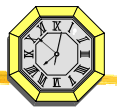
PARIS DESCARTES

ISI1 : Ingénierie des Systèmes d'Information

L3 MIAGE
Mme HEIWY
Septembre 2007 - Décembre 2007

PLAN DU COURS



RAPPELS SUR

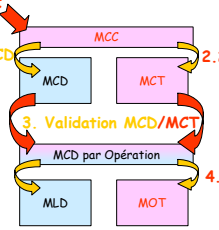
METHODE MERISE : Les principaux modèles

	DONNEES	TRAITEMENTS
Conceptuel	MCD: Modèle Conceptuel des Données — Propriétés, Entités, Relations	MCT: Modèle Conceptuel des Traitements — Processus, Evènements/ Résultats, Opérations, Synchronisations
Logique Organisationnel	MLD: Modèle Logique des Données — Relationnel, Tables, Champs	MOT: Modèle Organisationnel des Traitements — Procédures, Poste de travail, Tâches
Physique Opérationnel	MPD: Modèle Physique des Données — Base de données	MOPT: Modèle Opérationnel des Traitements — Programmes

LA METHODE MERISE : La démarche

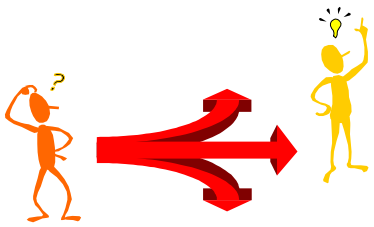
Plan

1. Construction du MCC
2.1 Construction du MCD
2.2 Construction du MCT
3. Validation MCD/MCT
4.1 Construction du MLD
4.2 Construction du MOT



Remarque:
MCC: Modèle Conceptuel de Communication

CONCEPTION DE SCHEMAS DE BASES DE DONNEES RELATIONNELLES



INTRODUCTION: *Problèmes soulevés par une mauvaise conception*

PROPRIETAIRE									
NV	MARQUE	TYPE	PUISS.	COUL.	NSS	NOM	PRENOM	DATE	PRIX
672 RH 75	Renault	Clio	6	Rouge	100	MARTIN	Jacques	10.02.95	10 000
800 AB 64	Peugeot	206	6	Verte	100	MARTIN	Jacques	11.06.90	50 000
686 HK 75	Citroen	Saxo	4	Bleue	200	DUPOND	Pierre	20.04.96	5 000
720 CD 60	Citroen	Xsara	10	Bleue	200	DUPOND	Pierre	20.08.90	20 000
400 XY 75	Renault	Kangoo	8	Verte	300	FANTOMAS	Yves	11.09.91	25 000

Une mauvaise conception des entités et des associations pose des problèmes.

Dans l'exemple ci-dessus, plusieurs types d'anomalies existent:

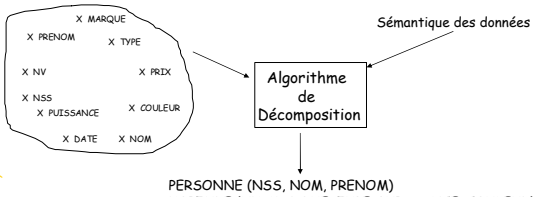
- Données **redondantes**, *par exemple MARTIN Jacques et DUPOND Pierre apparaissent 2 fois,*
- **Risque d'incohérences lors des mises à jour**, *par exemple, si on modifie le prénom de Monsieur Martin, il faut le faire à deux endroits,*
- Nécessité **d'autoriser les valeurs nulles**, *pour conserver par exemple des voitures sans propriétaire.*

L'APPROCHE PAR DECOMPOSITION

Introduction

Le Principe:

- A partir d'une relation composée de tous les attributs décomposer cette relation en sous-relations sans anomalie.
- Processus de raffinement successif devant aboutir à isoler les entités et associations du monde réel.



PERSONNE (NSS, NOM, PRENOM)
VOITURE (NV, MARQUE, TYPE, PUISSANCE, COULEUR)
POSSEDE (NSS, NV, DATE, PRIX)

L'APPROCHE PAR DECOMPOSITION

Notions préliminaires (1)

Projection

Opération consistant à supprimer des attributs d'une relation et à éliminer les tuples en double qui pourraient apparaître dans la nouvelle relation.

La projection de R de schéma $R(A_1, A_2, ..., A_n)$ sur les attributs $A_{i1}, A_{i2}, ..., A_{ip}$ (avec $_{ij<>ik}$) est une relation R'de schéma $R'(A_{i1}, A_{i2}, ..., A_{ip})$.

La projection R' est notée $\Pi\ A_{i1}, A_{i2}, ..., A_{ip}\ (R)$

Exemple de projection de la relation PROPRIETAIRE sur les attributs NOM et PRENOM:

PROPRIETAIRE	
NOM	PRENOM
MARTIN	Jacques
DUPOND	Pierre
FANTOMAS	Yves

L'APPROCHE PAR DECOMPOSITION

Notions préliminaires (2)

La jointure naturelle (Opération inverse de la projection)

La jointure naturelle de deux relations R et S de schéma respectifs $R(A_1, A_2, ..., A_n)$ et $S(B_1, B_2, ..., B_p)$ est une relation T ayant pour attributs l'union des attributs de R et S, soit :$\{A_1, A_2, ..., A_n\} \cup \{B_1, B_2, ..., B_p\}$ et pour tuples tous ceux obtenus par concaténation des tuples de R et S ayant mêmes valeurs pour les attributs de même nom. Ainsi, on a :

$\Pi\ A_1, A_2, ..., A_n\ (T)=R$

$\Pi\ B_1, B_2, ..., B_p\ (T)=S$

Exemple de jointure naturelle entre R et S:

R	
MARQUE	COUL.
Renault	Rouge
Peugeot	Verte
Citroen	Bleue
Renault	Verte

$\bowtie$

S	
COUL.	PUISS.
Rouge	6
Verte	6
Bleue	4
Bleue	10
Verte	8

=

R |><| S		
MARQUE	COUL.	PUISS.
Renault	Rouge	6
Peugeot	Verte	6
Peugeot	Verte	8
Citroen	Bleue	4
Citroen	Bleue	10
Renault	Verte	6
Renault	Verte	8

L'APPROCHE PAR DECOMPOSITION

Décomposition

Remplacement d'une relation $R(A_1, A_2, ..., A_n)$ par une collection de relations $R_1, R_2, ..., R_n$ obtenues par projections de R et telles que la relation résultat des jointures $R_1 \bowtie R_2 \bowtie ... \bowtie R_n$ ait même schéma que R.

Deux décompositions possibles de la relation VOITURE

Décomposition 1

VOITURE				
NV	MARQUE	TYPE	PUISS.	COUL.
120 GH 51	Renault	Clio	6	Rouge
100 CV 56	Renault	Clio	6	Verte

R1		
NV	TYPE	COUL.
120 GH 51	Clio	Rouge
100 CV 56	Clio	Verte

R2		
TYPE	MARQUE	PUISS.
Clio	Renault	6

Décomposition 2

V1	
NV	TYPE
120 GH 51	Clio
100 CV 56	Clio

V2		
TYPE	PUISS.	COUL.
Clio	6	Rouge
Clio	6	Verte

V3	
TYPE	MARQUE
Clio	Renault

L'APPROCHE PAR DECOMPOSITION

Décomposition sans perte

Décomposition d 'une relation R en $R_1, R_2, ..., R_n$ telle que pour toute extension de R, on ait : $R= R_1 \bowtie R_2 \bowtie ... \bowtie R_n$.

La conception d 'une base de données relationnelle peut se ramener à la décomposition de la relation universelle (composée de tous les attributs) en sous-relations sans anomalie pour obtenir une décomposition sans perte.

DECOMPOSITION
Exemple et Contre-Exemple

R1		
NV	TYPE	COUL.
120 GH 51	Clio	Rouge
100 CV 56	Clio	Verte

⋈

R2		
TYPE	MARQUE	PUISS.
Clio	Renault	6

⇓

Exemple

R1 ⋈ R2				
NV	MARQUE	TYPE	PUISS.	COUL.
120 GH 51	Renault	Clio	6	Rouge
100 CV 56	Renault	Clio	6	Verte

V1	
NV	TYPE
120 GH 51	Clio
100 CV 56	Clio

⋈

V2		
TYPE	PUISS.	COUL.
Clio	6	Rouge
Clio	6	Verte

⋈

V3	
TYPE	MARQUE
Clio	Renault

⇓

Contre-Exemple

V1 ⋈ V2 ⋈ V3				
NV	MARQUE	TYPE	PUISS.	COUL.
120 GH 51	Renault	Clio	6	Rouge
120 GH 51	Renault	Clio	6	Verte
100 CV 56	Renault	Clio	6	Rouge
100 CV 56	Renault	Clio	6	Verte

FONCTIONNELLES
Définition

Dépendance Fonctionnelle (DF)

Soit $R(A_1, A_2, ..., A_n)$ un schéma de relation, et X et Y des sous-ensembles de :$\{A_1, A_2, ..., A_n\}$; On dit que X -> Y (X détermine Y, ou Y dépend fonctionnellement de X) si pour toute extension r de R, pour tout tuple t1 et t2 de r, on a :

$\Pi X (t1) = \Pi X (t2) \Rightarrow \Pi Y (t1) = \Pi Y (t2)$

Plus simplement, un attribut (ou groupe d 'attributs) Y dépend fonctionnellement d 'un attribut (ou groupe d 'attributs) X, si, étant donné une valeur de X, il lui correspond une valeur unique de Y (et ceci quel que soit l 'instant considéré).

