CS 160: Lecture 7

Professor John Canny
Spring 2003
Feb 19
Outline

- User testing for Lo-fi prototype
- Discount usability engineering
- Heuristic evaluation overview
- Heuristics
- How to perform a HE
- HE vs. user testing
- How well does HE work
Iterative Design

Design
- task analysis
- contextual inquiry
- scenarios
- sketching

Prototype
- low-fi
- paper, DENIM

Evaluate
- low-fi testing,
- Heuristic eval
Preparing for a User Test

- Objective: narrow or broad?
- Design the tasks
- Decide on whether to use video
- Choose the setting
- Representative users
10 steps

1. Introduce yourself
2. Describe the purpose of the observation (in general terms), and set the participant at ease
   * You're helping us by trying out this product in its early stages.
   * If you have trouble with some of the tasks, it's the product's fault, not yours. Don't feel bad; that's exactly what we're looking for.
10 steps (contd.)

3. Tell the participant that it's okay to quit at any time, e.g.:
   * Although I don't know of any reason for this to happen, if you should become uncomfortable or find this test objectionable in any way, you are free to quit at any time.

4. Talk about the equipment in the room.
   * Explain the purpose of each piece of equipment (hardware, software, video camera, microphones, etc.) and how it is used in the test.
5. Explain how to “think aloud.”
   * Explain why you want participants to think aloud, and demonstrate how to do it. E.g.:
   * We have found that we get a great deal of information from these informal tests if we ask people to think aloud as they work through the exercise. Would you like me to demonstrate?

6. Explain that you cannot provide help.
10 steps (contd.)

7. Describe the tasks and introduce the product.
   * Explain what the participant should do and in what order. Give the participant written instructions for the tasks.
   * Don’t demonstrate what you’re trying to test.

8. Ask if there are any questions before you start; then begin the observation.
10 steps (contd.)

9. Conclude the observation. When the test is over:
   * Explain what you were trying to find.
   * Answer any remaining questions.
   * Discuss any interesting behaviors you would like the participant to explain.

10. Use the results.
    * When you see participants making mistakes, you should attribute the difficulties to faulty product design, not to the participant.
Using the Results

- Update task analysis and rethink design
  - rate severity & ease of fixing CIs
  - fix both severe problems & make the easy fixes

- Will thinking aloud give the right answers?
  - not always
  - if you ask a question, people will always give an answer, even if it is has nothing to do with the facts
  - try to avoid specific questions
Severity Rating

- Used to allocate resources to fix problems
- Estimates of need for more usability efforts
- Combination of
  - frequency
  - impact
  - persistence (one time or repeating)
- Should be calculated after all evals. are in
- Should be done independently by all judges
Severity Ratings (cont.)

0 - don’t agree that this is a usability problem
1 - cosmetic problem
2 - minor usability problem
3 - major usability problem; important to fix
4 - usability catastrophe; imperative to fix
Debriefing

1. Conduct with evaluators, observers, and development team members
2. Discuss general characteristics of UI
3. Suggest potential improvements to address major usability problems
4. Dev. team rates how hard things are to fix
5. Make it a brainstorming session
   * little criticism until end of session
Low-fi assignment

Your evaluation will use the techniques just described: User testing and severity ratings.

Next topic (for later in the course): discount usability methods and heuristic evaluation.
Break
Discount Usability Engineering

Cheap
- no special labs or equipment needed
- the more careful you are, the better it gets

Fast
- on order of 1 day to apply
- standard usability testing may take a week

Easy to use
- can be taught in 2-4 hours
Discount Usability Engineering

Based on:

* Scenarios
* Simplified thinking aloud
* Heuristic Evaluation
Scenarios

- Eliminate parts of the system
- Compromise between horizontal and vertical prototypes
Simplified thinking aloud

- Bring in users
- Give them real tasks on the system
- Ask them to think aloud
Other budget methods

- **Walkthroughs**
  - put yourself in the shoes of a user
  - like a code walkthrough

- **Low-fi prototyping**

- **Action analysis**
  - GOMS (add times to formal action analysis)

- **On-line, remote usability tests**

- **Heuristic evaluation**
Heuristic Evaluation

- Developed by Jakob Nielsen
- Helps find usability problems in a UI design
- Small set (3-5) of evaluators examine UI
  - independently check for compliance with usability principles ("heuristics")
  - different evaluators will find different problems
  - evaluators only communicate afterwards
    - findings are then aggregated
- Can perform on working UI or on sketches
Why Multiple Evaluators?

- Every evaluator doesn’t find every problem
- Good evaluators find both easy & hard ones
Heuristic Evaluation Process

- Evaluators go through UI several times
  - inspect various dialogue elements
  - compare with list of usability principles
  - consider other principles/results that come to mind

- Usability principles
  - Nielsen’s “heuristics”
  - supplementary list of category-specific heuristics
    - competitive analysis & user testing of existing products

- Use violations to redesign/fix problems
Heuristics (original)

- H1-1: Simple & natural dialog
- H1-2: Speak the users’ language
- H1-3: Minimize users’ memory load
- H1-4: Consistency
- H1-5: Feedback
- H1-6: Clearly marked exits
- H1-7: Shortcuts
- H1-8: Precise & constructive error messages
- H1-9: Prevent errors
- H1-10: Help and documentation
Revised Heuristics

