INSTITUTO DE COMPUTAÇÃO
UNIVERSIDADE ESTADUAL DE CAMPINAS

New approaches for groupware scheduling

Fábio Silva *Rogério Drummond*

Technical Report - IC-07-20 - Relatório Técnico

June - 2007 - Junho

New approaches for groupware scheduling

Fábio Silva
fabio.silva@students.ic.unicamp.br

Rogério Drummond
rog@ic.unicamp.br

Institute of Computing
University of Campinas
13084-971 Campinas, SP - Brazil

Abstract

Groupware scheduling is an essential and continuous activity. Complexities on conciliating schedules of several people make proper time management a challenge. It involves sophisticated communication mechanisms to support information sharing and scheduling negotiation among users of diverse systems. Also, scheduling on group demands application support to deal with over-constrained schedules and rescheduling, but such needs have not yet been fulfilled. Based on a review of past approaches for this problem, their successes and failures, new approaches for groupware scheduling are presented. The development of calendaring and scheduling protocols is reviewed, as well as the application of promising techniques on the fields of optimization, artificial intelligence and mixed-initiative interfaces. Towards a new generation of more intelligent and powerful groupware scheduling tools that will increase productivity by minimizing coordination effort, a multidisciplinary application of knowledge from these areas is proposed.

1 Introduction

Conciliating schedules is one of the most important, repeating, and time-consuming tasks on daily routine, especially on offices. Work assignments require knowledge about the availability of each member in a team. Scheduling meetings involves coordinating all attendees towards a common feasible period of time. These negotiation processes can still demand reservation of other resources, such as room and equipment.

Schedule management is subject to a number of constraints, that can take several forms. Deadlines, calendar conflicts, priorities, preferences, cancellations and reschedulings, to name a few, make this conciliation even more complicated. Groupware scheduling is concerned on offering good tools that deal with such complex environment.

Whereas calendaring and scheduling software have long been classified as personal information management (PIM) systems, we agree with Erickson [20]. He argues that information systems for which intensive information sharing in a group context is the rule should be studied by a broader perspective, that of group information management (GIM).

The use of calendar information is inherently a decision-making process – scheduling – which changes how people organize their time. This characteristic is what distinguishes groupware scheduling applications from other information systems. That demands the study of aspects that go beyond typical technological issues of data sharing, such as distributed storage, search and retrieval mechanisms. There are individual and social aspects [44] to be taken under consideration. Since scheduling is also part of the universe of human subjective choices, adherence to work practices, high usability levels and social acceptance of these systems comes out to be as important as their features.

Developing appropriate groupware scheduling systems is still an open problem for which there is no satisfactory solution. Currently, for instance, if a group meeting needs to be scheduled, most of the work is still done via a series of phone calls and emails on which each person availability and preferences are collected by a person that gets in charge of this coordination. The adoption of proprietary solutions has somehow changed this scenario when all attendees are from the same organization, but with increasing complexity of social network relationships, this is hardly the case. Fortunately, emerging protocols are about to overcome the barrier of organizational bounds [16].

Nevertheless, while the widespread use of these communication standards will guarantee interoperability, there is a series of problems that they do not solve. For example, if the schedules of a group meeting attendees offer no common free timeslot, some cancellation will have to be done. How to best rearrange these schedules? The lack of systems capable to assist the user to answer questions like this indicates a gap between users needs and existing systems skills, which must be fulfilled.

In this article, we survey past approaches for groupware scheduling, analyze the reasons that prevented them to be totally successful on fulfilling user needs, and describe new directions for research and software development on the field that can mitigate future frustration.

On next section, we summarize relevant concepts to this study. The analysis of up-to-date approaches, and the presentation of new approaches indicated by their successes and failures are done, respectively, on sections 3 and 4. The last section of the article is devoted to some conclusions that should be subject of consideration by further development and research projects.

2 Reasoning about groupware scheduling

Groupware scheduling is one of the most relevant applications of computer-supported cooperative work (CSCW). As a decision-making process, it encompasses some distributed systems issues, such as voting protocols and concurrence. The comprehension of this dynamics is fundamental. Although most research on this area is specially concerned with meeting scheduling use cases, their results are applicable to other forms of groupware scheduling, such as work assignment, as well.

2.1 Distributed meeting scheduling formalized

Sen and Durfee [51] present a formal study on distributed meeting scheduling that offers an excellent starting point to discuss groupware scheduling issues.

They outline a general negotiation protocol which describes how attendees of a group event converge into a meeting time:

1. the organizer (also known as host or initiator) sends a meeting time proposal.

2. the attendees (also known as invitees or participants) respond to the organizer.

3. if there is no agreement on the meeting time, go back to 1.

Each iteration is called a round. The efficiency of a negotiation is measured in terms of the number of rounds needed to reach consensus, and on the size of the messages exchanged between the organizer and the attendees.

Based on this protocol, three types of strategies were studied:

- *announcement strategies*: when a meeting is announced, whether the organizer offers only one meeting time (the best strategy), or n meeting time options (the good strategy).

- *bidding strategies*: when the organizer receive responses, whether the attendees simply accept or deny the proposal (the yes-no strategy), or propose other meeting time options (the alternatives strategy).

- *commitment strategies*: when an attendee or organizer is involved on simultaneous negotiations, whether meeting times suggested by others are tentatively blocked in their schedules, so that no other meetings can be scheduled in these timeslots (the committed strategy), or not (the non-committed strategy).

The first two strategies are concerned with communication. They balance demands for privacy (which means exchanging less information) with demands for quickly converging on meeting times (which can be sped up by exchanging more information).

The good announcement strategy and the alternatives bidding strategy yield to faster convergence, specially when the schedules of the attendees are dense and the meeting duration is considerable. In other words, whenever people are willing to share more information, groupware scheduling systems can take advantage of this to make the negotiation process more efficient.

We believe that, although both strategies incur in a greater communication cost, since the messages convey more information, the increment on message sizes is not significant so as to represent network overload. As long as communication resources are not a bottleneck, and this is true for typical situations, the number of rounds becomes the only representative measure of a negotiation efficiency.

The last strategy, commitment, is relevant in the case of multiple concurrent scheduling negotiations. New groupware scheduling applications will make negotiation process more dynamic, and this will become the usual situation. Two or more schedulings in course can eventually compete for the same free time until the meeting is confirmed. When the commitment strategy is taken, it makes schedules more dense by locking resources. This reduces meeting time options for other negotiations, making them longer and more difficult, but it also avoids a meeting time under consideration to be taken by others before its completion.

These interdependencies among negotiations must be wisely treated by future systems. They should indicate to the user the most appropriate strategy for each situation. For example, the committed strategy may be used for high priority meetings, and not for those with low priority.

2.2 Rescheduling issues

A similar trade-off arises when over-constrained schedules demand rescheduling. The problem of when to reschedule was investigated by Modi and Veloso [37]. Cancelling a meeting in favor of another is a key decision, because a rescheduling can further imply in cascading disruption throughout a chain of users connected by meetings in common.

Their work identified that the use of previously fixed strategies, either deciding by always or never rescheduling an existing meeting so as to open room for a new one, leads to sub-optimal schedules or infeasibilities. Thus, groupware scheduling tools should implement intelligent mechanisms to help the user make the best choices.

We see rescheduling as part of normal scheduling workflow, and not as an exception. Unpredicted changes do occur: they must be expected and better supported by groupware scheduling systems. Otherwise, the users are, again, compelled to use demanding, out-of-band mechanisms, like telephone and email, to solve these situations.

3 A review on the approaches for groupware scheduling

Computer-aided scheduling is by no means a new idea [49]. On the late 1960s and 1970s, with the discovery of important results on complexity theory, and the realization that most scheduling problems were NP-complete [22], the research initiatives were concentrated on designing algorithms to efficiently solve them. First attempts to computerize scheduling procedures were made with a focus on very specific planning activities, like aircraft scheduling [25] and university timetabling [6].