LES DEPENDANCES FONCTIONNELLES
Exemples

Par exemple, la relation VOITURE a les dépendances fonctionnelles:

NV -> COULEUR
TYPE -> MARQUE
TYPE -> PUISSANCE
(TYPE, MARQUE) -> PUISSANCE

Par contre ci-dessous, ce ne sont pas des dépendances fonctionnelles:

PUISSANCE ↛ TYPE
TYPE ↛ COULEUR

Remarque : **Pour trouver les dépendances fonctionnelles il faut utiliser la sémantique des attributs et non les quelques valeurs que l'on trouve dans la base.**

LES DEPENDANCES FONCTIONNELLES
Propriétés (1)

Les DF obéissent à plusieurs règles d'inférences triviales:

1. **Réflexivité** : $Y \subseteq X \Rightarrow X \rightarrow Y$; cette règle stipule que tout ensemble d 'attributs détermine lui-même ou une partie de lui-même.
   Exemple: *(TYPE, MARQUE) -> TYPE*
   (TYPE, MARQUE) -> MARQUE
2. **Augmentation** : X -> Y => XZ -> YZ; cette règle signifie que si X détermine Y, les deux ensembles d 'attributs peuvent être enrichis par une même troisième.
   Exemple: *NV -> COULEUR*
   NV, TYPE -> COULEUR, TYPE
3. **Transitivité** : X -> Y et Y-> Z => X->Z
   Par exemple, des dépendances: NV-> TYPE et TYPE ->PUISSANCE
   on déduit NV -> PUISSANCE

LES DEPENDANCES FONCTIONNELLES
Propriétés (2)

Autres règles : Les axiomes D 'Amstrong

4. **Union** : X -> Y et X -> Z => X -> YZ
5. **Pseudo-transitivité** : X -> Y et WY -> Z => WX -> Z
6. **Décomposition** : X -> Y et $Z \subseteq$ Y => X -> Z

Dépendance Fonctionnelle Élémentaire

Dépendance fonctionnelle de la forme X -> A, où A est un attribut unique non inclus dans X ($A \not\subset X$) et où il n 'existe pas X ' $\subset$ X tel que X ' -> A;

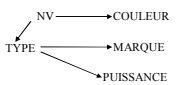
Remarque : **La seule règle d 'inférence qui s 'applique aux DF élémentaires est le Transitivité.**

DEPENDANCES FONCTIONNELLES

Graphe de dépendances fonctionnelles

Un graphe qui permet de visualiser les dépendances fonctionnelles élémentaires est appelé *graphe des dépendances fonctionnelles*.

Exemple1: Graphe des dépendances fonctionnelles de la relation VOITURE.
F = {NV -> TYPE; TYPE -> MARQUE; TYPE -> PUISSANCE; NV -> COULEUR}



Exemple2: Graphe des dépendances fonctionnelles de la relation CODE POSTAL (CODE, VILLE, RUE).
F = {(VILLE, RUE) -> CODE; CODE -> VILLE}

FERMETURE TRANSITIVE ET COUVERTURE MINIMALE

A partir d'un ensemble de DF élémentaires, on peut composer par transitivité d'autres DF élémentaires. On aboutit à la notion de fermeture transitive d'un ensemble F de DF élémentaires.

Fermeture Transitive

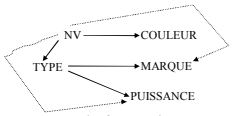
Ensemble des DF élémentaires considérées enrichi de toutes les DF élémentaires déduites par transitivité.

Exemple:

à partir de : F={NV -> TYPE; TYPE -> MARQUE; TYPE -> PUISSANCE; NV-> COULEUR}

on déduira :

F+ = F ∪ {NV-> MARQUE; NV-> PUISSANCE}



On définit l'**équivalence entre deux graphes de DF** par le fait qu'ils aient même fermeture transitive.

ET COUVERTURE MINIMALE

Couverture minimale

Ensemble F des DF élémentaires associé à un ensemble d 'attributs vérifiant les propriétés suivantes:

1. Aucune dépendance dans F n 'est redondante : c 'est à dire, pour toute DF f de F, F-f n 'est pas équivalent à F;
2. Tout DF élémentaire des attributs est dans la fermeture transitive de F (notée F^+).

Exemple:

F={NV -> TYPE; TYPE -> MARQUE; TYPE -> PUISSANCE; NV-> COULEUR}

est une couverture minimale pour l 'ensemble des DF de VOITURE.

Remarque: **La couverture minimale est essentielle pour décomposer les relations sans perte d 'informations.**

LA NOTION DE CLE

Clé d 'une relation

Sous-ensemble X des attributs d 'une relation $R(A_1, A_2, ..., A_n)$ tel que:

1. $X \rightarrow A_1, A_2, ..., A_n$
2. Il n 'existe pas de sous-ensemble $Y \subset X$ tel que $Y \rightarrow A_1, A_2, ..., A_n$.

Une clé est un ensemble minimum d 'attributs qui détermine tous les autres.

Exemple : *NV est une clé pour VOITURE.*
NV, TYPE n 'est pas une clé pour VOITURE.

Il peut y avoir plusieurs clés pour une relation, on en choisit une que l 'on appelle « **clé primaire** ».

NORMALES *La première forme normale*

Objectif :

1. Permettre une décomposition sans perte d 'informations à partir de la notion de DF;
2. Représenter des associations et des entités canoniques du monde réel.

Première Forme Normale

Une relation est en **première forme normale** si tout attribut contient une valeur atomique.

- La 1FN permet d 'obtenir des tables rectangulaires,
- Elle évite les domaines composés de plusieurs valeurs.

Exemple *: PERSONNE(NOM, PRENOMS) sera décomposée en*

PERSONNE1 (NOM, PRENOM1) et PERSONNE2 (NOM, PRENOM2).

NORMALES *La deuxième forme normale*

Deuxième Forme Normale

Une relation est en **deuxième forme normale** si et seulement si :

1. Elle est en première forme,
2. Tout attribut n 'appartenant pas à une clé ne dépend pas que d 'une partie de cette clé.

La 2FN permet d 'éliminer certaines redondances en garantissant qu'aucun attribut n'est déterminé seulement par une partie de la clé.

Exemple: *FOURNISSEUR (NOM, ADRESSE, ARTICLE, PRIX)*
La clé est le couple (NOM, ARTICLE).
on a les DF (NOM, ARTICLE) -> PRIX; NOM -> ADRESSE. Une partie de la clé (NOM) détermine un attribut n 'appartenant pas à la clé.
Cette relation n 'est donc pas en 2FN. Elle pourra être décomposée en deux relations:
FOURNISSEUR (NOM, ADRESSE); PRODUIT (NOM, ARTICLE, PRIX) qui sont en 2FN.

NORMALES *La troisième forme normale*

Troisième Forme Normale

Une relation R est en **troisième forme normale** si et seulement si:

1. Elle est en deuxième forme normale,
2. Tout attribut n'appartenant pas à une clé ne dépend pas d'un attribut non clé.

La troisième forme normale permet d'assurer l'élimination des redondances dues aux dépendances transitives.

Exemple: la relation VOITURE (NV, MARQUE, TYPE, PUISSANCE, COULEUR) n 'est pas en troisième forme normale puisque l 'attribut non clé TYPE détermine MARQUE et PUISSANCE.

Cette relation peut être décomposée en deux relations:

VOITURE (NV,TYPE, COULEUR) et MODELE (TYPE, MARQUE, PUISSANCE)

PROPRIETES D'UNE DECOMPOSITION EN 3FN

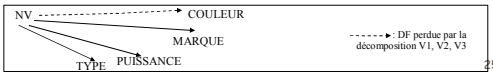
Décomposition préservant les dépendances fonctionnelles

Décomposition $\{R_1, R_2, ..., R_n\}$ d 'une relation R telle que la fermeture transitive des DF de R est la même que celle de l 'union des DF de $\{R_1, R_2, ..., R_n\}$.

Remarque : **Les dépendances fonctionnelles sont des règles indépendantes du temps que doivent vérifier les valeurs des attributs.**

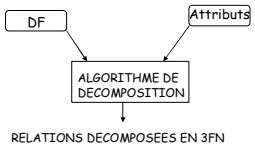
Exemple :

- *La décomposition de la relation VOITURE (NV, MARQUE, TYPE, PUISSANCE, COULEUR) en R1(NV, TYPE, COULEUR) et R2(TYPE, MARQUE, PUISSANCE) préserve les DF,*
- *alors que la décomposition en V1(NV, TYPE), V2(TYPE, PUISSANCE, COULEUR), V3(TYPE, MARQUE) ne les préserve pas.*



ALGORITHME DE DECOMPOSITION (1)

- Il existe au moins une décomposition en troisième forme normale préservant les DF et sans perte.
- Une telle décomposition s 'obtient par un algorithme ayant en entrée l 'ensemble des attributs ainsi que les DF.



