Cours IHM-1
Conception des interfaces

3 - Critères ergonomiques Heuristiques de Nielsen

Critères ergonomiques
(Bastien / Scapin)

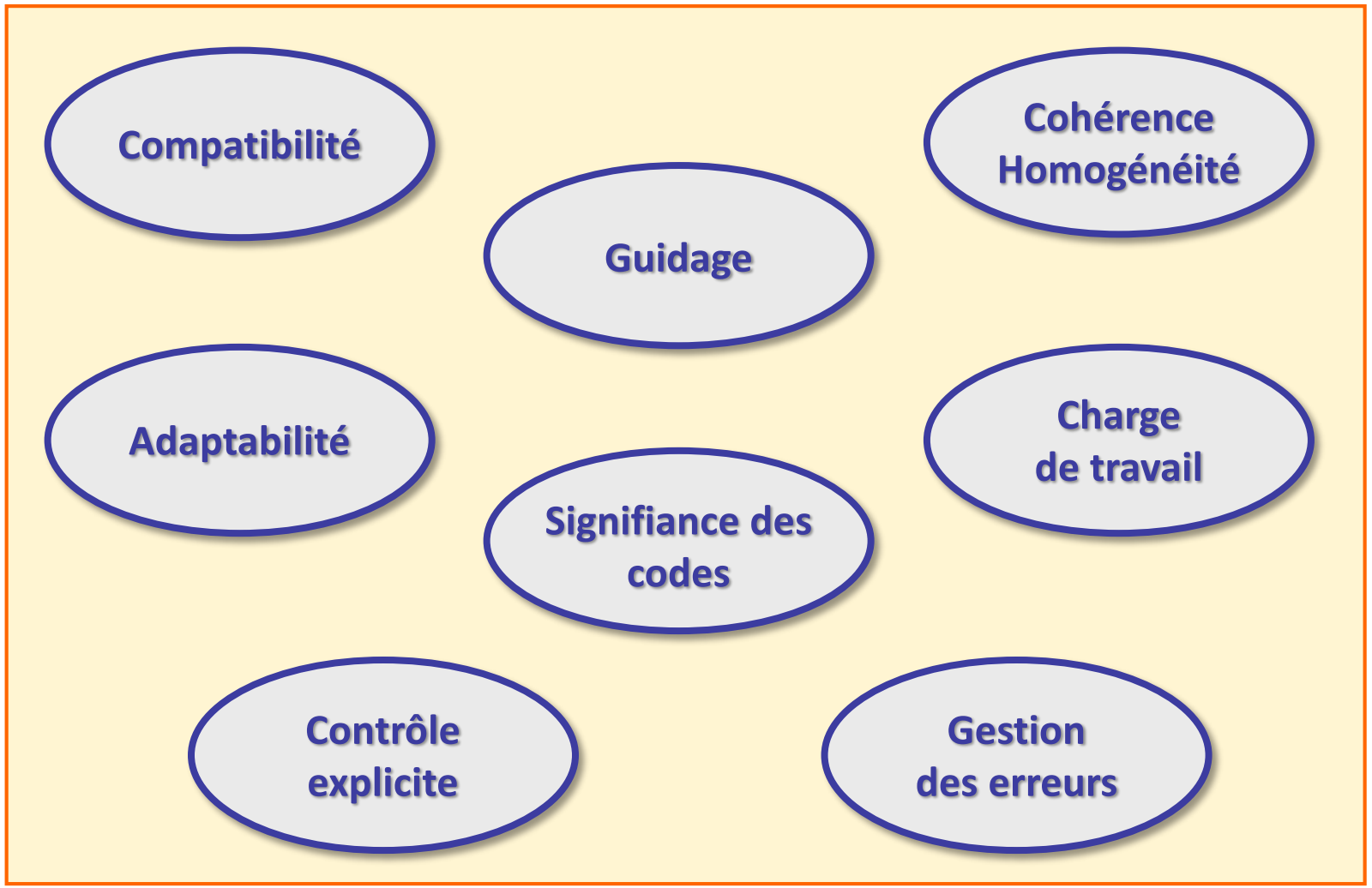
Critères ergonomiques [1]

- Les **critères ergonomiques** constituent une classification (typologie) des règles de base qui conditionnent l'utilisabilité d'une interface.
- L'établissement de critères ergonomiques vise principalement deux objectifs complémentaires :
  - Ils permettent d'**évaluer l'utilisabilité** d'un logiciel (audit ergonomique) en servant de base pour l'établissement de grilles d'évaluation (check-list).
  - D'autre part, et c'est encore plus important, ils servent de **guide lors de la conception** d'une interface utilisateur en aidant les concepteur à prendre en compte - dès la phase initiale du développement - les aspects ergonomiques du logiciel.
- Malgré quelques différences dans la terminologie, le niveau de détail et des nuances de classification, les chercheurs du domaine (C. Bastien, D. Scapin, J-F. Nogier, H.X. Lin, …) s'accordent sur la liste et la classification de ces critères ergonomiques de base qui sont également à l'origine de certaines normes dans le domaine.

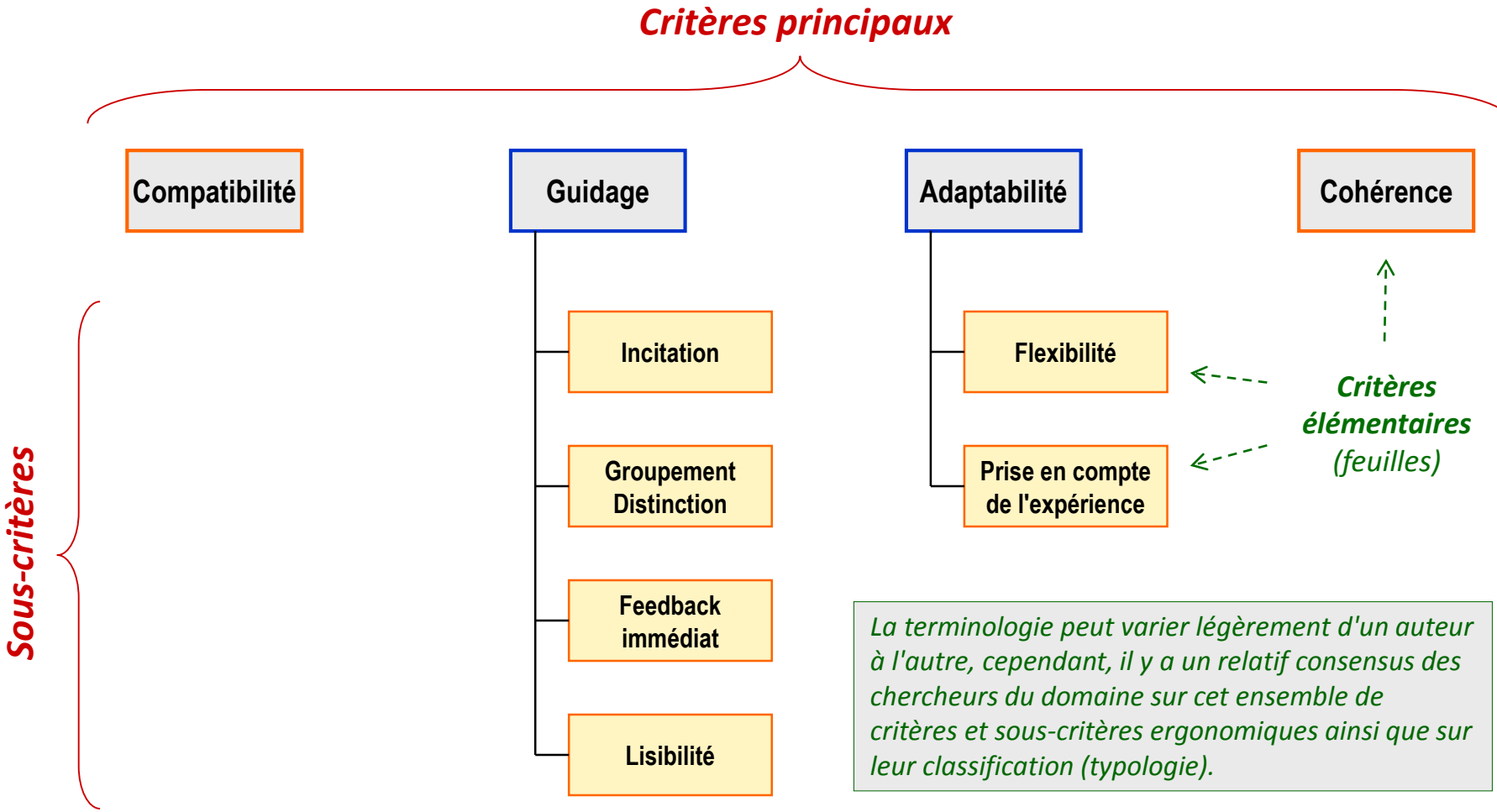
Critères ergonomiques [2]

- A l'origine de cette notion de **critères ergonomiques**, les ergonomes Christian Bastien et Dominique Scapin ont procédé à la synthèse d'environ 900 recommandations rassemblées dans le domaine de l'ergonomie informatique.
- Ils ont cherché à classifier ces recommandations et ont abouti à une liste d'une petite vingtaine de critères élémentaires (critères de base) répartis dans 8 dimensions (familles principales).
  - Ces 8 dimensions ne sont pas totalement orthogonales et il subsiste quelques dépendances entre certaines familles. Cependant, cette classification permet d'aborder de manière beaucoup plus rationnelle la notion d'ergonomie et les nombreuses recommandations associées.
- Ces critères offrent également un cadre et un vocabulaire communs.
- Ils ont fait l'objet d'évaluations expérimentales qui ont démontré qu'ils offraient des avantages mesurables par rapport à l'utilisation d'autres références (notamment la norme ISO 9241-10).

