Palm Redesign – Group 10

Executive Summary

Over the past four months, we analyzed the Palm Pilot interface and discovered many existing interface problems. Now, we propose design ideas to improve it and create a better product. The interdisciplinary background of the team is diverse; we have knowledge in anthropology, computer science, cognitive science, linguistics, interaction design and cognitive psychology. We decided to focus our redesign on the *Date Book* application because data from all of the methods can be applied. We noticed the majority of usability issues came in the *Date Book* because this application was our primary focus while applying the methods. The redesign addresses issues of controls, visibility, labeling, and screen organization in this application. Our report includes a narrative that describes a fictional user, Simon Newell, who creates an appointment with an alarm reminder using our improved interface. As Simon goes through each step of his task, we describe each of our redesign ideas. We have also included a retrospective of this project discussing the effectiveness of the HCI methods we used: Contextual Inquiry (CI), Contextual Design (CD), Heuristic Evaluation (HE), Cognitive Walkthrough (CW), Design Re-labeling (DR), and Think-Aloud protocol (TA).

Redesign

Focus

We have decided to redesign the task of scheduling an appointment. Scheduling appeared in the contextual inquiry (CI) and was part of our work redesign. Additionally, we created an affinity diagram on all of our usability aspect reports (UARs) and found that scheduling was a common theme in all of the HCI methods we performed.

Scenario

Today is Saturday the fourth. Simon Newell needs to add an appointment from 7:15 AM to 8:45 AM next Monday the sixth. He needs to set an alarm for 10 minutes before the appointment so he has enough time for travel.

Step 1: *Simon navigates to the calendar application (Fig 1).*

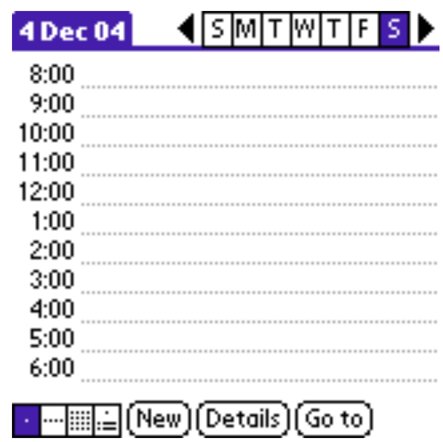

Fig 1. Original day view screen

Design Change 1 – Vertical week widget arrows

Current Design:

In the *Day View* of the *Date Book*, the *Week Widget* depicts the current day in the context of the current week (Fig 2). To either side of the widget, there are right and left arrows, which function to move forward and backward to the next and previous weeks, respectively.

Problem:

Our cognitive walkthrough (CW) found this was a problem because users can erroneously believe that the left and right arrows adjust the current day rather than the current week (G10-CW-01). The arrows map naturally to moving the selection to the next or previous day because the arrows are pointed right and left. When the user taps an arrow on the *Week Widget*, the *Day View* instead adjusts by one week.

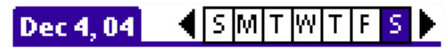

Fig 2. Original week widget

The widget does not provide any feedback for this action which is a violation of the "visibility of system status" heuristic. Feedback for a week change is only indicated by a change in the current date label at the other end of the screen. The user must remember the previous date and calculate that the date has changed by 7 days.

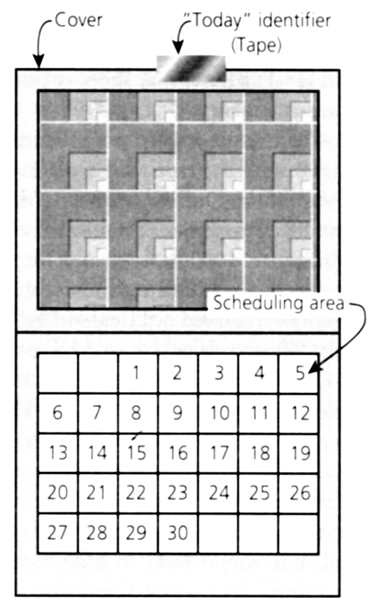

Fig 3. Monthly Calendar Artifact (Beyer/Holtzblatt, 180)

From our CI, we found that users commonly arrange weeks vertically, as in a month view (Fig 3). Therefore, up and down movement matches this conceptual metaphor. The current *Week Widget* uses right and left arrows to provide this navigation, which does not match well to the user's artifacts. (CI-01, Physical Model, Wall Calendar).

Solution:

We replaced the left and right arrows with up and down arrows, respectively (Fig 4). The up arrow moves to the previous week and the down arrow moves to the next week. The new arrow placement matches the layout from the artifact models where weeks flow from top to bottom.

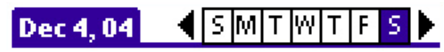

Fig 2. Original week widget

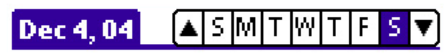

Fig 4. Redesigned week widget

The feedback of advancing the date should be improved. Our solution is to animate the changing of the week so that tapping down "scrolls down" to the next week. Similarly, tapping up "scrolls up" to the previous week (Fig 5).

