Redesigning the Palm Pilot

Contents

1. Executive Summary

A team of four Masters Students in Human Computer Interaction comprised of diverse backgrounds in Cognitive Science, Computer Science, Psychology, Computer Engineering, and Design, carefully analyzed the Palm Pilot to understand where the problems lie. The analysis techniques of Contextual Inquiry (CI), Contextual Design (CD), Heuristic Evaluation (HE), Cognitive Walkthrough (CW), and Think Aloud (TA) illuminated both the problems with the Palm's interface and the primary needs of users. Using an affinity diagram and prioritizing issues by importance or severity, we consolidated all of our Usability Aspect Reports (UARs) and Contextual Design breakdowns to generate design ideas. Two themes emerged from this consolidation. The first major theme centers on the Date Book; it is a crucial application, but it is designed so poorly that users express frustration and anger towards it. Effectively satisfying the scheduling needs of busy people is a difficult task to accomplish and the Palm Pilot unsatisfactorily supports these needs. Our redesign of the Date Book alleviates these problems and frustrations by modifying where salient information is located and how this information is interacted with. The second major theme is that users want to know what applications are open so that they know what the system is doing and can easily switch between them. Users would switch back and forth between applications but needed to go through an unnecessary step each time. We are removing this step and at the same time bringing the interactions with the Palm Pilot closer to what is familiar to a computer user. The following pages will more thoroughly explore these two major themes and the redesign suggestions we are proposing as they relate to use case scenarios.

2. Redesign Ideas & Use Case Scenarios

2.1. Scenario #1. Automatic Meeting Time Coordination

In this scenario, our user is eating in a cafeteria on campus, when a student approaches her and asks to set up a meeting time. The solution for this scenario was inspired primarily from our Contextual Design data. In our Sequence Diagram, we modeled an interaction between the user and a student in which a breakdown occurred during an attempt to schedule a meeting (CI – L275-289). We feel that this type of task is common enough that it warrants a design solution, and since the Palm is a portable organizational tool, it is the perfect candidate to affect this solution.

2.1.1. Problems Encountered & Proposed Design Solutions

- **Problem**: Users with scheduling tasks are influenced by others who request meeting time with her (CD: Cultural Model). This can lead to breakdowns when attempting to find a time that works for all parties (CD: Sequence Model).

  Solution: We propose a networked calendar where users can check their schedules against other users' schedules (DI-04).

- **Problem**: Users are influenced by others who need to meet with them; also, other individuals inform users of meeting obligations (CD: Cultural Model).

  Solution: Account for users' network of contacts while scheduling since they deeply influence the schedule (DI-06).

- **Problem**: User spent too much time in her roll as 'scheduler', which took away time from her primary role as 'professor' (CD: Workflow, Cultural, Sequence Models).

  Solution: The Palm should support more efficient role switching (DI-16).

2.1.2. Interaction Screen Shots

In the steps outlined below, we'll demonstrate how the Palm could be redesigned to automatically suggest meeting times between two or more people.

Step 1: Select "Coordinate Meeting" Option.

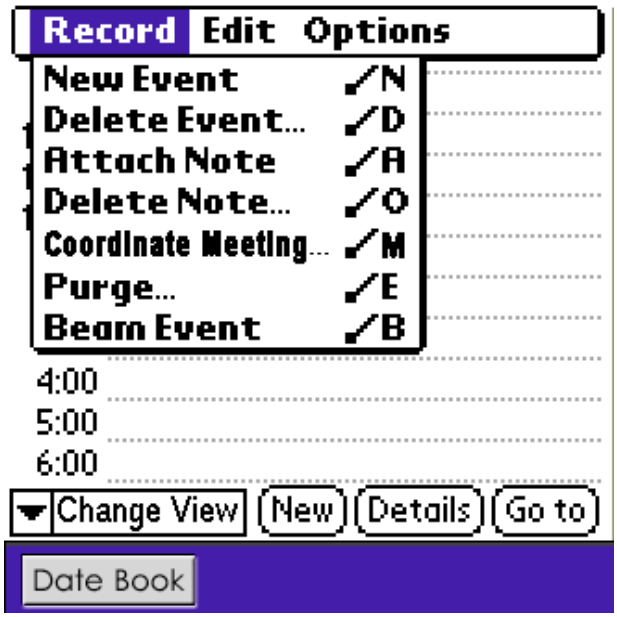

Figure 2.1.1:
Menu screen with "Coordinate Meeting" option.

Our user and the student pull out their Palm Pilots. The user goes to the Date Book application, accesses the menu, and selects the 'Coordinate Meeting' item under 'Record'. We felt that the Coordinate Meeting functionality wouldn't be accessed often enough to be placed on the screen by default, and so should be placed under a menu item. The Record menu contains functions similar to Coordinate Meeting, such as creating and deleting events.

Minor Issues & Design Solutions in Step 1.

- **Problem**: Buttons for Different Date Book Views Unclear: The buttons for Day View, Week View, etc. are dots and don't represent their meaning well (HE-49).

  Solution: Change 'View By' option to drop-down menu to explicitly labeled drop down menu rather than ambiguous icons (DI-23).

Step 2: Select Users for Meeting Coordination.

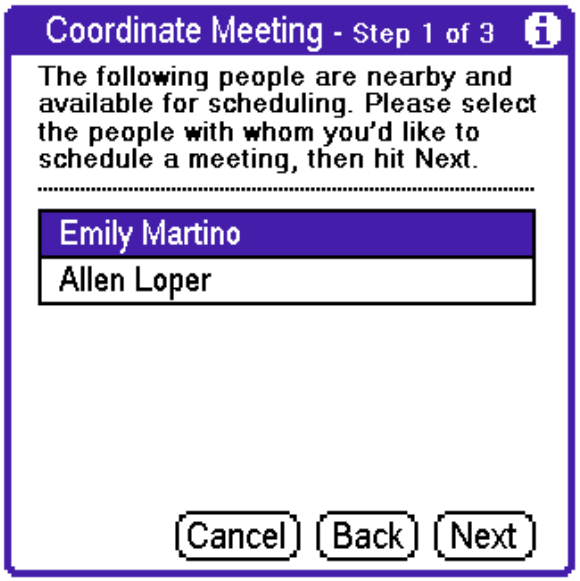

Figure 2.1.2:
Screen to select nearby people to network for meeting times.