Critères ergonomiques [3]

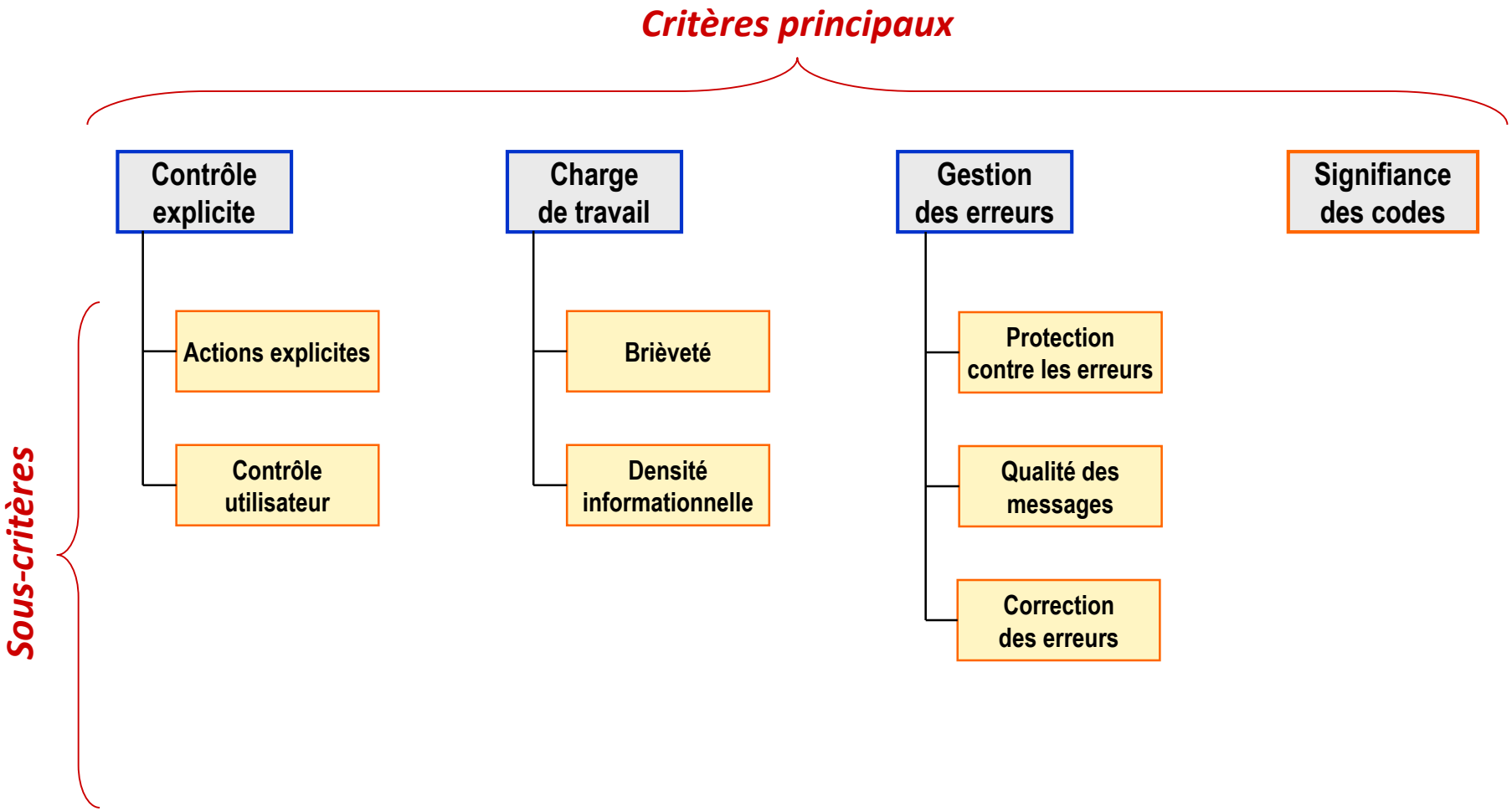


Critères ergonomiques [4]



*Toutes les boîtes avec une **bordure orange** représentent des **critères élémentaires***

Critères ergonomiques [5]



*Toutes les boîtes avec une **bordure orange** représentent des **critères élémentaires***

Compatibilité

- La **compatibilité** est la capacité du logiciel à s'intégrer dans l'activité réelle des utilisateurs.
- Ce critère mesure l'adéquation du logiciel avec le contexte physique et social dans lequel il est utilisé (environnement de travail).
- L'objectif est de réduire le transfert de connaissance entre le métier et le logiciel.

Logique métier ⇔ Logique du logiciel

Recommandations :

- Parler le langage de l'utilisateur (éviter le jargon informatique)
- Utiliser des métaphores familières
- Agencer les éléments de l'interface en fonction de la tâche de l'utilisateur
- Présenter les informations de façon cohérente par rapport aux autres supports de travail (documents papier, formulaires, organigrammes, directives, etc.)
- L'accès aux fonctions doit être compatible avec la tâche de l'utilisateur

Guidage

- Le critère de **guidage** regroupe l'ensemble des moyens mis en œuvre pour assister l'utilisateur dans l'emploi du logiciel.
- Faire connaître à l'utilisateur l'état du système et lui permettre d'établir les liens de causalité entre ses actions et l'état résultant.
- L'objectif est de faciliter l'utilisation du système et son apprentissage.
- L'utilisateur doit comprendre les interactions que l'on attend de lui par un fonctionnement qui apparaît clairement (⇨ moins d'hésitations).
- On distingue
  - Le **guidage explicite** (par ex. messages, bulles d'aide, boîtes de dialogue, …)
  - Le **guidage implicite** (par ex. griser les fonctions inactives, autres artifices, …)
- Le critère de guidage est décomposé en quatre sous-critères :
  - **Incitation**
  - **Groupement / Distinction**
  - **Feedback immédiat**
  - **Lisibilité**

Incitation (Guidage)

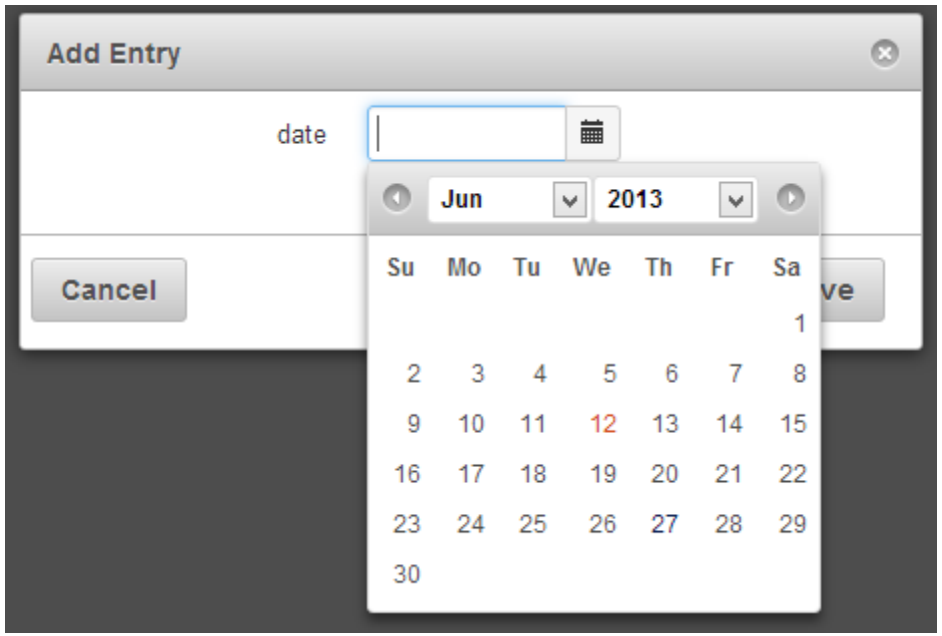
- Le critère d'incitation réunit les moyens visant à conduire l'utilisateur à effectuer des actions spécifiques.
- L'incitation aide l'utilisateur dans son interaction avec le logiciel en lui fournissant les éléments nécessaires pour l'utiliser correctement.
- Il évite des apprentissages fastidieux et réduit les risques d'erreur.

Recommandations :

- Griser les fonctions non disponibles (options de menu, boutons, …)
- Fournir la liste des saisies attendues (listes déroulantes, codes à utiliser, …)
- Donner le format de saisie des données (dates, dimensions, …)
- Modifier la forme du curseur (pointeur de la souris) pour donner des indications sur l'opération à effectuer
- Indiquer clairement les champs obligatoires (* ou autre indicateur)
- Montrer clairement comment aller en avant et en arrière
- Afficher des bulles d'aides *(Tooltips)* sur les éléments non-triviaux (icônes des barres d'outils par exemple)

Incitation (Guidage)

- Inciter l'utilisateur à saisir correctement les données.
- Indiquer ou imposer le format (dates, dimensions, unités, …)



Incitation (Guidage)

- Un exemple d'incitation réussie :
  - Le numéro d'abonné est divisé en trois blocs de trois chiffres
  - Une fois un premier bloc saisi, le curseur se déplace automatiquement au bloc suivant

Incitation (Guidage)

- Un exemple ou l'incitation est si mauvaise que l'on doit explicitement dire à l'utilisateur où il doit cliquer !

Groupement - Distinction (Guidage)

- Le critère de **groupement** consiste à guider l'utilisateur en groupant les informations et les fonctions de même type.
- Un utilisateur considère généralement que ce qui se ressemble va fonctionner de la même manière (principe de similarité).
- Deux attributs graphiques sont utilisés dans ce but :
  - Le **format** de présentation (couleur, forme, syntaxe, …)
  - La **position** dans l'interface (localisation, bordure, …)
- A l'inverse, le critère de **distinction** consiste à séparer clairement ou à attribuer des propriétés distinctes à des objets qui sont différents ou qui se comportent de manière différentes.