ALGORITHME DE DECOMPOSITION (2)

La Procédure de décomposition ou normalisation

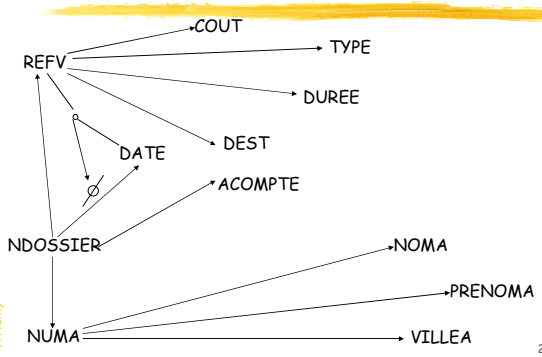
- Partir d 'un graphe de DF,
- « Editer les attributs isolés dans C ».
- **Réduire (C)**
- « Editer la relation composée de tous les attributs restants »;

La procédure **Réduire (C)**

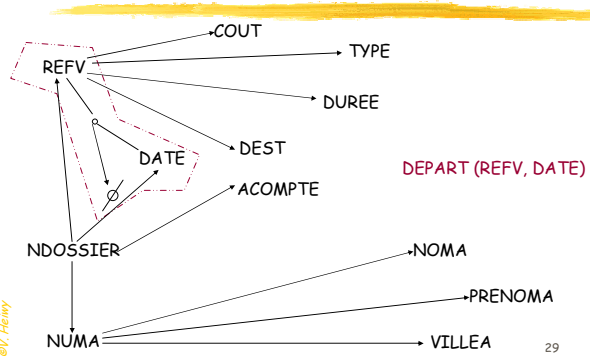
- Tant que « une DF n 'inclut pas tous les attributs » faire
  - « Rechercher le plus grand ensemble d 'attributs X tel que X -> A_1 , ..., X -> A_n »
  - « Editer la relation R<X, A_1, A_2, ..., A_n > »;
  - « Eliminer les DF figurant dans R de C »;
  - REDUIRE C
- fin tant que;

fin;

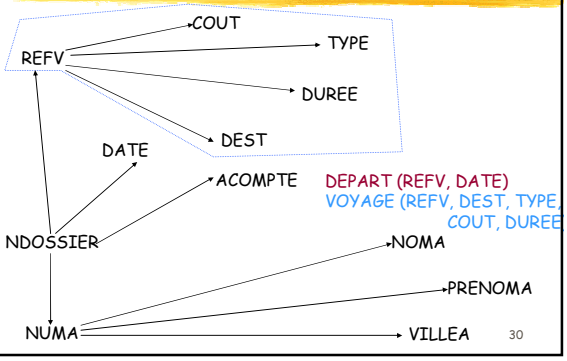
DECOMPOSITION: *Application à la base des* **Voyages (1)**



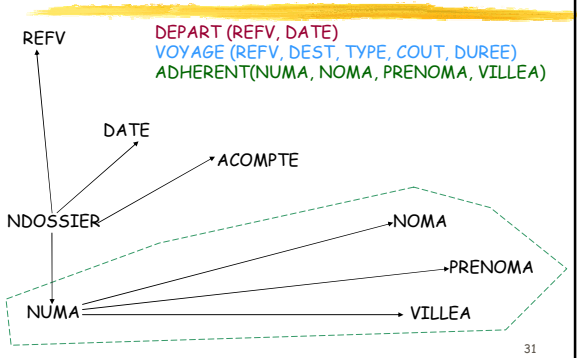
DECOMPOSITION: *Application à la base des* **Voyages (2)**



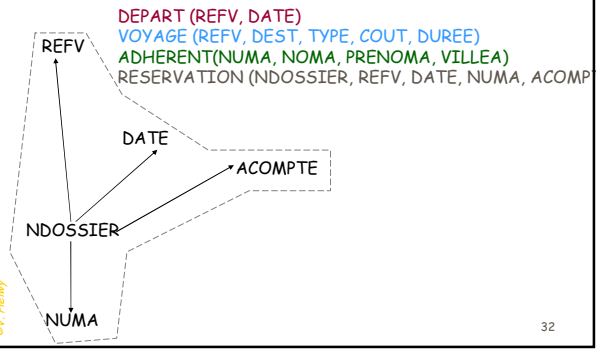
DECOMPOSITION: *Application à la base des* **Voyages (3)**



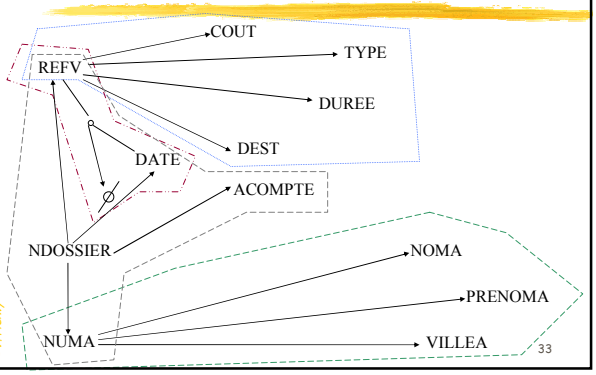
DECOMPOSITION:
Application à la base des **Voyages (4)**



ALGORITHME DE DECOMPOSITION:
Application à la base des **Voyages (5)**



DECOMPOSITION:
Application à la base des **Voyages (6)** Plan



LES EXTENSIONS MERISE: *Plan*

- **EXTENSIONS DES MODELES DE DONNEES**
  - **Généralisation et Spécialisation**
  - **Les contraintes d'intégrité fonctionnelles (CIF)**
  - **L'identification relative**
  - **Les Contraintes sur associations**
  - **Les Contraintes de stabilité**
  - **L'historisation**
  - **Passage au modèle logique**

LES EXTENSIONS MERISE
Généralisation et Spécialisation

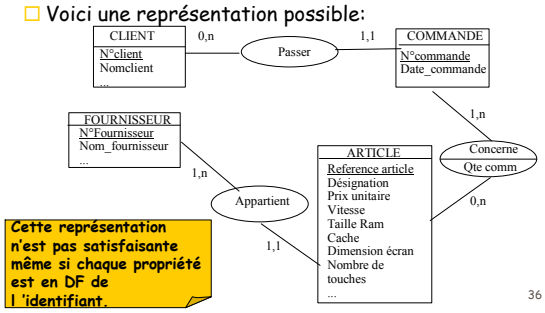
Exemple (1)

- Une entreprise vend du matériel informatique. Les articles vendus peuvent être des UC, des périphériques ou des combinaisons de plusieurs articles.
- Certaines **propriétés** définissant un article sont communes aux UC et aux périphériques : *la référence, le prix unitaire*, etc.
- Chaque type d'articles possède des caractéristiques propres. *La vitesse du processeur* pour l 'UC, *le nombre de touches* pour le clavier, etc.

EXTENSIONS:*Généralisation et Spécialisation*

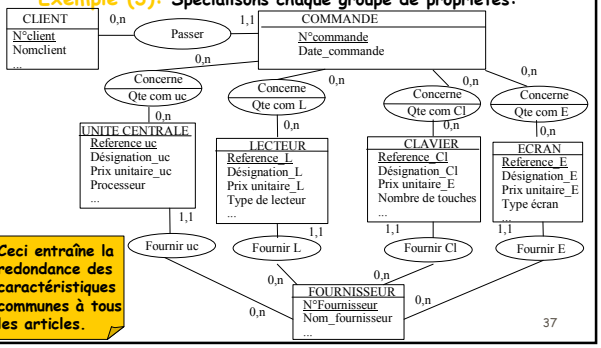
Exemple (2)

- Voici une représentation possible:



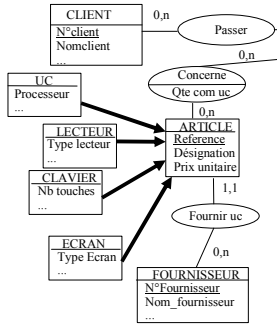
EXTENSIONS: *Généralisation et Spécialisation*

Exemple (3): Spécialisons chaque groupe de propriétés:



Ceci entraîne la redondance des caractéristiques communes à tous les articles.

EXTENSIONS: *Généralisation et Spécialisation*



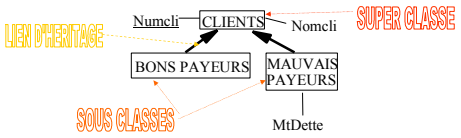
L 'introduction de la notion de sous-type permet de tenir compte du mécanisme d'**héritage**.

Ceci est obtenu est sortant de l'objet générique (le super-type) les propriétés spécifiques à chacun des sous-types et en ne conservant que les **propriétés communes**.

Les sous-types héritent du super-type, donc des propriétés communes.

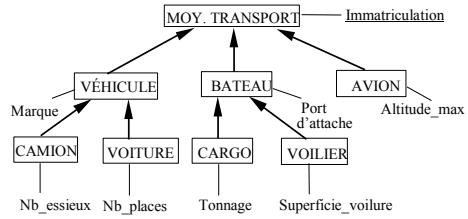
EXTENSIONS: *Généralisation et Spécialisation*

- **SOUS TYPE D'ENTITES**
  - LORSQU'UN SOUS ENSEMBLE D'ENTITES POSSEDE DES INFORMATIONS SPECIFIQUES, IL EST POSSIBLE DE CRÉER UNE SOUS CLASSE D'ENTITES;
  - TOUTES LES SOUS CLASSES **HERITENT DES ATTTRIBUTS** DE LA CLASSE MERE, ELLES PEUVENT AVOIR DES ATTRIBUTS SPECIFIQUES;
  - **L'IDENTIFIANT** DE LA CLASSE MERE EST AUSSI UN IDENTIFIANT DE SES SOUS-CLASSES.



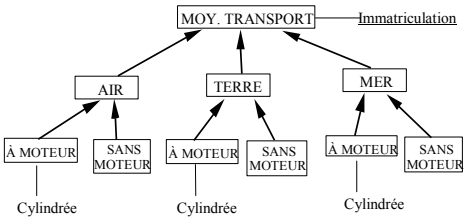
EXTENSIONS: *Généralisation et Spécialisation*

HIERARCHIE D'HERITAGE: EXEMPLE



EXTENSIONS: *Généralisation et Spécialisation*

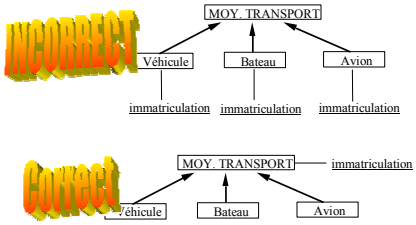
REDONDANCE DANS UNE HIERARCHIE D'HERITAGE



EXTENSIONS: *Généralisation et Spécialisation*

- **PROBLEMES LIES A LA CONCEPTION**

PLACEMENT D'UN ATTRIBUT DANS UNE HIERARCHIE DE SOUS CLASSES D'ENTITES

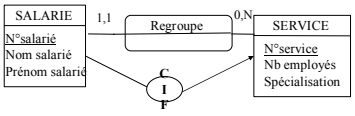


EXTENSIONS: LA CIF

- **CIF: CONTRAINTE D'INTEGRITE FONCTIONNELLE**

  ELLE EST DEFINIE SUR UNE ASSOCIATION ET REPRESENTE LE FAIT QUE L'UNE DES ENTITES DE SA COLLECTION EST IDENTIFIEE SANS AUCUN DOUTE PAR LA CONNAISSANCE D'UNE OU PLUSIEURS AUTRES.