Fig 5. Redesigned week widget animation

Tradeoff:

Since the new arrows are not diametrically opposed, it may not be clear that they represent opposite actions. In addition, the up and down arrows can cause confusion with the scrolling arrows in design change 2. We have attempted to mitigate this effect by placing the arrows in a bounding rectangle with the widget.

Scenario

Step 2: *Simon navigates to the next week by tapping on the down arrow in the week widget.*

Step 3: *Simon then taps on the 'M' button in the week widget to select Monday.*

Design Change 2 – Occluded times and scroll arrows

Current Design:

The current *Date Book* application only displays time from 8:00 AM to 6:00 PM in the day view (Fig 6). To add a time outside those displayed, the user must either tap a displayed time (to bring up the set time dialog) or tap *New*.

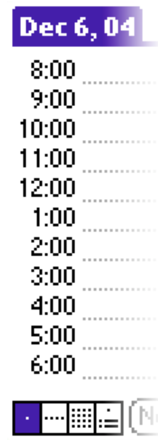

Fig 6. Original *Day View* time labels

Problem:

During our think-aloud study, we found that the novice user had great difficulty adding an event time outside of those displayed in the *Date Book* application. The user attempted several methods such as navigating through different views of the *Date Book* and opening other applications before finding an appropriate way to enter specified time. This indicates to us that adding times outside those displayed is difficult for users (G10-TA-03).

Solution:

We propose an up arrow above the topmost displayed time and a down arrow below the bottommost displayed time (Fig 7). During the TA, the task was to add an appointment at 7:00 AM. The user attempted to scroll upwards by clicking on the area above 8:00 AM (TA-Video1-1:59). These new arrows allow the user to quickly access a time not currently displayed on the screen.

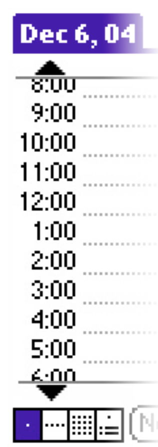

Fig 7. Redesigned *Day View* time labels with scroll buttons and occlusion

We partially occlude the topmost displayed time and the bottommost displayed time as affordances to suggest times exist beyond those displayed. To further emphasize the occlusion, gray lines bound the times displayed on the screen.

Tradeoff:

By adding the arrows, gray lines, and occlusion, less space is available on the Palm for scheduling information. Each of the arrows take up one line, and the occlusions together take up another line. These additions also add to the visual clutter on the screen.

Scenario

Step 4: *The Palm currently starts the day at 8:00 AM. Since Simon needs to set an appointment before this, he scrolls up by tapping on the up arrow.*

Step 5: *Simon then taps on the 7:00 time label to create an appointment around 7:00 AM (unchanged aspect of the interface). The "Set Time" screen appears*

Design Change 3 – *Set Time* controls reorganized

Current Design:

When the *Set Time* dialog box is launched (Fig 8), *Start Time* is selected by default. The user goes through the following steps:

1. Set the *Start Time* by tapping on the required hour and minute fields in the column on the right.

2. To enter the *End Time*, tap on the field below *End Time* label.

3. Set the *End Time* by tapping on the required hour and minute fields in the column on the right.

Fig 8. Original *Set Time* screen

Problem:

Our CW predicted that users may not understand that the hour and minute columns are modal controls that can be applied to both the end time and start time (G10-CW-02). Additionally, the TA data suggests that the user might miss the *End Time* function (G10-TA-12).

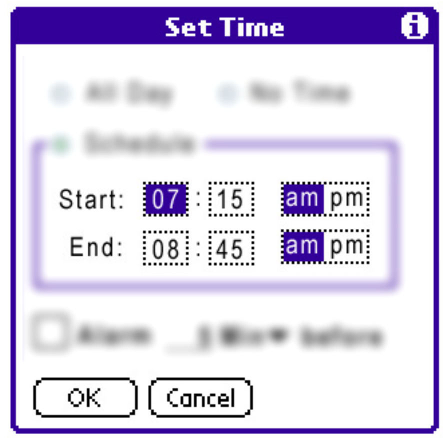

Fig 9. Redesigned *Set Time* screen – Start and End time widgets

Solution:

We have placed time fields next to both *Start Time* and *End Time*. Our hypothesis is that the proximity of the time fields to the labels will make the association between them clearer to the user (Fig 9). Also, instead of a time column for hours and minutes in the dialog box, we propose a *drop side-ways menu widget* (Fig 10). This widget drops to the side of the selected field so that a list of 12 elements can be displayed. The selected item is highlighted to indicate the relationship to the user.

