THE NEED FOR GOOD USER INTERFACES

• Costs of hardware & software vs. "personware"

• To users, the interface is the system

• Communication vs. computation

• The real issue: Ensuring usability
  * Ease of learning
  * Speed of user task performance
  * User error rate
  * Subjective user satisfaction
  * Retention over time

• Usability "in the large": Ease of use, plus usefulness

• Web usability – Users won't tell you if you don't have it!
OBJECTIVES FOR THIS COURSE

Course is designed to help you develop more usable interaction designs for graphical user interfaces (GUIs) and Web applications by:

• Understanding and applying interaction design guidelines

• Using an iterative, evaluation-centered star life cycle for user interaction development

• Participating in systems analysis, including user, needs, task, and functional analyses

• Doing conceptual and detailed design

• Establishing usability specifications

• Building rapid prototypes

• Performing formative usability evaluation

• Iteratively refining the interaction design

• Knowing how to get started with these new ideas
COURSE MATERIAL

• Evolution of a good GUI or Web design requires:
  * Product — content, human factors of an interaction design
  * Process — techniques and tools for developing an interaction design

• Significant cause of poor usability in product is lack of understanding of proper development process
INTERACTION DESIGN IS NOT USER INTERFACE SOFTWARE DESIGN

• Developing a GUI or Web user interface involves:
  * Interaction component — how a user interface works, its "look and feel" and behavior in response to what a user hears, sees, and does
  * Interface software component — code that instantiates the interaction component

<table>
<thead>
<tr>
<th>User interface development</th>
<th>Development of user interaction</th>
<th>Development of user interface software</th>
</tr>
</thead>
</table>

• Premise: Describing interaction from user’s view should result in more usable design than describing it from software or programmer view.

  What is easiest for the programmer is rarely best for the user...

• "One head, two hats" – emphasizes different roles