Heuristic Evaluation

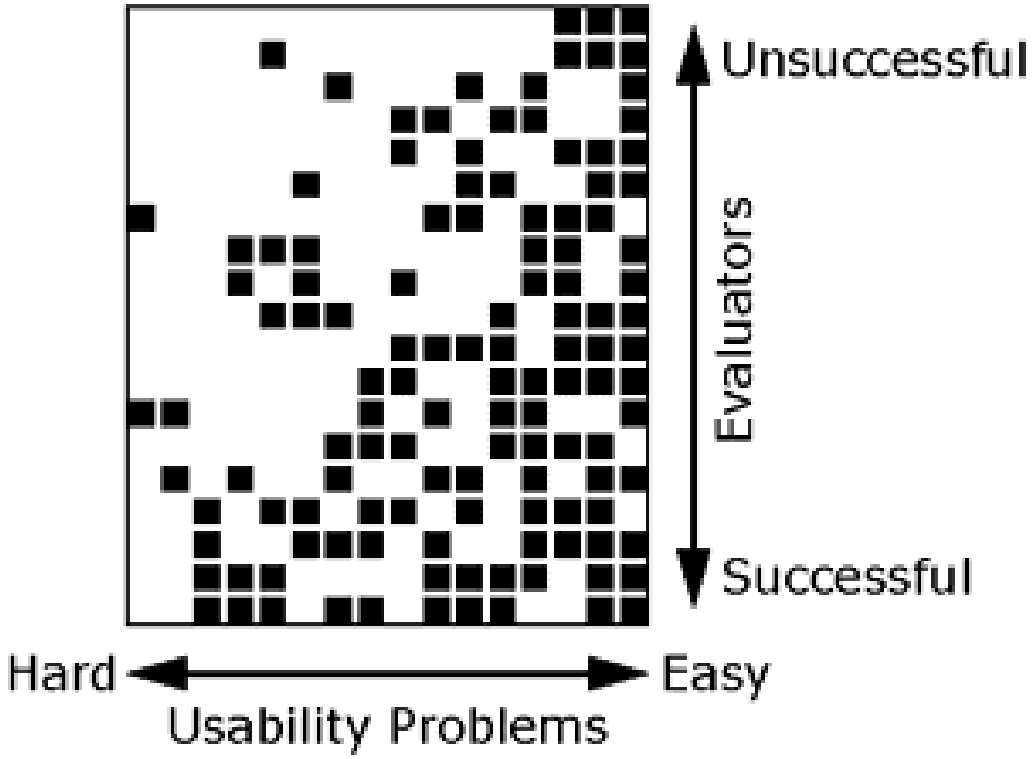
(adapted from Berkeley GUIR & Caitlin Kelleher)

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

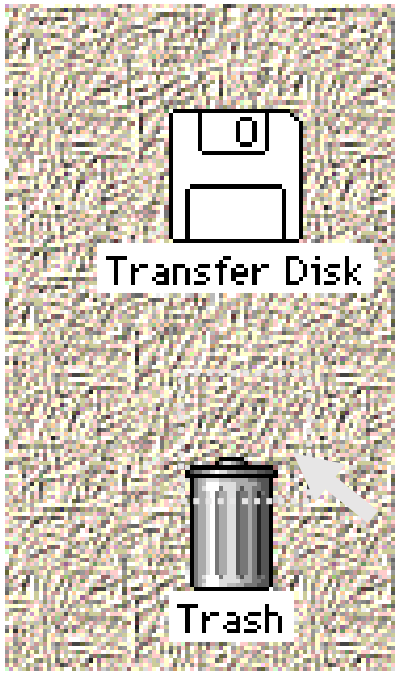
Heuristics (original)