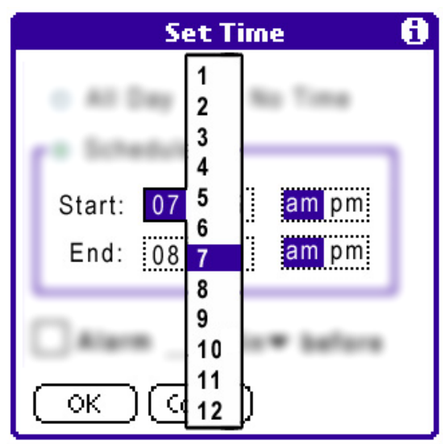

Fig 10. Redesigned *Set Time* screen – drop sideways widget

In the redesign, *Start Time* and *End Time* both have their own time controls. This is an improvement over the old interface, where the time controls are modal; their effects depend on whether *Start Time* or *End Time* are selected.

The proposed layout uses less space compared to the current layout of interface elements. The time fields now have AM/PM distinctions. This eliminates scrolling because 12 hours of the day can be presented on one screen while 24 hours cannot. This solution preserves a good aspect of the existing design: Graffiti is not needed to enter event times (G10-HE-53).

Tradeoff:

The proposed *drop sideways* menu is inconsistent with other drop down boxes in Palm UI. Also, when the *drop sideways* menu is active, it hides some information on the screen behind the widget.

Design Change 4 – *All Day*, *No Time*, and *Scheduled*

Current Interface:

The current Palm interface places two buttons, *All Day* and *No Time* on the *Set Time* screen (Fig 11). The *All Day* button dismisses the screen and sets an event from 8:00 AM to 6:00 PM. The *No Time* button also dismisses the screen and creates an event with a diamond glyph.

Fig 11. Original *Set Time* screen – *All Day* and *No Time* buttons

Problem:

One problem with these buttons is that the visual layout is not semantically clear (G10-HE-36). The *All Day* and *No Time* buttons affect the current *Start Time* and *End Time*. However, this relationship is not represented in the visual layout of the screen.

Currently, the buttons are set apart from the start time and end time boxes. Another problem is that the states of *Scheduled*, *All Day*, and *No Time* are mutually exclusive; only one can be selected. Push-buttons do not visually communicate this exclusivity.

Solution:

It is a GUI convention to show mutually exclusive choices as radio buttons. So, we have modified the interface by changing the *All Day* and *No Time* buttons into radio buttons. We have added a third button, called *Schedule*, to make event scheduling an explicit choice rather than an implicit choice. These three buttons have been clustered at the top of the screen along with a box to better indicate that they relate to the start and end time (Fig 12).

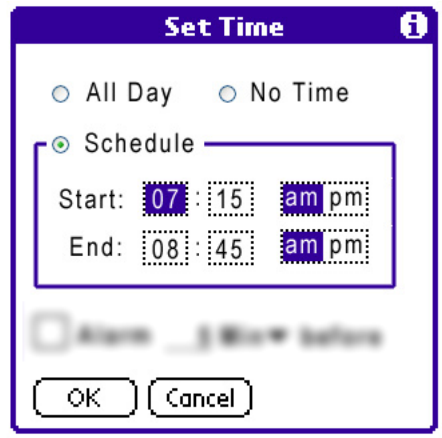

Fig 12. *Set Time* screen with reorganized time options.

The screen will initially display with the *Scheduled* button selected. When users select the *All Day* button, the time settings will display 8:00 AM to 6:00 PM. When users choose the *No Time* button, the start time and end time will display nothing. In both cases, the time settings will gray-out to indicate inactivity. This behavior gives the user more feedback about the function of the buttons.

Tradeoff:

Placing the three radio buttons above the time settings better indicates their function. However, any buttons placed at the top of the screen are more likely to be read first by users. This is a tradeoff because the *All Day* and *No Time* buttons are expected to be used less than the time settings. Another tradeoff is that the use of the radio buttons require an extra tap in the task when setting an *All Day* or *No Time* appointment.

Design Change 5 – Added alarm settings to *Set Time* screen

Current Interface:

In the current Palm interface, appointment alarm settings are only available in the *Event Details* screen.

Problem:

The alarm setting adjusts the timing characteristic of an appointment. However, the *Start Time* and *End Time* are presently placed on one screen while the alarm setting is placed on another. Our TA indicates that the user had difficulty locating the alarm settings (G10-TA-09, G10-TA-14). In addition, the default for alarm units is always minutes, even when a *No Time* event is selected. Setting an alarm for a few minutes before a *No Time* event does not make much sense since it would likely alert the user a few minutes before midnight the day before the event. We feel this is a problem, but did not originally notice it during the heuristic evaluation.

Fig 13. *Set Time* screen with new new alarm controls.

Solution:

Since the space required for time settings has been reduced, we can now comfortably fit the alarm settings in the *Set Time* screen (Fig 13). The new alarm settings operate similar to those in the *Event Details* screen. When the alarm checkbox is not selected, the alarm options appear grayed-out rather than hidden from view. We have done this because keeping the available options visible provides more cues for the alarm controls.