Recommandations :

- Regrouper les informations de même type par le format ou par la position (regroupement par menus, entourer avec une bordure, …)
- Distinguer par une présentation différente ou un emplacement différent, les informations distinctes pour éviter les risques de confusions

Groupement - Distinction (Guidage)

Un exemple bien réussi, bonne structuration de la page

Mauvais groupement (risque de confusion)

Feedback immédiat (Guidage)

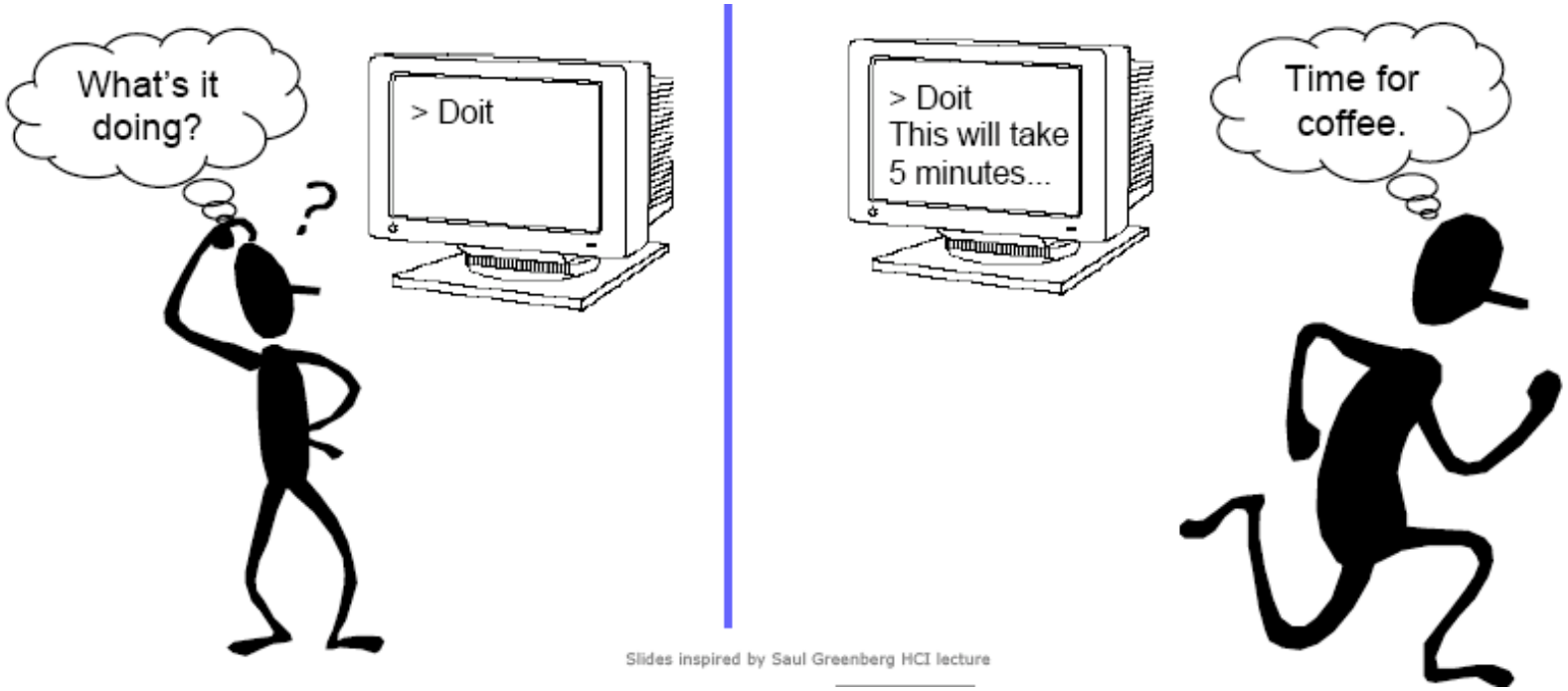
- Le critère de **feedback immédiat** réunit tous les éléments qui servent à montrer à l'utilisateur ce qu'est en train de faire le système et en lui offrant une rétroaction en réponse à chacune de ses actions.
- Le respect de ce critère, appelé parfois "**retour utilisateur**", contribue grandement à accroître la **confiance** de l'utilisateur.
- Il établit plus facilement le **lien entre les actions qu'il effectue et l'état résultant du système**; il comprend ainsi mieux son fonctionnement (se forge une image mentale) et apprend plus rapidement à l'utiliser.

Recommandations :

- Le logiciel doit répondre à toute action de l'utilisateur par un changement dans la présentation de l'interface (visuel, sonore, ...)
- Indiquer les modes de fonctionnement du système (état)
- Signaler les traitements longs par une indication d'attente (sablier, barre de progression, animation, message, …)
- Toujours faire apparaître les saisies utilisateur
- Rendre visible les traitements réalisés par le logiciel et indiquer clairement lorsqu'ils ont échoué

Feedback immédiat (Guidage)

- Signaler les traitements longs par une indication d'attente (disque, barre de progression, animation, message, …). Trois ordres de grandeur :
  - **0.1 s** : Non perceptible par l'utilisateur
  - **1 s** : Perceptible (un indicateur est nécessaire)
  - **10 s** : L'utilisateur peut faire autre chose
    (annoncer le temps d'attente et la progression, si possible)



Slides inspired by Saul Greenberg HCI lecture

Lisibilité (Guidage)

- Le critère de **lisibilité** consiste à faciliter la perception des informations textuelles et iconographiques par un choix judicieux de leurs propriétés et de leur disposition.
- Une attention particulière doit être apportée :
  - Aux **polices** de caractères (diversité, taille, attributs, minuscules/majuscules, …)
  - Aux **couleurs** en général (choix, contraste, but, …)
  - À la **disposition** des éléments (alignements, espacements, mise en page, …)
  - Aux **icônes** (dimensions, choix des symboles, …)
- Un nombre important de règles peuvent être associées à ce critère.

Recommandations :

- Utiliser une police de caractère lisible (à l'écran, éviter les italiques en petite taille, …)
- Soigner les espacements (interligne) et les alignements (justification)
- Adapter la taille des libellés et des icônes afin de garantir leur interprétation
- Soigner les contrastes (choix des couleurs pour le fond et le premier plan)
- Prendre en compte la configuration de la machine cible (résolution, …)

Cohérence - Homogénéité

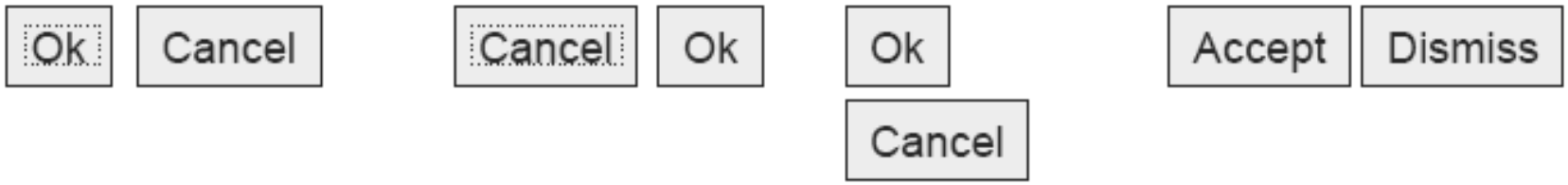
- Le critère de **cohérence** concerne l'**homogénéité** globale de l'interface homme-machine.
- L'objectif est de respecter une **logique cohérente** pour :
  - La **présentation** (graphisme, localisation, vocabulaire, format, syntaxe, …)
  - Le **comportement** (réaction du système, messages, retours sonores, …)
- L'homogénéité rend le système stable, donc prévisible aux yeux de l'utilisateur. Le temps de recherche de l'information est diminué.
- Rend l'apprentissage plus rapide car **ce qui est appris à un endroit est également applicable ailleurs** (généralisation).

Recommandations :

- Utiliser le même schéma d'agencement pour toutes les fenêtres (gabarit d'écran appelé aussi *tracé régulateur*)
- La sémantique des boutons de la souris doit être constante
- Le même vocabulaire doit être utilisé pour désigner les mêmes fonctions
- Utiliser une organisation et une syntaxe cohérente pour les menus
- Utiliser de manière cohérente les symboles graphiques (icônes, couleurs, …)

Cohérence - Homogénéité

- Maintenir une logique cohérente de présentation et de comportement.
- Menus, barres d'outils, boîtes de dialogue, ...

