Introduction to Human-Computer Interaction

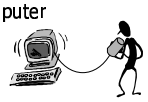
Professor James Landay
May 28, 2004

Outline

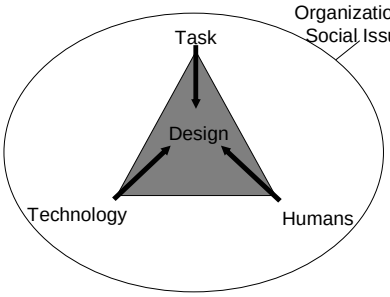
§ HCI Introduction
§ Tips on Designing Good UIs

Human-Computer Interaction (HCI)

§ Human
  – the end-user of a program
  – the others in the organization
§ Computer
  – the machine the program runs on
§ Interaction
  – communication between the user & computer
    • the user tells the computer what they want
    • the computer communicates results



HCI is About Good Design



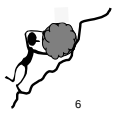
User Interfaces (UIs)

§ Part of application that allows users
  – to interact with computer
  – to carry out their task

HCI = design, prototyping, evaluation, & implementation of UIs

Why Study HCI?

§ Major part of work for "real" programs
  – approximately 50%
§ UW graduates work on "real" software
  – intended for users other than "us"
§ Bad user interfaces cost
  – money (5% ↑ satisfaction → 85% ↑ in profits)
  – lives (Therac-25)
§ User interfaces hard to get right
  – people are unpredictable

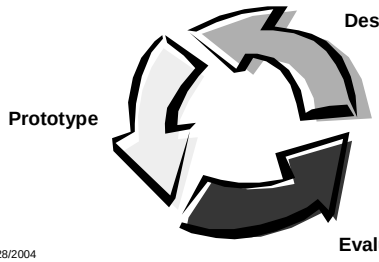


Goal of Capstone HCI Course

§ Learn to design, prototype, & evaluate UIs
- tasks of prospective users
- cognitive/perceptual constraints affecting design
- techniques for evaluating UI designs
- importance of iterative design for usability
- technology used to prototype & implement UIs
- *how to work together as a team*
- *communicating results to a group*

Iterative Design

§ Driven by the variability in human performance



How to Design and Build UIs

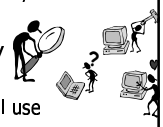
§ User-centered design
§ Rapid prototyping
§ Evaluation
§ UI Programming
§ Iteration

User-centered Design

§ "Know thy User"
- cognitive abilities
  - perception, physical manipulation, & memory
- organizational / job abilities

§ Task Analysis & Contextual Inquiry
- observe existing work practices
- create examples & scenarios of actual use
- try-out new ideas before building software

§ Keep users involved throughout project

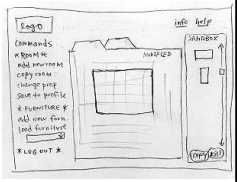


Rapid Prototyping

§ Build a mock-up of design
§ Low fidelity techniques
- paper sketches
- cut, copy, paste

§ Interactive prototyping tools
- Visual Basic, HyperCard, Director, HTML, Denim, etc.

§ UI builders
- Visual Studio, Eclipse, etc.



Mobile *interior* Designer

Sketching & Storyboarding

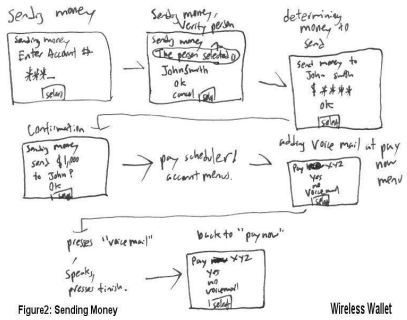
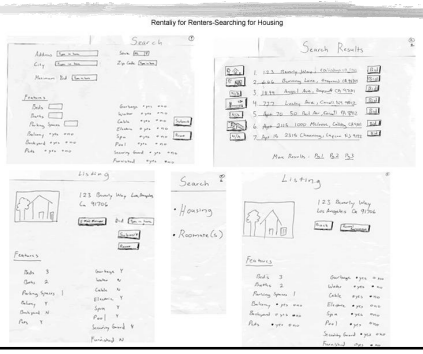
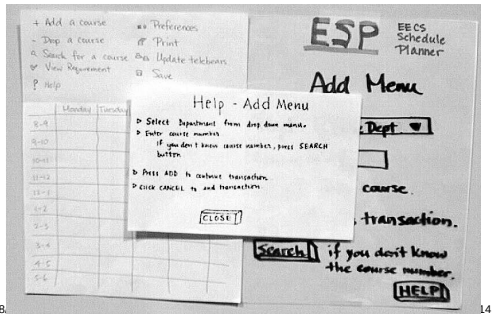


Figure2: Sending Money Wireless Wallet

Low-fi Prototyping & Testing



Low-fi Prototyping & Testing



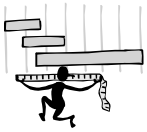
ESP

Low-fi Prototyping & Testing

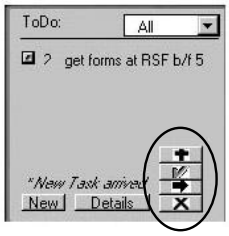
Travelshare

Evaluation

- *Test with real users* (participants)
- Build models
- Low-cost techniques
  - ***expert evaluation (HE)***
  - walkthroughs



