

**Mythe de l'erreur humaine**  $\mathcal{H}$  Les *échecs* d'un système PM sont souvent dus au **design**. Pour  $\downarrow$  erreur  $\mathcal{H}$  :

- ▷ Design qui tient compte des **limitations** et de la **fiabilité** des  $\mathcal{H}$ .

## Principes de design



- ▷ Utilisabilité
- ▷ Expérience de l'Utilisateur (UX)
- ▷ **Psychopathologie** : frustrations courantes
- ▷ Permettent de **critiquer**, **analyser** et **convervoir** interfaces.

## Causes d'échecs

- ▷ Fonctionnalité
  - ▶  $\mathcal{U}$  ne connaît pas fonctions de l'objet
  - ▶ L'objet ne fait pas ce que  $\mathcal{U}$  désire.
- ▷ Visibilité
  - ▶  $\mathcal{U}$  ne **voit** pas certaines infos de l'objet
  - ▶  $\mathcal{U}$  ne sait pas quelle séquence de contrôle est nécessaire pour atteindre son but.
  - ▶ E.g. Lumières enfoncée pour passage piétons
- ▷ Feedback
  - ▶ Comment  $\mathcal{U}$  sait si les opérations ont réussi ?
  - ▶ Comment  $\mathcal{U}$  sait s'il y a une erreur en cours de route ?

**Buts du UX** | MAUSSEE : Mémorabilité, Apprentissage, Utilité, Sécurité, Satisfaction, Efficience, Efficacité.

**Définition de l'utilisabilité** Degré selon lequel un produit peut être utilisé par des  $\mathcal{U}$  *identifiés*, pour atteindre des **buts définis** par l'**efficacité**, l'**efficience** et la **satisfaction**.

- ▷ **Efficacité** : atteindre le but 
- ▷ **Efficience** : *effort* et ou *temps* **minimal** 
- ▷ **Satisfaction** : évaluation subjective par  $\mathcal{U}$

## Où les designer se trompent

- ▷ Ne comprennent pas  $\mathcal{U}$  et leurs limitations
- ▷ Ne prévoient pas différents **contextes d'utilisation**
- ▷ Absence de **modèle détaillé** du fonct.
- ▷ Absence de **feedback** par l'objet.

**Pourquoi le design est-il difficile** Les **interactions** sont complexes et difficile à définir. Par ailleurs, les tâches sont complexes et *implicites*. Il faut distribuer *raisonnablement* les tâches à la machine et à l' $\mathcal{U}$  pour éviter que l'un ou l'autre ne soit pas confronté à une complexité excessive.

**Principe de découvrabilité** L' $\mathcal{U}$  doit savoir immédiatement à quoi l'objet sert, comment l'utiliser et quelles sont les opérations possibles.

- ▷ **Affordance** ce que l'O permet de faire. Un *signifiant* est un élément qui permet de rendre l'**affordance** visible.
- ▷ **Signifiants** indiquent que l'affordance  $\exists$  et ne doivent pas être **contradictoire**.
- ▷ **Anti-affordance** permettent de masquer visibilité d'un aff. et contribue à la gestion d'erreur.
- ▷ **Correspondance** Permet de faire l'association lors de l'utilisation (direction volume, mode on/off)