The user is presented with the first step of a Coordinate Meeting wizard. Her position in the wizard (step 1 of 3) is clearly labeled, following the heuristic of providing system status. There are three buttons at the bottom of the screen: Cancel and Back, which are inspired by the heuristic of allowing user freedom, and Next, which is inspired by the heuristic of preventing errors (the user has to make their selections and confirm them by clicking Next). There are two people within the user's range who have their preferences set to allow the coordinating of meetings, and Emily Martino, the student, is one of them. The user taps on the rectangle containing Emily's name; this was inspired by Fitts' law (Model Human Processor), in that the larger the clickable area (an entire rectangle as opposed to just a name), the faster the action performance time. Emily's name is highlighted, and the user hits the Next button.

Step 3: Select Window for Meeting Start Time.

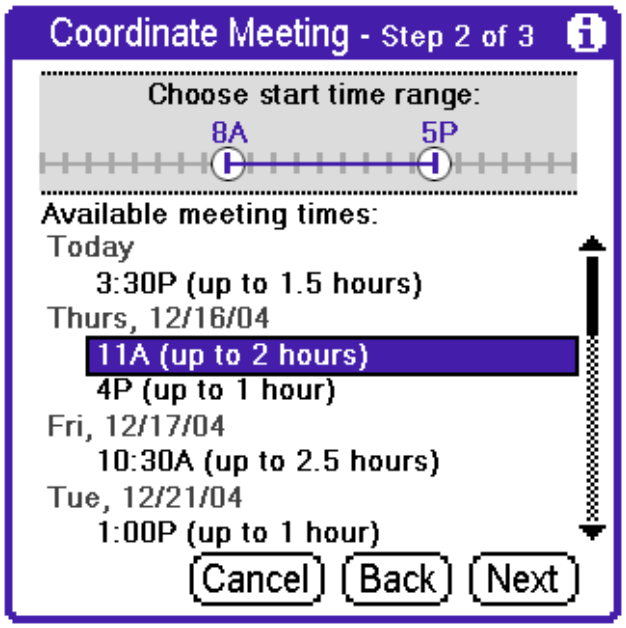

Figure 2.1.3:
Screen to select a window of time for the meeting start time.

In this step, the Palm has determined all available meeting times for the user and the student (based on the appointments in their Date Book and the given range of start times), and asks the user to select a start time. Notice the time line with the gray background. In order to reduce screen clutter, we wanted to only show available times between the hours of 8 AM and 5 PM by default. However, if the user would like to see more or fewer available times, she can adjust this timeline, and the visible dates and times displayed below would automatically adjust. The user sees that she and the student both have two hours available beginning at 11 AM on Thursday. She taps this time, which produces a highlight behind the time, and then hits the Next button.

Step 4: Confirm New Meeting & Add to Date Book.

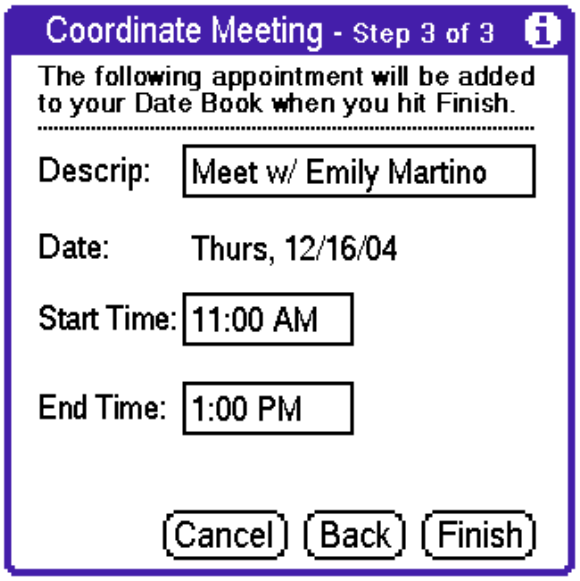

Figure 2.1.4:
Screen to accept meeting time.

In this final step, the user is asked to confirm the appointment. A default description is automatically entered for her, but she can change it if she wants. Since the system knows that the available time window for this appointment is two hours, it sets the start and end times by default, but the user may adjust them if desired. This particular interaction would need to be further developed. How would the user select a new end time? Would they be prevented from selecting an end time that violated the available time window? We decided that the Date would not be editable on this screen, since modifying it would require a complex interaction in order to prevent the user from violating the coordinated dates/times. Instead, the user could use the back button if she wanted to modify these items. In our scenario, the user is satisfied with the default values, and hits the Finish button.

Step 5: Confirm Addition of Meeting to Date Book.

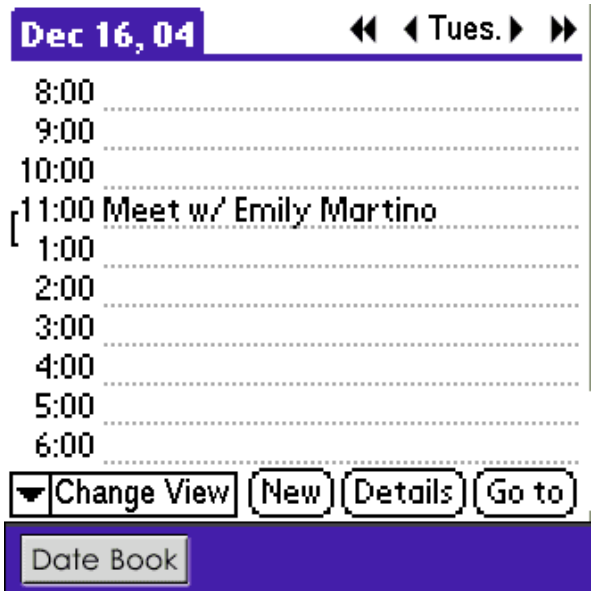

Figure 2.1.5:
Date Book entry confirmation screen.

