University Center Abdelhafid Boussouf – Mila

HUMAN COMPUTER INTERACTION

Chapter 4
Ergonomic Rules in User Interfaces

2025 - 2026

BASTIEN & SCAPIN'S CRITERIA

THE 8 ERGONOMIC CRITERIA OF BASTIEN & SCAPIN

- In 1997, French researchers Bastien and Scapin synthesized over 900 usability recommendations into a robust model of 8 core ergonomic criteria.

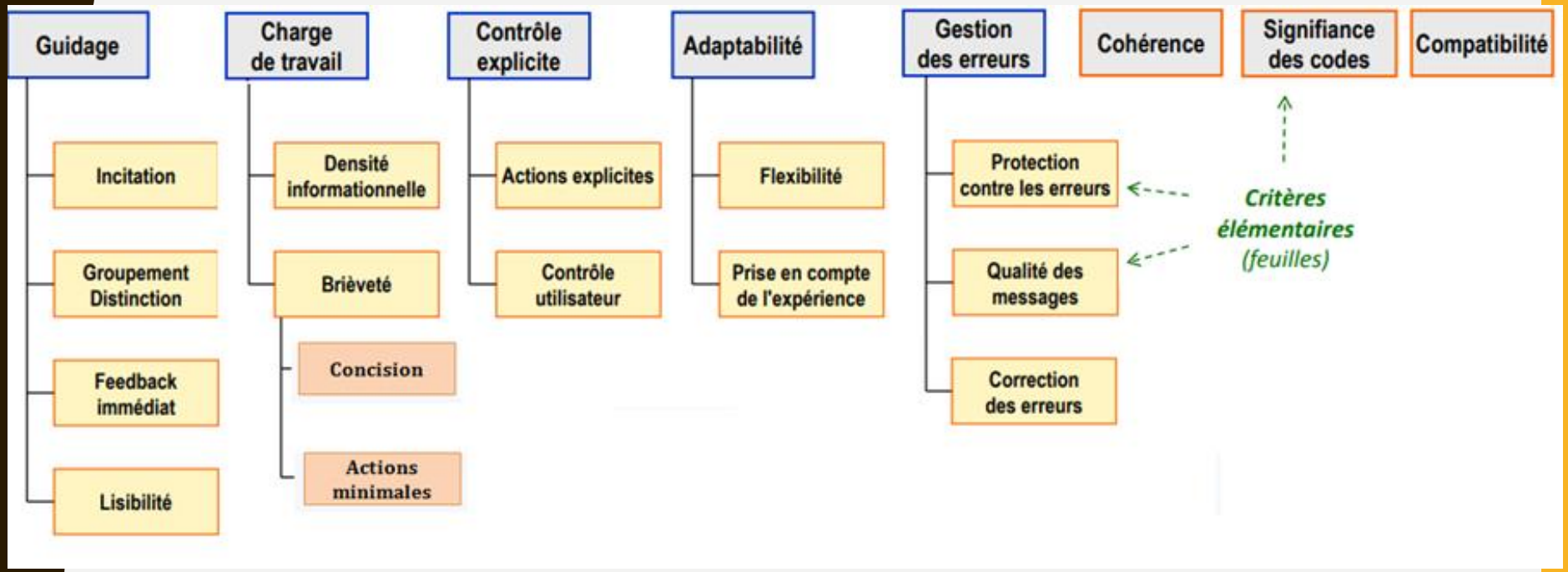
- This framework is not just theoretical; studies have proven it to be more effective at uncovering interface flaws than other established standards.

THE 8 ERGONOMIC CRITERIA OF BASTIEN & SCAPIN

The 8 Main Ergonomic Criteria (Bastien & Scapin, 1997):

1. Guidance (Prompting, Grouping, Immediate Feedback)
2. Workload (Brevity, Information Density)
3. Explicit Control
4. Adaptability (Flexibility, User Experience)
5. Error Management (Protection, Quality of Error Messages, Error Correction)
6. Consistency
7. Significance of Codes
8. Compatibility

THE 8 ERGONOMIC CRITERIA OF BASTIEN & SCAPIN



1. GUIDANCE

- Means helping users understand what they can do and how to do it.
- **Explicit guidance:** messages, help bubbles, dialog boxes
- **Implicit guidance:** greying out inactive options, visual cues

Sub-criteria:

1. **Prompting (Incitation):** Encourage correct actions with hints or format suggestions.
2. **Grouping/Distinction:** Organize elements coherently (by color, shape, position).
3. **Immediate Feedback:** The system must react to user actions (e.g., progress bars, cursor changes).
4. **Readability:** Ensure text, icons, and layout are easily readable and well-organized.

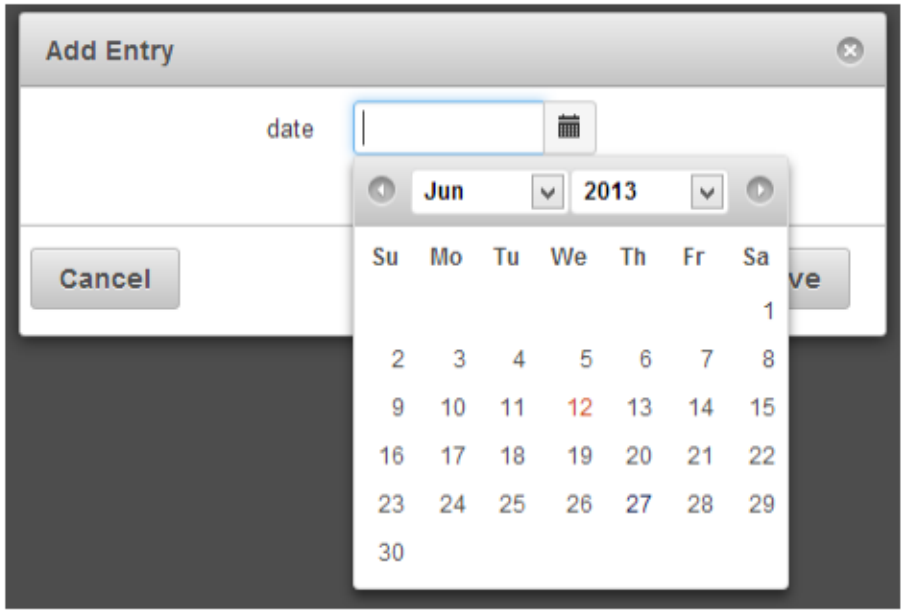
1.1 PROMPTING

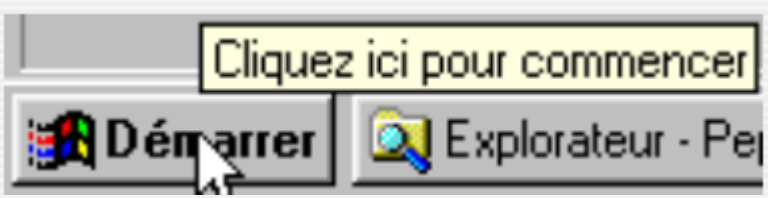
- Encourage users to perform specific actions by providing them with cues. For example, guide data entry by indicating the appropriate format and acceptable values: Date (dd/mm/yy) __ / __ / ___.

Recommendations:

- Gray out unavailable functions (menu options, buttons, etc.)
- Provide a list of expected inputs (dropdown lists, codes to use, etc.)
- Show the required data entry format (dates, dimensions, etc.)
- Change the cursor shape (mouse pointer) to provide hints about the operation to be performed
- Clearly indicate mandatory fields (* or other indicator)
- Clearly show how to move forward and backward
- Display tooltips on non-trivial elements (toolbar icons, for example)

1.1 PROMPTING

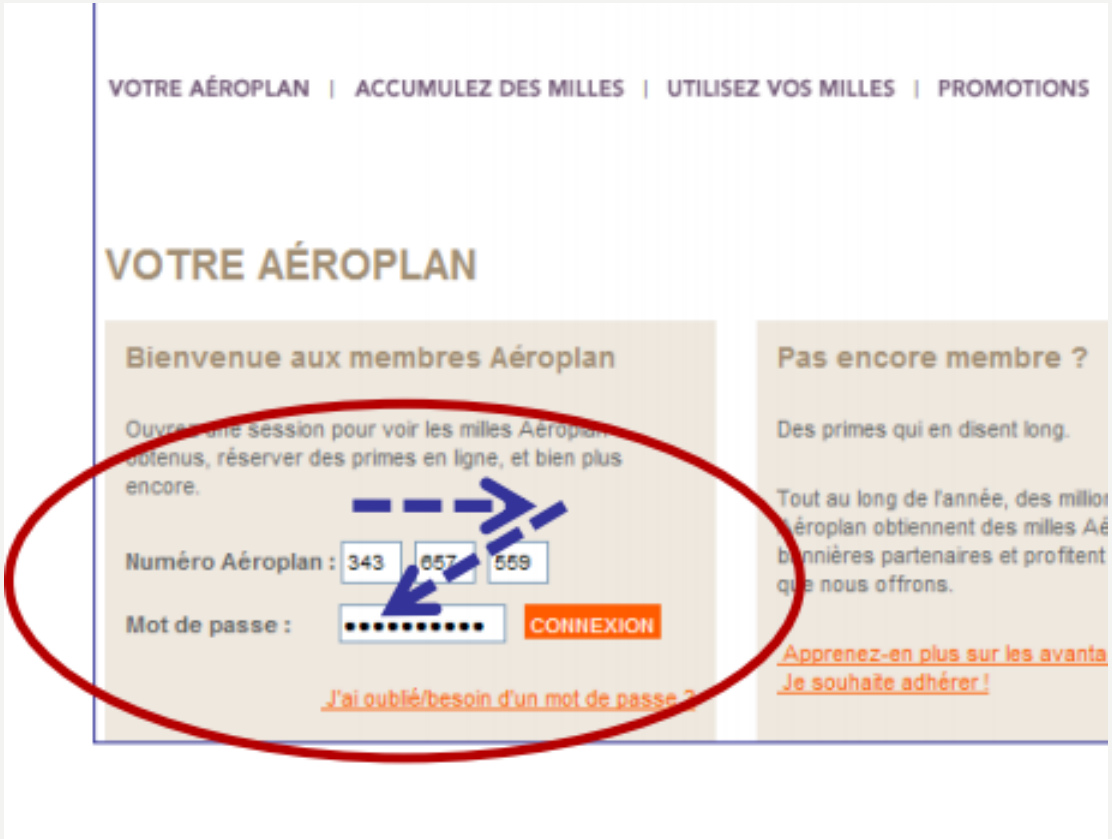




1.1 PROMPTING

- The subscriber number is divided into three blocks of three digits

- After the first block is entered, the cursor automatically moves to the next block



1.1. PROMPTING: COUNTEREXAMPLE

- An example where the prompting is so poor that the interface must explicitly tell users where to click!