3.1 Primary focus on the study of user and group interfaces

It was in the 1980s, with the popularization of microcomputers on offices, that initial studies about the use of calendar systems on this environment were taken. At that time, the increasing concern on usability issues motivated the analysis of psychological factors related to the computerization of scheduling.

The study conducted by Kelley and Chapanis [28] has shown that individual preferences on the use of personal calendars could determine the adoption of computer-based calendaring and scheduling systems. Based on that, Kincaid et al. [29] have first assessed a series of mandatory and desirable requirements that these systems should attend. Demands for calendar management flexibility, with the use of graphical interfaces, multiple views (daily, weekly, monthly), reminders, booking of resources and automatic scheduling, among other facilities, heavily influenced the development of applications from then on, and became the basis for contemporary tools.

Several office automators were proposed. Greif and Sarin [24] presented the MPCAL and RTCAL prototypes, scheduling systems that assumed a centralized environment and the existence of a shared database. This database coordinated all scheduling steps: from free-time search to meeting proposals in negotiation rounds.

Ehrlich [18] pointed out some considerations with respect to the use of systems like these: maintaining electronic versions of calendars is worth only if most people use them, and unauthorized scheduling of an appeearently free time can motivate total rejection from users. Grudin [26] additionally concludes that "electronic calendars are not electronic versions of paper calendars", but communication artifacts, what influences the way how people use them. That contributes to make consistent evaluation of CSCW a non-trivial task.

3.2 First-generation systems

Only on the late 1980s, with the introduction of that known as the first PIM system on the market, Lotus Agenda, the idea of using the computer as a time organizer was brought to desktop users out of universities and technology companies. Agenda was a DOS application that allowed users to take time-related notes.

On the early 1990s, Lotus replaced Agenda by Lotus Organizer, which had a graphic interface that used the paper-based organizer metaphor. Meeting Maker was launched for Macintosh, and Microsoft developed Schedule+, a Windows application. These systems presented more powerful direct-manipulation user interfaces that used calendar-based metaphors and marked initial popularization of calendaring software. Later on, Schedule+ was replaced by Microsoft Outlook, an integrated organizer that incorporated its features along with email, contact management, note taking, and other features. Gradually, Outlook took market leadership from Organizer.

As long as the use of these applications on corporate environments became common, software vendors incorporated groupware scheduling on them. These proprietary solutions enabled the transition from personal to group-aware time management. While this generation of systems had the merit to enable calendaring and scheduling automation and increase productivity within organizational bounds, they did not allow information sharing among different platforms due to the lack of communication standards.

As shown on figure 1 [42], the integration of these desktop systems into a group environment was made by specific servers that implemented proprietary communication protocols. For instance, Microsoft Schedule+ and Outlook would integrate only with the use of Microsoft Exchange Server, and so on.

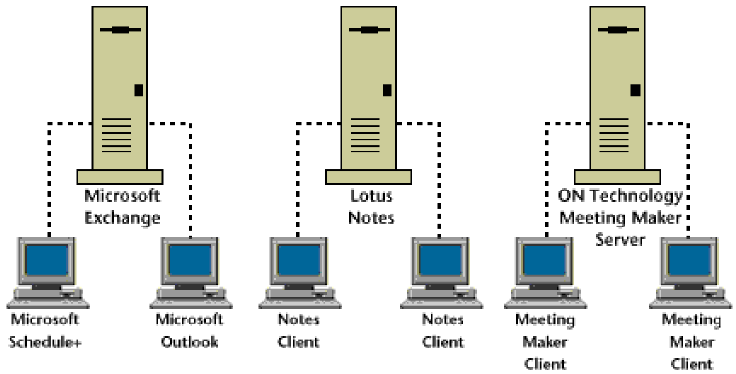

Figure 1: Groupware calendaring and scheduling via proprietary solutions

3.3 Communication protocols and second-generation systems

[1]In 1996, major software vendors agreed on the need of non-proprietary standards. That resulted on the creation of a Calendaring and Scheduling Working Group (CalSch), in the Internet Engineering Task Force (IETF)[2], dedicated to the establishment of open standards for calendaring and scheduling on the Internet [41].

Figure 2 [42] depicts the scenario that was aimed by the CalSch initiative: communication among a broad range of calendaring and scheduling applications enabled by standards that assured interoperability.

Based on previous work from the Versit Consortium on the vCalendar format for calendaring and scheduling data, the CalSch group developed and published in 1998 the following specifications [34]:

- The Internet Calendaring and Scheduling Core Object Specification (iCalendar), described by RFC 2445 [13], is a data model that defines how to textually represent calendar information.
- The iCalendar Transport-Independent Interoperability Protocol (iTIP), described by RFC 2446 [54], that establishes an interoperable standard for the execution of calendaring and scheduling operations using iCalendar objects.
- The iCalendar Message-Based Interoperability Protocol (iMIP), described by RFC 2447 [12], which is an implementation of iTIP over email.

3.3.1 iCalendar

The iCalendar specification defines four main calendar components: events, to-dos, free-busy time and journal entries. All calendar components are described with the use of pa-

[1]For a more throughout review of the history of calendar interoperability, the reader is referred to the paper by Dusseault and Whitehead [16], on which this section is based

[2]The IETF is an Internet standards body that promoted the specification of important protocols, such as HTTP

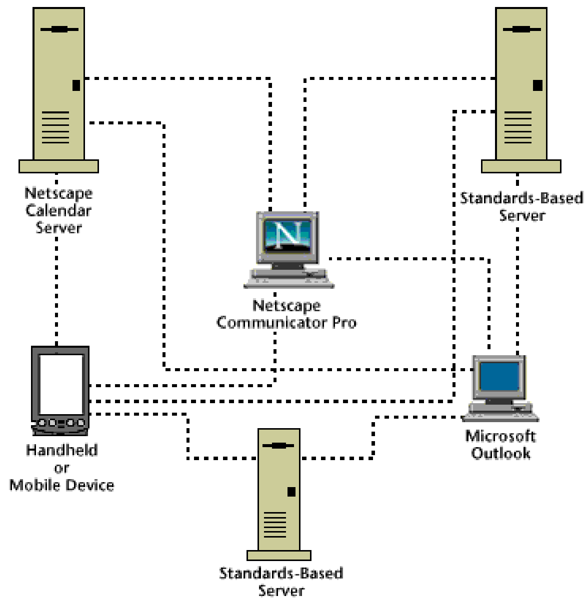

Figure 2: Heterogeneous systems integration foreseen by the CalSch initiative

rameters and attributes that convey information. For instance, events have a "description" parameter that is a textual representation of their purpose.

Events are the main entity on calendars. They represent all kinds of scheduled tasks, such as activities, meetings, appointments and anniversaries. The most important attributes of an event are its start and end times, attendees, and location, but other associated information, like its priority, can be specified as well. iCalendar also has a powerful mechanism that allows the description of recurring events which repeat with a certain frequency.

To-dos are used to represent pending tasks and their status. The most relevant parameter of a to-do is its due-date.

The free-busy time component allows the expression of available and non-available periods of time. These components are used to convey non-scheduled timeslots that can be used to guide proposals prior to scheduling negotiation.

Journal entries are the least used calendar components and do not have a relevant role on scheduling. They are intended to be used solely to relate time-dependent pieces of information (e.g., diary notes).

3.3.2 iTIP

The iTIP protocol does not define a specific transport protocol for exchanging iCalendar data. It just conceptually describes how calendaring and scheduling operations are executed with iCalendar objects in terms of methods. We present the basic behavior of these methods by explaining how they act on event components:

- publish – announces an event.

- request – starts a negotiation process by inviting the attendees of an event.
- reply – accepts or declines an event proposal.
- add – adds an instance to an existing recurring event.
- cancel – cancels an event, or an instance of a recurring event.
- counter – counterproposes a different event description during negotiation.
- declinecounter – declines a counterproposal.

iTIP has a messaging-sequence mechanism. As, depending on the transport protocol, scheduling messages may arrive out-of-order, it guarantees consistency by obsoleting all previous messages referent to a same event negotiation when a new one arrives. This is enforced by the use of unique identifiers and timestamp marks on the messages.