Once the user has completed the Coordinate Meeting wizard, she is brought to the day view of the day for which she scheduled the appointment. This provides confirmation for the user that her appointment has been added to the date book. Also, from this screen the user could hit the Details button to access other features for their appointment, such as setting a reminder. All the people with whom this meeting was coordinated – in this scenario, Emily Martino – would be sent an email with an appointment attachment, which they could then accept and have the appointment automatically added to their calendar as well (see Scenario 3).

Minor Design Ideas & Inspiration in Step 5.

- **Problem**: The user goes past Oct. 29 when navigating the calendar by week (TA-16). Also, the Cognitive Walkthrough determined that the function of the original navigation arrows was unclear (CW-01).

  Solution: In Date Book day view, single arrow navigates day by day, double arrow navigates week by week (DI-01).

2.1.3. Tradeoffs / More Issues to Explore

One important addition to explore and further develop is coordinating meetings with people who are not physically nearby. We think that this is a common user task that should be supported; however,

further Contextual Inquiries would need to be conducted to determine the validity of this notion. These CIs would give light to whether Saturday and Sunday should be included in the list, for example. Before this solution gets implemented, it should undergo further user testing to determine if it meets user needs. A good method would be a Think-Aloud usability study, since this solution is task-based and the feature is new.

One tradeoff for this solution revolves around the fact that it is implemented as a wizard. While wizards theoretically make the task easier by walking the user through the process in a step-by-step fashion, they also sacrifice the flexibility and potential time-savings of more open-ended solution. The wizard solution would be appropriate for novice users, but perhaps another available option should be considered for expert users.

2.2. Scenario #2. Adding Events / Appointments in Date Book "Term" View

In this scenario, the professor is setting up her schedule for next semester. She wants to be able to visualize her courses so she can make sure that there are no conflicts before confirming all of her appointments. The primary inspiration for this redesign idea comes from the CD Artifact Model — the professor kept her semester weekly calendar on a separate piece of paper. By introducing a module that supports tentative schedule planning, users from academia (or users whose schedule changes every set period of time) might be more inclined to use their Palm device to set up their schedule.

2.2.1. Problems Encountered & Proposed Design Solutions

- **Problem**: Inefficient paper calendar is used to plan out semester schedule; requires data to be entered twice (CD: Workflow, Artifact Models).

  Solution: New Date Book "Term" view addresses repeated weekly planning without using the 'repeating' appointment functionality repeatedly (DI-02).

2.2.2. Interaction Screen Shots

In the steps outlined below, we'll demonstrate how the Palm could be redesigned to automatically handle repeated scheduling in a "Term" view.

Step 1: Select the Date Book Term View.

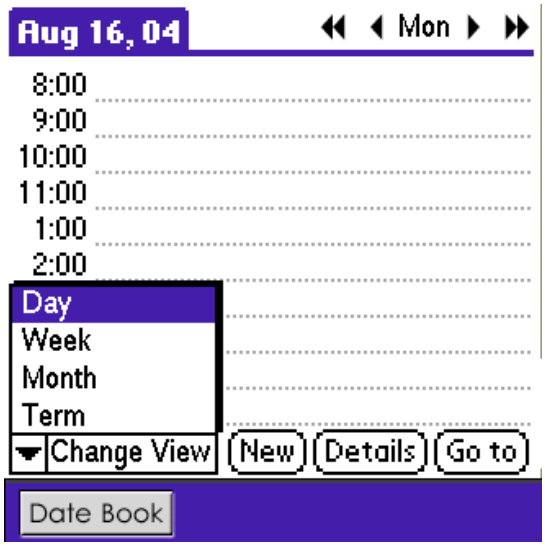

Figure 2.2.1:
Day View in Date Book with "Change View" drop-down menu expanded.

Figure 2.2.1 shows the standard Palm interface for the Date Book, with one minor exception. Instead of the icons to represent the various views, a drop-down menu is now used. A new "Term" view is incorporated for the new feature in this scenario. The professor expands the "Change View" menu by clicking on it, and then selects the "Term" view.

Minor Issues & Design Solutions in Step 1.

- **Problem**: Buttons for Different Date Book Views Unclear: The buttons for Day View, Week View, etc. are dots and don't represent their meaning well (HE-49).

  Solution: Change 'View By' option to drop-down menu to explicitly labeled drop down menu rather than ambiguous icons (DI-23).

Step 2: Create a New Term.

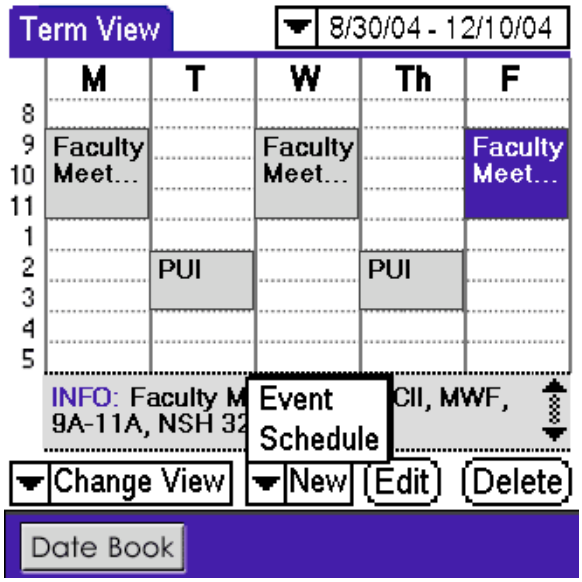

Figure 2.2.2:
User's current term is displayed. The "New" drop-down is expanded in this screen shot.

Figure 2.2.2 shows the professor's current semester schedule. She can click on any of the events to get a description about the event and the details of the time and place of the event. In this case, however, her goal is to begin planning for the next semester. The user clicks the "New" drop-down menu and selects "Term."

Step 3: Set Up Parameters for New Term.

