Multi-Agent-based Approach for Meeting Scheduling System

Takao KAWAMURA, Yusuke HAMADA, and Kazunori SUGAHARA
Department of Information and Knowledge Engineering, Faculty of Engineering, Tottori University
4–101, Koyama-Minami, Tottori 680–8552, JAPAN
+81 857 31 5217
{kawamura, sugahara}@ike.tottori-u.ac.jp

Kengo KAGEMOTO and Shinichi MOTOMURA
Graduate School of Engineering, Tottori University
4–101, Koyama-Minami, Tottori 680–8552, JAPAN
+81 857 31 6100
motomura@tottori-u.ac.jp

Abstract—*In this paper, a meeting scheduling system based on mobile agent technology is proposed. The users of the system do not need to input all of their schedules unlike the existing groupwares. When a user intends to call a meeting, he only inputs information about the meeting. On behalf of the inviter, mobile agents move around each invited user's computer to ask whether he can join the meeting and negotiate with him if necessary. Therefore, the inviter's work becomes less compared with using the existing groupwares. We have developed the system with our mobile agent framework on Java platform and confirmed its effectiveness through experiments.*

1.INTRODUCTION

The necessity of face-to-face meeting has not decreased at all, even though interactive media such as telephone, facsimile, and email have diffused widely. In general, the following steps are required to organize a meeting.

1. Specify who should join the meeting and a timeframe during which the meeting must be held.
2. Send emails or phone all the participants a number of times to collect their schedules.
3. Merge their schedules and fix the date and time of the meeting.
4. Select appropriate participants, ask them to open the schedule, and return to step 3 if there is no date and time available, otherwise go to step 5.
5. Notify the arranged date and time of the meeting to all the participants.

Those are time-consuming routine tasks, especially steps 3 and 4.

Some sort of groupware would assist us to organize a meeting. The term groupware refers to software applications, such as Lotus Notes/Domino [1], Microsoft Exchange Server [2], and Cybozu Share360 [3], designed to allow a group of users on a network to work simultaneously on a project. Groupware may provide services for communicating, group document development, scheduling, and tracking.

To utilize a groupware to organize a meeting, all schedule of participants must be managed by the groupware server, therefore all possible participants are requested to input their schedules into the server always. If anyone fails to maintain his schedule on the server, the scheduling function of a groupware becomes useless. In addition, although existing groupwares have a function to find the date and time on which all participants schedule is open, they don't have a function to negotiate with appropriate participants to open the schedule. That is to say, the above most difficult steps 3 and 4 are left to people.

In this paper, a meeting scheduling system based on mobile agent technology is proposed to reduce time and effort on meeting scheduling. Although there are many reports on the agent-based meeting scheduling, especially algorithms or strategies for negotiation among multi-agents [4, 5, 6, 7, 8], there are few on developing an application system which includes the following features:

1. Nobody is requested to input his all schedule into a server before any meeting is intended to call.
2. An agent selects appropriate participants and asks them to open the schedule.

This paper is organized in 5 sections. The design of the proposed system and an implementation of the system on Java platform are described in Section 2 and 3, respectively. In Section 4, we present the result of experiments. Finally, in Section 5, we describe some concluding remarks.

2.Meeting Scheduling System

In this section the design of the proposed meeting scheduling system is presented.

Main Concept

It is the important point in designing a meeting scheduling system that the means to be used for collecting schedule data and for negotiating. Email and web are popular means to do those things, i.e., requests from the system to users are sent via email, on the other hand, replies from users to the system are sent directly via email or via URL informed by the requesting email. However, we consider those means are not enough because email is often disregarded. We therefore design the proposed system as a multi-agent system, i.e., we make agents migrate to user's PC for collecting schedule data and for negotiating. A migrated agent opens a window on user's PC for collecting schedule data and for negotiating and it doesn't disappear until the user answers. However, it is necessary to leave email and web as alternative means for the case that a user cannot login over a long period of time when on a business trip etc.

Figure 1 illustrates the overview of the proposed system. As shown in Fig. 1, there is a scheduling server which runs always as long as the proposed system provides the meeting scheduling service. A user management agent which stands on the scheduling server manages user information that include login name, password, IP address of the user's computer, and online/offline status. Login name and password are static while IP address and online/offline status are updated dynamically when a user logins to the system. A user can use any computer on the network because his login name is tied up with the IP address of his computer dynamically. A scheduling agent corresponding to a meeting creates query agents for each participant of the meeting. Each query agent asks one participant's schedule concurrently. The scheduling agent does the rest of the work for meeting scheduling.