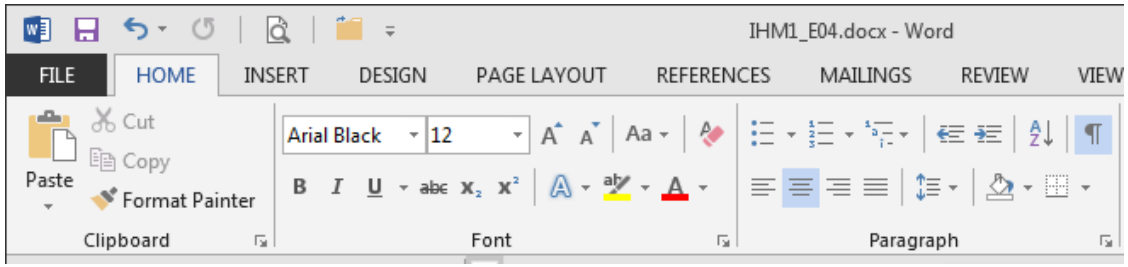


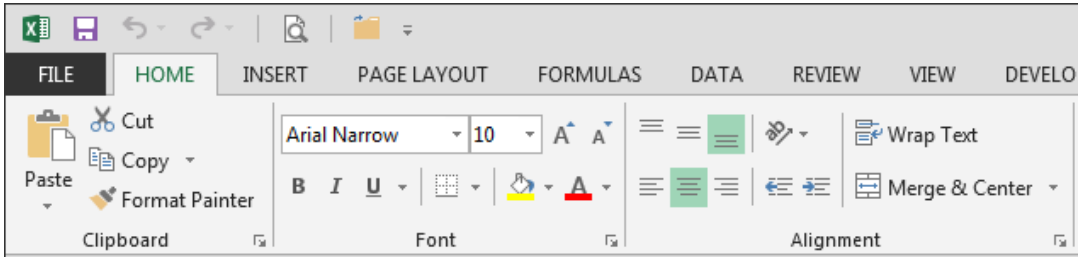
Remarque : *Certains shareware utilisent volontairement ce genre d'incohérences (le bouton* "Continuer" *qui change constamment de place par exemple) pour inciter l'utilisateur à se procurer la version payante du logiciel.*

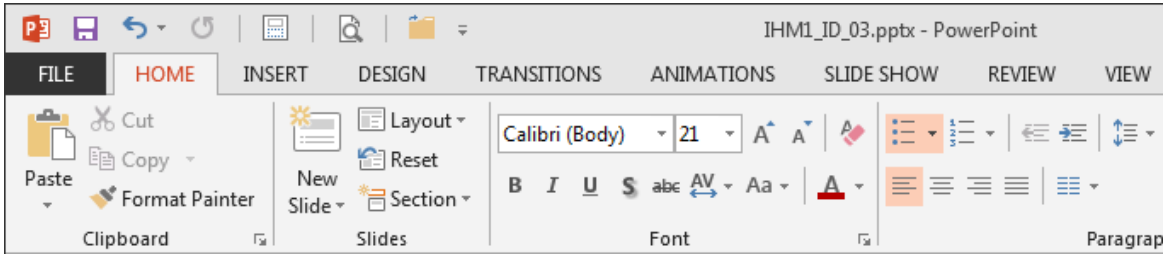
Cela illustre bien la *"**force de l'habitude**" (mémoire visuelle et mémoire du geste) que l'on peut exploiter positivement ou négativement.*

Cohérence - Homogénéité

- *Microsoft Office, Word, Excel, PowerPoint :* une certaine cohérence dans les menus (mais qui pourrait être encore largement améliorée).







Adaptabilité

- L'**adaptabilité** d'un système est caractérisée par la capacité de son interface à réagir et à s'**adapter en fonction du contexte** et selon les besoins et les préférences de ses utilisateurs.
- Plus les manières offertes pour effectuer une action sont nombreuses et plus les chances que l'utilisateur maîtrise l'une d'entre elles sont importantes.
- Un mécanisme unique peut difficilement convenir à la fois à tous les utilisateurs potentiels (notamment s'ils sont nombreux comme dans le cas d'applications très généralistes).
- Le critère d'adaptabilité se décompose en deux sous-critères élémentaires :
  - **Flexibilité**
  - **Prise en compte de l'expérience de l'utilisateur**

Flexibilité (Adaptabilité)

- Le critère de **flexibilité** concerne les moyens mis à la disposition des utilisateurs pour personnaliser l'interface afin de tenir compte des préférences, des aptitudes, des habitudes, du contexte d'utilisation et de la manière de travailler de chacun d'eux.
- Il vise également à offrir à l'utilisateur la possibilité d'effectuer une tâche ou activer une fonction de différentes manières, par exemple :
  - Par un menu déroulant
  - Par un menu contextuel
  - Par une icône dans une barre d'outils
  - Par un raccourci clavier

Remarque : Les différents moyens offerts doivent être complémentaires et pas simplement redondants !

Recommandations :

- Permettre d'effectuer les tâches (fonctions) à la fois au clavier et à la souris
- Autoriser le déclenchement d'une commande fréquente depuis plusieurs endroits dans l'application.
- Permettre à l'utilisateur de paramétrer le logiciel selon ses préférences
- Fournir un moyen rapide d'accéder aux commandes des menus (raccourcis)

Prise en compte de l'expérience (Adaptabilité)

- Le deuxième sous-critère de la flexibilité est la **prise en compte de l'expérience de l'utilisateur** qui concerne les moyens mis en œuvre pour s'adapter aux différents niveaux d'expérience des l'utilisateurs.
- Pour satisfaire ce critère, des moyens différenciés doivent être prévus pour tenir compte des différences d'expérience qui existent entre les utilisateurs selon leur profil.
- Il est à noter que l'expérience des utilisateurs peut varier dans le temps.

Recommandations :

- Permettre à l'utilisateur de définir son niveau d'expérience.
  Le demander ou le déterminer automatiquement par des métriques
  (par exemple le nombre ou le taux d'erreurs, les temps de réaction, etc.).
- Guider l'utilisateur débutant (novice) pas à pas dans la réalisation de la tâche.
- Mettre en place un guidage fort en créant des assistants *(wizard)*.
- Donner aux utilisateurs expérimentés les moyens d'effectuer leur tâche de manière rapide et efficiente (même si c'est parfois au détriment du guidage).

Contrôle explicite

- Le critère de **contrôle explicite** concerne les aspects liés au **degré de maîtrise** qu'a l'utilisateur sur les traitements réalisés par le système interactif.
- Il se décompose en deux sous-critères élémentaires :
  - **Actions explicites**
  - **Contrôle utilisateur**
- Quand les opérations du système résultent directement des actions des utilisateurs, on observe moins d'erreurs et la compréhension du fonctionnement de l'application est facilitée (représentation mentale).
- C'est un facteur important d'acceptation du logiciel : les utilisateurs n'aiment pas être *menés par le bout du nez* et se sentir asservis à la machine.
- Si l'utilisateur a, en tout temps, le contrôle du dialogue, les réactions du système sont mieux prévisibles et l'apprentissage s'en trouve facilité.

Actions explicites (Contrôle explicite)

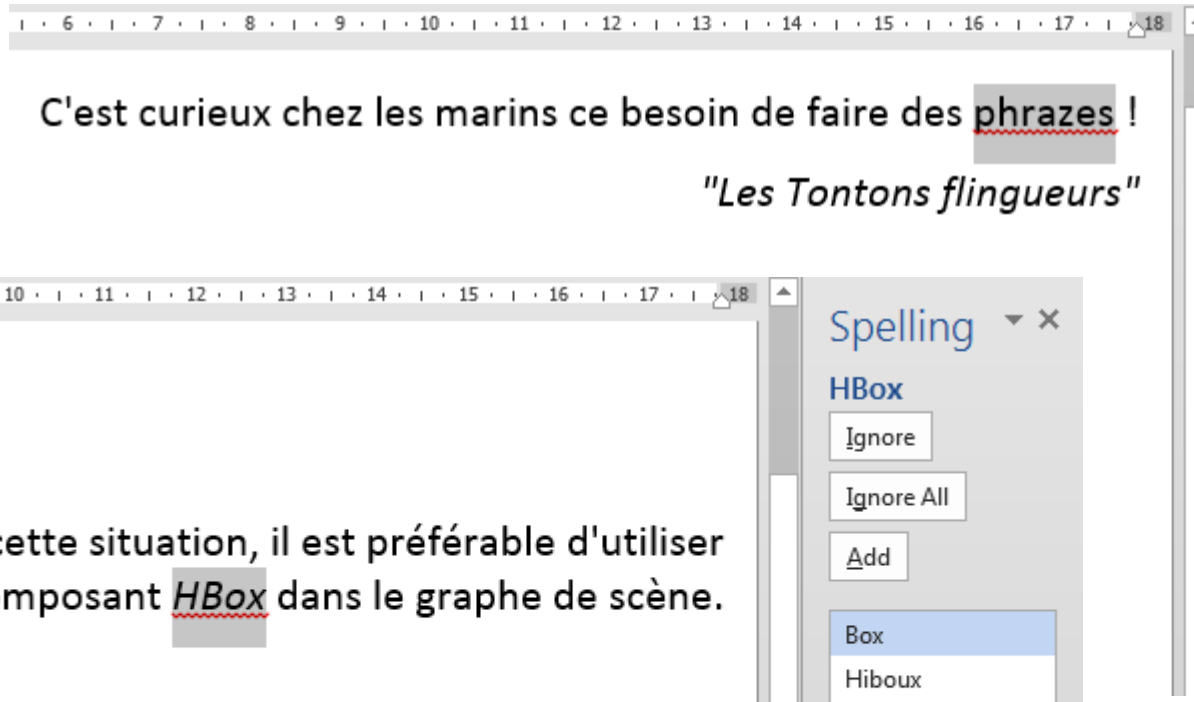
- Le critère d'**actions explicites** concerne la relation qui existe entre les actions de l'utilisateur et les traitements qui seront effectués par l'application en réaction à ces actions.
- Seule les opérations **explicitement demandées** par l'utilisateur doivent être effectuées par le système. Des automatismes peuvent être tolérés s'ils ont été **explicitement acceptés** par l'utilisateur.
- Les opérations devraient être effectuées **au moment où on les invoque** sauf naturellement s'il s'agit d'actions qui sont explicitement différées, par leur nature ou selon les souhaits de l'utilisateur (déclenchement temporisé, synchronisation avec la fin d'un autre processus, etc.).

Recommandations :

- Ne pas déclencher d'opérations sans le consentement explicite de l'utilisateur.
- Déclencher l'opération immédiatement après l'action de l'utilisateur ou, sinon, indiquer clairement que l'opération sera différée (ou qu'elle ne peut pas être effectuée).

Actions explicites (Contrôle explicite)

- Par exemple : Demander confirmation à l'utilisateur avant d'effectuer des changements dans un document.



