CS 315 – Intro to Human Computer Interaction (HCI)

Elevator Pitch

- A **short**, **simple description** of your business idea that anyone could understand by the time you ride up three floors in a typical elevator.
- Must haves:
  - **Interesting opening:** opening much immediately grab interest of recipient
  - **Passion**: if you are not excited about your idea, no one else will be
  - **Short:** you only have one minute deliver your pitch
  - **Caution**: Do not take too long to get to the pain/problem that you are solving—you could lose the listener's attention

Elevator Pitch

- Kiss of death:
    - Too much detail
    - Geek speak

Elevator Pitch

- Each team will have 1 minute to give the "elevator pitch" for your leading idea.
- Only one member of the team will give the pitch
- If time
  - ▫ Why you're interested
  - ▫ Relevant skills

Prep for a few minutes

- I'll be timing the elevator pitch... Cut off at 1 minute!
  - Team Member Names
  - Problem
  - Idea
  - Your special skills

2 Minute Project Madness

- Names
- Problem
- Idea
- Your special skills

Working in teams

- In real project teams, the work gets done BETWEEN meetings, not during meetings. Meetings are used to review and discuss work products, to reach consensus, and to assign more tasks and deadlines.
- Assign someone to chair the meeting, someone to take notes, and someone to be timekeeper. Switch roles each meeting.
- It is not reasonable to expect that everyone will contribute exactly equally. Instead, assume different people will bring different strengths.

Working in Teams

- **COMMUNICATE!** This means LISTENING as much as TALKING.
  - Let your team members know what's going on, if you're having problems.
  - If you exchange an email with someone on your team, be sure to cc: the other team member(s).
- **Have fun!**
  - Remember - this is a learning experience. As the semester progresses, patience gets short and anxiety gets high. Understand, respect, and support each other.

Golden Rules, Heuristics, First Principles

The 8 golden rules of interface design

1. Strive for consistency
2. Cater to universal usability
3. Offer informative feedback
4. Design dialogs to yield closure
5. Prevent errors
6. Permit easy reversal of actions
7. Support internal locus of control
8. Reduce short term memory

Nielsen's Take: Ten Usability Heuristics

1. Simple and natural dialog
2. Speak the user's language
3. Minimize user memory load
4. Consistency
5. Feedback
6. Clearly-marked exits
7. Shortcuts
8. Good error messages
9. Prevent errors
10. Help and documentation
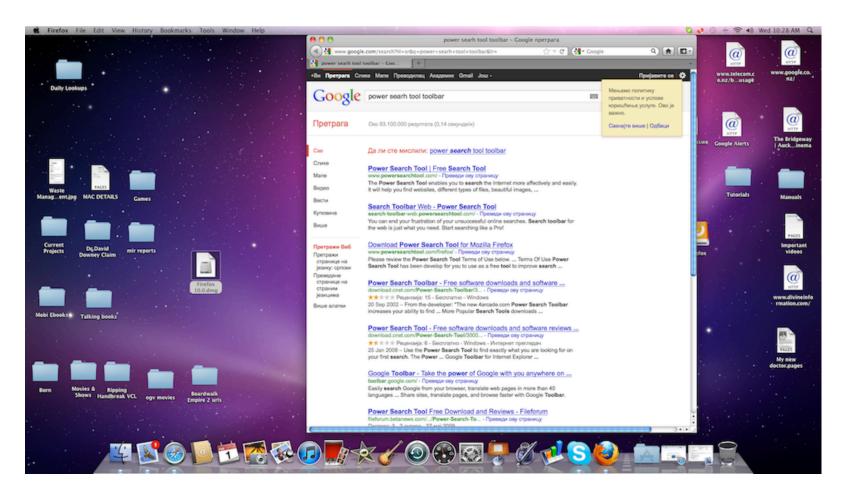
Tognazzini's Take: Sixteen First Principles