- 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

H2-1: Visibility of system status

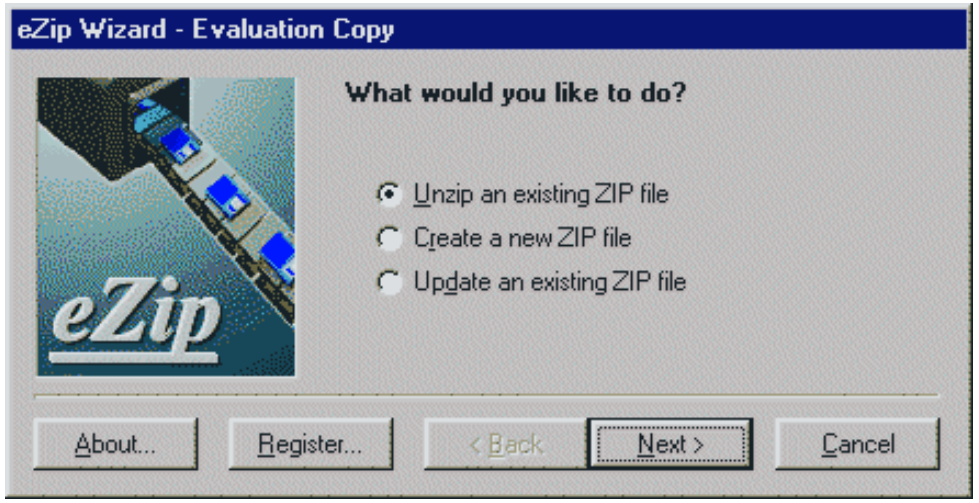
- keep users informed about what is going on
- example: pay attention to response time
  - 0.1 sec: no special indicators needed, why?
  - 1.0 sec: user tends to lose track of data
  - 10 sec: max. duration if user to stay focused on action
  - for longer delays, use percent-done progress bars