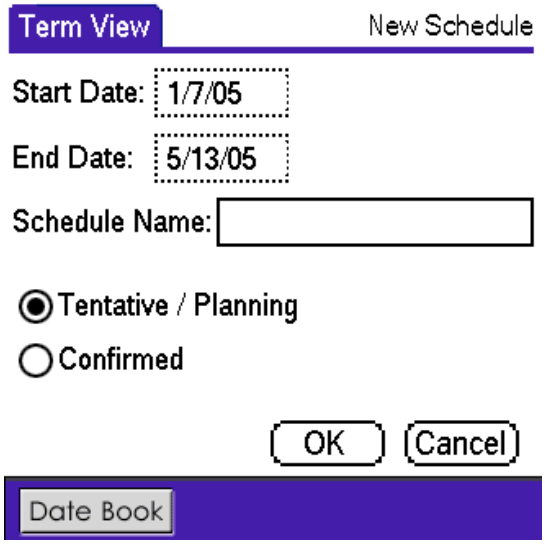

Figure 2.2.3:
Screen to set up a new term.

Figure 2.2.3 shows the screen for instantiating a new schedule. The start date and end date are set once for all events. The time saved is significant since the user would have to set the recurrence for each event individually under the old Palm system. KLM analysis would be appropriate here to determine

the amount of time that the user would save by using this interface, but the fact remains that a frequent user task has been consolidated into one step.

The professor can choose to name the schedule (e.g., "Spring 2005") and note whether the schedule is Tentative (being planned) or Confirmed. Once the schedule is confirmed, the appointments will be placed in her date book automatically, but for now, she leaves the tentative option selected and clicks "OK" to confirm the information.

Step 4: Add a New Event to the Term.

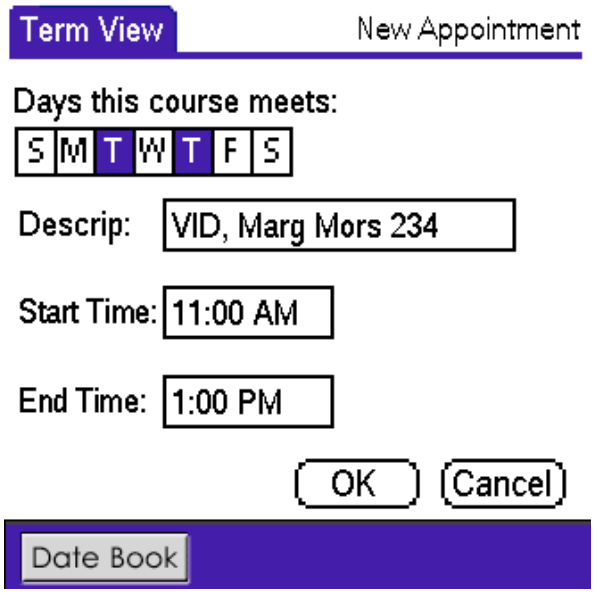

Figure 2.2.4:
Screen to Add a New Event to the term.

Adding a new event to a term is similar to adding a new event in the day view; the only difference is that recurrence is incorporated into the event by default. This consistency meets the Consistency & Standards heuristic. The user can choose the start time and end time in the same way that the user is familiar with in the Date Book. The user can confirm the new event by clicking "OK".

It is important to note that while this handles the circumstances found in the Contextual Design data, it does not handle the instance where an event takes place within a Term schedule but does not start and end on the same dates (e.g., a "Mini" course). We acknowledge that further CI work would need to be done to determine the necessity of such a feature.

Step 5: Finish Term Planning.

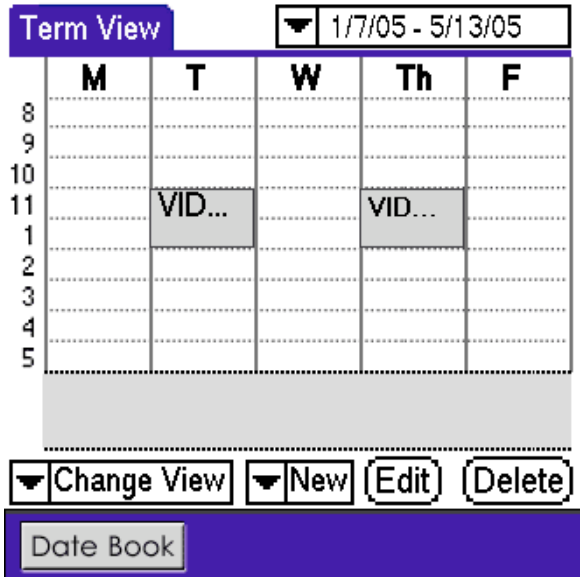

Figure 2.2.5:
New Term with One Event added.

Upon confirming the first event, the professor now views her updated semester. She can repeat steps 3 and 4 to add events to her semester and continue planning. If there were any conflicts, she can choose an event and Edit it, or Delete it from her schedule entirely.

2.2.3. Tradeoffs / More Issues to Explore

While this design proposal fixes a very salient breakdown noticed in the Contextual Inquiry and Design, this solution has some considerations that are outside of the scope of the redesign. First of all, the terminology to be used is not immediately clear — following the CD would promote a "semester" view, but there are schools that don't operate on a semester system. The notion of a "term" is somewhat more vague, but is our attempt to arrive at an all-encompassing label. Further CIs would be necessary to answer these smaller questions and also to determine how people plan and to validate this design suggestion, but based on the available CI data, this seems to be a salient design solution. Additionally, we acknowledge that the idea of a Term View would be more beneficial for people in academia than business people. Therefore, it is possible that this functionality could be encompassed in a separate plug-in that can be downloaded only if the user wants to use it.

A tradeoff to this design is that it adds a level of complexity to the Date Book. Making this feature an optional plug-in would alleviate some of that confusion, but there still might be a learning curve involved for new users.

2.3. Scenario #3. Scheduling Meeting Announcements Received via E-Mail

In this scenario, the professor is receiving an email announcing the Town Hall Meeting for her department. The email will allow her to add the event to her schedule automatically. In this scenario, the user has a conflict, so decides that she will leave the Town Hall meeting early and notes this on her Date Book. The primary inspiration comes from the CD Cultural Model – the user's scheduling tasks are influenced by others who inform her of meeting obligations. This appointment was likely scheduled without consulting the professor directly, which varies from Scenario #1 where the professor was directly involved in scheduling it.

2.3.1. Problems Encountered & Proposed Design Solutions

- **Problem**: Meeting times set without consulting user can create scheduling conflicts (CD: Cultural Model).

