Scheduling meetings using participants' preferences

José A. Pino and Hugo A. Mora
Universidad de Chile, Santiago, Chile

Introduction

Electronic group calendaring systems were among the first applications of the research and development efforts in the field of computer-supported cooperative work (CSCW). There are some products in the market, but their use by people in organizations is very limited. It is a relevant question to ask why there is such little success in this area. The answer will probably be a complex combination of several factors. The purpose of this paper is to present a hypothesis for one of these factors and a way to solve it.

Our study begins with the way meeting time coordination has been approached in the past. In one basic strategy, people feed their time availabilities into the computer system and it is the system which figures out the best time slot for a meeting. In a second approach, it is a coordinator – usually an executive or his secretary – who chooses the best time for a meeting from those considered "feasible" by the system.

Grudin (1989) analyzes why these simple meeting scheduling systems fail. He mentions the fact that "free time" in managers' calendars is often not truly free. Their free time may be an initial approach but the real availability has to be checked with the respective secretaries. By contrast, the subordinates have to do the additional work of keeping their own calendars updated for the benefit of those who call the meetings: usually the managers. This asymmetry in the benefits of such a system is a cause of users' frustration and probably the rationale for the system's failure (Robinson, 1991).

An interesting field study on the use of simple meeting scheduling systems is reported by Kincaid *et al.* (1985). They found that only one person out of a group of 40 supported users gave an account of consistent use. Moreover, this user utilized the mail system to set up meetings. When asked why they did not use the meeting scheduling system, users said "they did not perceive the calendar to give an accurate representation of a user's availability".

The users' availability does not seem then to be a trivial issue. On the one hand, we have an implicit claim from executives to dispose of the time of others, which is an ethical issue as identified by Egger and Wagner (1992). On the other, employees, especially those with seniority and/or professional status, may have an opinion on the best use of their company time.

Meeting time coordination

Greif and Sarin (1987) present two meeting scheduling systems: MPCAL and RTCAL. MPCAL allows several group roles. A meeting is scheduled by creating a proposal in a personal calendar. That proposal is written as an unconfirmed meeting into calendars of participants. They can accept, reject or put on hold the request and these responses are returned to the meeting caller's calendar.

RTCAL is a synchronous system having a shared workspace which displays a description of the meeting to be scheduled, and a filtered view of the participants' calendars that shows only blocks of free and busy time. Participants can negotiate with each other over a telephone connection. They can also take turns, controlled by the "chairperson", to enter commands on the shared workspace for browsing the calendar and proposing specific meeting times.

Sugihara *et al.* (1989) view meeting scheduling as an optimization problem called "Timetable Rearrangement" problem. Given a timetable of meetings and an additional meeting to be scheduled, the problem is to rearrange the timetable so that the total number of persons obliged to change their schedules is minimized.

Beard *et al.* (1990) have proposed to consider each person's calendar as a virtual overhead transparency. If the time slots are painted in such a way that the worst slots are the darkest, then stacking the calendars shows the group's availabilities in lightly painted time slots and also the bad meeting times (dark slots).

Sen and Durfee (1991) have approached the subject from a distributed artificial intelligence point of view. They have developed agents that can negotiate over scheduling options on behalf of their associated humans. The scheduling process is initiated by a host, which invites other participants' agents for a meeting at the best (earliest) needed interval. The invitees respond with bids to the host, which tries to find an agreement. If not, it will send new proposals, and so on. Eventually, a common time interval is found and the meeting is scheduled or the meeting is marked as not schedulable. The model global performance is measured by the success ratio in scheduling the total number of meetings during a time period.

Mitchell *et al.* (1994) have a machine learning approach to meeting scheduling. They developed a learning personal assistant. This consists of a software module which learns users' scheduling preferences through routine use, enabling it to give customized scheduling advice to each user. With an accumulated experience of five user-years, the result is that the assistant learns something, but the accuracy of its advice is not high.

Ephrati *et al.* (1994) distinguish two types of meeting scheduling scenarios: "open" and "closed" systems. In open systems, the potential participants in a meeting have no obligation to meet one another. Thus, the individuals themselves determine the feasibility of a meeting. In closed systems, on the contrary, the participants belong to the same organization and have the obligation to attend a meeting, if feasible. In a certain sense, the organization has a certain ownership over the users' time.

