MIT/LCS/TM-218

"COOPERATIVE OFFICE WORK, TELECONFERENCING AND CALENDAR MANAGEMENT: A COLLECTION OF PAPERS

Irene Greif

May 1982

Cooperative Office Work, Teleconferencing and Calendar Management: A Collection of Papers

Irene Greif

May 6, 1982

Abstract

This technical memo consists of a collection of papers that have been presented at conferences. They all present results of research in the "Multi-person Informational Work" project in the Office Automation Group.

Keywords: office automation; calendar management; desk-to-desk conferencing

This research was supported by the Advanced Research Projects Agency of the Department of Defense and was monitored by the Office of Naval Research under Contract Number N00014-75-C-0661.

Table of Contents

Computer Support for Cooperative Office Activities

INTRODUCTION

Office automation research and development has focused on support for a single worker, with the major part of the software of a computer-based office workstation designed to support *single-person* activities. *Multi-person* activities typically are supported only by the inclusion of a communication facility in the form of a mail and message subsystem. In our research we have taken an alternative approach based on information about working relationships and their communications protocols.

In this paper we describe cooperative work in two application areas -- calendar management and document writing. For each we suggest some roles that individuals might assume in typical working relationships. Examples of communications needs indicate how a database of working relationship information can be used in cooperative activities. We conclude with some comments on workstation design based on communication through a (shared) database subsystem.

WORKING GROUP SITUATIONS

People in offices often work together in planning, document writing, design, project management, etc. Some office activities involve more than one person *simultaneously* acting, perhaps sharing information, perhaps cooperating to make a decision or to solve a problem. They may co-locate for a meeting, converse on the telephone or attend a teleconference.

Other activities do not require simultaneity. For example, a proposed schedule may be sent back and forth among a group of people until all agree on a conference time. In these cases, the interaction among members of the group constitutes a working *meeting over time.*

Computer-based workstations can support both kinds of multi-person activities -- *meetings over time* as well as *simultaneous* meetings. We have built prototypes systems for both in our own laboratory.[1]

---

[1] Reported in [Greif, 1981] and [Greif, 1982].

To support *meetings over time* we attempt to identify working relationships and support them through protocols special to the relationship. Often the system can compose and send an appropriate message based on information about the relationship between sender and recipient. For example, writing a meeting onto one's own calendar can trigger a request to the "conference room" calendar for a reservation, eliminating a call or message to the secretary who maintains that calendar.

To support *simultaneous* meetings we provide a shared workspace for all participants. On a computer terminal the shared space is viewed in a section of every participant's screen which displays the common information. Another area of the screen is used for a private work space. In some applications the information in one part of the screen may be closely related to that in the other, e.g. schedule cards with blocks of free and occupied time may appear in the shared space while appointment details appear only in the private space.

Information about working relationships underlies both mechanisms. In addition to aiding individual users as they work in meetings over time, this information is the basis for distinguishing between and coordinating information displayed in the shared and private workspaces of a simultaneous meeting. In the rest of the paper we focus on this one aspect of support for cooperative activities. We take most of our examples from meetings over time and consider uses of a database of working relationship information to support cooperation.[2]

CALENDAR MANAGEMENT

While individual time management support in the form of personal reminders and ticklers can serve an important function, the real power of a calendar management package is in its facilities for coordination among individual calendars. If it can support scheduling of meetings among individuals (without the sacrifice of privacy of personal calendar information), access to public "resource" calendars (e.g. for conference room reservations) and notification of company-wide events from a central calendar, then an on-line calendar becomes an attractive alternative to the convenient paper calendar in the pocket. Calendar management *is a cooperative activity* and calendar management software should be designed and built with this in mind.

---

[2] In [Greif, 1982] additional aspects of simultaneous meetings are discussed including mechanisms for yielding the "floor" at a meeting, voting, etc.

Roles and Working Relationships

An individual may choose to share his calendar information with others, but the calendar is his private database and he controls all access to it. Acting as the owner of his calendar, he may give access to his secretary or to certain working group members so that they can write in his calendar to schedule meetings. Since strangers may want to request appointments, he might also specify that members of the public can "write" in his calendar -- an appointment added by an individual member of the public is interpreted as a request for a meeting at a given time. A group member might only be able to cancel an appointment that he himself added to the owner's calendar while a secretary may have broader capability to modify the schedule.

A secretary and a manager might have to cooperate in maintaining the manager's calendar. Their working relationship definition includes

- a distinction between personal and business appointments (only business appointments would be handled by both people)
- conventions about erasures (should an appointment cancelled by the secretary disappear or be brought to the manager's attention as "cancelled")
- conventions about notification (how should new appointments written by one be brought to the other person's attention?)

The working arrangement might change over time and its "definition" should be modifiable.

Communications Needs

People tend to keep their own schedules in individual databases. When those databases are implemented on paper calendars, secretary/manager cooperation can be quite complicated. Often the secretary and the manager keep their own separate records of the manager's schedule and have to communicate on a regular basis to keep the copies consistent. Scheduling of meetings involves similar coordination among "group members." Just as the secretary and manager have to keep their calendars consistent, meeting participants must maintain the consistency of information about the meeting. Typically the initial scheduling is done in several passes starting with collection of scheduling constraint information followed by negotiation and selection of a meeting time. Coordination is then needed to communicate changes, e.g. to change the meeting location.

Most commercially available on-line calendars are either individual user tools or support

communication by using the mail subsystem for notification. We have two comments on the use of electronic mail for scheduling meetings. First, simple notification (a one way communication) is often not adequate. Negotiation might involve a series of communications based on examination of calendar information about conflicts, etc. Second, even when simple notification is appropriate, this approach provides only a poor approximation to an intelligent tickler file. A mail subsystem will only be able to inform the user that he has new mail -- not that he has a new meeting. A calendar subsystem designed to interpret changes by users as communication might be able to report the "You have a new meeting with Joe on your calendar but it conflicts with your squash game."

Integration of messages with the calendar information becomes even more important in meeting scheduling when large numbers of messages may be received regarding a single meeting. The user should be able to choose to be notified only after a significant set of answers have been received. That may mean anything from waiting until all answers are received, to notification after a majority of people have declined to attend, to notification only if no one can attend. These kinds of summaries are generally not within the capabilities of mail facilities.

Communication through Database Changes

For calendar management we suggest that the calendar be organized so that

- users assume a *role* when using the calendar (such as owner, secretary, public)
- items in the calendar are annotated with information about roles
- items in the calendar are annotated with information about status of the item: who it was written by, whether it is "new" from the point of view of certain users, etc.
- descriptions of working relationships are maintained in the database

The calendar management subsystem would then be able to support cooperation in the following ways:

- the calendar information can be presented in a manner appropriate to one's role (e.g. only business appointments will be displayed to the secretary).
- users can be notified of *new* items, *changed* items, etc.
- users can be relieved of composing text for messages to be sent to, e.g., meeting participants, whenever all of the necessary information can be gathered from calendar entries

Notification can take many forms from highlighting new appointments when they happen to be displayed, to flashing a warning on the screen that there are new appointments to sending mail to participants who are unlikely to look at their calendars. Whether or not there is notification can be role or working relationship dependent -- the manager might want to be informed of new appointments that are early in the morning, but might prefer not to have special notification of appointments during normal working hours for which he regularly checks his calendar. This agreement can be described in the working relationship part of his database.

Communication about meeting scheduling can be accomplished in a similar manner. A meeting can be highlighted on a participant's calendar when it is a request requiring his answer. It may remain flagged as tentative while its time is being negotiated. It may become highlighted on the caller's calendar once all participants have responded to the requests. A summary of the caller's calendar might include the notice that "All the answers are in on another meeting." The caller can then examine a report on the answers and decide either to schedule the meeting or to continue negotiations.

DOCUMENT WRITING

Since document preparation is a major activity in most offices, word processing support has played a prominent role in office automation. Word processing subsystems of workstations provide support for the individual worker and do not directly support groups of people who must cooperate in the work of producing a document.

Working Relationships

An author may be someone who can create, add to or modify a document. An editor can change an existing document. A commentator might be able to use "blue-pencil" to edit a document. That is, he can indicate need for changes but old and new versions are visible and can be reviewed by the author. Publication consists of making the document available to the public in some form. A publisher may have this ability to "release" a document. Co-authors of a document may cooperate in a number of possible working relationships. If they work as equals, each may have authority to make final changes to the document, although in practice co-authors may want to review each other's changes. Co-authors need not have equal status. Each may have primary responsibility for a section of the document (one author may act as "editor" of his section and "commentator" on

others). When authors have different skills and knowledge, as with an engineer and a technical writer, many passes of editing, commenting and rewriting may be required to meet the standards of both.

Communication Needs

Co-authors need to be able to inform each other of the status of their work, to request comments from other authors, and to write comments on each others work. When working on documents written on paper, group members often use the document itself as the communication medium. Marks in blue-pencil and comments on margins allow editors and commentators to communicate with authors. If an author gives copies of his paper to a number of people, he may compare individual's comments or even collate their comments onto one master copy before proceeding to modify his text.

When working on-line one might choose to simulate the paper approach by keeping separate copies for each author, editor, etc. However, one should not underestimate the value of that blue-pencil when shifting from paper to on-line word processing. Collating of comments and coordination of divergent copies can be very difficult on-line since it is so easy for individuals to make widespread changes that completely overwrite old copy. Comments may be mixed in with the text and authors can easily be confused as to which version of the document (their own or one that an author revised) they are viewing.

Communication through Database Changes

To support joint document writing on-line we suggest that documents be stored in a database that is organized as follows:

- the documents should be structured: chapters, sections, paragraphs, sentences should be separate items in the database[3]
- multiple versions of important components of each document should be maintained (such as versions reflecting changes of different individuals)
- comments should be components of documents that can be associated with other components (so that a comment can be "about" a particular sentence)

---

[3][Hammer et al, 1981] describes a text processing system that uses document structure to aid the individual user in document preparation.

