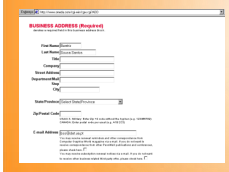
Other Interaction/Dialog Styles



Paulo Dias, Beatriz Sousa Santos

deti departamento de electrónica telecomunicações e informática

universidade de aveiro

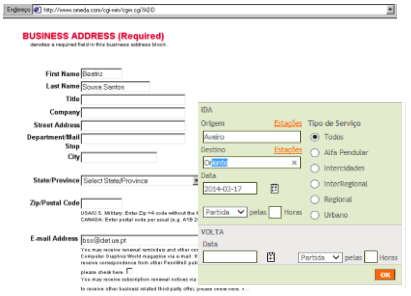
1

Interaction styles – Possible classification

- Menus
- Direct manipulation
- Fill-in-forms
- Dialog boxes
- Function keys
- Command languages
- Natural languages
- 3D interfaces
- …
- Often two or more styles are used simultaneously

2

Fill in forms



3

Fill in forms

- Fill in forms are particularly useful for routine, clerical work or for tasks that require much data entry
- The concept already existed long ago
- They were first used as the only style in a UI
- Currently they are often used with other styles

4

Main advantages and disadvantages

- Advantages (potential)
  - Self-explanatory
  - Recognition instead of recall
  - Allow many different inputs (unlike menus)
  - Give context and guide the user
  - New functionality is visible (unlike command languages)
- Disadvantages
  - Imply knowledge of valid inputs
  - Error prone
  - Not very flexible
  - Consume screen space

5

User profile to whom fill-in-forms are adequate

- Knowledge and experience:
  - Moderate or high typing skill
  - High or moderate task experience
  - Moderate or low application experience
  - Moderate to high computer literacy
- Task characteristics:
  - Moderate to high frequency of use
  - Low training
  - Highly structured task

6

Fill in form design: relevant aspects

- Organization and layout
- Titles and fields
- Input formats
- Instructions and help
- Navigation
- Error handling

Fill in form design guidelines - Layout

Avoid unfamiliar layouts

Example:

Zip code:
Name:
Country:
Address:
City:

Better:

Name:
Address:
Zip code:
City:
Country:

Fill in form design guidelines – Titles and fields

- Alignment of titles

Not a good solution

```
Name:              ------------------------------------
Title:             ------------------------------------
Rank:              ------------------------------------
Telephone number:  ------------------------------------
```

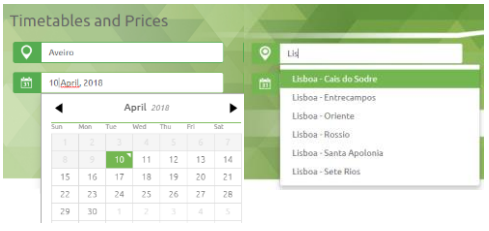
```
Name: ------------------------------------------------
Title: -----------------------------------------------
Rank: ------------------------------------------------
Telephone number: ------------------------------------
```

Better solutions

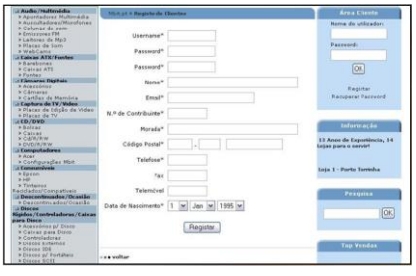
```
              Name: ------------------------------------
             Title: ------------------------------------
              Rank: ------------------------------------
  Telephone number: ------------------------------------
```


Fill in form design guidelines – Titles and fields

Differentiate titles and fields; do not show the cursor over fields



Fill in form design guidelines – Instructions



Often indicated by *

Fill in form design guidelines – Instructions

Input format must be familiar and clear

Better:

Date:_____
(eg. 1/12/2000)

Date:___/__/_____
(eg. 1/ 12 /2000)

Date:_____
(e.g. 01122000)

Time:_____
(eg. 8-15)

Time:___-_____
(e.g. 08-15)

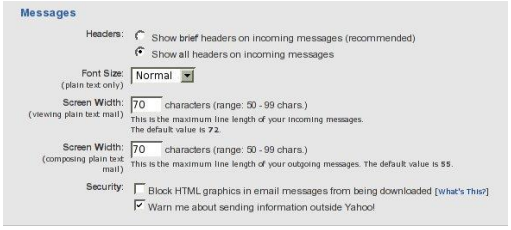
Time:_____
(e.g. 0815)

Card#:_____
(eg. 123456789012)

Card#:___-____-_____
1234-5678-9012)

