USABILITY
AND
USER-INTERFACE DESIGN

What is Usability?

- ISO 9241 defines usability as "the extent to which a product can be used by specified users to achieve specified goals with effectiveness, efficiency and satisfaction in a specified context of use."

  International Organization for Standards (1998) ISO 9241 – 11: Ergonomic requirements for office work with visual display terminals (VDTs) – Part II: Guidance on usability.

User-Interface Design

- Perspectives:
  - Cognitive
  - Social Psychological
  - Organizational
  - Psychophysiological

Cognitive Perspective

- HCI viewed as information exchange and processing
- Models focus on individual reasoning, problem solving, thinking
- GOMS model
  - Goals - what's the user want to do?
  - Operators - what mental/physical options are available (e.g. choices, keystrokes, mouse clicks)?
  - Methods - what sequence(s) of operators are available?
  - Selection rules - how are choices made between different methods?

Social Psychological Perspective

- HCI as a social process:
  - Video-conferencing
  - Collaborative tasks
- 'Humanizing' computers
  - Creating more 'natural' interfaces (voice, 'HAL').

Organizational Perspective

- Interface design to optimize information workflow
  - Network structures
  - Teamwork

  - Virtual workgroups
  - Remote teleoperations
  - Remote habitats (e.g. Mars simulation)

Psychophysiological Perspective

- Interface design based upon human physical capabilities and potential adverse health effects
  - Input device design
  - Screen design
  - Health problems (computer vision syndrome, carpal tunnel syndrome etc.)

Evaluation Methods

- Expert panel analysis (e.g. human factors professionals)
- Predictive analysis (e.g. Information theory)
- Audits, guidelines and standards (e.g. Microsoft Interface Design Guidelines)
- Objective Metrics (e.g. screen character size, button size, screen colors etc.)
- Dialogue error analysis - errors predicted as a consequence of screen layout and dialogue structure.

Prototyping

- Horizontal prototyping tests different features at reduced functionality.
- Vertical prototyping tests limited features at realistic depths.
- Scenarios are minimalist prototypes that simulate the interface for limited features and functionality

Scenarios

- Scenarios describe:
  - the behaviors of a specific user/user group….
  - Using a specific set of features….
  - On a specific computer…
  - To achieve a specific outcome……
  - Under specified circumstances…..
  - Over a specified time interval.
- Scenarios are used:
  - During interface design to express and understand how users will interact with the eventual system.
  - During early stages of interface design to gather useful design feedback with constructing an elaborate prototype.

Usability Testing

- What does the client really want?
- What does the user really need?

- What does the software really need to do?

Usability Issues

- How easy is it to use the software ?
- Does the software have all the required features?
- What does the user learn from using the software?

Usability Measures

- User performance (speed, errors)?
- User preferences?
- User behavior:
  - What do they expect to happen?
  - What features do they use?
  - What features don't they use?
  - Where do they have problems?
  - Where do they go, where don't they go?
  - What new features are needed, what features are redundant?

Consistency

("Principle of least astonishment")

- Certain aspects of an interface should behave in consistent ways at all times for all screens
- Terminology should be consistent between screens
- Icons should be consistent between screens
- Colors should be consistent between screens of similar function

Screen Color Combinations

Simplicity

- Break complex tasks into simpler tasks
- Break long sequences into separate steps
- Keep tasks easy by using icons, words etc.
- Use icons/objects that are familiar to the user

Human Memory Limitations

- Organize information into a small number of "chunks"
- Try to create short linear sequences of tasks
- Don't flash important information onto the screen for brief time periods
- Organize data fields to match user expectations, or to organize user input (e.g. autoformatting phone numbers)

Human Memory Limitations

- Provide cues/navigation aids for the user to know where they are in the software or at what stage they are in an operation
- Provide reminders, or warnings as appropriate
- Provide ongoing feedback on what is and/or just has happened
- Let users recognize rather than recall information
- Minimize working memory loads by limiting the length of sequences and quantity of information - avoid icon mania!