3.3.3 iMIP

The iMIP protocol is an email binding for iTIP. Whereas it has enabled the exchange of calendar and scheduling information on the iCalendar format via email transport, it was not broadly adopted by the most used applications. iMIP also suffers from email latency issues not desirable on the transmission of sensitive time information. This and other limitations on the use of email architecture to accomplish calendaring and scheduling motivated a huge development effort towards a real-time transport protocol for iTIP.

3.3.4 A real-time binding for iTIP

Before being closed, the CalSch group has also worked for years on a totally new, "on-the-wire" transport protocol for iTIP – the Calendar Access Protocol (CAP) – that eventually became published as the RFC 4324 [46]. Due to CAP complexity, it was abandoned.

Egen [17] highlights some practical reasons that make the establishment of open protocols for calendaring and scheduling so difficult: unlike other applications, such as email, it is not possible to develop a protocol that incorporates the best practices of existing implementations, because there are major design choices that heavily influence its use. In addition to this, there is a number of non-trivial technical issues that must be addressed, such as timezones, daylight saving times, interpretation of recurring events and search for free and busy time.

On that interval, other alternatives for calendar sharing were adopted. Apple iCal has successfully used the Web Distributed Authoring and Versioning (WebDAV) protocol to do so: iCalendar events were mapped to HTTP resources and calendars were modeled as WebDAV collections. Soon, other second-generation applications, like Mozilla Calendar, started using WebDAV as a way to promote open calendaring.

The vacuum left by the end of the CalSch group is being fulfilled by the Calendaring and Scheduling Consortium (CalConnect), an organization founded on 2004 with the objective of stimulate the development and adoption of open calendaring and scheduling standards.

CalConnect joins software developers organized on technical committees to debate protocol issues and promote standards following the IETF standards process.

With CalConnect support, standards for the use of WebDAV to enable calendaring and scheduling on the Internet are on discussion. The Calendaring Extensions to WebDAV (CalDAV) protocol was developed.

The emerging CalDAV protocol is the promise to enable calendar sharing and exchange of scheduling messages in a standard way. It shall make possible calendaring and scheduling among several devices by extending the WebDAV protocol for use with the popular iCalendar data format, and has been adopted by major software vendors (e.g., Apple, Google, Mozilla, Oracle, and Yahoo). Figure 3 [16] illustrates how the use of these extensions will rapidly enable interoperability among heterogeneous devices that already make use of the HTTP "stack" of protocols.

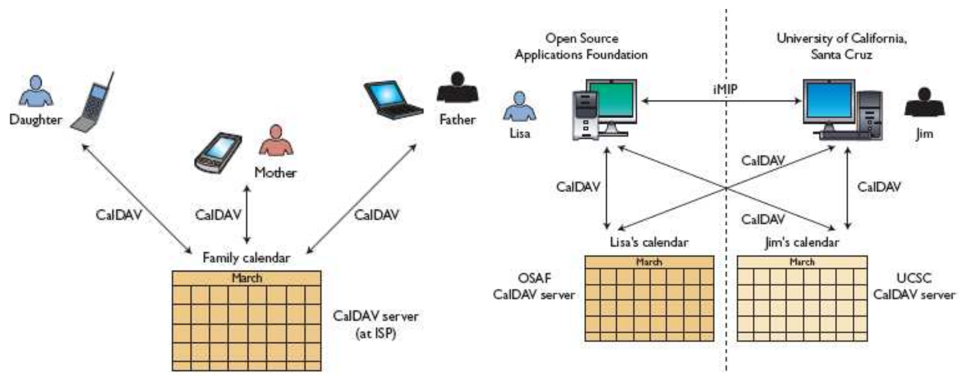

Figure 3: Integration among different devices and applications with CalDAV

CalDAV is a client-to-server protocol. Its calendar-access feature enables calendar sharing by extending WebDAV capabilities, and has already been published as the RFC 4791 [10]. The specification of a calendar-schedule extension for CalDAV is in progress [11]. It uses scheduling inbox and outbox collections to keep track of sent proposals and received invitations, what makes possible multiple application access and delegation use cases on which an user manages a calendar in behalf of another (e.g., a secretary).

CalConnect also organizes interoperability testing events to check specification adherence and implementation issues, not only for CalDAV but also of current specifications. With the IETF Calendaring and Scheduling Standards Simplification (Calsify) working group, revision and simplification of these specifications is in course. Internet drafts intended to become the RFC "bis" versions of iCalendar [14], iTIP [8] and iMIP [35] are being discussed. These new RFCs will solve specification problems that prevent plentiful interoperability and will clarify procedures for protocol extensions.

3.4 Protocols weaknesses

iCalendar and iTIP have a serious limitation: they do not enable the adoption of the good negotiation strategy and the use of more than one alternative proposal during scheduling negotiation. There is no way to (counter-)propose a set of possible start times for events. That inflexibility can also represent a huge communication cost. In the worst case, if only the nth event proposal sent by the organizer is acceptable by all attendees, offering one event description per proposal leads to n negotiation rounds. Conversely, in the best case, if the organizer had proposed n alternative event descriptions, only one negotiation round would have been enough.

Additionally, Sayers and Letsinger [48] discuss some weaknesses of the ontology represented by iCalendar: it uses a "flat" structure to describe events, which does not favor incomplete knowledge representation. While this is not a big problem for publishing calendar data, it is a fundamental requirement during scheduling negotiation. They argue that a typical scheduling negotiation initiates with a request similar to "I'd like to arrange a meeting with you in Cuppertino on Thursday morning, or in Palo Alto on Friday afternoon", and that a reasonable answer would be "I could meet at any time on Thursday, but would prefer sometime between 10 and 11".

On the CoolAgent ontology they propose, an event is a combination of instances composed of three dimensions: people, place and time. These entities can be flexibility organized in a hierarchical recursive structure that allows disjunctive and conjunctive relationships (see figure 4 [47]).

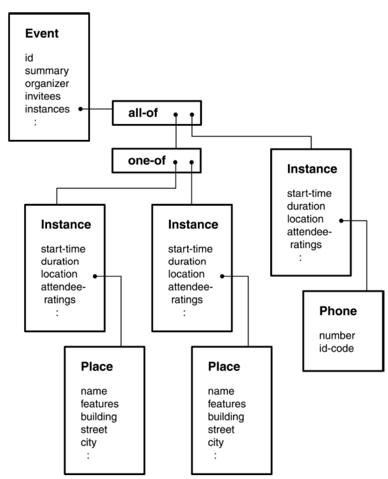

Figure 4: Flexible hierarchical structure of the CoolAgent ontology

iCalendar is somehow flexible on representing the attendees of an event. It is possible, for example, to attach an attribute to each attendee parameter defining if the participation of that attendee is required or optional. However, as mentioned, it is not possible to have several possible start times associated with the same event description. The problem of representing locations as entities, and not as event attributes, is about to be solved with the venue component proposal [43].

3.5 Limitations of second-generation systems

Second-generation systems are characterized by a strong focus on usability. Nowadays, users have many choices among popular desktop applications, like Microsoft Outlook, Apple iCal – which implements the transparency metaphor [2] (figure 5) –, Lotus Notes, Oracle Calendar, and Mozilla Sunbird. Web-based solutions, such as Yahoo Calendar and Google Calendar, have been able to mimic desktop systems interaction features thanks to the advancement of AJAX[3] programming techniques.

Figure 5: Direct manipulation interface of Apple iCal

Most of these systems support the iCalendar data format and are backwards-compatible with the earlier vCalendar format, still very used in mobile devices. Many have achieved, in some degree, interoperability on calendar sharing by using WebDAV, and now, are towards the adoption of CalDAV. HTML- and XML-based solutions have also been used to publish calendar data.

However, real-time scheduling is still supported only by proprietary solutions and application-specific connectors. Even so, effervescence on standards development – namely, the caldav-schedule feature – indicates that solutions for these basic communication issues are in course, what opens space for innovation on groupware scheduling.

