Interfaces

Human-Computer Interaction

Graphical User Interfaces
Learning Goals
you should be able to

• explain how tools augment and constrain our power to think and act, define the “myth of human error” and give examples that dispel this myth;

• explain strengths and weaknesses of human-computer interfaces, referring to concepts such as familiarity and consistency, mappings and metaphors, feedback, negative transfer, or additional concepts that you identify.
Tools constrain our power to think (and act)

Example: Roman number system:

<table>
<thead>
<tr>
<th>I</th>
<th>II</th>
<th>III</th>
<th>IV</th>
<th>V</th>
<th>VI</th>
<th>VII</th>
<th>VIII</th>
<th>IX</th>
<th>X</th>
<th>XI</th>
<th>...</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>10</td>
<td>11</td>
<td></td>
</tr>
</tbody>
</table>

M: 1000   D: 500   C: 100   L: 50   X: 10   V: 5   I: 1

(try multiplying MMMMCMXCIX * MMMCMLIV)
The Myth of Human Error

• **myth**: faulty use of technology is usually the result of human error

• **fact**: many so-called human errors are actually errors in design
Being a good pilot, requires knowing what to do if the engine stalls.

When pilots deliberately stalled the engine, an alarm would sound (don’t land the plane with the wheels up). So they’d turn off the alarm.

→ conditioned response

stall → button;

stall → button;
The Myth of Human Error

Problem: negative transfer

What happens now when a pilot stalls?

The Harvard control panel
- horn cut-out button

The T-33 control panel
- spare fuel tank jettison button
An interesting observation:
The power button on nearly every electronic device is a circle with a vertical line inside, the eject symbol is always a horizontal line below an upward pointing arrow. All of these examples have become natural instinct to many people across the globe.
HCI research @ UBC

involves researchers from CS, Psychology, Commerce, Forest Resource Management and Engineering - see, e.g.:

- Variant view: visualizing sequence variants in their gene context.  

- “Multi-Layered Interfaces to Improve Older Adults’ Initial Learnability of Mobile Applications”  
  http://dl.acm.org/citation.cfm?doid=1838562.1838563
Learning Goals [review]

You should be able to:

- explain how tools augment and constrain our power to think and act, define the “myth of human error” and give examples that dispel this myth;

- explain strengths and weaknesses of human-computer interfaces, referring to concepts such as familiarity and consistency, mappings and metaphors, feedback, negative transfer, or additional concepts that you identify.
Interfaces

Text-based Interfaces
Learning Goal
you should be able to

• use basic features of text-based interfaces such as Unix or search engines, with knowledge of the ways that special symbols are interpreted (or misinterpreted) by such interfaces.
RQ: have things always been this way?

Given that interface tools are known to constrain our ability to think and act, why hasn't the human-computer interface evolved a novel change in structure since its development? We still use, in most cases, a hard wired keyboard and/or mouse.

Why do some applications require interfaces to function whereas some do not?
Human Computer Interaction (HCI)

- The early days
Human Computer Interaction (HCI)

- The early days
- Punched cards
Human Computer Interaction (HCI)

- The early days
- Punched cards
- Terminals and keyboards
  - text based interfaces
RQ:

As the technology drawn on for computer metaphors changes, might certain metaphors be rendered unintelligible? For instance, the use of the floppy disk for "save" has outlived the floppy disk itself now, and future generations may find the icon puzzling.
RQ: skeuomorph

A **skeuomorph** /ˈskjuːəmɔrf/ is a derivative object that retains ornamental design cues from structures that were necessary in the original.

http://en.wikipedia.org/wiki/Skeuomorph
Human Computer Interaction (HCI)

- The early days
- Punched cards
- Terminals and keyboards
  - text based interfaces
- The mouse
  - graphical user interfaces

... but text is still important!
Unix

• operating system deployed in the early 1970s to enable real-time sharing of computing resources among multiple users and tasks

• supports purely text-based commands to act on data

• There are many flavours of Unix. Linux is a follow up to Unix. Apple’s OS X is another. Want Unix-like commands on your PC? http://www.cygwin.com/
Unix Commands

expressiveness is provided through:

– *different commands*, e.g. `cp`
– *options* specified after a – sign, e.g. `ls`
– combining commands using the *unix pipe* `|`
– *redirecting* output displayed on the screen into a file using `>`
More Unix Commands

- `more file1 ... filek`: displays file contents one screenful at a time
- `sort file1 ... filek`: sorts lines of all files together and lists the result
- `sort -u file1 ... filek`: remove duplicate lines from the sorted list
- `grep string file1`: search a file for a string
Unix Commands

• examples:
  – `sort file*`: sort all files starting with `file`
  – `ls | more`: list contents of current directory and display one screenful at a time
  – `ls > file`: store the list of contents of the current directory in `file`
Unix Commands

navigating directories

- special directory names:
  - ~: home directory
  - ..: parent directory
  - .: current directory
Absolute vs relative paths

Absolute paths:

- specify location of a file (or directory) from the home (or root) directory
- use “~” to access home directory, “/” to access root directory
- useful to access data without worrying about what the current (working) directory is at the time
Absolute vs relative paths

Relative paths:

- specify location of a file (or directory) from the current (working) directory
- often use ".." to access higher-level directories
- useful for accessing data in a way that does not have to be changed when parts of the directory tree are copied or moved to another location.
Unix – tips & tricks

• Learn the special symbols (*, >, |, ~, ..).
• Know the difference between absolute and relative paths.
• Know how to navigate the directory tree.
• Watch out for subtle syntax errors (missing / extraneous spaces, \ instead of /).
• In Unix, use the ‘man’ command to get help and examples of use for other commands (e.g., man ls).
• Analyse sequences of commands one command at a time: What are the effects of each command?
• Practice and experiment a lot – this will be highly relevant for both of your exams (and matters much in the real world)!
Unix Command Summary

- syntax is critical
  - spelling errors cause commands to fail
  - extraneous spaces cause commands to be misinterpreted
- special symbols can be used in restricted ways
  - awkward to have special symbols e.g., spaces *
  - \ | in file names
  - workaround: putting \ before a symbol in a file name ensures it is interpreted correctly
How to shoot yourself in the foot in UNIX

% ls
  foot.c foot.h foot.o toe.c toe.o
% rm * .o
rm: .o: No such file or directory
% ls
%

What happened?
If you do shoot yourself in the foot…

On our undergrad machines, you can change into the “.snapshot” directory.
Administrative Question

Do we have to know specifically how to use UNIX for the midterm, while the concept was not covered much both in the module or the lab?

Answer: Anything that was covered in any component of the course (lab, reading, class, ...) is relevant for your exams. Unix was covered in Lab 0 and in the reading for Module 2, so it is definitely relevant.
Learning Goal [review]
you should be able to

- use basic features of text-based interfaces such as Unix or search engines, with knowledge of the ways that special symbols are interpreted (or misinterpreted) by such interfaces.