- information should be maintained for each document about roles, working relationships and actions by individuals.

With this data a joint document writing subsystem can provide user friendly interface to "blue-pencilled" text with the following options:

- a comparison of old and new versions
- old version with comments in a side margin
- the new version with changed sentences highlighted
- a summary of the "status" of the documents such as an outline in which a chapter heading is highlighted if that chapter has been changed.

As in the calendar, these presentations can all be refined to be role specific. Thus changes could be highlighted only if they are relevant to the user *in his current role.* For example, one might see different changes highlighted when one acts as "author in charge of chapters one and two" than one would as, say "publisher." Notification can be done automatically, where appropriate, so that an author might be informed that, "Your co-author has completed his editing changes to section three."

SUPPORTING COOPERATIVE ACTIVITIES

In both calendar management and joint document writing we have indicated some of the ways in which information about roles and working relationships can be used to support cooperative activities. Some of these ideas have already been incorporated in system designs: our own calendar system, PCAL, supports a simple version of the secretary/manager working relationship described above [Greif, 1981]; some mail systems [Galley, 1978] and [Holt and Cashman, 1981] have made use of the notion of roles to aid in tracing the flow of requests and in assigning responsibility for actions; and we are currently designing a joint document writing system that will help co-authors and editors work together.

Our immediate plans for further research on cooperative office work are in the following two areas:

- user interface design -- to determine appropriate presentation of information, for example, to aid users in distinguishing their own work from that of others

- database organization -- to abstract from our prototype implementation general principles for combining working relationship information with application specific information.

Spanning both of these areas is research on the user interface to the database: we will develop an end-user language for defining new working relationships. The definitions will be in terms that are relevant to the working situation, but will be translated into definitions of database structures and constraints on views of that data.

Longer term goals for the project include extension of our approach to a wider range of cooperative activities and implementation of cooperative applications subsystems on an integrated multi-function workstation.

THE OFFICE WORKSTATION

Many office workers are already taking advantage of computer support tools for individuals, such as word processing or tickler packages, and therefore store much of their own private data on-line. When this application-specific information (e.g. the appointments or documents) is *structured* and *integrated with role information*, it is transformed into a powerful communications substrate for an office workstation.

If individual application subsystems of a workstation (including electronic mail) are built on top of a common database of the form described above, there can be a single uniform way for the user to specify how information in *any application area* is to be organized and presented to him when he is working. When the user first appears at his workstation, he can view a summary of his entire work environment highlighting notification of new mail, new calendar items, action pending on documents ("your co-author has been changing the report") and even the fact that there is mail *about* a calendar item.

We do not want to suggest that on a workstation supporting cooperative activities, communication through electronic mail will be eliminated. We do suggest, however, that the user should be able to communicate with others even while working within a specific application subsystem. Integration of *all* communication, whether through mail or applications subsystems, can have impact on all of the user's interactions with the workstation: the user will be better served, even when he works individually, due to the increased support for organization and presentation of information that is characteristic of this database-oriented workstation design.

REFERENCES

Greif, I. 1981. "PCAL: A Personal Calendar." MIT-LCS Technical Memo TM-213. December, 1981.

Greif, I. 1982. "Calendar Management as a 'Multi-Person Activity.' " MIT-LCS Office Automation Group Working Paper. WP-030. January, 1982

Galley, S., A. Vezza and M. S. Broos. 1978. "Data Based Message Service User's Manual." Internal documentation. MIT Laboratory for Computer Science. Programming Technology Group. September, 1978.

Hammer, M., R. Ilson, et al. 1981. "Etude: An Integrated Document Processing System." Proceedings of the 1981 Office Automation Conference. AFIPS. March 1981.

Holt, A. W. and P. M. Cashman. 1981. "Designing Systems to Support Cooperative Activity: An Example from Software Maintenance Management." IEEE Computer Society's Fifth international Computer Software and Applications Conference. Los Alamitos, California. November, 1981.

TELECONFERENCING AND THE COMPUTER-BASED OFFICE WORKSTATION

Irene Greif[1]
MIT Laboratory for Computer Science
Cambridge, Massachusetts

1. Introduction

The computer-based office workstation will soon be standard office equipment. While initial products will provide communication support in the form of electronic mail, more advanced, "integrated" workstations will provide additional communication support greatly enhancing people's ability to work together at a distance. Prototype systems are already under development in several research centers which support "person-to-person" communications *without requiring the user to type text messages*. The potential of more advanced office workstation facilities than standard electronic mail and computer conferencing should be considered in planning for teleconferencing systems.

To illustrate the potential of the computer, this paper describes our research on how *groups* of people *interact and cooperate* in carrying out typical office activities. The research has involved identifying working relationships among people and their communications protocols. We are designing computer-based workstations for multi-person activities and implementing command packages suitable for carrying out these protocols.

This paper describes our pilot study on CALENDAR MANAGEMENT. The design illustrates support of two kinds of cooperative activities: *simultaneous* meetings and *meetings over time*. The ways in which we support these kinds of activities generalize to joint document writing as well as many other applications. To illustrate this, we also present two scenarios from the area of document writing. Our conclusion is that office workstations will play a significant role in supporting groups of people working together -- they can provide a kind of support that is different from and complements electronic mail, computer conferencing and other electronic conferencing media.

---

[1]This research was supported by the Advanced Research Projects Agency of the Department of Defense and was monitored by the Office of Naval Research under Contract Number N00014-75-C-0661.

2. Supporting Cooperative Work

2.1. Group Interactions

Groups of people have evolved various ways to work together without reliance on computers. For example, they can convene at one location for a conventional meeting, they can meet at a distance via telephone conference (without sharing a visual workspace), they can hold a video conference if they wish to share exhibits or they can work together in a "meeting over time" through correspondence via mail or memos. The most popular use of computers to date in support of group work has been to provide an electronic medium for meetings over time. This extension of electronic mail has come to be known as "computer conferencing." Currently an electronic mail or message package is the standard for facilitating group work on computer-based office workstations.

As more office work and data is kept on-line, we see increasing opportunities (and need) for additional kinds of communication support. We can take some examples from current text editing or word processing facilities. Today authors writing a document together on-line are left to their own devices to maintain the integrity of the document. If both edit it at once the changes of one will probably be lost. If one makes changes to be read and approved by the other it is difficult (if not impossible) for the reader to find out where the changes are and what the old version looked like. The author who has made revisions typically will send mail to his co-author saying, first, where the new version can be found (in what "file") and then describing the changes. If two authors "link" their terminals[2] in order to make changes visible to both in real-time, again, they must devise and enforce their own protocols for taking turns writing.

These kinds of coordination problems in both "simultaneous" meetings and "meetings over time" (also referred to in this paper as real-time and delayed modes of communication, respectively) can and should be supported directly as part of the application facility. The same kinds of communication issues arise in joint document writing, joint design (with graphic display), calendar management, project management, on-line debugging and teaching facilities and many other applications.

2.2. System Support

We have developed techniques for supporting two kinds of meetings -- *simultaneous* meetings and *meetings over time*. Our prototype system for CALENDAR MANAGEMENT illustrates support for both kinds of meetings.

---

[2] Linking terminals makes everything typed on one terminal appear on both.

To support *simultaneous* meetings we provide a shared workspace for participants. The shared space is displayed on every participant's screen so that everyone sees the same information. Another area of the screen is used to display a private work space. In some applications the information in one part of the screen may be closely related to that in the other. The private space may contain some sensitive data while a corresponding "filtered" version may appear in the shared space. For example, in the calendar, schedule cards with blocks of free and occupied time appear in the shared space while appointment details appear in the private space. We presume that the co-workers can speak to each other through a (conference call) telephone connection and are simply using the shared space for viewing data.[3]

To support *meetings over time* we attempt to identify working relationships and support them through protocols special to those relationships. In addition to aiding workers in synchronizing their efforts (as mentioned above) we can support privacy constraints on data by providing different users with appropriate views of the shared data. Also, the system often can compose and send appropriate messages based on information about the user's working relationship to another person. In the calendar, writing a meeting onto one's own calendar can trigger a request to the "conference room" calendar for a reservation, eliminating a call or message to the secretary who maintains that calendar. This relieves the user of composing the text of a message when all of the information has already been entered into a database.

3. Calendar Management

Calendar management is an office activity that involves cooperation and information sharing among groups of co-workers. Ideally a calendar program should support communication protocols for calendar sharing, for negotiating times for meetings, and for informing people of changes to their schedules [Greif, 1980a].

The following sections describe first the personal calendar and then several of the experimental "multi-person" facilities that augment its functionality. The examples illustrate several kinds of working situations:

- information sharing -- people working in a meeting over time while sharing a single database of information,

- real-time session -- people working together simultaneously,

---

[3] This assumption could clearly be reconsidered if for instance we were trying to provide a working environment for the deaf community. Mail-like meetings over time as in DEAFNET [] may be more appropriate for that purpose however, since typing is rather slow for general conversation in real-time.

- message passing -- people working together in meetings over time without necessarily sharing a common database.

3.1. A Personal Calendar

PCAL, the personal calendar, can be used to make, cancel and change appointments [Greif, 1982]. A brief appointment description (the "keywords") can be entered at a date and time. The appointment record may also include an end time, comments and a list of participants. Any of this information can be changed, and the calendar will be updated accordingly. Repeating appointments such as classes can be added as "series" of appointments to the calendar.

A note can be associated with a date, but not at a specific time. The date or the text of a note can be changed, a note can be erased, and as with appointments, one can specify a series of daily or weekly notes.

A calendar can contain more than one appointment at a single date and time. When a new appointment conflicts with an old one, the user is notified, but the new appointment is still entered. It can be rescheduled by changing its date and/or time. Alternatively, the "undo" command can be used to remove it from the calendar.