We have also added the word "before" to the dialog to indicate that the alarm will trigger a set amount of time before the scheduled event. In order to fit the extra word, we change the "*Days, Hours, Minutes"* drop-down menu to *"Days, Hrs, Min"*. In addition, the alarm settings currently default to a unit of minutes for all types of events. We change this so that the default units for *No Time* events is now days instead.

Tradeoff:

Adding the alarm settings to the *Set Time* screen creates more visual clutter. Abbreviating the alarm time units may cause some confusion.

Scenario

Step 6: *Simon makes some adjustments to the Time Settings. He sets the Start Time minute field to 15. The end time reflects the change to keep the meeting duration the same (unchanged). Simon then sets the End Time minute field to 45. With the appointment time set correctly, Simon now taps on the checkbox next to alarm and enters 15 into the field. Happy with the changes, Simon taps the OK button. The Day View screen reappears and features a flashing cursor where Simon can enter a name for the appointment.*

Design Change 6 – Re-label *Details* button to *View*

Current Design:

In the *Date Book* application, the current Palm interface includes a *Details* button that is flanked directly by two other buttons: *New* and *Go to*. The *Details* button is also adjacent to *graffiti shift indicator* and the four buttons that change the *Date Book* view mode (Fig 14). If the user wants to view more information about a particular event, he needs to select a particular appointment and then tap on *Details*.

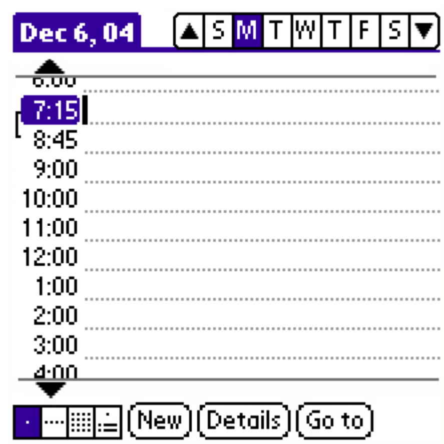

Fig 14. Original *New, Details*, and *Go to* buttons

Problem:

The problem with this design is that the details button can easily get lost because it is surrounded by a great deal of clutter – seven other objects exist within the same horizontal range (G10-TA-09). More importantly, the details button is not as semantically "action-oriented" when compared to the two surrounding buttons. (G10-TA-08) The *Go to* and *New* buttons have stronger scents for their respective behaviors since they respectively convey the idea of moving to another location, and creating something that did not exist previously. Our CW predicts that users may experience problems understanding the meaning of the *Details* button (G10-CW-03).

Solution:

Our solution to this problem is to change the *Details* button label to *View*. We have made this labeling change because *View* is a verb; it denotes action and thus makes a better button label. Also, changing the term from *Details* to *View* eliminates three letters, which helps to decrease screen clutter. To better communicate the usage of the *View* button, it will be disabled and grayed out until an event is selected (Fig 15). When an event is selected, the event time and *View* button are both highlighted orange (Fig 16). This helps the user associate the button with the event. In addition, the use of color helps increase the redundancy gain of the button, which thereby

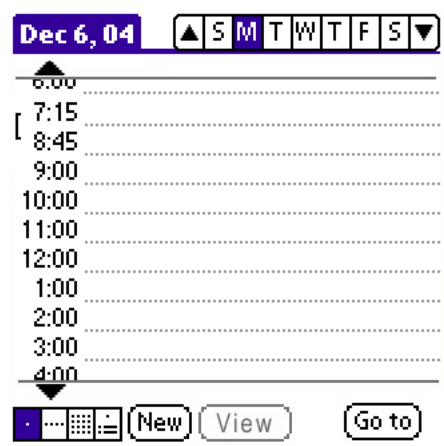

Fig 15. Redesigned *New, View*, and *Go to* buttons

increases the chances that the association will be recognized[1]. We have moved the *Go to* button to the right side of the screen because the *New* and *View* buttons both operate on appointments, and the *Go to* button is used to change the current day. Although this change is not motivated by data, we feel that the separation of the buttons will help users to group button functionality. The change in button organization helps the user understand the screen quickly and most importantly reduces the clutter. This minor change does not violate any heuristic.

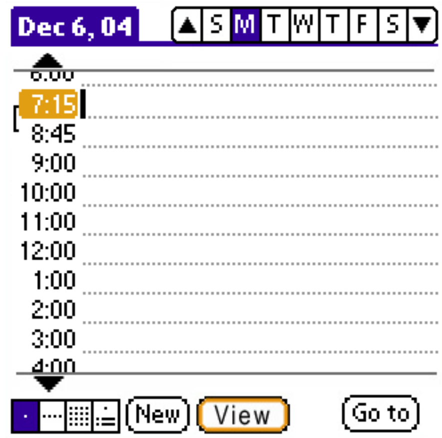

Fig 16. Redesigned *New, View*, and *Go to* buttons – selected event

Tradeoff:

Although we believe the term *View* is an improvement over the term *Details*, it still does connote a sense of passiveness. For example, the user can view and read an appointment, but she might not be able make any new changes to it. In addition, the use of a colored button is not consistent with current Palm UI standards.