As long as CalDAV becomes successful and enables transparent calendaring and scheduling data exchange among heterogeneous applications, platforms, and devices, more sophisticated approaches will be demanded.

3.6 Scheduling as an optimization problem

Parallelly to the popularization of desktop calendaring and scheduling applications, the idea of applying optimization techniques to build more efficient schedules has become usual.

This concept has originated a group of systems oriented to optimal resource allocation.

[3]AJAX (Asynchronous JavaScript and XML) is a web development technique used for creating interactive web applications

These systems were classified by Pinedo and Chao [45] in: commercial systems, application-specific systems and academic prototypes.

Commercial systems are not projected for a specific user, but adjust to a large user base. However, they usually focus a particular aspect of scheduling, facing it as a resource allocation and task assignment problem. The main entities of these applications are not events, but projects that contain phases and tasks to be executed. These tasks can have precedence relationships, and have a demand for resources, whose use must be managed. Microsoft Project is the most popular tool on this category.

Whereas the time of people can be mapped to resources, project management scheduling systems are appropriate for mid and long-term planning. They do not fit well for groupware scheduling on a daily basis (e.g., for negotiating a meeting scheduling). While managers and planners can find these tools specially useful, they hardly could be adapted for use by "average users" of desktop applications, who would need to learn project management concepts and adapt their work practices. For instance, on Microsoft Project and its counterparts, the use of time is typically tracked with the use of Gantt charts, as opposed to the calendar metaphor present on traditional calendaring and scheduling systems.

Application-specific systems are extremely elaborated, and designed for very specific scheduling activities. These systems require an elevated degree of capacitation from their users, and are used for solving scheduling problems whose constraints and optimality criteria are well-defined, such as manufacture production planning.

Academic prototypes, in turn, improve and develop algorithmic techniques that have an unexploited potential to influence the conception of broadly used systems. Previous work has focused optimization strategies for very well-defined scheduling problems, like personnel rostering, school and university timetabling and bus and train scheduling. The Practice and Theory of Automated Timetabling (PATAT) Conferences[4], for instance, have yielded an unquestionable contribution for the comprehension and treatment of real-world large-scale scheduling problems. Nevertheless, research on the optimization field has, mostly, privileged modeling and solving strategies to generate scheduling solutions which better satisfy the complex constraints that appear in these problems and make them hard to solve.

However, groupware scheduling is not a well-defined problem like those. The only clear (or hard) constraints are: (1) no coincident events should be scheduled for the same person; and (2) event due dates should be respected. But there are uncountable implicit (or soft) constraints – like time preferences – that the user aims to satisfy while scheduling. Regardless of being possible, specifying these conditions explicitly in terms of scheduling constraints would be too demanding, what goes against the need of simple and effective tools. Additionally, on groupware scheduling, optimality criteria is not clear either. Since events are scheduled as long as they are proposed and have their times negotiated, optimization objectives such as those used on application-specific schedulers (e.g., minimum makespan) do not make sense.

Sugihara et al. [55], who developed a meeting-scheduler for office automation, note that the real scheduling problem in groupware is not to build new schedules, but to change the existing ones. It is impossible to reallocate all events at each new scheduling. In this

[4]http://www.asap.cs.nott.ac.uk/patat/patat-index.shtml

scenario, optimization can be used to avoid the disruption costs inflicted by a sequence of reschedulings. They define a general Timetable Rearrangement (TR) optimization problem whose objective is to minimize the number of changes needed to schedule an incoming event if there is no free time for it. An algorithm for solving TR is also presented.

When there is more than one possible time for event scheduling, in turn, the optimization criteria should maximize the user satisfaction with the final schedule. In such case, scheduling preferences should be represented, which demands the use of artificial intelligence mechanisms.

3.7 Scheduling as an artificial intelligence problem

Computer-aided scheduling can also be supported by artificial intelligence (AI) techniques [57].

For instance, several AI methods have been used to suggest scheduling times based on past scheduling activities, what is useful when several timeslots are available. The value of machine learning in scheduling has been long perceived [1]. Kozierok and Maes [32] earlier proposed a learning interface agent for scheduling meetings that would suggest meeting times.

The Calendar APprentice (CAP) described by Mitchell et al. [36] has a broader scope: it tries to predict several other event characteristics, such as location and attendees. Blum [5] showed that, if the number of predicted characteristics is criteriously selected, the accuracy of algorithm predictions increases.

Mueller [40] proposes SensiCal, a calendar with common sense. This system would have general knowledge about the world, and would be able to, for example, warn a user about inviting a vegetarian to a steak house (see figure 6).

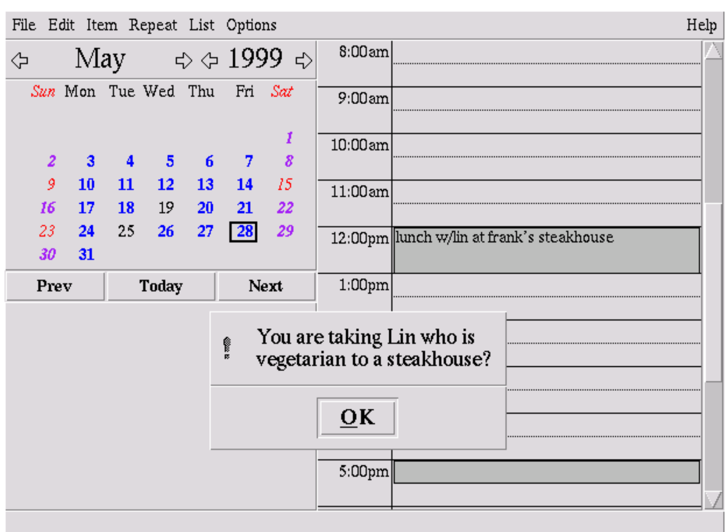

Figure 6: SensiCal common sense knowledge base in action

To overcome the problems brought when rescheduling is needed, Modi and Veloso [37] suggest to assess the difficulty of scheduling with other people, so as to determine the scheduling rearrangement that would be more suitable. Shintani et al. [52] propose a preference revision mechanism that adapts local user preferences so that meeting proposals are more easily accepted by others.

Again, in AI field, the construction of *de facto* usable scheduling mechanisms has not been of much interest as well: learning and inference techniques based on domain specific information – which is not the case for groupware scheduling – have been emphasized. While most proposals could be extremely useful, some are too demanding for real-world applications, requiring data that users are not willing to inform and whose inference can be difficult.

Crawford and Veloso [7] offer a realistic approach to personal scheduling without assuming that complex, extensive training data is available. They observe that, unless the user is always negotiating with the same people, it is difficult to build a representative knowledge base from which different scheduling negotiation behaviors can be recommended. They propose a very simple but powerful learning model that infers user time scheduling preferences only based on previously scheduling meetings. The mechanism models time of day preferences (e.g., whether the use prefers to have meetings in the morning or afternoon) and back-to-back preferences (i.e., whether the user prefers scheduling events in "blocks" or interleaved with free timeslots). It becomes effective rapidly, with few learning examples, and dynamically adapts to preference changes. A method like this could be used in a real implementation to suggest scheduling times for non-overconstrained timetables, and be adapted to help a TR algorithm to choose schedule rearrangements considering user preferences.

In opposition to the complexity of some proposals, the effectively of simple solutions must be highlighted. On Google Calendar, for instance, if the user types "lunch with Alice 5pm", the system is wise enough to create an event whose description is "lunch with Alice", for which the start time is 5 pm. Even an apparently simple mechanism like this can be very demanding, if complexities such as parsing different languages are accounted.

3.7.1 Contemporary projects

Currently there are two research projects concentrated on the development of groupware scheduling systems heavily-based on AI techniques. The CMRADAR project [38] aims to build a personal assistant agent for calendar management that increases user productivity by assisting and autonomously performing usual tasks. The CALO (Cognitive Assistant that Learns and Organizes) project has a broader scope: act not only on scheduling assistance, which is the role of the PTIME (Personalized Time Manager) module [3], but also on all other aspects of office work, including project management and document organization.