Calendar information is presented on a screen that is divided into sections (windows) that are used to display different kinds of information [cf. Figure 1]:

- The top two lines of the screen are used to show *status* information including the calendar name and current operation. Also, a "current" day and "past" day are shown. The current day is the day that was affected by the last PCAL operation. It will be used as the default date in commands until a new date is specified.

- The *main* section of the display contains calendar information such as lists of appointments.

- The bottom line of the screen is the *command line*. The normal prompt in the command line is `pcal>`. Users type commands and their arguments in this window. A small text editor is available in this window so that it is possible to modify a command line once it is written and before it is executed.

- The line above the command line is the *message line* used to display short error messages and other information about PCAL operations.

- There is a *help window* that appears, overlapping the side of the main display whenever the user requests help. This same window is sometimes used for presenting information about the last PCAL operation.

- At the user's option, command arguments can be entered by filling in a form. The form appropriate to the current command will appear at the bottom of the main window in the

forms window [cf. Figure 2].

3.2. Sharing Calendars

Scenario. A secretary and manager communicate about the maintenance of the manager's calendar. Many of their communications are implicit in changes to the manager's calendar.

System Support. The secretary and manager are sharing access to a single database (the manager's calendar database). Each can change the calendar, though the secretary's access may be restricted to "business hours." Certain appointments might be designated as "personal" by the manager and the system will support enforcement of access rights [cf. Figure 3].

During certain standard hours ("office hours") when the manager has agreed in advance to be available, the secretary can communication new appointments to him simply by writing them in the calendar -- he will see them when he looks at his calendar at the start of his office hours. Other additions or changes can cause a "message" to be sent. This message may take the form of highlighting an appointment, setting an alert (so that the manager at his workstation is told to look in his calendar), or even sending mail to the manager. The form of a "message" is determined by user preference indicated in a user profile. Examples of situations requiring "messages":

- a new appointment added by the secretary during business hours other than "office hours" may trigger a "message" to the manager.

- the manager may cancel an appointment that was originally entered by the secretary. The appointment will disappear from the manager's view of his calendar, but will be flagged for action in his secretary's view. (The secretary may need to follow through with phone calls, etc. to complete cancellation.) [cf. Figures 3 and 4].

- the manager can explicitly flag a meeting as requiring secretary action.

Some system commands. There must be system commands for describing the working relationship by naming one's secretary, setting office hours, business hours, etc. Each user must be able to display a list of all new items, or items that require his attention. Also, each user might want to ask explicitly to have the other notified of a new appointment perhaps overriding the default agreed on in the working relationship.

Note that this last feature can be thought of as adding a "notify secretary" field to an appointment form as viewed by the manager. The manager can check off this field and so does not have to compose a message describing the appointment. That is, notification becomes another *calendar operation* and is not the same as composing and sending mail.

3.3. Simultaneous "Meeting to Schedule a Meeting"

Scenario. Secretaries meet to arrange a conference of their managers.

System Support. There is a shared space in which a merge of the "schedule cards" of the *meeting* participants (the managers) appears, showing free and blocked time [cf. Figure 5].[4] Each *session* participant, (each manager's secretary), can also see his manager's calendar for the corresponding time period in another (private) part of the screen.

The shared space can be changed by, e.g. adding and removing individual schedules from the merge. Meeting times can be proposed: the system records who has conflicts at each time and also supports voting on acceptance of proposals. A review of the proposals can be made before a final time is chosen [cf. Figure 6]. A user may cancel an appointment on his personal calendar in order to free a slot for the meeting. This change to his private calendar must be reflected appropriately in the shared space.

Some system commands. One person at a time can execute commands on the shared data. These commands include proposing a meeting item (which triggers a vote), committing a meeting time, adding or removing a schedule card, changing the data being viewed. (The shared and private spaces are both changed so that they remain coordinated.) Participants can request "control," the caller of the session can preempt control, and individuals can leave the session temporarily. There are commands for voting on proposals and for reviewing the outstanding proposals.

3.4. Scheduling a Meeting in Delayed Mode

Scenario. Co-workers want to schedule a meeting. Each maintains his own private calendar.

System Support. A group of people are authorized to write in each other's calendars in order to schedule working meetings. The caller of a meeting writes the meeting onto his own calendar and the system "sends" requests to others by writing in their calendars. They may accept or decline this "invitation." Since more complex protocols for negotiating a meeting time are also supported, the meeting may initially specify a choice of times. The system alerts the "caller" when answers are received or after a timeout. The system will also maintain consistency among the calendars if there are changes made to the meeting after the time is set.

Some system commands. Schedule meeting (to write the appointment on the caller's calendar),

---

[4] Note that this shared space may have to be constructed from views of a number of databases: Each schedule card is a "filtered" view of one user's calendar, suitable for public viewing. The "merge" of the cards showing possible meeting times is not necessarily a view of any one database. User may keep their calendars in separate databases.

check participants (to look at participants' calendars when deciding on a time), send invitations, check status (to review answers).

3.5. Conference Room Reservations

Scenario. Scheduling a meeting place by signing up for a public resource.

System support. Writing a meeting onto one's own calendar can trigger a request to the "conference room" calendar for a reservation. This is similar to scheduling a meeting with other people since two private calendars must now be kept consistent. In fact, the user can arrange that cancellation on his own calendar will automatically cause cancellation on the conference room calendar.

The conference room calendar is a shared calendar with a particular working relationship (different from that of the secretary/manager described above): individuals in the "public" can sign up for the resource, no one else is notified of these new appointments, and each user can only cancel his own reservations. There may also be another "role" in this calendar: a "monitor" might have additional authority to cancel appointments made by members of the "public." Presumably the conflict policy for this calendar also differs from that of the personal calendar since conflicting appointments should not be written onto the calendar for a non-sharable resource.

Some system commands. The usual commands for adding and removing appointments from the calendar are used: if a "location" is specified when the appointment is entered then the conference room calendar will automatically be updated whenever the appointment is changed or cancelled.

3.6. Tentative Meetings

Scenario. Several times are kept available for a meeting until shortly before it will occur [Greif, 1980b].

System Support. The system coordinates the participant calendars to keep track of the possible times. Each calendar is aware of all possible times for the meeting. Users can make appointments that conflict with some of the possible times as long as they don't make this meeting impossible. When only one time is left, or at a date shortly before the meeting time, all are notified of the final schedule.

Some system commands. Similar to those for scheduling a meeting.

4. Joint Document Writing

The calendar example illustrates communication through shared data and in real-time sessions. In communication through shared data we allow users to act in certain "roles" that summarize their

- access rights to the calendar data base
- ability to perform operations in the calendar
- desires to be informed of other user's changes.

In a real-time session each is provided with a view of a shared workspace. Each user also has a private view of any shared data. The session is controlled through commands for

- requesting control,
- voting,
- reviewing proposals,
- entering and leaving sessions.

A number of these same facilities can be useful in other applications. In the area of joint document writing it is most important that these facilities for communication be integrated with sophisticated text processing facilities, taking advantage of document structure to support working relationships of the co-authors. The following sections describe two different document writing situations which can be supported by such an integrated facility.

4.1. On-line Instruction Manuals

Scenario. Imagine a situation in which documentation for a computer system is provided and maintained on-line. The documentation is written by a variety of kinds of people, possibly from different organizations. Each may have a different *role* both in viewing and updating the documentation:

- vendor -- provides original documentation for system as sold to the user organization
- local system developer -- produces descriptions for his own changes to the system
- educational consultant -- provides tutorials for the system. Note that there may be a number of such consultants involved since the vendor and the user organization might commission their own tutorials.
- users -- write comments on the documents

* to point out trouble spots

* to remind themselves of how something "really" works

* to note suspected "bugs."

Reading the Document. Most people think of text as unstructured and not easily processed by a computer. In fact, documents have a lot of structure that can be taken advantage of in processing text:

- chapters, sections, paragraphs

- guidelines for different kinds of readers (level of detail, order of reading.)

What is more, there can be privacy constraints on the document:

- prerelease of update may be read by certain people for feedback before the general population of readers see it

- the supervisor or system operator may be able to read additional information about privileged commands.

Advanced text processing systems such as ETUDE [Hammer, 1981] aid the user in viewing the "logical structure" of his document. He can view a summary of this structure in terms of chapters, sections, etc. along side (on a small part of the screen) a formatted presentation of the "outward appearance" of his document (as it will appear on paper). The document's structure can also be taken advantage of in supporting working relationships: people will have different views of the document and the system can take advantage of associations between parts of documents.

Communication Support. *Bug reports* are *comments* that are addressed implicitly to "someone who can help." If a bug report is associated with a part of the documentation written by local system developers, the bug report might be routed to that group. If a section was developed by the vendor the bug report might be sent directly back to the vendor. It might be channeled through a user consultant group first, to check whether it is a misunderstanding on the part of the user, as opposed to a true bug report.

We should emphasize that notification need not involve the mail system. Just as users are now told that they have new mail when they log on to their workstations, the appropriate users might be informed that there are new document comments to be read. When such a user looks at the document he can be show a view of the document that brings these new comments to his attention.

Changing the Document. The system can support instruction manual updates by assuring that all

views are appropriately updated and by initiating communication related to the change. For example, a vendor update:

- may be accompanied by an update to the tutorial
- may require an update to the tutorial (so that the educational consultant must be notified)
- may require an update to local version and/or local version tutorial (so that the local group should be notified)
- may respond to comment -- perhaps the person who wrote that comment should be notified
- may eliminate an old section and its associated comments.

4.2. Author and Editor -- Real-time Sessions

A document can be created with information about author and editor and used in meetings over time. Each might like to be notified of work by the other. It will be possible to associate comments with components of the text. The document will have several possible outward appearances including presentations that highlight changes.

Occasionally in writing a document, there is a time when final negotiation of wording requires a meeting in real-time. With adequate support for a shared workspace, such meetings can be accomplished at a distance and can take advantage of computer support for text processing.