  Solution: We propose a networked calendar where user can check her schedule against a set event time (DI-04).

- **Problem**: User influenced by other people informing them of obligations or optional events (CD: Cultural Model).

  Solution: Other people can suggest appointments for users in the network; Palm notifies user of suggested appointments and checks user's schedule for conflicts. (DI-05).

2.3.2. Interaction Screen Shots

In the steps outlined below, we'll demonstrate how the Palm could be redesigned to integrate an emailed appointment proposal into the Date Book.

Step 1: View E-Mail Sent by Colleague.

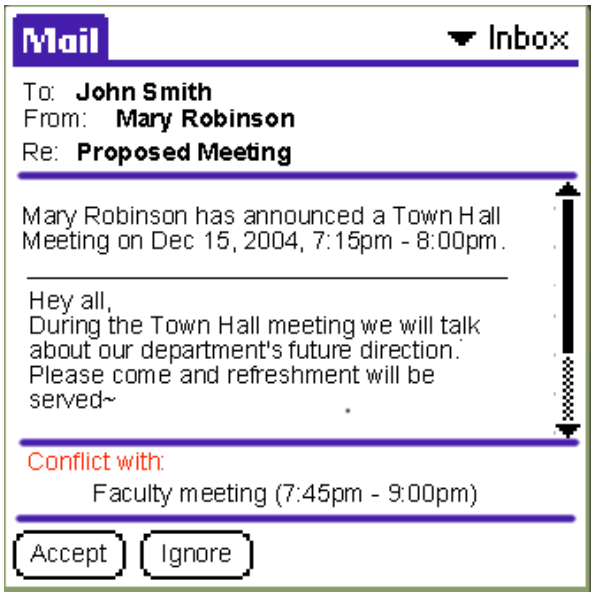

Figure 2.3.1:
Meeting Announcement E-mail with Scheduling Conflicts.

The user receives a system-generated e-mail with a message customized by the sender. The palm checks the user's schedule and notes any existing time conflicts. She can choose to accept the meeting or to ignore it if there is another meeting that she cannot miss. In this case, the user decides that she wants to go and leave early, so she clicks "Accept".

Step 2: Confirm New Appointment.

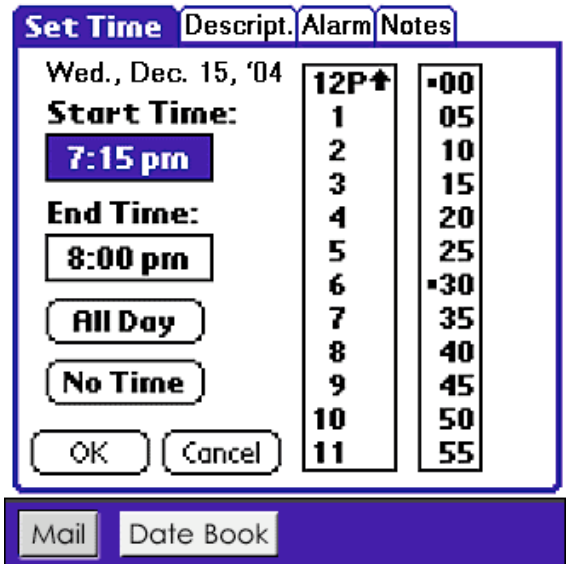

Figure 2.3.2:
New Appointment Screen.

Figure 2.3.2 shows the user at the standard New Event screen from the Day View of the Date Book. The professor decides to change the end time to 7:40 PM so that she can make her faculty meeting after the Town Hall meeting.

Minor Issues & Design Solutions in Step 2.

- **Problem**: There is no indication of the "Start Time" field being in focus for "Set Time" window (CW-02).

  Solution: Highlight start time to indicate that it is currently active (DI-20).

Step 3: Confirm Addition of Meeting to Date Book.

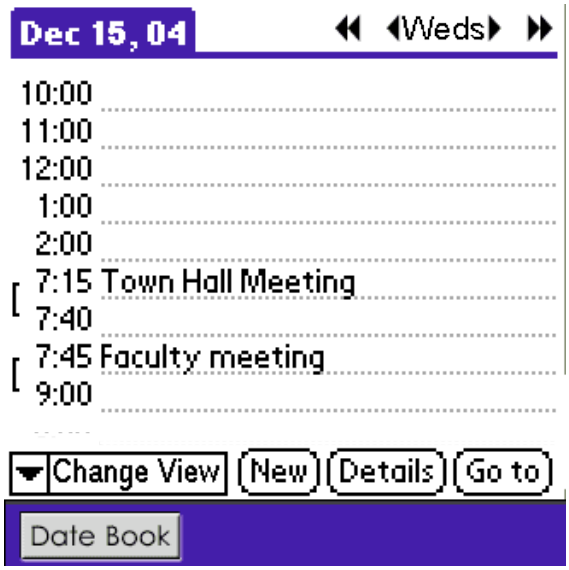

Figure 2.3.3:
Updated Date Book with Town Hall Meeting.

Once the user has confirmed the new event time, she is brought to the day view of the day for which she scheduled the appointment. This provides confirmation for the user that her appointment has been added to the date book. Also, from this screen the user could hit the Details button to access other features for their appointment, such as setting a reminder.

2.3.3. Tradeoffs / More Issues to Explore

This design takes a common task found by the user in the contextual inquiry — checking a set meeting against her schedule — and automates it within the Palm Pilot. This saves a fair amount of time and effort on the part of the user. However, it brings up a few new questions centered on the sender of the meeting announcement. What type of system support does she need? Would she benefit from the

system automatically tracking confirmations to these types of meetings? A CI on a member of the support staff would help in answering these questions.

Another issue is the difficulty we had in simulating received email in the Palm system. It was difficult to simulate this task in the emulator since we could not send an email; however, we intend for the email to work much the same way (including reply, forward, etc.); simply with the added capability of adding an appointment to the user's Date Book (or rejecting it based on a scheduling conflict). Thus, Figure 2.3.1 is a proof of concept primarily.

A final issue is the possible need to incorporate a prioritization for these different kinds of meetings. It may be the case that someone wants to accept the meeting but considers it low priority. This aspect was not implemented because it was not supported by the data, but we acknowledge that it might be something that users would find helpful.

