Qualitative Evaluation

CS160: User Interfaces
Maneesh Agrawala and Jeffrey Nichols
Review: Decision Making

Hick’s Law – cost of taking a decision:

\[ T = a + b \log_2 (n + 1) \]

- Time depends on number of options
  - Choosing a movie at Blockbuster
**Review: Practice**

Task time on the nth trial follows a power law

\[ T_n = T_1 n^{-a} + c \]

where \( a = .4 \), \( c \) = limiting constant

Applies to skilled behavior

- Sensory
- Motor

**Review: Fitts’ Law**

Hand movement based on series of microcorrections

\[ X_i = \text{remaining distance after ith move} \]

relative movement accuracy remains constant \( \Rightarrow \)

\[ \frac{X_i}{X_{i-1}} = \varepsilon \]

Then

\[ T = I_m \log_2 \left( \frac{2D}{S} \right) \]
Review: Keystroke Level Model

Describe the task using the following operators:

- K: pressing a key or a pressing (or releasing) a button
  \[ t_K = 0.08 \text{ to } 1.2s \] (0.2 good rule of thumb)
- P: pointing
  \[ t_P = 1.1s \] (without button press)
- H: Homing (switching device)
  \[ t_H = 0.4s \]
- D(n,l): Drawing segmented lines
  \[ t_D = 0.9n + 0.16l \]
- M: Mentally prepare
  \[ t_M = 1.35s \]
- R(t): system response time
  \[ t_R = t \]
Converting Temp. Design 1

Convert 92.5

Assume the focus is on the dialog box, so typing on the keyboard will enter text in the text field directly

MKKKMK (3.7s)

HMPKHMKKKKMK (7.15s)

Average: 5.4s

Converting Temp. Design 2

HMPKPK (4.35s)

H + 3(MPKSK) + MPKPK (21.9s)

Average: 13.125s
Converting Temp. Design 3

Simple text interface with the following prompt:

“To convert temperatures, type the numeric temperature, followed by C if it is in degrees Celsius or F it is in degrees Fahrenheit. Then press enter key. The converted temperature will be displayed”
Converting Temp. Design 4

Type in the temperature to be converted. The converted temperature will appear on the right as you type.

MKKKK (2.15s)

Average: 2.15s
Pros and Cons

“To convert temperatures, type the numeric temperature, followed by C if it is in degrees Celsius or F if it is in degrees Fahrenheit. The converted temperature will be displayed.”

What GOMS Can Model

Task must be goal-directed
- Some activities are more goal-directed
  - Creative activities may not be as goal-directed

Task must be a routine cognitive skill
- As opposed to problem solving
- Good for things like machine operators

Serial & parallel tasks (CPM-GOMS)
Advantages of GOMS

- Gives qualitative & quantitative measures
- Model explains the results
- Less work than user study – no users!
- Easy to modify when UI is revised

Research: Need tools to aid modeling process since it can still be tedious

Disadvantages of GOMS

- Not as easy as other evaluation methods
  - Heuristic evaluation, guidelines, etc.
- Takes lots of time, skill, & effort
- Only works for goal-directed tasks
- Assumes tasks expert performance without error
- Does not address several UI issues,
  - readability, memorizability of icons, commands
Today’s Topics

- Discount Usability Engineering
- Heuristic Evaluation
- The Process of Heuristic Evaluation
- Pros and Cons of Heuristic Evaluation

Discount Usability Engineering
Iterative Design

- **Design**
  - Brainstorming
  - Task analysis
  - Contextual inquiry

- **Prototype**
  - Low-fi, paper

- **Evaluate**
  - Low-fi testing,
  - Qualitative eval
  - Quantitative eval

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
Quantitative Testing is Costly

It's very expensive – you need to schedule (and normally pay) many subjects.

It takes many hours of the evaluation team’s time.

A user test can easily cost $10k’s

Examples: Discount Usability Eng.