A real-time session for document writing, e.g. between an editor and an author, might involve three major workspaces: a shared space containing one view of the document; a private space for each participant which displays his personal view of the document (probably his comments on the text); another private workspace to be used for composition of sentences before they are revealed in the shared space.

Imagine that the editor is "controlling" the session. He might work through his private space to find his "this is awful" comment. Using interactive text editing facilities he finds this comment and highlights it. The result in the shared space is that the corresponding part or the document (perhaps a sentence of a paragraph) becomes highlighted. The editor, speaking on the phone to the author, can say "let's look at this sentence" knowing that the author will see it highlighted: he doesn't have to describe the sentence to the author to help the author find it on his own copy.

To make changes to a document the writers can either work directly in the shared space or can compose sentences in their private workspaces and move them into the shared space when compete. The system can keep track of "alternatives" for parts of the document and these alternatives can be

reviewed before a decision is made. The manner in which alternatives are displayed depends on a combination of user preference and terminal capability. Possible arrangements include seeing two version of the document side-by-side, each with a different alternative presented at the appropriate place or seeing an old sentence crossed out with the new sentence written in another color just above it.

5. Conclusions

The computer-based office workstation can play a significant role in cooperative work, and in replacing meetings. Advanced techniques in text processing, window management and graphics can and should be taken advantage of in providing communication support. Computer conferencing systems (such as EIES [Hiltz, 1981]) to date have presented the user with a mail-like interface that limits the user's ability to integrate this tool into his daily work routine. Increasingly, individuals are carrying out their daily activities on personal workstations that provide advanced facilities based on latest techniques in interface design making systems easy to learn and easy to use. They will be more comfortable working with other people if they can continue to rely on these high powered tools whenever appropriate.

Integration of advanced personal support tools with communication aids is achievable in the near future. By building on the successful introduction of office workstations into offices, computer-based tools for cooperative work will have impact on an organization's perceived need for both meetings that entail travel and for the use of more expensive teleconferencing facilities.

6. References

Greif, I. 1980a. "Support Tools for Calendar Activities." MIT Office Automation Group Working Paper, WP-025. August 1980.

Greif, I. 1980b. "Notes on the Design of a Calendar System." in *Proceedings of Computer Networking Symposium*. December, 1980.

Greif, I. 1982. "PCAL: A Personal Calendar." Internal documentation. MIT - LCS Technical Memo TM- 213 . January 1982.

Hammer, M., R. Ilson, et al. 1981. "Etude: An Integrated Document Processing System." Proceedings of the 1981 Office Automation Conference. AFIPS. March 1981.

Hiltz, Starr Roxanne and M. Turoff. 1981. "The Evolution of User Behavior in a Computerized

Conferencing System." CACM 24,11. November, 1981.

7. Figures

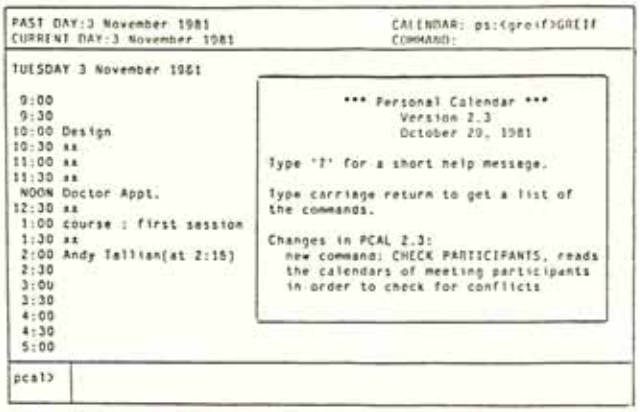

Figure 1: This is the initial display of the calendar as it appeared when PCAL was started on November 3, 1981. The information in the "help" window tells new users how to read a help message and informs regular users of program modifications.

Figure 2: An appointment form has "popped up" on the screen. A question mark in a field of a form results in a short help message about the contents of that field. The start time is a required field -- the appointment cannot be made without that information.

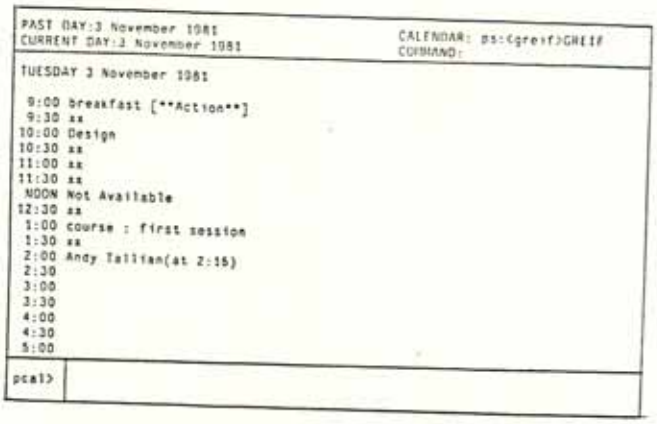

Figure 3: This is the secretary's view of the same calendar shown in Figure 1. There are two differences: the doctor's appointment at noon is evidently marked as personal by the manager; the secretary simply sees that time as "not available"; also, the secretary sees an additional meeting at 9am that is flagged for ACTION.

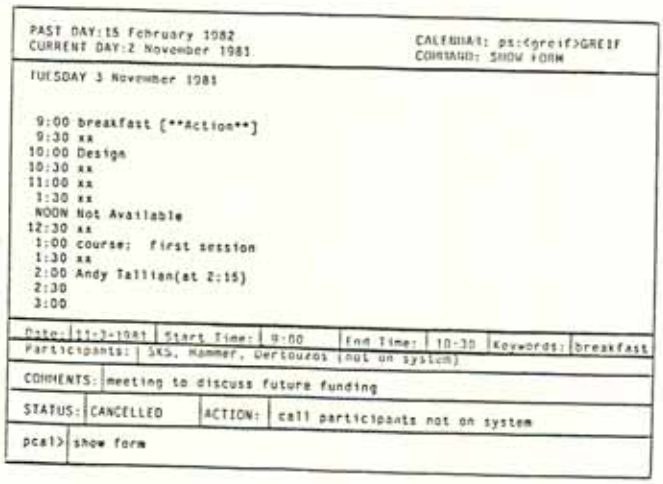

Figure 4: On examining the appointment form for more details the secretary finds out that the appointment has been cancelled but that it is necessary to call Dertouzos, the only meeting participant who does not keep his calendar on-line. Note the additional fields in the form as compared to the form in Figure 2.

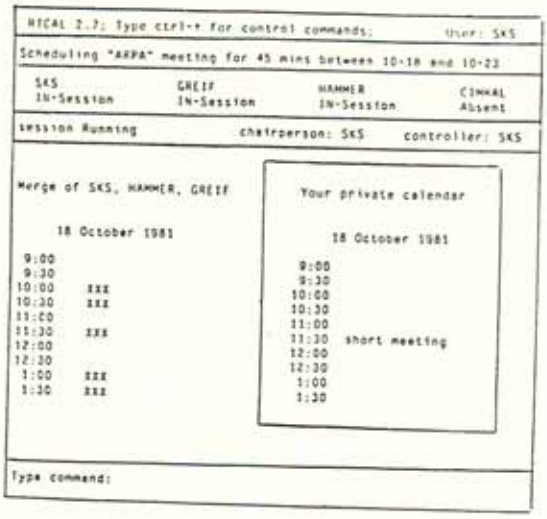

Figure 5: The display of RTCAL presents both shared and private workspaces. In this case the session has been called to schedule an "ARPA" meeting. The session is "Running" after three of the four invited people agree to participate. Their schedule cards are merged and displayed in the shared space to the left. The screen is shown from the point of view of "SKS" (Sunil Sarin) so that his private calendar is displayed to the right.

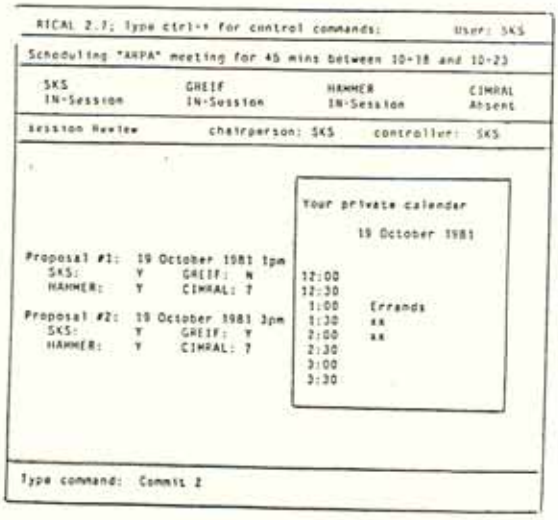

Figure 6: A review of the votes is shown in the shared space and the second proposal is selected.

The User Interface of a Personal Calendar Program

Irene Greif

May 5, 1982

Abstract

The paper describes PCAL, the personal calendar program that is in use on our laboratory's DEC-2060. Users can add, remove or change items in their calendars through a simple command language or by filling in "forms" that prompt the user to provide necessary information. Design decisions and planned improvements are discussed in a concluding section.

This research was supported by the Advanced Research Projects Agency of the Department of Defense and was monitored by the Office of Naval Research under Contract Number N00014-75-C-0661.

1. Introduction

We have designed and implemented a prototype calendar management system as part of a research project on computer support for group activities [Greif, 1982]. Calendar management is an example of an office activity that involves cooperation and information sharing among groups of co-workers. Ideally a calendar program should support communication protocols for calendar sharing, for negotiating times for meetings, and for informing people of changes to their schedules. Our research project is aimed at designing and building support tools for such cooperative activities.