Collecting schedule data

When a user of the proposed system intends to call a meeting, he/her selects participants from registered users. In addition, he/her provides the further information about the meeting that consists of a timeframe during which the meeting must be held, the duration of the meeting, and the subject of the meeting. Then a scheduling agent corresponding to the meeting is created on the user's computer to arrange the meeting. It migrates to the scheduling server immediately so that the inviter can logout from the system before the meeting is arranged. The IP address of the scheduling server is assumed as known.

Then query agents for each participant of the meeting are created. Each query agent obtains the IP address of a participant's computer from the user management agent, and then migrates to the participant's computer to ask his/her schedule concurrently. Both the inviter and the invitees are requested to classify each hour in the timeframe into three categories: "free", "tentative", and "busy". A query agent may ask a user whether he can join a meeting only when the meeting will be held across his/her tentative hours. In addition to the three categories, an inviter of a meeting can select the category named "preferable" in which he hopes the meeting will be held.

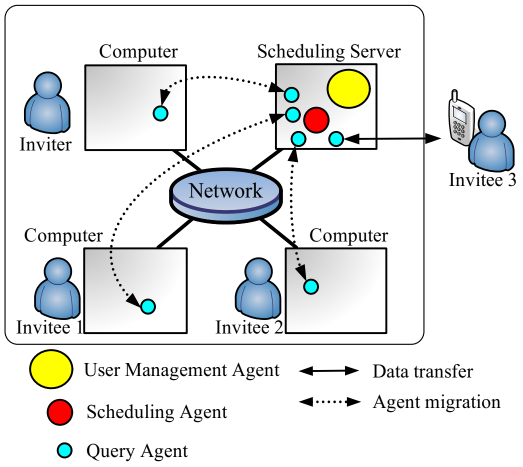


Figure 1 - Overview of the meeting scheduling system.

If a user doesn't login to the system when a query agent intends to migrate to his/her computer, the query agent waits a certain period of time on the scheduling server. After timeout, the query agent sends an email to the user informing him/her of a particular URL and waits his/her connection as a web server. The user can tell his/her schedule to the query agent through the URL.

Scheduling algorithm including negotiation

After all the participants' schedules are collected successfully, the scheduling agent arranges the meeting by using the following procedure:

1. Merge all the participants' schedules and generate a list which contains all the possible continuous hours up to the meeting length. Hours classified as busy by one more participants are excluded.

2. Order the list by the number of hours relating a negotiation, by the number of persons relating a negotiation, and by the inviter's preference.

3. The procedure exits with the success status if no negotiation is needed, i.e., the number of hours relating a negotiation of the top element of the list is zero.

4. Retrieve each element of the list and negotiate with the related participants to open their schedules until a nego-

tiation succeeds or the list is empty. The procedure exits with the success status when the former case occurs. The latter case means that the return status of the procedure is failure.

As an example, we examine how a scheduling agent negotiates with participants when user A intends to hold a two hours meeting with user B between July 5th and July 6th. We assume that user A and user B inputted their schedules shown in Table 1. A blank cell in Table 1 means that user A or user B is free on that time.

Table 1 - Schedules of user A and user B between July 5 and July 6.

	July 5		July 6	
	user A	user B	user A	user B
9	tentative	tentative		busy
10	tentative	tentative		busy
11				busy
12		tentative		busy
13		tentative	busy	
14		busy	busy	
15	preferable	busy	busy	
16	preferable	busy	busy	

Table 1 shows that some negotiation is needed to hold a two hours meeting between July 5th and July 6th. Table 2 shows durations to be selected as candidates for negotiation in the order of priority. A scheduling agent negotiates with user A and/or user B to open the schedule in that order. Note that it avoids useless negotiation. For example, If one of user A and user B denies to open the schedule on the third duration in Table 2, it does not ask the other user for opening the schedule since the negotiation succeeds only if both two persons agree to open the schedule. In addition, if a negotiation on the third duration fails, it does not intend to negotiate on the fourth duration in Table 2 since there is no chance that the negotiation succeeds.