• Walkthroughs
  – Put yourself in the shoes of a user
  – Like a code walkthrough
Cognitive Walkthrough

Formalized technique for imagining user’s thoughts and actions when using an interface

- Given detailed description of interface
- Select task
- Tell a story motivating user actions required to do task
- Interface should give motivations via prompts/feedback
- Breakdown in motivations imply problem with interface

Walkthroughs are difficult to do when tasks are ill defined and can be accomplished in many ways

Examples: Discount Usability Eng.

- Walkthroughs
  - Put yourself in the shoes of a user
  - Like a code walkthrough
- Action analysis
  - GOMS (add times to formal action analysis)
- **Heuristic evaluation**

- Low-fi testing
- On-line, remote usability tests
  - E.g. Mechanical Turk
Heuristic Evaluation

Usability Heuristics

“Rules of thumb” describing features of usable systems
  – Can be used as design principles
  – Can be used to evaluate a design

Example: Minimize users’ memory load

Pros and cons
  – Easy and inexpensive
    • Performed by experts
    • No users required
    • Catch many design flaws
  – More difficult than it seems
    • Not a simple checklist
    • Cannot assess how well the interface will address user goals
Heuristic Evaluation

Developed by Jakob Nielsen (1994)

Can be performed on working UI or sketches

Small set (3-5) of evaluators (experts) examine UI
- Check compliance with usability heuristics
- Different evaluators will find different problems
- Evaluators only communicate afterwards to aggregate findings
- Use violations to redesign/fix problems

Original Heuristics

H1-1: Simple and 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

Also developed by Nielsen.
– Based on factor analysis of 249 usability problems
– A prioritized, independent set of 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
Heuristic: Visibility (Feedback)

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
  - 1.0 sec: user tends to lose track of data
  - 10 sec: max. duration if user to stay focused on action
- Short delays: Hourglass
- Long delays: Use percent-done progress bars
  - Overestimate usually better

Heuristic: Visibility (Feedback)

Users should always be aware of what is going on

- So that they can make informed decision
- Provide redundant information
Heuristics: Match System & World

H2-2: Match between system & real world
- Speak the users’ language
- Follow real world conventions
- Pay attention to metaphors

Bad example: Mac desktop
- Dragging disk to trash
- Should delete it, not eject it

Heuristics: Match System & World

Speak the users’ language (H1-2)

- Withdrawing money at ATM
- Use meaningful mnemonics, icons and abbreviations
Heuristics: Control & Freedom

H2-3: User control & freedom
- “Exits” for mistaken choices, undo, redo
- Don’t force down fixed paths like the old BART ticket machine...

Wizards
- Must respond to Q before going to next
- Good
  - For infrequent task (e.g. Internet Config)
  - Beginners (2 versions in WinZip)
- Not good
  - For common tasks

Heuristics: Control & Freedom

- Mark exits: Users don’t like to be trapped!

- Strategies
  - Cancel button (or Esc key) for dialog
    - Make the cancel button responsive!
  - Universal undo
Heuristics: Consistency

H2-4: Consistency and standards

Heuristics: Errors and Memory

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

- Promote recognition over recall
  - Recognition is easier than recall

- Describe expected input clearly
  - Don’t allow for incorrect input

- Create orthogonal command systems
  - Using generic commands that can be applied to all interface objects

Preventing Errors

Error types

- Mistakes
  - Conscious decision with unforeseen consequences

- Slips
  - Automatic behaviors kicking in
    - Drive to the store, end-up in the office
    - Press enter one time too many...
  - Mode errors
    - Forget the mode the application is in
  - Loss of activation
    - Forget what your goals were
Forcing Functions

Interlock mechanisms
– Switching from P to D in a car

Lockin mechanisms
– No eject button for floppy disk on Mac

Lockout mechanisms
– Exit stairways

Heuristics: Flexibility

Edit
<table>
<thead>
<tr>
<th>Cut</th>
<th>ctrl-X</th>
</tr>
</thead>
<tbody>
<tr>
<td>Copy</td>
<td>ctrl-C</td>
</tr>
<tr>
<td>Paste</td>
<td>ctrl-V</td>
</tr>
</tbody>
</table>

