**ANALYSE DES BESOINS**

**CONCEPTION DES INTERFACES GRAPHIQUES**

Lors de la conception d’une application informatique, une des premières phases est celle de la conception de ses interfaces. Elle se fait à partir de l’analyse de l’expression des besoins (cahier des charges) et se doit d’y répondre fidèlement.

Deux types de questions doivent alors être traités :

### des questions de fond : quelles fonctionnalités doit assurer l’application pour répondre aux besoins exprimés, quelles données sont concernées, quels traitements sont réalisés ?

### des questions de forme : comment rendre l’utilisation de ces interfaces la plus confortable et donc efficace possible pour l’utilisateur.

#### I - Maquettage des fenêtres de dialogue

Les fenêtres de dialogue permettent à l’utilisateur de **saisir** et/ou **visualiser** des informations **affichées** (issues de la base, ou résultats de calcul).

Leur conception peut se faire en dessinant une **maquette**, qui doit mettre en évidence les **informations concernées**, ainsi que les **traitements** qui seront réalisés.

Réaliser la maquette d’une fenêtre de dialogue consiste à :

❶ Etudier son **contenu** :

⮚ Définir la liste de toutes les informations (données issues de la base de données) devant y figurer.

❷Définir **son rôle** :

⮚ Définir, pour chaque information, le traitement qui sera effectué : si elle est **affichée,** ou **saisie, ou affichée puis saisie** (pour les listes déroulantes).

D’où la nécessité d’une **légende** au bas de la maquette.

**❸** Organiser la mise en page :

⮚ **Ordonner et regrouper** les informations selon :

* leur centre d'intérêt par thème,
* ou un ordre logique de présentation, ou d’exploitation,
* ou leur côté fonctionnel, lisible et esthétique.

⮚ Choisir si nécessaire un **libellé pertinent** chaque information.

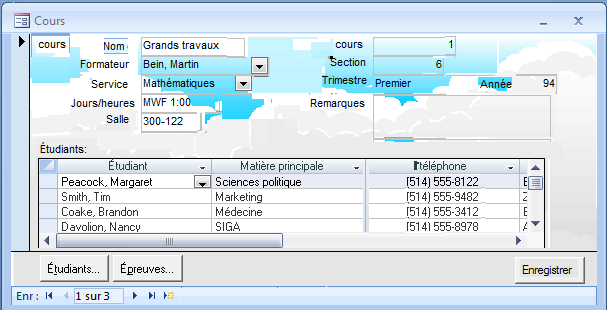
⮚ Utiliser les **composants graphiques**, couleurs, police … les plus adaptés répondant au mieux aux objectifs d’ergonomie, d’habitude, d’efficacité, d’esthétique, de lisibilité.

⮚ Ne pas oublier les **boutons de commande (ou liens)** permettant de passer à d’autres fenêtres ou d’exécuter des traitements, et conformes au diagramme d’enchainement.

**Faire valider la maquette auprès des utilisateurs.**

Exemple :

⮱Pour aboutir à la réalisation de l’écran de dialogue suivant (de l’application « Etud et Cours ») :



⮱ Le concepteur a dû se poser et répondre aux questions suivantes :

Que doit faire cet formulaire ? Quelles données y placer ? Quel traitement réaliser sur ces données ? Comment les représenter (liste, zone de texte, …) ? A quel endroit ? Comment les libeller ? ….,

⮱ Il aura alors réalisé la maquette suivante :

Rôle : visualiser et/ou modifier les informations d’un cours, ainsi que les étudiants inscrits à un cours.

**COURS** 

COURS – Nom : NomCours . Cours : RefCours .

Formateur : NomFormateur . Section : RefSection

Service : NomService . Trimestre : Trim . Année : Année

Jours & Heures : JoursEtHeures . Remarques :

Salle : NumSalle .

