Human-Computer Interaction

YOU HAVE CHRONIC MAKJOBBS CRAPPUS
BUT THATS NOT WHY YOU PUKED.

HAVE YOU BEEN EXPOSED TO ANY
USER INTERFACES DESIGNED BY
ENGINEERS?

YOU HAVE INTERFACE
POISONING. YOU'LL
BE DEAD IN A WEEK.

Copyright © 2003 United Pasteur Syndicate, Inc.

Credits

- This weeks slides provided in part by Dr. Blair Nonnecke
- In part from: http://www.cs.siue.edu/hci/NSF/NSF_Semester.htm
- In part taken from:
  - Designing Interactive Systems: People, Activities, Contexts, Technologies
    - David Benyon
    - Phil Turner
    - Susan Turner
Definition of HCI

- **Human-computer interaction (HCI)** is:
  “concerned with the design, evaluation and implementation of interactive computing systems for human use and with the study of major phenomena surrounding them” (ACM SIGCHI, 1992, p.6)

Practical Goals of HCI

- Develop usable products
- Involve users in the design/evaluation process
- Practice throughout out software life cycle
What do HCI professionals do?

- **interaction designers** - people involved in the design of all the interactive aspects of a product
- **usability engineers** - people who focus on evaluating products, using usability methods and principles
- **web designers** - people who develop and create the visual design of websites, such as layouts
- **information architects** - people who come up with ideas of how to plan and structure interactive products
- **user experience designers** - people who do all the above but who may also carry out field studies to inform the design of products

Usability Definitions

- **ISO**
  - The **effectiveness**, **efficiency**, and **satisfaction** with which specified users achieve specified goals in particular environments

- **Preece et al**
  - A measure of the ease with which a system can be learned or used, its **safety**, **effectiveness** and **efficiency** and the **attitude** of its users towards it
Usability goals

- Effective to use (produce desired effect)
- Have good utility (be useful)
- Efficient to use
- Easy to learn
- Easy to remember how to use
- Safe to use
- Satisfying to use

Usability Goals for a VCR

- How long should it take and how long does it actually take to:
  - use a VCR to play a video?
  - use a VCR to pre-record two programs?
  - use an authoring tool to create a website?
Why Study HCI?

- Usability of design is often an afterthought, something to be added at the end
- Management doesn’t understand how good designs are created
- A good UI is not an inspiration but the result of a process (and lots of hard work)
- Following UI level guideline is not enough

Why Study HCI (cont.)

- Most programmers believe they are UI experts
- Most programmers think users are just like themselves
- Knowing how to use a UI development tool, doesn’t mean you can make a usable interface
- Good design means profit/efficiency
- Makes YOU more employable/marketable
**How to design**

Need to take into account:

- **Who** the users are
- **What** activities are being carried out
- **Where** the interaction is taking place

**Example:**

- compare the differences between a cell phone and a public pay phone
- how do these differences manifest themselves in the design and use of phones?

---

**Users’ needs**

- understand what people are good and bad at
  - easier to recall than remember (menus vs. command line)
- understand how they currently do things and then make it better, e.g., how do people read hardcopy magazines and how would you make it better in softcopy?
Users’ needs (cont)

- listen to what people want and get them involved
- use tried and tested methods, e.g.,
  - observation
  - prototyping
  - questionnaires
  - interviews

Some Design Guidelines

- prototype, prototype, prototype
- evaluate with ‘real’ users
- have at least one alternate design
- develop in B&W first
- use scenarios/use cases to validate, focus, and evaluate design
More Design Guidelines

- provide feedback
- speak the users language (not techno-geek)
- provide help
- prevent errors
- be consistent (links, terminology, layout, etc.)

Design Principles

- Over the years many principles of good interactive system design have been developed.
- Design principles can be very broad
  - such as 'make things visible'
- They can be more specific
  - such as 'provide clearly marked exits'
- There are also good design principles that derive from psychology
  - such as 'minimize memory load' (i.e. do not expect people to remember too much).
  - 7 +/- 2
Principles and Patterns

- Has led to established patterns of interaction in certain circumstances
  - such as the ‘undo’ command in a Windows application, or the ‘back’ button on a Web site
  - the greying out of inappropriate options on menus.

- Design principles can
  - guide the designer during the design process
  - can be used to evaluate and critique prototype design ideas.

- Help to orientate the designer to key features of good design and sensitize the designer to important issues.

