• Next week CDS in class
Why the Mac, Win, KDE all stink.

Flow & Csikszentmihalyi
Flow

• Research done by Csikszentmihalyi (pronounced chick-sent-ME-high). Focused on when people are the most optimized. When they are most creative and productive.

• Syn.:
  – In the Flow
  – In the Zone
  – In the Groove (not like Madonna, but then again maybe…)

http://www.ccp.uchicago.edu/faculty/Mihaly_Csikszentmihalyi/html/
http://www.brainchannels.com/thinker/mihaly.html
Quote from Csikszentmihalyi

• “… being completely involved in an activity for its own sake. The ego falls away. Time flies. Every action, movement, and thought follows inevitably from the previous one, like playing jazz. Your whole being is involved, and you're using your skills to the utmost.”
Quote from Csikszentmihalyi

• “How does it feel to be in ‘the flow’?
  – Completely involved, focused, concentrating - with this either due to innate curiosity or as the result of training
  – Sense of **ecstasy** - of being outside everyday reality
  – Great **inner clarity** - knowing what needs to be done and how well it is going
  – Knowing the activity is **doable** - that the skills are adequate, and neither anxious or bored
  – Sense of **serenity** - no worries about self, feeling of growing beyond the boundaries of ego - afterwards feeling of transcending ego in ways not thought possible
  – **Timeliness** - thoroughly focused on present, don't notice time passing
  – **Intrinsic motivation** - whatever produces flow becomes its own reward”
© 1998 Csikszentmihalyi
Quotes from Csikszentmihalyi

• On enjoyment versus pleasure. A hot bath or receiving a massage is pleasurable but not enjoyable. “…that the difference was that pleasure lacked a sense of achievement or active contribution to the result.”

• “…having to learn to control our consciousness. This might come from practicing certain disciplines such as arts, sports or religions. Difficulty is often found in trying to apply these disciplines with others, such as with partner or children or workmates!”

• According to the surveys “15% of people say they have never experienced the flow, 15-20% say they experience it every day (or several times a day), with the rest in between.”

• There is no apparent correlation between intelligence and flow. Instead it is more related to the approach a person takes towards the tasks they wish to achieve, whether its assembly line work or brain surgery. In fact, surgeon that perform the same surgeries repeatedly, e.g. plastic surgery, actually complain of boredom and dissatisfaction.
Most UIs break the flow

- Dialog Boxes
- No memory of what I have done in the past i.e. printer dialog
- Annoying notifications
- Can’t find what I’m looking for on the menu or buttons
- Saving documents
- Popups
- Email notification
- (IM) Instant Messaging
- No undo feature
- Broken links
- Anything that forces me to think about something outside the current task

- UI Design Goal – facilitate users getting into flow and staying there.
Windows UI Stinks

• Where to start …
  – Why the border at the top of the screen between the application title and the top of the screen
  – The auto-hiding taskbar – easily triggered
  – Cascading menus have a linear “zone” not a triangular zone like the Mac
  – Autosave on Office usually doesn’t work IMHO
  – Every new version of the OS they change the look and feel, for no god reason (other than marketing)
X on Linux

• Most of the same problems as window
• How many buttons should a mouse have?
Mac

- One button mouse
- Half size function keys on some keyboards
- Some of the OS X features go backwards in terms of usability and flow
Think

• Why am I doing this?
• What would make my users more empowered?
• What are their goals?
• What gets them into flow? What breaks the flow?
• How much of a learning curve difference can they tolerate?
• What do they already know how to do really well (like “QWERTY”) that if changed would actually be suboptimal?
• Questions
• Comments

• 7.25
Users

Understanding the users is critical to design of user interfaces
Basic User Goals

• Not looking stupid
• Avoiding making big mistakes
• Getting an reasonable amount of work done
• Enjoyment of the experience or at least avoiding boredom and frustration
Timeframes for Achieving the Goal