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

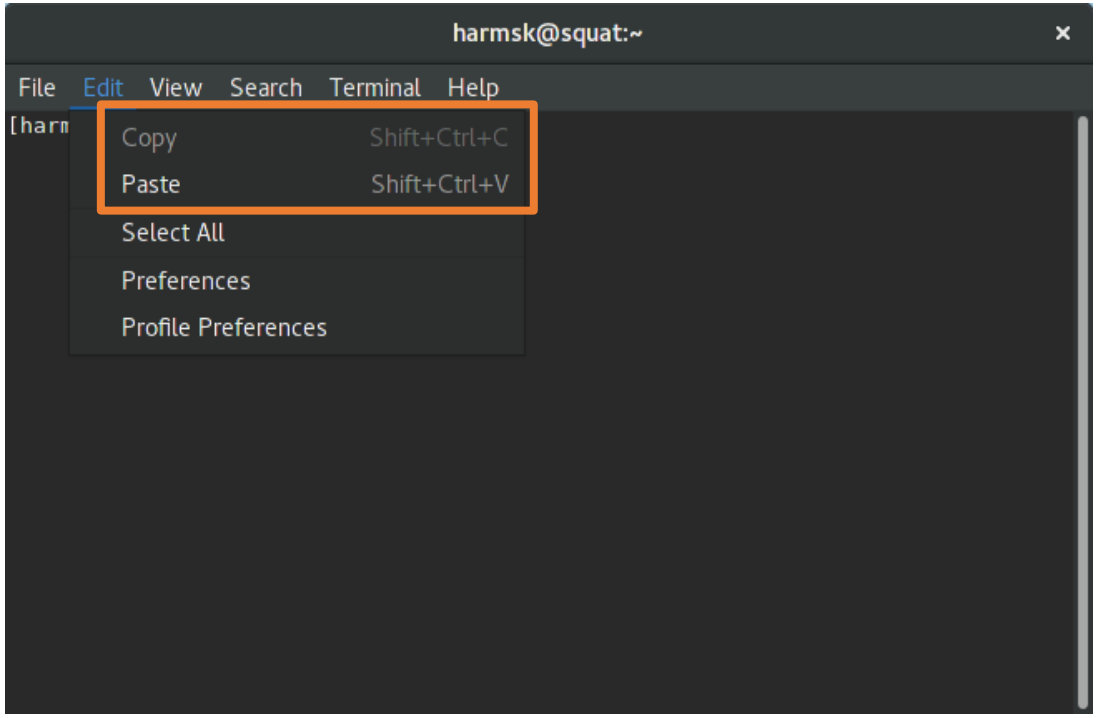
H2-3: User control & freedom



- “exits” for mistaken choices, undo, redo
- don’t force down fixed paths

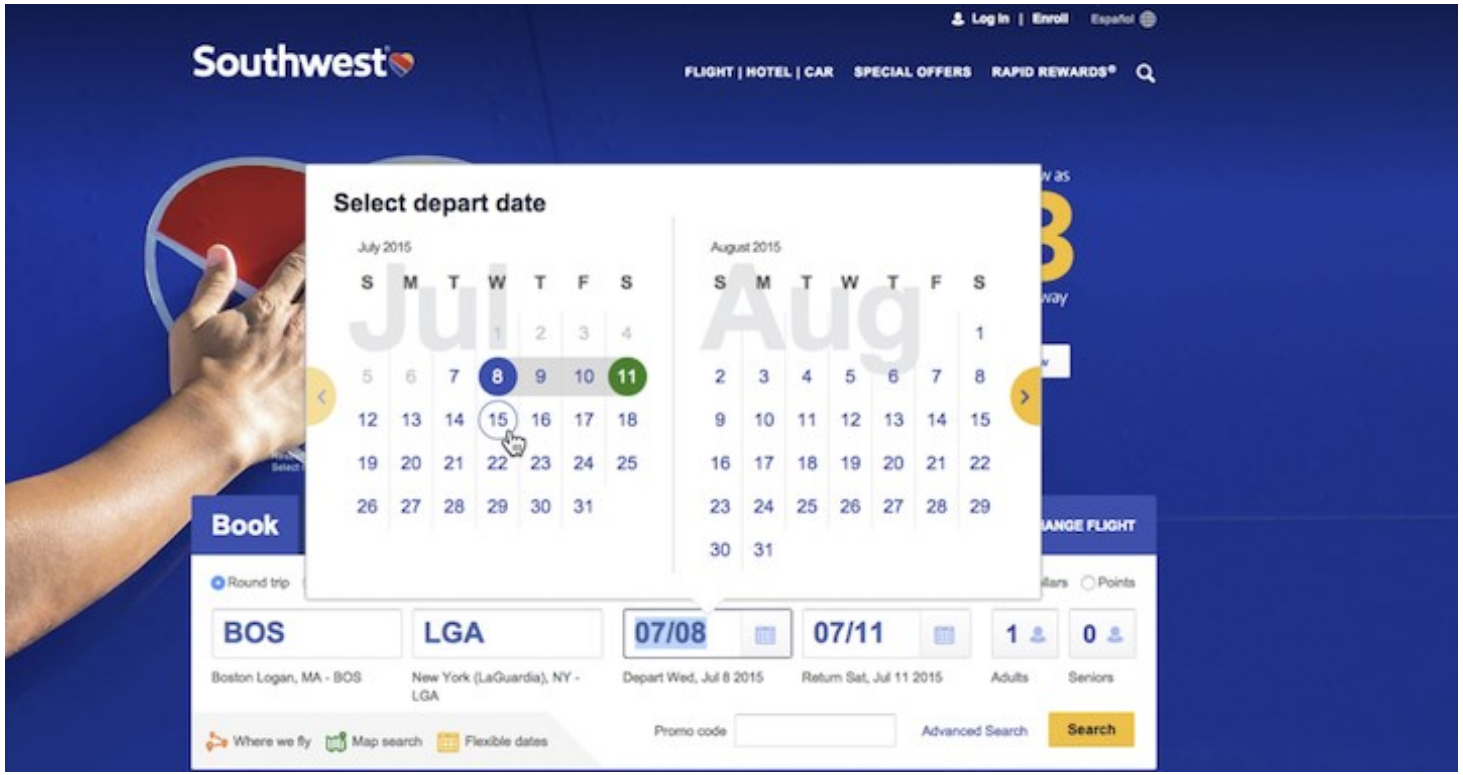
- Wizards
  - must respond to Q before going to next
  - for infrequent tasks
    - (e.g., modem config.)
  - not for common tasks
  - good for beginners
    - have 2 versions (WinZip)

