HCI Practices Summary

References:
The art:

- Edward Tufte, “Visual Display of Quantitative Information,” “Envisioning Information,” and especially “Visual Explanations”
- Don Norman, “Design of Everyday Things,” “The Invisible Computer”

The “science”:

- Misha Pavel’s two lectures “Engineering Designs of Interfaces”
- Jakob Nielsen, “Useability Engineering” (1993) Academic Press
- Lewis and Rieman, “Task-Centered User Interface Design,” chapters 3,4,5 (online, shareware, see the course website for URL)

The practice:

- Gomoll and Nicoll, Apple user observation guidelines (on the course website)
- Rettig CACM article on Lo-Fi Prototyping (on the course website)
- Gould et al., 1984 CACM article on the Olympic Messaging System’s design process (on the course website)
- Hix and Hartson, “Developing User Interfaces” Wiley (1993)
- Beyer and Holtzblatt, “Contextual Design”
- Schneiderman, “Designing the User Interface: Strategies for Effective Human-Computer Interaction”

The world – consists of many experts, consultants, and veterans of design projects which pushed the envelope of human-computer interaction. They offer lots of advice. I’ll try to distill out what we will need for the projects in this course.

Stages of any project:

- The Concept – new idea that solves a Problem
- Get to know the context – visit the users, interview, organize…
- Get concrete about who are the users and what are they going to accomplish
    - Break this down into tasks that can be analyzed
    - Each task a complete job, result specified but not means, and performed by a specific class of user
- Check the design without involving “users”
    - Up close – the “cognitive walkthrough”
    - Overall – heuristic evaluation by experienced developers
    - Get more precise only to check points of failure
- Tests with users
    - Do it as early as possible
    - In context or in the laboratory?
- Analyze the results, which can be voluminous
    - The problem is mostly organizing, sorting, and capturing improvements in the production version
    - The solution is no one method, but planning for this, and getting everyone’s involvement is essential

Organization issues – a continuing issue in big companies or small:

The role of the interface “expert,” as an outsider, as a team member, and various ways of making this everyone’s job.

The “peanut butter” philosophy – not!

The Cognitive Walkthru

Equivalent to code or design walkthrough

Step through a sequence of user actions and their results

Ideal for analyzing ease of learnability or spotting gaps and bugs

Requirements:

- Detailed design of prototype interface and system behind it
- Task the user wants to do
- Written list of actions needed to accomplish task on prototype, with results of each action:
    - New system state
    - Feedback to user
- At each step, tell the story:
    - What does the user expect, see, think, learn…

Four questions:

- Is user trying to get the result produced, or something else?
- Is the action visible in the interface or hidden somewhere?
- Is it obvious what the action does?
- Is the feedback visible, appropriate, encouraging…?
- BUGS??

Heuristic Evaluation

Multiple experts review the design, asking good obvious questions. Combine their lists of comments for best coverage.

Not task-oriented, but catches different things.

This advice is distilled from years of watching mistakes happen

Evaluation points – does this:

- Offer simple, natural, in-order dialog
- Speak the user's language
- Require minimum memory load
- Use consistent, standard terms and images
- Offer appropriate feedback
- Have clearly marked exits
- Permit and expose short cuts
- Handle errors (or prevent them)
- Integrate Help and documentation at the right points

-- (Jakob Neilsen)

A good counter-example to following heuristic guidelines is Lewis' story "the worst interface ever," in Appendix M of Lewis and Rieman. It was a multi-mode user interface run amok. Either cognitive walkthrough or heuristic evaluation, whatever you want to call them, would have saved this product much grief, but apparently no one wanted to hear.

Testing with Users

The paper on the Olympic Messaging System describes how they obtain continuous user input throughout the design process. This is pretty “hi fi” prototyping, however, as they built their prototype on a toolkit that would let the final test version be automatically made into the shipped system. For our purposes, we will stay “low fi.”

Marc Rettig’s views on Low Fi prototyping

- Advantages
  - Early, cheap, can be iterated multiple times
  - Demonstrate and enhance key ideas without distractions
  - Get to know your users (or find them)
  - Progressive evaluation as the design evolves, not “summary” after it is too late.
- Disadvantages to Hi Fi prototyping
  - Rigid, hard to incorporate what you learn in product
  - Tendency to defend pet ideas
  - Users focus on superficial characteristics, polish
  - If prototype is buggy, testing stops
  - Can set too high expectations, when the parts not finished as expected to work as well as those that are
  - Comes too late in the process

Lo Fi Prototype Test Procedure

Build the model of your project out of “gan” materials:

Stiff cardboard, foam, colored paper, markers, adhesive tape, acetate sheets… wood is OK, too. Don’t work too hard on appearance, because you don’t want to confuse this with a finished product, just communicate the basic idea and its function.

Don’t think too long, since testing will influence the design more radically than more hours of introspection can.

Build general case pieces, then use a copier to turn these into templates which are marked up for specific states of the prototype.

Select users whose background, level of computer experience, domain familiarity match the intended user population.

Practice first with your friends to get the bugs out of the test procedure.

This takes 3-4 Roles, > 4 people to execute (+ user).

- Greeter and Facilitator
- The Computer (who executes the prototypes actions)
- Observers

Test Process

Remember to keep the pressure down, don't take the feedback personally, avoid giving the impression that this is almost finished, or you will get the wrong reactions, raise excessive expectations.

Ethical and practical concerns:

- Don't underestimate the stress a user may feel
- Don't embarrass them: it's your fault, not theirs, if they fumble
- Stop test at any time if they wish
- Protect their privacy in record-keeping
- Obtain informed consent in writing

Greeter introduces users, gets consent, keeps them waiting for their turn without letting them see what is going on.

Facilitator hands written instructions to the user, who tries to follow them or accomplish what they say. Asks questions to determine what they are thinking. Does not explain how the device works or what should have happened…

Computer manipulates models and pictures of screens to provide the device's expected feedback. He can say what the screen might say or the machine might do. Describes the appearance only, not the internals.

Observers observe – put remarks and possible solutions, ideas on lots of little cards or post-its

Video is useful, but time-consuming to review.

Debrief the user at the end of the test, but don't expect recall to be uniform, as solved problems recede in memory.

Expect each test to take about an hour. We'll see if we can get through the class's projects in two lab sessions of an afternoon each. Project members are facilitator, computer. (if there is only one member, he is computer) Take turns as observers on each other's projects.