Etudiants :

|  |  |  |  |
| --- | --- | --- | --- |
|  | Etudiant | Matière principale | Téléphone |
|  | NomEtu,PrénomEtu | LibelléMatière | TélEtu |
|  | « | « | « |
|  | « | « | « |
|  | « | « | « |

Remarques

****

****

Enregistrer

Epreuves…

Etudiants…

****



Légende :

…… : affiché puis saisi

…… : affiché

Tout le reste : saisi ou affiché

#### II - Quelques notions générales d’ergonomie

D’une manière générale : « l’ergonomie regroupe les connaissances de physiologie, de psychologie et des sciences voisines appliquées au travail humain dans la perspective d’une meilleure adaptation à l’homme des méthodes, des moyens et des milieux de travail. »

Dans le domaine des logiciels, l’ergonomie s’intéresse au dialogue homme machine et tente donc d’apporter une meilleure adéquation aux besoins des utilisateurs.

#### 2.1 - Objectifs généraux

Rendre une interface ergonomique, c’est tendre vers les objectifs suivants :

⮚ La **concision** : il faut limiter le nombre d'interventions de l'utilisateur.

⮚ La **cohérence** : un même concept doit toujours être utilisé de façon similaire dans un contexte d'utilisation similaire dans toute l’application : répartition des zones, vocabulaire, couleur, mise en page, ordre des boutons, ordre des menus, …)

⮚ Le **retour d'information** : toute action de l'utilisateur doit amener un retour d'information rapide.

⮚ La **structuration des activités** : l'application doit être structurée selon une hiérarchie de niveaux de complexité croissante de façon à toujours proposer à l'utilisateur des fonctions simples.

⮚ La **gestion des erreurs** : l'utilisateur doit être orienté vers une méthode lui permettant de résoudre son problème au lieu de se contenter d'un simple message élémentaire. Les messages d’erreur doivent indiquer le lieu, la cause de l’erreur et le moyen de correction.

⮚ La **flexibilité** : toute application doit pouvoir être facilement personnalisée par les utilisateurs de façon à s'adapter à leurs propres habitudes et perception des choses.

#### 2.2 - Les principes généraux de conception

Quelques principes généraux peuvent être suivis pour atteindre ces objectifs :

**a) Au niveau du vocabulaire**

* Utiliser les termes de l’utilisateur.
* Respecter les métaphores (utilisées par l’utilisateur), pour l’aider à se créer une image mentale du thème traité.

*Ex :*  D*ocument, classeur, dossier, formulaire*

*Tout le vocabulaire de dessin : pinceau, gomme, palette; peindre …*

* Utiliser : des verbes pour les actions, des substantifs pour les rubriques.
* Eviter les mots à plusieurs sens, les synonymes, et les contre sens.

**b) Au niveau du dialogue**

* respecter **l’ordre des informations** à saisir, ou un ordre cohérent avec la tâche à réaliser.
* respecter **l’ordre des séquences**, si la tâche est découpée en séquences répétées.
* offrir une **hiérarchie de menus** si toutes les opérations sont trop nombreuses ou trop variées ; et dans ce cas :
* rendre l’accès au menu principal toujours possible
* pouvoir passer à un autre niveau sans revenir aux niveaux initiaux.

**c) Structuration des formulaires**

- garder la structuration de l’écran cohérent dans toute l’application, respecter les liens sémantiques entre les données.

* utiliser les attributs visuels (couleur, intensité, clignotement …) en respectant ceci :
* pas plus de 2 attributs simultanément, et pas plus de 2 intensités lumineuses
* pas plus de 4 couleurs (sauf si codage par couleur)
* respecter la signification des couleurs : rouge = erreur, vert = continuer, jaune = attention, et couleurs spécifiques à la tâche.
* pas de clignotement ininterruptible
* utiliser les cadres de groupage sémantique si besoin

**d) Favoriser une approche objet-action**

Approche objet-action : On présenta d'abord une liste d'objets sur lesquels on désire travailler, puis, pour chacun d’eux, la liste des actions que l'on peut réaliser.