Overall view of good design

- Designing interactive systems from a human-centred perspective is about…
- …helping people access, learn and remember the system…
- …giving them the sense of being in control, knowing what to do and how to do it…
- …safely and securely…
- …in a way that suits them.
Top level principles

- Three top level principles
  - Learnability, effectiveness and accommodation
  - Twelve main principles categorized in the three top level categories

- Systems should be
  - Learnable. Principles 1–4 are concerned with access, ease of learning and remembering
  - Effective. Principles 5–7 are concerned with ease of use and 8–9 are concerned with safety
  - Accommodating. Principles 10–12 are concerned with accommodating differences between people and respecting those differences

Principle 1 - Visibility

- Try to ensure that things are visible so that people can see what functions are available and what the system is currently doing.
- This is an important part of the psychological principle that it is easier to recognize things than to have to recall them.
- If it is not possible to make it visible, make it observable.
- Consider making things ‘visible’ through the use of sound and touch.
The common commands and defaults are made visible.
Visibility and sensible grouping makes people aware of other options.

**Figure 5.4** A formatting palette.

---

**Principle 2 - Consistency**

- Be consistent in the use of design features.
- Be consistent with similar systems and standard ways of working.
- Conceptual consistency is about ensuring the mappings are consistent, that the conceptual model remains clear.
- This involves being consistent both internally to the system and externally as the system relates to things outside of it.
- Physical consistency is ensuring consistent behaviours and consistent use of colours, names, layout and so on.
Principle 3 - Familiarity

- Use language and symbols that the intended audience will be familiar with.
- Where this is not possible because the concepts are quite different from those people know about, provide a suitable metaphor to help them transfer similar and related knowledge from a more familiar domain.
- e.g. recycle bin = delete?

Principle 4 - Affordance

- Design things so it is clear what they are for;
  - for example make buttons look like buttons so people will press them.
  - Make a slot for inserting a credit card look like a credit card slot!
- Affordance refers to the properties that things have (or are perceived to have) and how these relate to how the things could be used.
- Buttons afford pressing, chairs afford sitting on and post-it notes afford writing a message on and sticking next to something else.
- Affordances are culturally determined.
Media Player

Buttons on media players have affordance - they suggest what they can be used for because we are familiar with similar devices.

**Figure** The RealOne Player ® with two slider controls.  
*Source: courtesy of RealNetworks, Inc.*

---

**Top level principle of Learnability consists of**

- Principle 1 Visibility
- Principle 2 Consistency
- Principle 3 Familiarity
- Principle 4 Affordance
Principle 5 - Navigation

- Provide support to enable people to move around the parts of the system; maps, directional signs and information signs.
- Menus are often used for navigation, signs (labels) indicate where else you can go in the system.

Principle 6 - Control

- Make it clear who or what is in control and allow people to take control.
- Control is enhanced if there is a clear, logical mapping between controls and the effect that they have.
- Also make clear the relationship between what the system does and what will happen in the world outside the system.
Principle 7 - Feedback

- Rapidly feed back information from the system to people
- so that they know what effect their actions have had.
- Constant and consistent feedback will enhance the feeling of control.
- E.g. guitar
- Pressing a button or pulling a lever

Safe and Secure

- Principle 8 - Recovery
  - enable recovery from actions, particularly mistakes and errors, quickly and effectively.
- Principle 9 - Constraints
  - provide constraints so that people do not try to do things that are inappropriate.
  - In particular people should be prevented from making serious errors
  - through properly constraining allowable actions and seeking confirmation of dangerous operations.
**Top - level principle of Effectiveness**

- Ease of Use which consists of
  - Principle 5 - Navigation
  - Principle 6 - Control
  - Principle 7 - Feedback
- Safe and Secure which consists of
  - Principle 8 - Recovery
  - Principle 9 - Constraints

**Principle 10 - Flexibility**

- Allow multiple ways of doing things
- so as to accommodate users with different levels of experience and interest in the systems.
- Provide people with the opportunity to change the way things look or behave
- so that they can personalize the system.
Aesthetics

- Principle 11 - Style
  - designs should be stylish and attractive

- Principle 12 - Conviviality
  - Interactive systems should be polite, friendly, and generally pleasant.
  - Nothing ruins the experience of using an interactive system more than an aggressive message or an abrupt interruption.
  - Conviviality also suggests joining in and using interactive technologies to connect and support people.