Table 2 - Possible durations of the meeting on July 5 sorted in the order for negotiation.

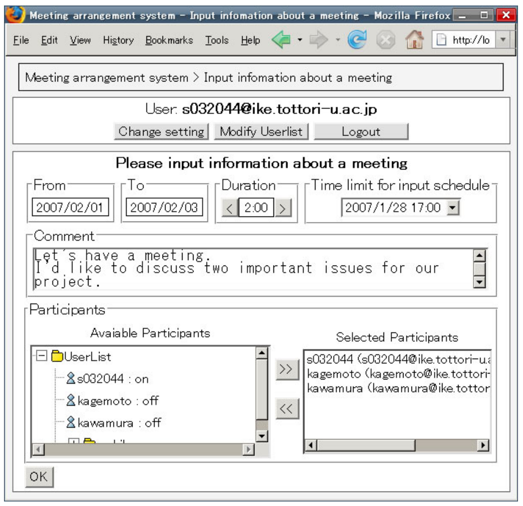
Duration	Number of related hours	Number of related persons
11–13	1	1
12–14	2	1
10–12	2	2
9–11	4	2

At last the scheduling agent notifies the result of the meeting arrangement to both the inviter and the invitees.

3.IMPLEMENTATION

An implementation of the proposed system has been developed on Java platform by using our mobile agent framework named Maglog[9, 10, 11] . The user interface of the proposed system has been implemented as a Web application using Ajax technology.

As mentioned in Section 2, when one intends to call a meeting, he needs to select participants and to provide the further information about the meeting that consist of a timeframe during which the meeting must be held, the duration of the meeting, and the subject of the meeting. Figure 2 shows a sample screen-shot of selecting participants and inputting information about a meeting. One can select participants through moving leaves from the tree in the `Available Participants` pane to the `Selected Participants` pane. If a selected node is leaf then one person corresponds to the node is added to the participants list, otherwise persons correspond to descendant leaves of the dropped node are added to the participants list at once. An `:off` suffix on a leaf node indicates that the corresponding person is in the offline status.


Meeting arrangement system - Input infomation about a meeting - Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://lo
Change setting
Modify Userlist
Logout
>>
<<
OK

Meeting arrangement system > Input infomation about a meeting

User: s032044@ike.tottori-u.ac.jp

Please input information about a meeting

From: 2007/02/01
To: 2007/02/03
Duration: 2:00
Time limit for input schedule: 2007/1/28 17:00

Comment:
```
Let's have a meeting.
I'd like to discuss two important issues for our
project.
```

Participants

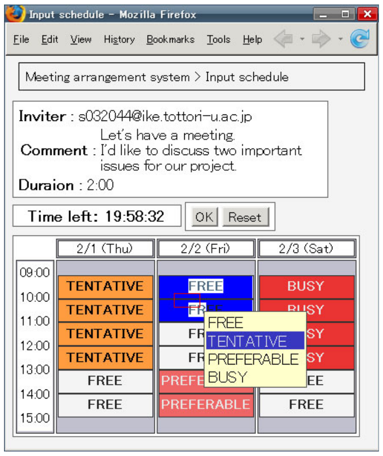
Avaiable Participants
- UserList
  - s032044 : on
  - kagemoto : off
  - kawamura : off

Selected Participants
- s032044 (s032044@ike.tottori-u.a
- kagemoto (kagemoto@ike.tottori-
- kawamura (kawamura@ike.tottor

Figure 2 - Window for selecting participants and inputting information about a meeting.

When the `OK` button is pressed, a scheduling agent is created and migrates to the scheduling server immediately. Continuously, query agents for each participant are created. each query agent migrates to a participant's computer to ask his/her schedule by opening a schedule window as shown in Fig. 3. Through this window, one can input his schedule; select a range of hour-cells by holding down the left mouse button and dragging the mouse over the cells; right-click on the cells; choose whether he will be `free`, `tentative`, or `busy` in the pop-up selection box. Only an inviter of a meeting can choose the fourth option `preferable`. Note that a blank cell means that the user is free on that time therefore there is no need to select the `free` option ordinarily. One should select it only when he intends to overwrite other options.

