

### Les principaux domaines de l'informatique



... abordés dans le cadre de ce cours:



- La Programmation
- Les Systèmes d'Exploitation
- Les Systèmes d'Information
- I
  - La Conception d'Interfaces
  - Le Calcul Scientifique
  - Les Agents Logiciels



### Interfaces Humains-Machine (IHM)



Pourquoi faut-il s'intéresser à la conception et au développement des *Interfaces Humains-Machine* (IHM) ?

Une mauvaise interface peut provoquer le rejet de la part des utilisateurs, leur frustration, voire leur anxiété, face au système qu'ils ont à utiliser.

Inversement, une bonne IHM amplifie les sensations positves de succès et de contrôle.

Mais qu'est-ce qu'une bonne IHM?

D'une façon générale, une bonne IHM est une interface que l'utilisateur ne remarque plus!



### Objectifs de la conception des IHM (1)



# Concrètement, les *objectifs généraux* pour la conception d'une IHM sont:

(US Military Standards for Human Engineering Design Criteria)

- permettre la *réalisation* des tâches prévues,
- minimiser l'investissement nécessaire pour pouvoir utiliser l'interface
- garantir des interactions fiables
- favoriser la *standardisation*



### Objectifs de la conception des IHM (2)



#### • Permettre la réalisation des tâches prévues:

La **richesse fonctionnelle** doit être **adaptée**. Un ensemble insuffisant de fonctions rend un système inutilisable, quelque soit la qualité de sont interface.... mais une richesse fonctionnelle trop importante rend un système difficile à maîtriser.

Une analyse fonctionnelle doit être réalisée pour recenser l'ensemble des tâches et sous-tâches véritablement nécessaires, ainsi que leur fréquence d'utilisation.

#### • Minimiser l'investissement de l'utilisateur:

Minimiser la durée d'apprentissage et le niveau de compétences requis, ce qui peut s'obtenir, entre autre, par un mimétisme plus ou moins marqué avec d'autres IHM auxquelles l'utilisateur à déjà été confronté.



### Objectifs de la conception des IHM (3)



#### Garantir des interactions fiables:

Garantir un bon degré de fiabilité lors des interactions.

La confiance que place l'utilisateur dans le système est souvent fragile! Les interactions offertes par une bonne IHM doivent donc contribuer à **augmenter la confiance de l'utilisateur**: fonctionnement sans erreurs, organisation fonctionnelle claire et cohérente, stabilité dans le temps, ...

#### Favoriser la standardisation:

La standardisation permet de **réduire les temps d'apprentissage**, augmente la confiance et les performances des utilisateurs (moins d'erreurs) et **améliore** la **portabilité des systèmes**.



### **Domaines connexes**



Le développement d'IHM est une activité multidisciplinaire.

Il faut des specialistes:

- en *pychologie* et *facteurs humains* pour prendre en compte les concepts issus des *théories de la perception et de la cognition*;
- en *conception logicielle* pour utiliser au mieux les techniques informatiques disponibles;
- en *développement de matériel* pour mettre à profit les progrès dans le domaine de la conception de *nouveau périphériques*, et offrir un accès au système au plus grand nombre de personnes (handicapés compris)
- en *conception graphique* pour la fabrication des «layouts» qui seront utilisés dans le cas d'interfaces visuelles
- ..., en traitement de la parole, en ergonomie, ...



### Principales étapes de la conception



Les principales étapes de la conception d'une IHM sont:

- 1) **déterminer l'ensemble des tâches** que l'IHM devra permettre de réaliser: une bonne IHM est une IHM dont les objectifs fonctionnels sont clairement identifiés;
- 2) **déterminer les caractéristiques principales des utilisateurs** qui seront amenés à utiliser l'IHM (leur *profil*): la qualité d'une IHM est directement dépendante de sont adéquation avec la population d'utilisateurs pour laquelle elle est prévue;
- 3) **proposer plusieurs prototypes** d'interface qui seront discutés et évalués par les concepteurs et les **utilisateurs potentiels**: une bonne IHM naît le plus souvent de la diversité... et plusieurs pistes doivent donc être explorées;
- 4) **produire une spécification explicite** de l'IHM, décrivant à la fois les *contraintes fonctionnelles* et les *contraintes de layout*; un **manuel d'utilisation** et une **référence technique** pourront également être produits durant cette phase;
- 5) réaliser l'IHM proprement dite (phase d'implémentation effective);
- 6) **évaluer l'IHM** produite sur la base d'indicateurs reconnus (voir ci-après).