Polite Software

<table>
<thead>
<tr>
<th>Is interested in me</th>
<th>is taciturn about its personal problems</th>
</tr>
</thead>
<tbody>
<tr>
<td>Is deferential to me</td>
<td>is well informed</td>
</tr>
<tr>
<td>Is forthcoming</td>
<td>is perceptive</td>
</tr>
<tr>
<td>Has common sense</td>
<td>is self-confident</td>
</tr>
<tr>
<td>Anticipates my needs</td>
<td>stays focused</td>
</tr>
<tr>
<td>Is responsive</td>
<td>is fudge-able</td>
</tr>
<tr>
<td>Gives instant gratification</td>
<td>is trustworthy</td>
</tr>
</tbody>
</table>

34
Twelve Principles for good human-centred design

- 1. Visibility
- 2. Consistency
- 3. Familiarity
- 4. Affordance
- 5. Navigation
- 6. Control
- 7. Feedback
- 8. Recovery
- 9. Constraints
- 10. Flexibility
- 11. Style
- 12. Conviviality

Designing for Windows Applications

- The familiar combination of windows, icons, menus and pointer, called a WIMP interface.
- Designing for Windows applications is still dominated primarily by issues of usability.
- In particular the key issue is consistency.
  - There are clear guidelines for issues such as menu layout, ordering, dialogue boxes and use of the other 'widgets' associated with graphical user interfaces.
Designing for Windows Applications (continued)

- There are standards for providing **constraints** such as greying out items on a menu that are not relevant at a particular point.
- A tool-kit, or design environment such as Visual Basic will probably be used that will help to ensure the design confirms to an overall style.

Screen design

- is a key issue in Windows environments
- attention needs to be paid to the layout of objects on a screen.
- Avoiding clutter will help to ensure **visibility**.
  - Attention needs to be paid to the use of appropriate, non-clashing colours and the careful layout of information using tables, graphs or text as appropriate.
- Often the designer can talk to the actual future users of the system
  - This will help the designer to ensure **familiar** language is used.
- A good design will ensure that there is easy error recovery e.g. 'are you sure you want to destroy the database!'
  - A good example of designing for recovery is the undo command
Affordances In Windows

- are provided by following Windows design guidelines.
- People will expect to see a menu at the top of the screen and will expect the menu items to be displayed when the header is clicked on. Items that are not greyed out will afford selecting.
- The various ‘widgets’ such as check boxes, radio buttons and text entry boxes should afford selecting because people are familiar with the standards.
- Care needs to be taken to ensure that opportunities are easily and correctly perceived.

Navigation in Windows

- People move around the application by selecting items from menus and then by following dialogue structures.
- Many Windows applications make use of ‘wizards’.
- These provide step by step instructions for undertaking a sequence of operations, allowing users to go forward and backwards to ensure that all steps are completed.
Control

- Users have to initiate actions,
- some features that provide security features are undertaken automatically.
- Many applications, for example, automatically save people’s work
- this helps with recovery if mistakes are made.

Feedback

- is provided in a variety of ways.
- A ‘bee’ symbol or an ‘egg timer’ symbol are used to indicate that the system is busy doing something.
- Counters and progress bars are used to indicate how much of an operation is complete.
- Feedback can be provided through sound such as a beep when a message is received on an e-mail system
- or a sound to indicate that a file has been safely saved.
Flexibility

- is provided with things such as short-cut keys allowing more expert users to use combinations of keyboard controls in place of using menus to initiate commands and navigate through the system.
- Many Windows applications allow the user to set their own preferences
- to configure features such as the navigation bars, the menu items and to disable features that are not often used.

Style and conviviality

- Windows applications are rather limited as they should remain within the standard design guidelines.
- Error messages are one area where the designer can move towards a more convivial design
  - by thinking hard about the words used on the messages.
  - All too frequently messages appear very abruptly and interrupt people unnecessarily.
Web Page Design

- Users focus on the top-left portion of the screen first
- Web pages must be able to answer:
  - Where am I?
  - What's here?
  - Where can I go?
- Back button will be used… make it work!
- Know your user, e.g., screen size, download speed, etc.
- Users will always double and triple click

Prototype

- Definition of a prototype
- A model on which something is based.
- A reasonably close facsimile of the final product
Why Prototype?