If a user doesn't login to the system when a query agent intends to migrate to his/her computer, the query agent sends


Input schedule - Mozilla Firefox
File Edit View History Bookmarks Tools Help
Meeting arrangement system > Input schedule
OK
Reset

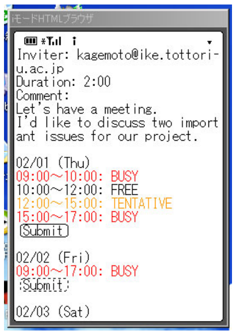
Inviter : s032044@ike.tottori-u.ac.jp
Comment : Let's have a meeting. I'd like to discuss two important issues for our project.
Duraion : 2:00

Time left: 19:58:32

	2/1 (Thu)	2/2 (Fri)	2/3 (Sat)
09:00	TENTATIVE	FREE	BUSY
10:00	TENTATIVE	FREE	BUSY
11:00	TENTATIVE	FR	SY
12:00	TENTATIVE	FR	SY
13:00	FREE	PREFE	EE
14:00	FREE	PREFERABLE	FREE
15:00			

FREE
TENTATIVE
PREFERABLE
BUSY

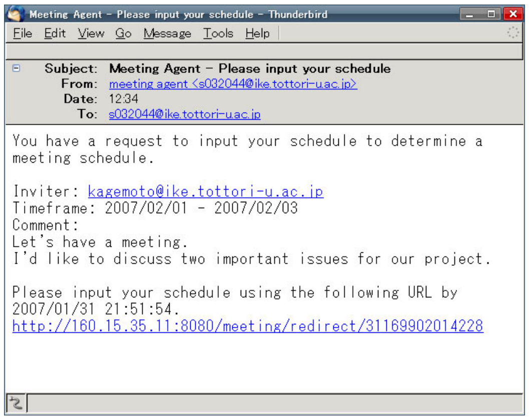
Figure 3 - Window for an online user to input schedule data.



```
Inviter: kagemoto@ike.tottori-
u.ac.jp
Duration: 2:00
Comment:
Let's have a meeting.
I'd like to discuss two import
ant issues for our project.

02/01 (Thu)
09:00~10:00: BUSY
10:00~12:00: FREE
12:00~15:00: TENTATIVE
15:00~17:00: BUSY

02/02 (Fri)
09:00~17:00: BUSY

02/03 (Sat)
```

Figure 5 - Window for an offline user to input schedule data.

an email to the user informing him/her of a particular URL as shown in Fig. 4. As shown in Fig. 5, the user can tell his/her schedule data to the query agent through the URL.


Meeting Agent - Please input your schedule - Thunderbird
File Edit View Go Message Tools Help

Subject: Meeting Agent - Please input your schedule
From: meeting agent <s032044@ike.tottori-u.ac.jp>
Date: 12:34
To: s032044@ike.tottori-u.ac.jp

```
You have a request to input your schedule to determine a
meeting schedule.

Inviter: kagemoto@ike.tottori-u.ac.jp
Timeframe: 2007/02/01 - 2007/02/03
Comment:
Let's have a meeting.
I'd like to discuss two important issues for our project.

Please input your schedule using the following URL by
2007/01/31 21:51:54.
http://160.15.35.11:8080/meeting/redirect/31169902014228
```

Figure 4 - Email sent to an offline user inviting him/her to a meeting.

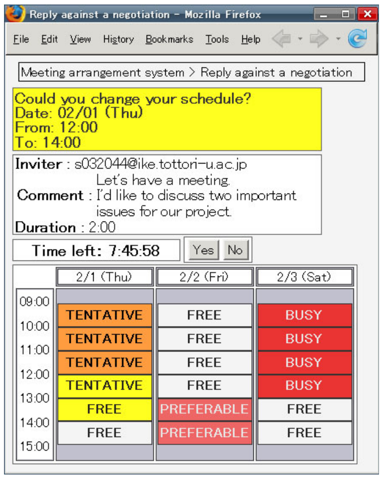
Figure 6 shows a window for negotiation. If a user which is asked to open his/her schedule doesn't login to the system, an email is send to the user informing his/her of a particular URL to negotiate as the case to input his/her schedule.

If a user doesn't login to the system when a query agent intends to migrate to his/her computer, the query agent sends an email to the user informing him/her of a particular URL as shown in Fig. 7. As shown in Fig. 8, the user can negotiate with the query agent through the URL.

