CSE 440: Introduction to HCI
User Interface Design, Prototyping, and Evaluation

Lecture 12: Inspection-Based Methods
James Fogarty
Daniel Epstein
Brad Jacobson
King Xia

Tuesday/Thursday
10:30 to 11:50
MOR 234
Today

In-Class

   Inspection-Based Methods
   Heuristic Evaluation of Paper Prototypes

Revise Prototypes

Usability Testing Check-In for Friday

   Changes from Inspection
   Changes from First Usability Test
Inspection-Based Methods

We have cut prototyping to its minimum

- Sketches, storyboards, paper prototypes
- Rapid exploration of potential ideas

But we need evaluation to guide improvement

- Evaluation can become relatively slow and expensive
- Study participants can be scarce
- May waste participants on fairly obvious problems
Inspection-Based Methods

Simulate study participants

Instead of actual study participants, use inspection to quickly and cheaply identify likely problems

Inspection methods are rational, not empirical

Today we cover two complementary methods

Heuristic Evaluation
Cognitive Walkthrough
Heuristic Evaluation

Developed by Jakob Nielsen
Helps find usability problems in a design
Small set of evaluators examine interface

three to five evaluators
independently check compliance with principles
different evaluators will find different problems
evaluators only communicate afterwards

Can perform on working interfaces or sketches
Nielsen’s 10 Heuristics

Too few unhelpful, too many overwhelming

“Be Good” versus thousands of detailed rules

Nielsen seeks to create a small set

Collects 249 usability problems
Collects 101 usability heuristics
Rates how well each heuristics explains each problem
Factor analysis to identify key heuristics
Nielsen’s 10 Heuristics

Visibility of system status
Match between system and the real world
User control and freedom
Consistency and standards
Error prevention
Recognition rather than recall
Flexibility and efficiency of use
Aesthetic and minimalist design
Help recognize, diagnose, and recover from errors
Help and documentation

Nielsen, 1994
1. Visibility

Visibility of system status

The system should always keep users informed about what is going on, through appropriate feedback within reasonable time.
1. Visibility

Visibility of system status

The system should always keep users informed about what is going on, through appropriate feedback within reasonable time.

Anytime wondering what state the system is in, or the result of some action, this is a visibility violation.
2. Real World Match

Match between system and the real world

The system should speak the users’ language, with words, phrases and concepts familiar to the user, rather than system-oriented terms. Follow real-world conventions, making information appear in a natural and logical order.
2. Real World Match

Match between system and the real world

The system should *speak the users’ language*, with words, phrases and concepts *familiar to the user*, rather than *system-oriented terms*. Follow real-world conventions, making information appear in a *natural and logical order*.

Refers to word and language choice, mental model, metaphor, mapping, and sequencing
3. User in Control

User control and freedom

Users often choose system functions by mistake and will need a clearly marked “emergency exit” to leave the unwanted state without having to go through an extended dialogue. Support undo and redo.
3. User in Control

User control and freedom

Users often choose system functions by mistake and will need a clearly marked “emergency exit” to leave the unwanted state without having to go through an extended dialogue. Support undo and redo.

Not just for navigation exits, but for getting out of any situation or state.
4. Consistency

Consistency and standards

Users should not have to wonder whether different words, situations, or actions mean the same thing. Follow platform conventions.
4. Consistency

Consistency and standards

Users should not have to wonder whether different words, situations, or actions mean the same thing.
Follow platform conventions.

Internal consistency is consistency throughout the same product. External consistency is consistency with other products in its class.
5. Error Prevention

Error prevention

Even better than good error messages is a careful design which prevents a problem from occurring in the first place. Either eliminate error-prone conditions or check for them and present users with a confirmation option before they commit to the action.
5. Error Prevention

Error prevention

Even better than good error messages is a careful design which prevents a problem from occurring in the first place. Either eliminate error-prone conditions or check for them and present users with a confirmation option before they commit to the action.

Try to commit errors and see how they are handled. Could they have been prevented?
6. Recognition not Recall

Recognition rather than recall

Minimize the user’s memory load by making objects, actions, and options visible. The user should not have to remember information from one part of the dialogue to another. Instructions for use of the system should be visible or easily retrievable whenever appropriate.
6. Recognition not Recall

Recognition rather than recall

Minimize the user’s memory load by making objects, actions, and options visible. The user should not have to remember information from one part of the dialogue to another. Instructions for use of the system should be visible or easily retrievable whenever appropriate.

