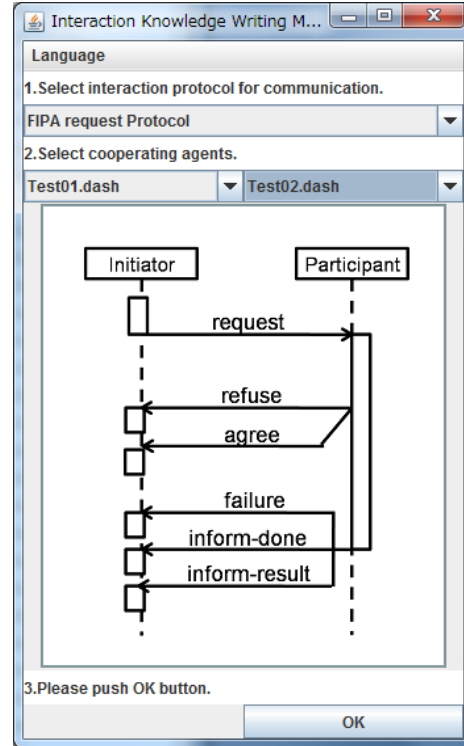


Figure 8. GUI of design support mechanism of agent interaction knowledge

6. CONCLUSION

As described in this paper, we proposed a design support facility of agent systems based on a repository-based multiagent framework to provide an efficient and systematic design environment for agent system designers. A prototype of IDEA is implemented. Experimental results demonstrate the effectiveness of IDEA in the development of agent systems. And now we are developing some experimental functions for designing agent system; "(EF1) Design Support Mechanism of Agent Interaction Knowledge" and "(EF2) Q-Learning Module for Intelligent Agent System". (EF1) provides the GUI to design interaction-protocol based agent knowledge, so we can reduce the burden of developer in system development (Fig.8). (EF2) provides the GUI to develop the intelligent agent equipped with the Q-learning based agent knowledge, and provides the execution environment of learning, so we can make more efficient multiagent system (Fig.9, Fig.10).

In future works, we will enhance the analysis function of agent behavior to test and debug the agent system more smoothly. Moreover, we will expand the proposed design environment by accumulating numerous design cases of agent-based intelligent applications.

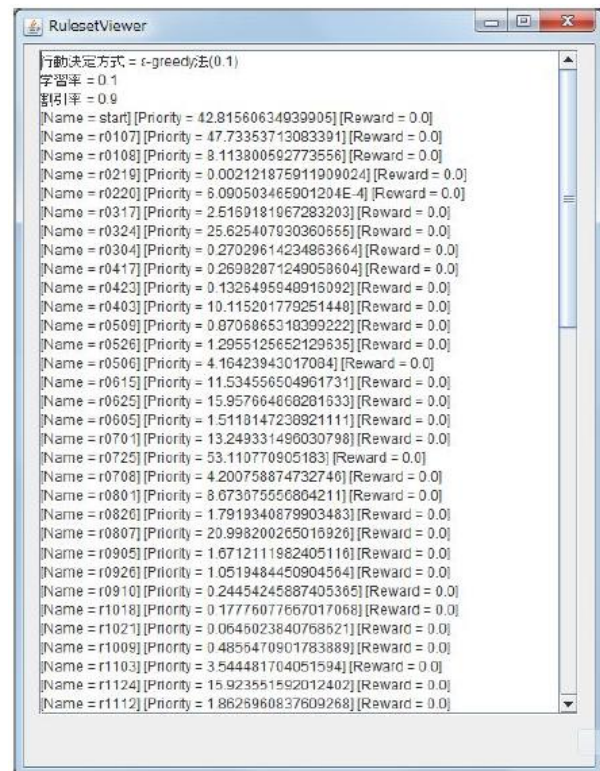


Figure 9. GUI of Q-learning module to monitor the learning status of intelligent agent