Contrôle utilisateur (Contrôle explicite)

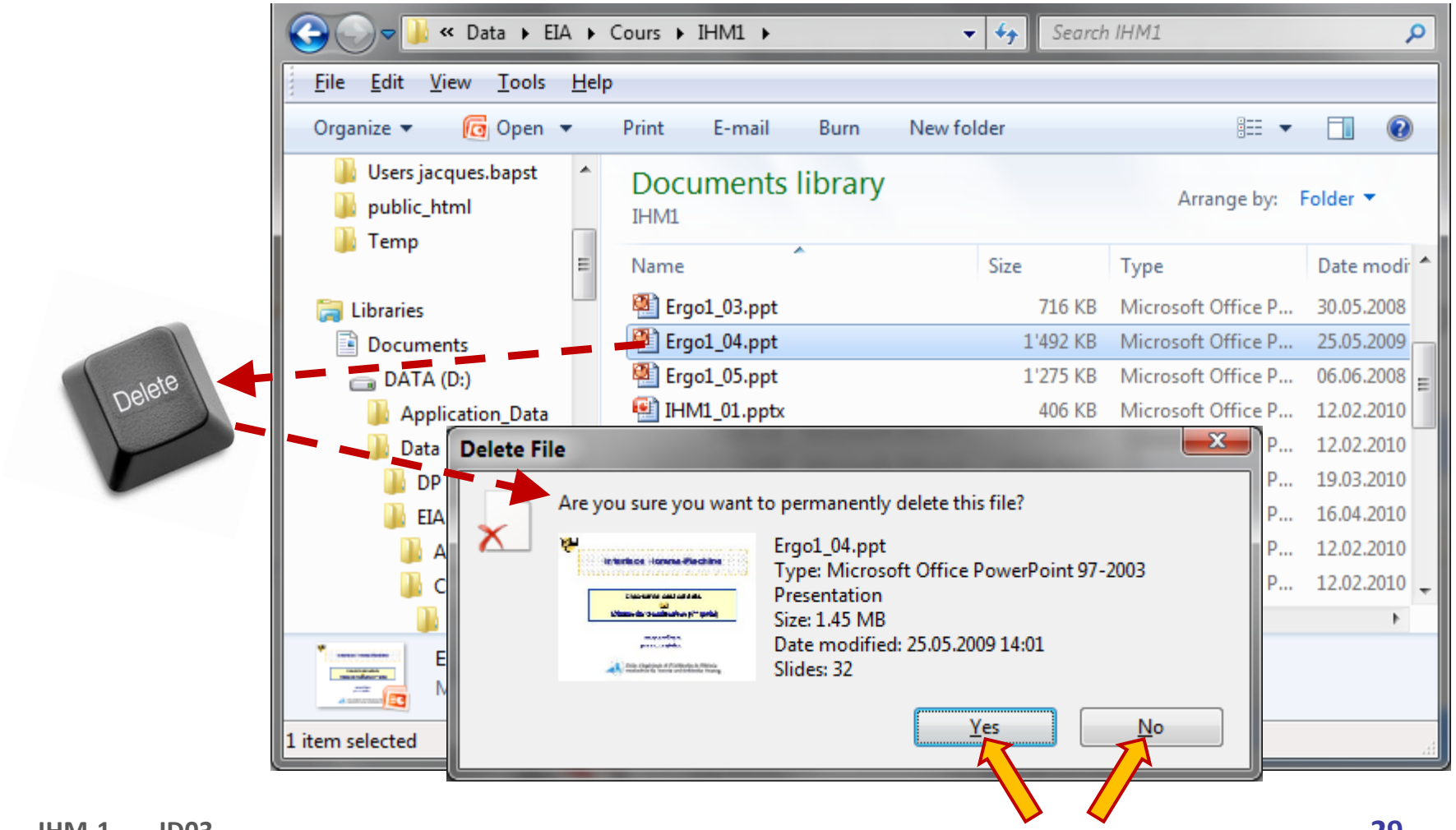
- Le critère de **contrôle utilisateur** concerne le fait que l'utilisateur doit toujours avoir la main sur le système et en contrôler les opérations et leur déroulement (interrompre, reprendre).
- Il vise à rendre l'utilisateur autonome dans son interaction avec le système en lui donnant la **maîtrise du processus**.
- L'utilisateur doit avoir, en permanence, le contrôle sur le logiciel.
- Si possible, ses actions devraient lui être facilitées et des options de contrôle appropriées devraient lui être fournies en fonction de l'état actuel du système (prise en compte de la situation courante).

Recommandations :

- Offrir à l'utilisateur une validation explicite des commandes importantes ou difficilement réversibles
- Offrir la possibilité d'interrompre les traitements longs
- Autoriser les retours en arrière *(Undo)*
- Permettre, en tout temps, de quitter la fonction courante ou même, le logiciel

Contrôle utilisateur (Contrôle explicite)

- Offrir à l'utilisateur une validation explicite des commandes importantes ou difficilement réversibles



Gestion des erreurs

- Le critère de **gestion des erreurs** regroupe les différents moyens visant à **éviter** ou à **réduire** les erreurs de l'utilisateur et lui permettre, le cas échéant, de les **corriger** afin de conserver l'intégrité du système.
- L'objectif prioritaire est de minimiser les interruptions dues aux erreurs (maximiser la performance, l'efficience, la productivité).
- Trois sous-critères sont à prendre en compte :
  - **Protection contre les erreurs** (éviter que l'utilisateur n'en commette)
  - **Qualité des messages d'erreur** (informer clairement l'utilisateur)
  - **Correction des erreurs** (lui permettre de les corriger)

Recommandations générales :

- Tout mettre en œuvre pour éviter les erreurs (attitude défensive)
- Engager un dialogue lorsque des actions de l'utilisateur peuvent conduire à des situations irréversibles (ou des conséquences fâcheuses)
- Avertir l'utilisateur au plus tôt et le guider vers la résolution du problème
- Faciliter l'exploration et l'apprentissage du système

Protection contre les erreurs

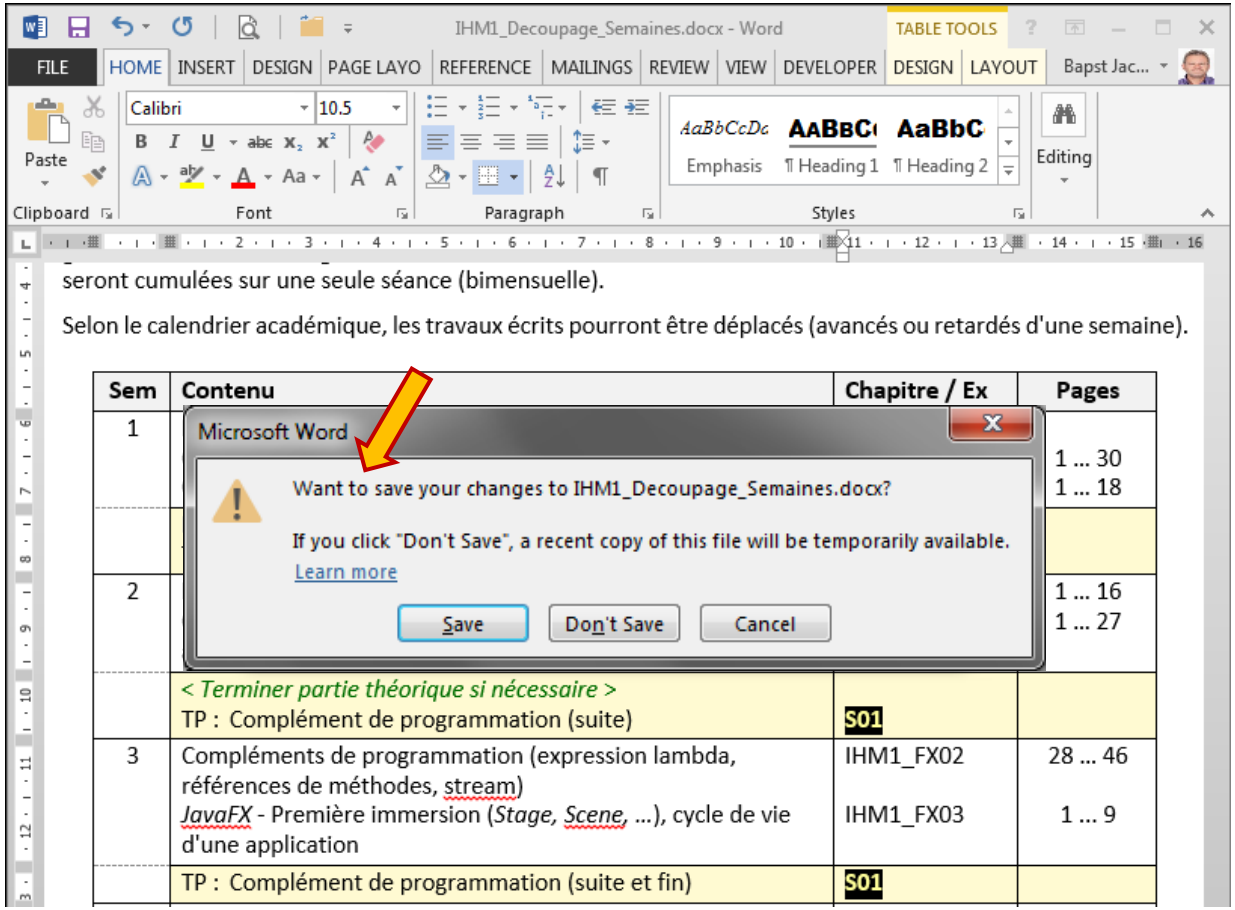
- Le critère de **protection contre les erreurs** vise à éviter que l'utilisateur commette des erreurs et que, si cela se produit, il en soit averti immédiatement (c'est en fait du guidage).
- L'objectif est d'éviter que les interruptions dues aux erreurs ne nuisent au bon déroulement de la tâche.
- On prévient les erreurs en guidant l'utilisateur de manière à ce qu'il ne soit pas enclin à en commettre (lien avec critère de guidage/incitation).
- Si une erreur survient malgré tout, il faut la détecter au plus tôt et avertir l'utilisateur avec un message clair.

