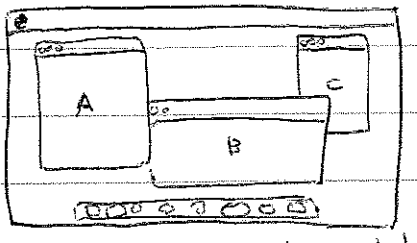
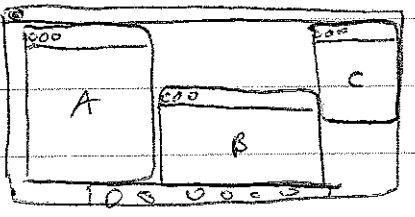
Design Notebook #3

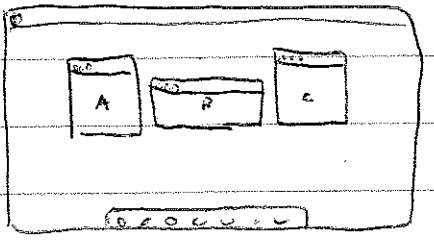
The subject of today's design notebook is the Mac OS X feature, Exposé. Expose is a feature of the OSX operating system that allow users, at a ~~click~~ press of a keystroke or wave of the mouse, to see a visual overview of all currently opened windows. This way, users can sort through a mass of windows and locate a specific one. In the most recent version of OS X, version 10.6, Exposé was given a face lift. So in a desktop instance where you had:



The old Exposé organized the windows like:



Now, in 10.6, Expose would render the following:



4

There are a few usability problems that have arisen from the latest Exposé. First, to list the changes (as I have observed, maybe not officially).

OLD EXPOSÉ	NEW EXPOSÉ
- attempts to position windows close to original positions (retain x, y coordinates)	- windows laid out in a grid. Grid itself is rigid.
- tries to fill up as much of the screen space as possible. So, windows size is zoomed in.	- windows are sized smaller than their original size.
- on hover with mouse, windows are fully highlighted so user knows which window is highlighted.	- on mouse hover, windows have a blue glowing border around them. DARK BLUE ON DARK GREY!

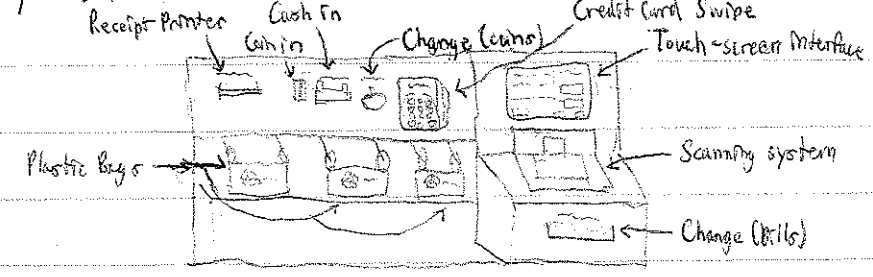
I have found that since the new Exposé has been released, my productivity and utility in using Exposé has dropped dramatically. There is increased time in perceiving which window is actually the one desired. This can be attributed mainly to the decreased sizes of window and increase in distances between windows (Fitts Law).

Since the new Exposé had been contributing to productivity loss, I have switched back to the old Exposé.

10/16/10

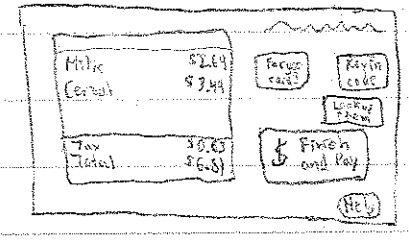
Entry #3

The Albertsons located in the Campus Plaza by Campus Dr. and California has self-checkout systems installed for customer use. I have actually seen these systems popping up elsewhere too, such as the Lowe's located in The District. But since their appearance, I've been surprised by their robustness and user-friendliness.



Very rough sketch of the entire system

Pardon the ugly sketch, but the entire checkout system consists of multiple devices wired together. There is the touch-screen display, a scanning system for barcodes, a machine for swiping credit cards, several change dispensers, a machine to take in bills and coins, and a receipt printer. And of course there are plastic bags for the food items, the only purely physical product in this system. All the customer needs to do is to scan all of the items in their cart through the scanner, bag them, and pay. Every scanned item is displayed on the touch-screen display so that customers can keep track of the total.



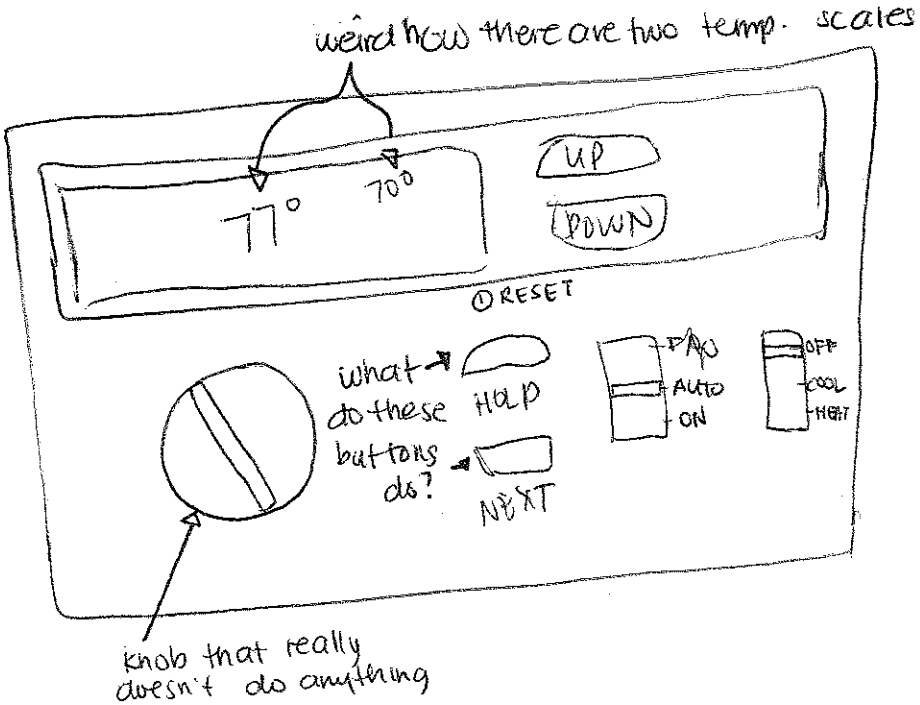
Above is a mock-up of the display while scanning. All the important tasks are easily accessible to the user. If an item has no barcode (ex. produce), you can look up the item on the computer. The only problem with this