3.8 Understanding the limitations of up-to-date approaches

Research based on the use experience of first- and second-generation systems brings some comprehension about their limitations.

Mosier and Tammaro [39] highlight the partial-use problem, that occurs when some people involved on a meeting negotiation do not use groupware scheduling tools. The use of mechanisms to incentive widespread adoption of these systems was already a concern for Ehrlich [18].

Palen [44] observes that, if individual and social issues are not properly addressed, they can lead to unexpected use of scheduling systems. For example, if total freedom for scheduling free time is supposed due to a system default setting, users tend to employ "defensive scheduling", by blocking *a priori* periods of time not intended to be available for any purpose. She also argues that individual-centered development approaches have problems because groupware work cannot be evaluated in a laboratory.

Blandford and Green [4] identify a series of work practices that are not currently supported by groupware scheduling tools, and derive a requirements list that should be considered for implementation on new systems.

3.9 Insightful approaches

Several proposals for groupware scheduling, including those from optimization and artificial intelligence fields, have tried to solve the problem by acting in the behalf of the users (e.g., [19, 50]). However, it is becoming clear that these methods should not be used standalone, but in cooperation with human capabilities, especially on such ill-defined, subjective domain. This is to be done with interactive (and intelligent) interfaces that enable the user to guide the core algorithms.

These intermediary solutions (e.g., [23]), also known as hybrid and mixed-initiative scheduling, conceal user experience and cognitive capabilities with computational resources for data manipulation and solution search.

A system can make use of two mechanisms to capture user information: inquiring and observation. Systems that adopt the first strategy dialogue with the user through the interface, posing direct questions to gather input. Observer systems use machine learning algorithms to build a model of the user behavior.

3.9.1 LookOut

LookOut, a Microsoft Outlook add-in proposed by Horvitz [27], uses both techniques. The system has a mixed-initiative interface, in which the user and the system are actors, and whose actions are supported by the use of artificial intelligence. LookOut analyses the contents of emails exchanged between users that are negotiating a meeting. A Bayesian network mechanism is triggered to decide, based on the text of each received email, on the user schedule, and on his past interactions with the system, which is the best time to schedule the meeting and whether it would be convenient to interrupt him to suggest this. If so, LookOut asks the user if the suggested scheduling should be executed. Figure 7 [27] shows the interaction sequence from message receiving to scheduling conclusion.

With respect to the adoption of mixed-initiative interfaces, specially those that use animated characters, Swartz [56] considers some trade-offs between the gain on communication comprehension and the possibility of rejection by the user.

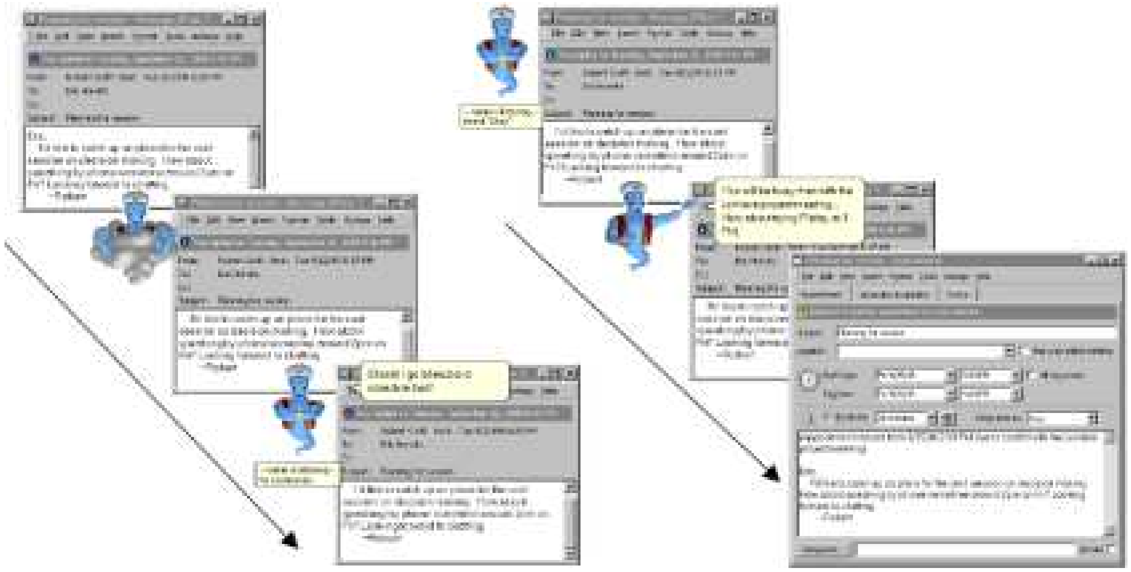

Figure 7: Scheduling on LookOut: the user receives an email and the system suggests a scheduling action, which is executed after user approval

3.9.2 The Human Guided Search framework

Klau et al. [31] developed the Human Guided Search (HuGS) framework, a solution that actively involves the user on optimization processes. As the users take part on the search for optimal solutions, they are able to input knowledge of real-world constraints, and can better understand and justify system outcomes. Moreover, the user expertise can guide optimum search on a solution space and help the system to escape from local minima, which results on faster convergence to better solutions.

This is done with the use of visual metaphors that allow the user to control the optimization process, sometimes observing the core algorithms working, sometimes modifying directly the solution. A mechanism of mobilities (*low*, *medium*, *high*) operated by the user controls which variables can modified by the system. At each iteration, a variable with high mobility is chosen for a modification that cannot involve a low mobility variable. This model of mobilities has a lot of similarities with scheduling negotiation levels of commitment, on which events could be classified in confirmed (low mobility), tentatively scheduled (medium mobility) and in negotiation (high mobility). An user could assign a lower mobility to a high priority event during its negotiation, and reserve suggested times by indicating to the system that the committed strategy should be taken.

HuGS is implemented in a Java toolkit for interactive optimization, and was applied to various complex combinatorial problems, like map labelling and graph layout. A tabu search implementation [30] was used to build an interactive application for jobshop scheduling [33] (see figure 8). With adaptations, the HuGS toolkit could be used to properly implement an interactive TR algorithm, for example.

User Hints, a similar framework proposed by Nascimento and Eades [15], looks for

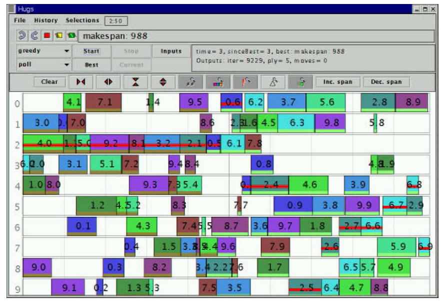

Figure 8: HuGS interactive application for jobshop scheduling

helping the user not only to get optimal solutions, but also to refine the problem definition itself during the search process.

4 New approaches

Based on the experiences of last decades, and on the factors that have contributed positively and negatively to the current status of groupware scheduling, we propose new approaches that should be considered for the development of future solutions. These approaches derive from lessons learned upon past work and on the perception that other directions will be demanded. Groupware systems must support the increasingly complexity of scheduling negotiation scenarios which users have to deal with.

We live on a post-industrial, knowledge-based, service-based society where one of the key factors for productivity is information. How it flows, its fitness and quality. The individual planning is more dynamic in response to new information. The quality and speed of individual results can be enhanced with proper information and interaction with other experts.

With the ease of today communication technologies, it is expected that an individual schedule will be fine grained with more small tasks many constituted of distance conferencing. The work time can be roughly divided into “expertise work” (EW) and “non-expertise work” (NW) which comprise all the book-keeping and administrative work. An individual should maximize the relation between EW and NW.

The knowledge-based enterprise is often valued by the sum of its expertises, but its productivity is more related to the proper interactions among the individual components and how information flows. Efficient global scheduling mechanisms can accelerate the flow of information keeping the individuals efficiently exercising their expertises.

Frequently meetings will be necessary for productivity, but the effort on scheduling those meetings must be kept to a minimum. More so, once detected a need for an appointment, it should be scheduled quickly to a proper date. Most of the work should be done by the scheduling system, otherwise we would be increasing the non-expertise work.