A tradeoff with this design solution is considering how the process would work if the e-mail were received on a computer. Would it behave in the same way on the computer that it does on the Palm device, or would there be a difference in functionality? It seems feasible to support this design idea without overhauling every computer-based calendar system if the focus is kept on the Palm Pilot.

2.4. Scenario #4. Application Switching

When the user checks her schedule in the morning, a reminder pops up telling her that her daughter, Janine, has a birthday in two weeks. She switches to Memo Pad and writes herself a note about it. After writing the note, she switches back to the Date Book to see what her schedule for the day is like. Once she's done browsing her schedule, she quits out of Date Book and returns to Memo Pad. The need for this design idea comes from the Think-Aloud usability testing (TA-19), Heuristic Evaluation (HE-15), and the Cultural Model from the Contextual Design.

2.4.1. Problems and Encountered & Proposed Design Solutions

- **Problem:** Users cannot easily switch between applications. Every time a user wants to switch to a different application whether it is already open or not, he must go to the Applications page. (TA-19, HE-15)

  Solution: We are eliminating this extra step by implementing a taskbar that displays all applications that are currently open. Therefore, the user can now see what applications are open and with one click of the Stylus, switch to a different application. Since we are assuming our users are experienced with computer use and they are familiar with Windows or a similar operating system, they can recognize and efficiently use a taskbar (DI-25).

- **Problem:** Users are not sure which applications are open and which one they are working on. (TA-04)

  Solution: The taskbar helps to solve this problem, but we are also ensuring differentiation among applications on the taskbar by displaying the button representing the current application as "active" (DI-08).

2.4.2. Interaction Screen Shots

In the steps outlined below, we'll demonstrate how the Palm could be redesigned to allow for faster switching between applications.

Step 1: User Opens Date Book and Birthday Reminder Pops up

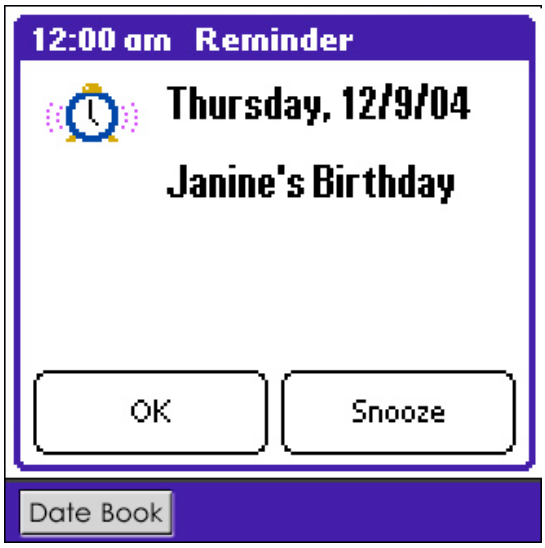

Figure 2.4.1:
Reminder screen for daughter's birthday.
Task bar now includes Date Book application.

The first thing that our user wants to do is to open up her Date Book to find out what her schedule is for the day. So, she goes to the Application page and selects Date Book. Before she can view her Date Book though, a reminder pops up that her daughter, Janine, has a birthday in two weeks. So, once the Date Book is selected, the system opens up the taskbar and places a tab in the taskbar that shows our user that the Date Book is open. Since the reminder takes up the whole screen, it is useful to have the current application showing on the taskbar so that she knows the Date Book application is still open. This plays into the second major theme.

Step 2: Go to Applications Page to Open Memo

Figure 2.4.2:
Applications Screen for Palm Pilot.

Not wanting to forget the birthday reminder, our user wants to make a note of it in the Memo Pad. So he switches back to the Application page and opens up Memo Pad. Here, the Applications page has changed slightly because the taskbar is now visible showing that the Date Book is open. After seeing the taskbar open, the user should expect this minor change.

Step 3: Enter New Memo

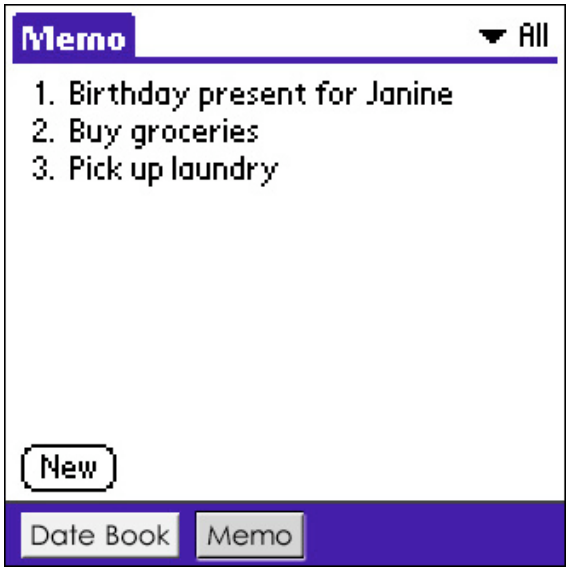

Figure 2.4.3:
Memo Pad with new note added for birthday gift reminder.

Another feature of the redesign is that when the user selects Memo Pad from the Application page and is in the Memo Pad application, a new tab appears in the taskbar to reflect this. It is placed directly next to the last opened application, in this case Date Book. This allows the user to see her actions take effect immediately.

Step 4: Enters Memo and Switches Back to Date Book

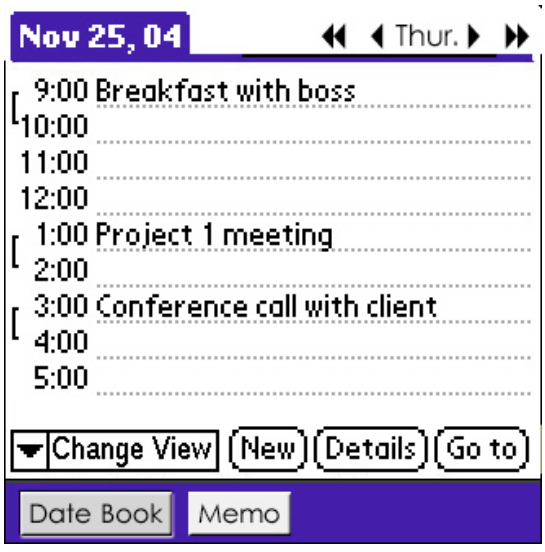

Figure 2.4.4:
Date Book screen. User switched back quickly with the taskbar functionality.