4. Experiments

This section presents experimental results obtained from the implementation of the proposed system described in the previous section.

The experimental environment consists of PCs with Intel


Reply against a negotiation - Mozilla Firefox
File Edit View History Bookmarks Tools Help
Meeting arrangement system > Reply against a negotiation
Yes
No

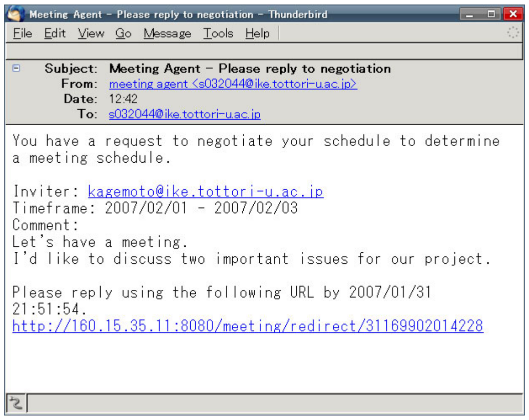
Could you change your schedule?
Date: 02/01 (Thu)
From: 12:00
To: 14:00

Inviter : s032044@ike.tottori-u.ac.jp
Comment : Let's have a meeting. I'd like to discuss two important issues for our project.
Duration : 2:00

Time left: 7:45:58

	2/1 (Thu)	2/2 (Fri)	2/3 (Sat)
09:00	TENTATIVE	FREE	BUSY
10:00	TENTATIVE	FREE	BUSY
11:00	TENTATIVE	FREE	BUSY
12:00	TENTATIVE	FREE	BUSY
13:00	FREE	PREFERABLE	FREE
14:00	FREE	PREFERABLE	FREE
15:00			

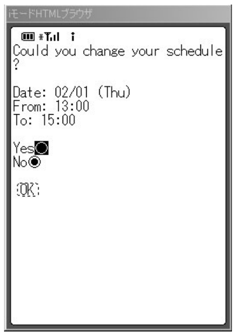
Figure 6 - Window for negotiation with an online user.


Meeting Agent - Please reply to negotiation - Thunderbird
File Edit View Go Message Tools Help

Subject: Meeting Agent - Please reply to negotiation
From: meeting agent <s032044@ike.tottori-u.ac.jp>
Date: 12:42
To: s032044@ike.tottori-u.ac.jp

```
You have a request to negotiate your schedule to determine
a meeting schedule.

Inviter: kagemoto@ike.tottori-u.ac.jp
Timeframe: 2007/02/01 - 2007/02/03
Comment:
Let's have a meeting.
I'd like to discuss two important issues for our project.

Please reply using the following URL by 2007/01/31
21:51:54.
http://160.15.35.11:8080/meeting/redirect/31169902014228
```

Figure 7 - Email sent to an offline user to ask him/her for opening the schedule.



Could you change your schedule
?

Date: 02/01 (Thu)
From: 13:00
To: 15:00

Yes
No

OK

Figure 8 - Window for negotiation with an offline user.

Pentium4 3.0GHz processor. They are connected through 1000Base-T Ethernet and are running on Turbolinux 10 operating system whose kernel version is 2.6.0. The version of the Java language runtime environment is 1.4.2, and its heap size is 512MB.

We examined how the number of meetings were able to be scheduled without "Out of Memory" error. The result is shown in Figs. 9 and 10. Although the number of schedulable meetings decreases as the number of participants increases, the number of agents where the maximum number of meetings is scheduled concurrently on the scheduling server is almost constant.

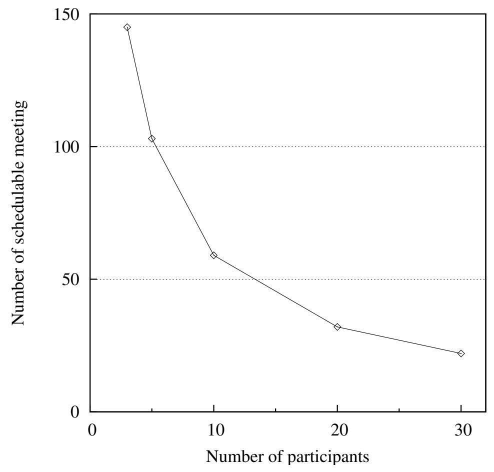


Figure 9 - The change of the number of schedulable meetings when the number of participants increases.