H2-4: Consistency and standards



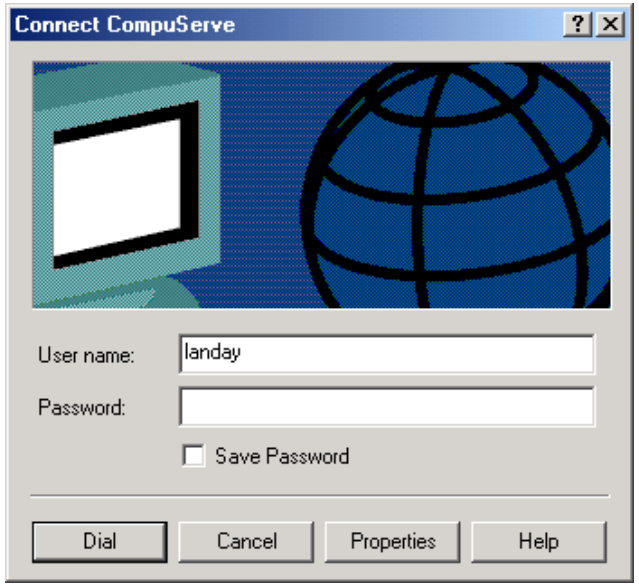
- Do different words, situations, or actions mean the same thing?
  - Login?
  - Sign in?
- Follow Platform Conventions
  - Instagram “share” function doesn’t integrate with system share function