The user enters into Memo Pad that she needs to buy her daughter a birthday present. Then she switches back to Date Book to check her schedule for the day, but keeps Memo Pad open so she can

come back to the task of buying the birthday present. Here, the user needs to keep both applications open because she needs information from both. After entering in the memo, she wants to check her schedule so she quickly switches to Date Book. She can do this without going back to the Application page, like the current Palm Pilot requires the user to do. Instead, she can just click the application she wants to switch to on the taskbar.

2.4.3. Tradeoffs / More Issues to Explore

The tradeoff of implementing the taskbar is that it takes up important screen real estate thus the current application loses a portion of what is displayed. We feel the taskbar is worth losing some screen real estate because it allows for more efficient use of the Palm, but this does not mean that other solutions could be thought of to maintain screen real estate while still having the taskbar. Our solution would be to perform Think-Aloud usability testing on this new design to see if the problem identified above is fixed without introducing further issues.

Another issue to explore further, which our scenario did not support, is when there are more applications open than can fit on the taskbar. We considered implementing left and right arrows to allow scrolling through the applications in the event that the user opens more than three applications. Since most users will probably not use more than three applications, we left this as an issue for future consideration.

3. Retrospective

Designing an interface will never be the same. It no longer consists of having the best idea on the team and developing it straight away; while this ideology still manages to persist, it ignores the most important phrase in Human-Computer Interaction: "The user is not like me." In the spirit of the HCI mantra, a number of methods and concepts were explored throughout this semester in the hopes of keeping the user's needs at the forefront of the design. These methods include Contextual Inquiry & Design (CI/CD), Model Human Processor (MHP), Heuristic Evaluation (HE), Cognitive Walkthrough (CW), Think Aloud (TA), and Interface Relabeling (IR). In effort to evaluate the effectiveness of these methods, the cost of conducting each method will be compared to the benefits accrued. Analysis of the cost will take into account not only the perceived monetary cost of

running this in a business setting, but also the time and manpower necessary to conduct the method effectively. Analysis of the benefits will consider the amount of design-relevant information and the importance of this information (determined by its usefulness in the redesign). Finally, the value of the method will be discussed in relation to the other methods used this semester.

3.1. Contextual Inquiry & Design (CI/CD)

Contextual Inquiry involves interviewing a person in the context of her work with the goal of creating a product to support that work; Contextual Design develops a series of models to convey the information gleaned from the inquiry. Effective CI/CD seems relatively costly — the inquiry requires not only the time of one or two members of the design team to conduct the inquiry, but also the commitment of a potential user willing to let the design team impose on their workspace for a period of time. Iterated over 10 – 20 inquiries, and suddenly the time invested is quite immense. Contextual Design requires at least half a day per user (depending on the length of the inquiries) to construct her work models, plus a period of time to consolidate the models derived from each inquiry. Since effective CD suggests that the entire design team be present, the number of labor hours required is quite large.

Nevertheless, the benefits accrued from CI/CD are worth the effort. First of all, CD affords the capability to visualize the user's work, which will be useful throughout the design process. Furthermore, 26 breakdowns were recorded from the CD overall. As seen in Appendix A: Design Ideas and Support, a vast majority of the completely new design ideas in this project came from the these breakdowns (e.g., the design idea of Date Book "Term" View comes from the "redundant calendars" breakdown in CD), indicating that the design-relevant information found in the CD is relevant on a global level to the software. In contrast, there weren't any local design changes (e.g., improper labels for buttons) found from the CD data.

However, it is not the case that *all* breakdowns from the CD are design-relevant. In fact, the affinity diagram used for Group 8's redesign consisted of a group of irrelevant points that were primarily from the CD; in short, breakdowns existed in her work that were *not* relevant to scheduling (e.g., user misheard a student question, CI transcript 302-304). While in the other methods it is possible to differentiate between relevant and irrelevant information (using UAR ratings), it is impossible to do so with the CD as learned in this class — at first look, each breakdown has equal value. Although the value of having a holistic view of

the user's work was stressed in class (suggesting all breakdowns should be kept on the models), there was no discussion of how to treat a breakdown that had been officially deemed "irrelevant" at the end of the modeling process — seemingly an important step in larger projects than the one completed in class.

Summary. Despite the time and effort necessary to conduct the CI/CD process, and despite a nominal amount of irrelevant information, the CI/CD is a very powerful method for generating innovative design ideas. As a formative technique, the greatest value of CI/CD is at the beginning of the design process, but the CD models are relevant throughout the process.

3.2. Model Human Processor (MHP)

The Model Human Processor discussed some Psychological factors relevant to how the humans process information. While using principles from MHP costs little, it does require sufficient knowledge of the principles before they can be effectively applied. Thus, not everyone on the design team will be able to use it. For instance, in Scenario 1 (step 2), we employed Fitt's law in designing the contact selection screen to ensure the easy selection of contacts. This usage indicates the Model Human Processor is more effective as a tool to make good design decisions on the fly; while it could be used to evaluate a prototype, it seems far too open-ended. Methods like GOMS or Cognitive Walkthrough (which are based on MHP) would seem to be more effective to evaluate a prototype since they will result in concrete UARs for the design team to work on.

Summary. MHP is useful for making decisions throughout the design process, but it is not as powerful for evaluating a piece of software (or a prototype) since it is somewhat open-ended and since there is no clear process to derive UARs from the MHP. Instead, MHP seems to give justification to UARs that come from the other methods learned in class.

3.3. Heuristic Evaluation (HE)

Heuristic Evaluation involves a team of evaluators evaluating a piece of software using certain design rules of thumb. As a discount usability method, HE is relatively inexpensive. Finding problems in the interface is a relatively short process; writing the UARs seems to be the longest leg of the process.