Ephrati *et al.* introduce three strategies to schedule meetings: calendar oriented scheduling, meetings oriented scheduling and schedule oriented scheduling. In the first strategy, the users assign "convenience points" to available time slots for a certain meeting, which gets scheduled by the system in the "winning" slot. In the second strategy, the users assign convenience points to combinations of meetings and time slots. In the third strategy, the users express preferences over entire schedules, where each schedule identifies one possible ordering of all future meetings. However, these strategies are manipulable by the users. The authors introduce the Clarke Tax as a method to punish users' manipulable behaviors.

An encouraging study on users' acceptance of scheduling tools has been recently presented by Mosier and Tammaro (1997). The successful use of the meeting maker scheduling product in the case study is attributed by the authors to a variety of causes: very good training and support, graphical quality of the product user interface, provision of a good calendar by the tool and characteristics of the tasks being done by the users.

The latitude model

The complexity of the relations among individuals working for an organization makes it difficult to accept as realistic the closed system hypothesized by Ephrati *et al.* (1994). Despite the fact that employees are paid by their organization, it is obvious they have tastes, preferences, motivations, etc. and in many cases their time cannot be assigned in an arbitrary way by a person or a software system (this does not mean the strategies designed by Ephrati *et al.* would arbitrarily assign schedules. On the contrary, their strategies are intended to provide satisfaction to all users' preferences). This may occur especially in the case of professionals and executives.

It is clear also that the open system hypothesis does not hold either within an organization: people would not refuse an appointment with the boss unless a very reasonable excuse exists.

We propose the following underlying model, which we call the latitude model: an employee must attend most of the meetings, but he may decline some invitations. People usually have a wide variety of reasons to prefer meetings in certain time slots, defer or anticipate meetings, prefer to schedule meetings one next to the other, or have them distributed over the day, etc. Preferences may indeed change over time (e.g. "this week I am feeling sleepy in the mornings".) Such preferences may be important for the person when deciding attendance at meetings. For instance, someone may choose not to attend a certain meeting next Monday because he would spend too much time moving to and from the proposed meeting location (especially on Mondays, his busiest days). Another person may not like to skip his lunch because of a meeting. Many people would prefer not to disclose some of these reasons to others and it is their right to keep them private.

From the latitude model, we propose a computer-based system which would help people to schedule meetings themselves. Unlike the Ephrati *et al.* (1994) approach, this system is not based on computing the best choice derived from the conveniences expressed as users' votes. The following example is interesting to illustrate why we have made this choice. Suppose a group of people meet face-to-face to schedule a long meeting and for some people "next Thursday at 10am" is a good time slot. Suppose, also, that another person has a strong reason to avoid that time slot. If he explains that reason, some people may change their preferences, and perhaps, a more satisfying time slot for the whole group may be agreed on. If a voting system were used in the first place, the possibilities for negotiation or mutual interchange of points of view would have been limited.

On the other hand, the use of a synchronous negotiation system such as RTCAL (Greif and Sarin, 1987) is too demanding on the participants. It seems exaggerated to require they have to meet (same time, at least) in order to schedule a meeting. Thus, a practical scheduling negotiation system should be distributed and asynchronous.

The basic strategy to scheduling which is being proposed is both simpler and more complex than the previous approaches to this subject. The new strategy is simple because it is not expected that the computer system would solve the problem, but rather it should help people to do the problem solving. This approach then is precisely what has been characterized as "computer supported cooperative work" (Schmidt and Bannon, 1992).

The proposed approach is also more complex than previous attempts in the sense that it incorporates people's preferences. In our approach, a time slot is not simply available or not. It may be available depending on a conjunction of factors, but these may remain private if the person chooses this: the rest of the group will only know the fact this person prefers not to have a meeting at this time or alternatively that he has a strong preference for this time slot. The same type of conditions are applied to meeting location, subject and participants.

Conflict resolution and negotiation

It is desirable that if negotiation is to be held, the system should provide the means to make it short. People can propose schedules which are likely to be accepted if they know other people's preferences beforehand. Now, we find again the pervasive privacy vs. awareness dilemma (Hudson and Smith, 1994). How to inform to make meeting scheduling negotiation efficient but at the same time respecting privacy concerns? One answer is to let people tell other people whatever they wish on this subject: a person informs all details he considers meaningful for those who schedule meetings which he might be invited to attend. The incentive to let others know his own restrictions and preferences is that proposals for meetings will be most satisfying to people who give as much information as possible, provided there is good will among all people involved.