Our first task was to study cooperative office activities to identify user needs. In parallel, we also began a program of implementing prototype application systems for user evaluation. This paper describes the user interface of one of those prototypes: a personal calendar system called PCAL. PCAL provides the basic calendar operations that allow an individual to maintain records of his own appointments. This facility primarily provides support for the individual. It is important to our research on cooperative activities because it has created a community of users who are maintaining their calendars on-line: they will be available to help us evaluate communications features as they are added to the calendar.

2. The Personal Calendar

2.1. Functionality of PCAL

PCAL can be used to make, cancel and change appointments. A brief appointment description (the "keywords") can be entered at a date and time. The appointment record may also include an end time, comments and a list of participants. Any of this information can be changed, and the calendar will be updated accordingly. Repeating appointments such as classes can be added as "series" of appointments to the calendar.

A note can be associated with a date, but not at a specific time. The date or the text (keywords) of a note can be changed, a note can be erased, and as with appointments, one can specify a series of daily or weekly notes.

These appointments and notes are stored in a personal calendar database (referred to hereafter as the calendar). One other kind of record can be stored in the calendar: a seminar. It is like an appointment in that it has a date and time, but it also contains seminar information such as speaker and host names.

Notes and appointments (and seminars) are retrieved from the calendar by date or date and time.

They can be displayed in any of three ways:

- keywords for all appointments (and notes) for a day are displayed. Appointments between 9am and 5pm appear in a tabular form showing blocks of free and occupied time. [cf. Figures 1-3, 6, 7]

- keywords for all appointments for a day (or week) are listed in time order [cf. Figure 5].

- a single appointment is written out in full detail including comments and participant list [cf. Figure 4]

It is possible to look at other people's calendars, although the current version of PCAL does not allow one to write in anyone else's calendar. There is one exception: a user who is designated as "secretary" for another person's calendar can write in that calendar.

A calendar can contain more than one appointment at a single date and time. When a new appointment conflicts with an old one, the user is notified, but the new appointment is still entered. It can be rescheduled by changing its date and/or time. Alternatively, the undo command can be used to remove it from the calendar.

2.2. The User Interface -- Screen Layout

The screen is divided into sections (windows) that are used to display different kinds of information:

- The top two lines of the screen are used to show *status* information including the calendar name and current operation. Also, a "current" day and "past" day are shown. The current day is the day that was affected by the last PCAL command that was executed. It will be used as the default date in commands until a new date is specified.[1]

- The *main* section of the display contains calendar information such as lists of appointments.

- The bottom line of the screen is the *command line*. The normal prompt in the command line is **pcal>**. Users type commands and their arguments in this window. A small text editor is available in this window so that it is possible to modify a command line once it is written and before it is executed.

- The line above the command line is the *message line* used to display short error messages and other information about PCAL operations.

- There is a *help window* that appears, overlapping the side of the main display whenever the user requests help. This same window is sometimes used for presenting information

---

[1] Whenever the current day is changed, the last current day is remembered as the past day since it is likely to be a date that the user will want to refer to again easily. PCAL will recognize "CURRENT" and "PAST" as names of days, just as it recognizes "MONDAY," "TUESDAY," etc.

about the last PCAL operation.

- At the user's option, command arguments can be entered by filling in a form. The form appropriate to the current command will appear at the bottom of the main window in the *forms window*.

Some sample screen layouts appear in the figures at the end of the paper. At start-up the display will be as in Figure 1. That is, the appointments for "today" will be displayed in the main window, the status window will have been initialized, and some information about the current version will appear in the help window.

There are some relationships between the data shown in various windows. For example, once the date entry of a form is filled in, the main display is changed to show the appointments for that day [cf. Figure 7].

2.3. The Command Language

2.3.1. The Command Line

Commands are in the form of short phrases such as "schedule appointment" or "cancel 1-1 noon."[2] It is not necessary to type the whole command name, a prefix will do. However, if the characters don't identify a single command, a list of the possibilities will appear [cf. Figure 3]. An incorrect command can be edited before it is executed.

2.4. Forms

Information (arguments) can be given to most commands on the command line. If all of the necessary information is provided, the operation will be performed. If some additional information is required, then a form will appear with blanks for the rest of the information. If part of a command line cannot be understood, it will be ignored and a form will be displayed.

The easiest way to learn about PCAL is to type commands with no arguments -- the appropriate form will appear, prompting for the necessary information.

---

[2] The commands are listed in the appendix.

2.4.1. Filling in Forms

The forward and back arrow keys[3] can be used to move from entry to entry in the form. Up arrow "enters" the form -- the form will be processed by PCAL. Down arrow "quits" from the form -- the form will disappear and the current command will be aborted.

Within an entry some simple editing commands can be used; the commands can be listed by typing ↑H (backspace). To find out what kind of information is expected in an entry of a form, type a "?" in that entry.

2.4.2. Short Command Lines

There are a number of short cuts for typing in appointments and notes without typing a command name. The experienced user may choose to make appointments, add notes and show new days by using these abbreviations, rather than by filling in forms. PCAL will make the following assumptions about the intended commands:

- Typing a date will change the current day and display that day's appointments.

- Typing a date, start time, end time (optional) and keywords will cause an appointment to be made at the specified date and time. If no date is given, the current date is used as a default. Thus one can enter a new appointment on the current day by typing only a time and keyword..

- Typing a date and keywords (but no times) will result in adding a note to the list of notes for that date.

The default command if none can be recognized is "schedule." Thus a new seminar can be created by typing "seminar" rather than "schedule seminar."

2.5. Help Facilities

2.5.1. Reading Help

The commands can be listed by typing a carriage return (after an empty line) in response to the pcal> prompt. "Help" or "?" in the command window will produce a short help message. For help that is specific to a command type "help <command name>" or the command name followed immediately by a "?". Many command names are several words in length: a command name followed by a space and a "?" lists the possible next words. (If there are no "next words" then the help message will explain the meaning of the command as typed.)

---

[3] The escape sequence equivalents of these arrow keys can be used as well on terminals that do not have special keys.

2.5.2. Forms Help

When arguments are not supplied with the command on the command line, a form will appear in the lower part of the main display [cf. Figure 6]. Arguments can then be specified by filling in this form. Thus in a sense the forms facility itself provides a kind of "help" for the user by prompting him for arguments.

There are two kinds of help available explaining how to fill in a form. To find out what kind of information to type into an entry, one types "?" in that entry [cf. Figure 6]. The help message will explain:

- what kind of information to enter: e.g. a date,
- what interpretation PCAL will place on this information: e.g. a time might be a start of an appointment or an end of an appointment, or even a length of an appointment,
- whether the information is required or optional: generally dates and start times are required, end times, comments and participant lists are optional.

The second kind of help is about the "forms editor." One can always type ↑H (backspace) to read help on how to use this forms editor. It explains the use of the "arrow keys" (or their corresponding escape sequences) to move around from entry to entry or to exit from the form. This help message will also list the editing commands that can be used within each entry of the form.[4]

2.6. Format of Time and Date Information

2.6.1. Time

Times can be written in the form HH:MM using 12-hour time, followed by AM and PM. Only an "A" or a "P" need be typed. Times without a sufficient number of characters for the form HH:MM in them will be padded with zeros as follows:

```
12:2    => 12:02
12      => 12:00
```

DEFAULTS: When AM or PM is not specified, times from 7 through 11:59 are interpreted as AM, times from 12:01 through 6:59 are PM. (12:00 or 12:00PM is noon, 12:00AM is 0:00, printing as MIDNIGHT. PCAL will recognize the word NOON as a time). PCAL will also interpret times between

---

[4] The editing commands are EMACS-like control-character commands.

12:01 and 23:59 according to a 24 hour clock.[5]

2.6.2. Dates

In commands that require a date, the default is the "current day" shown at the top of the screen. The current day is set to the actual date ("today") when PCAL is entered, and is changed in accordance with the dates of the appointments, meetings, etc. that are made or cancelled.

Dates are entered in one of the following forms:

- <month>-<day>-<year>: the month must be written as a number, 1 through 12, not as a name (such as "January"); the year can be "19yy" or simply "yy".

- <month>-<day>: When no year is specified, the year will be set so that the specified day is in the *future* (with respect to today's date). Thus, if today is 2-1-1981, the date 2-3 will be understood to be 2-3-1981, whereas 1-31 will be understood as 1-31-1982.[6]

- One of the words "TODAY," "YESTERDAY," "TOMORROW."

- A name of one of the days of the week: "MONDAY", "TUESDAY" etc. (these are interpreted with respect to the "current" day)

- "NEXT" or "PREVIOUS," which are the days after and before the current day, respectively. ("CURRENT" is also a legitimate date. It is rarely necessary to explicitly write the date "CURRENT" since leaving the date unspecified will have the same effect.)

- "PAST" is the day that was current before the last calendar operation. This date is also displayed at the top of the screen.

3. Experience with PCAL

PCAL has been used since September 1981. Over a hundred people at a number of locations have tried PCAL and given us feedback: about twenty people at MIT now use PCAL regularly.[7] We keep a log of users and of unhandled error conditions for debugging purposes. We expect to extend the logging facility to gather more information for human factors studies.

The users have for the most part been computer professionals who maintain their own calendars.

---

[5] We originally planned to implement the calendar with 24-hour clock only. There are a number of successful calendars, even in non-military settings in this country, with 24-hour clocks. However, reaction to a prerelease of PCAL from a small set of users was so strongly against this approach that we changed the time conventions before the initial release.

[6] There is one exception to this convention. The ARCHIVE command works only on dates in the past. Arguments to this command that do not specify a year will be assumed to be in the past.

[7] There are additional regular users at other sites, but we do not have records of usage on machines other than our own DEC-20/60.

A few people have begun working with their secretaries to maintain their calendars on-line.

3.1. Self-teaching

After a system announcement was made, people began using PCAL. All of the computer scientists have taught themselves to use the calendar. Several asked for off-line documentation to read, but most people were able to make and cancel appointments without training.