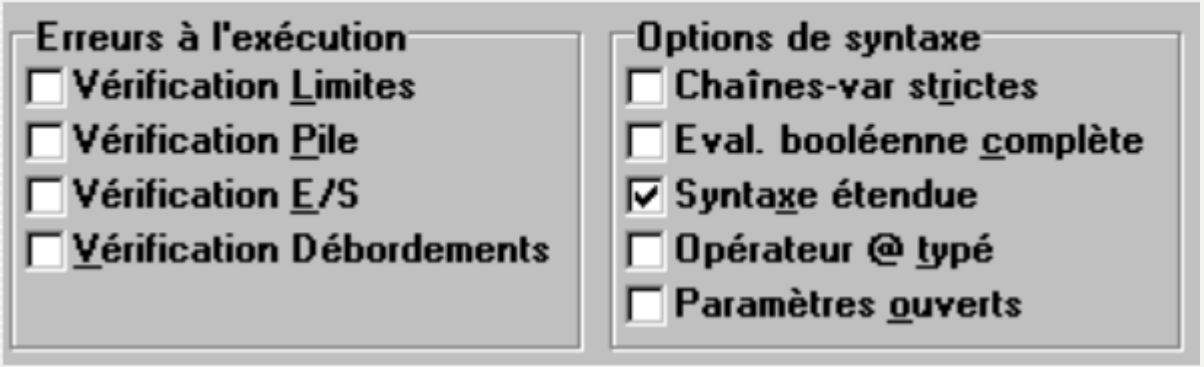
1.2. GROUPING / DISTINCTION BETWEEN ITEMS

- Group different visual elements in a coherent and organized way. The objective is for the visual characteristics of the interface to convey meaning.

- Two graphical attributes are used for this purpose:
- The format/presentation (color, shape, syntax, ...)
- The position in the interface (location, borders, ...)

1.2. GROUPING / DISTINCTION BETWEEN ITEMS

- A faire



- A ne pas faire

1.3. IMMEDIATE FEEDBACK

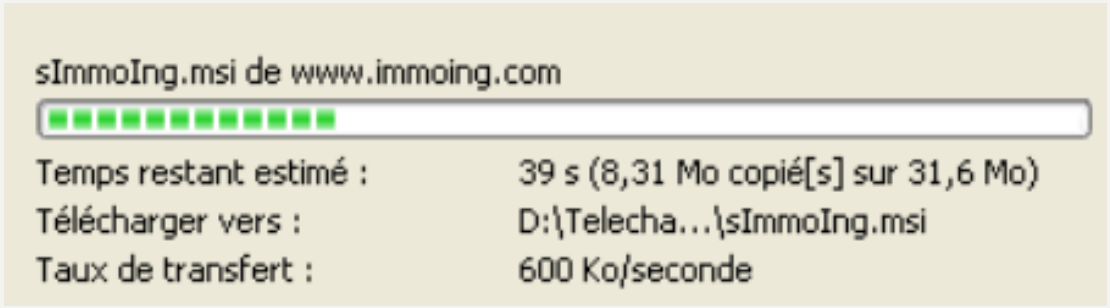
- This involves informing the user about the effectiveness of their actions. In all cases, the computer must respond to the user based on their actions and requests (example: timer).
- The system must react to each of the user's actions, regardless of the processing time (long or not).

Recommendations:

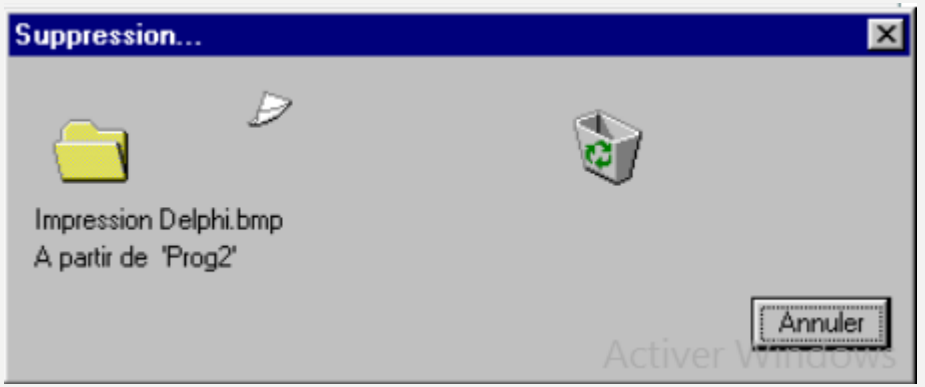
- The software must respond to every user action with a change in the interface presentation (visual, auditory, etc.)
- Indicate the system's operating modes (status)
- Signal long processing times with a wait indicator (hourglass, progress bar, animation, message, etc.)

1.3. IMMEDIATE FEEDBACK

- **To do:**
- Change an object's appearance sélectionné
- Change the cursor for short waits
- Use a progress indicator for longer tasks (>5s)
-



Not to do



1.4. READABILITY

- The readability criterion involves facilitating the perception of textual and graphical information through judicious choices of their properties and layout.
- Particular attention should be paid to:
- Fonts (variety, size, attributes, lowercase/uppercase, ...)
- Colors in general (selection, contrast, purpose, ...)
- The layout of elements (alignment, spacing, page composition, ...)

Recommendations:

- Use a readable font (on screen, avoid small italics, ...)
- Pay attention to spacing (line spacing) and alignment (justification)
- Adjust the size of labels and icons to ensure their interpretation
- Ensure good contrasts (choice of colors for background and foreground)
- Consider the target machine's configuration (resolution, ...)

1.4. READABILITY

- **To Do**

  Ce texte est plus lisible : noir sur fond blanc, caractères sans empattement

- **Not to Do**

  Ce texte est plus difficile à lire : mauvaise association de couleurs, caractères avec empattement

2. WORKLOAD

The workload criterion encompasses all means aimed at reducing the user's perceptual, memory, and physical load. The goal is to minimize both the amount of information the user must process and the number of elementary actions required to complete a given task.

The workload criterion is divided into two sub-criteria:

- Brevity
- Information density

2.1. BREVITY

- Limit the reading, data entry, and steps users must go through
- Additionally, we must facilitate the cognitive processing of information. The size of groups as well as individual elements (words, buttons, links...) should therefore be adapted.

Recommendations:

- Limit the number of options in a menu or dropdown list
- Avoid overly long labels
- Reduce the number of elementary actions needed to achieve a given goal
- Prevent users from having to remember information from one window to another, perform calculations, or enter information that can be deduced by the system.

2.1. BREVITY

- Identify essential vs. useful functions: Avoid feature overload, as excess can hinder usability

2.1.1. CONCISENESS

- Reduce the workload at the perceptual and memory levels (perception and memorization) for individual input or output elements.
- Example:

Exemple : **taille du libellé d'un lien** :

Pour nous contacter, vous pouvez cliquer sur ce lien et alors on vous répondra très vite

versus

contactez-nous

2.1.1. CONCISENESS

- To Do

- Not to Do

2.1.2. MINIMALIST ACTIONS

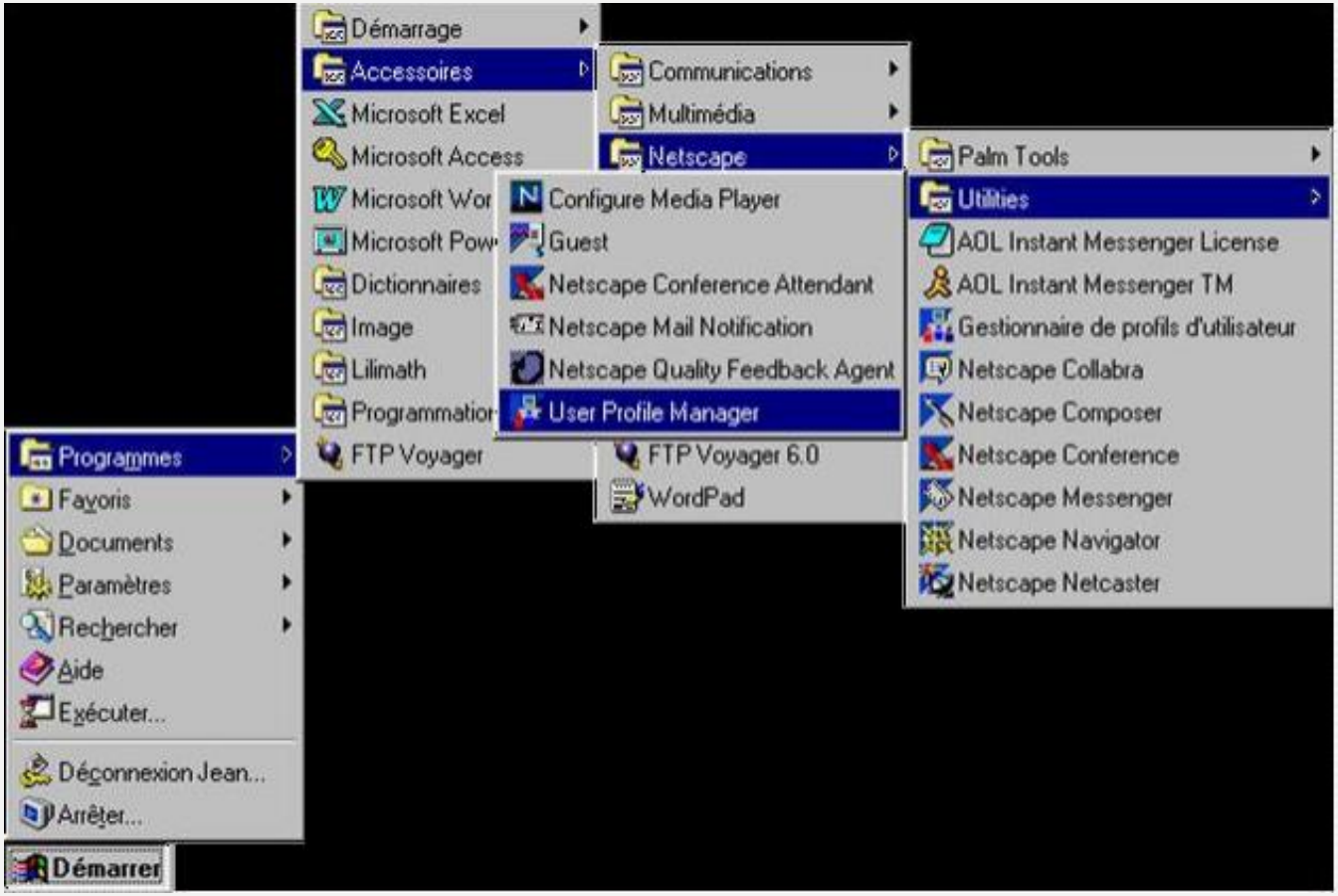
- Limit the number of steps users must go through.
- For example, do not ask users to enter data that can be deduced by the system.

Counter Example:

GUICHET AUTOMATIQUE EN SERVICE
Insérer votre carte
Entrer votre code d'identification : **1234**
Sélectionner une opération : **retrait**
Donner le montant du retrait : **7500 F**
Le montant doit être un multiple de **1000 F**
(La machine éjecte la carte)
GUICHET AUTOMATIQUE EN SERVICE...
(L'utilisateur doit tout recommencer)

2.1.2. MINIMALIST ACTIONS

- **Note to Do**



2.2. INFORMATION DENSITY

- Reduce the workload from a perceptual and memory perspective, for sets of elements rather than individual items (what needs to be perceived and memorized overall by the user).
- For example, limit the information density on the screen by displaying only necessary information.