Fill in form design guidelines – Instructions

Instructions to fill the fields should be clear

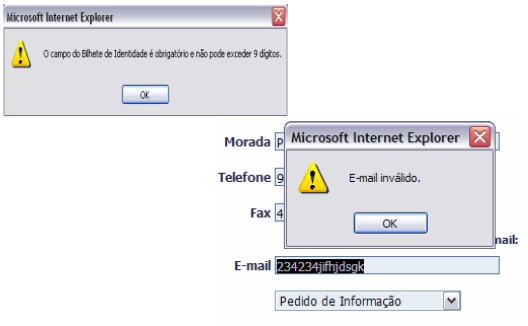


18

18

Fill in form design guidelines – Errors

Examples of clear error messages:

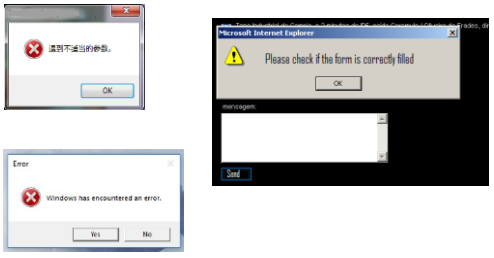


20

20

Fill in form design guidelines – Errors

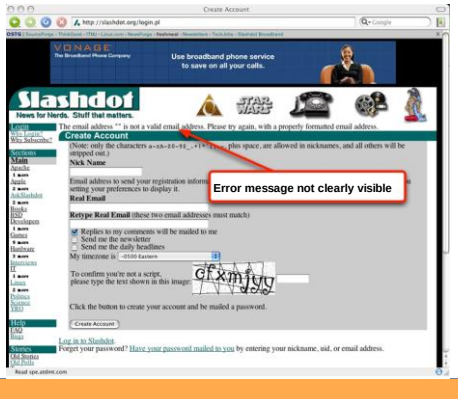
Messages not clear, nor helpful



21

21

Fill in form design guidelines – Errors



22

22

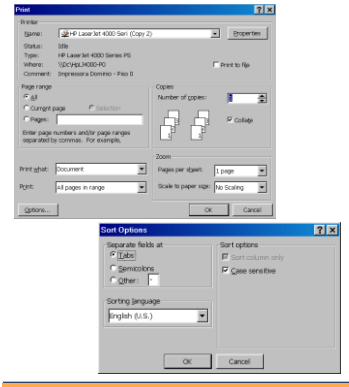
Interaction styles – Possible classification

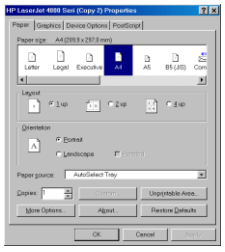
- Menus
- Direct manipulation
- Fill-in-forms
- Dialog boxes
- Function keys
- Command languages
- Natural languages
- 3D interfaces
- …
- Often two or more styles are used simultaneously

23

23

Dialog Boxes





24

24

Dialog Boxes

- Combination of Menus and Fill-in-Form.
- A few directives (intern)
  - Representative title
  - Sequence from *Top-left* to *bottom-right*
  - Consistent layouts
    - Margins, grids, space, lines, boxes
    - Terminology, font, alignment
  - Standardized buttons (OK, Cancel)
- A few directives (extern)
  - Smooth appearing / disappearing
  - Size to avoid occlusion
  - Appearing close to items but without occlusion
  - Easy to close

Interaction styles – Possible classification

- Menus
- Direct manipulation
- Fill-in-forms
- Dialog boxes
- Function Keys
- Command languages
- Natural languages
- 3D interfaces
- …
- Often two or more styles are used simultaneously

Function Keys

- Two types (Hard and Soft Keys):
  - *Hard Keys* – Always invoke the same functionality (as the keys of a calculator and some specific keys of PCs)
  - *Soft Keys* – invoke different functionality according the context of use (as the keys (F1...Fn) and the generic keys of an Automated Telling Machine, e.g. Multibanco)
- PCs have 12 generic Keys (F1 a F12) and a few other specific keys

Keys that invoke specific functionality in PCs and MACs

Function Keys – Hard keys

Hard function keys have abbreviations of default actions printed on/besides them

Specific keyboard

Start menu key

Function Keys – Soft keys

Soft function keys don't have abbreviations of default actions printed on/besides them, they may have "F-number" designations.