The secretarial staff did not find the calendar easy to learn through on-line help. The small number of secretaries who have tried PCAL did succeed in learning to use a subset of the features. PCAL is usable by people who are not computer experts, but to date these users have all required some amount of training. We consider this to be at least in part an indication of inadequacy of the on-line help facility.

3.2. Hidden Commands

Many questions and requests for added features revealed that people were not finding out about the short versions of commands or about some of the multi-word commands such as SCHEDULE SERIES. They used the first level of commands and filled in forms. While a number of people find this interface adequate for long term use, there is a need for more support for introducing "advanced features" and abbreviations.

3.3. Reverse Video

One of the most commented-upon features of PCAL is the use of reverse video. Evidently no other programs in our laboratory have taken advantage of this feature of the local terminals. This and the use of multiple windows surprised and pleased a number of people. It also led to an interesting observation about user interface design. The VT52 does not support reverse video. Although many people in our laboratory use that terminal, the group of people who developed PCAL all used terminals with reverse video capability. While several users of the VT52 did compliment us on the appearance of the calendar, we were surprised the first time we looked at the display. Our design of forms depended heavily on the use of reverse video for readability. In fact, we have since changed the layout to support PCAL on the VT52.

3.4. Choosing Sensible Defaults

Some of the short forms of commands are based on choosing sensible defaults for the "verb" in the command phrase. Thus if someone types "2-3 9:00 breakfast," it is likely that he is scheduling an appointment. If he types "2-3 call Joe," he is probably making a note (there is no time specified). A

date alone seemed to be best interpreted as a request to see that day's appointments. While many people never discovered that they could omit the command names from command lines, those who did were comfortable with our choice of defaults: there were no mismatches between their expectations and the actions of the system.

However, in the area of partially specified dates (in particular, for dates without a year) we found that it was very difficult to decide on sensible defaults. For example, the defaults for dates described above (section 2.6.2) confused a number of users. Alternatives suggested by these users were equally confusing to other individuals. There are some similar problems with the defaults for times, although mistaken interpretations of times have not caused as much confusion.

It would seem to be useful to develop a better warning system to let people know when a questionable default is being taken. We are considering ways of identifying odd cases and highlighting dates and times that are completed according to the system's defaults.

3.5. The User's Model of Calendars

Several people found that PCAL did not match their model of how a calendar should work. That model, shaped by years of experience with paper calendars, suggests that one should be able to write directly on a calendar and erase items from the calendar. Thus the command orientation occasionally put people off. We find that the commands and forms are essential to extending the functionality of the calendar beyond that of ordinary paper calendars. Nevertheless it seems important to provide the new user (or the user who has no interest in advanced features) with an interface that is natural. Presumably, an interface in which one writes directly on the calendar would require less training of new users. However, the problem of helping users learn about advanced features may be further complicated by this change. The user will now have to learn two modes of interaction: writing directly onto the calendar and using commands.

3.6. Command Language Design

The command language was designed to use a number of familiar (and non-threatening[8]) command names. We introduced a number of alternatives for removing items from the calendar: users can "cancel" appointments, "erase" notes, and "clear" blocks of time. It appears now that the command language ought to be simplified. Perhaps cancel, erase and clear can be synonyms so that people can choose the command that seems most natural. The top-level menu should be much

---

[8] The "clear day" function was originally accomplished by a "kill day" command, but we followed human factors guidelines [Good, 1981] and used less threatening language.

shorter than the current 18-entry one.

3.7. Help Facilities

There are many ways to ask for help in PCAL: ?; help; help help; help <command name>; <command name>?; and ↑H. The number of choices can confuse the user. This problem could be alleviated somewhat by the use of special function keys. However, there do, in fact, seem to be too many kinds of help available. It is not clear how to make it most convenient for the user to get the kind of help he wants without being overwhelmed by the possibilities.

4. Plans for Research in User Interfaces

To date we have introduced only a few "multi-person" features to the calendar. One example is the ability to designate a user as one's secretary. PCAL's success as a personal calendar has depended on user acceptance of its interface. Informal evaluation through user feedback has shown that it is convenient to use for maintaining a person calendar. The forms editer and the use of forms for simplifying the interface have greatly enhanced ease of use of PCAL.

We plan to modify the interface and to provide the user with alternative modes of interaction with the calendar. The naive user should be shown a simple calendar that matches his current model of calendars. We expect to expand the use of forms for more of the calendar display so that the user will write onto a "form" for a day or week of his schedule.

The system should aid the more experienced user in learning about new and more powerful features different from those he can anticipate from his experience with paper calendars. Support of the user as he changes his "model" of an application, whether calendar management or another office activity, will be a major thrust of our continued research in user interface design.

5. References

Good, M. 1981. "ETUDE and the Folklore of User Interface Design." MIT Office Automation Group Memo. OAM-030. Massachusetts Institute of Technology. Cambridge, Mass. March, 1981.

Greif, I. 1982. "Computer Support for Group Office Activities." Proceedings of the 1982 Office Automation conference. AFIPS. San Francisco, California. April, 1982.

Liskov, B. et al. 1979. CLU Reference Manual. MIT-LCS Technical Report. TR-225. Massachusetts Institute of Technology. Cambridge, Mass. October, 1979.

Larry Rosenstein. 1980. "How to Use the ECOLE Redisplay System." Internal Documentation. Massachusetts Institute of Technology. Cambridge, Mass. December, 1980.

Appendix I -- List of Commands

The following is a list of the calendar commands. Arguments are written in brackets, e.g. <date>. Command phrases consist of multi-word commands, e.g. "schedule appointment," followed by arguments.

Some commands require an appointment as an argument. An appointment is specified by its date and time. If the appointment is on the currently displayed day, then only the time need be supplied. If there is more than one appointment at a given date and time, a numbered list of alternatives will appear in the help window -- the user identifies the appointment by number.

- **For adding new items to the calendar:**
  - * note
    - note series
  - * schedule (same as schedule appointment)
    - schedule appointment
    - schedule meeting
    - schedule seminar
    - schedule series
- **For examining calendar entries:**
  - * list (same as list day)
    - list day <date>
    - list week <date>
  - * look at <calendar name>
  - * print (same as print day)
    - print day <date>
    - print week <date>
  - * show (same as show day)

- show day <date>
- show form <appointment specification>
- show users

- **For removing items from the calendar:**
  * archive <start date> <end date>
  * cancel <appointment specification>
    - cancel series <appointment specification>
  * clear (same as clear day)
    - clear day <date>
    - clear am <date>
    - clear pm <date>
  * erase (same as erase note)
    - erase note <date>
    - erase series <date>

- **Other operations:**
  * change <appointment specification> (or <date>)
    - change note
  * check
    - check participants <appointment specification>
  * escape (for debugging only)
  * help <command name>
    - help help
  * quit
  * redisplay
  * set
    - set secretary

* undo

Appendix II -- Implementation Notes

PCAL is written in the programming language CLU [Liskov et al, 1979] and runs on the DEC 2060. It supports a number of terminals typically used on that machine including the VT100, VT52 and Heath H19. We chose to this hardware base because it is available to most of the laboratory members.[9] We have applied design guidelines developed in research on advanced office workstations design [Good, 1981] to the available technology. This influenced our command language design. However, the hardware base did not support use of pointing devices or of special calendar function keys.

A window package for an office workstation had already been developed in our group [Rosenstein, 1980]. The package was available in CLU and we used it for PCAL. The forms facility was not available as part of the window package and we developed it ourselves.

Several people at other sites are using PCAL. Since CLU is not generally available on other machines we have simply transported the executable code to other DEC 20/60's. The only portability issue so far has been in adjusting for differences in terminal type numbering conventions at different sites.

---

[9] Other office work, such as text processing and messaging, is done in our laboratory on timesharing systems that were not particularly designed for the office. There are no commercial word processors in our laboratory and most systems that people here are familiar with have not been designed to be either easy to learn or to use.

Appendix - The PCAL Display

Figure 1

```
PAST DAY:3 November 1981                          CALENDAR: ps:<greif>GREIF
CURRENT DAY:3 November 1981                       COMMAND:

TUESDAY 3 November 1981

 9:00                          *** Personal Calendar ***
 9:30                                Version 2.3
10:00 Design                         October 29, 1981
10:30 xx
11:00 xx                      Type '?' for a short help message.
11:30 xx
 NOON                         Type carriage return to get a list of
12:30                         the commands.
 1:00 course : first session
 1:30 xx                      Changes in PCAL 2.3:
 2:00 Andy Tallian(at 2:15)     SCHEDULE MEETING and SCHEDULE APPOINTMENT
 2:30                           now both have comments/participants.
 3:00
 3:30                           new command: CHECK PARTICIPANTS, reads
 4:00                           the calendars of meeting participants
 4:30                           in order to check for conflicts
 5:00

pcal>
```

Fig. 1: The calendar display for November 3, 1981[10]

[10] The 1pm appointment is part of a series. The series keyword is "course". The keyword for the November 3 instance of the course series is "first session."