4.1 Facing groupware scheduling as an interdisciplinary challenge

At the extent of our knowledge, no previous approach has combined all the following elements to offer a groupware scheduling solution: optimization and artificial intelligence "solving" techniques, mixed-initiative interaction with these core solving algorithms, and support to standard calendaring and scheduling protocols for interoperable communication. In our opinion, these four topics will guide the development of next-generation systems.

We strongly believe that no research area should be privileged over the others. In fact, we identify on the most successful approaches so far a huge concern about not concentrating on a single view of groupware scheduling. The focus must be on modelling and proposing solutions for scheduling problems from the user point of view, and on developing applications using results from diverse disciplines.

4.2 Introducing new concepts

Demands for more powerful groupware scheduling systems will, certainly, require the development of new concepts that incorporate and model their requirements.

From a diagnostic of usual situations that are not adequately expressed on current groupware scheduling systems, some such concepts are now introduced.

4.2.1 Liquid tasks

Commonly, people are in charge of mid- and long-term assignments that consume a large amount of time. Normally, they have a due date that often is an external hard requirement. The estimate of total time required for completion is another optional parameter. It can be a gross estimate initially and get more precise with time. Due to the nature of the task, there can be requirements like minimum and maximum duration of each slot dedicated to it. Mechanical tasks that require little intellectual effort and no special concentration can be executed even in small slots since may require very small start-up time, while tasks that require more concentration and profound reasoning should have longer minimum duration with no interruptions.

We call this kind of activity a *Liquid Task* (LT), because it "fills" feasible free slots until its completion. It keeps filling future slots as it overflows previous ones that satisfy its constraints. Since it is a long-term assignment, it is unrealistic to pre-allocate slots as hard appointments, since unpredicted cancellation will certainly occur. Our proposal is that the scheduling system will be continuously providing soft allocations of slots satisfying the constraints. But a liquid task allocation can give away to a higher priority event, it can be pushed further to the future (conditions allowing).

Some LTs will have special requirements like need of equipment, room, etc. This is an open issue that applies to any event, and in its general form can not be solved. Nevertheless, should be mentioned, and restricted cases can be handled.

The scheduler will do the best allocation given the requirements for the LTs. The user should carefully specify and review it periodically. As shown later, most of the work can be handled by the optimizing engine.

The LTs became even more appealing when considered as group tasks. As seen before, the interaction increases the productivity individually and in a group environment. Naturally some of the LTs can also be a group effort. There can be lists of required and optional attendees, and number of minimum optional attendees. Since scheduling a single group task is complicate and better be mostly done automatically, the same is true for a group LT. What should be assured is the realization of interaction with negligible cost on the part of the user.

The tendency is towards most of the groups tasks be remote events. New enhanced professional applications are becoming available, like Google Docs[5], that allows multi-user editing of documents and spreadsheets, and complete distributed collaboration environments like Adobe Acrobat Connect Pro[6].

Without leaving the working desk, one can alternate between individual and group activities. The scheduling should not be an imposition, but a recommendation based on the hard commitments and on the analysis and optimization of the LTs. The group LT that have been scheduled should be considered as usual events, since they are hard commitments that other people depend on.

Conferencing opens a new way of working that we have to get used to. Besides the common uses of continuous conversation it can be used as if the participants are working on their on but sharing a big table. When needed the collaborators can be reached right away. Better than in a big table, if anyone receives a phone call or any other distracting interruption, it will not disturb the others. This kind of conferencing is not as intense, but can be very effective. When one has an enquire, the proper expert should be at hand. In most cases these are tasks they must do any way, we are just synchronizing for better results.

Examples of such collaboration are parallel software development. The client developer may be writing the first application that is a representative of future uses of a new API and serves as a validation and refinement for the proposed API. Client and server developers should interact frequently to progress. A long-term group LT it a perfect fit. When client and server developers are teams, the scheduling issues can hardly be managed in an ad-hoc manner. An automatic solution is high desirable.

Writing a book or a complex report can benefit from group LT schema. Most of the work is done alone and without interruption. Some times, however, one can spend hours on a couple paragraphs. And some external input can bring some new insight. Just reading what the other is writing could inspire new ideas.

One could argue that a liquid task is a to-do, since most popular applications allow to-dos be registered and have their progress status updated by the user. However, a LT has important characteristics that ordinary to-dos do not have:

- minimum granularity: the minimum requirements of contiguous time for the task, which also include the start-up time.
- maximum granularity: the limit of contiguous time that turns out to be useful for the

[5]http://docs.google.com/support (accessed June 2007)
[6]http://www.adobe.com/products/acrobatconnectpro/productinfo/features (accessed June 2007)

task.

- remaining time to complete: a dynamic characteristic which is updated as time goes on. It tends to become more accurate as the due date approximates.

The scheduling application should keep track of changes over time. Since it has the history of LT allocations into normal events, advanced forecasts of critical situations can be provided to the user.

Moreover, liquid tasks can be the link between personal group-aware schedulers and scheduling and planning systems such as those that work under the project management perspective: a project phase or task could be mapped to a LT, and as this LT is scheduled into events, the project management tool can automatically gather information both to re-plan incomplete tasks based on work already done and to generate planned-versus-realized reports.

4.2.2 Imprecise events

In fact, the notion of liquid tasks is contained within a more general class of events, which we call *imprecise events*. An imprecise event is an outline of a non-scheduled event (or set of events), for which no fixed start and end times are defined, that may inherits all properties of a liquid task.

Given that each individual can honor certain periods of the day to different kinds of activities, in an imprecise event this can be stated as well. Additionally, each of these periods can be ranked, so as to represent the suitability or likelihood of having them scheduled in fact. A one hour long business meeting taking place on Monday between 8 am and 11 am - preferred - or on Thursday 2 pm is an example of an imprecise event.

The ability to express an event imprecisely is specially important during scheduling negotiation, when the attendees have to agree about an event description that initially is not exact. Inexact event specifications are suitable for refinement during the negotiation process, until consensus is reached and a complete event description is established.

4.2.3 Alternative events

Some situations may, additionally, require multiple alternative event descriptions (imprecise or not). These *alternative events* can be used during scheduling negotiation as a way to communicate multiple options that the attendees can evaluate and choose among. It should also be possible to rank these alternative events, allowing the users preferences to become part of the negotiation.

The imprecise event example could be extended and be represented by two imprecise event descriptions that are alternative: a 1 hour long business meeting taking place at office A on Monday between 8 am and 11 am - preferred -, or at office B on Thursday 2 pm or Friday after 4:30 pm.

As shown, multiple alternative event descriptions and available periods of time can help a group of attendees to faster converge into a description that is acceptable for all [51]. Whereas in most situations that means to reach consensus about an event time that satisfies a common feasible timeslot, other event properties can be under negotiation as well (e.g., location, duration, required and non-required attendees).

4.3 Pursuing standardization of extended protocols

While there is now comprehension on the need of using optimization and artificial intelligence via collaborative user interfaces, these approaches have almost ignored the need of standard, broadly adopted calendaring protocols to convey meeting negotiation data. Practical solutions that deal with the limitations of existing protocols are urged.

iCalendar-related standards represent more than a decade of work upon the development of protocols aiming interoperable communication between different systems. As long as new approaches for groupware scheduling are taken, they will, inevitably, lead to more protocol requirements to support additional features. When designing new systems, extensions for these protocols may be considered. If implemented, their standardization via open specifications is the only way to guarantee that interoperability is kept.

4.3.1 Imprecise and alternative events for iCalendar and iTIP

iCalendar does not provide a way to describe an event imprecisely or to represent multiple event alternatives. This can not be done by using several iTIP messages either.

We have submitted to the IETF Internet-Drafts directory a proposal of extensions for iCalendar and iTIP [53] that overcome these limitations. This specification draft is under discussion by members of the Calsify working group. It is intended to bring the power of more flexible event expression into the standards, while being realistic on the treatment of compatibility issues with current applications.