It is important that the social environment be positive. To see it, suppose the contrary: there is person A who wants to damage person B. If B provides information about his preferences and restrictions, A may try to schedule meetings exactly at times when it is bad for B. The incentives do not work in that case: a person gets punished because he provides useful information for others. Another example of this undesirable behavior occurs when a boss wants to get rid of a subordinate and uses the information provided by the subordinate to do the same tactics so the victim may wish to move elsewhere inside or outside the organization. This procedure may succeed according to the boss' short-term goals but at the cost of severely damaging the trust within the group.

The case study reported by Orlikowski (1996) on the successful use of Lotus Notes by a group of people also asked for "the need to ensure that those with access (to the database) do not use it to blame, punish, or construct convenient scapegoats" on the basis of the information people have stored in the system. In a previous paper, Orlikowski (1992) presented a case in which Lotus Notes use had limited initial success because the firm's social culture was very competitive, among other reasons.

The situation in which people use the voluntarily stored information to damage its authors shows symptoms of a bad organizational working environment, and this should be promptly improved by management. We assume in what follows that such cases do not occur: a person belonging to a healthy group cannot be penalized for letting other people know his own meeting preferences and restrictions for the benefit of the whole group.

Designing a system to implement the latitude model

A first issue to be settled concerns the identification of the group that the meeting scheduling system is to support. We choose to include in the group all people who may eventually have meetings. Therefore, in a typical organization this large group will include all persons with access to a workstation. This choice allows the negotiation protocol to start only when a meeting instance is to be scheduled.

This choice is similar to the implicit concept behind an electronic message: the actual connection or binding between sender and receiver is only activated when a message is ready to be sent; before that moment, any electronic mail address is eligible to be a receiver. This choice has the advantage of allowing meetings to be scheduled with unanticipated participants and it is useful for meetings of *ad hoc* committees or multi-departmental projects, for example.

As mentioned in the previous sections, scheduling has to occur in an asynchronous distributed fashion: it is too expensive for people to do meeting scheduling at the same time and/or in the same place. Furthermore, people have to express their preferences beforehand in order to simplify the negotiation process. How can these preferences be stated?

One possible way to express preferences is in natural language. However, the subtleties of natural language utterances make it difficult for a computer system to extract information. If the system does not get the information, the support it can give is no more than that provided by electronic mail. Moreover, users must be guided about the type of personal information which will be useful to other people in order to schedule meetings. Therefore, this guidance offers opportunities for simplifying the information extraction as well.

The user interface, then, can have some type of menu-based structure in order to let users specify their guided preferences. The design must have graphical, visual appeal. The system should provide facilities to express various degrees of agreement to accommodate preferences of the type "I do not like this time slot, but if it is the only possible, I would agree", as different from "this time slot is impossible for me". Privacy can be ensured by the system by showing to anyone requesting a meeting simply the time slots painted with different colors, depicting total availability, no availability, preference for not having the meeting at that time slot, etc., but not showing the reasons behind each preference.

A meeting scheduling session would begin then by any user (subsequently identified as the "coordinator") specifying a meeting in terms of: list of invited persons, subject, place, meeting duration. The best time for the meeting is to be negotiated with the participants.

The negotiation should proceed in three stages. The first involves the coordinator using the previously stated preferences. In a certain way, this step is reminiscent of the negotiation among agents suggested by Mitchell *et al.* (1994) but it is much simpler than that process. This stage should shorten the negotiation involving people.

The second stage occurs after an appropriate time for the meeting has been found by the coordinator. He sends a request to each invited person specifying all details. They receive it in their calendars and are required to confirm or decline the invitation.

The final stage is the confirmation of the meeting sent by the coordinator if all invited persons accept the proposed meeting or cancellation if at least one of the invitees declines the invitation. In the latter case, the coordinator can start a new meeting scheduling process. A new meeting may be attempted by excluding the person who refused participation, replacing him with another person or changing some of the meeting characteristics (e.g. place, duration). The proposed meeting can also be definitively discarded.

In this negotiation we can notice the importance of every guest having an updated preferences profile, since otherwise outdated information can mislead the coordinator and force a second negotiation round. It is important also that people confirm or decline meeting invitations. Frequent reminders to people who have not replied yet can be implemented to be issued automatically by the system.