Figure 2

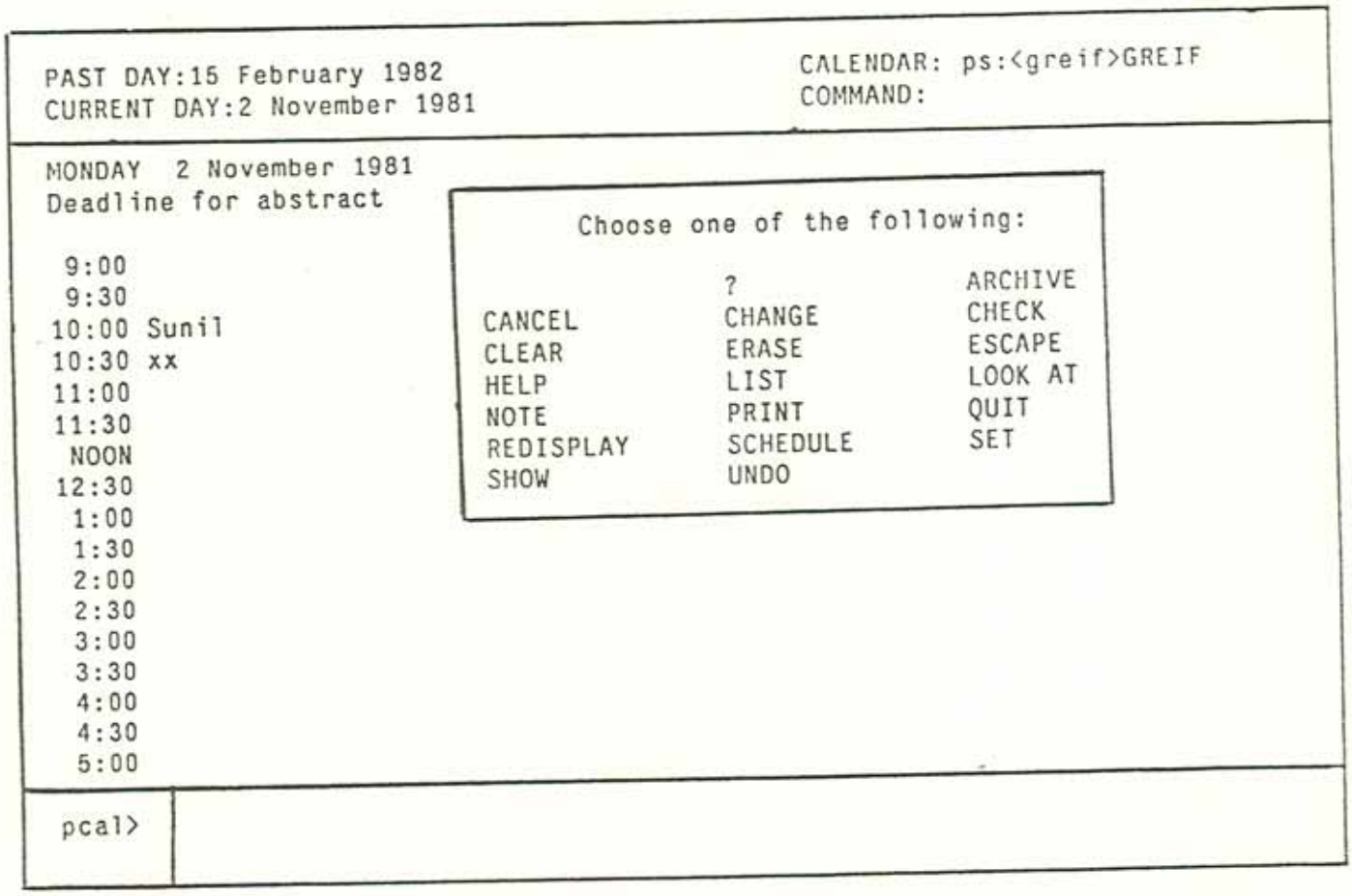

Fig. 2: The list of commands in PCAL
obtained by typing carriage return

Figure 3

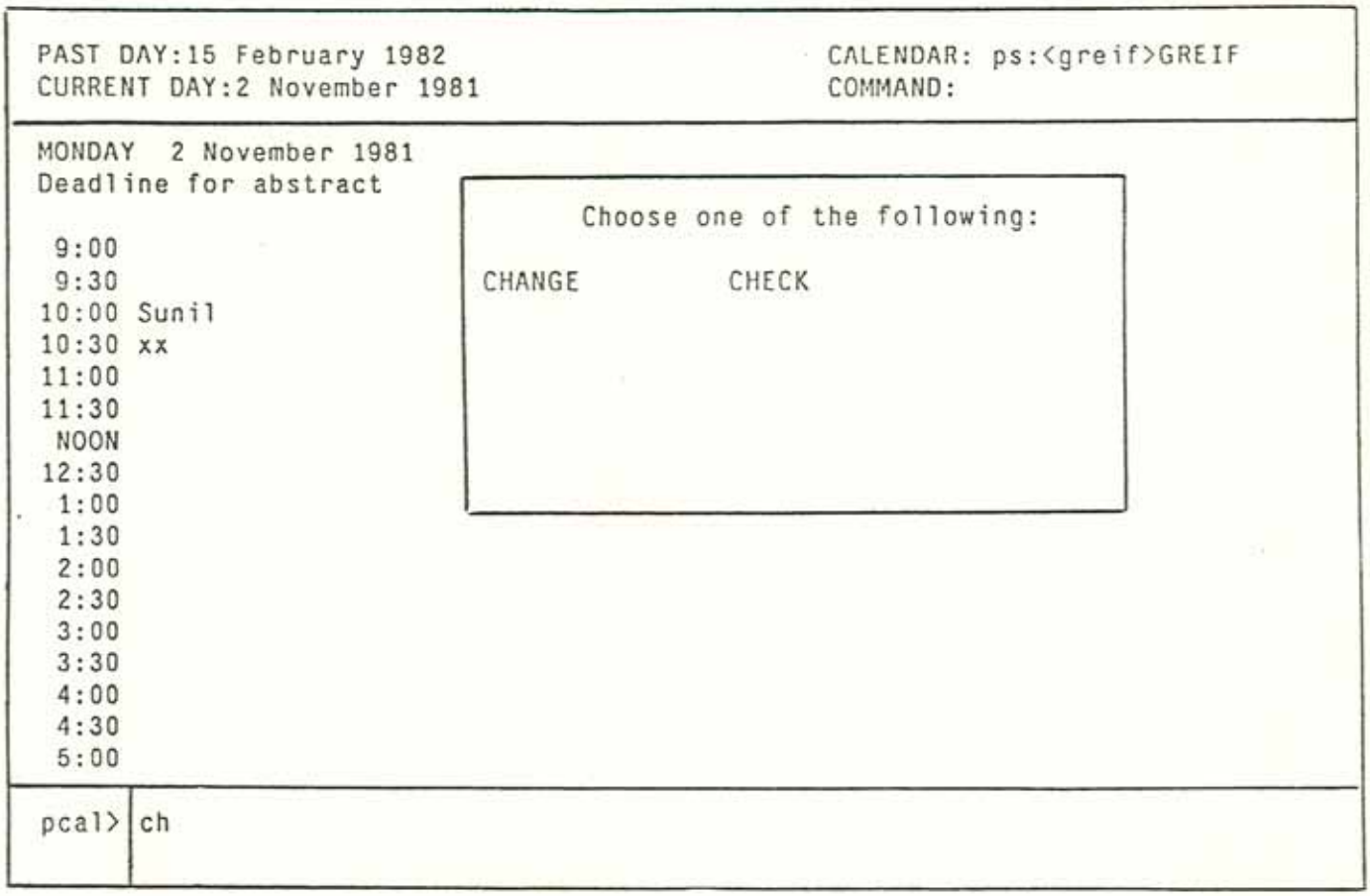

Fig. 3: An ambiguous command is entered. The options are listed in the "help" window.

Figure 4

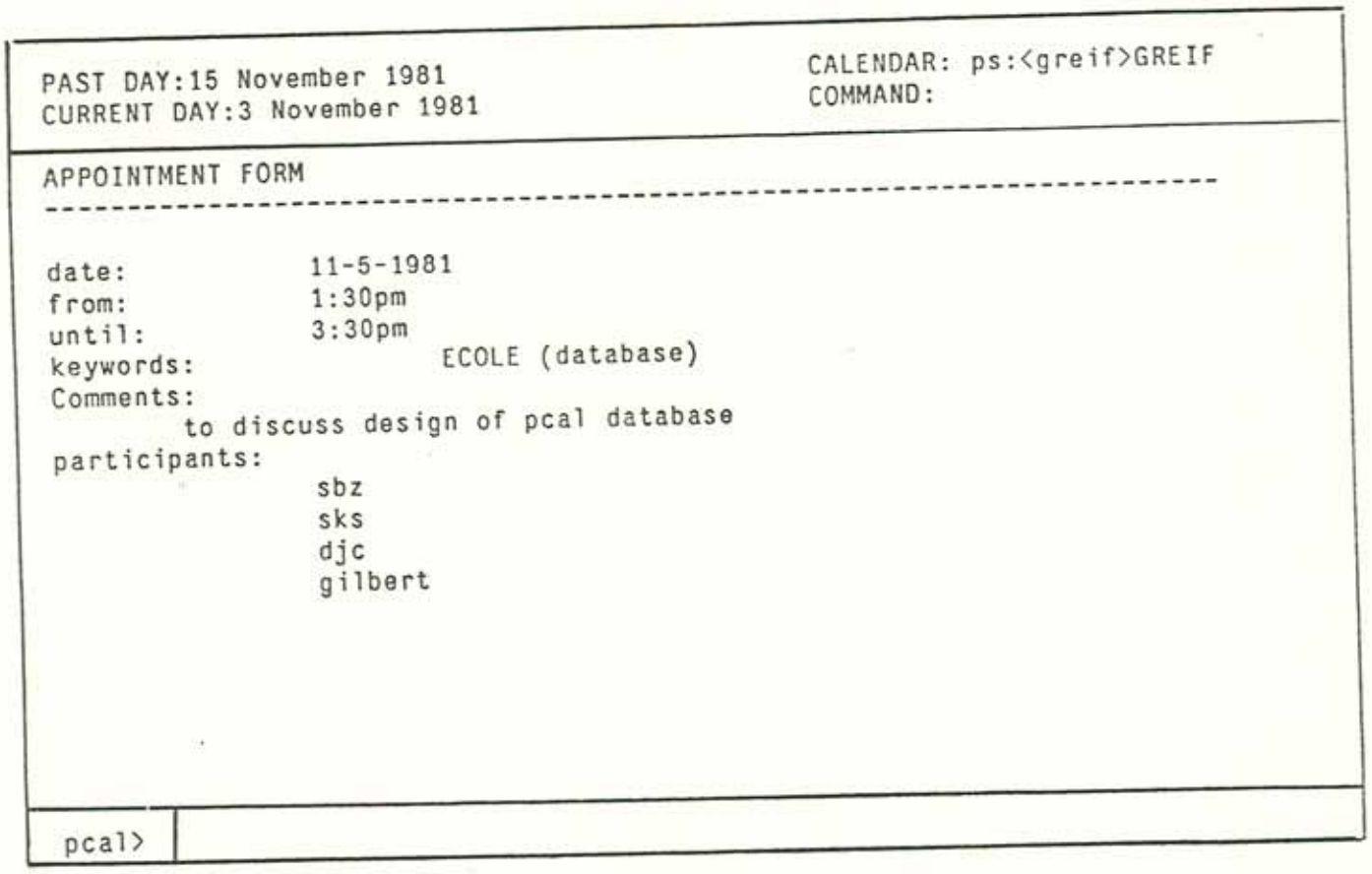

Fig. 4: The result of a "show form" command.