### Critères d'évaluation



Dans le but de *quantifier* (mesurer) la qualité d'une IHM, quelques *indicateurs* sont fréquemment employés:

#### 1. la durée d'apprentissage:

la durée moyenne nécessaire pour qu'un utilisateur typique maîtrise les fonctions pour lesquelles l'IHM a été développée;

#### 2. la rapidité d'exécution:

la durée moyenne de réalisation d'un ensemble test de tâches par un groupe d'utilisateurs de référence;

#### 3. le taux d'erreur:

le nombre **et la nature** des erreurs faites par le groupe d'utilisateurs de référence lors de la réalisation de l'ensemble test de tâches;

### 4. la mémorisation dans le temps:

c'est-à-dire l'évolution dans le temps des critères précédents (qu'en est-il après une heure d'utilisation, un jour, une semaine, ... ?);

#### 5. la satisfaction subjective:

la satisfaction subjective des utilisateurs pourra être mesurée par le biais de questionnaires ou d'interviews face-à-face.

Il n'est souvent **pas possible d'optimiser l'ensemble des critères** mentionnés ci-dessus et des **compromis** devront être faits selon les domaine d'application des IHM.



### **Domaines d'application (1)**



Les principaux domaines d'application pour les IHM se répartissent dans trois grandes catégories:

- ⇒ les systèmes critiques (généralement temps réel)
- ⇒ les systèmes commerciaux et industriels
- ⇒ les systèmes personnels et de loisirs



### **Domaines d'application (2)**



### • Les systèmes critiques:

(contrôle aérien, pilotage d'avions ou de centrales nucléaires, appareillage médical, ...)

Dans ce domaine, la *fiabilité* (faible taux d'erreur) et la *performance* (temps de réponse très court) sont centrales, y compris et en particulier dans des **conditions de stress** pour les utilisateurs.

Elles seront souvent obtenues au prix de durées d'apprentissage plus longues, et la mémorisation est garantie par des entrainements fréquents (drills)



### **Domaines d'application (3)**



### • Les systèmes commerciaux et industriels:

(applications de type transactionnels, dans le domaine des banques, assurances, gestion des stocks, comptabilité, réservations d'avions, de trains ou d'hôtels, de la vente, ...)

Dans ce domaine, le facteur déterminant est le coût.

La **formation des utilisateurs** coûte cher; la durée d'apprentissage devra donc être réduite... tout en garantissant cependant des **taux d'erreurs acceptables** (car les erreurs représentent également un coût, souvent mesurable).

La **rapidité d'exécution** est également importante, car elle conditionne le nombre d'opérations (ou de transactions) réalisées.

Ainsi, en 1988, une étude américaine a montré que, pour l'IHM du service de renseignement téléphonique d'un opérateur (équivalent du 111 en Suisse), une réduction de 0.8 sec de la durée moyenne des appels (15 sec) représentait une économie annuelle de l'ordre de 40 millions de dollars!



### **Domaines d'application (4)**



### • Les systèmes personnels et de loisirs:

(suites bureautiques, jeux, applications éducatives, systèmes d'exploitation, ...)

Dans ce domaine, la *satisfaction subjective* des utilisateurs est l'élément central car c'est souvent le critère déterminant pour le choix d'un système... et la concurrence est féroce.

La rapidité de l'apprentissage et de faibles taux d'erreurs seront également déterminants.

Une difficulté supplémentaire dans ce domaine est l'extrême variété des utilisateurs (du néophyte à l'expert confirmé), qui va souvent nécessiter une structure à plusieurs niveaux pour les interfaces (multilayered interfaces).