In conducting the Heuristic Evaluation, we found 51 unique problems in the Palm interface — a far greater number than the group found using any other methods. The average severity rating for the problems was

2.22, suggesting that the problems found were of average significance (about the same as CW and TA). At first look, then, it appears that the HE is quite effective in finding usability problems.

Nevertheless, we felt that the problems discovered with the HE tended to be more nit picky, despite the average rating of "minor usability problem." This discrepancy is likely due to the lack of restrictions on the HE; the task is simply to "Go forth and find problems," as opposed to other methods where the task is far more structured. Furthermore, the group noticed a tradeoff between strict adherence to the standard heuristics and innovation. At times, an interface breaks a heuristic on purpose, either to achieve a desired end or simply as a mode of expression. So, it seems important to add structure to the HE as a whole; part of this structure should involve thoughtful consideration of the interface, and perhaps the rejection and/or addition of heuristics beyond Neilsen's standard 10.

Summary. Heuristic Evaluation is a cost effective method to arrive at a large number of problems in an interface in a short period of time. Although Nielsen's standard 10 heuristics might not be ideal for every interface, the process involved is certainly relevant to any interface one might encounter. Even if more structure or preparation is necessary before conducting an HE, it is still more cost effective than some of the other methods explored in this class.

3.4. Cognitive Walkthrough (CW)

Cognitive Walkthrough involves a task-based approach to a particular interface. Each task is broken into steps and thoroughly scrutinized. Depending on the tasks chosen for a particular interface, the CW can take longer than the HE; however, it does not require test users or require any cost beyond the time of the evaluators.

In conducting the Cognitive Walkthrough, Group 8 found three problems with an average severity rating of 2.0. While the problems were approximately as severe in the CW as in the HE, there were far fewer issues discovered. Scaling this finding to twelve tasks (a reasonable number considering the breadth of capability in the Palm Pilot) still yields 42 problems, which falls short of matching the number of problems found in the HE. Still, while there are certainly fewer problems with the same severity rating (on average), these problems seem to have greater urgency with respect to novice users. Therefore, applying priority, CW problems would seem to take precedence before HE.

Another significant point encountered in the CW is that it deals with hypothetical users engaged in *exploratory learning*. This assumption served as suitable criteria for a number of success stories. Was a particular label clear? No. Would the user find it under the mantra of exploratory learning since it's the closest thing to what she wants? Yes. Therefore, the Cognitive Walkthrough seems to test whether a particular interface is "good enough" rather than ideal; perhaps giving credence to the notion above — if the interface fails to pass the "good enough" test, then those problems should be fixed with haste.

Summary. While Cognitive Walkthrough takes a fair amount of time to complete for a large number of tasks, it is cheaper than other methods and avoids the need to recruit participants. Unlike other methods, it seems to return whether an interface is "good enough" for a user engaged in exploratory learning, rather than testing for an ideal interface. This isn't necessarily a negative aspect; in fact, it gives more credence to the validity of the UARs found during the process.

3.5. Think Aloud (TA)

Think Aloud Usability Testing involves giving participants a number of tasks and asking them to think aloud during the process. While it involves a fair amount of preparation and analysis time for the testers, as well as the recruitment of participants, the actual process usually only takes an hour or two. Overall, TA takes longer than HE or CW, but the higher quality of the results (as discussed below) makes it worthwhile.

Overall, Group 8 found 21 unique problems in the Think Aloud session, with an average severity rating of 2.33. While this is slightly higher than the other methods, this is not completely valid for comparison since not all UARs were rated in this assignment.

The UARs found in the Think Aloud seem to have a very salient tradeoff: On the one hand, problems found seem to be more pertinent since they are found by actual users; on the other hand, users are not all the same, and it might be the case that one user (out of many) struggles with a part of the system that was irrelevant to the original task (e.g., Preferences). Given this tradeoff, it seems quite important to not necessarily value every TA UAR higher than those found in other methods.

However, there are cases when the TA would seem to trump the other methods — namely, whenever a good aspect goes against a finding from a previous method's problem report. In the Think Aloud, Group 8 noticed that the user did not have a problem with the Set Time screen, even though the Cognitive

Walkthrough had suggested that the user would not know which task he was meant to perform. Still, this did not effect the redesign substantially; while the group acknowledged the discrepancy, it was determined that *one* user's success did not trump the general finding in the CW; more think aloud studies would need to be done before determining that the issue was not a problem.

Summary. Think Aloud Usability Testing gives the design team a good glimpse into the user's experience with the software, and helps them to make changes that are helpful to the user in the long run. However, since there is a great discrepancy with users, one user is not enough to draw any significant conclusions; usability testing should be done with as many users as possible in order to make sure the problems addressed are truly problems for a majority of users.

3.6. Interaction Relabeling (IR)

Interaction Relabeling involves taking an object that is completely different from the object being redesigned and redesigning the interface through that object. While this is a very inexpensive and quick method to perform, it did not have a great influence on the final design solution. It did give the group members an opportunity to step back and view the Palm Pilot at a more holistic level, but in completing our redesign, the IR exercise was never referred to.

Summary: While it is entertaining, Interaction Relabeling is not very inspiring to our design process.

3.7. Closing Remarks

While it is true that every method is important in evaluating the usability of the interface, it is certainly not the case that every method contributes on the same level. As noted in Section 3.1, Contextual Inquiry and Design tend to suggest design recommendations that result in larger-scale solutions. These solutions often include completely new modules that did not exist in the original design. In the case of a completely new system, CI/CD would be appropriate since it acts primarily as a formative method in the first place.

The other methods practiced in homework were summative in nature, and therefore tended to give more minute design changes, including issues with labels, affordances, and visibility. Although on average, Think-Aloud usability testing resulted in the most severe UARs, followed by HE and then CW, we felt that Cognitive Walkthrough resulted in the most actionable items since, as discussed in Section 3.4, UARs that result from CW are instances where the interface failed to pass the "good enough" test.

Perhaps the most valid point off of which to base a redesign effort is one that arises from multiple methods. In Section 3.5 we discussed the case where a Think Aloud Good Aspect Report might trump an earlier Problem Report that contradicted it. On the other hand, a Problem Report that is found by multiple methods seems to be more actionable since it is found under multiple contexts.