Metaphors

- **Metaphors** are used to draw an analogy between the way the dialog works and some object the user is familiar with e.g.
  - Originally a “trash can” on Macintosh
  - Note the difference in the metaphor!
  - Implies the ability to take work home
  - A common storage & classification device

Some Metaphors Don’t Work

Transferring mechanical models to computers can restrict an application where these restrictions are not necessary.

Paper calendars are limited in the size of paper available and show one month at a time.

Computerised calendars are often presented one month at a time. Is this restriction necessary? What opportunities are lost?

Choose Your Metaphors Carefully

- A metaphor should
  - make it easier for the user to understand the function and purpose of the software
  - **NOT** limit the range of useful features offered
  - enable the user to predict ALL the functions available
  - **NOT** behave differently on computer from real life

Application Posture

- **The posture of an application** is the way it presents itself to the user
  - **Sovereign**
  - **Transient**
  - **Daemonic**
  - **Parasitic**

Choose Your Metaphors Carefully

- **Sovereign (royal) posture**
  - only application on screen
  - monopolises user’s attention for long periods
  - usually running continuously
  - usually large with lots of related features, functions
  - e.g. word processor, spreadsheet
Topic: Interface Design Supplementary Slides

Application Posture

- Transient posture
  - single function application
  - called when needed - appears, performs job and leaves,
  - temporary
  - e.g. scanner software

Application Posture

- Daemonic posture
  - works quietly and invisibly in the background
  - possibly doing vital tasks
  - often no need for human intervention
  - talks with the user when they must be manually adjusted for changing circumstances
  - e.g. printer driver

Application Posture

- Parasitic posture
  - continually present like a sovereign
  - performs a supporting role
  - small and superimposed on another application
  - often report progress
  - e.g. Windows clock, eyeballs to track mouse cursor, paper clip

Design for Sovereign Posture

- Sovereign users are experienced users.
  - while each user spends some time as a novice, this is only a short time relative to the total amount of time spent using the software.
  - design for optimum use by experienced users!
  - sovereign users are in flow - don't interrupt!
  - OK to take up video real estate (multiple toolbars etc)
  - Use muted colours and textures (users will be looking at this for a long time!)

Design for Transient Posture

- Buttons etc can be smaller, more tightly packed because users will be familiar with them
- Window should be maximised, avoid resizing to move from one document to another
- Use the full screen and all its possibilities
- Tune the application to avoid delays

Design for Sovereign Posture

- Avoid taking more space than necessary - support rather than replace the sovereign application
- Use bright colours to differentiate this application from background sovereign
- ‘Gizmos’ can be larger and bolder than in a sovereign application
- Build instructions into the interface - user doesn’t use this often enough to remember
- All information and facilities should be available on the surface - single window
**Design for Transient Posture**
- Avoid alternative views and dialog boxes
- Use simple controls - push buttons
- Avoid controls on window edges
- Provide a simple keyboard as well as mouse interface
- Window should be less than full size, must be moveable, may be reshapable

**Design for Daemonic Posture**
- Avoid taking more space than necessary - avoid full-screen control panels
- Mostly should be invisible
  - this includes status reports since user often not aware of the existence of this application
- Must be able to be reached
  - use a Windows control panel or
  - create a transient program that runs a launchable application to configure the daemon
- See design tips for transient applications

**Design for Parasitic Posture**
- Should be simple and often bold in reporting information
- Avoid taking more space than necessary - get out of the way of the sovereign application when necessary
- Are not the focus of the user’s attention
- Use small features because on screen for long periods and user will learn where it is
- Design to support sovereign but not interfere

**Grammar - Gizmos**

<table>
<thead>
<tr>
<th>Gizmo</th>
<th>Data Entry Mode</th>
<th>Validation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Text box:</td>
<td>free text</td>
<td>no-auto checks</td>
</tr>
<tr>
<td>List box:</td>
<td>select one item from a list</td>
<td>limited choice</td>
</tr>
<tr>
<td>Combo box:</td>
<td>select an item or enter text</td>
<td>selects nearest match</td>
</tr>
<tr>
<td>Option button:</td>
<td>choose one option only</td>
<td>limited choice</td>
</tr>
<tr>
<td>Check box:</td>
<td>choose one or more options</td>
<td>limited choice</td>
</tr>
<tr>
<td>Toggle button:</td>
<td>toggle choice on or off</td>
<td>limited choice</td>
</tr>
<tr>
<td>Command button:</td>
<td>perform a preset function</td>
<td></td>
</tr>
</tbody>
</table>

**Text Boxes for Display**
- A text box is used to display data. The form and text box are BOUND to a column in the data table.
- Normally you set the box so that it is obvious this is OUTPUT. You can do this by making the box the same colour as the background and removing the box outline.
- This field will be skipped in the Tab order. If the user selects the field they can copy the data but not change it (locked).

