User Interface Design: An Introduction and Overview

Joseph A. Konstan
Dept. of Computer Science & Engineering
University of Minnesota
konstan@cs.umn.edu
Topics in User Interfaces

- Understanding Humans -- Psychology
- Human-Computer Interaction
- Design Process and Strategies
- Interface Evaluation
- Tools for Interface Development
- Technology of Interfaces and Tools
Goals for Today

- Overview of Field
- A Sampling of Psychology
- A Design Process -- TCUID
- Some Usability Engineering Issues
- a little pitch for further education …

- Have some fun, play some games!
Psychology

- Human capabilities and limitations
- Perception and cognition
- Implications for UI design

- *Design of Everyday Things*
  by Donald Norman
A Two-Player Game

- Start with the numbers 1 ... 9
- Pick alternatingly without replacement
- A winner has exactly 3 numbers that add up to 15
- If all numbers are used, and nobody wins, it is a draw
A Two-Player Game

<table>
<thead>
<tr>
<th></th>
<th>8</th>
<th></th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>1</td>
<td>5</td>
<td>7</td>
</tr>
<tr>
<td>4</td>
<td>9</td>
<td>2</td>
<td></td>
</tr>
</tbody>
</table>
Human Capabilities

● Humans are very good at:
  » recognizing (images, voices, etc.)
  » associative memory
  » explaining phenomena

● Humans are very limited in:
  » short-term memory
  » complex, multi-layered tasks
  » perfection
Brain Hemisphere Research

- “Left Brain”
  » methodical, logical, step-by-step
  » symbolic, works with components
  » generally dominant

- “Right Brain”
  » holistic, intuitive, rapid
  » handles missing values
  » works with gestalts
Limits of Human Memory

- **Short-Term Memory**
  - instant recall
  - limited capacity
  - fragile

- **Long-Term Memory**
  - slower recall, depends on organization
  - rote memory vs. relationships vs. explanation
  - “muscle memory”
Models of User Action

- Goals
- Intention to Act
- Action Plan (sequence of action steps)
- Observe State of World
- Interpret State
- Evaluate to form new...
- Change the World
- Execution of Actions
Humans Err

- Humans are not perfect!
- Slips -- errors in automatic actions
  » tied to skilled behaviors
  » easy to detect
- Mistakes -- errors in intention or logic
  » e.g., false generalizations
  » may be hard to detect
Where Does This Put Us?

● The Problem
  » humans are imperfect!!

● Possible Solutions
  » yank them out of the process
    – lose benefits of human strengths
  » design for imperfect users
Put Support into the Interface

- Affordances
- Visibility of Controls
- Feedback
- Conceptual Models
- Mappings

- Information in the World
- Constraints
- Error Avoidance and Handling
- Standardization
Affordances

● What something can be used for
  » a button (or plate) affords pushing
  » a knob affords turning

● Cultural (and learned) affordances
  » a scrollbar affords scrolling
  » various cursors afford operations

● Key: helps the user discover possibilities
  » where would you hide a safe in your house?
Visibility of Controls and Information

- Don’t hide the controls!
  - telephone systems: hold, transfer, etc.
  - VCR programming
- Make status available
  - well-designed display (34% complete)
  - use sound if needed (click/beep/etc.)
- Don’t distract with irrelevant displays
  - dynamics and prominence reflect importance
Feedback

● Don’t hide the results!
● Make feedback immediate
  » did I hit the button? (visual or audio)
  » did I have an effect? (cursor change?)
● Each action should have an effect
  » promote exploration
Conceptual Models

- Rote memorization prevents inference and adaptation
  - users *will* develop conceptual models
    - but they will likely be wrong!