  UNE ASSOCIATION BINAIRE AYANT DES CARDINALITES (0,1) OU (1,1) EST UNE CIF.

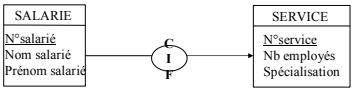


  DE LA CONNAISSANCE DU "N° SALARIE", ON PEUT DEDUIRE LE "N° DE SERVICE" AUQUEL IL APPARTIENT

EXTENSIONS: LA CIF

- **CIF: CONTRAINTE D'INTEGRITE FONCTIONNELLE**

  SI L'ASSOCIATION EST VIDE (i.e. N'A PAS DE PROPRIETES) ET QU'IL N'EXISTE PAS D'AUTRE ASSOCIATION ENTRE LES DEUX ENTITES, ON PEUT REMPLACER L'ASSOCIATION PAR UNE CIF.

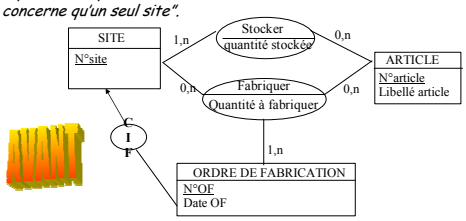


EXTENSIONS: LA CIF

- **CIF: CONTRAINTE D'INTEGRITE FONCTIONNELLE**

  LES CIF PERMETTENT DE SIMPLIFIER LES ASSOCIATIONS DE DIMENSION SUPERIEURE A 2.

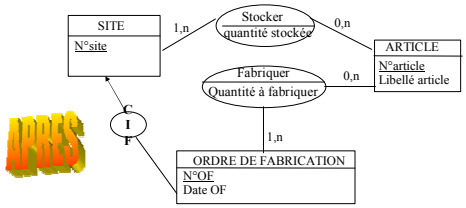
  Exemple: L'entreprise industrielle où: "Un ordre de fabrication ne concerne qu'un seul site".



EXTENSIONS: LA CIF

- **CIF: CONTRAINTE D'INTEGRITE FONCTIONNELLE**

  Exemple (suite): L'entreprise industrielle où: "Un ordre de fabrication ne concerne qu'un seul site".

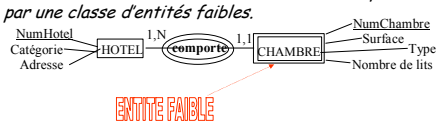


EXTENSIONS: *IDENTIFICATION RELATIVE*

- **CLASSES D'ENTITES FAIBLES**

  UNE CLASSE D'ENTITES FAIBLES EST UN ENSEMBLE D'ENTITES FAIBLES DE MEME TYPE DEFINI PAR RAPPORT A DES ENTITES DE MEME TYPE.

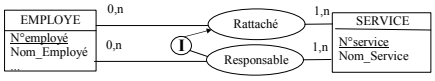
  Exemple: Dans un hôtel, une chambre est identifiée de manière unique par son numéro. Si l'on gère plusieurs hôtels, le n° de chambre ne suffit plus pour identifier de manière unique une chambre parmi toutes les chambres de tous les hôtels. La classe d'entité ***chambre*** *est alors représentée par une classe d'entités faibles.*



EXTENSIONS: *LES CONTRAINTES ENSEMBLISTES*

La contrainte d'inclusion

- ***La contrainte d'inclusion exprime que l'ensemble des occurrences d'une association est comprise dans l'ensemble des occurrences d'une autre.***

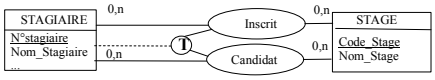


- Ici, les employés peuvent être rattachés à plusieurs services.
- Les services peuvent être mis sous la responsabilité d'un ou plusieurs employés.
- **Un employé ne peut être responsable que de services auxquels il appartient.**
- Tout couple (*employé, service*) participant à *Responsable* doit figurer parmi les couples (*employé, service*) de *Rattaché*.

EXTENSIONS: *LES CONTRAINTES ENSEMBLISTES*

La contrainte de totalité

- ***La totalité précise que toutes les occurrences d'une entité impliquée dans deux associations ou plus sont présentes dans au moins l'une d'entre elles.***

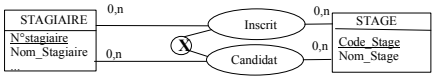


- Ici, l'entité STAGIAIRE est choisie comme pivot de la contrainte.
- Cette contrainte s'appelle aussi **OU Inclusif** ou **Couverture**.
- **La contrainte impose que STAGIAIRE participe au moins une fois à l'une des deux associations.**
- Le contrôle de couverture s'effectue par rapport au pivot, ici STAGIAIRE et non STAGE.

EXTENSIONS: *LES CONTRAINTES ENSEMBLISTES*

La contrainte d'exclusion (1)

- ***Elle interdit qu'une occurrence d'une entité impliquée dans deux associations ou plus soit présente dans 2 d'entre elles.***

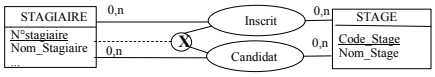


- Ici, un STAGIAIRE ne peut être à la fois candidat et inscrit dans une même formation.
- Le pivot implicite est le couple (*stagiaire, stage*), la vérification d'exclusion se fait donc sur les couples (*stagiaire, stage*) dont une occurrence ne peut se trouver à la fois dans *candidat* et dans *inscrit*.

EXTENSIONS: *LES CONTRAINTES ENSEMBLISTES*

La contrainte d'exclusion (2)

- ***C'est une contrainte symétrique, comme la contrainte de totalité.***
- ***Il est possible de préciser son pivot.***

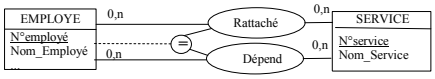


- Ici, un STAGIAIRE ne peut être à la fois candidat et inscrit quelque soit le stage concerné.
- **Règle de gestion**: un stagiaire est soit candidat à l'un quelconque des choix, soit inscrit, mais pas les deux à la fois.

EXTENSIONS: *LES CONTRAINTES ENSEMBLISTES*

La contrainte d'égalité

- ***C'est la combinaison de deux inclusions symétriques, l'ensemble des valeurs du pivot participant à une des associations contraintes devant être inclus dans l'ensemble des valeurs participant à l'autre et réciproquement.***

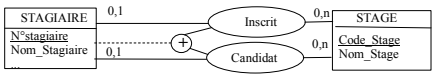


- Ici, l'ensemble des occurrence de *EMPLOYE* (le pivot) participant à l'association *Rattaché* doivent participer à l'association *Dépend* et réciproquement.

EXTENSIONS: *LES CONTRAINTES ENSEMBLISTES*

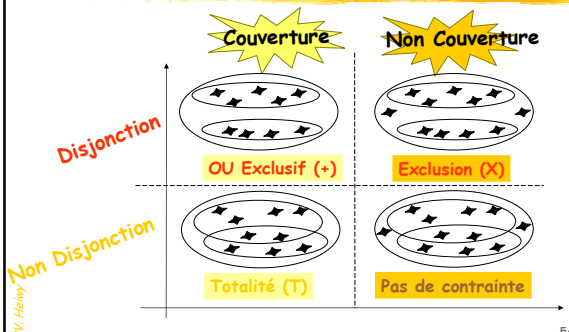
La contrainte de OU exclusif

- ***Elle combine une totalité et une exclusion et revient donc à vérifier que la jointure entre les relations impliquées soit vide.***



- **Totalité**: un *stagiaire* est au moins *candidat* ou *inscrit*.
- **Exclusion**: un *stagiaire* ne peut être à la fois *candidat* et *inscrit*.
- **OU exclusif**: un *stagiaire* est soit *candidat*, soit *inscrit* mais pas les deux à la fois.
- Remarque: Cette contrainte est parfois notée **O** au lieu de **+**.

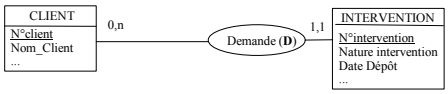
EXTENSIONS: *Bilan sur les contraintes ensemblistes*



EXTENSIONS: Les contraintes de stabilité

SUR UNE RELATION

- Une relation est dite permanente (P) ou définitive (D) si ces occurrences ne peuvent être ni modifiées ni détruites tant qu'existent les occurrences des objets qu'elle relie.



- Dès sa création, l'association *Demande* devient immuable. Une intervention ne pouvant exister que parce qu'elle est reliée à un client donné.

EXTENSIONS: Les contraintes de stabilité

SUR UNE PATTE

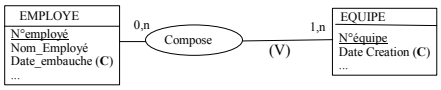
- On appelle contrainte de verrouillage (V) la contrainte de stabilité sur une patte. La patte est dite verrouillée si toutes les occurrences de l'association dans lesquels intervient une occurrence de l'entité doivent être créées en même temps que l'occurrence de cette entité.

- Cette *équipe* n'existe qu'en ce qu'elle est le regroupement de *personnes* précises. Si l 'on modifie sa composition, ce n'est plus la même équipe. On crée donc en même temps *N° équipe* et les occurrences de *Compose* dans laquelle il intervient. Après création ces occurrences ne sont plus modifiables.

EXTENSIONS: Les contraintes de stabilité

SUR UNE PROPRIETE

- Une propriété peut être déclarée constante (*C*). Etant donné une occurrence de l'entité ou de l'association qu'elle qualifie, la valeur correspondante de cette propriété ne peut être modifiée.
- L 'identifiant est par définition constant.