Scenario

Step 7: *Before Simon can enter a name for his new appointment; he accidentally taps on another part of the screen. The system automatically deletes his scheduled appointment.*

---

1 Introduction to Human Factors Engineering, Wickens, Christopher D. et al. Prentice Hall, 2004.

Design Change 7 – Don't delete unlabeled appointments.

Current Design:

When the user taps next to a time in the *Day View*, the Palm creates an appointment. If the user then taps next to another time, the newly created appointment is automatically deleted. This is a good feature in that it allows the user to recover from an error quickly. However, if the user enters the time for an event through the *Set Time* screen, the Palm will delete the entered data if the user taps elsewhere on the screen before entering a label.

Problem:

This is a problem because the Palm destroys user's work without prompting or allowing for recovery (G10-TA-06). The Palm provides no feedback that the appointment was deleted. The user may want to wait until later to enter an appointment name. However, the Palm prevents the user from creating an appointment with no name by automatically deleting unlabeled appointments.

Solution:

If the user explicitly sets the *Start Time*, *End Time,* or other appointment information, the system should not automatically delete the appointment. Instead, the system should keep the appointment, and affix the label "unlabeled appointment" (Fig 17). This label prompts the user to label the appointment.

Fig 17. Redesigned behavior – unlabeled appointment.

Tradeoff:

The main tradeoff to this problem is that there may be more clutter on the screen in the form of unlabeled appointments. This is similar to the "New Event" issue as seen in the CI data (L327-L350). The other tradeoff is that a user may set up the boundary times of an appointment, but then decide that they do not actually want the appointment. In this instance, the user would need to select the unlabeled appointment and delete it rather than just tapping on a new time entry.

Scenario

With our change, Simon's work is not lost. His appointment is automatically labeled as "unlabeled appointment". Simon can now click on the label to edit it as desired.

A few days later, Simon wants to check on his appointments for the following day. Simon is in the car and does not want to take out the Palm stylus.

Step 8: *Simon presses the physical Date Book button and is brought to day view. He then tries to use the physical arrow buttons to see his appointments for tomorrow.*

Design Change 8 – Add physical JoyPad

Current Design:

The current Palm device features a pair of up and down physical buttons, which provide different functionality in different contexts (Fig 18). In the *Day View* of the *Date Book* application, the user can use the up and down buttons to move to the previous and next day, respectively. Our HE shows that the orientation of the buttons does not match the interface elements in this context (G10-HE-02).

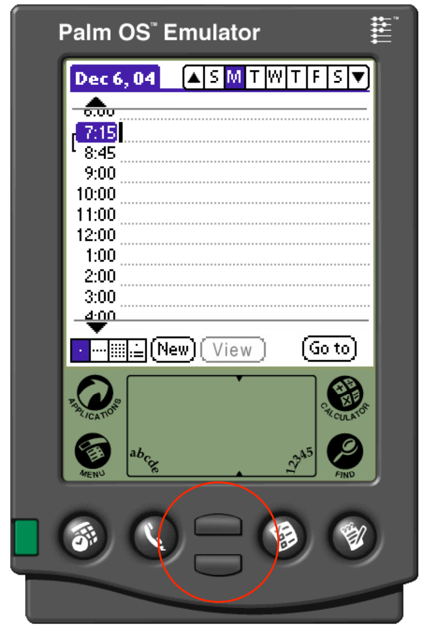

Fig 18. Original physical buttons

This mapping is poor for two reasons. First, pressing the up and down button moves the day selection in the week widget left and right. Second, artifact models from our CI showed that days flow from left to right. The current up and down buttons do not correspond to user's left-to-right conceptual mappings in the context of changing days (CI, Artifact Model).

Solution:

Our redesign replaces the two physical buttons with a 4-way JoyPad (Fig 19). In the *Day View* of the *Date Book*, the user can use the left and right arrows to move the view by one day. The up and down arrows can be used to scroll the view to earlier or later times of the day. The JoyPad functions as a redundant control for the onscreen controls described above.