• Calendar Periods – Hourly, Morning, Afternoon, Daily, Weekly, Monthly, Yearly …
• Project related – by a certain milestone or date
• ASAP
• Faster than manual methods
General Goals Shared By Most

• Accomplish the required tasks
• Learn as little as possible to do those tasks
• Do those tasks in as few steps and as little amount of time as possible
• Do them in the easiest way i.e. avoid doing more work to achieve the same goal.
  – Bad example: the computerized checkbook.
Avoid

• Software that is rude – dialog boxes that point out my mistakes
  – “Error: wrong XYZ”
  – “Are you sure you want to delete XYZ?”
• Making the user look or feel stupid
• Feature centric or data centric - be goal centric
• Speaking in developer’s terms or developer jargon, instead think in terms the user would use.
Management goals

• Reduce cost of an operation
  – Shorten time needed
  – Lower the skills of the operator required
  – Lower training required
  – Lower mistakes and error rates
  – Automate steps

• Increase profits
  – Sell more in a give amount of time
  – Sell new things at a higher price

• Reduce risk
Different view of users

• Call center
  – 1000’s of people – “3% of the US working population is currently employed in call centers, for a total of 1.55 million agent positions in the US call center market.”
  – High turnover – “average annual turnover of 26% for full-time reps, and 33% for part-timers.”
  – Low pay means low skills – “top-paid full-time agents ranged from $6.90 to $40 (median $15”
  – Cost of doing business
  – Customer Dissatisfaction – “The UK’s call center industry is chronically short of skilled staff and therefore more than one in ten calls leave customers feeling ‘irritated, annoyed or furious.’”

Different view of users

- Traders
  - Highly paid
  - Highly Educated
  - Highly Training
  - Turnover is very expensive
  - New technology is adopted quickly and discarded quickly

Different view of users

• Web site
  – Brochureware site
  – Little time to teach the user
  – Frustrate them and they leave
Understand the user pool

• Is it **novices** that will stay **novices**? – web users of a brochure site
• Is it **novices** that only some will become **experts**? – call center
• Is it **experts** that will need to quickly become **experts** on your software? – traders
• Is it **novices** that may become **practitioners** and a few become **experts**? – word processing program
• Is it **novices** that may become **experts** but have no time for training? – eBay.com
User “States”

Non-user
Novice
Practitioner
Expert

Mistreat the user and they can’t get to the next highest state and may return to the base state of non-user!
Models

• **Mental (Conceptual) Model** – how the user “thinks” it works.

• **Manifest** Model – how the software presents to the user that it works

• **Implementation** Model – how it is truly organized internally.

The manifest model should represent the users mental model more than the implementation model in good UI design.
• Questions
• Comments

• 7.75
Application-Level User Interface Design
Application “Flavors” (Types)

• Sovereign
• Transient
• Parasitic
• Daemonic
• Kiosk
Sovereign Applications

• They “own the desktop and user”
• Used for long periods
• Usually the only application on the screen
  – Takes full screen
  – Most space devoted to work area
  – Enough room for small widgets around borders (e.g. toolbars)
• Users will be intermediates or experts
• Ex: word processor, spreadsheet, video editing software, development IDEs
Sovereign Applications
Example: PowerPoint
Transient Applications

• Pop up for use, then leave
• Ex: login, email client, scanning software
• Users rarely develop a thorough familiarity
• Can also occupy full screen, temporarily
• Widgets must be easy to understand (e.g. icon buttons not appropriate)
Transient Applications
Example: Disk Defragmenter

![Disk Defragmenter](image)