In the case of very tight schedules, it may be impossible to fit a meeting without using a time slot marked by at least one user as not desirable. In this case, the coordinator must have at hand the organizational hierarchy levels of the participants in order to avoid choosing a time slot which is uncomfortable for guests having high hierarchical rank within the group.

GRACE: an example implementation

We have developed a prototype implementation of the latitude model. It is called GRACE and runs on a network of Sun Sparc workstations with SunOs/Xwindows. The objective of this prototype was to show that the model could be implemented as a comfortable work environment for the users.

Each GRACE user has a window on his workstation displaying the current day's list of activities and the current month's calendar (Figure 1). There are also menu buttons to:

- schedule meetings and personal activities,;
- review invitations;
- state personal preferences; and
- browse other users' information.

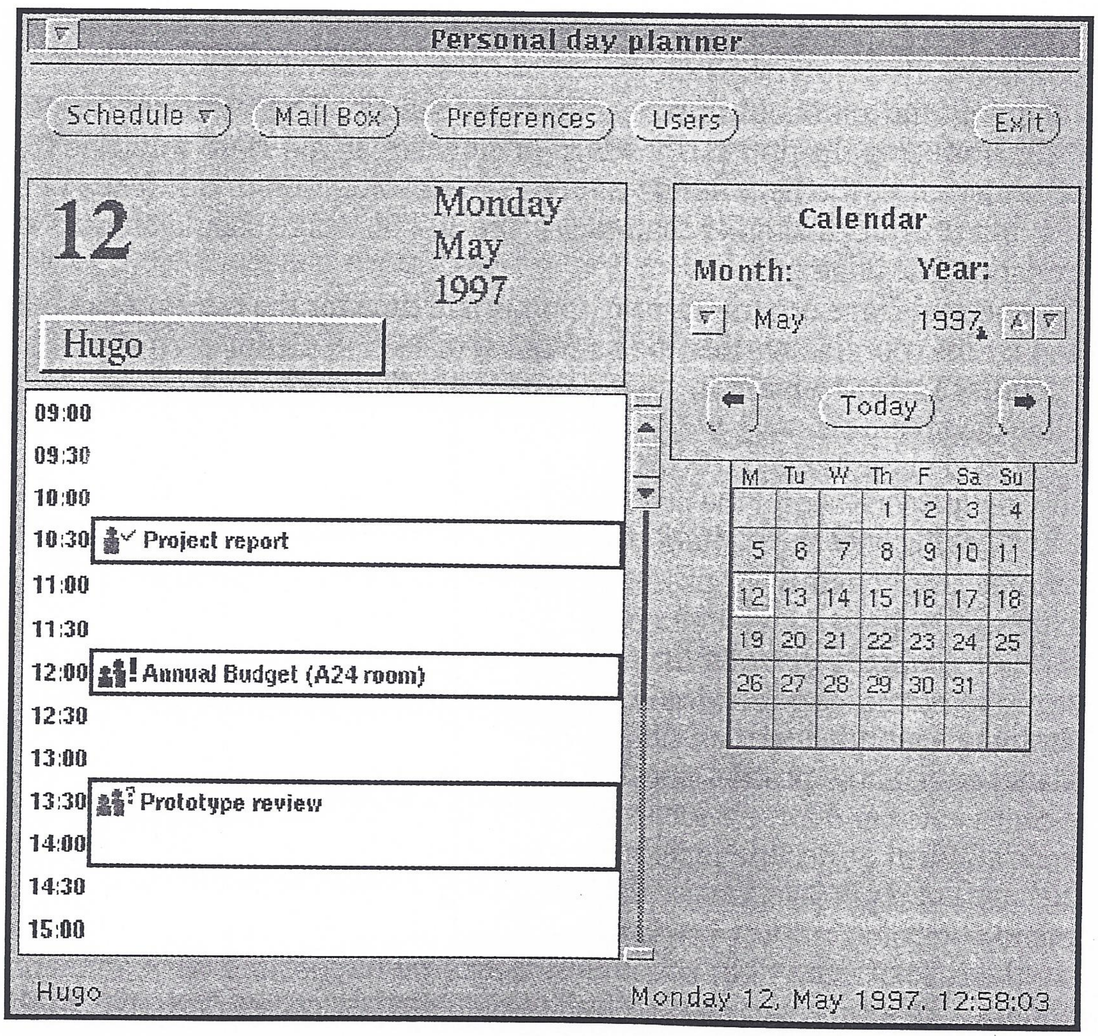

Figure 1
Meetings and activities window

To state his personal preferences, the user is presented with a window in which he can place his weekly preferences for meetings (Figure 2). The week is shown as a matrix of half-hour colored slots. The colors correspond to the preferences and restrictions which can be applied to any of the slots. Three choices are of clear meaning: yes ("I like this slot for any type of meeting"), no ("it is impossible for me to attend any meeting at this time") and indifferent (default: "I do not prefer nor dislike this slot for meetings"). There are also two choices to specify agreement or disagreement provided certain conditions apply: "yes, when..." and "no, when...".

The conditions are implemented as rules which apply over the meeting profile the coordinator is proposing. The rules are specified on a separate window which is opened when the user selects the conditional preferences. This window displays the choices already made by the user ("apply rule on", "status"). Selecting "when" chooses the participant's name, coordinator's name, place or subject about