- *Date d'embauche* dans EMPLOYE, et
- *Date de création* pour EQUIPE ne peuvent varier au cours de la vie des entités qu'elles qualifient.

EXTENSIONS: MCD Analytique

MCDA = MCD Brut + contraintes + sous-types.

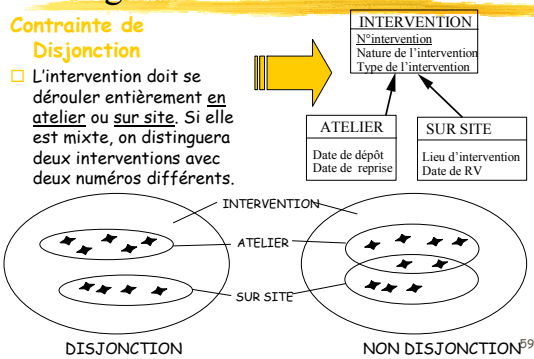
Contraintes sur l'héritage

- DISJONCTION: Une occurrence de l'entité générique appartient au plus à l'un des sous-types.
- COUVERTURE: Une occurrence du super-type appartient au moins à l 'un des sous-types.

EXTENSIONS: Contraintes sur l'héritage

Contrainte de Disjonction

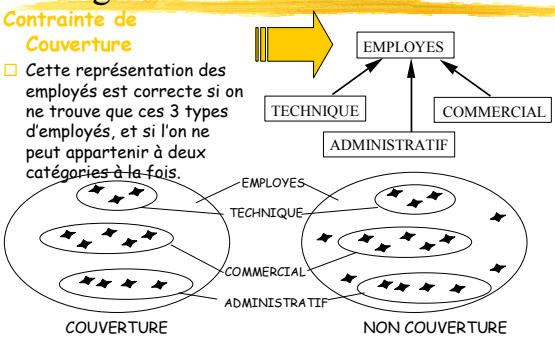
- L'intervention doit se dérouler entièrement en atelier ou sur site. Si elle est mixte, on distinguera deux interventions avec deux numéros différents.



EXTENSIONS: Contraintes sur l'héritage

Contrainte de Couverture

- Cette représentation des employés est correcte si on ne trouve que ces 3 types d'employés, et si l'on ne peut appartenir à deux catégories à la fois.

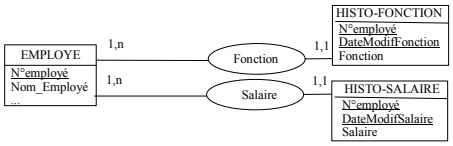


EXTENSIONS: Historisation

Objectif:

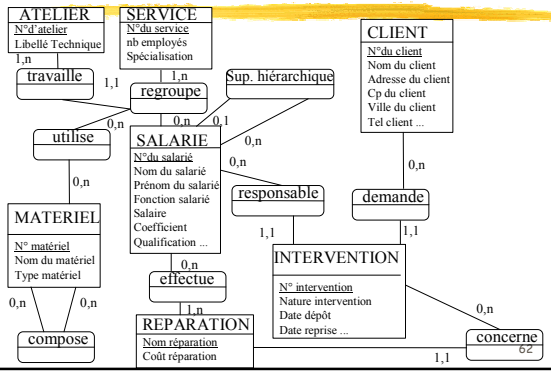
- Pouvoir conserver les différents états successifs d'une propriété non constante.

Exemple:

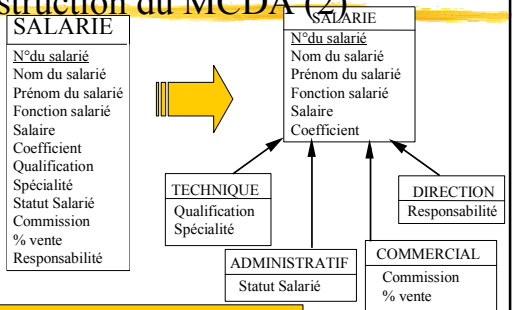


***Remarque**: Si la **fonction** et le **salaire** varient **simultanément**, on ne crée qu'une seule entité **Historisation**.*

EXTENSIONS: Exemple de construction du MCDA (1)

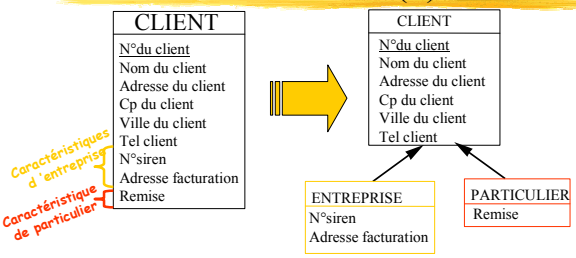


EXTENSIONS: Exemple de construction du MCDA (2)



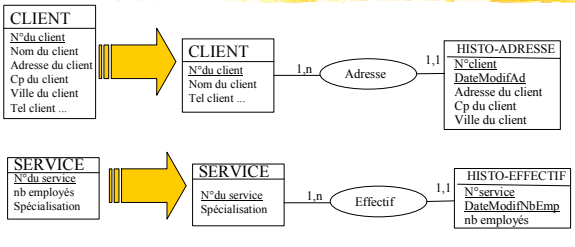
- Les 4 sous-types assurent la couverture totale.
- Aucun salarié ne peut appartenir à plus d'une catégorie
- Les 4 sous-types héritent des propriétés du super-type.

EXTENSIONS: Exemple de construction du MCDA (3)



- **Règle de couverture**: un client appartient obligatoirement à la catégorie ENTREPRISE ou PARTICULIER.
- **Règle de disjonction**: un client ne peut appartenir qu'à l'un ou l'autre des sous-types.

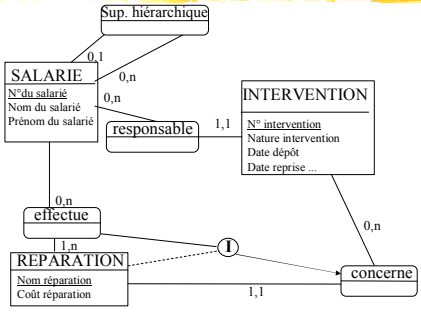
EXTENSIONS: Exemple de construction du MCDA (4)



IDEM pour SALARIE avec SALAIRE et FONCTION

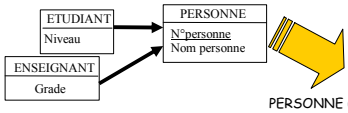
Après cela, on pose les contraintes ...

EXTENSIONS: Exemple de construction du MCDA (5)



EXTENSIONS: Passage au MLD (1)

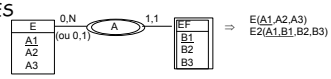
- **Règle 1**: Transformation des entités : **IDEM**
- **Règle 2**: Transformation des Associations à cardinalités multiples: **IDEM**
- **Règle 3**: Transformation des Associations comportant une cardinalité maximale à 1: **IDEM**
- **Règle 4**: Transformation des sous-types d'entité



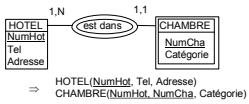
PERSONNE (N°PERSONNE, NOM PERSONNE)
ETUDIANT (N°PERSONNE, NIVEAU)
ENSEIGNANT (N°PERSONNE, GRADE)

EXTENSIONS: Passage au MLD (2)

- **Règle 5** : TRANSFORMATION DES CLASSES D'ENTITES FAIBLES

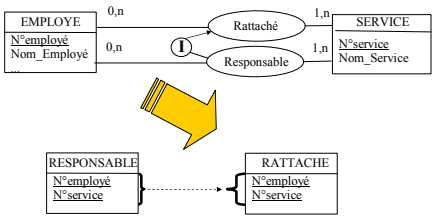


- EXEMPLE



EXTENSIONS: Passage au MLD (3)

- **Règle 6**: Prise en compte de la contrainte d'inclusion

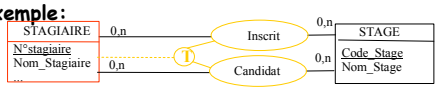


Le couple (N°employé, N°service) de la table RESPONSABLE ne pourra prendre que des valeurs également attribuées à ce couple dans la table RATTACHE.

EXTENSIONS: Passage au MLD (4)

- **Règle 7**: Prise en compte de la contrainte de totalité

  La vérification se fait en utilisant la projection et l'union.
- **Exemple:**



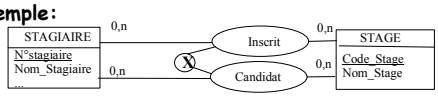
- Les ASSOCIATIONS en jeu sont Inscrit et Candidat
- Le Pivot est STAGIAIRE (clé: N° Stagiaire)

On vérifie que :

Project($T_{STAGIAIRE}$, N° Stagiaire) ⊂
∪(Project($T_{Inscrit}$, N° Stagiaire) , Project($T_{candidat}$, N° Stagiaire))

EXTENSIONS: Passage au MLD (5)

- **Règle 8**: Prise en compte de la contrainte d'exclusion
  - Les associations liées par une contrainte d'exclusion vont être transformées en relations.
  - L'intersection de la projection de ces relations sur leur clé doit être vide.
- **Exemple:**

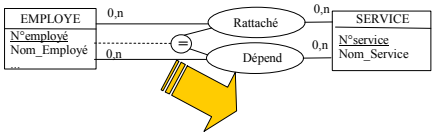


On vérifie que :

Project($T_{Inscrit}$, (N° Stagiaire, Code Stage))
∩ Project($T_{candidat}$, N° Stagiaire, Code Stage) = ∅

EXTENSIONS: Passage au MLD (6)

- **Règle 9**: Prise en compte de la contrainte d'égalité
  - Equivaut à une inclusion réciproque.
  - On vérifie l'inclusion dans les deux sens