People should never carry a memory load
6. Recognition not Recall

Addresses visibility of features and information where to find things

Visibility addresses system status and feedback what is going on

Problems with affordances may go here

hidden affordance: remember where to act
false affordance: remember it is a fake
7. Flexibility and Efficiency

Flexibility and efficiency of use

Accelerators -- unseen by the novice user -- may often speed up the interaction for the expert user such that the system can cater to both inexperienced and experienced users. Allow users to tailor frequent actions.
7. Flexibility and Efficiency

Flexibility and efficiency of use

Accelerators -- unseen by the novice user -- may often speed up the interaction for the expert user such that the system can cater to both inexperienced and experienced users. Allow users to tailor frequent actions.

Concerns anywhere users have repetitive actions that must be done manually. Also concerns allowing multiple ways to do things.
8. Aesthetic Design

Aesthetic and minimalist design

Dialogues should not contain information which is irrelevant or rarely needed. Every extra unit of information in a dialogue competes with the relevant units of information and diminishes their relative visibility.
8. Aesthetic Design

Aesthetic and minimalist design

Dialogues should **not contain information which is irrelevant or rarely needed**. Every extra unit of information in a dialogue competes with the relevant units of information and diminishes their relative visibility.

Not just about “ugliness”. About clutter, overload of visual field, visual noise, distracting animations, and so on.
9. Error Recovery

Help users recognize, diagnose, and recover from errors

Error messages should be expressed in plain language (no codes), precisely indicate the problem, and constructively suggest a solution.
9. Error Recovery

Help users recognize, diagnose, and recover from errors

Error messages should be expressed in plain language (no codes), precisely indicate the problem, and constructively suggest a solution.

Error prevention is about preventing errors before they occur. This is about after they occur.
10. Help

Help and documentation

Even though it is better if the system can be used without documentation, it may be necessary to provide help and documentation. Any such information should be easy to search, focused on the user’s task, list concrete steps to be carried out, and not be too large.
10. Help

Help and documentation

Even though it is better if the system can be used without documentation, it may be necessary to provide help and documentation. Any such information should be easy to search, focused on the user’s task, list concrete steps to be carried out, and not be too large.

This does not mean that the user must be able to ask for help on every single item.
Heuristic Evaluation Process

Evaluators go through interface several times
  inspect various dialogue elements
  compare with list of usability principles

Usability principles
  Nielsen’s “heuristics”
  supplementary list of category-specific heuristics
    (competitive analysis or testing existing products)

Use violations to redesign/fix problems
Examples

Can’t copy info from one window to another
  violates “Minimize memory load” (H6)
  fix: allow copying

Typography uses different fonts in 3 dialog boxes
  violates “Consistency and standards” (H4)
    slows users down
    probably wouldn’t be found by usability testing
  fix: pick a single format for entire interface
Heuristics
Heuristics
Heuristics

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

46%
Visibility of system status

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: maximum duration if user to stay focused on action
longer delays absolutely require percent-done progress bars
Heuristics
Heuristics

Mac desktop

Dragging disk to trash should delete, not eject it

Match system to real world

Speak the user’s language
Follow conventions
Heuristics
Heuristics

“Mailto”, “protocol”?

Match system to real world

Speak the user’s language
Heuristics
Heuristics

Flexibility and Efficiency of Use

accelerators for experts (e.g., keyboard shortcuts) allow tailoring of frequent actions (e.g., macros)
Heuristics
Heuristics

Help recognize, diagnose, & recover from errors

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

Adobe Illustrator

You are saving this document in Adobe Illustrator 9.0 format. Saving this document in an older format may disable some editing features when the document is read back in.

Yes    No
Heuristics

User Control and Freedom
Prevent Errors
### Heuristics

**The Radiation Dosimetry Program**

| **Please Enter Desired Dose (in Rems)** | 0.0001 |
| **Enter Substance**                   | Polonium |
| **Isotope Number**                    | 211 |
Heuristics

The Radiation Dosimetry Program

| Please Enter Desired Dose (in Rems) | 0.0001 |
| Enter Substance                   | Polonium |
| Isotope Number                    | 211     |

Prevent Errors
Heuristics

Caution: Changing your RAID configuration will erase all its data in the drive (J:), are you sure?

If you wish to continue, enter the confirmation number "029732" below and click Yes.

Confirmation Number: 029732

Yes  No
Heuristics

Prevent Errors
Heuristics
Heuristics

User control & freedom
- provide “exits” for mistaken choices, undo, redo
- don’t force down fixed paths

Wizards
- must respond to question before going to next
- good for beginners, infrequent tasks
- not for common tasks
- consider having 2 versions (WinZip)
Heuristics
Heuristics

Consistency & Standards
Heuristics

% rm cse440*
%

![Confirm Multiple File Delete](image)
Heuristics

% rm cse440*
%

Error prevention
Recognition rather than recall
Visibility
Heuristics
Heuristics

Aesthetic & Minimalist design

no irrelevant information in dialogues
Heuristics

Select an award style using the scroll bar. When you've found a style that suits you, press OKAY to create that award and open the editor.