which the rule is to be evaluated. Once stated, the rule is paraphrased by the system in plain English at the bottom of this window. Figure 2 shows an example rule based on the coordinator's name. Rules based on participants' names are similar to this one (this allows the user to specify she likes to attend meetings when X is also invited or when Y is not invited).

If the condition is based on subject or place, the user is prompted to write a string describing the subject or place over which he wants the rule to be applied. The current implementation supports just full match between the subject or place of the meeting in negotiation and the subject or place stated by the user in the rule. As an extension to the present implementation, partial string matching can be easily developed. The current implementation can be practical if there is a conventional way to reference places or subjects, for instance, "A24 room", or "budget".

The week planner window also has the choice of specifying "other preferences". These refer to global conditions. Thus, the user may want to limit the total time dedicated to meetings per week or the number of meetings on a certain subject per week or the number of participants in each meeting. The system provides another window which lets the user set these upper limits.

When a user wants to schedule a meeting, he is provided with a window to specify the guests, another window to specify the subject and place, and a third window to analyze and propose the day, time and duration of the meeting. This last window is shown at the left-hand side of Figure 3. After specifying the date, the window will show with colors, the preferences of the group for all the time slots of that day. Figure 3 depicts the state after the coordinator chose 15:30 as the time for the meeting. It can be seen the system displays a small subwindow showing Luis has a problem, Francisco does not and "myself" is particularly interested to have this meeting at this time (double check sign). If the coordinator wants to see further detail concerning the activities and preferences of the group members during that day, he is