- Models should help people adapt to new situations
  - gulf of execution -- not knowing *how*
  - gulf of evaluation -- not knowing *whether it worked*
Mappings

- Humans infer from mappings
  - layout of light switches in a room
  - controls on a range
- Natural mappings are easiest, but ...
  - avoid mappings that don’t generalize
Information in the World

● Avoid relying on memory alone
  » menus and toolbars
● Support memory aids
  » never require remembering information between screens
● Great precision is not required
Constraints

- Narrow the task search space
- Physical Constraints
- Semantic Constraints
- Cultural Constraints
- Logical Constraints
Error Avoidance/Handling

● Design to prevent slips
  » different things should look different
  » consistent confirmation is useless
  » immediate confirmation can be nearly useless

● Simplify tasks
  » make decision trees narrow or shallow
Error Avoidance/Handling

- Support recovery from errors
  - undo operations and back-up versions
  - support exploration towards a goal
- Prevent errors with forcing functions
  - don’t make illegal operations available
  - disable buttons or menus
  - turn illegal operations into legal ones
Standardization

● If all else fails, ...
  » fewer things to memorize
  » shorter learning time
  » clocks should run clockwise
Examples

- Stove Control Design
- Refrigerator controls
- Light Switches
- One-button slide projectors
- Doors
- Phones
Stove Control Design
Examples

- Stove Control Design
- Refrigerator controls
- Light Switches
- One-button slide projectors
- Doors
- Phones
Refrigerator Controls

NORMAL SETTINGS  C  AND  5
COLDER FRESH FOOD  C  AND  6 - 7
COLDEST FRESH FOOD  B  AND  8 - 9
COLDER FREEZER  D  AND  7 - 8
WARMER FRESH FOOD  C  AND  4 - 1
OFF (FRESH FD & FRZ)  0
Examples

- Stove Control Design
- Refrigerator controls
- Light Switches
- One-button slide projectors
- Doors
- Phones
The UI Design Process

- Several processes “promoted”
- Common elements
  » Focus on users
    – tasks, scenarios
    – activities, work context
    – communication
    – personas
Task-Centered User Interface Design

- Identify users and tasks
- Develop tasks into scenarios
- Use tasks/scenarios in design and evaluation

- Based on book by Lewis and Rieman
Users

● Who is going to use the system?
  » if you can’t find a user -- you’re in trouble
  » “everyone” is not a user
  » “the designer” is not a good user

● Go talk with the user
  » too busy?
    – how will they have time to evaluate/use it?
    – are there good surrogate users?
Talking with the Users

- What do they know?
  - systems, skills, etc.
- What do they do?
  - tasks
- How do they do it now?
  - scenarios
- What do they want to do?
  - new tasks
Users Sometimes Bite!

- Users aren’t all-knowing
  - they may not understand the possibilities
  - they may have a very narrow view

- They aren’t designers
  - learn about the tasks from the users
  - use your design skills to create a design
  - get user feedback on the design/prototype
Tasks

- **What is a task?**
  - a specific description of a complete job that specific users want to accomplish
  - not tied to how they would do the job

- **Detailed**
  - some typical details are important

- **Complete job**
  - covers transitions between sub-tasks
Example Task

- Professor Konstan receives a phone call from his department head asking whether he can attend a one-hour meeting the following Friday morning at 9. He should check his calendar, add the meeting unless he is teaching or traveling then, and send e-mail to reschedule any appointments that have to be missed for this meeting.
Why Tasks

- Tasks are fundamental to TCUID
  - determine who actually uses the system
  - sets goals for system functionality
  - basis for system design
  - basis for comparative evaluation
  - basis for user testing
How Many Tasks?

- Depends on nature of problem
  - 3-5 general-purpose tasks for a simple system
  - separate tasks for special-purpose cases (maintenance, installation)
  - 10+ tasks for complex systems
  - depth/quality more important than number of tasks
From Task to Design

- Write-up tasks, circulate among users
  » clarify missing details
- Rough out an interface, using existing systems or designs where possible
- Sketch out how each task would be accomplished in the interface: develop scenarios
Scenarios

● Specific instance of system use
  » for a particular task
  » for a particular interface
  » what would the user do, in detail

● Example
  » double-click on Outlook icon, click the calendar icon, …
Properties of Scenarios

● Interface-dependent
● Detail appropriate to user, task, interface
● Brings forward issues
  » how components work together
  » design arguments
  » tricky parts of the interface
● Guideline to create prototype
Interface Design Strategies