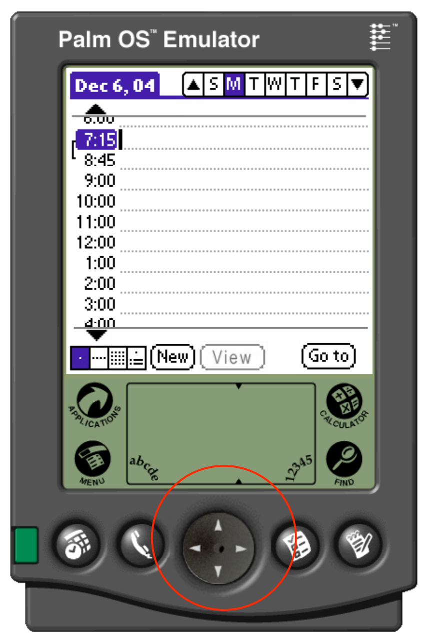

Fig 19. Redesigned physical buttons

The JoyPad is also effective in situations where the user needs to scroll, but removing the stylus from the Palm chassis and tapping buttons on the screen is too cumbersome. The onscreen scrolling controls are tiny and are hard to tap because they are only a few square millimeters in size. Our hypothesis is that the JoyPad would be faster in many situations. We recommend using the GOMS method to validate this hypothesis.

The JoyPad adds an additional horizontal control that fixes the flawed day mapping in the *Date Book*. It will also benefit applications that want to provide additional left-right movement using physical buttons.

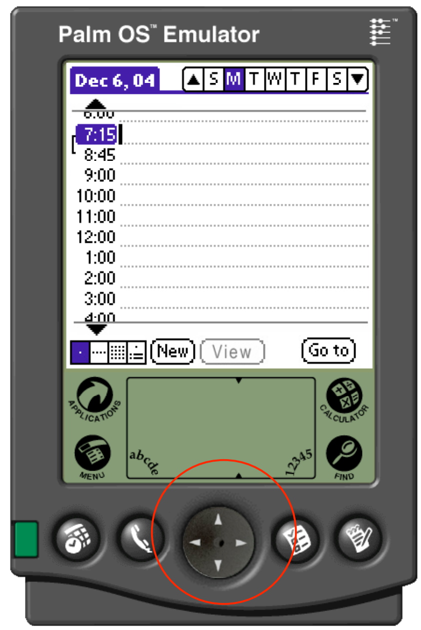

Fig 19. Redesigned physical buttons

Tradeoff:

The JoyPad does have a drawback in the day view context of the *Date Book*. The up-down controls can map to 2 possible actions: scrolling the time of the day, or scrolling the current week. However, if the user has an incorrect conceptual model and tries the up-down controls, she will see the entire screen scroll to show different times. This feedback is immediate, very salient, and will quickly correct the user's mental model. Since this drawback is so minor, we believe that this change is an improvement over the existing controls.

Scenario

Step 9: *Simon is able to use the JoyPad by pressing right to move to the next day. He scrolls up to see his 7:15 appointment.*

Retrospective

Overview

In the past four months, we have learned and practiced a set of methods: contextual inquiry, contextual design, heuristic evaluation, cognitive walkthrough, think-aloud protocol and design re-labeling. Unfortunately, the time constraints of the semester did not permit us to practice the GOMS method on the Palm Interface until after we completed this report. Overall, we found that analyzing the think-aloud participant study provided us with the most convincing data for our redesign. In terms of usefulness, cognitive walkthrough was ranked second due to vast amount of time required to produce three UARs. We ranked heuristic evaluation third only because our data was scattered over the entire interface; If we had previously agreed just to focus on the Date Book, then we would have probably collected relevant data from this method. We deemed the CI/CD data fourth in usefulness because the ratio of data generated to data used was low; We only used selected parts of our CD data to support our redesign because we lacked the time to use the data to radically redesign the work. We ranked redesign re-labeling fifth most useful to our redesign process simply because we were not able to generate useful redesign ideas from this activity.

Contextual Inquiry (CI) / Contextual Design (CD)

The data from the contextual inquiry method allowed us to gain a valuable perspective on how users manage scheduling. The analysis through contextual design contributed radical new design ideas such as an automatic synchronization design we developed while we walking the models. These models offered us rich data about scheduling, without the real-world overhead of conducting the CI. Since we had access to a single interview, the models produced were focused on one participant's perspective. At the same time, some of us were unaware of the Palm's scheduling capabilities, and this let us explore innovative avenues for scheduling by observing how the participant schedules in her own life. Lacking of familiarity with Palm Pilot scheduling helped us think out of the box. Because we were not completely familiar with the Palm interface, we were able to be more open in our redesign ideas during the CD. This particular method is wider in scope than other techniques, because it focuses on the scheduling task holistically.