Recommendations:

- Display only information relevant to performing the task (simple dialog boxes, graphical representations, etc.)
- Avoid overly cluttered screens (break them down if necessary)
- Avoid too many links in text displayed on a web page
- Avoid overly verbose text (simple dialogue, short sentences)
- Prioritize recognition (symbols, icons)

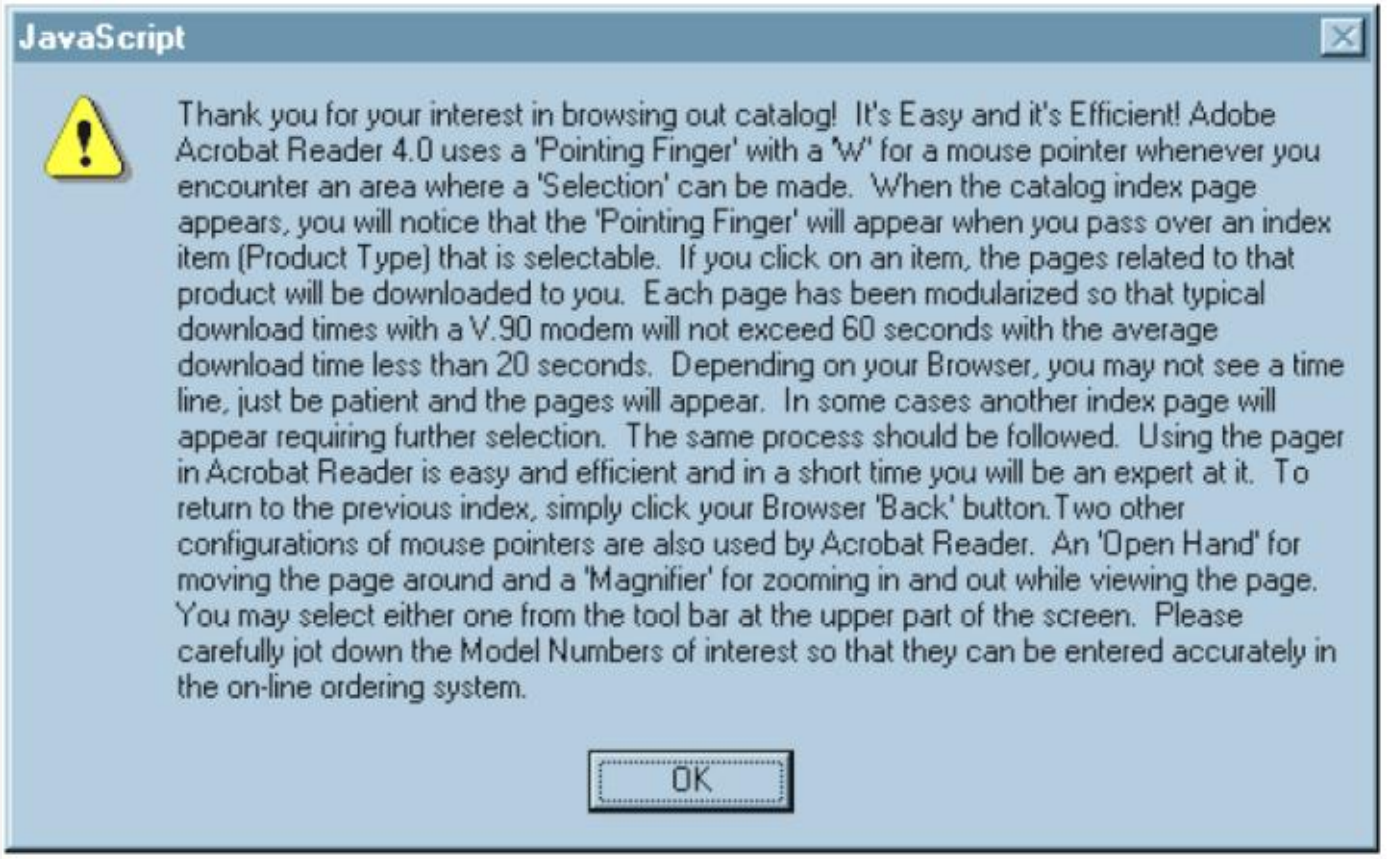
2.2. INFORMATION DENSITY

- Rich content but not **IDEAL**

2.2. INFORMATION DENSITY

A message you don't really feel like reading...

- Avoid overly wordy text (use simple dialogue, short sentences).



3. EXPLICIT CONTROL

- The explicit control criterion concerns aspects related to the degree of control the user has over the processing performed by the interactive system.

- It is broken down into two elementary sub-criteria:
  - Explicit Actions
  - User Control

3.1. EXPLICIT ACTIONS

- Make clear the relationship between how the application works and the users' actions.
- For example, entering commands should end with a clear "finish" indicator ("Enter", "OK"), and editing possibilities should be available beforehand.

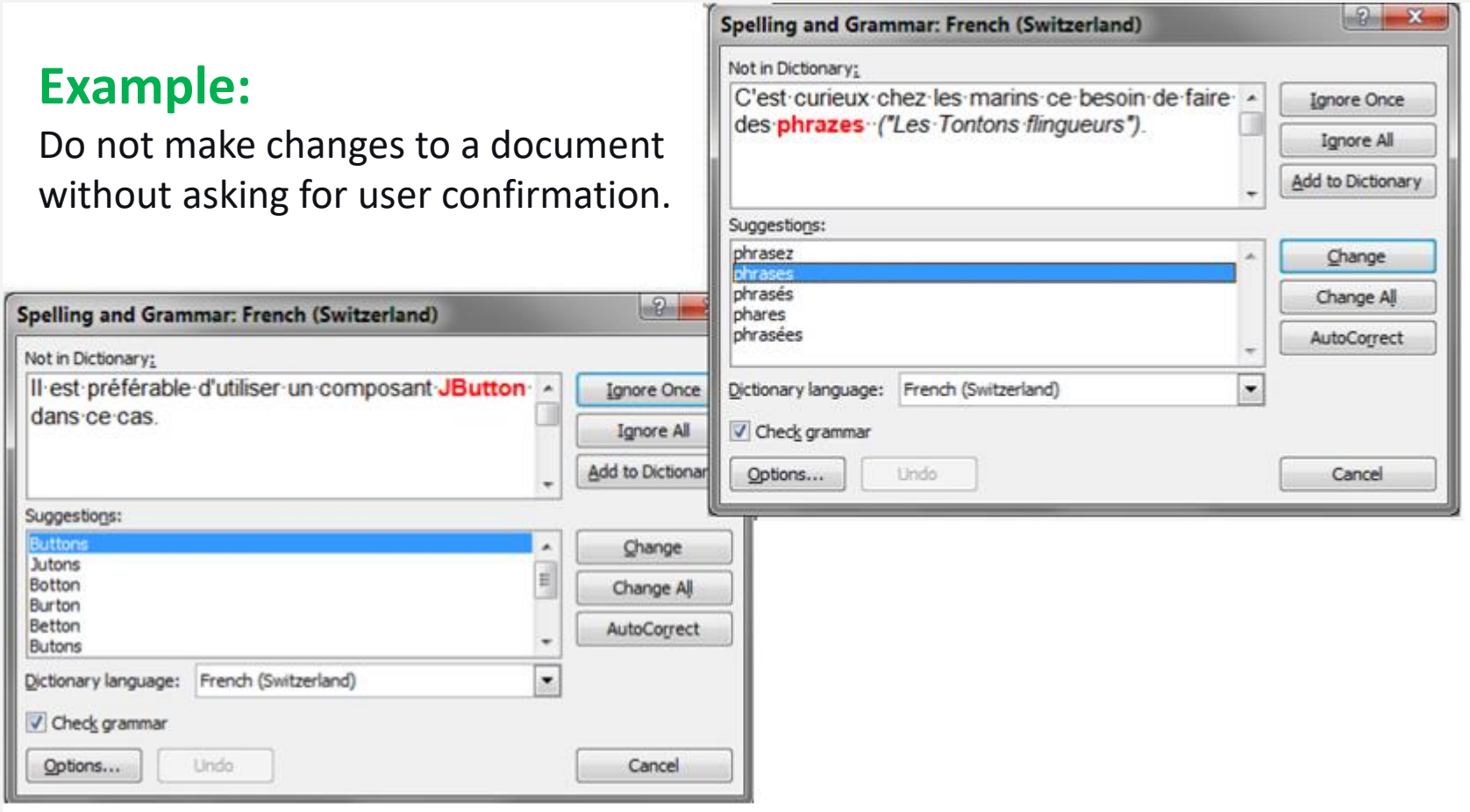
Recommendations:

- Do not trigger operations without the user's explicit consent.
- Trigger the operation immediately after the user's action or, if not, clearly indicate that the operation will be delayed (or that it cannot be performed).

3.1. EXPLICIT ACTIONS

Example:

Do not make changes to a document without asking for user confirmation.



3.2. USER CONTROL

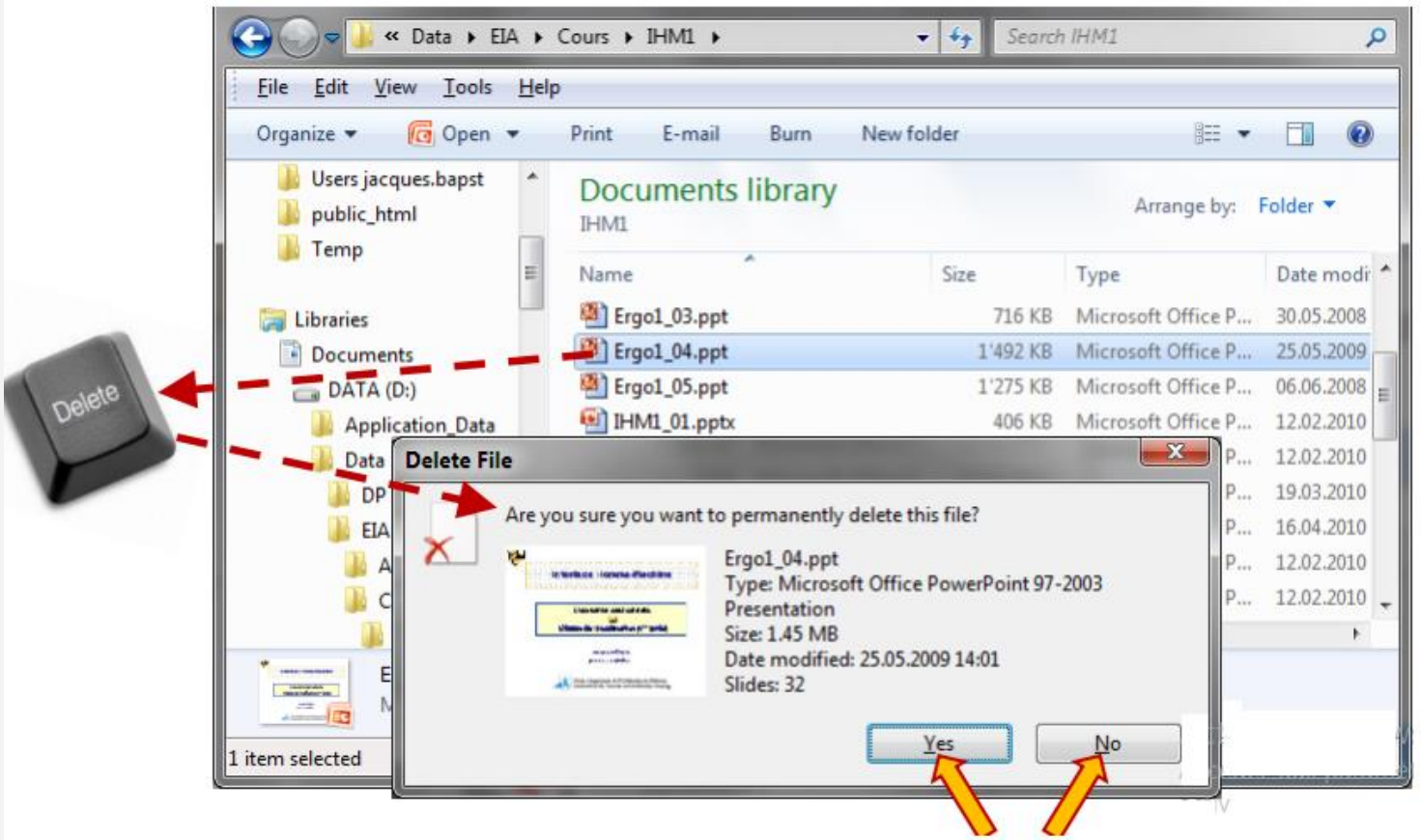
- The user control criterion refers to the fact that the user must always maintain authority over the system and control its operations and their progression (interrupt, resume).
- The user must permanently maintain control over the software.