Figure 5

```
PAST DAY:15 February 1982                    CALENDAR: ps:<greif>GREIF
CURRENT DAY:2 November 1981                  COMMAND:

MONDAY  2 November 1981
Deadline for abstract
     ** DAY FREE **

TUESDAY 3 November 1981
10:00   12:00   Design
13:00   14:30   course:  first session
14:15           Andy Tallian

WEDNESDAY       4 November 1981
Pick up rug
     ** DAY FREE **

THURSDAY        5 November 1981
call technical writer
13:30   15:30   ECOLE (database)
19:00   20:30   dinner

FRIDAY  6 November 1981
     ** DAY FREE **

pcal>
```

Fig. 5: The list format for the display

Figure 6

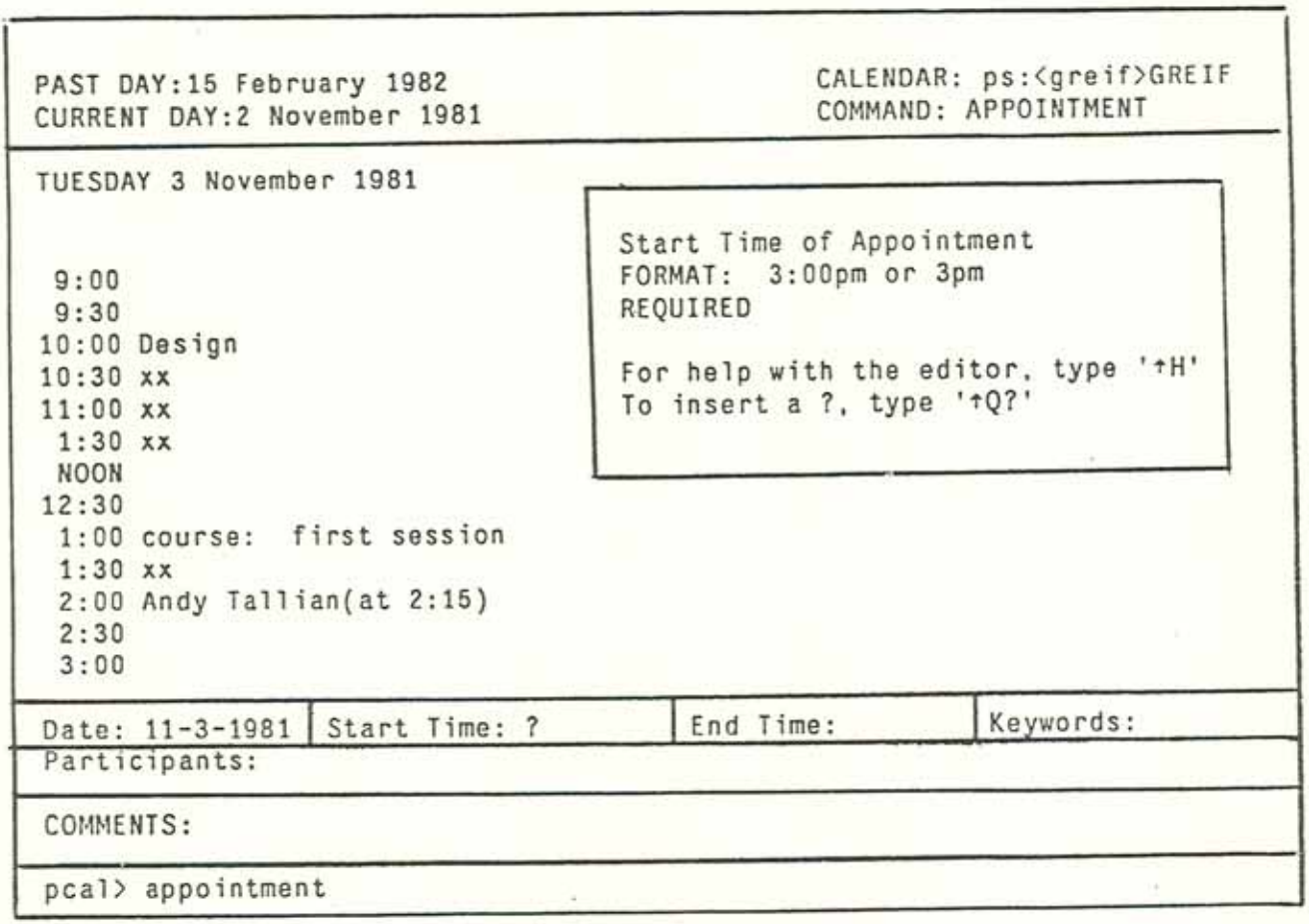

Fig. 6: A question mark in a field of a form results in a short help message about the contents of that field.

Figure 7

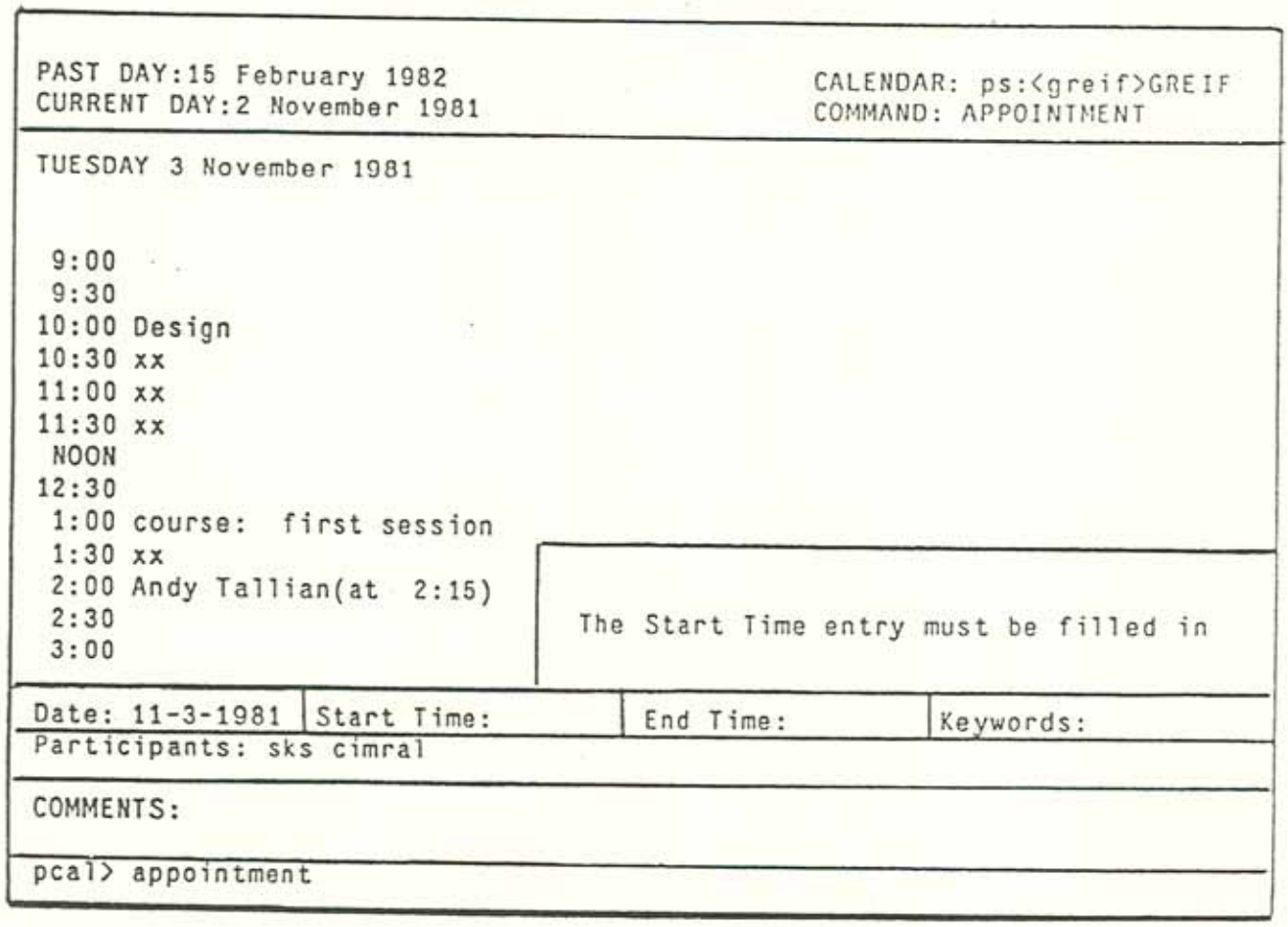

Fig. 7: An appointment form is displayed. The main display shows the appointments for the day in the "date" field.

The up arrow key was hit before a *required* field (the start time) was filled in, hence the error message.