Interaction Techniques 1

SWE 632
Fall 2015
• HW 5 due today

• HW 6 due next week
Interaction Techniques
Interaction technique

• A method by which a user can perform an action or sequence of actions with a computer.

• Might encompass software (e.g., accelerators on a menu) and/or specialized hardware (momentum scrolling on iOS)
Designing interaction techniques

• Many possible interaction techniques

• Can be organized around computing modality
  • Desktop, mobile, tablet, car, tangible, spoken, …

• Influenced by interaction techniques supported by underlying GUI toolkit
  • AWT, SWT, Cocoa Touch, .NET, Bootstrap, …

• Many important choices to be made in how individual GUI elements are used and how they work together
Designing interaction techniques

• Overall goal is to design for usability, particularly task performance
  • Understand user tasks, design interaction techniques that support these tasks
  • Interaction techniques describe the low level design of how users interact with software to accomplish tasks
Guidelines for interaction techniques

• Often tradeoffs between alternative interaction techniques

• What’s the right choice?

• It depends…
  
  • No absolute right answer

  • Goal is to optimize for task performance, which might lead to different choices in different situations

• Understanding interaction techniques help provide options and understand tradeoffs
Gulfs of execution and evaluation
Norman’s 7 stages of action

1. Goal (form the goal)
2. Plan (the action)
3. Specify (action sequence)
4. Perform (action sequence)
5. Perceive (the state of the world)
6. Interpret (the perception)
7. Compare (outcome w/ goal)
Hartson & Pyla Interaction Cycle

- cognitive affordances (Hartson & Pyla) → signifiers (Norman)
Planning
Planning

• Support users in how they **determine** what to do

• Help users determine what they **can** do
Clear system task model

• Help users accomplish goals by providing clear model of how users should view system in terms of tasks

• Design to match users’ conception of high level task organization

• Help users understand what features exist and how they can be used

• Help users decompose long tasks into small pieces

• Keep task context visible to minimize memory load
Effective planning

• Help users plan most efficient ways to complete tasks

• Keep users aware of task progress, what has been done and what is left to do

• Provide constraints to avoid transaction completion slips
  • e.g., prevent users from starting task and accidentally throwing away work mid-task
Example
Orchestration & interaction flow

• Interaction flow - the next thing the interface wants to do is exactly what user expects
  • Follow users’ mental model
  • Let user direct software
  • Keep all related tools available
• Surprises interrupt interaction flow
• Interfaces should be invisible
Interaction flow guidelines

• Don’t use dialogs to report normal behavior
• Separate commands from configuration
• Don’t ask questions, give users choices
  • Give users default input, show possible options
• Make dangerous choices hard to reach
• Design for the probable, provide for the possible
Progressive disclosure

• a.k.a. details on demand

• Separate information & commands into layers

• Present most frequently used information & commands first
Translation

goals → action sequence
Signifiers

• a.k.a “cognitive affordances” [Hartson & Pyla]

• Goals
  • Show which UI elements can be manipulated
  • Show how they can be manipulated
  • Help users get started
  • Guide data entry
  • Suggest default choices
  • Support error recovery
Hinting

• Indicate which UI elements can be interacted with

• Possible visual indicators
  • Static hinting - distinctive look & feel
  • Dynamic hinting - rollover highlights
  • Response hinting - change visual design with click
  • Cursor hinting - change cursor display
Help users predict outcome of actions

- What does this do?
- Should I click it?
Help users determine where to get started

Double-click to edit

Double-click to edit
Visual design considerations

• Should be visible, noticeable, & legible

• Reduce complexity w/ visual design organization & structure, module & program

• Presented before needed
Clarity of wording (Example)

Function Editor

```
* Number
/
function calculate(command, numbers) {
  if ("*/+-".indexOf(command) == -1) {
    throw 'command not recognized';
  }
  if (!numbers instanceof Array || numbers.length === 0) {
    throw 'numbers not valid';
  }
  switch (command) {
    case '+':
      var res = sum(numbers[0], numbers[1]);
      return res;
    case '*':
      var res = product(numbers[0], numbers[1]);
      return res;
    default:
  }
}
```

1. Line 15: Missing semicolon.
Clarity of wording

• Choose words carefully
• Speak the user’s language
• Avoid vague, ambiguous terms
• Be as specific as possible
• Clearly represent domain concepts
Data entry