Recommandations :

- Mettre en évidence les commandes non disponibles (griser) *Prévention*
- Fournir la liste des valeurs possibles, des unités, … *Prévention*
- Détecter les erreurs au plus tôt et avertir immédiatement l'utilisateur
- Minimiser les saisies au clavier (si possible, listes à choix)
- Prévenir les risques de perte de données (demander confirmation)

Protection contre les erreurs

- Prévenir les risques de perte de données (demander confirmation)



Qualité des messages d'erreurs

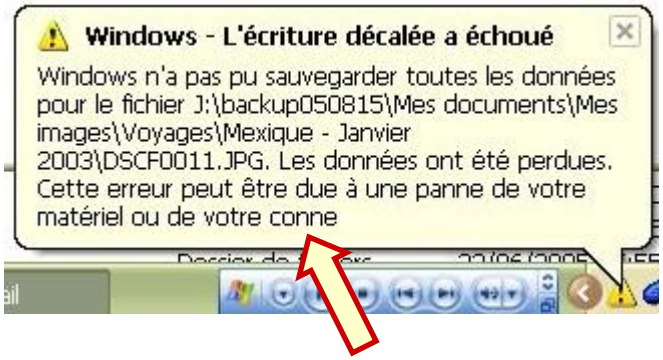
- Le critère de **qualité des messages d'erreur** concerne la pertinence, la facilité de lecture et l'exactitude des informations données aux utilisateurs sur la nature des erreurs commises ainsi que sur les actions à entreprendre pour les corriger.
- Pour que la correction de l'erreur soit aisée, le message d'erreur doit indiquer la **nature de l'erreur**, sa **cause**, ainsi que les **moyens de la corriger** (la qualité des messages favorise l'apprentissage du système).
- Le contenu des messages doit tendre à dédramatiser les erreurs aux yeux de l'utilisateur (diminue son sentiment de culpabilité et le stress induit).

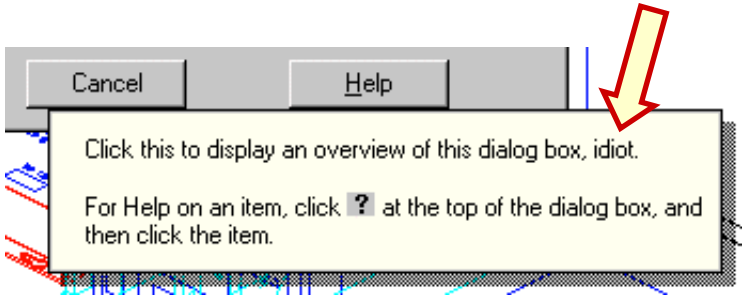
Recommandations :

- Placer les messages d'erreur là où l'utilisateur est censé regarder
- Afficher des messages d'erreur explicites (utiliser le langage de l'utilisateur)
- Éviter les textes trop longs (rester bref, utiliser des liens, des références, ...)
- Éviter les textes réprobateurs
- Faire en sorte, si possible, que les textes soient auto-suffisants. Trouver le bon compromis entre concision et exhaustivité

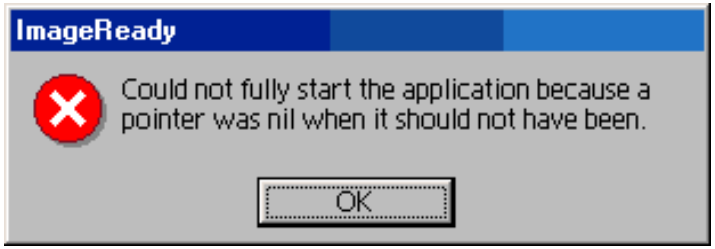
Qualité des messages d'erreurs

- Exemples à ne pas suivre !





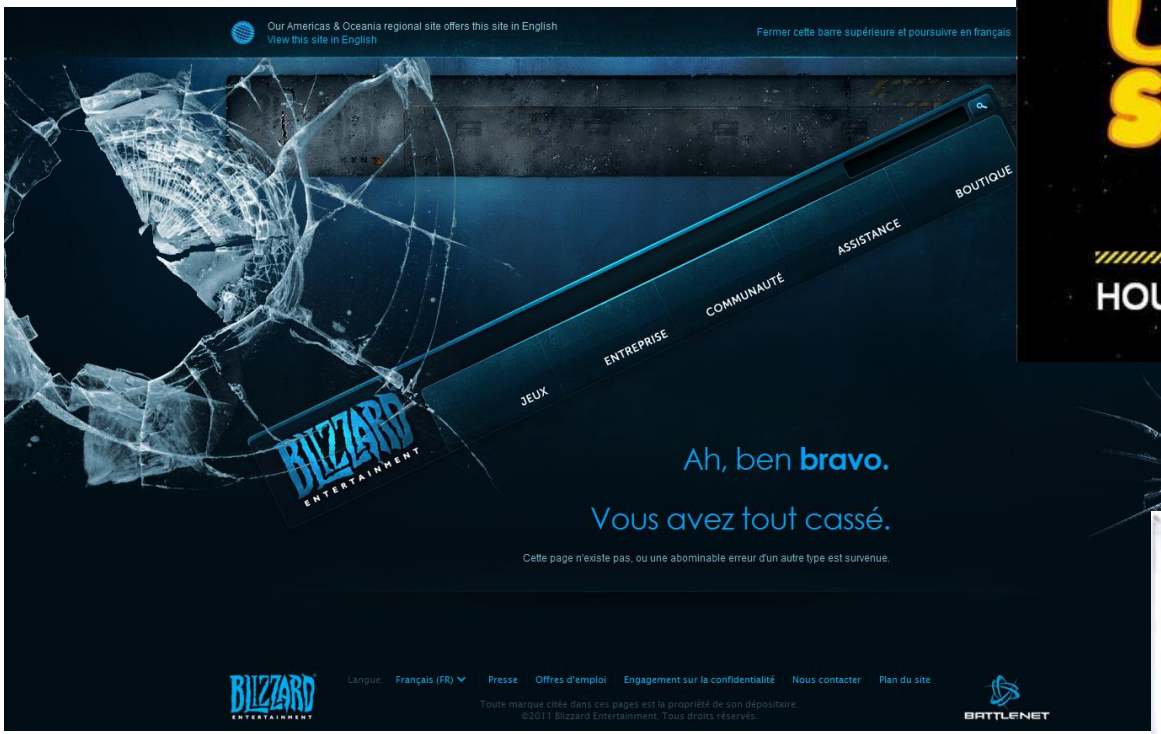
AutoCAD Mechanical

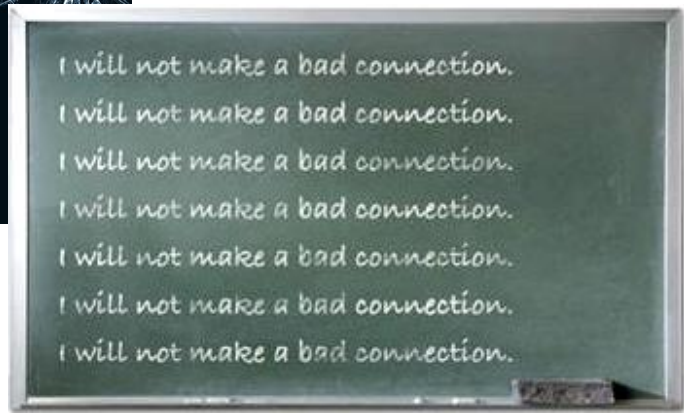


Adobe's *ImageReady*

Qualité des messages d'erreurs

- Exemples de pages associées à la fameuse ***Erreur 404*** (HTTP)





(Gestion des erreurs)

Correction des erreurs

- Le critère de **correction des erreurs** inclut tous les moyens mis à la disposition des utilisateurs pour leur permettre de corriger leurs erreurs.
- Offrir à l'utilisateur les moyens de corriger le plus facilement et le plus rapidement possible ses erreurs en lui permettant, par exemple, de ne corriger que la portion des données qui est erronée et ne pas le contraindre à ressaisir l'ensemble des informations.
- Tactiques différentes selon le type et la gravité de l'erreur :
  - Bloquer l'utilisateur tant que l'erreur subsiste (erreurs graves)
  - Lui permettre de continuer après une mise en garde (message, signal sonore)
  - Ne pas répondre à la commande erronée (un message est nécessaire)
  - Corriger automatiquement l'erreur (dans quelques rares cas)

Recommandations :

- Mettre en évidence le champ ou l'élément erroné
- Permettre d'annuler une action ou une série d'actions (par ex. supprimer des articles dans le caddie virtuel d'une application e-commerce).
- Proposer des alternatives (par ex. *"Related topics"* pour une recherche infructueuse)

Charge de travail

- Le critère de **charge de travail** regroupe l'ensemble des moyens visant à **réduire la charge perceptive**, **mnésique** et **physique** de l'utilisateur.
- L'objectif est d'allouer le maximum de ressource au système cognitif (raisonnement) et de minimiser les risques d'erreur en réduisant le nombre de stimuli du système sensoriel et le nombre d'activités motrices.
- On doit donc viser à minimiser à la fois la quantité d'informations que l'utilisateur doit prendre en compte ainsi que le nombre d'actions élémentaires qu'il doit réaliser pour accomplir une tâche donnée.
- Dans les interfaces utilisateurs classiques (écran, clavier, souris), la charge physique (ergonomie physique) intervient assez peu.
- Le critère de la charge de travail se décompose en deux sous-critères élémentaires :
  - **Brièveté**
  - **Densité informationnelle**

Brièveté (Charge de travail)

- Le critère de **brièveté** regroupe l'ensemble des moyens visant à **réduire la charge perceptive et mnésique** de l'utilisateur dans ses interactions avec les **composants** d'entrée ou de sortie de l'interface utilisateur (champs d'un formulaire, menus, *widgets*, ...).
- Il s'agit donc notamment de limiter autant que possible le travail de lecture et de saisie de l'utilisateur (on parle de **concision**).
- Ce critère inclut également la notion d'**action minimale** qui vise à minimiser le nombre d'actions nécessaires à atteindre un but, à accomplir une tâche.