H2-7: Flexibility and efficiency of use
– Accelerators for experts (e.g., gestures, shortcuts)
– Allow users to tailor frequent actions (e.g., macros)
**Heuristics: Aesthetics**

- **H2-8: Aesthetic and minimalist design**
  - No irrelevant information in dialogues

---

**Hueristic: Aesthetics**

From Cooper’s “The inmates are running the asylum”
Heuristic: Aesthetics
Simple and natural dialog (H1-1)
Present information in natural order

• Occam’s razor
  – Remove or hide irrelevant or rarely needed information
    • They compete with important information on screen
      – Pro: Palm Pilot
      – Against: Dynamic menus
  – Use windows frugally
    • Avoid complex window management

Heuristics: Help Users
H2-9: Help users recognize, diagnose, and recover from errors
  – Error messages in plain language
  – Precisely indicate the problem
  – Constructively suggest a solution
Good Error Messages

H2-10: Help and documentation
– Easy to search
– Focused on the user’s task
– List concrete steps to carry out
– Not too long
Types of Help

• Tutorial and/or getting started manuals
  – Presents the system conceptual model
    • Basis for successful explorations
  – Provides on-line tours and demos
    • Demonstrates basic features

• Reference manuals
  – Designed with experts in mind

• Reminders
  – Short reference cards, keyboard templates, tooltips...

• Wizards
  – Walks user through typical tasks
    • Users feel they are losing control
    • What if I do not have the information requested?

• Tips
  – Migration path to learning new features
  – Can become boring and tedious
Types of Help

• Context sensitive help

The Process of Heuristic Evaluation
Phases of Heuristic Eval. (1-2)

1) Pre-evaluation training
   – Provide the evaluator with domain knowledge if needed

2) Evaluation
   – Individuals evaluate interface then aggregate results
     • Compare interface elements with heuristics
   – Work in 2 passes
     • First pass: get a feel for flow and scope
     • Second pass: focus on specific elements
   – Each evaluator produces list of problems
     • Explain why with reference to heuristic or other information
     • Be specific and list each problem separately

Phases of Heuristic Eval. (3-4)

3) Severity rating
   – Establishes a ranking between problems
     • Cosmetic, minor, major and catastrophic
   – First rate individually, then as a group

4) Debriefing
   – Discuss outcome with design team
   – Suggest potential solutions
   – Assess how hard things are to fix
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
   – Fix: pick a single format for entire interface

   – Probably wouldn’t be found by user testing

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 evaluations are in

Should be done independently by all judges
Levels of Severity

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

Severity Ratings Example

1. [H2-4 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.
Debriefing

- Conduct with evaluators, observers, and development team members
- Discuss general characteristics of UI
- Suggest improvements to address major usability problems
- Development team rates how hard things are to fix
- Make it a brainstorming session
  - Little criticism until end of session

Pros and Cons of Heuristic Evaluation
HE vs. User Testing

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

HE doesn’t require interpreting user’s actions

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

Good to **alternate** between HE & user-based testing
- Find different problems
- Don’t waste participants

Why Multiple Evaluators?

- Every evaluator doesn’t find every problem
- Good evaluators find both easy & hard ones
Number of Evaluators

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

But always depends on market for product:
- popular products ⇒ high support cost for small bugs

Decreasing Returns

![Problems Found](chart1.png)
![Benefits / Cost](chart2.png)
Benefits 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
  • Customer calls to your customer service center

Tends to find more of the high-severity problems

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
• Have evaluators independently rate severity
• Combine the findings from 3 to 5 evaluators
• Discuss problems with design team

• Cheaper alternative to user testing
  – Finds different problems, so good to alternate
Next Time

Quantitative Evaluation

• How to Decide Which Variables To Manipulate and Measure. *Doing Psychology Experiments.* Chap 7. Marin.