• Indicate formatting constraints or directly use constraints

• e.g., dates
Consistency

• In use of **terms**
  
  • e.g., do not use “revise” and “edit” interchangeably

• In how commands **map** to UI interactions
Indicate functional groupings visually
Likely & useful defaults

• Default text, if relevant (e.g., date)

• Default cursor position

• Avoid requirements to retype & re-enter data
Prefer recognition over recall
Error prevention & recovery

- Use constraints (e.g., grey out & disable inappropriate buttons, end choices)
- Offer undo & redo
- Offer constructive help for error recovery
Be careful w/ modes

• Modes create inconsistent mapping
  • E.g., control S sometimes saves, sometimes sends email
  • Especially dangerous for frequent interactions that become highly automatic System 1 actions

• Avoid when possible

• Clearly distinguish if necessary
Metaphors & idioms
Metaphors

• One way to communicate interaction techniques is through metaphors to the real world
Metaphors - advantages

• Leverages understanding of familiar objects & their functions

• File cabinets, desks, telephones

• Provides **intuitive** understanding of possible affordances & eases mapping tasks to actions

• Open a folder, throw file in trash, momentum scrolling
Metaphors - disadvantages

• Tyranny of metaphor: ties interactions closely to workings of physical world

• Adds useless overhead in extra steps, wastes visual bandwidth

• Taken literally, becomes nonsensical
  • e.g., nesting folders 10 levels deep
Alternative - Idioms

• A consistent mental model of how something works
  • e.g., Files: open / close / save / save as

• Offers intuitive understanding of affordances & interactions

• Provides consistent vocabulary for describing interactions

• Only have to learn it once

• Might have originated in real world, but thought of in terms of mental model for UI interactions
Examples of idioms
Examples of idioms

- Email
- Clipboard: cut / copy / paste
- Format painter
- Newsfeed
- Follow item
Designing idioms

Idioms
Application-specific vocabulary

Compounds
Generic input and output actions & symbols

Primitives
Smallest indivisible actions & feedback

inputs

outputs

click
drag
keypress
delete
create
draw
double click
button click
selection

cursors
text

click
drag
keypress

drag

delete
create
draw
double click
button click
selection

scrolling
sorting
dialogues

edit fields
check boxes
highlighting

widgets

Graphics
Navigation
Navigation

• Among windows & screens
• Among panes or frames in a window
• Among tools and menus
• Within an information space
Possible navigation problems

- User can’t find desired location
- User loses track of location
- User can’t remember information from another location
Information foraging

• Mathematical model describing navigation

• Analogy: animals foraging for food
  • Can forage in different patches (locations)
  • Goal is to maximize chances of finding prey while minimizing time spent in hunt

• Information foraging: navigating through an information space (patches) in order to maximize chances of finding prey (information) in minimal time
Information environment

• Information environment represented as **topology**

• Information patches connected by traversable **links**

• Examples
  
  • Web pages, connected by links
  
  • Menu options & dialogs connected by commands
  
  • Locations on map, connected by search, scroll, move interactions with map
Traversing links

• Links - connection between patch offered by the information environment

• Cues - information features associated with outgoing links from patch
  • E.g., text label on a hyperlink

• User must choose which, of all possible links to traverse, has best chance of reaching prey
Scent

• User interprets cues on links by likelihood they will reach prey

  • e.g., do I think that the “Advanced” options are likely to have the option I’m looking for?
Simplified mathematical model

• Users make choices to maximize **possibility** of reaching prey per cost of interaction

• Predators (idealized) choice = \( \text{max} \ [V / C] \)
  
  • \( V \) - value of information gain, \( C \) - cost of interaction

• Don’t usually know ground truth, have to estimate

• Predator’s desired choice = \( \text{max} \ [E[V] / E[C]] \)
Design recommendations for navigation

- Organize information into functionally **related** groups

- Design effective **cues**, describing what will be found by traversing links

- Match **expectations** of user’s mental model

- Provide **search**
In Class Activity
Design a course catalog & registration system

- In groups of 2
  - Design a course catalog & registration system
  - Create sketches showing key screens
  - Should support
    - browsing course catalog, registering for classes, waitlists
    - building plan of courses to take over multiple semesters to fulfill degree requirements