### Conception d'une IHM (1)



La conception d'une IHM peut **être guidée**, à des niveaux variés de généralité, par:

- 1. des modèles conceptuels de haut niveau
- 2. des principes généraux de conception
- 3. des recommandations pratiques spécifiques



# Conception d'une IHM (2)



Un exemple de modèle conceptuel:

le modèle syntaxique-sémantique «objets-actions»

L'idée de ce modèle est de postuler qu'une bonne description d'IHM doit distinguer entre:

- ⇒ le niveau sémantique,
  dont l'objectif est de décrire les concepts (objets et actions)
  nécessaires à la réalisation des tâches pour lesquelles l'IHM est conçue;
- ⇒ le niveau syntaxique,
  dont l'objectif est de décrire les séquences de commandes ou
  d'actions spécifiques qui devront être effectuées pour réaliser les tâches,
  avec un système donné.

Seul le *niveau syntaxique* est donc **dépendant** des **détails d'implémentation liés à un système particulier**.



### Conception d'une IHM (3)

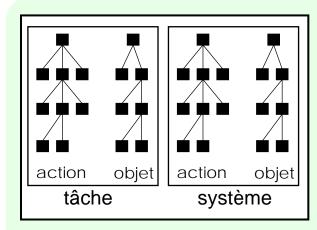


Le niveau sémantique est également décomposable en deux parties:

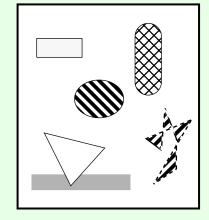
- ⇒ la description des concepts liés aux tâches;
- ⇒ la description des concepts liés à la réalisation de ces tâches par le biais d'un système informatique.

Il est important de bien noter la différence entre ces deux aspects, qui peuvent être très différemment maîtrisés par les utilisateurs un utilisateur peut en effet être un expert de la tâche et ne rien connaître à l'informatique ou inversement.

### Modèle conceptuel



niveau sémantique



niveau syntaxique



### Exemple de description de tâche (1)



Tâche: formatter un document

1. **Description conceptuelle** de la tâche dans le modèle sématique (ici sa **décomposition en sous-tâches**):

Formatter un document:

- 1. centrer le titre
- 2. mettre le titre en gras
- 3. justifier le corps du document

2. Description des objets et actions identifiés, sous la forme de concepts informatiques:

document = fichier manipulé par le traitement de texte utilisé.

Dans le traitement de texte utilisé:

- un *document* est décomposé en *paragraphes*, pouvant chacun être associé à des actions de formattage spécifiques;
- un paragraphe est constitué de *caractères*, pouvant chacun être également associé à des actions de formattage spécifiques;



### Exemple de description de tâche (2)



De (1) et (2), on peut obtenir la traduction informatique de la description conceptuelle:

- 1. appliquer l'action de formattage «centrer» au paragraphe contenant le titre;
- 2. appliquer l'action de formattage «mettre en gras» à l'ensemble des caractères du titre;
- 3. appliquer l'action de formattage «justifier» à l'ensemble des paragraphes du corps du document;
- 2. **Description syntaxique des actions identifiés**, dans le cadre d'un système et d'une IHM (typiquement graphique) donné:
  - Pour appliquer une action de formattage, il faut sélectionner les objects concernés, puis effectuer l'action.
  - Pour **selectionner les objets** (paragraphes ou caractères), il faut pointer le début de la zone d'écran concernéee à l'aide du curseur de la souris, appuyer et maintenir le bouton gauche [de la souris] tout en déplaçant le curseur vers la fin de la zone, et relâcher le bouton de la souris.
  - Pour **effectuer une action sur une zone**, il faut la sélectionner au sein des menus hiérarchiques (contextuels) apparaîssant après un *clic*-droit de la souris.



### Caractéristiques du niveau syntaxique



# Le *niveau syntaxique* est celui qui **pose le plus de problèmes** aux utilisateurs car:

- il est **fortement dépendant du système** utilisé, il faut donc recommencer par une phase d'apprentissage chaque fois que l'on change de système;
- il est difficile à décrire de façon structurée ou modulaire, et il est donc difficile pour l'utilisateur de s'en faire un modèle mental;
- il s'appuie en outre souvent sur des conventions arbitaires, qui doivent être mémorisées telles quelles.