<table>
<thead>
<tr>
<th>Properties</th>
</tr>
</thead>
<tbody>
<tr>
<td>Control Source</td>
</tr>
<tr>
<td>Back Color</td>
</tr>
<tr>
<td>Border Style</td>
</tr>
<tr>
<td>Tab Stop</td>
</tr>
<tr>
<td>Locked</td>
</tr>
</tbody>
</table>

**Text Boxes for Input**
- A text box can be used to input many types of data. The text box will be UNBOUND for input only, or BOUND to a column if you want to display and alter values.
- Normally you set the box so that it is obvious this is INPUT. You can do this by making the box a different colour from the background and having a clear box outline (these are the default settings).
- N.B it is ILEGAL to use tax file numbers for other purposes!

<table>
<thead>
<tr>
<th>Properties</th>
</tr>
</thead>
<tbody>
<tr>
<td>Control Source</td>
</tr>
<tr>
<td>Format</td>
</tr>
<tr>
<td>Decimal places</td>
</tr>
<tr>
<td>Input Mask</td>
</tr>
<tr>
<td>Status Bar Text</td>
</tr>
</tbody>
</table>

This field has formatting prompts as well as automatic checks for numeric data.

version: Jan 97  H. Smith
A list box can be used to restrict input of data to values currently saved. The list box is BOUND to the relevant column in the data table in which the values are stored. You can arrange the size of the box to display as many values at once as you like. If more values are available a scroll bar will appear.

You can arrange the size of the box to display as many values at once as you like. If more values are available a scroll bar will appear.

List boxes like these prevent the user entering invalid data but do require the use of a mouse.

A combo box displays the list of current values but also allows the user input new values or to select current values from the keyboard. The combo box may be BOUND to a column in the data table from which the values will be displayed. OR the possible values may be specified in the property sheet.

List boxes like these prevent the user entering invalid data but do require the use of a mouse.

Control Source Name
Row Source Type Table/Query
Row Source Select [Name] From [gizmos];

Properties
Control Source Name gizmos
Row Source Type Value List
Row Source gizmos; widgets; doodads

Option buttons, check boxes and toggle buttons can be used to provide a choice from a range of alternatives. Usually option buttons are used to select only one choice while check boxes are used to select as many as apply.

Toggle buttons may be labelled with icons. Typically these controls will be placed in a group so that the range of choices will be clear.

Option buttons, check boxes and toggle buttons can be used to provide a choice from a range of alternatives. Usually option buttons are used to select only one choice while check boxes are used to select as many as apply.

Toggle buttons may be labelled with icons. Typically these controls will be placed in a group so that the range of choices will be clear.

Command buttons are used to carry out program steps such as saving a record, accessing another file, etc. As a result, there is often scope for using icons in place of, or in addition to, text.

The event procedure may be generated using a wizard or may be coded as a macro or in Visual Basic.

Command buttons are used to carry out program steps such as saving a record, accessing another file, etc. As a result, there is often scope for using icons in place of, or in addition to, text.

The event procedure may be generated using a wizard or may be coded as a macro or in Visual Basic.

In human speech, closure points enable the speaker to take a breath and the listener to respond. In writing, closure points are given by ends of paragraphs and chapters.

In human-computer dialog, closure points allow the user to take a short mental break. Where delays in processing occur, these cause least interruption at closure points.

In human speech, closure points enable the speaker to take a breath and the listener to respond. In writing, closure points are given by ends of paragraphs and chapters.

In human-computer dialog, closure points allow the user to take a short mental break. Where delays in processing occur, these cause least interruption at closure points.

In human-computer dialog the following situations are examples of possible closure points:

- end of entering data into a gizmo
- end of a composite data item, such as customer address
- end of a sub-transaction, such as one item being ordered
- end of all transactions for this customer
- end of the day (or conversation period)
Closure Points and Data Validation

- Closure points are obvious stages at which to check for errors in the data presented so far.
- So...
  - use closure points as logical places to allow the user to check and correct the data entered so far
  - adjust the level of data validation to the level of closure point
  - remember - error messages interrupt the flow of the dialog - attempt to minimise this interruption

Error Messages

- Error messages may place the user in a different mode.
- In this example, the user is required to change from one mode (the mode of the main process) to respond to the error (error handling mode).
- Modeless feedback is preferable.

Approaches to Errors

- There are several ways of handling errors:
- Restrict the user's opportunities for making errors:
  - get users to select values rather than type them in (can use typing to select values though)
  - don't accept invalid formats (go back to start of field and insist on correct format) - take care with this!

Approaches to Errors

- Tell the user what is acceptable and validate the input
  - give patterns for data entry fields
  - show rules, examples next to the field
- Accept everything the user enters and deal with it somehow (Cooper’s preference)
  - save whatever the user enters
  - offer alternatives

Flow

- Flow - a state of concentration; a “condition of deep, nearly meditative involvement” (DeMarco & Lister).
- Techniques for inducing and maintaining flow
  - follow mental models
  - direct, don’t discuss
  - keep tools close at hand
  - give modeless feedback
  - don’t interrupt

References

Ch 12, 25-27