- Project(T_{DEPEND}, N°Employé) ⊂ Project($T_{RATTACHE}$, N°Employé)

ET

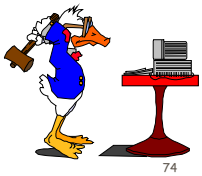
- Project($T_{RATTACHE}$, N°Employé) ⊂ Project(T_{DEPEND}, N°Employé)

EXTENSIONS: Conclusion

Plan

- Les extensions présentées rapprochent MERISE des modèles orientés objet
  - Spécialisation **Versus** Héritage
  - Entités faibles **Versus** ≈ Composition
- Les extensions sur les modèles des données ont été davantage intégrées que les extensions sur les traitements (cf. MEGA).
- MERISE OO versus UML…

LES IHM (INTERFACES HOMME-MACHINE): *Plan*



INTRODUCTION

- A quoi sert une interface?
  - A permettre l 'utilisation des fonctionnalités à des utilisateurs qui n 'ont pas forcément les connaissances informatiques du concepteur du logiciel;
  - A faciliter l'utilisation des fonctionnalités du logiciel.
- Que contient une interface graphique?
  - Un ensemble d 'objets graphiques (menu, boutons, fenêtres, etc.) permettant d 'accéder aux fonctionnalités de l 'application.
- Comment réaliser l'interface de la future application?
  - A partir du MOT et de la liste de ses procédures fonctionnelles (PF) => Menu général
  - PF interactive => Ecran de saisie (Formulaire)
  - PF automatique => Élément du menu (Editions, etc.)

CONSTRUCTION DE L 'INTERFACE

- LE **MOT** est composé d 'un ensemble de procédures fonctionnelles pouvant être
  - manuelles,
  - interactives,
  - automatiques.
- **Seules les procédures interactives et automatiques sont prise en compte pour la construction de l 'interface.**
- **L 'interface comprend**
  - Un menu général,
  - chaque option du menu général déclenche soit
    - l 'affichage d 'une fenêtre de saisie (***cas des PF interactives***); soit
    - L 'exécution d 'une ou plusieurs actions (macro, génération d 'un état, etc.; ***cas des PF automatiques***)

EXEMPLE 1 : La Société APL

- La Société APL (Association des Passionnés de la Lecture) regroupe des adhérents qui s 'engagent à commander au moins un livre par trimestre.
- APL édite et envoie à ses adhérents un catalogue chaque trimestre avec un bon de commande à remplir.
- APL traite les commandes et les règlements reçus et génère une facture pour chaque commande.
- Si la commande est incorrecte, une lettre de refus de la commande est envoyée à l 'adhérent.
- A la fin du trimestre, tous les adhérents qui n 'ont pas passé de commande se voient facturé le livre « sélection » du trimestre.
- Tous les adhérents en retard de règlement sont relancés, puis résiliés s 'ils ne régularisent pas leur situation dans le trimestre qui suit.

EXEMPLE APL: 1) LISTER LES PF DU MOT

N°PF	Nom	Type(I, M, A)	Poste de travail
1	Réception et mise en attente demande	M	Service adhérent
2	Détruire demande d'adhésion	M	Service adhérent
3	Transmission demande d'adhésion	M	Service adhérent
4	Enregistrer Demande d'adhésion	I	Service Informatique
5	Editer avis de confirmation	A	Ordinateur
6	Envoi catalogue	M	Service commercial
7	Reception commande	M	Cellule Commande
8	Transmission commande	M	Cellule Commande
9	Facturation	I	Service Informatique
10	Impression Facture	A	Ordinateur
11	Transmission Facture	M	Service Informatique
12	Edition lettre "Commande incorrecte"	I	Service Informatique
13	Impression lettre "Commande incorrecte"	A	Ordinateur
14	Elaboration Facture sélection	A	Ordinateur
15	Impression Facture sélection	A	Ordinateur
16	Préparation livres	M	Magasin
17	Expedition livres + facture	M	Magasin
18	Edition liste des mauvais payeurs	A	Ordinateur
18	Edition Relance "mauvais payeurs"	A	Ordinateur
18	Impression Relance "mauvais payeurs"	A	Ordinateur
19	Réception Règlement	M	Cellule Commande
20	Transmission règlement	M	Cellule Commande
21	Enregistrement règlement	I	Service Informatique
22	Résilier Adhérent	M	Service commercial
23	Transmission décision de resiliation	M	Service commercial
24	Résiliation (MAJ BD)	I	Service Informatique

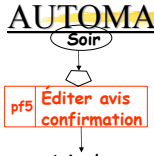
SELECTIONNER LES PF INTERACTIVES ET AUTOMATIQUES

N°PF	Nom	Type(I, M, A)	Poste de travail
4	Enregistrer Demande d'adhésion	I	Service Informatique
5	Editer avis de confirmation	A	Ordinateur
9	Facturation	I	Service Informatique
10	Impression Facture	A	Ordinateur
12	Edition lettre "Commande incorrecte"	I	Service Informatique
13	Impression lettre "Commande incorrecte"	A	Ordinateur
14	Elaboration Facture sélection	A	Ordinateur
15	Impression Facture sélection	A	Ordinateur
18	Edition liste des mauvais payeurs	A	Ordinateur
18	Edition Relance "mauvais payeurs"	A	Ordinateur
18	Impression Relance "mauvais payeurs"	A	Ordinateur
21	Enregistrement règlement	I	Service Informatique
24	Résiliation (MAJ BD)	I	Service Informatique

Chaque PF doit être décrite de manière détaillée.

79

EXEMPLE APL: 2) DESCRIPTION DETAILLE D 'UNE PF AUTOMATIQUE



Description PF N°5

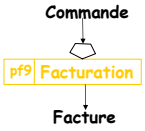
Éditer avis de confirmation

Nature:Automatique
Objet: Éditer pour chaque nouvel adhérent un avis de confirmation d 'adhésion.
Évènement en entrée: Soir
Données en entrée: Néant
Évènement en sortie: Néant
Données en sortie: Avis de confirmation d 'adhésion
Actions sur la base de donnée: Consultation des adhérents nouvellement saisis.

Description globale du déroulement de la PF
Sélection des adhérents dont la date d 'adhésion est la date du jour. Pour chacun d 'eux, composition et impression des avis de confirmation d 'adhésion.

80

EXEMPLE APL: 2) DESCRIPTION DETAILLE D 'UNE PF INTERACTIVE



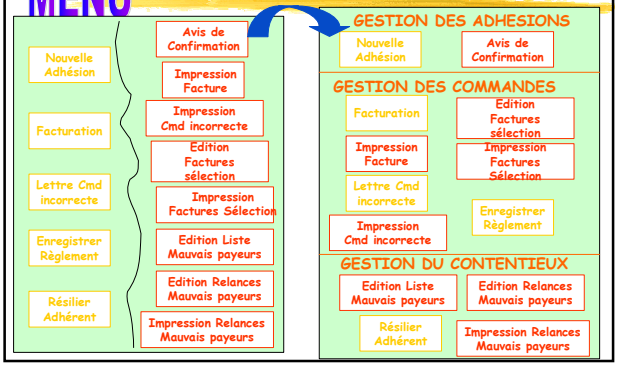
Description PF N°9 **Facturation**
Nature:Interactive
Objet: Enregistrer les informations d 'une commande pour générer une facture.
Évènement en entrée: Commande
Données en entrée: Bon de commande
Évènement en sortie: Néant
Données en sortie: Facture
Actions sur la base de donnée: Création de la FACTURE et des LIGNES-FACTURE associées.
Description globale du déroulement de la PF
Saisir le N° d 'adhérent de la facture, afficher pour contrôle les informations de cet adhérent (nom, prénom, adresse). Saisir les informations générales de cette facture (sa date) puis, pour chaque ligne de facture, saisir la référence du livre, afficher pour contrôle les informations de ces livres (titre et prix) puis saisir la quantité. Le total par ligne et le total général sont calculés automatiquement. Enregistrer dans la base de donnée les informations correspondantes (hors totaux).

81

CONSTRUIRE LE MENU GENERAL

N°PF	Nom
4	Enregistrer Demande d'adhésion
5	Editer avis de confirmation
9	Facturation
10	Impression Facture
12	Edition lettre "Commande incorrecte"
13	Impression lettre "Commande incorrecte"
14	Elaboration Facture sélection
15	Impression Facture sélection
18	Edition liste des mauvais payeurs
18	Edition Relance "mauvais payeurs"
18	Impression Relance "mauvais payeurs"
21	Enregistrement règlement
24	Résiliation (MAJ BD)

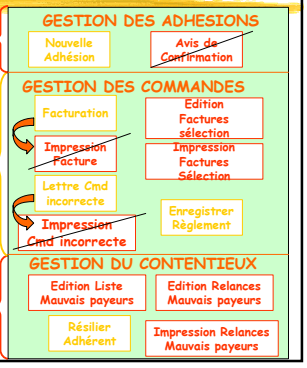
MENU GENERAL



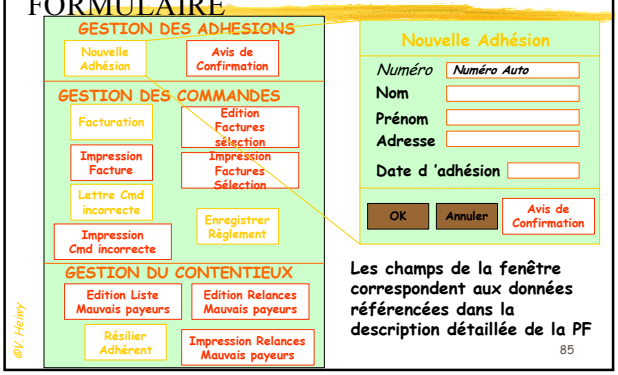
EXEMPLE APL : 3bis) AFFINER LE MENU GENERAL

MENU par thème

- **On peut regrouper les fonctionnalités par thème pour améliorer l 'ergonomie**
- **On peut également prendre en compte l 'enchaînement des procédures fonctionnelles présentées dans le MOT pour alléger le menu général**; par exemple ici en supprimant le bouton « impression facture » et en le plaçant dans la fenêtre de « facturation »; idem pour l '« Avis de confirmation » et la « commande incorrecte ».