<table>
<thead>
<tr>
<th>Volume</th>
<th>Session Status</th>
<th>File System</th>
<th>Capacity</th>
<th>Free Space</th>
<th>% Free Space</th>
</tr>
</thead>
<tbody>
<tr>
<td>(C:)</td>
<td></td>
<td>NTFS</td>
<td>2.00 GB</td>
<td>148 MB</td>
<td>7 %</td>
</tr>
<tr>
<td>(D:)</td>
<td></td>
<td>NTFS</td>
<td>4.00 GB</td>
<td>29 MB</td>
<td>0 %</td>
</tr>
<tr>
<td>Data 2 (E:)</td>
<td></td>
<td>NTFS</td>
<td>3.36 GB</td>
<td>716 MB</td>
<td>20 %</td>
</tr>
</tbody>
</table>

Estimated disk usage before defragmentation:

Estimated disk usage after defragmentation:

**Analyze**  **Defragment**  **Pause**  **Stop**  **View Report**

- **Fragmented files**
- **Contiguous files**
- **Unmovable files**
- **Free space**
Parasitic Applications

• Shares space with other applications
• Ex: clock, winamp, …
• Real estate is precious — no room for anything but essentials
Parasitic Applications
Example: WinAmp
Daemonic Applications

- Performs services for other programs
- No user interaction
- Often has associated configuration application (transient)
- Ex: web server, printer server/spooler, calendar alarm clock, …
Daemonic Applications
Example: Services under windows
Kiosk Applications

• Typically running on a machine dedicated to that application only, in a fixed location (kiosk)
• Some users may be experts, but most will be novices
• Must be obvious and self-explanatory - no possibility of documentation
• Ex:
  – Information kiosk (Tourist Info, Olympics, etc.)
  – Ticket-purchasing machine (MTA, Amtrak, NJ Transit)
  – Bank ATM
  – Web site!
Kiosk Applications
Example: ATM
Kiosk Applications
Example: WebSite
Taskflow

- Understand the tasks (and activities within a task)
  - Ask if the task is worth automating
  - Determine if the existing process is broken
  - Determine dependencies
- Determine the application flavor/type
- Map taskflow and activities to UI elements
  - Windows or screens
  - Dialog boxes
  - Toolbars
  - Etc.
ATM Tasks/Activities

• Tasks (a group of activities executed in a sequence)
  – GetMoney
  – DepositMoney
  – CheckAcctBalance
  – TransferMoney

• Activities (Shared among the Tasks – a step in a task)
  – Choose language
  – Read card (GetUserIDNumber)
  – Read password (GetUserPIN)
  – Choose SubTask
  – Perform SubTask
  – Ask if another operation is desired
ATM Tasks

GetMoney
1. AuthenticateUser (GetUserIDNumber and GetPIN)
2. If not authenticated retry twice then ABORT
3. SelectAcct - Select account to get money from
4. EnterAmount - Enter amount to receive
5. If amount > accountBalance display message and goto 4
6. DebitAcct - Debit account by requested amount
7. DispenseCash – open dispensor and dispense amount then close dispensory
8. PrintReceipt
9. END
10. ABORT

DepositMoney
1. AuthenticateUser (GetUserIDNumber and GetPIN)
2. If not authenticated retry twice then ABORT
3. SelectAcct - Select account to get money from
4. EnterAmount - Enter amount to deposit
5. If amount = 0 display message and goto 4
6. DepositEnvelope – open depository, accept envelope and then close
7. CreditAcct - Credit account by deposited amount
8. PrintReceipt
9. END
10. ABORT
ATM Tasks

CheckAcctBalance
1. AuthenticateUser (GetUserIDNumber and GetPIN)
2. If not authenticated retry twice then ABORT
3. SelectAcct - Select account to get money from
4. DisplayActBalance
5. END
6. ABORT:

TransferMoney
1. AuthenticateUser (GetUserIDNumber and GetPIN)
2. If not authenticated retry twice then ABORT
3. SelectAcct - Select account to get money from
4. EnterAmount - Enter amount to transfer
5. SelectAcct - Select account to put money in
6. If amountToTransfer > accountBalance display message and goto 4
7. DebitAcct - Debit account by requested amount
8. CreditAcct - Debit account by requested amount
9. PrintReceipt
10. END
11. ABORT
Taskflow Dependencies

