Usability and User Experience

Interaction design

Establishing requirements

Developing alternatives

Prototyping

Evaluating

Why go to this length?

Help designers:

- understand how to design interactive products that fit with what people want, need and may desire
- appreciate that one size does not fit all

  e.g., teenagers are very different to grown-ups
- identify any incorrect assumptions they may have about particular user groups

  e.g., not all old people want or need big fonts
- be aware of both people's sensitivities and their capabilities

Usability

Usability is a combination of:

- How fast it is to learn
- How much you have to think, intuitiveness
- How fast users can perform tasks with it, efficiency
- How many mistakes users make with it
- How much users must remember to use it, memorization
- How comfortable people are when using it
- How aesthetically pleasing it is, simplicity of design

Usability goals

Effective to use

Efficient to use

Safe to use

Have good utility

Easy to learn

Easy to remember how to use

The User Experience

How a product behaves and is used by people in the real world

- the way people feel about it and their pleasure and satisfaction when using it, looking at it, holding it, and opening or closing it
- "every product that is used by someone has a user experience: newspapers, ketchup bottles, reclining armchairs, cardigan sweaters." (Garrett, 2003)

Cannot design a user experience, only design *for* a user experience

The iPod Nano Touch

Why was iPod user experience such a success?

Quality user experience from the start

Simple, elegant, distinct brand, pleasurable, must have fashion item, catchy names, cool, etc.,

User experience goals

Desirable aspects

satisfying
enjoyable
engaging
pleasurable
exciting
entertaining
helpful
motivating
challenging
enhancing sociability
supporting creativity
cognitively stimulating
fun
provocative
surprising
rewarding
emotionally fulfilling

Undesirable aspects

boring
frustrating
making one feel guilty
annoying
childish
unpleasant
patronizing
making one feel stupid
cutesy
gimmicky

Cultural Differences?

Key points

Interaction design

- designing interactive products
  - usability
- create quality user experiences
  - UX
- taking into account a number of interdependent factors, including context of use, type of activities, cultural differences, and user groups
- multidisciplinary

Design Principles

Design principles

Generalizable abstractions for thinking about different aspects of design

The do's and don'ts of interaction design

What to provide and what not to provide at the interface

Derived from a mix of theory-based knowledge, experience and common-sense

Important Concepts

Visibility

Feedback

Constraints

Consistency

Affordances

Visibility

When functionality is hidden, problems in use occur

Occurs when number of functions is greater than number of controls

When capabilities are visible, it does not require memory of how to use

Remind person how to use something

Visibility


WE COULD DESIGN THE PRODUCT WITH A SIMPLE POINT-AND-CLICK INTERFACE . .
OR WE COULD REQUIRE THE USER TO CHOOSE AMONG THOUSANDS OF POORLY DOCUMENTED COMMANDS, EACH OF WHICH MUST BE TYPED EXACTLY RIGHT ON THE FIRST TRY
BEAR IN MIND, WE'LL NEVER MEET A CUSTOMER OURSELVES.
MAKE IT SO THEY HAVE TO REBOOT AFTER EVERY TYPO.


Pedal?

Feedback

Let someone know what just occurred

Can be sound that's made

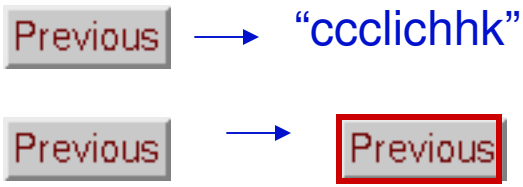
Can be change in physical state

Feedback

Sending information back to the user about what has been done

Includes sound, highlighting, animation and combinations of these

- e.g. when screen button clicked on provides sound or red highlight feedback:



Constraints

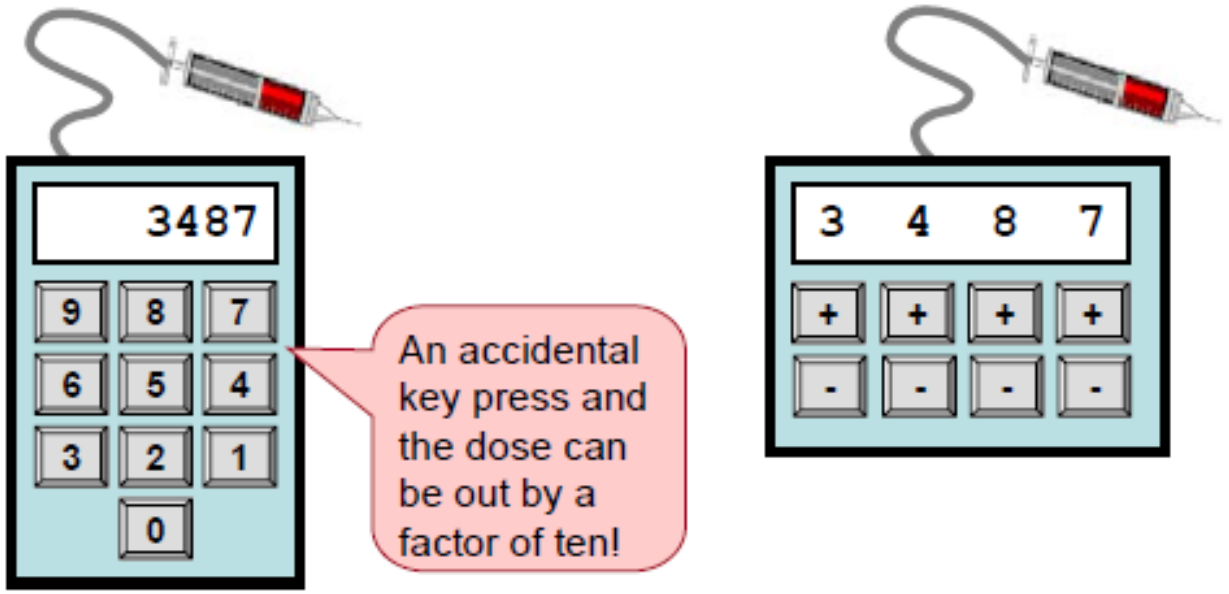
Restricting the possible actions that can be performed

Helps prevent user from selecting incorrect options

Physical objects can be designed to constrain things

- e.g. only one way you can insert a key into a lock

Decimal Point Errors Kill People[1]



[1] Thimbleby, H. (2008, September/October). Ignorance of interaction programming is killing people. *interactions*, 52-57.