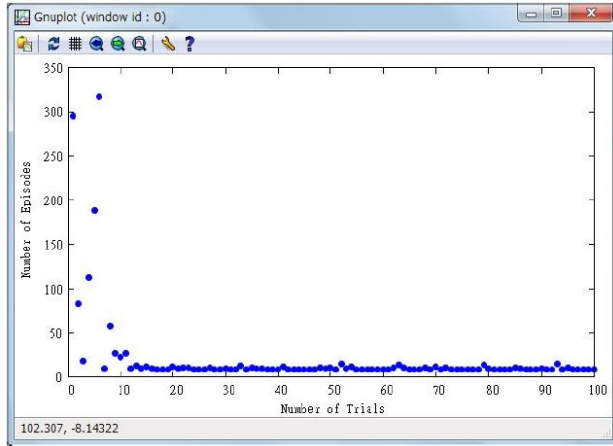


Figure 10. Graph generation function provided by Q-learning module

7. REFERENCES

- [1] Kinoshita, T., Sugawara, K. ADIPS framework for flexible distributed systems. In Proceedings of Pacific Rim International Workshop on Multi-Agents (PRIMA'98 in PRICAI'98), pp.161-175 (1998).
- [2] Fujita, S., Hara, H., Sugawara, K., Kinoshita, T. and Shiratori, N. Agent-based design model of adaptive distributed system. *Applied Intelligence*, Vol.9, No.1, pp.57-70 (1998).
- [3] Hara, H., Sugawara, K., Kinoshita, T. and Uchiya, T. Flexible distributed agent system and its application. In Proceedings of the Fifth Joint conference of Knowledge-based Software Engineering, pp.72-77 (2002).
- [4] Uchiya, T., Suganuma, T., Kinoshita, T. and Shiratori, N. An architecture of active agent repository for dynamic networking. In Proceedings of the First International Joint Conference on Autonomous Agents and Multiagent Systems, pp.1266-1267 (2002).
- [5] Uchiya, T., Takeda, A., Suganuma, T., Kinoshita, T. and Shiratori, N. A method for realizing user-oriented service with repository-based agent framework. In Proceedings of the 1st Int. Forum on Information and Computer Technology (IFICT2003), IPSJ, pp.119-124 (2003).
- [6] Suganuma, T., Imai, S., Kinoshita, T., Sugawara, K. and Shiratori, N. A flexible videoconference system based on multiagent framework. *IEEE Trans. on Systems, Man, and Cybernetics – Part A: Systems and Humans*, Vol.33, No.5, pp.633-641 (2003).
- [7] Imai, S., Kitagata, G., Konno, S., Suganuma, T. and Kinoshita, T. Developing a knowledge-based videoconference system for non-expert users. *Journal of Distance Education Technologies*, Vol.2, No.2, pp.13-26 (2004).
- [8] Konno, S., Iwaya, Y., Abe, T. and Kinoshita, T. Design of network management support system based on active information resource. In Proceedings of the 18th International Conference on Advanced Information Networking and Application, pp.102-106 (2004).
- [9] Kitagata, G., Matsushima, Y., Hasegawa, D., Kinoshita, T. and Shiratori, N. An agent-based middleware for communication service on ad hoc network. In Proceedings of the 19th Int. Conference on Advanced Information Networking and Application, pp.363-367 (2005).
- [10] Takahashi, A., Suganuma, T., Abe, T. and Kinoshita, T. Dynamic construction scheme of multimedia processing system based on multiagent framework. *The International Journal of Wireless and Mobile Computing*, Vol.2, No.1 (2006).
- [11] Abar, S., Konno, S. and Kinoshita, T. Autonomous network monitoring system based on agent-mediated network information. *The International Journal of Computer Science and Network Security*, Vol.8, No.2, pp.326-333 (2008).
- [12] Takahashi, H., Izumi, S., Suganuma, T., Kinoshita, T. and Shiratori, N. Multi-agent system for user-oriented healthcare support. *The International Journal of Informatic Society (IJIS)*, Vol.1, No. 3, pp. 32-41 (2009).
- [13] Kim, H., Kinoshita, T., Lim, Y. and Kim, T. A bankruptcy problem approach to load-shedding in multiagent-based microgrid operation. *Sensors* Vol.10, No.10, pp.8888-8898 (2010).
- [14] Bellifemine, F., Poggi, A. and Rimassa, G. JADE – a FIPA-compliant agent framework. In *Proc. of Practical Application of Intelligent Agents and Multi Agents*, pp.97-108 (1999).
- [15] Ghafoor, A., Rehman, M. ur, Khan, Z. Abbas, Ali, A., Ahmad, H. Farooq and Suguri, H. SAGE: next generation multi-agent system. In *Proceedings of Parallel and Distributed Processing Techniques and Applications*, pp.139-145 (2004).
- [16] Reticular Systems. AgentBuilder – An integrated toolkit for constructing intelligence software agents, available at <http://www.agentbuilder.com/>
- [17] Jeon, H., Petrie, C. and Cutkosky, M. JATLite: a java agent infrastructure with message routing. *IEEE Internet Computing*, Vol.4, No.2, pp.87-96 (2000).
- [18] Nwana, H.S., Ndumu, D.T., Lee, L.C. and Collins, J.C. ZEUS: a toolkit for building distributed multi-agent systems. *Applied Artificial Intelligence Journal*, Vol.13, No.1, pp.129-186 (1999).
- [19] Renquist, N. R., Andrew, H. and Andrew, L. Jack – summary of an agent infrastructure. In *Proceedings of 5th International Conference on Autonomous Agents*, (2001).
- [20] Smith, R. G. The contract net protocol: high-level communication and control in a distributed problem solver. *IEEE Trans. on Computers*, vol.29, no.12, pp.1104-1113 (1980).
- [21] Li X., Uchiya, T., Konno, S. and Kinoshita, T. Proposal for agent platform dynamic interoperation facilitating mechanism. In *Proc. of the 21th Int. Conf. Industrial, Engineering and Other Applications of Applied Intelligent Systems (IEA/AIE 2008)*, LNAI5027, pp.825-834 (2008).