5.Conclusion

In this paper, a meeting scheduling system based on mobile agent technology is presented to reduce time and effort on meeting scheduling. With the proposed system, all an inviter of a meeting has to do is to specify who should join the meeting, a timeframe during which the meeting must be held, the duration of the meeting, and the subject of the meeting. The rest is done by agents, i.e., query agents collect all the participants' schedule, and a scheduling agent searches the date and time available for the meeting, negotiates with appropriate participants if necessary, and finally notifies the arranged date and time of the meeting to all the participants. The proposed system simplifies the work not only of an inviter, but also of all participants. Instead of inputting all schedule into a server before any meeting is intended to call, one is requested to input only the schedule between the timeframe of a meeting.

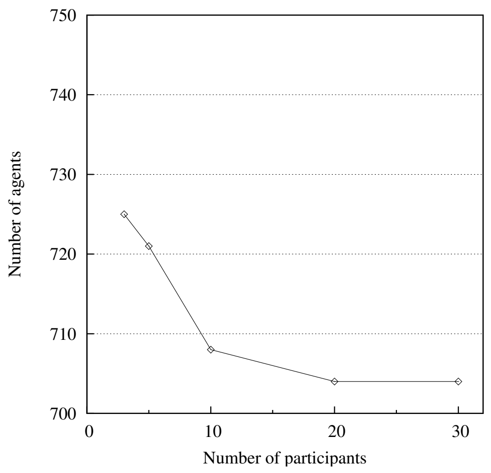


Figure 10 - The change of the number of agents when the number of participants increases.

To make the proposed system more practical, it is necessary to consider the priorities of participants, i.e., the key person of a meeting must join it; on the contrary, a meeting may be opened without persons with low prior. In addition to that, when the proposed system is used in a large organization, it is necessary to increase the number of schedulable meetings. That will be achieved by a distributed approach in future work.

References

[1] IBM: Lotus Notes/Domino (2005). http://www-306.ibm.com/software/lotus/.

[2] Microsoft: Microsoft Exchange Server (2005). http://www.microsoft.com/exchange/default.mspx.

[3] Cybozu: Share360 (2005). http://cybozu.com/.

[4] Sen, S. and Durfee, E. H.: On the design of an adaptive meeting scheduler, *Proceedings of the Tenth IEEE Conference on Artificial Intelligence for Applications*, pp. 40–46 (1994).

[5] Jennings, N. R. and Jackson, A. J.: Agent-based Meeting Scheduling: A Design and Implementation, *IEE Electronics Letters*, Vol. 31, No. 5, pp. 350–352 (1995).

[6] Bui, H. H., Venkatesh, S. and Kireonska, D.: Learning Other Agents' Preferences in Multiagent Negotiation, *Proceedings of the Thirteenth National Conference on Artificial Intelligence*, Vol. 2, pp. 114–119 (1996).

[7] Garrido, L. and Sycara, K.: Multi-agent meeting scheduling: Preliminary experimental results, *Proceedings of the Second International Conference on Multi-Agent Systems*, AAAI Press, pp. 95–102 (1996).

[8] Modi, P. J. and Veloso, M.: Multiagent Meeting Scheduling with Rescheduling, *Proceedings of the Fifth Workshop on Distributed Constraint Reasoning* (2004).

[9] Kawamura, T., Motomura, S. and Sugahara, K.: Implementation of a Logic-based Multi Agent Framework on Java Environment, *Proceedings of International Conference on Integration of Knowledge Intensive Multi-Agent Systems* (Hexmoor, H.(ed.)), pp. 486–491 (2005). Waltham, Massachusetts, USA.

[10] Motomura, S., Kawamura, T. and Sugahara, K.: Logic-Based Mobile Agent Framework with a Concept of "Field", *IPSJ Journal*, Vol. 47, No. 4, pp. 1230–1238 (2006).

[11] Motomura, S., Kawamura, T. and Sugahara, K.: A Logic-Based Mobile Agent Framework for WEB Applications, *Proceedings of the 2nd International Conference on Web Information Systems and Technologies*, pp. 121–126 (2006). Setubal, Portugal.