Certificate of Achievement

[Detail of certificate's text]

[Options: Cancel, OK]
Heuristics
Heuristics
Phases of Heuristic Evaluation

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

2) Evaluation
   * individuals evaluate interface & make lists of problems

3) Severity rating
   * determine how severe each problem is

4) Aggregation
   * group meets & aggregates problems (w/ ratings)

5) 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
  be specific & list each problem separately
Example Heuristic Violation

1. [H4 Consistency]

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.
How to Perform Heuristic 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 interface
- two or more locations that need to be compared
- problem with overall structure of interface
- something that is missing
  - common problem with paper prototypes
    (sometimes features are implied by design documents and just haven’t been “implemented” – relax on those)
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
Severity Rating

0 - Do not agree this is a problem.

1 - Usability blemish. Mild annoyance or cosmetic problem. Easily avoidable.

2 - Minor usability problem. Annoying, misleading, unclear, confusing. Can be avoided or easily learned. May occur only once.

3 - Major usability problem. Prevents users from completing tasks. Highly confusing or unclear. Difficult to avoid. Likely to occur more than once.

4 - Critical usability problem. Users will not be able to accomplish their goals. Users may quit using system all together.
Example Heuristic Violation

1. [H4 Consistency] [Severity 3]

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.
Why Multiple Evaluators?

Every evaluator doesn’t find every problem

Good evaluators find both easy & hard ones
Fixability Scores

1 - Nearly impossible to fix. Requires massive re-engineering or use of new technology. Solution not known or understood at all.

2 - Difficult to fix. Redesign and re-engineering required. Significant code changes. Solution identifiable but details not fully understood.

3 - Easy to fix. Minimal redesign and straightforward code changes. Solution known and understood.

4 - Trivial to fix. Textual changes and cosmetic changes. Minor code tweaking.
Debriefing

Conduct with evaluators, observers, and development team members
Discuss general characteristics of interface
Suggest potential improvements to address major usability problems
Development team rates how hard to fix
Make it a brainstorming session
Example Heuristic Violation

1. [H4 Consistency] [Severity 3] [Fix 4]

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.

Fix: Change second screen to "Save".
Results of Using HE

Discount: benefit-cost ratio of 48

cost was $10,500 for benefit of $500,000

how might we calculate this value?

in-house $\rightarrow$ productivity; open market $\rightarrow$ sales

Single evaluator achieves poor results

only finds 35% of usability problems

5 evaluators find $\sim$ 75% of usability problems

why not more evaluators?

Nielsen, 1994
Decreasing Returns

Nielsen, 1994
Alternative Inspection-Based Methods

Cognitive Walkthrough

- Helps surface different types of usability problems
- Consider this as a complement to heuristic evaluation

Action Analysis

- Low-level modeling of expert performance
- Be aware of GOMS, but you may never encounter it
Cognitive Walkthrough

Evaluation method based on:

A person works through an interface in an exploratory manner

A person has goals

The person is applying means-ends reasoning to work out how to accomplish these goals

Evaluation by an expert, who goes through a task while simulating this cognitive process
Preparation: Need Four Things

1) User description, including level of experience and any assumptions made by the designer
2) System description (e.g., paper prototype)
3) Task description, specifying the task the expert has to carry out, from a user’s point of view
4) Action sequence describing the system display and the user actions needed to complete the given task. One system display and one user action together are one step.
Cognitive Walkthrough Process

Designer/Developer prepares the required documents described on previous slide

Gives these documents to the usability expert

Expert reads the descriptions, and carries out the task by following the action list

At each step in action list, asks four questions

Record problems similar to heuristic evaluation
Believability

1) Will the user be trying to produce whatever effect the action has?

2) Will the user be able to notice that the correct action is available?

3) Once the user finds the correct action at the interface, will they know that it is the right one for the effect they are trying to produce?

4) After the action is taken, will the user understand the feedback given?
Action Analysis / Cognitive Modeling

GOMS: Goals, Operators, Methods, Selection

Developed by Card, Moran and Newell

Walk through sequence of steps
Assign each an approximate time duration
Sum to estimate overall performance time

1. Select sentence
   Reach for mouse H 0.40
   Point to first word P 1.10
   Click button down K 0.60
   Drag to last word P 1.20
   Release K 0.60
   3.90 secs
Inspection vs. Usability Testing

Inspection is

Is much faster
Does not require interpreting user actions
May miss problems or find false positives

Usability testing is

More accurate, by definition
Account for actual users and tasks

One approach is to alternate between them
Find different problems, conserve participants
CSE 440: Introduction to HCI
User Interface Design, Prototyping, and Evaluation

Lecture 12: Inspection-Based Methods

James Fogarty
Daniel Epstein
Brad Jacobson
King Xia

Tuesday/Thursday
10:30 to 11:50
MOR 234