In practical terms, an imprecise event is represented by an event component with additional properties to which several available periods of time are attached. For that, the free-busy component, which is already part of iCalendar, and the proposed availability component [9], a more flexible way to represent recurring intervals of free time, are used. Alternative events are collections of event components present on an iTIP scheduling message that convey both exact and imprecise event descriptions.

These extensions also specify new properties and parameters that allow the timeslots of imprecise events and each of the alternative events to be ranked.

4.4 Pursuing third-generation systems by developing realistic tools

Finally, we believe that the effective implementation of new approaches is complete only with the development of usable tools that can seamlessly integrate into groupware environments.

A third-generation of systems, more intelligent and more powerful, shall arise from the use of new concepts resulting of multidisciplinary approach. In last instance, the consolidation of this third-generation will occur as long as such applications become broadly available.

To follow these paths, we are developing iScheduler, a proof-of-concept tool that incorporates these promising approaches to better support groupware scheduling.

4.4.1 iScheduler: an interdisciplinary approach

iScheduler aims for an *i*ntelligent, *i*nteractive, *i*nteroperable groupware scheduler. Its purpose is to exercise, in a practical manner, the effectiveness of the presented approaches for the development of new groupware scheduling applications. We are specially interested on demonstrating the applicability of optimization and artificial intelligence mechanisms to better support user activities interactively, while preserving adherence to standard communication protocols.

iScheduler is being developed in Java. Its architecture can be seen on figure 9.

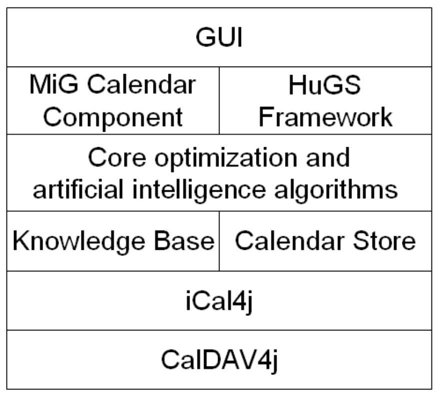

Figure 9: iScheduler architecture

iCal4j[7], a library for marshalling and unmarshalling iCalendar objects, is being extended to support imprecise and alternative events. With the help of these new protocol features, iScheduler will become scheduling negotiations more effective. Access to CalDAV servers is to be done with the use of the CalDAV4j API[8].

Negotiations will also take advantage on an artificial intelligence core with an algorithm to learn time scheduling preferences. Over-constrained situations are going to be treated with the use of TR algorithms, interactively guided by the user via the HuGS toolkit.

A high-quality graphical interface will be provided by the MiG Calendar Component[9]. It is under study an extension of the calendar component to implement the availability bars described by Faulring and Myers [21]. They are a clean solution to display simultaneously the availability of several attendees.

5 Conclusions

With the announced success of CalDAV on establishing itself as the standard protocol for calendar sharing and scheduling, due to its huge support from commercial vendors, existing

---

[7]http://ical4j.sourceforge.net/

[8]http://chandlerproject.org/Projects/CalDAV4jHome

[9]http://www.migcalendar.com. We are grateful to MiG InfoCom AB, that gently granted us an academic license of the MiG Calendar Component for this work

interoperability problems will be gone soon. Users tend to gain more freedom to choose their calendaring and scheduling client, even inside corporations.

The next step to be taken by application vendors will be towards more powerful and intelligent applications, able to wisely automate repetitive tasks and to reduce the user cognitive workload. While some additional features will not depend on changing communication semantics, others may require protocol enhancements to effectively work on groupware.

We agree that the process of designing protocols may not follow the pace of a fast-changing software market, since it receives input from a wide range of interested developers. But we also identify on the lack of standards-compliance one of the main reasons for the failure of a number of – otherwise outstanding – solutions developed so far. Incompatible proprietary forms of information sharing should not be adopted anymore. Under the auspices of an organization like CalConnect, demanded standardization processes can be conducted faster for the benefit of all.

An integrated approach implies on considering the relationships between these underlying communication protocols and the several mechanisms that will be on the core of next generation groupware scheduling systems. Past experiences demonstrate that focusing a unique research area implies on future problems to the adoption of proposed solutions in practice, due to the mutual influences of different areas of knowledge over each other. Optimization, artificial intelligence and interaction techniques can not provide a definitive solution for groupware scheduling by themselves, but do offer a breakthrough perspective if applied jointly.

We advocate for a relevant research path: future work on groupware scheduling should not consider these disciplines apart, but pay special attention to issues that arise when trying to conciliate them. Effort must be employed not only on innovations development, but also on addressing the implications of their adoption on an heterogeneous environment.

References

[1] Haldun Aytug, Siddhartha Bhattacharyya, Gary J. Kochlet, and Jane L. Snowdon. A review of machine learning in scheduling. *IEEE Trans. Engineering Management*, 41(2):165–171, May 1994.

[2] D. Beard, M. Palaniappan, A. Humm, D. Banks, A. Nair, and Y.-P. Shan. A visual calendar for scheduling group meetings. In *Proc. of the Conf. on Compuer-Supported Cooperative Work*, pages 279–291. ACM Press, October 1990.

[3] Pauline Berry, Bart Peintner, Ken Conley, Melinda T. Gervasio, Tomás E. Uribe, and Neil Yorke-Smith. Deploying a personalized time management agent. In Hideyuki Nakashima, Michael P. Wellman, Gerhard Weiss, and Peter Stone, editors, *5th International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, pages 1564–1571, Hakodate, Japan, 2006. ACM.

[4] Ann Blandford and Thomas R. G. Green. Group and individual time management tools: What you get is not what you need. *Personal and Ubiquitous Computing*, 5(4):213–230, 2001.

[5] Avrim Blum. Empirical support for winnow and weighted-majority algorithms: Results on a calendar scheduling domain. *Machine Learning*, 26(1):5–23, 1997.

[6] Andrew C. Brewer. Interactive scheduling system. *IBM Systems Journal*, 10(1):62–79, 1971.

[7] Elisabeth Crawford and Manuela Veloso. Learning dynamic preferences in multi-agent meeting scheduling. In Andrzej Skowron, Jean-Paul A. Barthès, Lakhmi C. Jain, Ron Sun, Pierre Morizet-Mahoudeaux, Jiming Liu, and Ning Zhong, editors, *Proc. of the 2005 IEEE/WIC/ACM Intern. Conf. on Intelligent Agent Technology*, pages 487–490, Compiegne, France, September 2005. IEEE Computer Society.

[8] C. Daboo. iCalendar Transport-Independent Interoperability Protocol (iTIP). draft-ietf-calsify-2446bis, 2007. IETF Internet draft. (Work in progress).

[9] C. Daboo and B. Desruisseaux. Calendaring availability. draft-daboo-calendar-availability, 2007. IETF Internet draft. (Work in progress).

[10] C. Daboo, B. Desruisseaux, and L. Dusseault. Calendaring extensions to WebDAV (CalDAV), 2007. RFC 4791.

[11] C. Daboo, B. Desruisseaux, and L. Dusseault. Scheduling extensions to CalDAV. draft-desruisseaux-caldav-sched, 2007. IETF Internet draft. (Work in progress).

[12] F. Dawson, S. Mansour, and S. Silverberg. iCalendar Message-Based Interoperability Protocol (iMIP), 1998. RFC 2447.

[13] F. Dawson and D. Stenerson. Internet Calendaring and Scheduling Core Object Specification (iCalendar), 1998. RFC 2445.

[14] B. Desruisseaux. Internet calendaring and scheduling core object specification (iCalendar). draft-ietf-calsify-rfc2445bis, 2007. IETF Internet draft. (Work in progress).

[15] Hugo A. D. do Nascimento and Peter Eades. User hints: a framework for interactive optimization. *Future Generation Comp. Syst*, 21(7):1171–1191, July 2005.

[16] Lisa Dusseault and James Whitehead. Open calendar sharing and scheduling with calDAV. *IEEE Internet Comput.*, 9(2):81–89, March 2005.