Consistency

Design interfaces to have similar operations and use similar elements for similar tasks

For example:

- always use ctrl key plus first initial of the command for an operation – ctrl+C, ctrl+S, ctrl+O

Main benefit is consistent interfaces are easier to learn and use

When consistency breaks down

What happens if there is more than one command starting with the same letter?

- e.g. save, spelling, select, style

Have to find other initials or combinations of keys, thereby breaking the consistency rule

- e.g. ctrl+S, ctrl+Sp, ctrl+shift+L

Increases learning burden on user, making them more prone to errors

Internal and external consistency

Internal consistency refers to designing operations to behave the same within an application

- Difficult to achieve with complex interfaces

External consistency refers to designing operations, interfaces, etc., to be the same across applications and devices

- Very rarely the case, based on different designer's preference

Keypad numbers layout

A case of external inconsistency

(a) phones, remote controls

1	2	3
4	5	6
7	8	9
	0	

(b) calculators, computer keypads

7	8	9
4	5	6
1	2	3
0		

Close an Application or Window

Alt-F4

Ctrl-W

Ctrl-Q (works in PowerPoint ’03, but not in Excel)

Esc

Alt-Q

F10, Ctrl-C, Ctrl-Z

Double-click: top left corner; single – top right

Task manager: terminate

Affordance

What is it?

Visual affordance:

Perceived and actual fundamental properties of an object that determine how it could be used

Chair is for sitting

Ball is for throwing

Button is for pushing

What does ‘affordance’ have to offer interaction design?

Interfaces are virtual and do not have affordances like physical objects

Norman argues it does not make sense to talk about interfaces in terms of ‘real’ affordances

Instead interfaces are better conceptualized as ‘perceived’ affordances

- Learned conventions of arbitrary mappings between action and effect at the interface
- Some mappings are better than others

Activity

- Physical affordances:

  How do the following physical objects afford? Are they obvious?

Affordances: to give a clue

Refers to an attribute of an object that allows people to know how to use it

- e.g. a mouse button invites pushing, a door handle affords pulling

Norman (1988) used the term to discuss the design of everyday objects

Since has been much popularised in interaction design to discuss how to design interface objects

- e.g. scrollbars to afford moving up and down, icons to afford clicking on

Successful vs. Failed

Complex things may need explanation, but simple things should not

If a simple thing requires instructions and pictures, it is likely a failed design

Example

Affordances - Insert something into holes

Constraints - Bigger hole for several fingers, small for thumb

Users - Often for right handed users only

Why Design is Hard

Number of things to control has increased dramatically

Displays are more virtual/artificial

Marketplace pressure

- Adding operations cheaper (computers)
- Adding controls expensive (real estate, cost)

Errors are becoming increasingly serious

Technology Trends

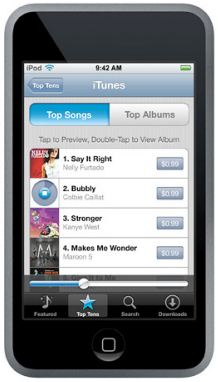
Technology Trends

Trends in technology have increased the need for improved UIs and UI design practices:

- Network/distributed systems allow access in remote locations, or across an enterprise
- Greater memory and faster processing are available at ever decreasing costs
- More people have access to computing power
- People are “on the move” (mobile computing)

Technology now exists for improved UIs and UI designs

Technology Trends (example)



UI issues for touch input

- How many fingers?
- Pressure thresholds (hysteresis)
- Taps vs. gestures
- Gesture recognition
- Performance benefits or deficits
- User learning and retention

New Technology is NOT Enough!

New interface technology alone does not produce usable interfaces!

- Graphical user interfaces (GUIs) are not intrinsically more usable than traditional console applications
- GUIs can be less usable if they are poorly designed

Usable interfaces require good design and a great deal of effort in their manufacture

User Interfaces: Code/Cost/Effort

Different Statistics

- UI is 47% - 60% of the total code
- The UI is minimally 29% of the software development project budget
- The UI may take as much 40% of the development effort

Interesting Stat

80% percent of software life cycle costs occur after the product is released, in the maintenance phase

- Of that work, 80% is due to **unmet or unseen user requirements**
- Only 20% of this is due to bugs or reliability problems

(Source: Karat, C. Usability engineering in dollars and cents. *IEEE Software*, May 1993, p 89.)

Good UIs are Hard to Build

Multiprocessing requirements

- Multiple inputs, redraws
- Synchronization, threads, deadlock prevention

Need abort, undo, redo

- Requires lots of state information to be kept

Real-time requirements

Must be robust

- Users do lots of odd things!

Good UIs are Hard to Build (2)

API & UI logic complexity

Reactive instead of proactive

- User dictates what the system should do

Hard to modularize (OO design helps)

Exhaustive testing of UIs is hard – how to ensure robustness?

Evaluation with users is time consuming

Where do we go from here?

Java GUI/Swing programming

Widgets

- Buttons, scroll panes, choosers, etc.

Event-driven programming

- Listeners, adapters

Model-View Controller

- Underlying architecture