### Principes généraux (1)



### Quelques principes généraux de conception d'IHM

#### 1. Connaître les utilisateurs

Une interface ne peut être satisfaisante pour tous les types d'utilisateurs. On distingue habituellement entre:

- les utilisateurs novices:
  - pas de connaissances syntaxiques,
  - peu de connaissances sur les concepts sémantiques et informatiques

La complexité (**richesse fonctionnelle**) de l'IHM doit être **limitée** à un sous-ensemble de tâches simples et fréquentes; l'aide en ligne doit être fortement développée et adaptée; les retours du système (confirmation d'action, messages d'erreur) doivent être fréquent et compréhensibles.

- les utilisateurs-experts débutants:
  - pas de connaissances syntaxiques,
  - bonne connaissance de la sémantique des tâches,
  - peu de connaissances sur les concepts informatiques

La richesse fonctionnelle peut être augmentée mais par le biais d'une approche syntaxique fortement structurée (par exemple des menus hiérarchiques)



### Principes généraux (2)



#### - les utilisateurs-experts intermittents:

- pas de connaissances syntaxiques,
- bonnes connaissances sur les concepts sémantiques et informatiques

Le problème est ici de pallier une mémorisation incomplète des éléments de syntaxe. La richesse fonctionnelle peut être maximale, mais une aide en ligne exhaustive, munie d'un système de recherche efficace, une aide contextuelle développée ainsi que des techniques de complétion automatique de commandes peuvent être utiles.

#### - les utilisateurs-experts intensifs (power user):

- bonnes connaissances syntaxiques,
- bonnes connaissances sur les concepts sémantiques et informatiques

Pour ces utilisateurs, l'élément primordial est la **rapidité de l'interaction**. Des courts-circuits (commandes abrégées) et des macros doivent être fournis, et les retours du système réduits au strict nécessaire.

Il est important de bien définir pour quelle classe d'utilisateurs une IHM est prévue. Si l'on veut une IHM utilisable par plusieurs classes d'utilisateurs, il faudra prévoir une **IHM à plusieurs niveaux**.



### Principes généraux (3)



#### 2. Définir les tâches

Il est non seulement important de bien **définir les tâches** pour lesquelles une IHM est conçue, mais aussi de **choisir correctement le niveau de granularité** de l'ensemble des fonctions qui seront accessible par le biais de l'IHM pour réaliser les tâches choisies.

Si le niveau de granularité est trop fin, l'utilisateur devra enchaîner un nombre trop important de fonctions pour réaliser une tâche. Inversement, si le niveau de granularité est trop grossier, l'interface devra proposer un nombre trop important de fonctions, ou l'utilisateur n'aura pas la possibilité de réaliser ses objectifs dans toutes leurs variétés.

Par ailleurs, la **fréquence moyenne d'utilisation** d'une fonction doit être **prise en compte** lors de son intégration dans l'IHM:

les fonctions fréquemment utilisées devront être également les plus faciles à mettre en œuvre, quitte à rendre des fonctions moins fréquentes un peu plus complexes.



### Principes généraux (4)



#### 3. Choisir le type d'interaction

Plusieurs types d'interaction sont envisageables pour la conception d'IHM. Parmi les plus courants, on peut citer:

- interaction à base de menus (éventuellement hiérarchiques);
- ⇒ interaction à base de formulaires ;
- interaction à l'aide d'un langage de commandes (éventuellement à l'aide du langage naturel);
- ⇒ manipulation directe (avions, voitures, réalité virtuelle, ...)

Les techniques à base de menus ou de formulaires permettent une bonne structuration de l'interaction et fournissent un bon support pour les utilisateurs novices ou débutants. Ils mènent cependant souvent à des IHM relativement **lentes**, et sont peu adaptées à des tâches complexes et des utilisateurs expérimentés.