✓ A tool to validate the user interface is usable and reveals the structure clearly
  • Provides a structure that fits the user’s work model

✓ A tool for communication between the design team and the user
  • User’s vocabulary
  • User becomes a co-designer
  • Iterative refinement

Prototyping Medium

✓ High Fidelity Prototypes
  • A.K.A. Hi-fi, Rapid Prototyping Tools
  • Examples
    ▪ Visual Basic
    ▪ HTML and scripting (ASP, Java)
    ▪ Director
    ▪ Powerbuilder
    ▪ Hypercard
  • Black Box Coding
Prototyping Medium

✓ Low Fidelity Prototypes

• A.K.A. Lo-fi, Paper Prototyping

• Examples
  ▪ Paper
  ▪ Post-it’s
  ▪ Tape
  ▪ Glue

• Play Computer

---

Paper Prototyping

■ What Is Paper Prototyping?
  ■ Early design tool.
  ■ Used to test a program’s usability.
  ■ Use simple office materials to create a paper-based simulation of an interface.
  ■ Rough draft thus open to correction and errors.
  ■ Lo-fidelity.
Paper Prototype Kit

Useful things to have in your kit
- Post-it notes (multiple sizes)
- Post-it tape
- Index cards
- Sturdy Paper / Index Folders
- Transparencies
- Pointer or pencil
- Hardware Props
- Scissors, straight edge

Lo-fi vs. Hi-fi


time to build

• Lo-fi:
  ▪ Quick to build
  ▪ 2 to 3 hours depending on number of screens and complexity

• Hi-fi:
  ▪ Screen quick, but black boxing can take a significant time commitment
  ▪ 2 to 3 weeks depending on complexity and team size
Quality of Feedback

• Lo-fi:
  ▪ Maintains focus on work structure and interface elements

• Hi-fi:
  ▪ Reviewers tend to focus on finishing touch issues
    ▪ Fonts, alignment, colors

Changeability

• Lo-fi:
  ▪ Invites change
  ▪ Because it is easy to change, changes can be made on the fly

• Hi-fi:
  ▪ Developers resistant to change
  ▪ Too much time and effort invested – feeling of ownership
<table>
<thead>
<tr>
<th>Score Board</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="https://via.placeholder.com/150" alt="Image" /></td>
</tr>
</tbody>
</table>

**Lo-fi vs. Hi-fi**

**Feature Creep (Creap)**

- **Lo-fi:**
  - Gives the impression that development is still in design

- **Hi-fi:**
  - Gives the false impression of being done

**Program bugs**

- **Lo-fi:**
  - No program bugs – no problem

- **Hi-fi:**
  - Single bug can bring prototype session to a halt (or kill a project)

**Users as Co-Designers**

- **Lo-fi:**
  - Invites the user to provide suggestions

- **Hi-fi:**
  - Gives the impression of being done


TIMING STUDIES

- Lo-fi:
  - Cannot be done
- Hi-fi:
  - Can be done

ANIMATION

- Lo-fi:
  - Cannot be done
- Hi-fi:
  - Can be done, but sometimes complicated

AND THE WINNER IS . . .

LO-FI!

FINAL SCORE

<table>
<thead>
<tr>
<th>Score Board</th>
<th>Lo-fi</th>
<th>Hi-fi</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>★★★★</td>
<td>★★</td>
</tr>
</tbody>
</table>

29
Running a Paper Prototype Session

- Roles
  - Facilitator
  - Human-Computer
  - Observer/note taker (video tape)

- Prepare Test Scenarios
  - Cover the functionality of the system you are testing

- Practice
  - The human-computer should practice the responses so that the transition of the screens are smooth.

During the Session

- Introduction
  - Give a general idea of what the user is going to do
  - DO NOT give an overall guide to the application
  - It is important to instruct the user to “Think out loud”
    - You may need to gently remind them of this during the session

- Start with an easy task or two to put the user at ease

- If the user sees something unexpected then ask what they did expect
  - Try making an on-the-fly design change
During the Session

- If they ask for help
  - **Do not give it to them.** The idea is to see if the program design makes sense to them.
  - Let them make mistakes. This gives your design team places to consider changes.
  - If they get completely lost then help them get back on track.
- Follow-up on problems and design ideas
- Ask about recommendations

After the Session

- Hold a design team post mortem meeting
  - Discuss the areas that the user seemed to have problems with or made suggestion about
  - Consider design alternatives
- Iterative Refinement
  - Make design changes or changes to the scenarios before next session