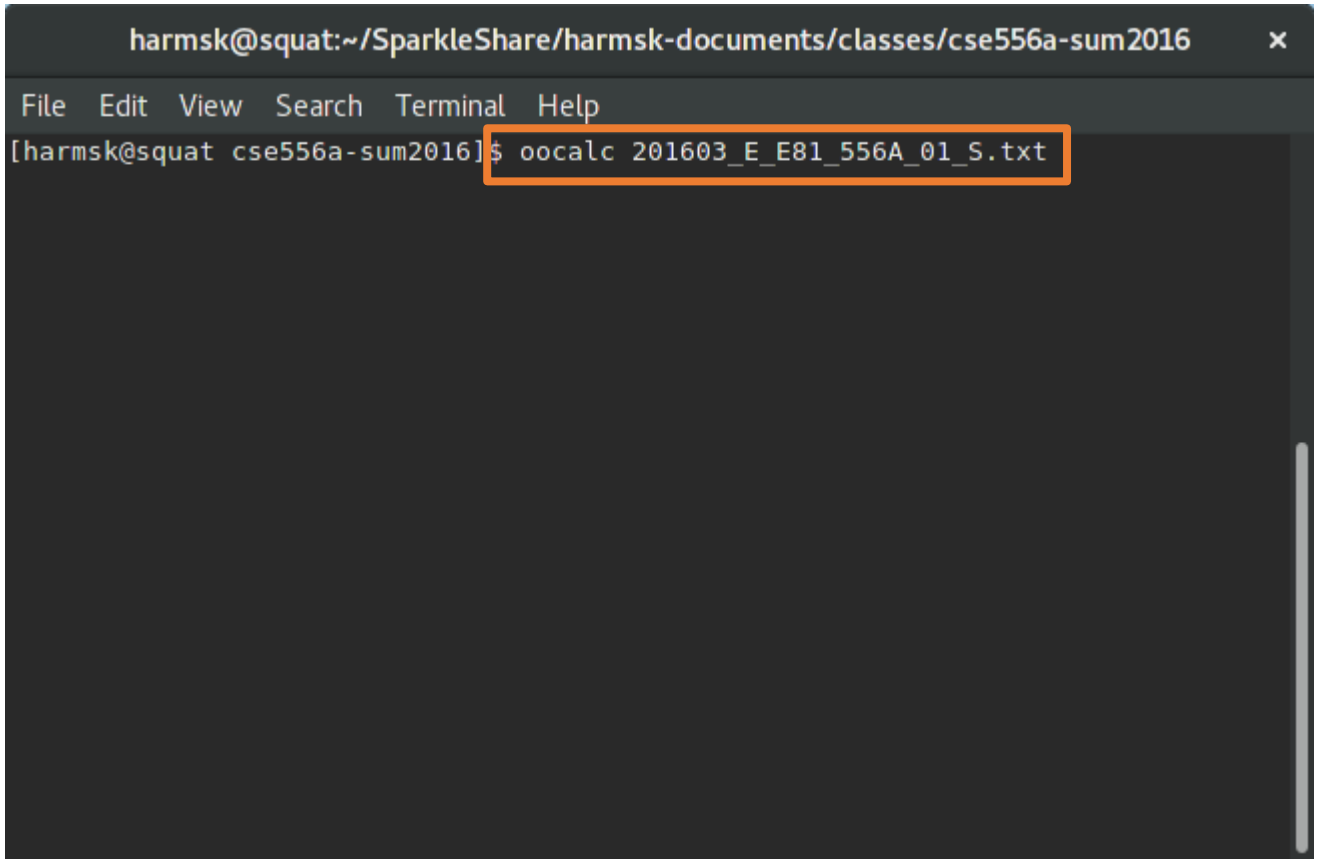
H2-5: Error prevention



- Handle error-prone conditions
  - Eliminate them
  - Check and give users the option
    - Deleting an entire album of photos

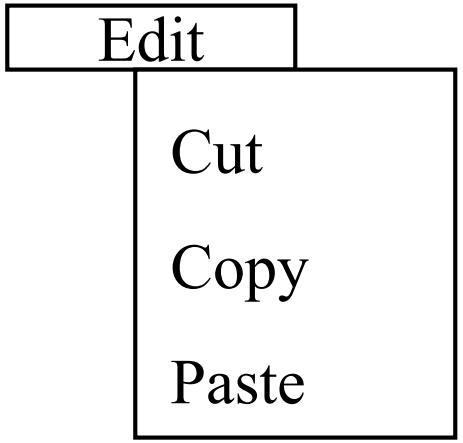
H2-6: Recognition rather than recall





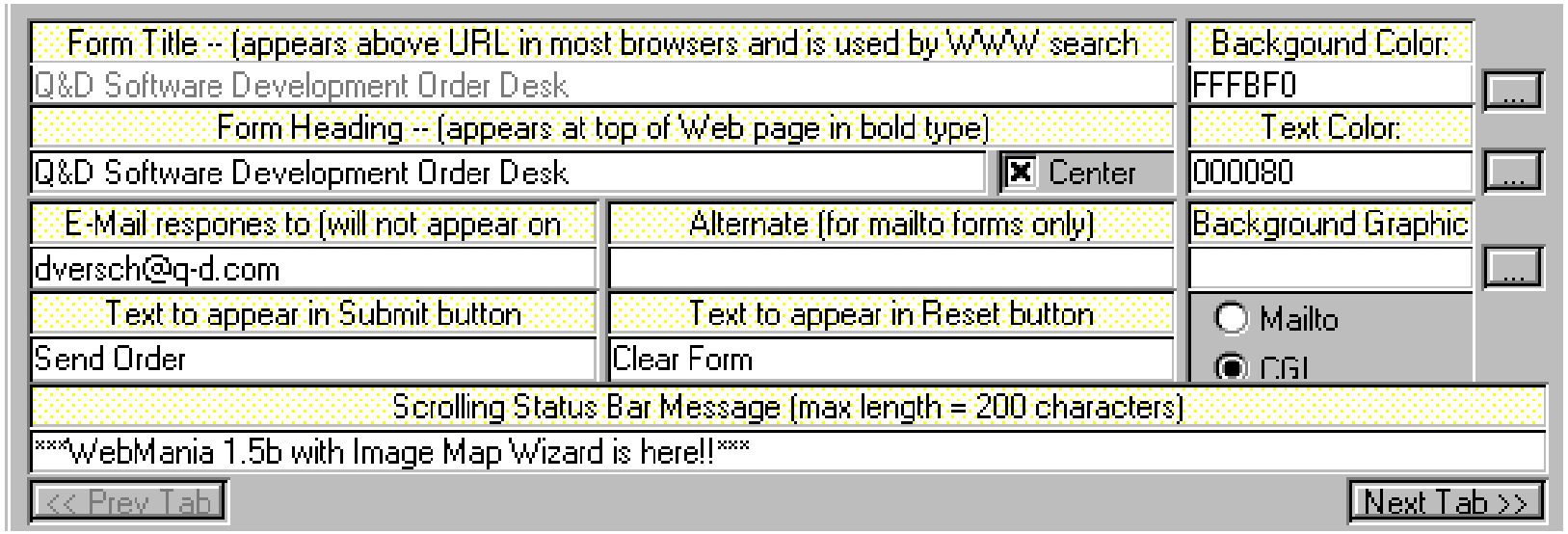
- make objects, actions, options, & directions visible or easily retrievable