Les techniques à base de langage de commandes sont plus complexes à mettre en œuvre (à la fois pour l'implémentation et pour l'utilisation) mais permettent une grande richesse fonctionnelle, et sont de ce fait bien adaptées aux utilisateurs-experts confirmés. Les durées d'apprentissage sont cependant plus élevées et leur mémorisation est plus délicate.



### Règles d'or (1)



Les huit *règles d'or* de la conception d'IHM:

#### 1. Assurer la cohérence

C'est la règle la plus souvent violée... alors qu'elle est normalement facile à respecter. Il s'agit par exemple d'assurer la **cohérence** *lexicale/terminologique* (i.e. les mêmes concepts sont systématiquement identifiés par les mêmes termes), et la **cohérence** *structurelle* (des actions composites similaires doivent pouvoir être réalisées par des séquences d'actions élémentaires similaires)

#### 2. Raccourcis dans l'interaction

Proposer des courts-circuits (commandes abrégés, macros, ...) pour les utilisateurs confirmés.

### 3. Retours système pertinents

Des retours système (system feedback) sont en particulier nécessaires pour confirmer le résultat de la réalisation des actions (ces retours doivent en règle générale être modestes pour des actions fréquentes et plus consistant pour des actions complexes ou rares). Des retours système (par exemple par le biais de changements dans la visualisation) sont également nécessaires pour signaler les modifications intervenues dans le système.



### Règles d'or (2)



#### 4. Identifier clairement la terminaison des actions

Ceci est particulièrement nécessaire dans le cas d'actions complexes, correspondant à un enchaînement de plusieurs actions atomiques (indivisibles), ou dans le cas d'actions critiques (sauvegardes, logout, ...).

#### 5. Offrir des mécanismes ergonomiques de correction des erreurs

Permettant en particulier à l'utilisateur de ne pas avoir à resaisir l'ensemble des paramètres d'une commande si l'un d'entre eux est erroné.

#### 6. Garantir la réversibilité des actions

Ce qui permet de réduire le *stress* de l'utilisateur lors de son interaction avec le système.



### Règles d'or (3)



#### 7. Augmenter le contrôle de la part de l'utilisateur

Ceci signifie en particulier d'éviter toute action «surprenante» de la part du système, la règle étant de préserver autant de possible la **causalité des actions**.

### 8. Réduire la surcharge de la mémoire à court terme des utilisateurs

La règle heuristique est que les utilisateurs stockent environ 7 éléments d'information dans leur mémoire à court terme. Des mécanismes doivent donc être disponibles pour réduire toute surcharge inutile: existence de pagination contrôlée, historique des commandes, ...



## Exemple de recommandations pratiques (1)



### Pour la présentation des données (display)

- ⇒ cohérence pour les labels et les conventions graphiques
- ⇒ standardisation des abréviations
- ⇒ cohérence du *layout* (structure des écrans produits)
- identification claire des écrans produits, permettant d'en déterminer la fonction, et de les référencer aisément
- ne présenter que les données pertinentes pour la tâche en cours
- utiliser des formats de représentation intuitif (par exemple graphiques) réduisant le besoin d'interprétation des données visualisées
- ne présenter des valeurs numériques que lorsque leur connaissance est effectivement nécessaire à l'utilisateur
- commencer par une conception monochrome pour le *display*, et n'ajouter les couleurs que progressivement, si elles apportent une aide effective à l'utilisateur
- ⇒ ...
- intégrer les utilisateurs dans le processus de conception des displays!



## Exemple de recommandations pratiques (2)



#### Pour attirer l'attention de l'utilisateur:

- ⇒ n'utilisez que 2 niveaux d'intensité
- ⇒ n'utiliser que 4 niveaux de taille
- ⇒ n'utiliser que 3 familles de caractères (fontes)
- ⇒ utilisez la vidéo inverse et le clignotement
- n'utilisez que 4 couleurs de base, en réservant les autres couleurs pour des usages occasionnels
- ⇒ en cas de clignotement d'éléments colorés, changez la couleur lors du clignotement
- **⇒** ...
- pour l'utilisation des sons, préférez les sons agréables pour les feedbacks positifs, et les sons désagréables pour les alarmes.