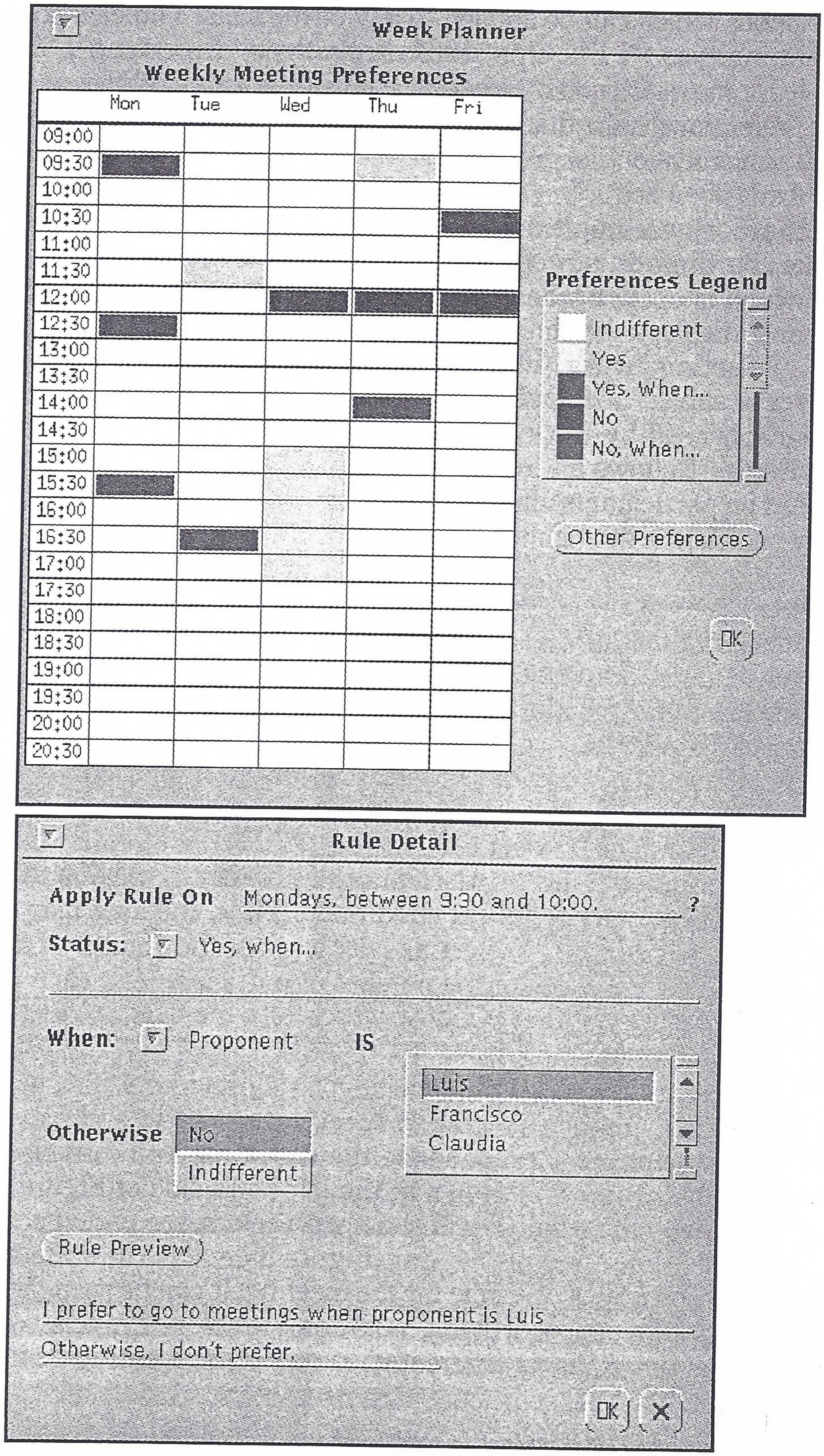

Figure 2
Windows for stating preferences

provided with the additional window shown at the right-hand side of Figure 3. The summary ("day's planner") in the meeting time window shows the system recommendations concerning the proposed meeting. The meaning of the slot colors is the same as the one used in the week planner window (Figure 2). The system makes recommendations based on the meeting profiles, the preference rules and the hierarchical positions of the group members, which are known by the system beforehand.

Once the coordinator has established the data for the proposed meeting, the second stage of the group protocol is done by the system. It sends messages to the mailboxes in the calendars of each of the guests. Finally, each invitee must accept, decline or postpone his decision concerning his attendance. Message sending and reception between coordinator and each guest is controlled by the system.