Recommendations:

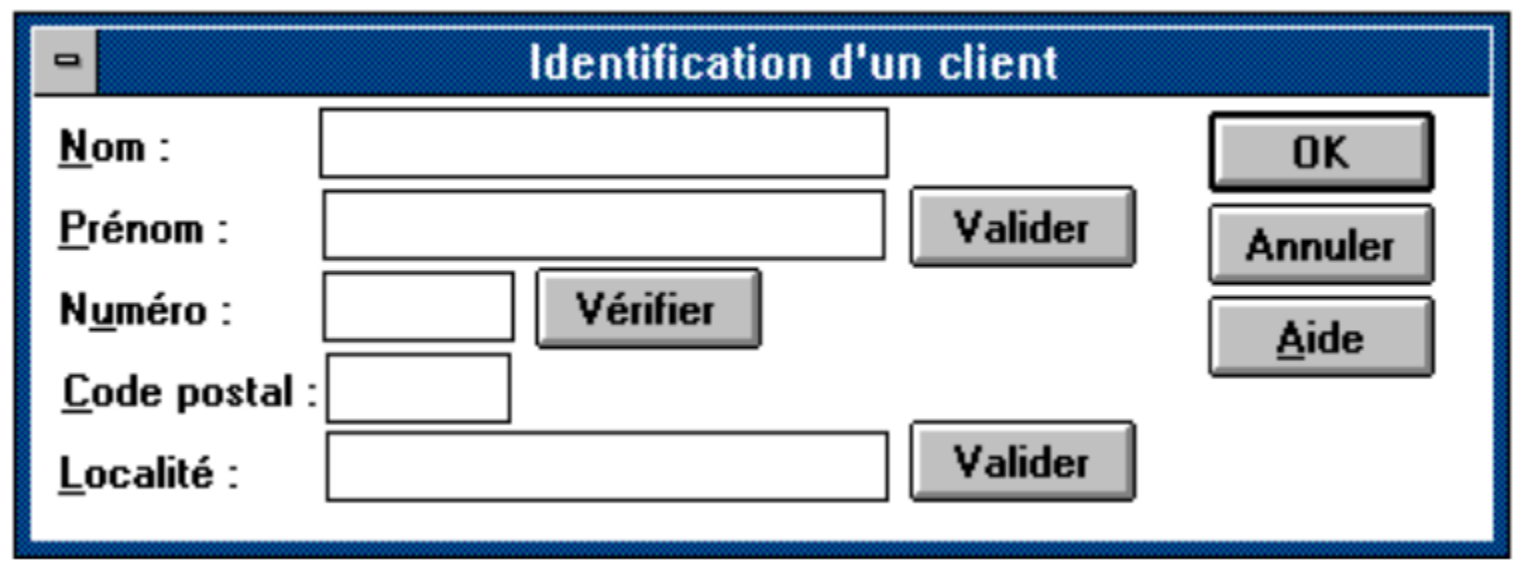
- Provide explicit confirmation for important or difficult-to-reverse commands
- Offer the ability to interrupt long processes
- Allow undo operations (Undo)
- Always permit exiting the current function or even the software itself

3.2. USER CONTROL: COUNTER EXAMPLE

- **Provide the user with explicit confirmation for important or difficult-to-reverse commands.**



3.2. USER CONTROL: COUNTER EXAMPLE



4. ADAPTABILITY

- **A system's adaptability is characterized by its interface's ability to react and adjust according to the context and the needs and preferences of its users.**

The adaptability criterion is divided into two elementary sub-criteria:

- **Flexibility**
- **Consideration of the user's experience**

4.1 FLEXIBILITY

- The flexibility criterion concerns the means provided to users to customize the interface according to their preferences, abilities, habits, context of use, and individual working methods.
- It also aims to provide users with the ability to perform a task or activate a function in different ways, for example:
  - Through a dropdown menu
  - Through a context menu (right-click)
  - Through a toolbar icon
  - Through a keyboard shortcut

4.2. CONSIDERATION OF THE USER'S EXPERIENCE

- The second sub-criterion of flexibility is the consideration of user experience, which involves the methods implemented to adapt to users' varying levels of experience.

Recommendations:

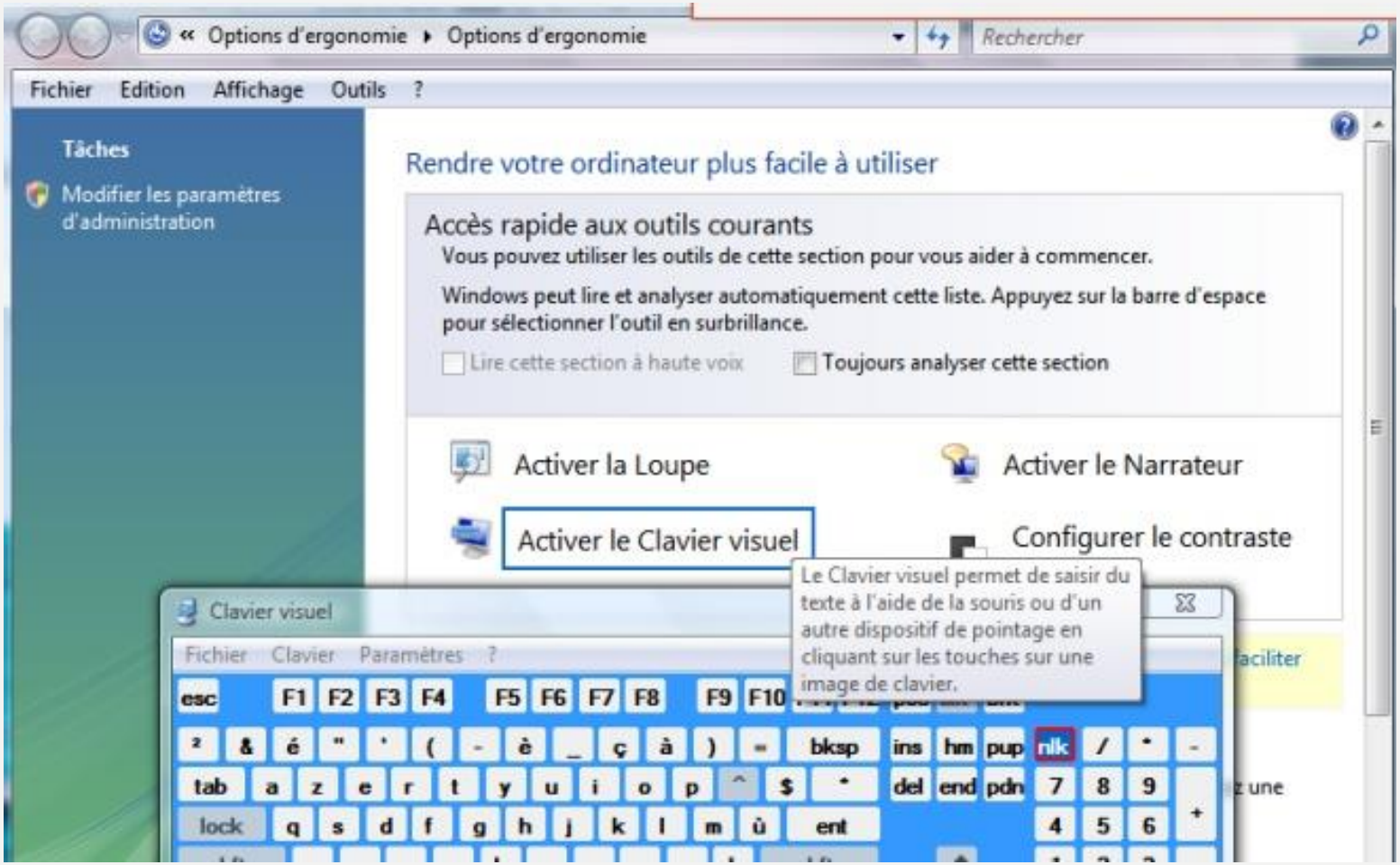
- Allow the user to define their experience level.
- Ask for it or determine it automatically through metrics (e.g., number or rate of errors, response times, etc.).
- Guide the novice user step by step through task completion.
- Implement strong guidance by creating assistants (wizards).
- Provide experienced users with the means to perform their tasks quickly and efficiently (even if it sometimes comes at the expense of guidance).

4.2. CONSIDERATION OF THE USER'S EXPERIENCE

- **Example:**

This interface allows users to search for a keyword using three different modalities.

4.2. CONSIDERATION OF THE USER'S EXPERIENCE



5. ERROR MANAGEMENT

- Methods aimed at both preventing or reducing errors, and correcting them when they occur.
- The primary goal is to minimize disruptions caused by errors (maximize performance, efficiency, productivity).
- Three sub-criteria must be considered:
  Protection against errors (prevent the user from making them)
  Quality of error messages (clearly inform the user)
  Error correction (enable the user to correct them)

General Recommendations:

- Do everything possible to prevent errors (defensive design)
- Initiate a dialogue when user actions could lead to irreversible situations (or undesirable consequences)
- Warn the user as early as possible and guide them toward resolving the problem
- Facilitate system exploration and learning

5.1. ERROR PROTECTION

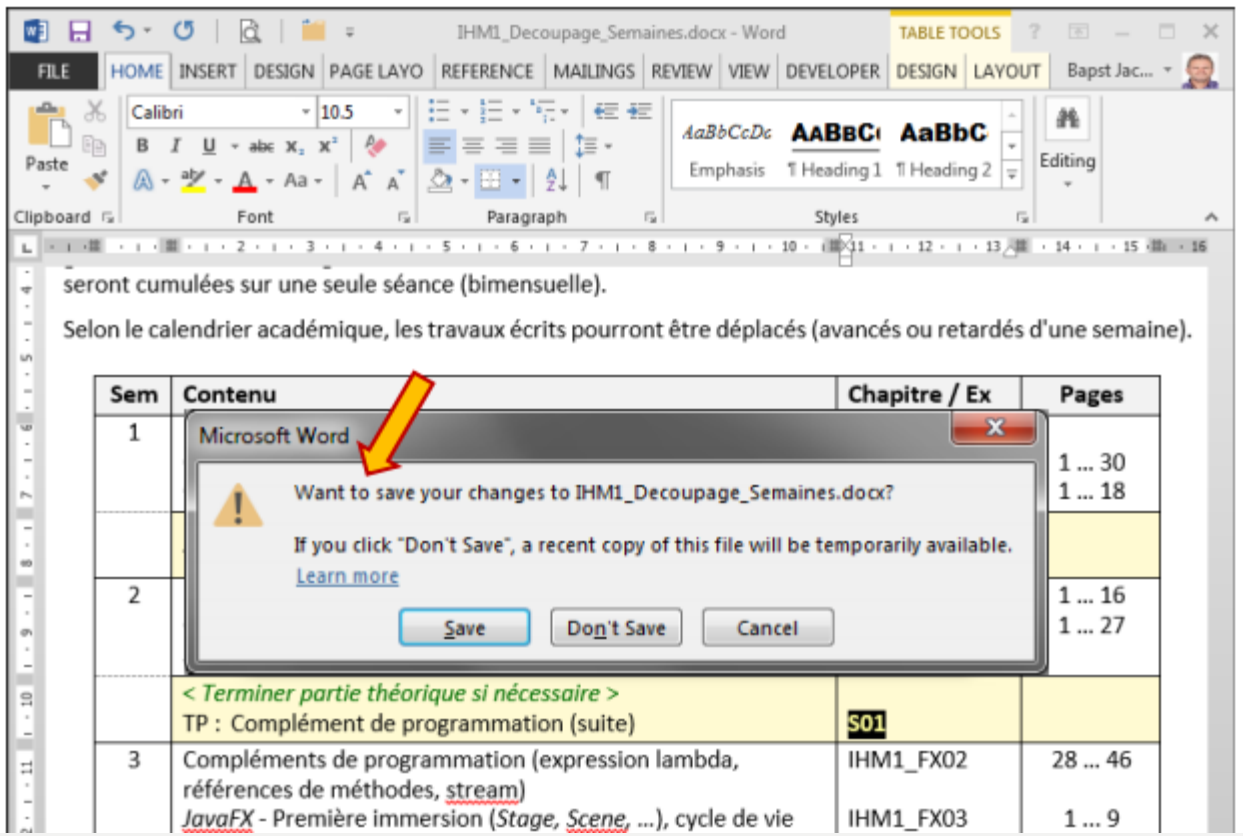
- Implement methods to detect and prevent errors (before validation).
- If an error still occurs, it must be detected as early as possible and the user must be warned with a clear message.
- For example, all possible actions on an interface must be considered, especially accidental keyboard presses, so that unexpected inputs are detected.