The majority of the CD models were not used in the final redesign, because they were not wholly focused on scheduling and the Palm. The models did, however, provide us with valuable insights into how scheduling plays a role in a user's work. We did find that the physical and artifact models were very useful in our final redesign, but we did not extract useful redesign data from the cultural, workflow or sequence models. Compared with other techniques, much of the collected data was not used. For example, the CI participant spent a great deal of time on the phone with T-mobile customer service which was good data, but not helpful to our final Palm redesign. Also, when the CI returned to the task of scheduling, the investigator in the film remained quiet and did not seem to probe the participant enough on scheduling tasks such as her extensive use of iCal. It seemed as if we prepared much analysis that was not used in an end product. In addition, the limitations posed by the redesign assignment forced us to only redesign the palm. Thus, we could not redesign other aspects of scheduling that would support a user's work. In addition, because we were novices at the technique, the process was long and arduous. With additional practice, we believe the technique will be faster and generate better data. The contextual interview and contextual design methods were the least useful in the amount of data that they created, because we only used a small portion of the data in the final redesign. However, they were extremely useful for seeing "the forest through the trees". We better understood the nature of a user's work in scheduling.

At first, recording all details in the contextual design did not seem useful. We realized later how valuable this technique could be. Preserving details allows for emergent patterns to be detected in data, which were previously thought to be irrelevant.

Heuristic Evaluation (HE)

We found that the heuristic evaluation method (HE) is useful because it easily discovers obvious and serious issues; it is an excellent usability method to use upstream in the design process if used before more expensive methods such as CW or TA.

Formally, evaluators in the HE process must write UARs for problems to explain the issues and provide potential solutions. The process of writing UARs forces evaluators to think about why issues are problems and how they violate the heuristics. This makes the data more objective since based well-known criteria (when using Nielsen's heuristics) and increases the credibility of the data to development teams. Writing formal reports also makes the data less likely to be forgotten or lost.

While the HE method has many good aspects, the process does come with some caveats and drawbacks. We found that writing up UARs can be time-consuming, especially if each evaluator writes them up individually. UAR writing can take time away from inspecting the interface for problems. Writing single, consolidated UARs after individual problems have been merged can be more efficient than having each evaluator write their own UARs; however, we did notice that overlapping UARs were a great help when they proposed two different design solutions and tradeoffs. When consolidating our UARs, we hardly looked at the individual UARS that we produced. In our HE consolidation meeting, evaluators did not need to look at their UARs to argue on behalf of them. We understand that in a diverse development team a UAR is necessary to justify the suggested problem, but when working with a usability team seems to generate unnecessary paperwork. UAR writing can also drain the stamina of the evaluation team because it is very time consuming.

To address these drawbacks, we propose several modifications to the method that can be employed when appropriate. Analysts could simply make short notes of problems during individual inspection and the group meeting is where problems are discussed, merged, or rejected. Problems can be combined at a consolidation meeting into more complete UARs, which can then be written up. The meeting is an ideal time to resolve questions about which heuristics apply to each problem. Even if evaluators must write their own UARs first, we recommend that they take brief notes at first while walking through the interface. After finding problems in an interface, then they should write these notes into UARs rather than writing formal UARs as each problem is discovered.

The HE process has other caveats aside from the UAR writing process. After we conducted CW and TA, we noticed serious problems that seemed obvious in retrospect but were only caught using CW or TA. This implies that, despite being low cost, HE should not be the only process for uncovering interface issues. HE can also miss problems because it is often performed without any specific task in mind. CW is better in this respect. Evaluators in HE are also unable to detect problems because it is sometimes hard to stick to "the user is not like me" mantra; problems go unnoticed simply because the evaluators are already familiar with the interface and may suffer from design blindness. It is therefore useful if evaluators who were not involved in the design perform HE.

The Nielsen's ten heuristics are often a good fit to many problems. However, at times, we felt like we were stretching their definitions in order to use them to describe problems. We have anecdotally heard that others experienced this with HE. The heuristics are not always adequate at addressing how easy something is to learn as well as human factors issues such as perception, affordances, and appropriateness of controls. These issues can be resolved by either modifying Nielsen's heuristics or creating heuristics which are customized to the interface at hand.

In sum, we find that heuristic evaluation is a valuable, high return usability inspection technique for most user interface development processes. It does have drawbacks, but these can be minimized when the evaluators adapt the process to their needs. Compared to the other methods, we felt that the HE process provided us with data almost as valuable as the think-aloud data. However, due to the fact that our HE UARs were spread out over the entire interface, we could not use the bulk of the HE data in our redesign.

Cognitive Walkthrough (CW)

We found that one of CW's main advantages is that it can be used to evaluate how easy it is to use an interface through exploratory learning. This is especially true when assumptions about the user are scaled down i.e. use minimal assumptions for novice users and more assumptions for expert users. In this sense, this method was useful to gain a better idea of how a novice user would use the Palm Pilot interface.

CW is flexible in that the assumptions can be used to tailor the method to different types of users. One of its most valuable features is that it allows for evaluation of learning through system interaction. Another feature is that analysts are forced to identify assumptions about the user's knowledge.

The process of identifying assumptions forced us to focus on a specific kind of user since we had to derive the assumptions from data from the CI. Although this was at times constraining, the focused nature of the cognitive walkthrough allowed us to understand the user more in depth.

Although assumptions may be wrong, they are grounded in user data. Forming assumptions forces analysts to step out of their own shoes.