for the user is that many produce items have variants that make it hard to tell one apart from other (ex. "large lemons" and regular "lemons"). Since the system trusts the user to pick the correct item, this could potentially lead to erroneous pricing.

The display is simple and easy to follow, which allows most people to intuitively understand how to use the system. Only a few buttons are available to the user, and they are clearly labeled as "Look Up Item", "Finish and Pay" and such. Payments are done by either inserting cash or swiping a credit or debit card, and change is dispense below and on the side of the display. However, bills are dispensed below the scanner and sometimes I find that I almost forget to get my change. But to prevent this, the system beeps until the cash is picked up. And to prevent theft, all items scanned must be moved to the bagged area, where the system calculates the weight of all items to ensure customers do not try to sneak out unpaid items.

The robustness of the system impressed me, as I have never had any major problems with the system crashing or errors. But I think the system might be large enough to scare away some first-time users, so it might be a good idea to consolidate some of the devices together (like the credit card reader with the touch display, for example).

4



I have to admit, although my roommates and I have lived in our apartment for quite a while now, we're still not too sure as to how our air conditioning system works. We have tried fiddling around with the controls and even reading the small instruction manual that comes with it, but all to no avail.

The two temperature gauges being flashed ~~do not refer~~ are not too indicative of what it is they are actually representing. If the temperature needs to be adjusted, we would normally use the up & down

switches, but at times none of the numbers would change. There's apparently a more complex procedure that has to be done involving the hold button, and a few other knobs, but it remained vague even on the instruction manual.

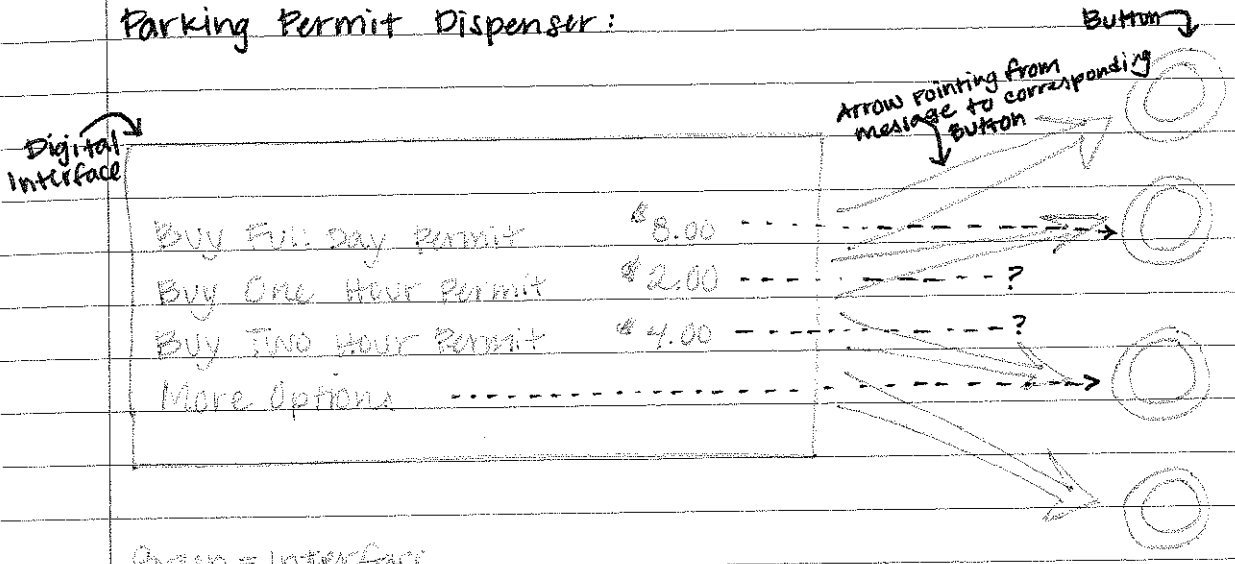
Although we do eventually get it running, the process and the actual interface itself could be a lot more streamlined and definitely a lot simpler. All that is really needed is a way to adjust the temperature if we needed heat, or if we wanted to blast the air conditioning in the apartment.

L/T

Mike Tong

Design Notebooke Week 1

Parking Permit Dispenser:



Green = Interface

Blue = Notes

So above is the interface for the self-serve permit dispenser at UCI. I work for parking so I come in contact with this pretty often - people have A LOT of trouble using it because of the confusing placement of the buttons. If you notice, for example, the button corresponding to the $2 permit is directly across from the $8 permit. Many people who think they are buying a full day & come back angry later w/ parking ticket in hand... the worst part is that the permit is not refundable which is kind of a problem if an error occurs when trying to buy a $20-40 permit.

Overhead View of Issues:

- Buttons are misleading
- Error rate is very high
- Permit is non-refundable
  (Errors are irreversable)
- high cost of failure