Recommandations :

- Limiter le nombre d'options dans un menu ou dans une liste déroulante
- Éviter les libellés trop longs
- Réduire le nombre d'actions élémentaires pour atteindre un objectif donné
- Éviter à l'utilisateur d'avoir à se souvenir d'informations d'une fenêtre à l'autre, d'avoir à faire des calculs ou de saisir des informations qui peuvent être déduites par le système.

Brièveté (Charge de travail)

- Quelles sont les fonctions nécessaires, utiles (l'abondance nuit dans certaines circonstances).

Densité informationnelle

(Charge de travail)

- Le critère de **densité informationnelle** concerne **la charge perceptive** et **mnésique** pour l'**ensemble des éléments** d'une interface utilisateur.
- La performance des utilisateurs est influencée négativement quand la charge informationnelle est trop élevée ou (c'est plus rare) trop faible.
- Il faut donc supprimer tous les éléments sans lien direct avec la tâche en cours et qui pourraient distraire inutilement les utilisateurs.
- Les bannières publicitaires (ou fenêtres pop-up) sont des exemples d'augmentation (inutile ?) de la densité informationnelle de l'interface.

Recommandations :

- N'afficher que les informations pertinentes pour effectuer la tâche (boîtes de dialogue simples, représentations graphiques, …)
- Éviter les écrans trop chargés (décomposer si nécessaire)
- Éviter les liens trop nombreux dans un texte affiché sur une page web
- Éviter les textes trop verbeux (dialogue simple, phrases courtes)
- Privilégier la reconnaissance (symboles, icônes)

Densité informationelle

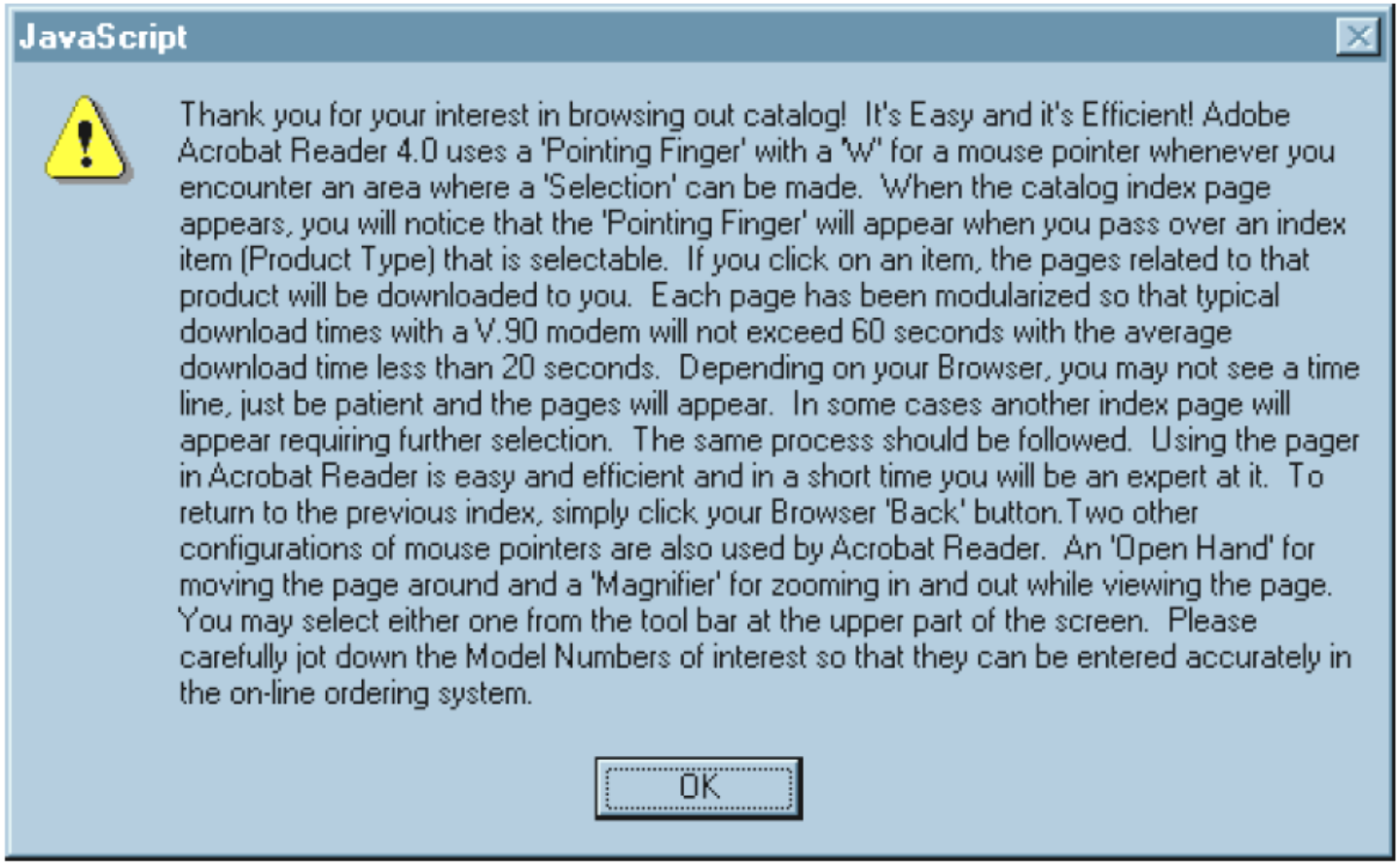
(Charge de travail)

- Richesse du contenu, mais certainement pas idéal.

(Charge de travail)

Densité informationelle

- Un message que l'on a pas trop envie de lire…
  - Éviter les textes trop verbeux (dialogue simple, phrases courtes).



Signifiance des codes et dénominations

- La **signifiance des codes et dénominations** caractérise l'adéquation entre l'objet, l'information ou le comportement présentés par l'interface et son référent (l'objet ou l'action qu'il représente).
- Il ne suffit pas de présenter un message ou un symbole à l'utilisateur, encore faut-il qu'il soit compréhensible pour lui.
- Des codes, dénominations et comportements *« signifiants »* disposent d'une relation sémantique forte avec leur référent.
- Dans une application professionnelle ou spécialisée, il est essentiel de s'imprégner du vocabulaire des utilisateurs afin de concevoir une interface *signifiante* pour eux.

Recommandations :

- Éviter les termes techniques (jargon), parler le langage de l'utilisateur
- Reproduire le comportement habituel des objets (par ex. la gomme)
- Définir explicitement et respecter les règles d'abréviation
- Prendre en compte les standards en vigueur (standards formels ou de facto) pour toutes les dénominations

Signifiance des codes et dénominations

- Billet de loterie virtuel que l'on gratte avec le curseur.

Check-list [1]

Critère	Question
Compatibilité	- Le logiciel correspond-il au contexte d'utilisation ? - Est-il adapté au profil des utilisateurs visés ? - Le vocabulaire de l'interface est-il celui employé par les utilisateurs ? - Les informations sont-elles présentées de manière cohérente par rapport aux autres supports de travail ? - L'accès aux commandes est-il adapté au contexte de réalisation de la tâche ?
Guidage	- L'utilisateur est-il assisté dans la façon de se servir du logiciel (en fournissant par exemple le format de saisie des données, une liste des valeurs possibles, etc.) ? - Une aide en ligne est-elle proposée ? - La documentation est-elle claire ?
- Incitation	- L'utilisateur est-il amené à effectuer des actions spécifiques ?
- Groupement / Distinction	- Les informations de même type sont-elles regroupées ? - Distingue-t-on les données différentes ?
- Feedback immédiat	- Le système fournit-il un retour aux actions de l'utilisateur ? - Les opérations réalisées par le système sont-elles perceptibles ?
- Lisibilité	- Les informations sont-elles correctement lisibles et interprétables ?

Check-list [2]

Critère	Question
Homogénéité **Cohérence**	- L'agencement des fenêtres est-il semblable (gabarit, tracé régulateur) ? - Les couleurs, les icônes, les éléments graphiques et les polices de caractères sont-ils utilisés de façon cohérente ? - Les formats de présentation des données sont-ils constants ? - Un vocabulaire uniforme est-il utilisé dans l'ensemble des fenêtres ? - Le fonctionnement de la souris est-il cohérent ? - Le logiciel est-il cohérent du point de vue de son comportement ?
Adaptabilité ***- Flexibilité***	- Différents moyens sont-ils offerts à l'utilisateur pour déclencher les mêmes commandes ? - Les commandes sont-elles également accessibles au clavier ?
- Prise en compte de l'expérience	- L'utilisateur peut-il paramétrer le logiciel selon ses préférences ? - Une alternative rapide est-elle proposée à l'utilisation des menus (par exemple des raccourcis clavier) ?
Contrôle explicite ***- Actions explicites*** ***- Contrôle utilisateur***	- Les fonctions sont-elles toujours explicitement activées par l'utilisateur ? - Peut-il quitter, abandonner facilement ou interrompre un traitement en cours ? - L'utilisateur peut-il revenir en arrière ? - L'utilisateur maîtrise-t-il tous les traitements réalisés par le système ?

