Tema 7. The interface

Projecte de Programació (PROP)
Index

User interface:

Components of a program that allow the dialogue between a user and a computer.

• Why studying the interface?
• Quality of an interface
• The user
• General principles for interface design
• Types of dialogue. Application of the principles

Pictures from http://www.usability.uk.com/
Why studying the interface?

Importance:

• Gate to access the functionality
• Bad design restrains the use of the functionality
• Up to 70% of the developing effort in interactive applications is devoted to the interface
Why studying the interface?

Difficulties:

• Most volatile part of a program
• Technical, psychological, aesthetic and social factors
• Evolves with technology:
  • Technical, psychological, aesthetic and social factors
  • Most volatile part of a program
• Evolves with the typology of user
• Evolves with peripherals, communication and process speed

• More and more complex:
  • Multimedia, 3D, voice, tactile interfaces (haptics), emotive computing, ...
Quality of an interface

What we ask from it:

• Access to the functionality
• Reliability
• Habitability, usability, ergonomics
Quality of an interface

External quality criteria:
- time to learn to use it
- time to solve user tasks (specially the most frequent ones)
- retention factor
- percentage of errors made by the users
- subjective satisfaction of the user
Quality of an interface

Internal factors:

• Independence dialogue/application logics. It makes easier:
  - Maintenance, reuse, portability
  - Use of specialized tools to aid design

  Made easier by OO technology

  Advantages of dividing the problem into sub-problems
The user

Typology:

- Frequent
- Occasional

Beginner
Average
Expert

Essential:

- Take into account the goals of the user
- Adapt to his/her knowledge, aptitudes and attitudes
- The user can use his/her previous knowledge to learn to use the program

With special needs
The user

Most common mistakes:
• Take more into account technology than the user
• Design for the incorrect user
• Try and change how the user works
The user

Mental or conceptual models of the user

• What does he/she wants? What does he/she knows?
• What concepts does he/she use? How does he/she reason?
• What language does he/she speak?

“The designer analyzes the content to be taught, the task to be performed or the information to be displayed, and defines its structure and functionality. This structure in the designer’s mind is called a conceptual model” – D. Norman [1]
The user

Metaphors:

• Desktop
  -> A “directory” becomes a “folder”

• House with rooms

• Gallery / museum

• City
Design principles

Some classics:


“Make sure that (1) the user can figure out what to do, and (2) the user can tell what is going on” [1]
Design principles

1. There is a conceptual model which is clear and suitable for the user
2. The system state is always visible
3. The tasks are performed easily, with flexibility and efficiency
4. The user short-term memory and attention are not overloaded
5. Diversity of users is taken into account
6. The interface is consistent
7. There are clear messages and informative feedback
8. There is help and documentation
9. There is error prevention
Design principles

1. There is a conceptual model which is clear and suitable for the user
   • The designer has developed a conceptual model appropriate for the user
   • In an ideal world, manuals are written first of all. The designer just follows the manual afterwards
   • The user can use his/her previous knowledge to learn to use the program
   • The user can acquire the rest of the necessary knowledge from the program
Design principles

1. There is a conceptual model which is clear and suitable for the user
   • The user can use his/her previous knowledge in order to learn to use the program
   • The design must not prevent the actions from being consistent with the user knowledge
Design principles

2. The system state is always visible
   • The user can see his/her situation in the system
   • The user can see the state of the conceptual model
   • The user can see the available options; the ones not applicable are switched off
Design principles

2. The system state is always visible
   • Compatibility of data entry with data display
   • There are obvious correspondences between
     - Intentions and actions
     - Actions and effects
Design principles

3. The tasks are performed easily, with flexibility and efficiency
   • Tasks should be simple
   • Technology should make the tasks simple
   • Do not overload the program with functionalities
   • The user has control and freedom (reversibility, ...)

---

DILBERT by Scott Adams

YOUR USER REQUIREMENTS INCLUDE FOUR HUNDRED FEATURES.

DO YOU REALIZE THAT NO HUMAN WOULD BE ABLE TO USE A PRODUCT WITH THAT LEVEL OF COMPLEXITY?

GOOD POINT. I'D BETTER ADD "EASY TO USE" TO THE LIST.
3. The tasks are performed easily, with flexibility and efficiency

- Most frequent tasks should be easily accessible
- Minimal input actions to achieve the desired effect
- Avoid repetitive tasks - alias, macros
Design principles

4. The user short-term memory and attention are not overloaded
   • Humans can remember 7 (+/-2) chunks of information short-term
   • Again: situation inside the system and system states are visible
   • If icons are used, their meaning must be visible
   • Use, not abuse, of display features to attract attention (font, colour, underline, blinking...)
   • Do not overload the interface, present only the relevant information for the current context
Design principles

4. The user short-term memory and attention are not overloaded
   - Example: *Hidden settings*
Design principles

5. Diversity of users is taken into account

• The user language is used
• Experienced users want to have control; novice users want to be guided
• Different levels of information
• Different levels of help
• Different interfaces
Design principles

5. Diversity of users is taken into account

- Different access methods
  - Menu → Files → Save
  - Ctrl-S
  - F2 key

- Different interfaces:
  - Command line
  - Menus and forms
  - Wizards

- Abbreviated methods for frequent users:
  Alias, macros, scripts
Design principles