Recommendations:

- Highlight unavailable commands (gray out)
- Provide a list of possible values, units, ...
- Detect errors as early as possible and warn the user immediately
- Minimize keyboard input (use selection lists when possible)
- Prevent risks of data loss (ask for confirmation)

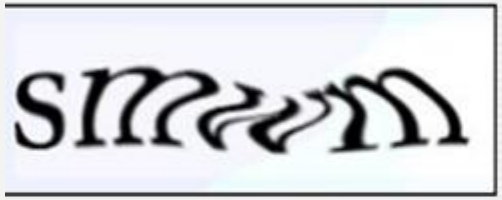
5.1. ERROR PROTECTION

- **Prevent risks of data loss (ask for confirmation)**

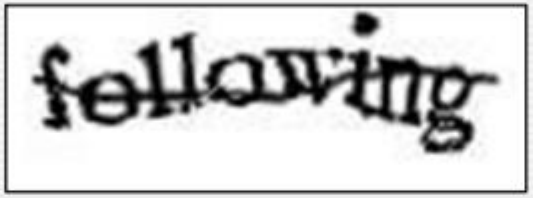


5.1. ERROR PROTECTION

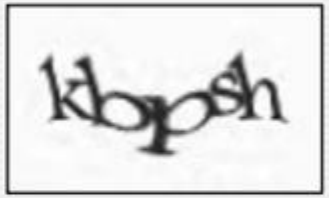
- **To Do :**
  - Captcha / tips : permettant de différencier de manière automatisée un utilisateur humain d'un ordinateur



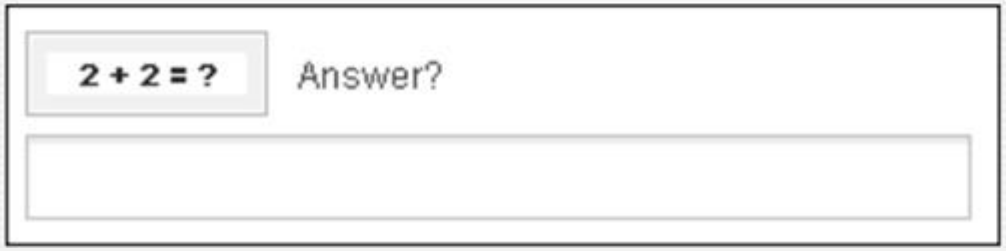
Distorsion des lettres et ajout d'un dégradé de couleur en fond



reCAPTCHA : ajout d'une ligne brisée



Imbrication des lettres les unes dans les autres (CAPTCHA actuel de Yahoo).



5.2. QUALITY OF ERROR MESSAGES

- The quality of error messages criterion concerns the relevance, readability, and accuracy of information provided to users about the nature of errors made and the actions required to correct them.

- To make error correction easy, the error message should indicate the nature of the error, its cause, and how to correct it (quality messages promote system learning).

- The content of messages should aim to downplay errors in the user's eyes (reduces feelings of guilt and stress).

Recommendations:

- Place error messages where the user is expected to look

- Display clear error messages (use the user's language)

- Avoid overly long texts (be brief, use links, references, etc.)

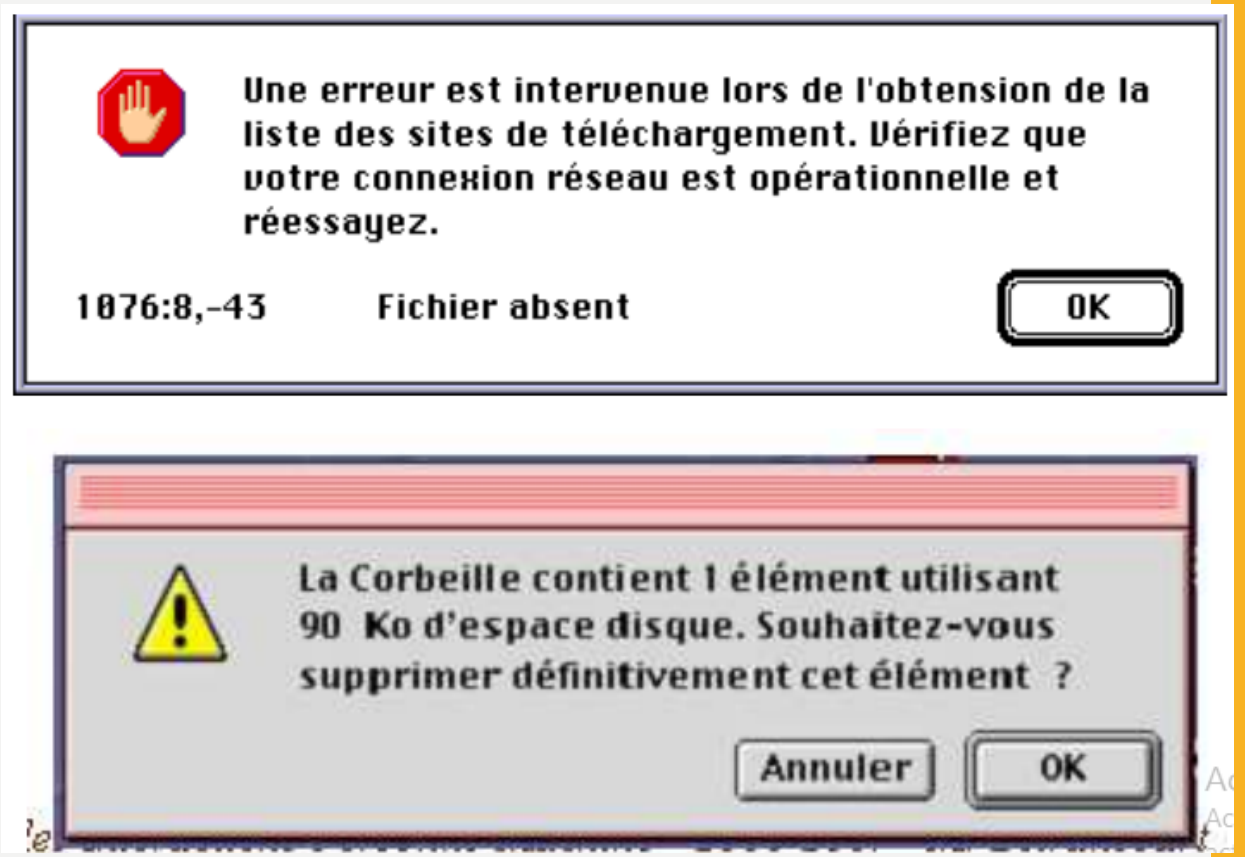
- Avoid reproachful/blaming language

- Ensure messages are self-sufficient when possible. Find the right balance between conciseness and comprehensiveness

5.2. QUALITY OF ERROR MESSAGES

Informative, constructive, understandable, friendly messages showing the cause and the

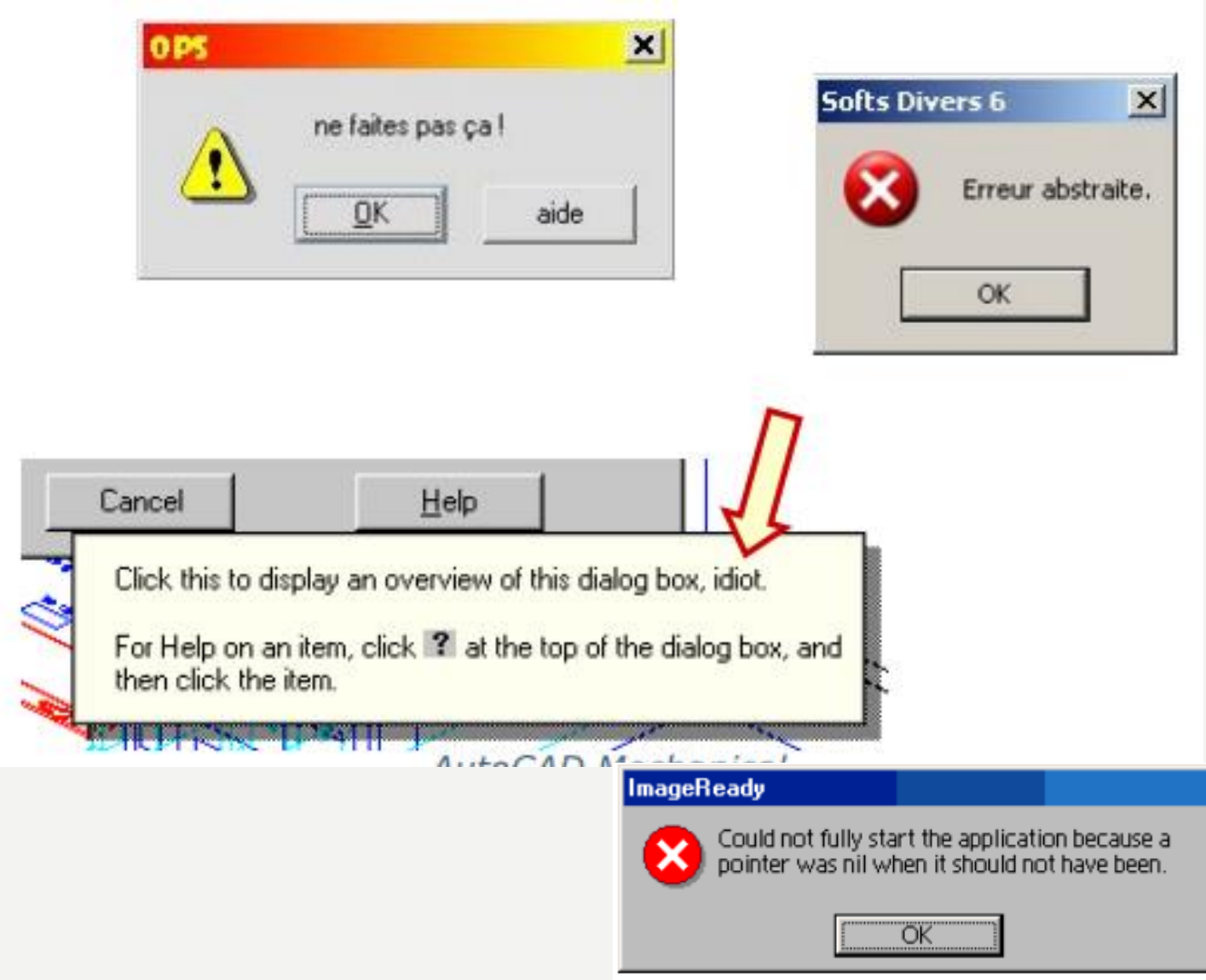
solution



4 Types of Messages:

1. Information
2. Warning
3. Blocking Error
4. Progress Indication

DO NOT DO


OPS
ne faites pas ça !
OK
aide
Softs Divers 6
Erreur abstraite.
OK
Cancel
Help
Click this to display an overview of this dialog box, idiot.
For Help on an item, click ? at the top of the dialog box, and then click the item.
ImageReady
Could not fully start the application because a pointer was nil when it should not have been.
OK