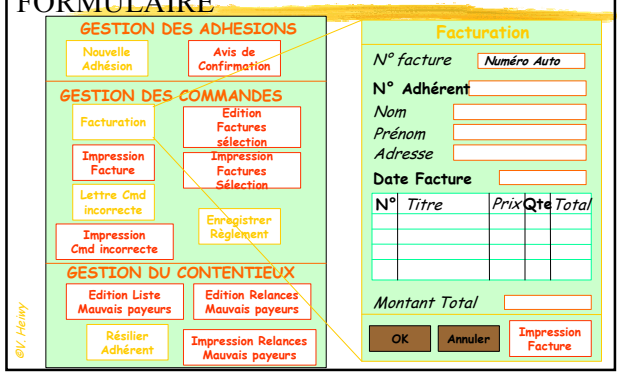


EXEMPLE APL : 4) A CHAQUE PF INTERACTIVE CORRESPOND UN FORMULAIRE

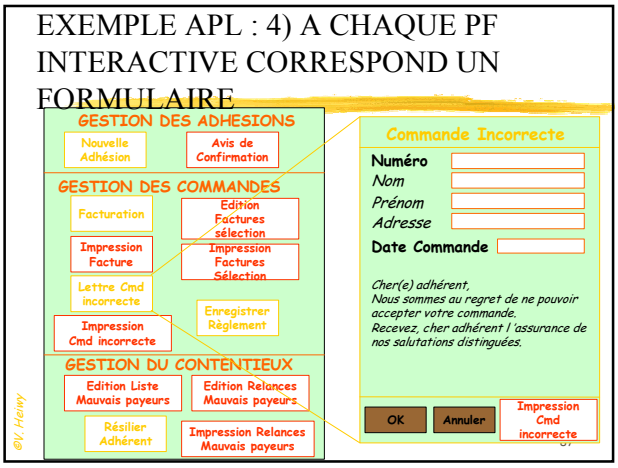


Les champs de la fenêtre correspondent aux données référencées dans la description détaillée de la PF

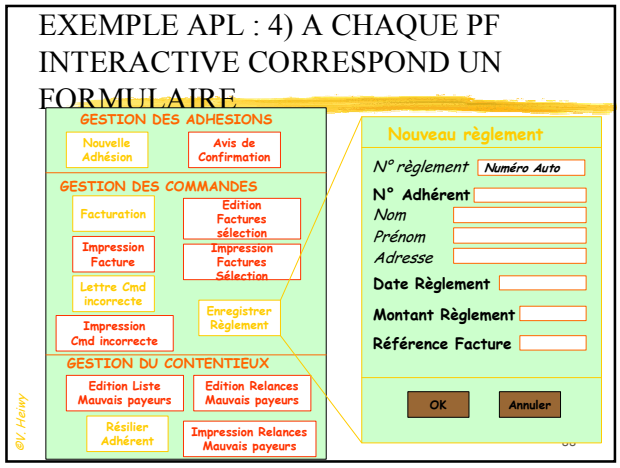
EXEMPLE APL : 4) A CHAQUE PF INTERACTIVE CORRESPOND UN FORMULAIRE



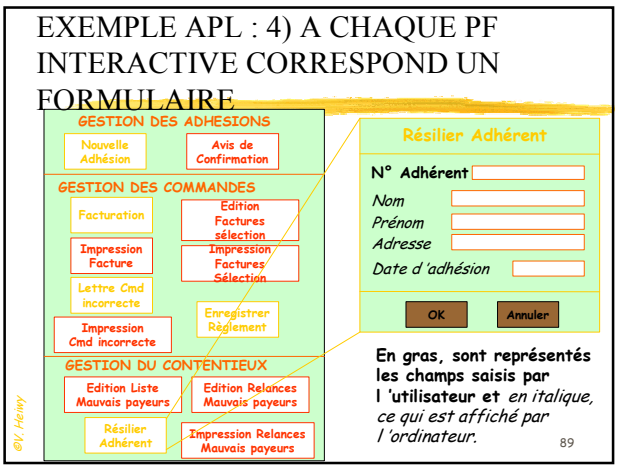
EXEMPLE APL : 4) A CHAQUE PF INTERACTIVE CORRESPOND UN FORMULAIRE



EXEMPLE APL : 4) A CHAQUE PF INTERACTIVE CORRESPOND UN FORMULAIRE



EXEMPLE APL : 4) A CHAQUE PF INTERACTIVE CORRESPOND UN FORMULAIRE



En gras, sont représentés les champs saisis par l 'utilisateur et *en italique, ce qui est affiché par l 'ordinateur.*

AUTOMATIQUE CORRESPONDENT DES ACTIONS

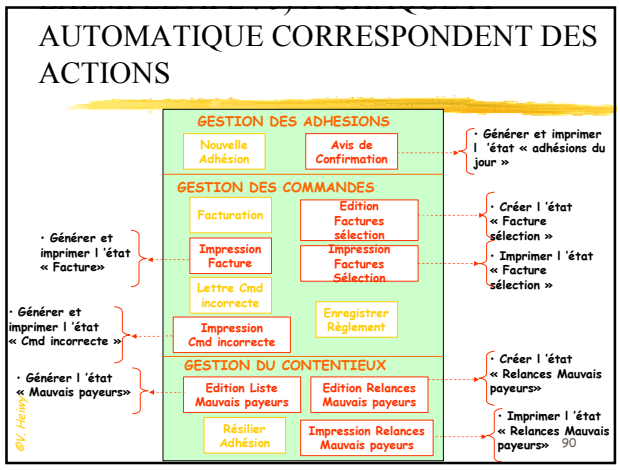
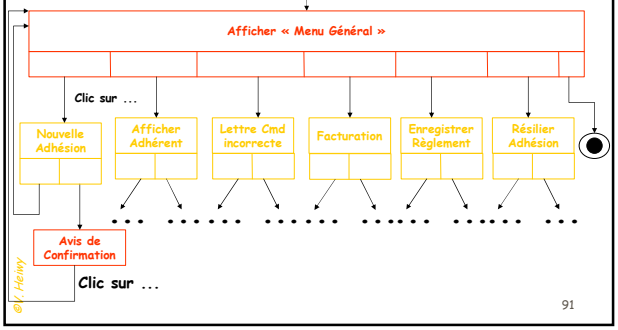


DIAGRAMME D ' ETATS POUR L'ENCHAINEMENT DES FENETRES



QUELQUES PRINCIPES D 'ERGONOMIE

- **LA COHERENCE**: Une même concept doit toujours être utilisé de façon similaire dans un contexte d 'utilisation identique.
- **LA CONCISION**: Elle doit permettre de limiter le nombre d 'interventions de l 'utilisateur afin d 'éviter les erreurs (par exemple typographiques).
- **LE RETOUR D 'INFORMATIONS**: Toute action de l 'utilisateur doit amener un retour d 'information rapide et pertinent afin de lui permettre d 'analyser rapidement le nouvel état de l 'application.
- **LA STRUCTURATION DES ACTIVITES**: L 'application doit être décomposée suivant une hiérarchie de niveaux de complexité croissante afin de proposer à l 'utilisateur des fonctions simples.
- **LA FLEXIBILITE**: Toute application doit pouvoir être facilement personnalisée par les utilisateurs de façon à l 'adapter à leurs propres habitudes et perception des choses.
- **LA GESTION DES ERREURS**: L 'utilisateur doit être orienté vers une méthode lui permettant de résoudre son problème au lieu d 'afficher un message élémentaire.

COMMENT ASSURER LA COHERENCE?

- cohérence avec le **matériel**
  - Adaptation à l '**écran** (définitions et palettes de couleur),
  - au **clavier** (touches de fonctions, touches spéciales),
  - à la **souris** (nombre de boutons utilisables),
  - à la **carte son** (désactivation des messages sonores si aucune carte n 'est détectée).
- cohérence avec le **logiciel**
  - les modules nécessaires (runtime, dll) au bon fonctionnement de l 'application doivent être présents sur les ordinateurs d 'accueil.
- cohérence **intra-applications (interne)**
  - toutes les fenêtres et boites de dialogue doivent respecter des règles communes de présentation et d 'organisation. Les menus et raccourcis claviers doivent avoir la même signification tout au long de l 'application.
- cohérence **inter-applications (externe)**
  - respect des règles communes à toutes les applications concernant les présentations , les méthodes d 'interactions avec des objets identiques et les réponses fournies aux mêmes actions de l 'utilisateur.

COMMENT RENDRE L 'INTERFACE TRANSPARENTE? *(intuitive, visuelle, concise, explicative et flexible)*

- **Intuitive**
  - Utilisation instinctive avec un temps d 'apprentissage réduit,
  - Doit refléter le travail habituel de l 'utilisateur sans l 'ordinateur,
  - Utilisation judicieuse de la souris, des icônes, des légendes et commentaires.
- **Visuelle**
  - Affichage de menus d 'actions et de listes de données au lieu de champs de saisie.
  - Pré-remplir le champs de saisie avec une valeur qui servira d 'exemple de référence.
- **Concise**
  - Utiliser des abréviations pour limiter la saisie,
  - Utiliser des raccourcis clavier et icônes pour accélérer l 'exécution des commandes,
  - Utiliser des macro-commandes pour personnaliser l 'interface et définir plusieurs niveaux d 'abstraction,
  - Utiliser les copier-coller et les valeurs par défaut pour réutiliser les informations déjà saisies.