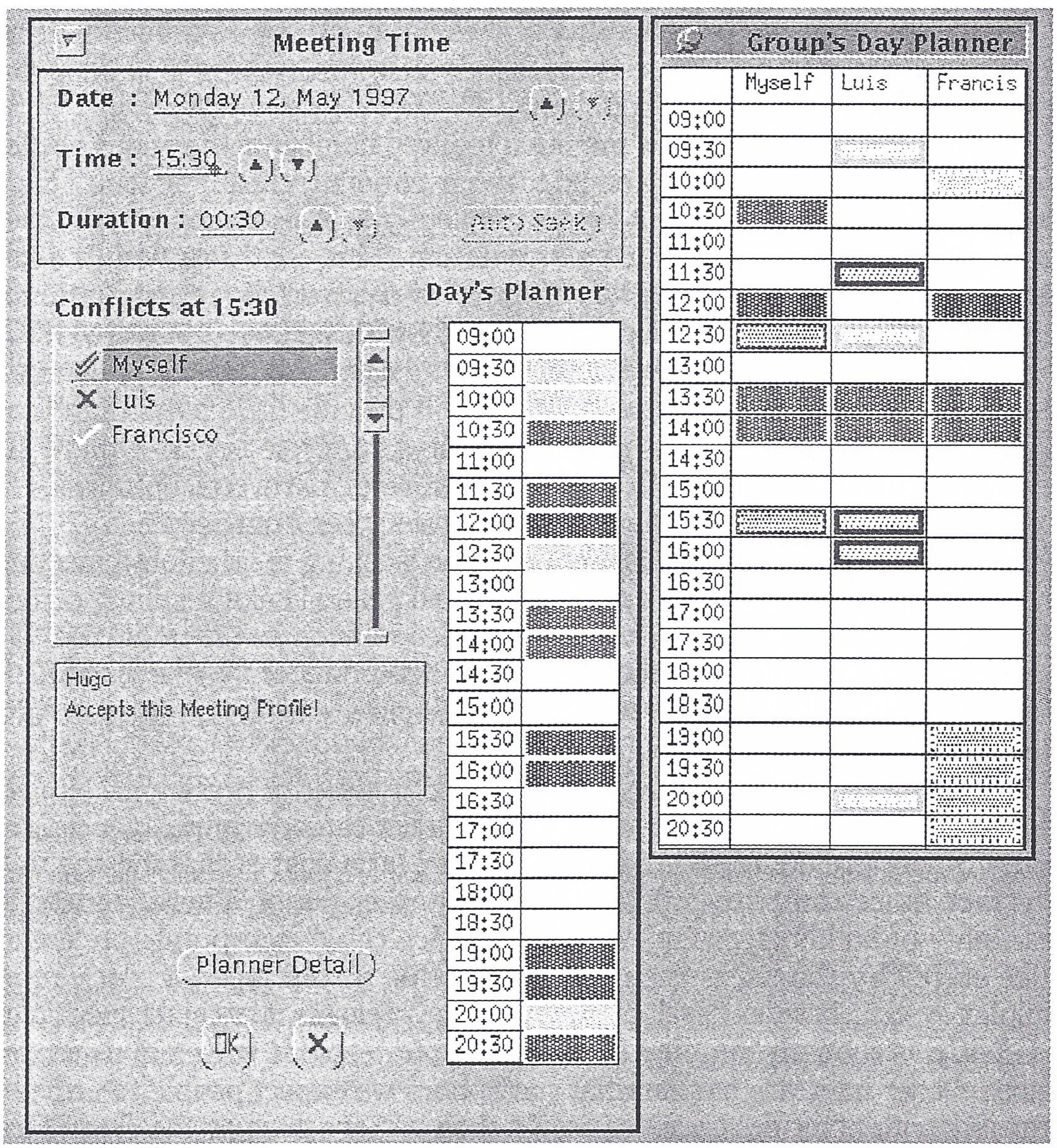

Figure 3
Proposing a meeting

Discussion and conclusions

Our contribution is to show that the latitude model for scheduling meetings is an option for the open or closed strategies. As its name implies, the model accommodates users' availability according to their own preferences and restrictions. Furthermore, such a model can be implemented with simple visual user interfaces.

The latitude model can be implemented with other features to make it satisfy additional requirements as well. For instance, the answers to the coordinator's requests for meetings can be implemented based on speech-acts in a way similar to the conversational model of the coordinator (Flores *et al.,* 1988).

The usefulness of an implementation of the latitude model is greatly brought down if users do not trust the information they provide will be used for the benefit of all. We made the assumption people are not punished in the organization for stating their calendar preferences and restrictions (with some privacy considerations). While this assumption may hold in many organizations, it is also possible that it does not in others. In the worst case, when nobody contributes preferences, the system functioning is similar to a simple scheduling tool with negotiation.

The use of the latitude model implies a reduced number of messages needed to schedule a meeting when compared to a conventional scheduling tool. This is good news for users who do not log on frequently to their computers. All they need will typically be to confirm an invitation. This contrasts with the low acceptability this type of user has of tools in which a longer negotiation is needed to schedule a meeting, as hypothesized by Mosier and Tammaro (1997).

One can expect that some users will declare all details about their preferences and specify many rules while other users will provide little more than their personal appointments. This should not be a surprise because there are individual differences in real life: there are extroverts, introverts, optimists, etc. It is not reasonable then to demand all their preferences from people.

The privacy features may puzzle a user who is trying to schedule a meeting. This may occur because some users have stated general rules disliking certain characteristics of the meeting being scheduled of such as the invitation to a specific guest or the length of the meeting. The coordinator may have to talk to the corresponding guests to try again. Many people are willing to speak or give a hint about their dislikes but will not explicitly write them for anyone to see.

