Lecture Objectives

At the completion of this unit you should be able to:
- understand the basic principles of good interface design

Human– Computer Interface Design

The interface is the link between the users and the computer

To many users, the interface is the system

The Basics of Interface Design

- Five Commandments:
  - Support “Transportability of Knowledge”
  - Be Consistent
  - Provide Feedback
  - Use Drab Colours
  - Make the User Boss

Transportability of Knowledge

- Make use of what the user already knows
  - eg File>Exit quits Windows applications
- Answers lots of design questions for you
  - What are menus? - things that drop down from the top
  - What's the first menu option? - File
  - What's the last menu option? - Help
  - What does the word exit mean? - quit the application.

Axiom: Follow Microsoft’s Lead

- Most users already know how to drive Windows applications.
- Don’t scramble the positions of the brake, accelerator and clutch pedals!
- Grit your teeth at Microsoft’s design errors and reproduce them
Be Consistent

- Always do the same thing in the same way
  - it's always Exit
  - Exit only means “Leave this application”
  - Exit never means “Go back to the previous screen.” or
    “Go back to the splash screen.”
- The (Drop-Down) Menus are always the same.
  - If menu options are inappropriate in particular locations, dim them.
- The words you use to refer to objects are the same across screens.

Transportability and Consistency Work Together

- Any exception means:
  - The user always has to stop and think:
    - “Is this the exception?”
    - “Do I have to think twice about this one?”
  - So any breakdown in consistency or transportability means:
    - The user is slowed down
    - The user is frustrated
    - The user makes more mistakes

Consistency Is Harder Than You Think

- You have to remember what you did 3 days ago
- You have to keep track of what your team does
  - company/project standards are useful
- You may have to revamp your system because
  - You better understand what the user wants
  - You better understand how the system works
  - You better understand the task
- Start with the assumption you’ll probably get the interface wrong at the beginning.
- Budget time to go back over your system to make it consistent

Consistency in Appearance and Performance

- These aren’t the same.
- You can see inconsistency in appearance but not in performance
- How (and when) does your system respond to an input error?
- When do you prompt the users?
- How does the user access the Help system? (i.e., does F1 always work?)
- How must the user enter dates, percentages, etc.? Where do forms appear on the screen?

Feedback Lets the User Know What’s Going on

- Examples
  - A button goes down when you click it
  - The mouse pointer turns into an hourglass while short tasks run
  - There’s a progress bar for long tasks
- Objects always respond to actions on them. Something always shows you that it’s worked
- Negative Example
  - Fill in a form, click on the Save button
  - Nothing happens……The user is left in doubt.

Provide Feedback Without Pestering the User

- A pop-up a message box saying ‘the data has been saved’?
  - Probably not - why not just clear the data entry areas on the form?
- Good feedback
  - Isn’t intrusive
  - Doesn’t require action on the part of the user
  - Doesn’t keep the user from ‘learning by exploration’
- Bad feedback takes the user’s mind off what they’re doing and makes them think about the system - always a bad thing
Learning by Exploration

- Users like to learn this way
- Let them - don’t be too fussy about what they can do, or the order they can do it in.
- Provide feedback when
  - They are about to fall off a cliff .. there aren’t many cliffs:
    - Deleting the corporate database
    - Killing the system/document/process they are working on
  - They ask for help:
    - Context sensitive ([F1]): help is good feedback if well done.
    - Bad help systems won’t be used.

Use of Colour

- Some colours cause problems eg red
  - possible male colour blindness, but more general
  - red seems dim, as very few cones in retina sensitive to red
  - 3D effect of red on green or green on red for everyone
  - red seems fuzzy around edges

Colour Aesthetics

- Colour perception is personal
- What looks good to you won’t necessarily look good to many others
  - What looks garish and ‘yuckie’ to you, may look wonderful to others
  - Drab Is Best
  - Grit your teeth, curse your high school teachers, and build drab systems with no pictures (except at the start) and colours like the Microsoft defaults.

Who’s Boss Around Here?

- In traditional computer programs
  - The program walks the user through a series of steps
  - The user has to perform these steps correctly and in the required order
  - The program (and programmer) knows in advance what the user is doing at any time.
  - The program and programmer are always in control (and there’s always a known point, called the control point, moving through the program)
  - So traditional programmers speak of the “flow of control” through the program.

There’s No Point of Control

- Control is passed to the user
- Within reason they can:
  - Do what they want in the order they want
  - Enter partial and incomplete data
  - Stop in the middle of a task
  - Leave blank and incorrect data on the screen for awhile
- The computer program is passive
- The user is active
- *The user becomes the boss*

The User-Program Dialog

- First, the user owns the screen, and enters anything they want in any order.
- The user passes control to the program, usually by clicking a button or selecting a menu option.
- The program does its checking and tells the user about any problems.
- The program never interrupts the user while they own the screen.
- No checking takes place until the user passes control to the program.
The Trade-off

- This approach makes programming harder (and programmers insecure!)
- It makes program use much easier.
- Given that the user is always the boss, lots of tasks are easier:
  - We know never to beep in righteous indignation
  - We know that error messages must be polite and respectful
  - We know never to use those ‘gizmos’ that edit input on the fly
  - We know we must use tools (e.g., drop down list boxes) so the user can’t make an error.
- User Errors are usually Programmer Errors

References

- Acknowledgement to Mr Dan Eaves for Interface Design lecture

References

- Don Norman’s The Design of Everyday Things (Doubleday Currency 1986)
  - Very useful and general discussions of the problems of designing interfaces applied to doors, windows, water taps and computer systems. A wonderful book.
- Alan Cooper’s About Face: The Essentials of User Interface Design (Programmer’s Press, 1995)
  - Practical advice combined with criticism of a number of standard aspects of interface design. Amusing, exciting, useful and frustrating.
  - The rules for designing consistent interfaces. Specific to Windows 95, so of no use at Monash but useful at home.