5.3. ERROR CORRECTION

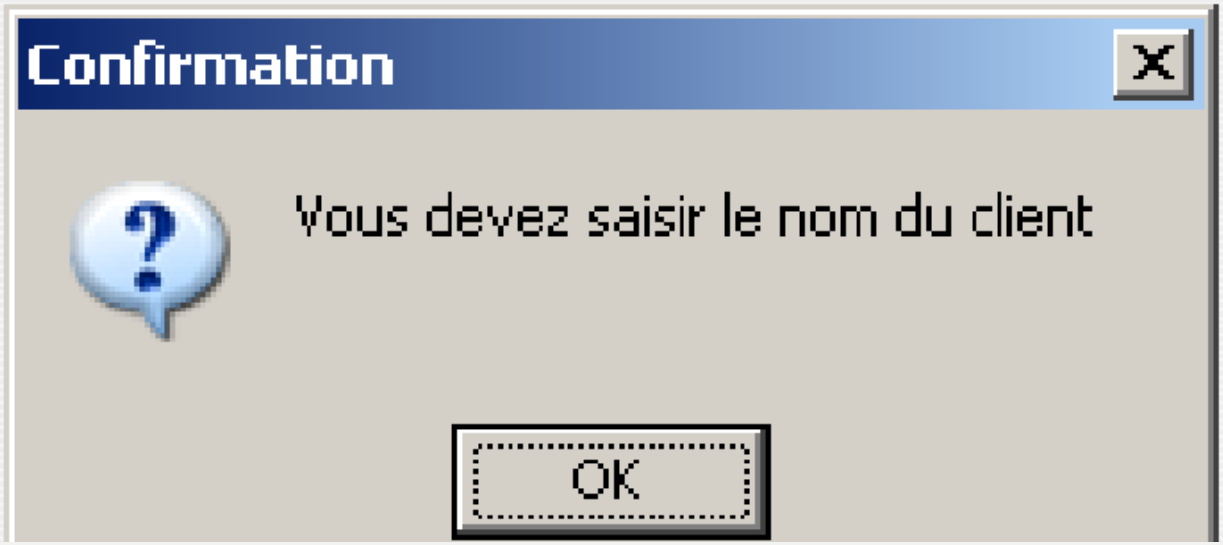
- The error correction criterion includes all means provided to users to enable them to correct their errors.

- Provide users with ways to correct their errors as easily and quickly as possible, for example, by allowing them to correct only the portion of data that is erroneous rather than forcing them to re-enter all information.

- Different tactics depending on the type and severity of the error:
  - Block the user as long as the error persists (serious errors)
  - Allow them to continue after a warning (message, sound signal)
  - Do not respond to the erroneous command (a message is necessary)
  - Automatically correct the error (in rare cases).

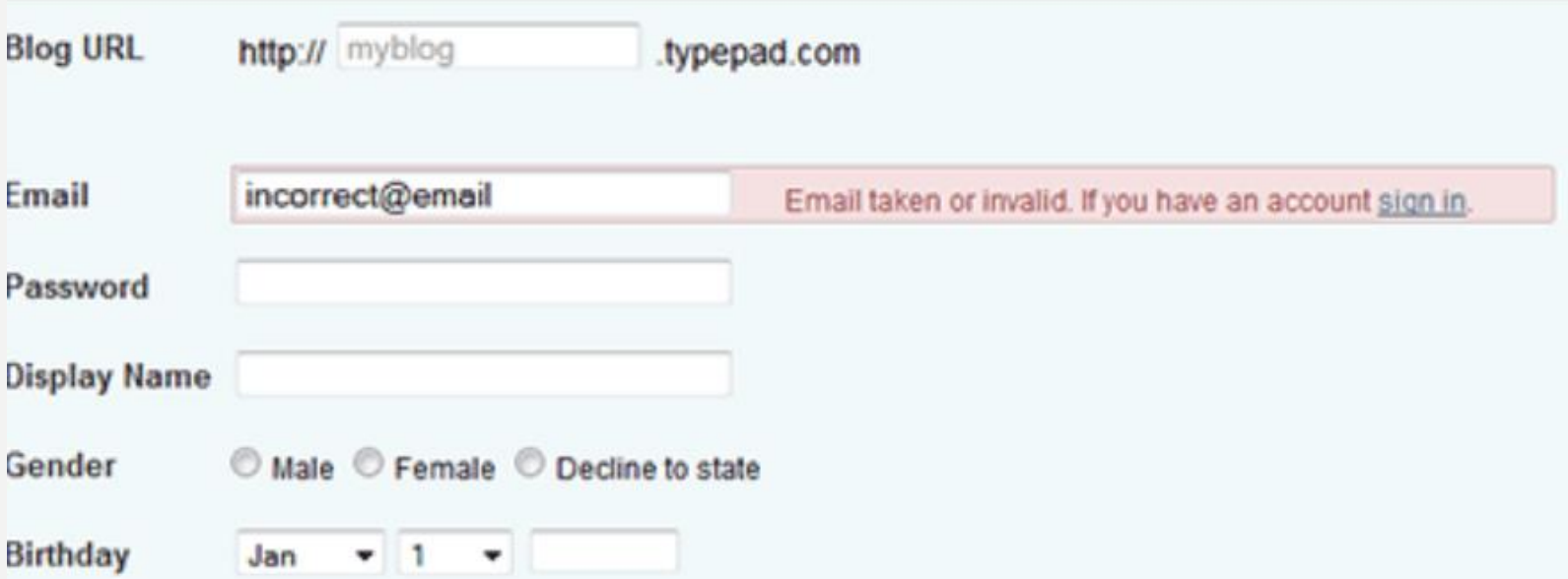
5.3. ERROR CORRECTION

- **To Do**
  - la commande annuler
  - messages explicites



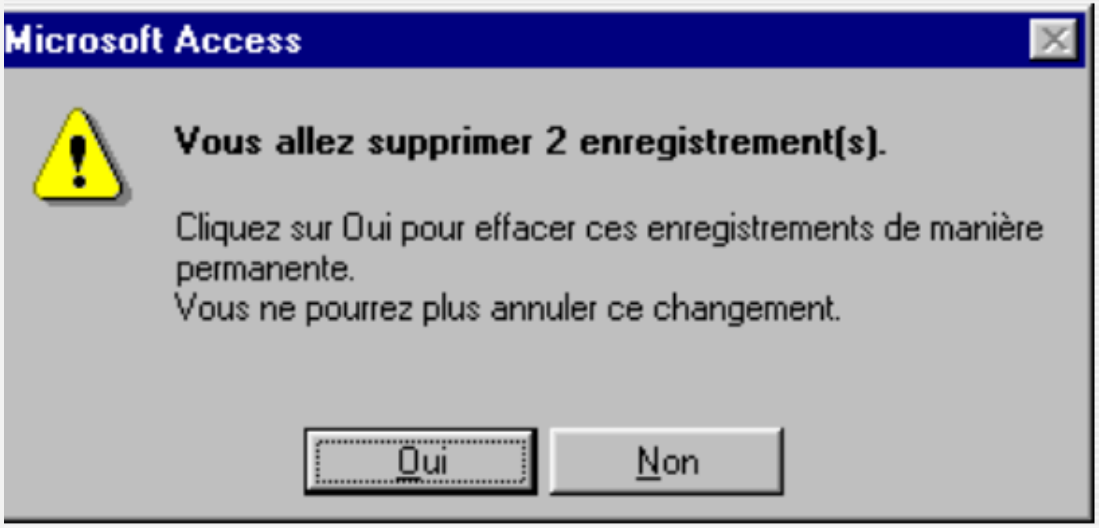
5.3. ERROR CORRECTION

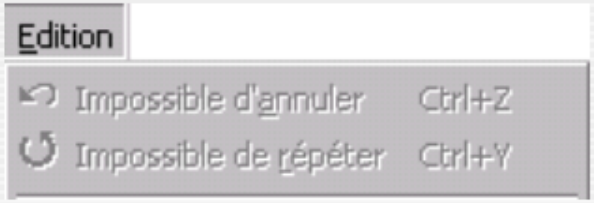
- To Do
  - la vérification des saisies
  - Mettre en évidence le champ/élément erroné



5.3. ERROR CORRECTION

Not to Do





6.HOMOGENEITY / CONSISTENCY

The consistency criterion concerns the overall homogeneity of the human-machine interface.

The goal is to maintain a coherent logic for:

Presentation (graphics, layout, vocabulary, format, syntax, ...)

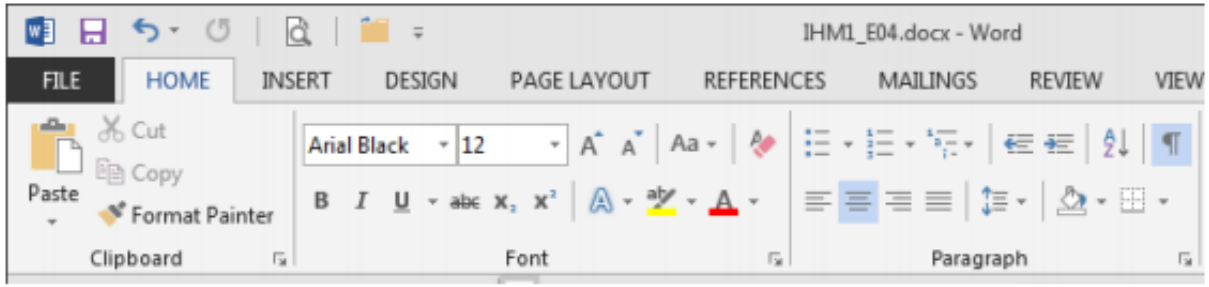
Behavior(system reactions, messages, auditory feedback, ...)