The GRACE implementation has a limited number of preference choices offered to the users. In future versions this number can be arbitrarily increased so that many preferences could be expressed (at a cost of making the user interface more complex or loaded with many options). There is also the possibility of building an implementation which could incorporate preference rules stated by the users themselves, but this would mean a much more complex system, besides developing a language to enable users state such rules.

GRACE is robustly running but we have not tested it with real users yet. Since its user interface has familiar metaphors we expect people to utilize it naturally with other desk accessories on their screens such as the clock. First users may start by just stating their personal activities and perhaps incorporating preferences in the calendar before trying to schedule meetings.

References

Beard, D., Palanlappan, M., Humm, A., Banks, D., Nair, A. and Shan, Y. (1990) "A visual calendar for scheduling group meetings", in Bikson, T.K. (Ed.), *Proceedings of CSCW'90, Los Angeles, CA*, ACM, New York, pp 279-90.

Egger, E. and Wagner, I. (1992), "Time-management: a case for CSCW", in Turner, J. and Kraut, R. (Eds), *Proceedings of CSCW'92,* Toronto, Canada, pp. 249-56.

Ephrati, E., Zlotkin, G. and Rosenschein, J. (1994), "Meet your destiny: a non-manipulable meeting scheduler", *Proceedings of CSCW'94, Chapel Hill, NC,* ACM, New York, October, pp. 359-71.

Flores, F., Graves, M., Hartfield, B. and Winograd, T. (1988), "Computer systems and the design of organizational interaction", *ACM Trans. on Office Information Systems*, Vol. 6 No. 2, pp. 153-72.

Greif, I. and Sarin, S., (1987), "Data sharing in group work", *ACM Transactions on Office Information Systems*, Vol. 5 No. 2, pp. 187-211.

Grudin, J., (1989), "Why CSCW applications fail: problems in design and evaluation", *Office: Technology and People*, Vol. 4 No. 3, pp. 245-64.

Hudson, S. and Smith, I. (1994), "Techniques for addressing fundamental privacy and disruption tradeoffs in awareness support systems", in Ackerman, M.S. (Ed.), *Proceedings of CSCW'94, Boston*, MA, ACM, New York, pp. 248-57.

Kincaid, C., Dupont, P. and Kaye, A. (1985), "Electronic calendars in the office: an assessment of user needs and current technology", *ACM Transactions on Office Information Systems,* Vol. 3 No. 1, pp. 89-102.

Mitchell, T., Caruana, R., Freitag, D., McDermott, J. and Zabowski, D. (1994), "Experience with a learning personal assistant", *Communications of the ACM*, Vol. 37 No. 7, pp. 81-91.

Mosier, J. and Tammaro, S. (1997), "When are group scheduling tools useful?", *Computer Supported Cooperative Work: The Journal of Collaborative Computing,* Vol. 6 No. 1, pp. 53-70.

Orlikowski, W. (1992), "Learning from NOTES: organizational issues in groupware implementation", in Turner, J. and Kraut, R. (Eds), *Proceedings of CSCW'92, Toronto, Canada,* New York, pp. 362-69.

Orlikowski, W. (1996), "Evolving with notes: organizational change around groupware technology", in Ciborra, C.U. (Ed.), *Groupware and Teamwork,* John Wiley & Sons, pp. 23-59.

Robinson, M. (1991), "Computer supported co-operative work: cases and concepts", in Hendricks, P.R.H. (Ed.), *Proceedings of Groupware 91*, Engineering Research Centre, Utrecht, The Netherlands, pp. 59-75.

Schmidt, K. and Bannon, L. (1992), "Taking CSCW seriously", *Computer Supported Cooperative Work (CSCW),* Vol. 1 No. 1-2, pp. 7-40.

Sen, S. and Durfee, E. (1991), "A formal study of distributed meeting scheduling: preliminary results", *Proceedings of the Conference on Organizational Computing Systems, Atlanta, GA*, ACM, New York, pp. 55-68.

Sugihara, K., Kikuno, T. and Yoshida, N. (1989), "A meeting scheduler for office automation", *IEEE Transactions on Software Engineering*, Vol. 15 No. 10, pp. 1141-6.