• Information flow
  – I.e. must know that user wants to transfer before determining which account

• Irreversible physical actions
  – Verify account balance before giving money

• Authorization
  – Must check password before displaying info
Resources

• Screen real estate
• Input mechanisms
  – mouse
  – keys
• User’s attention
• User’s short-term memory
Taskflow Themes

• Closure and Risk
• Freedom and Constraint
• Modality
• Fast Paths
Closure

• The psychological sense of completing a task
  – Note: *not* the actual fact of doing so.
• Interfaces should provide closure at appropriate times
  – E.g. Clicking “OK” in a dialog box
  – Audio or visual cue
  – Counterexample: ATM giving money, then card
• Especially important for periods of high risk
  – After cut but before paste
  – After major change but before save
Freedom vs. Constraint

• Constrained taskflows restrict the user’s choice of tasks and actions.
• Free taskflows do not.
• Taskflow dependencies provide some constraint, but you may want to provide more, or less
Constrained Taskflows

• Frustrating
• Calming — less thought and worry involved
• Less attention and human memory required
• Work may be lost if user must exit before the end
• Examples: kiosks, wizards, installers
• Implementations: nested dialogs, sequential panels or windows
ATMs: Constrained Taskflows

Choose Language

Read Card

Read Password

Choose Activity

Get Withdraw Amount

Ask about Receipt

Deliver Cash
Free Taskflows

• Can be upsetting for novice users
• Liberating for intermediate/advanced users
• If poorly designed, dead ends are possible
  – Ex: Must add customer to DB before filling order
  – Always provide a way to defer work
• Examples: document-centric applications
Taskflow Notes

- Avoid constraining just for programmer convenience (e.g. asking a series of questions with a series of message boxes)
- Human attention is always serial, not parallel, so using a free taskflow still requires you to provide flow guidance by using layout.
- For reasons of authorization or physical impracticability, some tasks can’t be done now; provide a “do later” option.
- Save work in progress.
Modes

• A *mode* is a state of the system that allows only a restricted set of actions

• Non-computer examples
  – VCR program mode
  – Telephone: dial, talk, busy, etc.

• Computer examples
  – vi
  – emacs
  – page preview in Microsoft Word
Advantages of Modes

• Multiplexes resources: screen, mouse, keyboard
  – Ex: same key can have different meanings in different modes

• Helps focus attention (prevent distractions)
  – Ex: irrelevant widgets can be hidden

• Guides users
  – Encourages correct actions
  – Prevents mistakes
Disadvantages of Modes

• Removes control from the user
• Switching modes takes extra work
• Complicates the “floor plan” of the application (functionality architecture)
  – Commands become harder to find
  – Users become disoriented
Mode Types

• Soft modes
  – Can coexist with other modes of the system
  – E.g. emacs buffer modes, separate windows

• Hard modes
  – Take over the system
  – Vanish when exited
  – E.g. vi modes, modal dialog boxes

• There is actually a continuum

• Even a “modeless” system has modes at some level (e.g. pulling down a menu)
Mode Guidelines

• Make sure the user is aware that there are multiple modes
• Make it clear how to move between modes
• Make it obvious what the current mode is
• Avoid dangerous multiplexing
  – Ex: same “Exit” button means “leave mode” in one mode and “exit program” in another
Fast Paths

- Provide a fast execution path for common tasks
- What is an example with ATMs?
- PowerPoint color selectors
- Toolbar buttons (e.g. Print)
- Avoid wasted “splash” screens on kiosks and web pages
- Provide for user-customizable fast paths (e.g. toolbar buttons)
• Questions
• Comments

• 8.25
Exercise

• Redesign an ATM assuming standard computer equipment (screen, keyboard, mouse)

• How would it be different if it were a home computer application instead of a kiosk app?
Chemical ATM Fast Path