While CW models the user more accurately than HE (where the user model is not explicit), it has some drawbacks. Because it is good for evaluating walk-up-and-use interfaces, correcting problems found in the CW can make the interface less efficient for expert users. This could lead to implementation of fixes that help a novice user but slow down an expert user. Also, compared to HE, CW is very narrowly focused and resource intensive for the number of UARs produced.

In general, our CW was very successful. Despite generating fewer UARs, the CW UARs had higher severity ratings and they identified serious usability issues. Thus, we chose to address all of them in our redesign. Also, the scope of the CW was inherently narrower than that of HE because we focused on a specific task in the *Date Book* application. Given that none of the UARs from CW and HE overlapped, we found that performing CW was critical to our redesign.

Think-aloud (TA)

Our group consensus was that the think-aloud protocol for participant testing was the most useful technique because it provided the richest empirical data for our redesign. In the CW and HE methods, we strived to ground our data in what we believed the user would do while keeping the mantra 'user is not like me' in mind. But trying to predict what the user would do in a completely objective manner with CW and HE is often difficult. Thus, observing data where we actually see the participant perform tasks is invaluable. The HE overlooked serious problems found by the TA because the TA was task-oriented while the HE was not.

The video data was more convincing mainly because it is undisputed evidence of a user having problems with a system. In a work setting, we imagine that it would be easier to persuade development teams with TA data as compared to the other HCI methods because it is difficult to argue with video proof. All of us also found the video quite fascinating, and had no qualms at all reviewing it a countless number of times.

We also found the UAR output to time-invested ratio very favorable compared to other methods. We attribute this to two reasons. First, it takes less time to construct a UAR on video evidence than it does on a set of heuristics or CW steps since the problem is inherently more visible—critical incidents are easily identifiable in a TA. In other words, there is less personal and group deliberation about a particular problem. As a result, there is a great deal more overlap among the problems found since all the analysts focused on the same video data. Second, by the end of the semester, we were also all more comfortable writing UARs so the whole process was much faster. Since this method is so valuable in the classroom and work settings, we recommend that the TA protocol analysis be moved to earlier in the semester. This would allow students to become accustomed to writing UARs on a data set that is easier to analyze. In addition, we have seen how valuable this method is in industry and feel that more time should be devoted to it in the semester. The tradeoff is that the CW and HE methods would be biased by the user data seen earlier in the semester.

Despite all this, we did happen to find a set of drawbacks that made this method less useful in the context of this class. First, we thought that it would be more valuable to evaluate more participants performing the same task. In the tapes we analyzed, the user became 'stuck' and, as stated in the homework, the observer should have intervened. We would have benefited from better TA data to work with while learning. Over half of one video was not particularly useful for TA analysis purposes. We would have found it useful to concentrate on more serious issues over a set of users since in a workplace setting we would (hopefully) not base our redesign decisions from a single user study.

Another drawback with our TA is that it was performed on an emulator and not on the actual Palm Pilot. We were unable to analyze some of the finer grained physical input issues that might have occurred with the stylus, Graffiti, and the physical buttons on the palm. For example, on the actual Palm Pilot, the two center buttons that represent up and down afford pressing but we believe that affordance is suppressed when using the emulator. The user data was slightly biased because his input modality was not the same as the physical device.

Furthermore, we feel that our data would have been richer had the video included the user's face in a split screen. At multiple points in the video, we heard the observers in the background and deduced that there was some problem going on. For example, when the user hovered over one button and paused, there was no way to determine if the user was either thinking or focusing his attention elsewhere such as glancing over to the instruction sheet.

As a group, we have confidence in the think-aloud protocol. We realize that asking the user to think-aloud without intervention, and basing critical incidents from what was voiced in the participant's working memory provides a reliable and valid way of discovering usability problems. That being said, we noticed usability problems that were not specifically voiced by the participant, possibly because the participant did not believe that he was encountering an issue or error (such as trying to set the alarm in the time preferences screen). It was also difficult to distinguish between when the user was annoyed by a minor usability problems and when the user voiced grief at a major usability problem. We believe that the TA method would be more valuable if more time in class was spent on explaining the good and bad criteria.

Design Re-labeling (DR)

If our focus for the final redesign was to come up with a radical redesign, design re-labeling might work. However, the class did not emphasize a process-driven approach to design from the our collected data. Also, the CI/CD, which had a design component, could have informed the design much more as compared to other methods. It was difficult to recall the breakdowns of the CD and design ideas. We would have possibly liked to move the two design-oriented methods—DR and CD—closer together so that we could better generate new design ideas.

In class, all the groups presented their redesign ideas generated through re-labeling. It was helpful to know how a single task can be achieved through so many completely different ways. However, it could have been useful if we examined a few group ideas in detail and performed design critiques. We also did not have a document of all the generated redesign ideas, and thus could not incorporate these ideas into our final redesign.