H2-7: Flexibility and efficiency of use



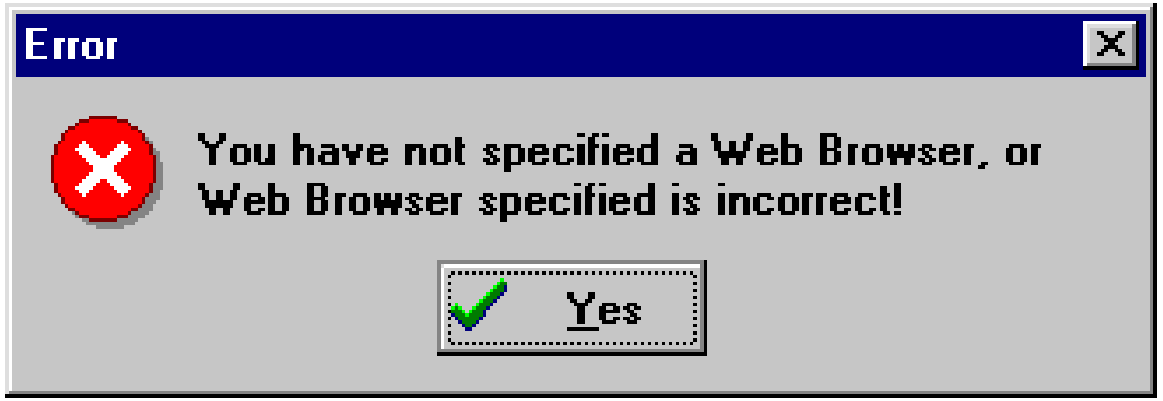
- accelerators for experts (e.g., gestures, kb shortcuts)
- allow users to tailor frequent actions (e.g., macros)