[17] P. Egen. An Introduction to Calendaring Standards - How it all started, 2005. The Calendaring and Scheduling Consortium.

[18] Susan F. Ehrlich. Strategies for encouraging successful adoption of office communication systems. *ACM Trans. on Office Inf. Syst*, 5(4):340–357, 1987.

[19] Eithan Ephrati, Gilad Zlotkin, and Jeffrey S. Rosenschein. Meet your destiny: A non–manipulable meeting scheduler. In *Proceedings of ACM CSCW'94 Conference on Computer-Supported Cooperative Work*, pages 359–371, 1994.

[20] Thomas Erickson. From PIM to GIM: personal information management in group contexts. *Commun. ACM*, 49(1):74–75, January 2006.

[21] Andrew Faulring and Brad A. Myers. Availability bars for calendar scheduling. In Gary M. Olson and Robin Jeffries, editors, *Extended Abstracts Proceedings of the 2006 Conference on Human Factors in Computing Systems*, pages 760–765, Montréal, Québec, Canada, April 2006. ACM.

[22] Michael R. Garey and David S. Johnson. *Computers and Intractability*. W. H. Freeman and Company, San Francisco, 1979.

[23] H. Goltz and D. Matzke. Combined interactive and automatic timetabling, 1999.

[24] Irene Greif and Sunil K. Sarin. Data sharing in group work. *ACM Trans. on Office Inf. Sys.*, 5(2):187–211, April 1987.

[25] Joseph E. Grimes. Scheduling to reduce conflict in meetings. *Commun. ACM*, 13(6):351–352, June 1970.

[26] Jonathan Grudin. Why CSCW applications fail: problems in the design and evaluation of organization of organizational interfaces. In *CSCW '88: Proceedings of the 1988 ACM conference on Computer-supported cooperative work*, pages 85–93, New York, NY, USA, 1988. ACM Press.

[27] Eric Horvitz. Principles of mixed-initiative user interfaces. In *Proc. ACM SIGCHI Conf. Human Factors in Computing Systems*, pages 159–166, Pittsburgh, PA, 1999. ACM Press.

[28] J.F. Kelley and A Chapanis. How professional persons keep their calendars: Implications for computerization. *Journal of Occupational Psychology*, 55:241–256, 1982.

[29] Christine M. Kincaid, Pierre B. Dupont, and A. R. Kaye. Electronic calendars in the office: an assessment of user needs and current technology. *ACM Trans. Inf. Syst.*, 3(1):89–102, 1985.

[30] Gunnar W. Klau, Neal Lesh, Joe Marks, and Michael Mitzenmacher. Human-guided tabu search. In *Proc. 18th National Conf. Artificial Intelligence, AAAI*, pages 41–47, Menlo Parc, CA, USA, July 2002. AAAI Press.

[31] Gunnar W. Klau, Neal Lesh, Joe Marks, Michael Mitzenmacher, and Guy T. Schafer. The HuGS platform: A toolkit for interactive optimization. Technical Report TR2002-08, Mitsubishi Electric Research Laboratories, June 2002.

[32] Robyn Kozierok and Pattie Maes. A learning interface agent for scheduling meetings. In Wayne D. Gray, William E. Hefley, and Dianne Murray, editors, *Proceedings of the International Workshop on Intelligent User Interfaces*, pages 81–88, New York, NY, USA, January 1992. ACM Press.

[33] Neal Lesh, Leonardo B. Lopes, Joe Marks, Michael Mitzenmacher, and Guy T. Schafer. Human-guided search for jobshop scheduling. Technical Report TR2002-43, Mitsubishi Electric Research Laboratories, 2003.

[34] B. Mahoney, G. Babics, and A. Taler. Guide to Internet Calendaring, 2002. RFC 3283.

[35] A. Melnikov. iCalendar Message-Based Interoperability Protocol (iMIP). draft-ietf-calsify-rfc2447bis, 2007. IETF Internet draft. (Work in progress).

[36] Tom Mitchell, Rich Caruana, Dayne Freitag, John McDermott, and David Zabowski. Experience with a learning personal assistant. *Commun. ACM*, 37(7):81–91, July 1994.

[37] P. Modi and M. Veloso. Multiagent meeting scheduling with rescheduling. In *The 5h International Workshop on Distributed Constraint Reasoning*, pages 198–201, Toronto, Canada, 2004.

[38] Pragnesh Jay Modi, Manuela M. Veloso, Stephen F. Smith, and Jean Oh. CMRadar: A personal assistant agent for calendar management. In Paolo Bresciani, Paolo Giorgini, Brian Henderson-Sellers, Graham Low, and Michael Winikoff, editors, *AOIS*, volume 3508 of *Lecture Notes in Computer Science*, pages 169–181. Springer, 2004.

[39] Jane N. Mosier and Susan G. Tammaro. When are group scheduling tools useful? *Computer Supported Cooperative Work*, 6(1):53–70, March 1997.

[40] Erik T. Mueller. A calendar with common sense. In *IUI '00: Proceedings of the 5th international conference on Intelligent user interfaces*, pages 198–201, New York, NY, USA, 2000. ACM Press.

[41] Netscape. More than 20 companies join Netscape to help define open standard for Internet calendaring and scheduling, 1996. Netscape Communications. (Press release).

[42] Netscape. Overview of calendaring and scheduling standards, 2003. Netscape White Papers.

[43] Charles Norris and Jeff McCullough. Internet calendaring and scheduling venue component specification. draft-norris-ical-venue, 2007. IETF Internet draft. (Work in progress).

[44] Leysia Palen. Social, individual and technological issues for groupware calendar systems. In *CHI*, pages 17–24, Pittsburgh, Pennsylvania, United States, 1999. ACM Press.

[45] M. Pinedo and X. Chao. *Operations Scheduling with Applications in Manufacturing and Services*. Irwin/McGraw-Hill, Boston, Mass, 1999.

[46] Doug Royer, George Babics, and Steve Mansour. Calendar access protocol (CAP), 2005. RFC 4324.

[47] Craig Sayers and Reed Letsinger. The CoolAgent ontology: A language for publishing and scheduling events. Technical Report HPL-2001-194, Hewlett Packard Laboratories, 2001.

[48] Craig Sayers and Reed Letsinger. An ontology for publishing and scheduling events and the lessons learned in developing it. Technical Report HPL-2002-162, Hewlett Packard Laboratories, 2002.

[49] Claire K. Schultz, Alan Brooks, and Phyllis Schwartz. Scheduling meetings with a computer. *Commun. ACM*, 7(9):534–541, September 1964.

[50] Sandip Sen. Developing an automated distributed meeting scheduler. *IEEE Expert*, 12(4):41–45, 1997.

[51] Sandip Sen and Edmund H. Durfee. A formal study of distributed meeting scheduling. *Group Decision and Negotiation*, 7(3):265–289, May 1998.

[52] Toramatsu Shintani, Takayuki Ito, and Katia P. Sycara. Multiple negotiations among agents for a distributed meeting scheduler. In *ICMAS*, pages 435–436. IEEE Computer Society, 2000.

[53] F. Silva and R. Drummond. Imprecise and alternative events for iCalendar and iTIP. draft-silva-events, 2007. IETF Internet draft. (Work in progress).

[54] S. Silverberg, S. Mansour, F. Dawson, and R. Hopson. iCalendar Transport-Independent Interoperability Protocol (iTIP) - Scheduling Events, BusyTime, To-dos and Journal Entries. RFC 2446, Internet Engineering Task Force, December 1998.

[55] K. Sugihara, T. Kikuno, and N. Yoshida. A meeting scheduler for office automation. *IEEE Trans. Softw. Eng.*, 15(10):1141–1146, October 1989.

[56] L. Swartz. Why people hate the paperclip: Labels, appearance, behavior, and social responses to user interface agents, 2003. Honors Thesis for Symbolic Systems Program, Stanford University.

[57] M. Zweben and M. S. Fox. *Intelligent Scheduling*. Morgan Kaufmann, 1994.