SICE Annual Conference 2010
August 18-21, 2010, The Grand Hotel, Taipei, Taiwan

Realization of Persistency for Meeting Scheduling System Based on Mobile Agent Technology

Takayuki Onishi, Takao Kawamura, Toshihiko Sasama and Kazunori Sugahara
Graduate School of Engineering
Tottori University
4-101, Koyama-Minami
Tottori, JAPAN
Email: {s052015, kawamura, sasama, sugahara}@ike.tottori-u.ac.jp

***Abstract*—We are developing a meeting scheduling system based on mobile agent technology. This system accesses to users from system side by the agent's autonomous operation. Moreover, this system negotiates rescheduling based on sharing information by the communication between agents. This system exists the scheduling server at the center of all agents migrate. The previous system had to constantly keep it running, because of agents migrate irregularly between the scheduling server and user's hosts. If the scheduling server is stopped, moving agents were lost on this system, and agents in user's hosts cannot migrate to other hosts. These wrong actions lead information's inconsistency. In this study, we implemented the function to stop the system temporarily, and the system support stopping the power supply for maintenance, etc. Therefore, a scheduling that stepped over the period when the power supply stopped became possible.**

***Keywords*—Mobile Agent, Groupware, Scheduling.**

I. Introduction

Recently, as the computer network develops, groupwares to advance the joint work efficiently are paid to attention. There is a function for adjustment of schedule in these groupwares. As a general characteristic of groupware with an existing schedule adjustment function, it is assumption that all participants input all schedules beforehand, and those schedules are shared. Therefore, it has large costs to input schedules and security problems of sharing information leakage. Moreover, when there is no convenient date for all participants, groupware that negotiated on the schedule change do not exist. In this situation, the convener has to negotiate voluntarily.

Then, we are developing a meeting scheduling system based on mobile agent technology [1]. This system need only input few label(select free/tentative/busy) to expected few days. In other words, it need not to input all schedule information, and clear the problems of input costs and information leakage. Moreover, this system accesses to users from system side by agent's autonomous operation, and negotiates rescheduling based on sharing information by the communication between agents. The autonomous support by these agents decrease the load of participants.

This system exists the scheduling server at the center of all agents migrate. In use of mobile agent technology of this system, the scheduling server must always keep it running. However, the case exists that want to stop the scheduling server temporarily for maintenance, etc. Because of agents migrate irregularly between the scheduling server and user's hosts, if the scheduling server is stopped, moving agents were lost on this system, and agents in user's hosts cannot migrate to other hosts. In groupwares based on server/client model, because the server collect all information, it keep information consistency when server is stopped. But in our system, agents on hosts (and on the scheduling server) have distributed information. Then, these agent actions when the scheduling server is stopped lead information's inconsistency in this agent based system. In this study, the function to stop the system temporarily is implemented on this system, like groupwares, by collecting all information to the scheduling server.

II. Meeting Scheduling System

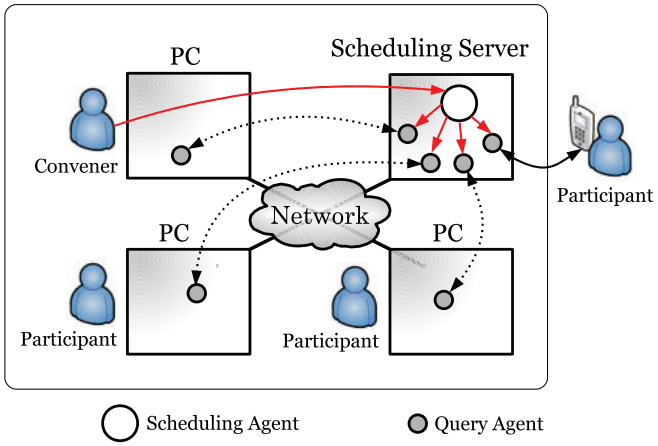

Fig. 1. Overview of the meeting scheduling system.

Fig. 1 is a overview of the system. This system are composed of many type agents, as to adjust schedule, to control users login/logout, etc.

Agents who adjust schedule are composed of scheduling agents and query agents. Firstly, a scheduling agent are created for meeting requested from a convener on the server. And this agent create query agents for each participant of this meeting. Each created query agent migrate to participant's host and request to input his/her schedule. The scheduling agents collect schedules from returned query agents, and search a free time of all participants of this meeting. If the agent cannot find a

free time, it migrates to some participants hosts they blocked to gather all and negotiates rescheduling. At last, each query agent inform them the result of scheduling.