6. Consistency

• of actions: similar things are done in a similar way
• with the conceptual model / metaphor
• of terminology
• in the screen layout
• of presentation: fonts, colours, sizes and situation of the windows, indentation, capitalization...
• with other applications: applicable standards are followed (O.S.'s, company's, package's...)

http://www.hcibib.org/hci-sites/GUIDELINES.html
Design principles

7. There are clear messages and informative feedback

Messages:
- Non-ambiguous informative messages, in the language of the user, relevant to the context

Feedback: the user can tell what’s happening
- Actions start, progress, and finish
- Something happens with each user action (pushed button, sound, colour change, …)
- Progress bar
- Report performed / not performed action
Design principles

7. There are clear messages and informative feedback

- Messages correspond to what the user knows, not to what the designer knows

“Creating hash table to index quickly…”

“Activating option @&*$%&, the program will work much better in most of the equipments. On the other hand, activating it might give problems later on. Do you want to activate the option @&*%& (Yes/No)?”
Design principles

9. There is error prevention
   • Errors are avoided, or they are easily managed
   • Design to make difficult to make errors
     - Check entries
     - Erroneous or not appropriate for the context options are switched off
   • Make error correction simple
     - Offer simple, constructive and specific instructions to recover from errors
     - Guide to repair only the faulty part. Erroneous actions should leave the system state unchanged
   • Ex: a form
     - Detect errors field by field, if possible
     - Detect all the errors, when pressing Accept
     - If there is one erroneous field, you do not have to input all the data again
     - Or if segmented form, but informing the user that 3 forms are coming
Design principles

9. There is error prevention

- Easy to undo reversible actions
- Difficult to do irreversible actions (confirmation)
- But design assuming that any error that can be made, will be made
<table>
<thead>
<tr>
<th>Types of dialogue</th>
<th>Sequential, initiative from the system:</th>
<th>Asynchronous, initiative from the user:</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>menus</td>
<td>direct manipulation</td>
</tr>
<tr>
<td></td>
<td>forms</td>
<td>events</td>
</tr>
<tr>
<td></td>
<td>formal language commands</td>
<td></td>
</tr>
<tr>
<td></td>
<td>natural language commands</td>
<td></td>
</tr>
</tbody>
</table>
Types of dialogue: Command line

Commands + parameters

Attention to:
- Mnemonic terms
- Consistent terminology
- Compulsory and optional parameters
- Default values, admissible ranges
- Security / irreversibility
Types of dialogue: Command line

Advantages:
• Experienced users: control and flexibility
• There is no ambiguity nor confusion
• Fast for complex tasks
• Allows creation of user-defined macros, alias, scripts…

Disadvantages:
• Memorization and training
• Inadequate for non-expert users
• Poor error handling
• Help can be poor: -help, man, …
Types of dialogue: Menus

Attention to:
• Put together close functions with clear criteria
  - similar operations
  - perform on similar objects
• Consistency:
  - terminology in the labels
  - same order in the options
• Switch of non-applicable options
• No intersections and exhaustive
• Depth of the menu tree
Types of dialogue: Menus

Advantages:
• few memorization from the user
• uniform access to the options
• structured decision making (taxonomic structure)
• possible without keyboard (e.g., touch screen)
• making errors is more difficult

Disadvantages:
• may slow frequent users
• does not allow to introduce parameters
• presents danger of many menus
Types of dialogue: Forms

Presenting information. Attention to:

• put together according to semantic criteria
• present the information in a coherent and intelligible way
• labelling consistent with con terminology, abbreviations
• consider quantity of information provided
• think about the way to present the information
Types of dialogue: Forms

Data-entry transactions. Attention to:

- Logical sequence of the fields to introduce
- Logical movement of cursor, tab, mouse
- Graphical distinction for compulsory and optional fields
- Minimize number of input actions:
  - Confirmation
  - Valid form (clues of valid values)
  - Default values
- Field-level help
Types of dialogue: Forms

Errors. Attention to:
- Correction at field or character level
- Relevant and positive error messages
- Clues about how to recover from error
- Minimize number of correction rounds
Types of dialogue: Direct manipulation

- Fast and incremental actions that provoke an immediately visible effect
- Examples: Desktop (action = drag&drop), computer-assisted design tools (add element to the design), games
- Visual, always valid representation of objects and actions (wysiwyg)
- Attention to:
  - consistency with metaphor
  - reversibility
Types of dialogue: Direct manipulation

Advantages:
- simple syntax and close to the metaphor
- fewer user errors
- easy learning - easy retention
- encourages exploration

Disadvantages:
- more resources (memory, resolution, speed)
- May be hard to program without tools (dynamic management of the domain in the screen...)
- Trend toward complexity
Types of dialogue: GUIs

- Graphical User Interface
- User action = event
- Asynchronous
- All types of elements: menus, forms, commands, shortcuts and function keys, direct manipulation (icons, drag&drop), contextual help
- Switching off options, it can be adapted to all user levels

Just one interface for all the users
Final comments

• Remember:

“Make sure that (1) the user can figure out what to do, and (2) the user can tell what is going on” – D. Norman

• Automatic tools to build interfaces
• (Semi-)automatic tools to analyze usability
• People with special needs: Accessibility