1. Anticipation
2. Defaults
3. Human Interface Objects
4. Protect the User's Work
5. Autonomy
6. Efficiency of the User
7. Latency Reduction
8. Readability
9. Color Blindness
10. Explorable Interfaces
11. Learnability — Limit Tradeoffs
12. Track State
13. Consistency
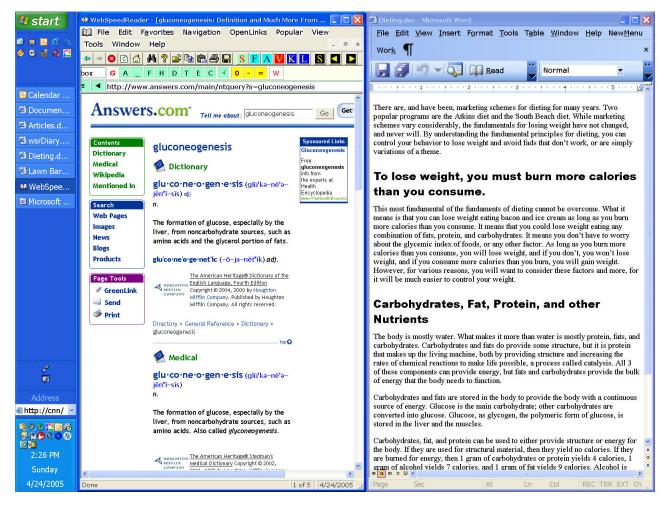14. Fitts's Law
15. Use of Metaphors
16. Visible Navigation

Fitts' Law

- Explain why a Macintosh pull-down menu can be accessed at least five times faster than a typical Windows pull-down menu.
- Suggest at least two reasons why Microsoft made such an apparently stupid decision.

Fitts' Law

Fitts' Law

Discerning Patterns in the Rules

- Despite variations in phrasing and emphasis, certain common themes emerge among these (and other) sets of golden rules — this should give you an idea for prioritization and generality
- One way to reason about these rules objectively is to relate them to how they influence the five metrics of learnability, efficiency, memorability, errors, and subjective satisfaction
- It remains to be seen whether these rules will converge into the "one, true HCI rulebook"

"One Rule to Rule Them All" — Prevent Errors

- If there is any single golden rule that distinctly rises above the rest, it would be Shneiderman #5: Prevent Errors (a.k.a. Nielsen #8 and #9, Tog #4)
- Consistency and feedback also enjoy multiple appearances in these lists, but they don't have the same bang-for-the-buck as error prevention

Prevent errors

- Make error messages specific, positive in tone, and constructive
- Mistakes and slips (Norman, 1983)
- Correct actions
  - Gray out inappropriate actions
  - Selection rather than freestyle typing
  - Automatic completion
- Complete sequences
  - Single abstract commands
  - Macros and subroutines

Integrating Automation vs. Control

- Sanders and McCormick (1993) suggest that we play to a human being's strengths as opposed to a machine's:
- Avoid routine, tedious, and error-prone tasks — automation
- Focus on making decisions, dealing with the unexpected, and planning for the future — control
- A corollary to control is predictability — we generally don't want the computer to "have a mind of its own"

Integrating Automation vs. Control

- The FAA says it well: "improve system performance, without reducing human involvement" and "train users when to question automation" (2003)
- Automation vs. control grows in significance as anthropomorphic and adaptive user interfaces grow in popularity and sophistication
- Microsoft's Office Assistants (or sometimes, Microsoft Office itself)
- Assorted "bots" and pseudo natural-language interfaces (online help, search engines)
- "Trained" spam filters
- Amazon and others' "your store" or "your page" features

Automation and human control

Humans Generally Better

Sense low-level stimuli

Detect stimuli in noisy background

Recognize constant patterns in varying situations

Sense unusual and unexpected events

Remember principles and strategies

Retrieve pertinent details without a priori connection

Draw on experience and adapt decisions to situation

Select alternatives if original approach fails

Reason inductively: generalize from observations

Act in unanticipated emergencies and novel situations

Apply principles to solve varied problems

Make subjective evaluations

Develop new solutions

Concentrate on important tasks when overload occurs

Adapt physical response to changes in situation

Machines Generally Better

Sense stimuli outside human's range

Count or measure physical quantities

Store quantities of coded information accurately

Monitor prespecified events, especially infrequent ones

Make rapid and consistent responses to input signals

Recall quantities of detailed information accurately

Process quantitative data in prespecified ways

Reason deductively: infer from a general principle

Perform repetitive preprogrammed actions reliably

Exert great, highly controlled physical force

Perform several activities simultaneously

Maintain operations under heavy information load

Maintain performance over extended periods of time