Insert your card to begin

Espanol   Francais
Handling Limited Real Estate

• Closed, highly space-efficient widgets
  – Menus, option menus, drop-down listboxes, combo boxes

• Dialog boxes

• Overlapping windows
Dialog Boxes

• Like transient applications
  – They show up briefly
  – They have “adequate” real estate
  – They must be easy to understand

• “A dialog box is another room. Have a good reason to go there.” — Alan Cooper
  – Modal dialog boxes are very easy to build, but that does not mean they are good.
  – E.g. dialog boxes for line spacing, drop shadow
  – Put functions on the window where they are used.
Nested Modal Dialogs

• Natural for programmers because they correspond to nested function calls

• **Bad** for users
  – Annoying to “go down”
  – Cumbersome to “go up”

• Instead use:
  – Fold-out panels (for rare options)
  – Tabbed dialogs (for many, unordered options)
  – Task lists (for multiple tasks)
Modeless Dialogs and Auxiliary Windows

• Not modal — may remain on screen
• Too many windows leads to *window thrashing* — difficulty in navigating a screen full of overlapping windows
• Problem with Macintosh Finder
Application “Floorplan”

- Graph of windows/modes and their relationships
- **Arc** = path user can get from one window to the other
- Keep very simple or hub-like
  - Low diameter
- Avoid changing the look of a “room” (e.g. changing menubar) — users navigate by position
Dealing with Window Thrashing

• Arranging
  – Cascading
  – Tiling
• Tabbing (e.g. tabbed dialogs, “Notebook”)
• Iconifying
• Selection
  – Window menu
  – MS Windows ALT + TAB
  – MS Windows “Taskbar”
Tabbing Example:
Options in PowerPoint
When to Use Other Windows

• Infrequent functions
• Functions with too many options to fit on the screen
  – E.g. printing, search/replace
• Dangerous or powerful functions
  – E.g. database administration
  – Maybe use another application entirely
• Copious non-essential output
  – E.g. error logs, debugging information
The Burdens of Another Window

- Breaks flow
- May obscure important information on main window
- Must be read and understood
- Must be filled in (if dialog box)
Easing the Burden of Other Windows

• Shortcuts for common cases
  – E.g. print button on toolbar
  – Allow user to define new shortcuts

• Memory — pop-up windows should remember:
  – Size & position
  – State of all widgets
  – Tab (if tabbed dialog box)

• Memory should persist across sessions
• Questions
• Comments

• 8.5
Layout
Affordance

• The perception of what a thing does by looking at it
  – Shape
  – Color
  – Depth
  – Symbols
  – Location
  – Relative and absolute size
  – Ex: the door is a pull but you’re supposed to push it. The sign even says “push.” The affordances are deceiving.
Application Modes

- App behavior can be changed by things the user does.
- Can be confusing/frustrating to users. Ex: editing mode versus printing mode in a word processor, copying files using explorer.
- Often based on a single threaded view of the users tasks.
- Can be a very powerful design approach. Ex: paint programs.
- Use modes that are user driven not implementation model driven or don’t create a mode based app!
- If mode based you need to communicate to the user clearly the current mode.
The center of your applications universe ...

• File centric
  – Most office applications still see the world as files to operate on
  – The web
  – If you need to know the format of the file its file centric i.e. a word file, a text file, …

• Document centric
  – Smaller application function gizmos that would focus on a specific part of a document – chart edits, text editors, spreadsheet editors,…

• Task centric
The hardware & OS to choose …

- If building a product to be released target the next release (or greater) of the hardware and software NOT what common today, unless you know they’ll never upgrade.

- Determine the users needs first and then pick the platform.

- Build for each platform separately.
Things to Ponder

• Create an AM / FM Radio that ONLY uses
  – Three controls – volume, tuning and band (AM or FM)
  – two knobs – volume and tuning
  – One knob - ??
• Questions
• Comments

• 8.9