● Find a tool that does all/part of the job
  » don’t write a new spreadsheet -- extend!
  » you won’t live long enough to re-invent Excel
● Work within an existing framework
● Borrow intelligently
  » know why the interactions were selected
    – Mac tool palette vs. menus
● Invent only when absolutely necessary
Interface Prototyping

● Why prototype?
  » easier/cheaper than building & discarding
  » learn about interface problems early
    – before extensive resources committed
  » identify hard parts of the design

● Can you use the final prototype as the product?
  » often
Goals of Interface Prototyping

- Bring out issues that are hard to see in the abstract
- Better gestalt for the interface
- Something to evaluate using heuristics
- Something for users to evaluate
  - informally
  - user testing
Prototyping Techniques

- Functioning Programs
- Stand-Alone Interfaces
- Dedicated Prototypes
- Paper Prototypes
A Surprising Finding

- In many circumstances, sketches work \textit{better than} higher quality prototypes for user evaluation.
  - users feel freer to suggest major changes
  - users focus on high level rather than color, labels, graphical details
  - some groups have generated sketches from high-quality prototypes for focus groups and other user evaluations.
TCUID Summary

● Who is going to do what?
● Choose representative tasks
  » scenario for current systems
● Rough out a design, borrowing where possible
● Think, evaluate
● Create a prototype
● Test it (with and without users)
● Iterate
● Build and maintain it
Interface Development Methodology

• Prototype and Iterate
  » keep iterating until it is good enough
  » evaluate along the way to assess

• What is Good? What is Good Enough?
  » set usability goals
  » should relate to tasks
Evaluation

- Without users
  - walkthroughs
  - heuristic/checklist
  - action analysis

- With users
  - test design/evaluation
Tricky Issues in Usability Engineering

- Not software engineering
  » don’t know requirements or specs
- Prototype/iterate
  » when to stop
- Quantitative usability goals?
Yeah, But Why Should I Care?

- Usability = $$$
  - Support costs
  - Reputation
  - Product reviews
Yeah, But What Can I Do?

- Hire people with HCI/UI background
  - Psych & Computer Science
- Make people aware of issues
- Train people!
  - Good place to pitch courses ....
Reference Materials

● Courses and Conferences
  » UPA (in two weeks, Scottsdale)
  » CHI 2004 (Vienna); CHI 2005 (Portland)

● Books
  » Highlights -- no system-specific books

● On-line resources
  » Well-connected on the web
Courses and Conferences

● Typical Computer Science Courses
  » UI Design, Evaluation, and Implementation
  » GUI Toolkits and their Implementation
  » HCI and UI Technology
  » Specialty Topics (CSCW, Ubicomp, Wearables, etc.)

● Annual Conferences
  » CHI*, UPA, CSCW, UIST, DIS/DUX, IUI, and many more …
  » see SIGCHI home page for details
References for Further Reading

- *Task-Centered User Interface Design* by Clayton Lewis and John Rieman
- *The Design of Everyday Things* by Donald Norman
- *A Guide to Usability* by Jenny Preece
- *Usability Engineering* by Jakob Nielsen
- *Developing User Interfaces* by Dan Olsen
References for Further Reading

- *Designing the User Interface* (3rd edition) by Ben Shneiderman
- *Human-Computer Interaction* by Jenny Preece et.al.
- *Developing User Interfaces: Ensuring Usability through Product and Process* by Hix and Hartson
- *Cost-Justifying Usability* by Bias and Mayhew
- *Readings in Human-Computer Interaction* (1st and 2nd editions) edited by Ronald Baecker, et. al.
- *Handbook of Human-Computer Interaction* (2 editions, edited by Martin Helander)
References for Further Reading

- *Interactive System Design* by Newman and Lamming
- *Bringing Design to Software* edited by Terry Winograd
- *The Art of Human-Computer Interface Design* by Brenda Laurel
- *The Visual Display of Quantitative Information* by Edward Tufte
- *The Human Computer Interaction Handbook* by Julie Jacko and Andrew Sears
Useful Resources on the Internet

● HCI Reference Pages
  » http://www.usableweb.com/
  » http://www.degraaff.org/hci/
  » http://www.hcibib.org/

● ACM SIGCHI
  » http://www.sigchi.org/

● Usability Professionals Association
  » http://www.upassoc.org/