Check-list [3]

Critère	Question
Gestion des erreurs	- Est-il possible d'explorer le logiciel sans risque ? - L'impact des erreurs est-il minimisé ?
- Protection	- Le système offre-t-il des moyens de prévenir des erreurs (boutons grisés, liste des valeurs possibles, affichage des unités, …) ? - L'utilisateur est-il prévenu rapidement de son erreur ? - L'utilisation du clavier est-elle minimale ? - L'utilisateur est-il averti lors d'opérations dangereuses ?
- Qualité des messages d'erreurs	- Les messages sont-ils bien visibles ? - La nature et les causes des erreurs sont-elles aisément identifiables ? - Les messages sont-ils explicites concernant les moyens de corriger l'erreur ?
- Correction	- Les erreurs peuvent-elles être facilement corrigées ? - Les éléments erronés sont-ils mis en évidence ? - Existe-t-il un moyen de récupérer des données détruites ?

Check-list [4]

Critère	Question
Charge de travail - ***Brièveté*** - ***Densité informationnelle***	- Le nombre d'opérations nécessaires pour effectuer les opérations est-il raisonnable (minimal) ? - La taille des listes affichées (menus, listes déroulantes, …) est-elle correcte ? - La longueur des libellés est-elle correcte (compréhensibles mais pas inutilement longs) ? - Des informations ou des fonctions inutiles sont-elles présentes dans l'interface ? - Les messages (d'information, d'erreur) sont-ils agréables à lire et compréhensibles ? - Des symboles pourraient-ils avantageusement remplacer des textes ? - L'utilisateur doit-il mémoriser des informations ?
Signifiance des codes et dénomination	- Les symboles, icônes et abréviations sont-ils compréhensibles par tous ? - Des métaphores connues sont-elles utilisées ? - Les standards en vigueur sont-ils respectés ? - Le vocabulaire des utilisateurs est-il utilisé ? - Y a-t-il un lien clair entre les éléments actifs de l'interface et les actions qu'ils déclenchent ?

Heuristiques de Nielsen

«Ten Usability Heuristics» Jakob Nielsen

Autres heuristiques

Les 10 heuristiques de Nielsen [1]

1 Visibilité de l'état du système

Le système devrait toujours tenir informé l'utilisateur de ce qui se passe, en fournissant un « retour » *(feedback)* approprié, dans un temps raisonnable.

2 Correspondance du système avec le monde réel

Le système devrait « parler » le langage de l'utilisateur, avec des mots, des phrases et des concepts qui lui sont familiers, plutôt que d'utiliser un langage propre au système. Suivre les conventions du monde réel, en faisant apparaître les informations dans une séquence naturelle et logique.

Les 10 heuristiques de Nielsen [2]

3 Liberté, contrôle de l'utilisateur

Par erreur, les utilisateurs choisissent parfois des fonctions du système et ils ont besoin d'une « sortie de secours », clairement libellée pour quitter la fonction non désirée, sans qu'il y ait besoin de passer par de multiples dialogues pour le faire. Le système doit permettre d'annuler/refaire *(undo/redo)* une action.

4 Cohérence et standards

L'utilisateur ne doit pas avoir à se poser des questions pour savoir si différents mots, situations ou actions signifient la même chose. Suivre les conventions liées à la plate-forme.

Les 10 heuristiques de Nielsen [3]

5 Prévention des erreurs

Au-delà de la conception de messages d'erreur clairs, il faudra en premier lieu être attentif à ce que le design permette de prévenir les problèmes que pourrait rencontrer l'utilisateur.

6 Reconnaître plutôt que se souvenir

Rendre visibles les objets, les actions et les options. L'utilisateur ne devrait pas avoir à se souvenir d'une information en passant d'une séquence de dialogue à une autre. Les instructions pour utiliser le système devraient être immédiatement visibles ou facilement accessibles, à chaque fois que l'utilisateur en a besoin.

Les 10 heuristiques de Nielsen [4]

7 Flexibilité dans l'utilisation

Les raccourcis - ignorés par des utilisateurs novices - permettent souvent d'accélérer les interactions pour les utilisateurs expérimentés. Ainsi le système peut convenir à la fois aux utilisateurs inexpérimentés et expérimentés. Autoriser les utilisateurs à personnaliser les actions récurrentes.

8 Esthétique et design minimaliste

Les dialogues ne devraient pas proposer d'informations qui ne sont pas pertinentes ou qui ne sont que rarement nécessaires. Chaque information placée dans un dialogue entre en concurrence avec les autres informations - et en particulier celles qui sont pertinentes - et diminue leur visibilité relative.

Les 10 heuristiques de Nielsen [5]

9 Faciliter l'identification, le diagnostic et la « récupération » des erreurs par l'utilisateur

Les messages d'erreur devraient être formulés en langage clair (pas de codes et de jargon), indiquer précisément le problème et suggérer une solution pour le résoudre.

10 Aide et documentation

Bien qu'il soit préférable que le système puisse être utilisé sans le recours à une documentation, il peut cependant être nécessaire de fournir de l'aide et de la documentation. Les informations de ce type devraient être faciles à trouver, centrées sur la tâche de l'utilisateur, indiquer concrètement les étapes à suivre et ne pas être trop longues.

Les 10 heuristiques de Nielsen [6]

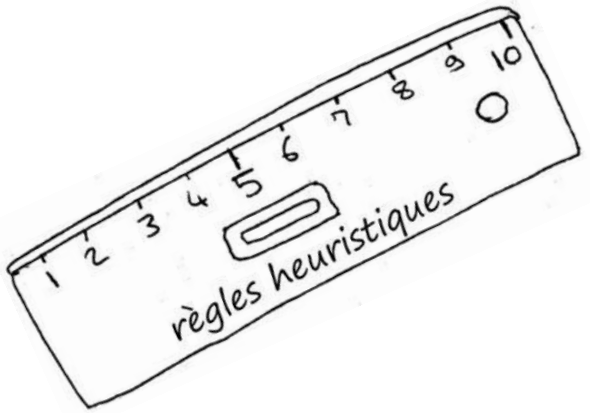
- Petite vidéo d'illustration des 10 heuristiques (en anglais) :

http://youtu.be/hWc0Fd2AS3s

Complément

Autres heuristiques [1]

- Il existes d'autres heuristiques avec un fort recouvrement et qui définissent des règles de base, que l'on peut rattacher aux principes suivants :
  - Facilité d'apprentissage
  - Simplicité
  - Visibilité
  - Contrôle Utilisateur et Liberté
  - Erreur
  - Efficience
  - Design graphique
- Parmi les heuristiques les plus courantes, on peut citer :
- **Les 4 règles de Norman**
  - Affordance
  - Mappage Naturel
  - Visibilité
  - Feedback



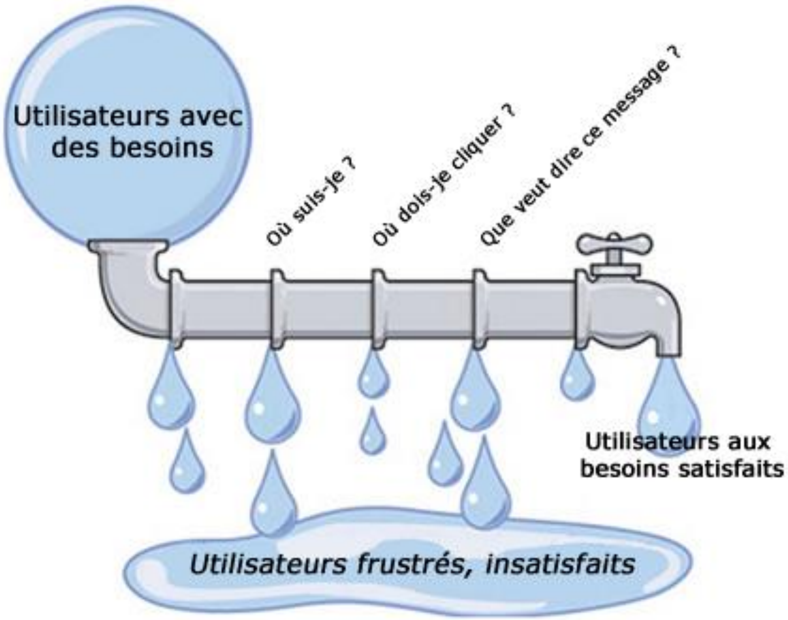
Autres heuristiques [2]

- **Les 8 règles de Shneiderman**
  - Faire un effort de cohérence interne (consistance)
  - Rendre possibles les raccourcis pour les usagers réguliers
  - Fournir des informations sur ce que fait le système (feedback)
  - Organiser le contenu et construire des dialogues avec une fin explicite (pour chaque étape)
  - Fournir un guidage et une rétroaction permettant d'éviter les erreurs
  - Rendre les actions réversibles (retours en arrière / *undo*)
  - Donner un sentiment de contrôle aux utilisateurs du système (c'est l'utilisateur qui est le maître)
  - Réduire la charge cognitive de la mémoire à court-terme (ne pas trop la solliciter)

Complément > > > < < <

Autres heuristiques [3]

- **Les 16 règles de Tognazzini**
  - Anticipation
  - Autonomie
  - Daltonisme
  - Consistance
  - Défauts
  - Efficience
  - Explorabilité
  - La loi de Fitts
  - Objets d'interface humain
  - Réduction de latence
  - Apprentissage
  - Métaphores
  - Protection du travail
  - Lisibilité
  - Retracer les états
  - Visibilité de la navigation



Complément