H2-8: Aesthetic and minimalist design



- no irrelevant information in dialogues

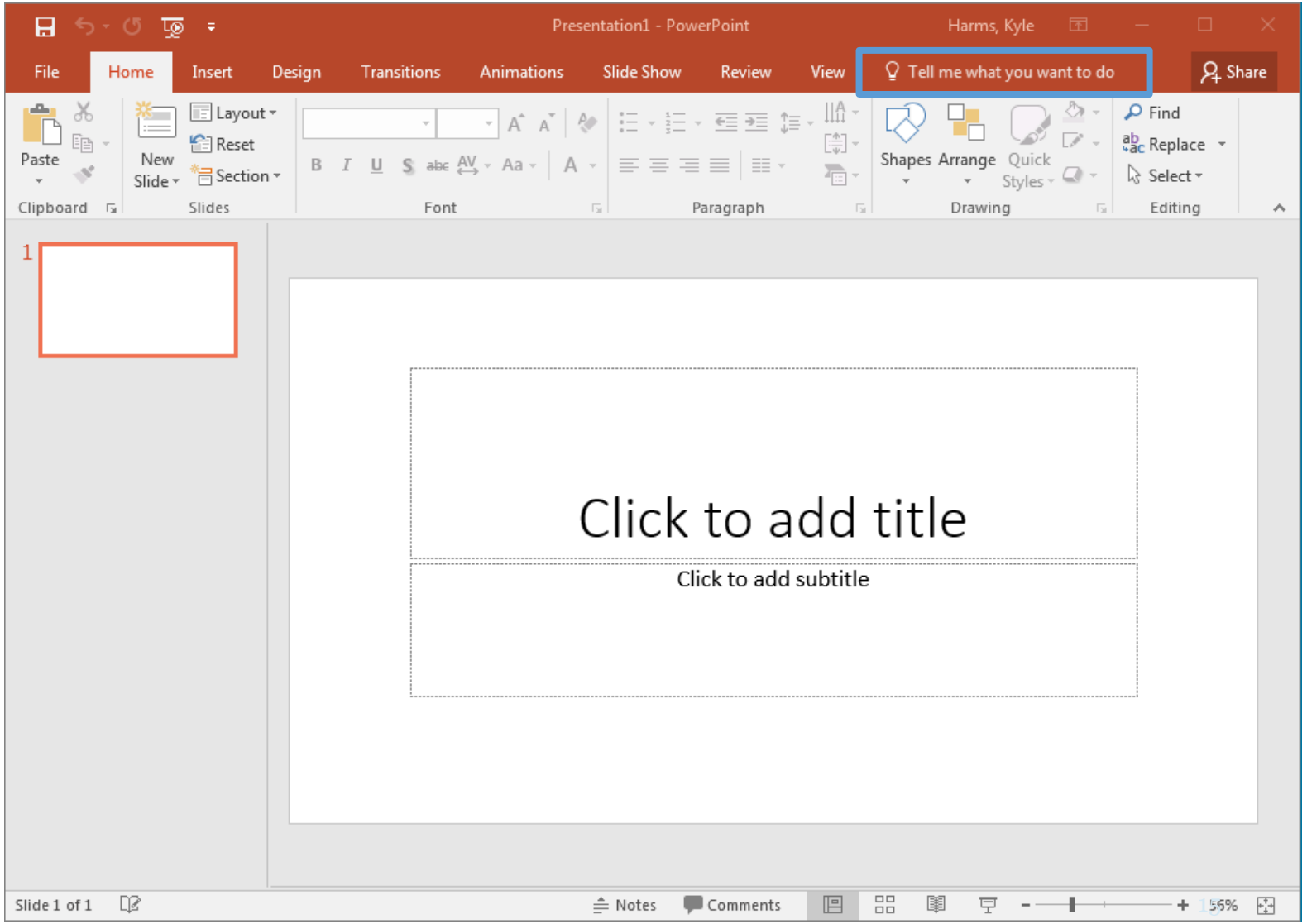
H2-9: Help users recognize, diagnose, and recover from errors



- error messages in plain language
- precisely indicate the problem
- constructively suggest a solution

H2-10: Help and documentation

- easy to search
- focused on the user's task
- list concrete steps to carry out
- not too large



Alternative Heuristics

- Addressing ADA concerns?

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 “Flexibility and efficiency of use” (H2-7)
  - 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

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

- Conduct with evaluators, observers, and development team members
- Discuss general characteristics of UI
- Suggest potential improvements to address major usability problems
- Dev. team rates how hard things are to fix
  - 0 – Trivial; 4 – Reengineer the entire system
- Make it a brainstorming session
  - little criticism until end of session

Severity Ratings Example

1. [H2-4: Consistency and standards] [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.