Their value can be programmed and may depend on combinations (shift, alt +…)

Function keys (generic)

https://en.wikipedia.org/wiki/Function_key

Main advantages and disadvantages

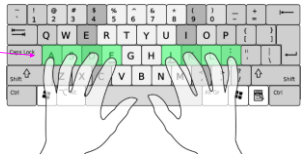
- Advantages (potential)
  - Self-explanatory
  - Recognition instead of recall
  - Easy to use
  - Flexible
  - Require little or no screen real estate
- Disadvantages
  - Limited number of keys
  - Hardware expansions are expensive

User profile to whom function keys are adequate:

- Knowledge and experience:
  - High or moderate task experience
  - Moderate application experience
- Task characteristics:
  - Low to high frequency of use
  - Low training or no training

Function Keys

Often used keys should be near the "home row"



Keys with serious consequences should not be easy to activate

(e.g. ctrl Alt Del)

Interaction styles – Possible classification

- Menus
- Direct manipulation
- Fill-in-forms
- Dialog boxes
- Function Keys
- Command languages
- Natural languages
- 3D interfaces
- …
- Often two or more styles are used simultaneously

Command languages

```
cd /tmp
echo "line 1
line 2
line 4" > tmp1$$
echo "line 2
line 3" > tmp2$$
diff tmp1$$ tmp2$$
rm tmp1$$ tmp2$$
```

```
guru99@VirtualBox:~$ history
    1  cat > sample
    2  cat sample
    3  cat sample ^a
    4  cat sample a
    5  cat sample | grep a
    6  cat sample | grep ^a
    7  useradd home
    8  useradd mycomputer
    9  sudo useradd mycomputer
   10  sudo adduser MyLinux
   11  sudo adduser mylinux
   12  vi scriptsample.sh
```

Command languages shall also be designed as to be as usable as possible

Command Language

- User types in commands in an artificial language
  - all knowledge in the head; low learnability

```
emails_32YSM~
instruções_mex_c_matlab~
java-how-to.txt
java-how-to.txt~
matlab-install.txt~
Notas_implementação_ProjectoPOO~
notas_ros.txt~
```

```
Ubuntu is configured with SSH and VNC servers that ca
n be accessed from the IP:
eth0: No such device

Now enter the screen size you want in pixels (e.g. 80
0x480), followed by [ENTER]:
800x480
Please select which Desktop environment you want to u
se, type the number to select it then press [ENTER]
1 - LXDE
2 - Gnome
Make your Selection:
1
```


Command Language

- Only text?

Command languages

Note that:
Command languages may be used not only through text but also via voice

e.g.
While driving a car to control the media, the phone or navigate

http://support.volvocars.com/uk/Pages/article.aspx?article=a8275b1eb0ed6a0fc0a8015159f7fdd6

Command Languages: Goals when designing

- Precision
- Compactness
- Ease in writing and reading
- Speed in learning
- Simplicity to reduce errors
- Ease of retention over time

Command Languages: Usability questions

- Does the language support necessary functions?
- Is it fast to enter a command?
- Is it easy to recognize what the command might do?
- Is it easy to recall a command?
- Are there few errors when using the language?

Main advantages and disadvantages

- Advantages (potential)
  - Powerful
  - Flexible
  - Efficient
  - Do not take much screen real estate
- Disadvantages
  - Difficult to learn
  - Not self-explainable
  - Error prone
  - Improvements are not visible

User profile to whom Command languages are adequate

- Knowledge and experience:
  - High task experience
  - High application experience
  - High computational literacy
  - High typing skill
- Task characteristics:
  - High usage frequency
  - Formal training

Command Language design

Balance richness and minimalism
(similar to semantic distance in direct manipulation)

Examples :

Rich	**Minimal**
Delete	Delete
Insert	Insert
Replace	
---	---
Copy	Copy
Move	Delete
Rename	
Delete	

(the functionality is the same)

Command Language design

Use a coherent syntaxe

Use a natural and easy to remember action-object grammar

```
VolB!FileA!D$$
FileA!VolB!ER$L!:KO:!*$$
```

Uncoherent syntaxe

```
search filea volb.
open filea volb.
list all lines with "KO".
```

or

```
s filea volb.
o filea volb.
lal "KO".
```


Command Language design

Command abbreviations should be simple and coherent
Easy to remember (not easy to recognize as for function keys)

Name	Abbreviations Poor:	Improved:
Move forward	MovF	MovF
Move backward	Mvb	MovB
Insert	I	Ins
Delete	Dl	Del
Replace	Repl	Rep
Search	Srch	Sea
Delete	X	Del
Send	Sn	Sen
Print	Prt	Pri
Search	Srch	Sea
Send	Sn	Sen
Find	Fi	Fin
Choose	Ch	Cho

Command Language design

- Allow the following interaction features:
  - Defaults
  - Command edition
  - Intelligent interpretation
  - Type-ahead
  - Feedback
  - Help and documentation
  - Make the language "user tailorable"

Example:
"delate": did you mean "delete"? Y or N

Example of a (complex) command with defaults

ls - Linux man page

Name

ls - list directory contents

Synopsis

ls [*OPTION*]... [*FILE*]...