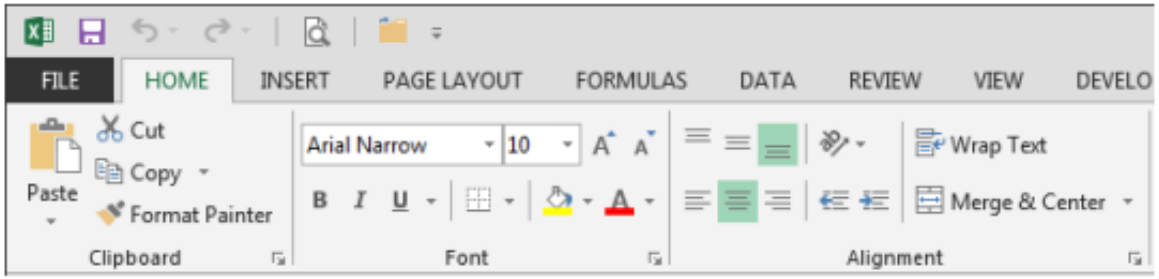
Recommendations:

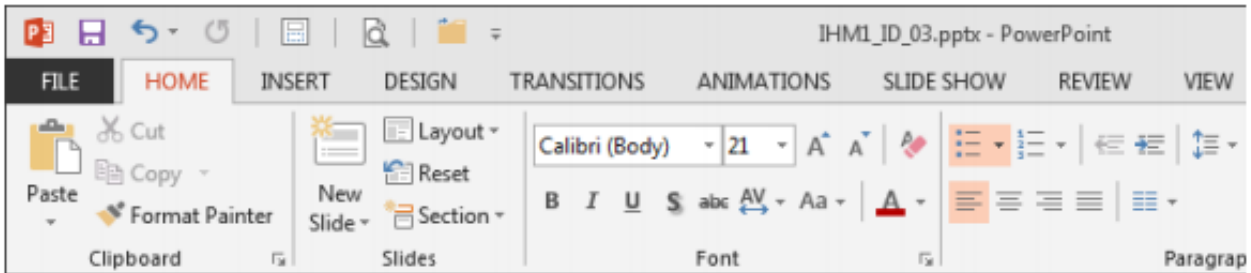
- Use the same layout scheme for all windows (screen template, also called regulatory layout)
- The semantics of mouse buttons should remain constant
- The same vocabulary should be used to refer to the same functions
- Use a consistent organization and syntax for menus
- Use graphical symbols (icons, colors, ...) in a consistent manner

6.HOMOGENEITY / CONSISTENCY

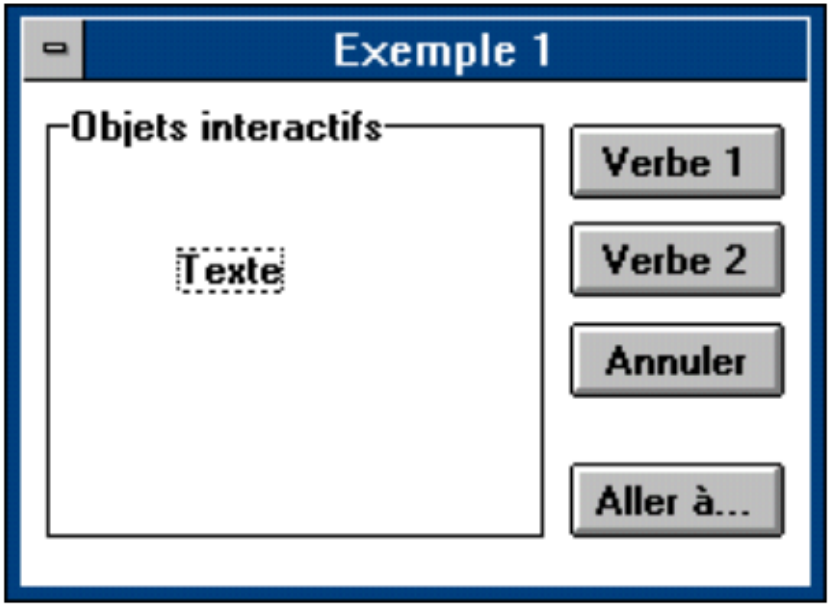
- **Microsoft Office, Word, Excel, PowerPoint: A certain consistency in menus (though there is still significant room for improvement).**

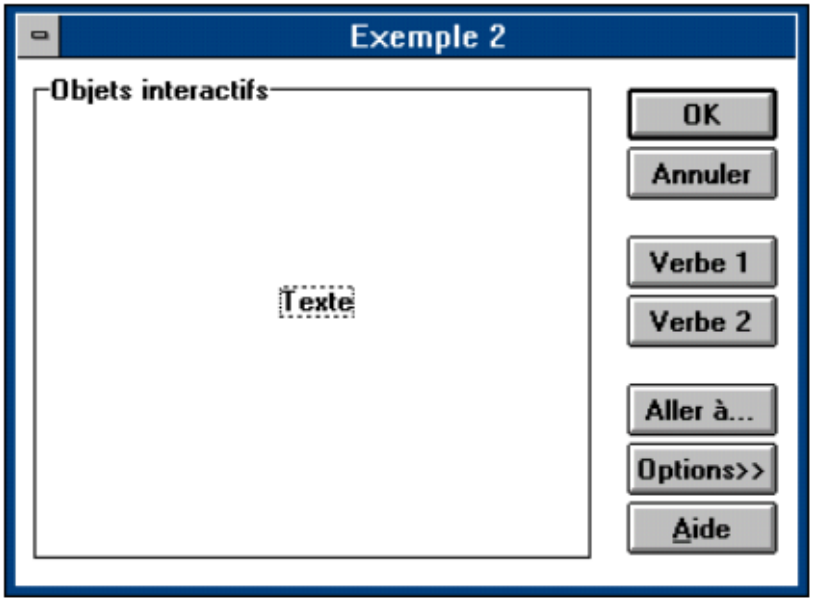






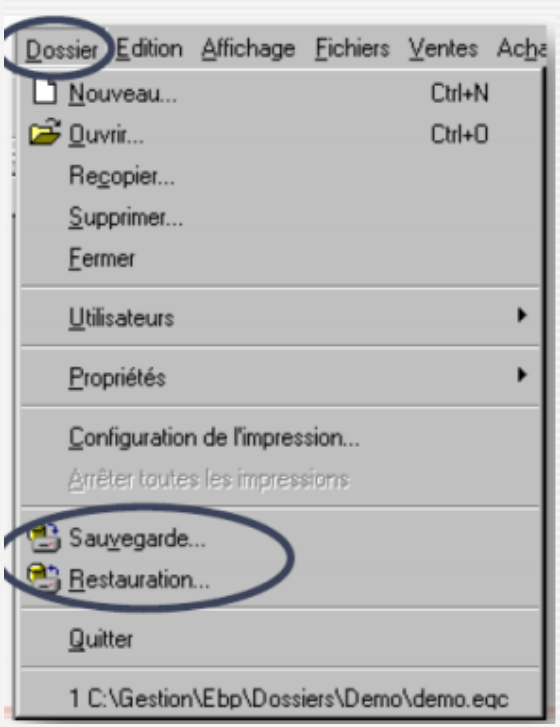
6.HOMOGENEITY / CONSISTENCY

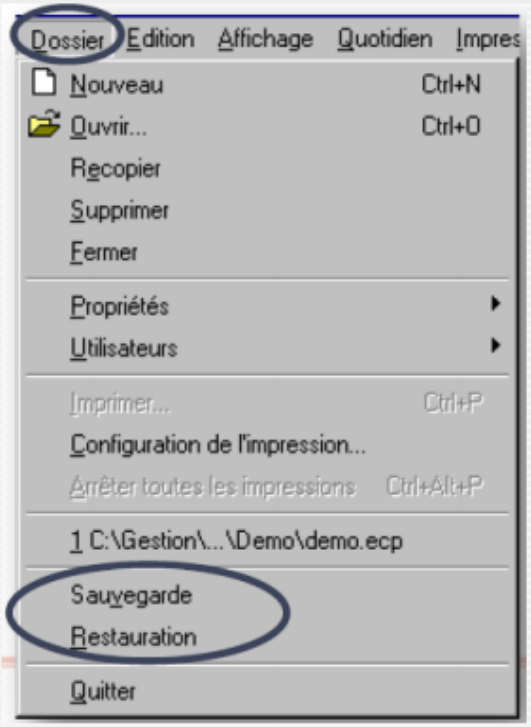




6.HOMOGENEITY / CONSISTENCY

Not to Do





7. MEANINGFULNESS OF CODES AND LABELS

- Meaningfulness of codes and labels characterizes the appropriateness between the object, information, or behavior presented by the interface and its referent (the object or action it represents).
- There must be a clear relationship between the displayed or entered object/information and its referent. In other words, there should be no ambiguity in the meaning given to words, icons, or more broadly, the entire language used.

Recommendations:

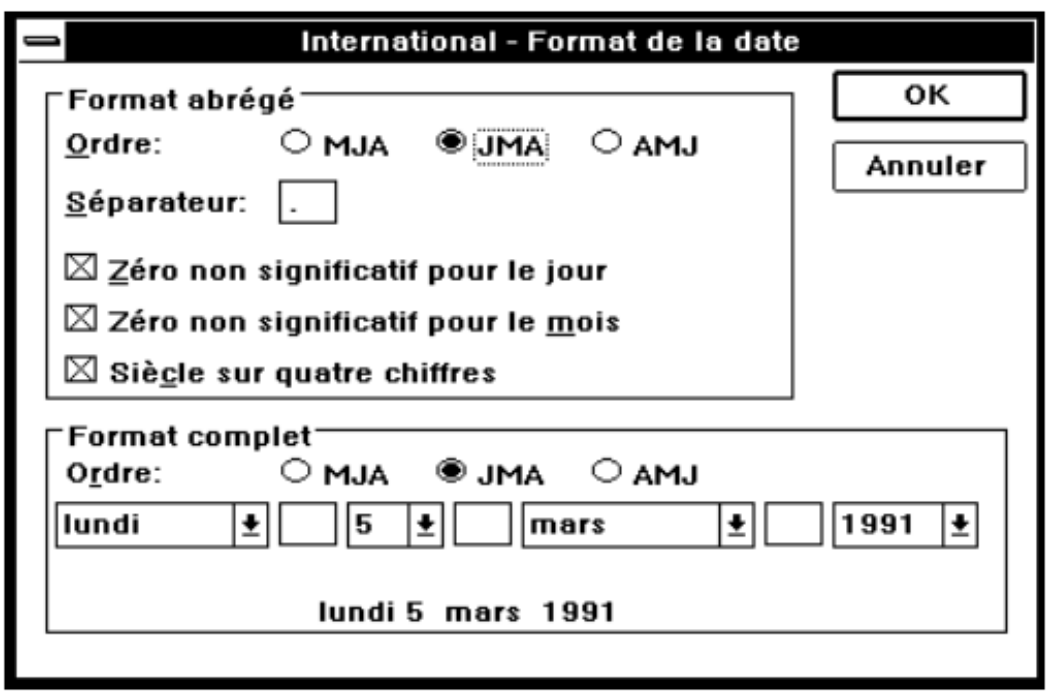
- Avoid technical terms (jargon); use the user's language
- Replicate the familiar behavior of real-world objects (e.g., an eraser)
- Explicitly define and adhere to abbreviation rules
- Consider existing standards (both formal and de facto) for all labels and terms

7. MEANINGFULNESS OF CODES AND LABELS

A virtual lottery ticket that you scratch with the cursor.

7. MEANINGFULNESS OF CODES AND LABELS

Counter Example:



8. COMPATIBILITY

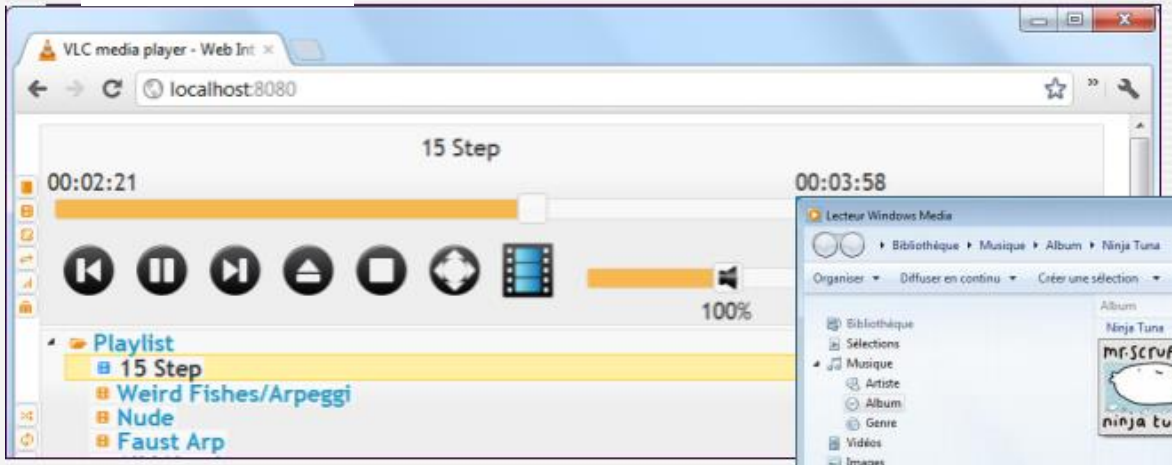
- Compatibility is the software's ability to integrate into the actual activities of users.
- There must be agreement between the characteristics of the users and their tasks on one hand, and the organization of outputs, inputs, and dialogue of a given application on the other.