Cognitive Directness

- Minimize mental transformations of information (e.g. using 'control+shift+esc+8' to indent a paragraph)
- Use meaningful icons/letters
- Use appropriate visual cues, such as direction arrows
- Use 'real-world' metaphors whenever possible (e.g. desktop metaphor, folder metaphor, trash can metaphor etc.)

Feedback

- Provide informative feedback at the appropriate points
- Provide appropriate articulatory feedback - feedback that confirms the physical operation you just did (e.g. typed 'help' and 'help' appear on the screen). This includes all forms of feedback, such as auditory feedback (e.g. system beeps, mouse click, key clicks etc.)

Feedback (con.)

- Provide appropriate semantic feedback - feedback that confirms the intention of an action (e.g. highlighting an item being chosen from a list)
- Provide appropriate status indicators to show the user the progress with a lengthy operation (e.g. the copy bar when copying files, an hour glass icon when a process is being executed etc.)

System messages

- Provide user-centered wording in messages (e.g. "there was a problem in copying the file to your disk" rather than "execution error 159")
- Avoid ambiguous messages (e.g. hit 'any' key to continue - there is no 'any' key and there's no need to hit a key, reword to say 'press the return key to continue')
- Avoid emotive messages (e.g. "illegal command" versus "unrecognized command")

System messages

- Avoid using threatening or alarming messages (e.g. fatal error, run aborted, kill job, catastrophic error)
- Use specific, constructive words in error messages (e.g. avoid general messages such as 'invalid entry' and use specifics such as 'please enter your name')
- Make the system 'take the blame' for errors

Anthropomorphization

- Don't anthropomorphize (i.e. don't attribute human characteristics to objects)
- Avoid the "Have a nice day" messages from your computer

Modality

- Use modes cautiously - a mode is an interface state where what the user does has different actions than in other states (e.g. changing the shape of the cursor can indicate whether the user is in an editing mode or a browsing mode)
- Minimize preemptive modes, especially irreversible preemptive modes - a preemptive mode is one where the user must complete one task before proceeding to the next. In a preemptive mode other software functions are inaccessible (e.g. file save dialog boxes)
- Make user actions easily reversible - use 'undo' commands, but use these sparingly
- Allow escape routes from operations

Attention

- Use attention grabbing techniques cautiously (e.g. avoid overusing 'blinks' on web pages, flashing messages, 'you have mail', bold colors etc.)
- Don't use more than 4 different font sizes per screen
- Use serif or sans serif fonts and upper case/lower case appropriately as the visual task situation demands:
  - HOW EASY IS IT TO READ THIS?
  - How easy is it to read this?
  - How easy is it to read this?
  - How easy is it to read this?
- Don't use all uppercase letters - use and uppercase/lowercase mix
- Don't overuse audio or video

Attention (con.)

- Use colors appropriately and make use of expectations (e.g. don't have an OK button colored red! use green for OK, yellow for 'caution, and red for 'danger' or 'stop')

- Don't use more than 4 different colors on a screen
- Don't use blue for text (hard to read), blue is a good background color
- Don't put red text on a blue background

Attention

- Use high contrast color combinations
- Use colors consistently
- Use only 2 levels of intensity on a single screen
- Use underlining, bold, inverse video or other markers sparingly
- On text screens don't use more than 3 fonts on a single screen

Display issues

- Maintain display inertia - make sure the screen changes little from one screen to the next within a functional task situation
- Organize screen complexity
- Eliminate unnecessary information
- Use concise, unambiguous wording for instructions and messages
- Use easy to recognize icons

Display issues

- Use a balanced screen layout - don't put too much information at the top of the screen - try to balance information in each screen quadrant
- Use plenty of 'white space' around text blocks - use at least 50% white space for text screens
- Group information logically
- Structure the information rather than just presenting a narrative format (comprehension can be 40% faster for a structured format)

Individual differences

- Accommodate individual differences in user experience (e.g. from the novice to the computer literate, young to old)
- Accommodate user preferences by allowing some degree of customization of screen layout, appearance, icons etc.
- Allow alternative forms for commands (e.g. key combinations through menu selections)

Usability Testing Process

- Define interface issues

- Define users
- Define scenarios
- Define methods
- Conduct testing
- Analyze data
- Formulate recommendations