Description

List information about the FILEs (the current directory by default). Sort entries alphabetically if none of **-cftuvSUX** nor **--sort**.

Mandatory arguments to long options are mandatory for short options too.

-a, **--all**
do not ignore entries starting with .
-A, **--almost-all**
do not list implied . and ..
--author
with **-l**, print the author of each file
-b, **--escape**
print octal escapes for nongraphic characters

You don't need to use all arguments; there are default values

-d, **--directory**
list directory entries instead of contents, and do not dereference symb
-D, **--dired**
generate output designed for Emacs' dired mode
-f
do not sort, enable **-aU**, disable **-ls --color**
-F, **--classify**
append indicator (one of */=>@|) to entries
--file-type
likewise, except do not append '*'
--format=*WORD*
across **-x**, commas **-m**, horizontal **-x**, long **-l**, single-column **-1**, verbos
--full-time
like **-l --time-style**=*full-iso*
-g
like **-l**, but do not list owner
--group-directories-first
group directories before files.
augment with a **--sort** option, but any
use of **--sort**=*none* (**-U**) disables grouping
-G, **--no-group**
in a long listing, don't print group names
-h, **--human-readable**
with **-l**, print sizes in human readable format (e.g., 1K 234M 2G)
--si
likewise, but use powers of 1000 not 1024
-H, **--dereference-command-line**
follow symbolic links listed on the command line

Relevant issues in Command Language design

- Lexicon
- Semantics
- Syntax
- Interaction

Usability problems with Unix

- Lexicon (word set): Abbreviations not suggestive of function
  - terse (short)
  - inconsistent
  - jargon (specific terminology/context)
- Syntax: Complex syntax
  - Action modifier(s) object(s)
- Semantics : Under utilization of commands
  - Unnecessary complexity to support many functions leads to complexity of most frequent
  - Hard to map commands to tasks
- Interaction: Lack of feedback

Example: Unix

- 400+ possible commands
- 20 commands (5%) account for 70% of usage
- 14 commands (3.5%) account for 50% of usage

Interaction styles – Possible classification

- Menus
- Direct manipulation
- Fill-in-forms
- Dialog boxes
- Function Keys
- Command languages
- Natural languages
- 3D interfaces
- …
- Often two or more styles are used simultaneously

Natural language

- Communication between humans and computers through natural language involves:
  - recognition
  - generation
- Natural languages as dialog style are not full blown natural languages, they are restricted natural languages
- Natural languages (as dialog style) differ in "habitability" (how easy and natural is it for users)

Note: It still is not possible to maintain a conversation with a computer as in 2001 A Space Odyssey

Note:
natural language as a dialog style and voice interaction are different things!

Natural language habiltability

- Habitability (mismatch between the users' expectations and the capabilities of a natural language) is related to the language domains:
  - Conceptual - the set of objects and actions provided by the language
  - Functional – what may be directly expressed by the language
  - Syntactic – syntactic forms that may be understood
  - Lexical - the variety of words that may be understood
- Conceptual model limitations are not very disturbing; however, limitations in any other domain make the language less habitable

Natural language: example

Example:

- Imagine an information system of a University including a data base with information about employees that may be accessed using a natural language:

  - Conceptual domain: information about employees

  - The question "What is the salary of the University Restaurant manager?" may be out of the functional domain and imply two questions due to functional domain limitations:
    - "Who is the University Restaurant manager?" (answer: Mr. XXX)
    - "What is the salary of Mr. XXX?"

  - "What is the salary of Mr. XXX?" may not be recognized (due to syntactic domain limitations) even if the information is stored in the DB

  - "What are the wages of Mr. XXX?" may not be recognized due to lexical domain limitations if wages does not belong to the language

Main advantages and disadvantages

- Advantages (potential)
  - Powerful
  - Flexible
  - Efficient
- Disadvantages
  - Assume problem domain knowledge
  - Imply clarification dialogs
  - Imply typing skills (if written)
  - Improvements are not visible
  - May create unrealistic expectations, foster irresponsible behaviours and generate negative reactions
  - Difficult and expensive to implement