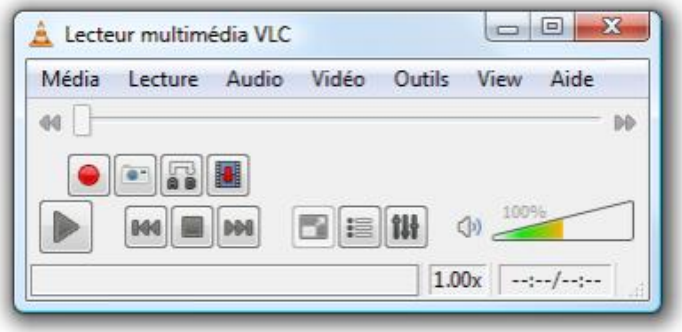
Recommendations:

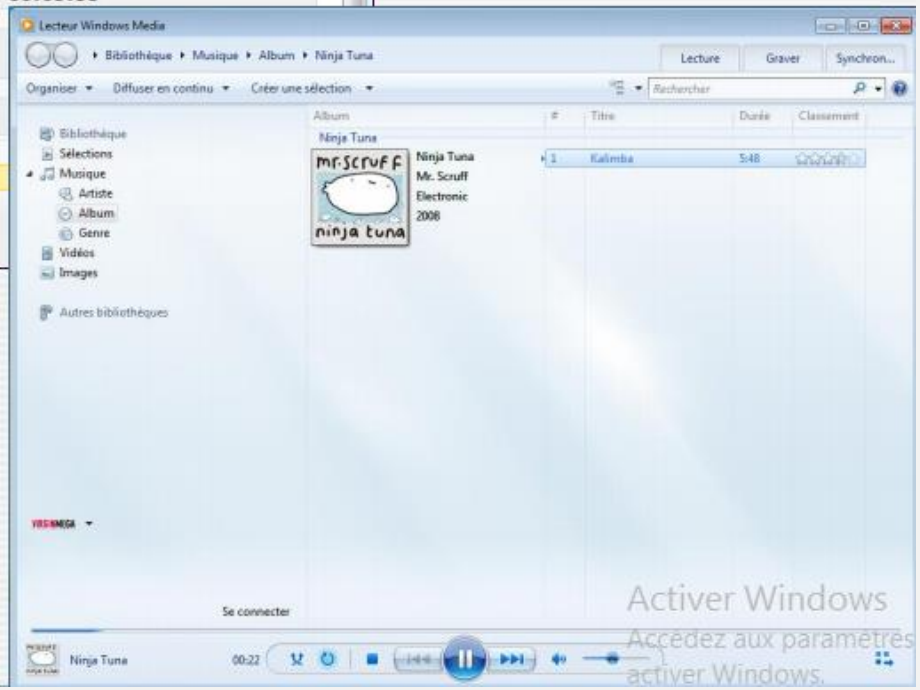
- Use the user's language (avoid computer jargon)
- Use familiar metaphors
- Arrange interface elements according to the user's task
- Present information consistently with other work materials (paper documents, forms, flowcharts, directives, etc.)
- Access to functions must be compatible with the user's task

8. COMPATIBILITY

- To Do







A CHECK-LIST (1)

Criteria	Verification Questions
Guidance	
Prompting	- Are unavailable functions grayed out? - Are expected inputs provided (dropdown lists, codes to use)? - Is the required data entry format indicated (dates, dimensions)? - Does the cursor shape change to indicate the operation? - Are mandatory fields clearly indicated? - Is navigation (forward/backward) clearly shown? - Are there tooltips for non-trivial elements?
Grouping/Distinction	- Are related items grouped together? - Are groups clearly separated? - Is the spatial layout logical and consistent? - Are formatting (color, font) and positioning used to show relationships?
Immediate Feedback	- Does the software respond to every user action? - Are the system's operating modes indicated? - Are long processes indicated with a wait signal (hourglass, progress bar)? - Is user input always displayed? - Is software processing visible? Are failures clearly indicated?
Readability	- Is a readable font used? - Is spacing (line spacing) and alignment taken care of? - Are the sizes of labels and icons adapted? - Are the color contrasts correct? - Is the target machine's configuration (resolution) considered?

A CHECK-LIST (2)

Criteria	Verification Questions
Workload	
Brevity	- Is the number of options in menus or dropdowns limited? - Are overly long labels avoided? - Is the number of actions to achieve a goal reduced? - Does the user have to remember information from one window to another? Perform calculations? Enter information the system can deduce?
Conciseness	- Are icons with labels used instead of long descriptions? - Are default values provided? - Is auto-complete used for common inputs? - Is complex information broken down? - Are calculated results shown? Minimal Actions - Are the steps the user must go through limited? - Is the user asked for data that the system can deduce?
Information Density	- Is only information relevant to the task displayed? - Are overly cluttered screens avoided? - Are there too many links in a text? - Is the text too verbose? - Is recognition (symbols, icons) prioritized?

A CHECK-LIST (3)

Criteria	Verification Questions
Explicit Control	
Explicit Actions	- Are operations triggered without the user's explicit consent? - Is the operation triggered immediately after the user's action? If not, is it clearly indicated that it is delayed/cannot be performed?
User Control	- Is explicit confirmation provided for important or hard-to-reverse commands? - Can long processes be interrupted? - Is undo possible? - Can the current function or software be exited at any time?

A CHECK-LIST (4)

Criteria	Verification Questions
Adaptability	
Flexibility	- Can the user personalize the interface? - Can a task be performed in different ways (dropdown, context menu, toolbar, shortcut)?
Consideration of User Experience	- Can the user define their experience level (or is it determined automatically)? - Is the novice user guided step-by-step (e.g., wizards)? - Can experienced users perform tasks quickly and efficiently?

A CHECK-LIST (5)

Criteria	Verification Questions
Error Management	
Error Protection	- Are unavailable commands grayed out? - Are lists of possible values/units provided? - Are errors detected early, and the user warned immediately? - Is keyboard input minimized (prefer selection lists)? - Are risks of data loss prevented (confirmation requested)?
Quality of Error Messages	- Are error messages placed where the user is likely to look? - Are error messages clear (user's language)? - Are texts too long? - Is the language reproachful? - Are the messages self-sufficient (concise yet comprehensive)?
Error Correction	- Can the user correct only the erroneous portion of data? - Is the user blocked for serious errors until correction? - Can the user continue after a warning for minor errors? - Is an error message displayed for an invalid command? - Is automatic correction used (rare cases)?

A CHECK-LIST (6)

Criteria	Verification Questions
Consistency	- Is the same layout scheme used for all windows? - Is the semantics of the mouse buttons consistent? - Is the same vocabulary used for the same functions? - Is the organization and syntax of menus consistent? - Are graphical symbols (icons, colors) used consistently?
Significance of Codes & Labels	- Are technical terms (jargon) avoided? - Does it replicate the familiar behavior of objects? - Are abbreviation rules explicitly defined and respected? - Are existing standards (formal or de facto) considered for all labels?
Compatibility	- Is the user's language used (avoiding computer jargon)? - Are familiar metaphors used? - Are interface elements arranged according to the user's task? - Is information presented consistently with other work materials? - Is access to functions compatible with the user's task?

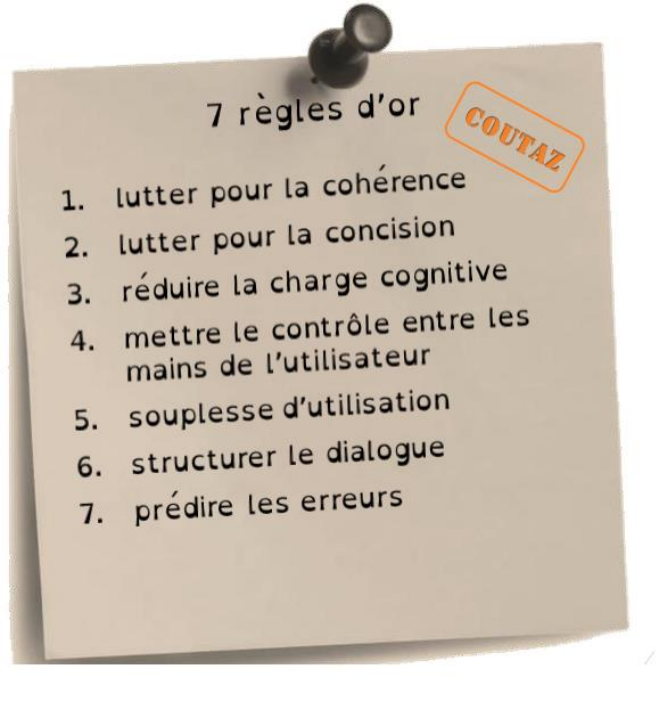
COUTAZ'S GOLDEN RULES

COUTAZ'S GOLDEN RULES

Coutaz's golden rules complement Nielsen's heuristics and focus on creating user interfaces that are **consistent, learnable, and supportive** of user goals.

They emphasize:

- **Ease of learning**
- **Simplicity**
- **Visibility**
- **User control and freedom**
- **Error prevention**
- **Efficiency**
- **Graphic design clarity**



OTHER COMMON HEURISTICS

OTHER COMMON HEURISTICS

- Several other sets of heuristics overlap with Nielsen's and define the same fundamental ergonomic principles.

NORMAN'S FOUR DESIGN PRINCIPLES

Affordance – The appearance of an object should suggest its use.

- *Example: A button looks pressable.*

Natural Mapping – Controls should have a logical relationship to their effects.

- *Example: The stove knobs align with their corresponding burners.*

Visibility – Important features should be visible and easily accessible.

- *Example: A visible "Save" icon instead of a hidden menu option.*

Feedback – The system should provide immediate feedback after an action.

- *Example: A "message sent" confirmation.*

SHNEIDERMAN'S EIGHT GOLDEN RULES

- Strive for consistency.

Enable shortcuts for frequent users.

Offer informative feedback for every action.

Design dialogs with clear beginnings and endings.

Provide guidance and feedback to help users avoid errors.

Support reversible actions (Undo/Redo).

Give users a sense of control — they should feel they are in command.

Reduce short-term memory load by keeping information visible.

TOGNAZZINI'S SIXTEEN PRINCIPLES

- Anticipation
- Autonomy
- Color-blindness consideration
- Consistency
- Defaults
- Efficiency
- Explorability
- Fitts's Law (optimize distance and size of interactive elements)
- Human interface objects
- Latency reduction
- Learning support
- Metaphors
- Work protection
- Readability
- State tracking
- Navigation visibility

Tognazzini's principles focus on designing intuitive, fast, and forgiving interfaces that support both novice and expert users.

THANK YOU