Based on factor analysis of 249 usability problems
A prioritized, independent set of heuristics
Revised Heuristics

- H2-1: visibility of system status
- H2-2: Match system and real world
- H2-3: User control and freedom
- H2-4: Consistency and standards
- H2-5: Error prevention
- H2-6: Recognition rather than recall
- H2-7: Flexibility and efficiency of use
- H2-8: Aesthetic and minimalist design
- H2-9: Help users recognize, diagnose and recover from errors
- H2-10: Help and documentation
Heuristics (revised set)

H2-1: Visibility of system status
- keep users informed about what is going on
- example: pay attention to response time
  - 0.1 sec: no special indicators needed, why?
  - 1.0 sec: user tends to lose track of data
  - 10 sec: max. duration if user to stay focused on action
  - for longer delays, use percent-done progress bars

Time Left: 00:00:19 searching database for matches

2/19/2003
Heuristics (cont.)

- **H2-2: Match between system & real world**
  - speak the users’ language
  - follow real world conventions

- **Bad example: Mac desktop**
  - Dragging disk to trash
  - should delete it, **not** eject it
H2-3: User control & freedom
* "exits" for mistaken choices, undo, redo
* don't force down fixed paths
  + like BART ticket machine...

Wizards
* must respond to Q before going to next
* for infrequent tasks
  + (e.g., modem config.)
* not for common tasks
* good for beginners
  + have 2 versions (WinZip)
Heuristics (cont.)

H2-4: Consistency & standards
Heuristics (cont.)

- H2-5: Error prevention
- H2-6: Recognition rather than recall
  * Make objects, actions, options, & directions visible or easily retrievable

MS Web Pub. Wiz.
- Before dialing
  * Asks for id & password
- When connecting
  * Asks again for id & pw
Heuristics (cont.)

- H2-7: Flexibility and efficiency of use
  - accelerators for experts (e.g., gestures, kb shortcuts)
  - allow users to tailor frequent actions (e.g., macros)
H2-8: Aesthetic and minimalist design
* no irrelevant information in dialogues
H2-9: Help users recognize, diagnose, and recover from errors
- error messages in plain language
- precisely indicate the problem
- constructively suggest a solution
Heuristics (cont.)

H2-10: Help and documentation
* easy to search
* focused on the user’s task
* list concrete steps to carry out
* not too large
Phases of Heuristic Evaluation

1) Pre-evaluation training
   * give evaluators needed domain knowledge and information on the scenario

2) Evaluation
   * individuals evaluate and then aggregate results

3) Severity rating
   * determine how severe each problem is (priority)
     + can do this first individually and then as a group

4) Debriefing
   * discuss the outcome with design team
How to Perform Evaluation

- At least two passes for each evaluator
  * first to get feel for flow and scope of system
  * second to focus on specific elements

- If system is walk-up-and-use or evaluators are domain experts, no assistance needed
  * otherwise might supply evaluators with scenarios

- Each evaluator produces list of problems
  * explain why with reference to heuristic or other information
  * be specific and list each problem separately
Examples

- Can’t copy info from one window to another
  * violates “Minimize the users’ memory load” (H1-3)
  * fix: allow copying

- Typography uses mix of upper/lower case formats and fonts
  * violates “Consistency and standards” (H2-4)
  * slows users down
  * probably wouldn’t be found by user testing
  * fix: pick a single format for entire interface
How to Perform Evaluation

Why separate listings for each violation?
* risk of repeating problematic aspect
* may not be possible to fix all problems

Where problems may be found
* single location in UI
* two or more locations that need to be compared
* problem with overall structure of UI
* something that is missing
  + hard w/ paper prototypes so work extra hard on those
  + note: sometimes features are implied by design docs and just haven’t been “implemented” – relax on those
1. [H1-4 Consistency] [Severity 3][Fix 0]

The interface used the string "Save" on the first screen for saving the user's file, but used the string "Write file" on the second screen. Users may be confused by this different terminology for the same function.
HE vs. User Testing

1. HE is much faster
   * 1-2 hours each evaluator vs. days-weeks

2. HE doesn’t require interpreting user’s actions

3. User testing is far more accurate (by def.)
   * takes into account actual users and tasks
   * HE may miss problems & find “false positives”

4. Good to alternate between HE & user testing
   * find different problems
   * don’t waste participants
Results of Using HE

- Discount: benefit-cost ratio of 48
  - [Nielsen94]
  - cost was $10,500 for benefit of $500,000
  - value of each problem ~15K (Nielsen & Landauer)
  - how might we calculate this value?
    + in-house -> productivity; open market -> sales

- Correlation between severity & finding w/ HE
Results of Using HE (cont.)

- Single evaluator achieves poor results
  * only finds 35% of usability problems
  * 5 evaluators find ~ 75% of usability problems
  * why not more evaluators???? 10? 20?
    + adding evaluators costs more
    + many evaluators won’t find many more problems
Decreasing Returns

- problems found
- benefits / cost

Caveat: graphs for a specific example
Summary

- Heuristic evaluation is a discount method
- Have evaluators go through the UI twice
- Ask them to see if it complies with heuristics
  * note where it doesn’t and say why
- Combine the findings from 3 to 5 evaluators
- Have evaluators independently rate severity
- Discuss problems with design team
- Alternate with user testing