User profile to whom Natural languages are adequate

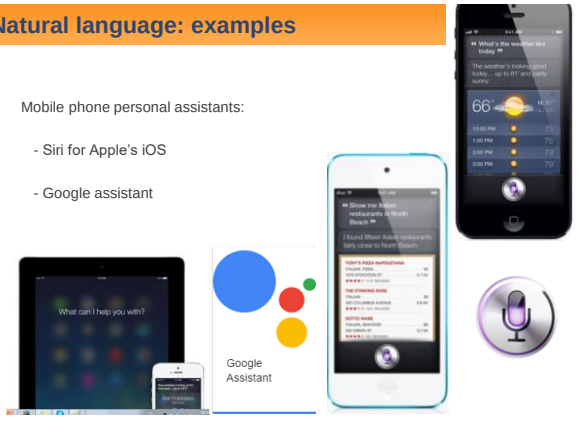
- Knowledge and experience:
  - High task experience
  - High application experience
  - High computational literacy
  - High typing skill
- Task characteristics:
  - High usage frequency
  - Formal training

Natural language: examples

Mobile phone personal assistants:

- Siri for Apple's iOS

- Google assistant



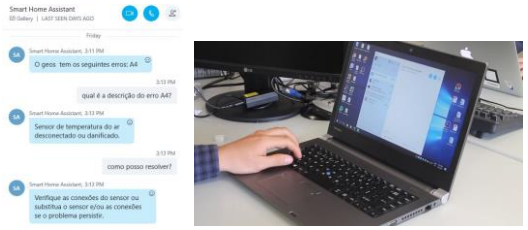
Natural language: voice

- Amazon Alexa

https://www.amazon.com/Amazon-Echo-Bluetooth-Speaker-with-WiFi-Alexa/dp/B00X4WHP5E

Natural language example

- User speaks in natural language, and system responds in the same way



Natural language: wizard of Oz

- A prototype that only works by having someone behind-the-scenes "pulling the levers and flipping the switches" (named after the classical film)
- A user interacts with an interface without knowing that the responses are given by someone

The "wizard" was a "man behind-the-scene"

http://www.usabilityfirst.com/glossary/

Main advantages and disadvantages of interaction styles

Interaction style	Main advantages	Main disadvantages	Application examples
Direct manipulation	Fast and intuitive interaction Easy to learn	May be hard to implement Only suitable where there is a visual metaphor for tasks and objects	Video games CAD systems
Menu selection	Avoids user error Little typing required	Slow for experienced users Can become complex if many menu options	Most general-purpose systems
Form fill-in	Simple data entry Easy to learn Checkable	Takes up a lot of screen space Causes problems where user options do not match the form fields	Stock control Personal loan processing
Command language	Powerful and flexible	Hard to learn Poor error management	Operating systems Command and control systems
Natural language	Accessible to casual users Easily extended	Requires more typing Natural language understanding systems are unreliable	Information retrieval systems

(Sommerville, 2010, chap.29)

Interaction styles – Possible classification

- Menus
- Fill-in-forms
- Direct manipulation
- Dialog boxes
- Function Keys
- Command languages
- Natural laguages
- 3D interfaces
- …
- Often two or more styles are used simultaneously

3D User Interfaces

- User interfaces involving 3D interaction (i.e. interaction in which the user's tasks are performed directly in a 3D spatial context).
- Are more and more used:

  - Virtual and augmented reality
  - 3D workspaces
  - Data Visualization …

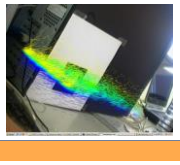
- But have some issues:
  - User disorientation (in the real world we have more information)

Interfaces 3D

- User interfaces involving 3D interaction (i.e. interaction in which the user's tasks are performed directly in a 3D spatial context).
- More and more popular:
  - Virtual and augmented reality
  - 3D workspaces
  - Data Visualization …
- some issues:
  - User disorientation (in the real world we have more information)

Applications of virtual and augmented reality

- Entertainment
- Training and simulation
- Data and Information Visualization
- Project review



Main bibliography

- B. Shneiderman et al., *Designing the User Interface- Strategies for Effective Humaman–Computer Interaction*, 5th ed., Addison Wesley, 2009
- Soegaard, Mads. Interaction Styles, 2010
  http://www.interactiondesign.org/encyclopedia/interaction_styles.html
  https://www.interaction-design.org/literature/book/the-encyclopedia-of-human-computer-interaction-2nd-ed/3d-user-interfaces
- Ian Sommerville, Software Engineering, 9 ed, Addison Wesley , 2010
  https://ifs.host.cs.st-andrews.ac.uk/Books/SE9/WebChapters/PDF/Ch_29%20Interaction_design.pdf