Menu d’une approche Action-objet Menu d’une approche Objet Action







Consulter

Modifier

Supprimer

Client

Produit

Fournisseur







Client

Produit

Fournisseur

Consulter

Modifier

Supprimer

Moins adaptée, car le groupage des actions Mieux, car le groupage par objet est

n’est pas toujours clair pour l’utilisateur plus intuitif et naturel pour l’utilisateur

#### + lire le power point joint Règles\_Ergonomie

#### III - Le graphe d’enchaînement des écrans

Ce graphe donne une vision d’ensemble de l’application et de la structure générale du dialogue entre l’application et l‘utilisateur. Il met en évidence la **navigation entre les différentes fenêtres (menus et de dialogue) de l’application** et se présente sous la forme d’un graphe, où :

* chaque nœud et feuille correspond à une fenêtre,
* chaque flèche correspond à une navigation possible entre deux fenêtres.

*Menu Principal*

Cours Formateurs Classes *Menu Statistiques*

Epreuves Etudiants

Résultats

Etudiants et Classes

Emploi du temps

Consignes et conseils pour réaliser le graphe d’enchaînement des écrans :

- Vérifier que **toutes les fonctionnalités** de l’application sont bien couvertes.

- **Respecter la cohérence** **entre le graphe et le rôle de chacun des fenêtres :** l’arborescence entre les fenêtres dépend de leur contenu, et inversement.

* **Nommer** chaque écran et définir clairement son **rôle**.
* Faire la distinction, dans le graphe, entre les **fenêtres de menus et de dialogue**.

- Rester le plus **logique** et **conforme à la future utilisation** de l’application.

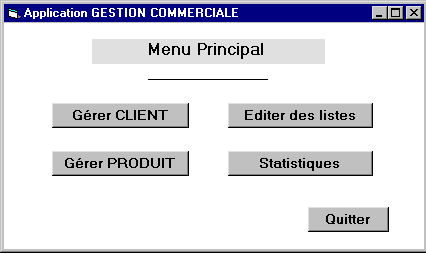
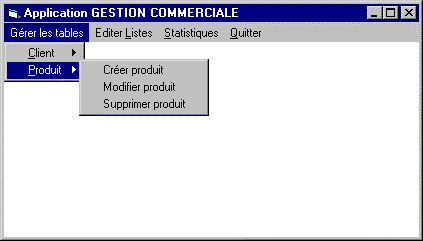
#### IV - Les menus

L’ensemble des options proposées par les différents menus de l’application doit :

* couvrir l’ensemble des fonctionnalités de l’application,
* **être hiérarchisées** (en menus et sous menus) pour faciliter leur accès par l’utilisateur,
* **être** conforme au graphe d’enchainement réalisé précédemment.

Lors de leur conception, il faut être vigilant à :

* l’ordre des menus, et de leurs options,
* le choix du vocabulaire : sa justesse, et la cohérence syntaxique
* leur ergonomie (voir partie II)

**Plusieurs présentations sont possibles :  *par boutons de commande*, *Menus déroulants...*

#### V - Cas D’utilisation de la méthode UML

La méthode UML (Unified Modeling Language) propose également des outils de conception d’interfaces.

#### 4.1 - Diagramme des cas d’utilisation

Ce **diagramme des cas d'utilisation** regroupe l'ensemble des cas d'utilisation du système, des acteurs et des relations entre eux.

Un cas d’utilisation correspond à une fonctionnalité, (qui elle-même pourra être réalisée par un ou plusieurs formulaires).

Un acteur correspond à un rôle joué par un utilisateur.

La relation entre un acteur et un cas d'utilisation est appelée **association** et correspond au fait que l'acteur déclenche et participe à un cas d'utilisation.

Formalisme  général :



*Exemple : Changer sa photo*



#### 4.2 - Description textuelle des cas d’utilisation

Chaque cas d’utilisation fait ensuite l’objet d’une description plus détaillée, sous forme textuelle, précisant les échanges précis entre l’acteur et l’application.

(Cette partie sera vue en SLAM3).