If a participant does not login to the system when a agent who adjust the schedule intends to migrate to his/her host, the agent waits a certain period of time on the scheduling server. After timeout, the agent sends an email to the participant. The email includes a URL, and the agent provide web interface for schedule input in this URL. Participants answer to the agent using this interface.

III. System Suspend/Resume Function

For the function of persistency, the following functions are required generally.

- Suspend: Stop the process, and save its statuses to secondary storage as a file.
- Resume: Using a file stored on secondary storage, create the process and copy statuses of suspended.

In this system, each agent and the scheduling server which manage all agents need a function of persistency.

In the previous system, the scheduling server must always keep it running. If the scheduling server want to stop for maintenance, etc., we can not stop the system, even temporarily. As mentioned in the previous section, if we stop the scheduling server in the situation that agent distributed many hosts, the system cannot keep information consistency. Then, we design and create the callback agent. When the server stop requested, the system create callback agents for all hosts and send there. In each host, the callback agent command to return to all agents in it. By working of callback agents, all agents who adjust the schedule are gathered in the scheduling server, and after, the use of the system is stopped temporarily. Because all agents are forbidden to migrate in this situation, the scheduling server can be stopped with information consistency. Fig. 2 is a overview of the system suspend function.

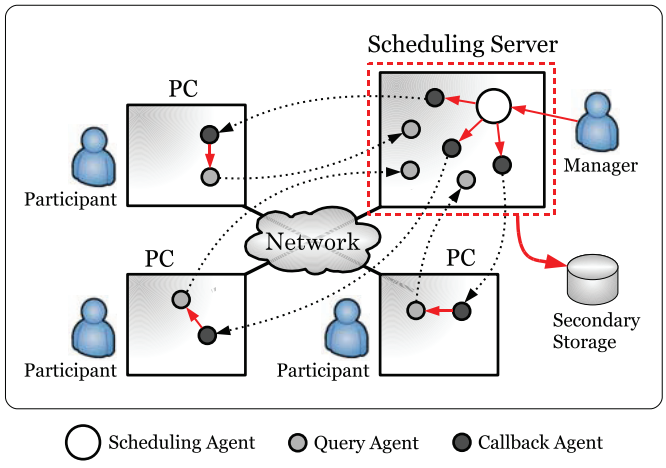

Fig. 2. Overview of the system suspend function.

To achieve stopping the system temporarily, the following functions are necessary.

- System suspend function: Firstly collect all agents in the system from any hosts, and suspend these agents. Next, suspend the scheduling server, and at last, shutdown the system.
- System resume function: Firstly start the system, and resume the scheduling server. Next, resume agents, and at last, agents migrate participant's hosts if necessary.

This system has been implemented using mobile agent framework named Maglog [2]. System suspend/resume function is implemented by using the function of persistency implemented on Maglog [3].

IV. Experiments

We experiment to verify the suspend/resume function of the system. The system for experiments include one scheduling server and twelve hosts to use participant.

They are many situations that may move through the system used: for example, some agents migrate and some agents request to input schedule in the same time, etc. In all situations used by twelve participants, the system was stopped safely, and was restarted with information consistency. As a result, it was confirmed that the system suspend/resume function was able to be implemented correctly.

V. Conclusion

In this study, we implemented the function to stop the system temporarily, and the system support stopping the power supply for maintenance, etc. Therefore, a scheduling that stepped over the period when the power supply stopped became possible.

Future tasks include support the meeting that exceeds time limit of scheduling while the scheduling server has stopped, and automation of stopping the system by registering time beforehand.

References

[1] Y. Hamada, S. Motomura, T. Kawamura, and K. Sugahara, "NAT Traversal Method for Multi-Agent-based Meeting Scheduling System," in *Proceedings of the Third International Conference on Internet and Web Applications and Services*, 6 2008, pp. 223–226, Athens, Greece.

[2] S. Motomura, T. Kawamura, and K. Sugahara, "Logic-Based Mobile Agent Framework with a Concept of "Field"," *IPSJ Journal*, vol. 47, no. 4, pp. 1230–1238, 4 2006.

[3] S. Motomura, J. Kishida, T. Kawamura, and K. Sugahara, "Realization of Persistency in a Multi-Agent Framework," in *Proceedings of IEEE International Conference on Integration of Knowledge Intensive Multi-Agent Systems*, 4 2007, pp. 28–33, Waltham, Massachusetts, USA.