Heuristic Evaluation vs. User Testing

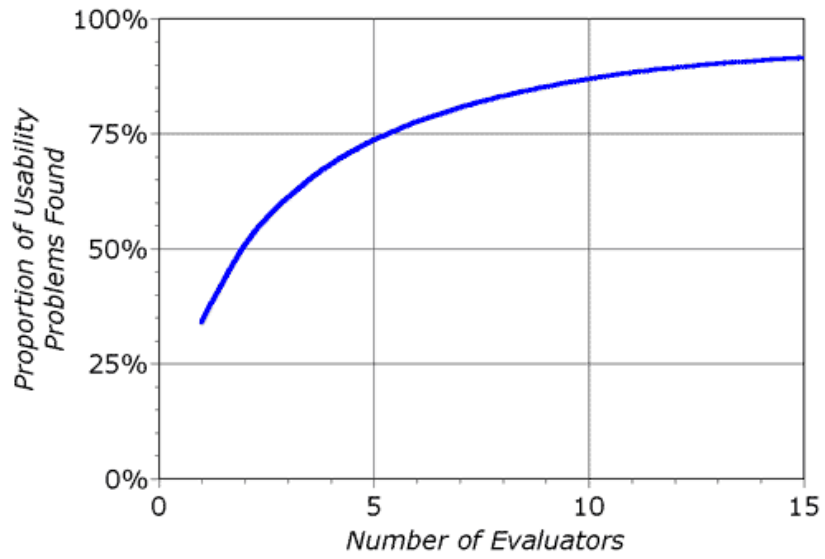
- HE is much faster
  - 1-2 hours each evaluator vs. days-weeks
- HE doesn’t require interpreting user’s actions
- User testing is far more accurate (by def.)
  - takes into account actual users and tasks
  - HE may miss problems & find “false positives”
- Good to alternate between HE & user testing
  - find different problems
  - don’t waste participants

Results of Using Heuristic Evaluation

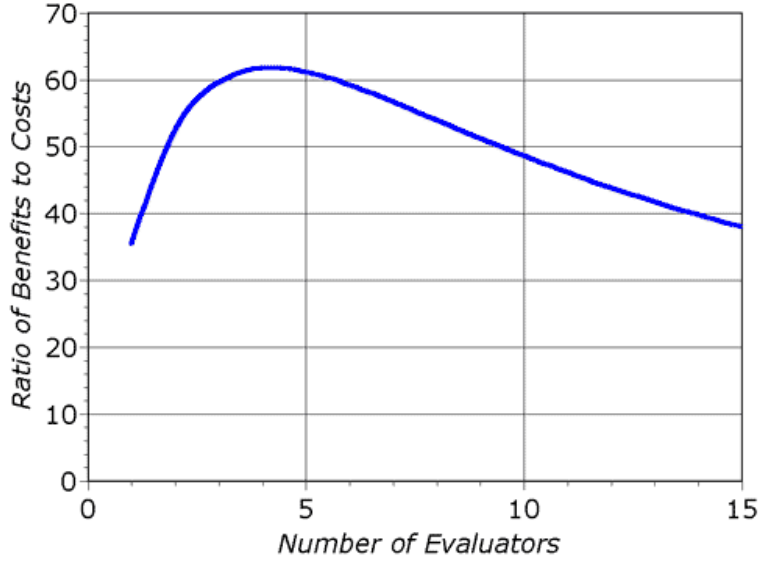
- Discount: benefit-cost ratio of 48 [Nielsen94]
  - cost was $10,500 for benefit of $500,000
  - value of each problem ~15K (Nielsen & Landauer)
  - how might we calculate this value?
    - in-house -> productivity; open market -> sales
- Correlation between severity & finding w/ Heuristic Evaluation
  - 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 & won't find more probs

Decreasing Returns

problems found



benefits / cost



- Caveat: graphs for a specific example

Summary

- User testing
  - important, but takes time (start on mock-ups)
  - use real tasks & representative participants
  - want to know what people are doing & why
    - i.e., collect process data
  - using bottom line data requires more users to get statistically reliable results

Summary (cont.)

- Heuristic evaluation
  - 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
  - alternate with user testing