Heuristic Evaluation



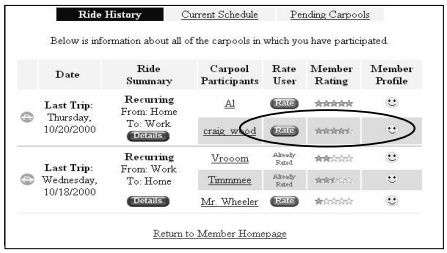
CATTLE

Heuristic Evaluation

Grub

Heuristic Evaluation

User Testing



TravelShare

Programming

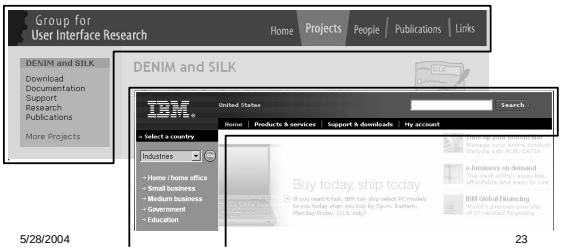
- Toolkits
- UI Builders
- Event models
- Input / Output models
  - Model-View-Controller

Tips on Designing Good UIs

- Follow the iterative, user-centered design process outlined above
  - i.e., involve users early & often – test!
- Good designers steal
  - Design Patterns based on this concept
    - In architecture, software engineering, & UI design
    - See *The Design of Sites* by van Duyne, Landay, & Hong

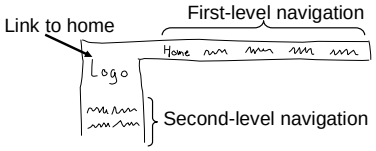
NAVIGATION BAR (K2)

- Problem: Customers need a structured, organized way of finding the most important parts of your Web site



NAVIGATION BAR (K2)

- Solution
  - captures essence on how to solve problem



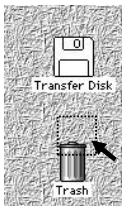
Tips on Designing Good UIs

§ Use Nielsen's Heuristics to guide you

Heuristics

§ 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

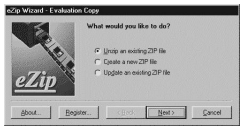
Heuristics (cont.)



§ Bad example: Mac desktop
- Dragging disk to trash
  - should delete it, **not** eject it

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

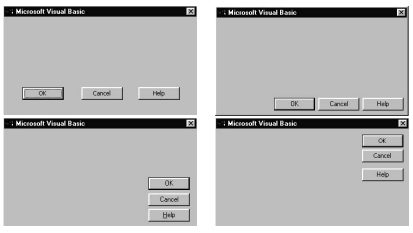
Heuristics (cont.)



§ H2-3: User control & freedom
- "exits" for mistaken choices, undo, redo
- don't force down fixed paths
  - like that BART machine…

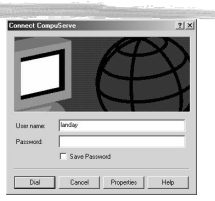
§ Wizards
- must respond to Q before going to next
- for infrequent tasks
  - (e.g., modem config.)
- not for common tasks
- good for beginners
  - have 2 versions (WinZip)

Heuristics (cont.)



§ H2-4: Consistency & standards

Heuristics (cont.)



§ MS Web Pub. Wiz.

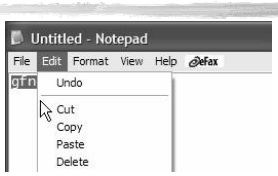
§ Before dialing
- asks for id & password

§ When connecting
- asks again for id & pw

§ H2-5: Error prevention

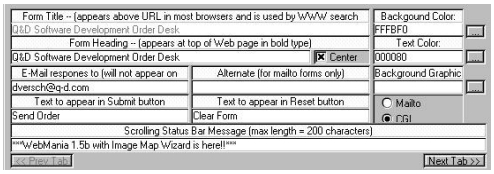
§ H2-6: Recognition rather than recall
- make objects, actions, options, & directions visible or easily retrievable

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)

Heuristics (cont.)



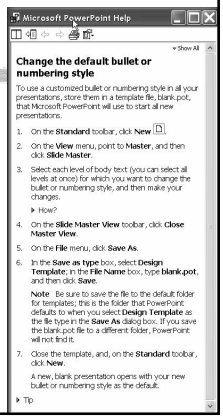
§ H2-8: Aesthetic and minimalist design
- no irrelevant information in dialogues

Heuristics (cont.)

§ 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



How CSE490 JL Fit into the Computer Science Curriculum

§ Most courses teach underlying technology
- compilers, operating systems, databases, etc.

§ CSE490JL concerned w/ design & evaluation
- assume students can program/learn new languages
  - technology is a tool to evaluate designs by prototyping
- build skills that will be important upon graduation
  - design & evaluation
  - creating complex systems
  - working on large teams
  - communicating results

Further Information

§ CS490 JL will be taught this coming Autumn

§ All class material from my previous version of this course archived at
http://guir.berkeley.edu/courses/cs160/2002_spring/