COMMENT RENDRE L 'INTERFACE TRANSPARENTE? *(intuitive, visuelle, concise, explicative et flexible)*

- **Explicative**
  - Proposer une aide (Aide ou « ? ») accessible depuis la barre d 'actions
  - Proposer une aide contextuelle accessible à partir de boutons de boite de dialogue ou de touches de fonctions (F1, ...).
  - Orienter l 'utilisateur vers une solution en cas d 'erreur détectée.
- **Flexible**
  - Doit s 'adapter à la configuration matérielle et logicielle automatiquement.
  - Doit pouvoir être personnalisée par l 'utilisateur (création de raccourcis-clavier, ajout d 'icônes dans les barres d 'outils, modification de valeurs par défaut, réorganisation de menus)
  - Plus l 'application s 'adresse à un grand nombre d 'utilisateurs, plus il faut lui permettre d 'être personnalisable.

CONCEPTION D 'INTERFACE GRAPHIQUE?

- Définition des différentes fenêtres (primaires, secondaires, boîtes de dialogue, boîtes de messages),
- Organisation des fenêtres primaires et secondaires
  - 3 règles
- Organisation des boites de dialogue et des contrôles qu 'elles contiennent
  - 10 règles
- logique d 'enchaînement des fenêtres et des évènements
  - Diagramme d 'états

GRAPHIQUE:
(1) Définition des différentes fenêtres

- **En liaison avec le MOT**
  - PF interactives
- **En respectant les principes d 'ergonomie**
  - énoncés précédemment
- **Fonctionnement des fenêtres**
  - Diagramme d 'états pour gérer les interactions ayant lieu à l 'intérieur d 'une fenêtre (réalisation des contrôles, affichage des messages d 'erreur).
- **Diagramme d 'états pour représenter les enchaînements entre les fenêtres**

(2) Organisation des fenêtres primaires et secondaires

- Règle 1: **Taille de la barre d 'actions** (elle ne doit occuper qu 'une seule ligne dans la fenêtre)
- Règle 2: **Taille des menus** (la barre des menus doit contenir 12 options au maximum et la barre d 'actions pas plus de trois niveaux)
- Règle 3: **Existence de la région client** (si elle n 'est pas utilisée, elle doit être supprimée)

CONCEPTION D 'INTERFACE GRAPHIQUE:
(3) Organisation des boîtes de dialogue et des contrôles

- Règle 4 : **Nombre de contrôles** (<30)
- Règle 5: **Taille et position des boîtes de dialogue** (minimum de surface et proche de l 'objet qui l 'a appelé sans le recouvrir)
- Règle 6: **Position et regroupement des contrôles** (par famille avec cadre de groupage, alignement des cadres et titres pour les contrôles sans légende)
- Règle 7: **Mise en Majuscule** (la 1ere lettre de chaque chaîne de caractères sauf pour les sigles et abréviations)
- Règle 8: **Valeurs initiales** (donner une valeur plausible chaque fois que c 'est possible)

CONCEPTION D 'INTERFACE GRAPHIQUE:
(3) Organisation des boîtes de dialogue et des contrôles

- Règle 9: **Types de Listbox** (listbox ou combo-box simples plutôt que déroulantes , visualisation de 3 éléments au moins sans scrolling)
- Règle 10: **Choix et disposition des boutons poussoirs** (au même endroit dans toutes les boites de dialogue et dans le même ordre)
- Règle 11: **Taille et structuration des champs de saisie** (adaptée au type et à la longueur des données à saisir)
- Règle 12: **Couleurs et polices** (homogénéité dans toute l 'application et avec les autres applications réalisées; *par exemple, utiliser les couleurs pour distinguer les champs à saisir des champs affichés par l 'application*)
- Règle 13: **Aide en ligne** (prévoir les fonctionnalités générales de l 'application et le fonctionnement de chaque boite de dialogue).

(4.1) logique d 'enchaînement des fenêtres

- Pour représenter les enchaînements, il faut prendre en compte
  - Les évènements (clic de souris, passage au dessus d 'un objet, etc.
  - L 'événement ouverture de la fenêtre: du code executant des instructions peut y être associé
  - Le découpage en modules (dans le cas d 'applications volumineuses)
  - Les paramètres transmis (lors de l 'ouverture d 'une fenêtre)

(4.2) modélisation de l 'enchaînement des fenêtres

- **Utilisation de scénarios pour décrire tous les cas d 'utilisation de l 'interface**
  - scénario normal
  - scénario d 'erreur
  - scénario d 'exception
- **Utilisation d 'un diagramme d 'états pour décrire l 'enchaînement des fenêtres, des boîtes de dialogue et des boîtes de messages**
  - Les **états** représentent l 'affichage d 'une fenêtre active et le travail de l 'utilisateur sur cette fenêtre. L 'état se termine par l 'appui sur une touche ou sur l 'activation d 'un bouton « fin de fenêtre ».
  - La **transition** prend en compte les données de la fenêtre correspondant à l 'ETAT déclenchant la transition, le traitement en fonction des activations fait apparaître une autre fenêtre.

RAPPEL SUR LE DIAGRAMME D 'ETATS

- LE DIAGRAMME D 'ETATS
- LES ETATS
- LES TRANSITIONS
- LES EVENEMENTS
- LES ACTIONS
- LES ACTIVITES
- EXEMPLES: APL et DAB

DIAGRAMME D 'ETATS

- **Le DIAGRAMME D 'ETATS** représente les traitements permettant de gérer le domaine étudié, les **TRAITEMENTS** (opérations) par rapport à des **ETATS** des classes (ici l 'interface).
- Il met en évidence **l'enchaînement des états** et fait apparaître **l'ordonnancement** de ces différents travaux.
- Il permet de représenter tous les états possibles ainsi que les **EVENEMENTS** provoquant les changements d'état.

DIAGRAMME D 'ETATS
Les états

- Un **ETAT** est une situation durable dans laquelle peut se trouver l 'interface (ou l 'une de ses fenêtres) et à laquelle on associe des règles et des activités particulières.
- Les objets de certaines classes n'ont qu'un seul état pendant toute la durée de leur existence.
- L'état est représenté par un rectangle aux angles arrondis.

Etat 1
activité i

- On passe d'un état à un autre suite à un événement.
- Tout évènement ne provoque pas un changement d'état.

DIAGRAMME D 'ETATS
Les Transitions

- Une TRANSITION est une relation entre deux états signifiant qu'un passage de l'un à l'autre est possible.
- Elle est orientée.
- Un EVENEMENT y est attaché et indique qu'un objet passe d'un état à un autre sous certaines conditions.
- La transition est représentée par une flèche entre deux états accompagnée du nom d'événement.

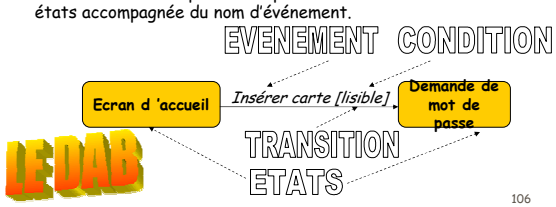


DIAGRAMME D 'ETATS
Les actions

- Une **ACTION** est une opération élémentaire et instantanée.
- Elle est rattachée à la réalisation d'un événement, ou à l'entrée dans un état, ou à la sortie d'un état.
- ***Exemple***: On associe à l'événement « cliquer sur annuler » l 'action « Effacer fenêtre facturation ».

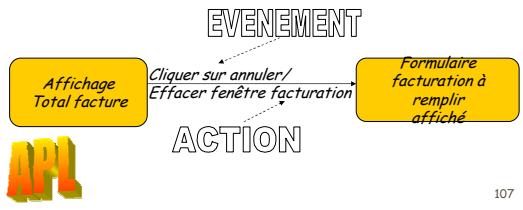


DIAGRAMME D 'ETATS
Les activités

- Une **ACTIVITE** est une action qui dure.
- Chaque activité est associée obligatoirement à un état d'objet et vice-versa.
- ***Exemple***: l'activité « imprimer reçu » est associée à l'état « terminé ».

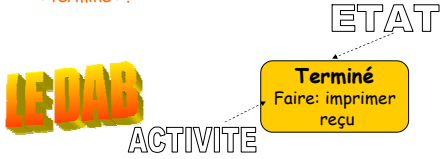
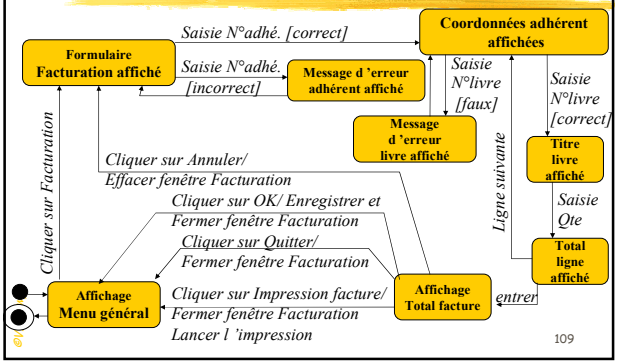


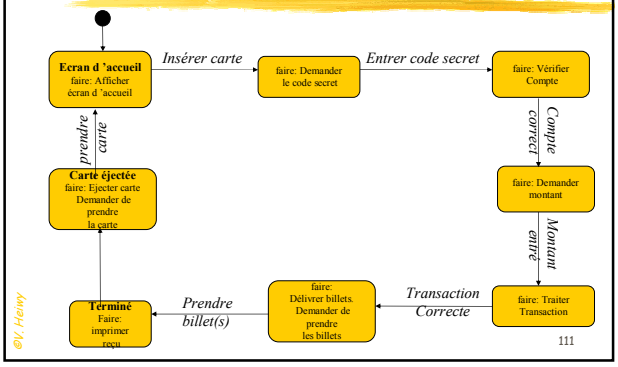
DIAGRAMME D 'ETATS
Exemple 1: APL- La facturation



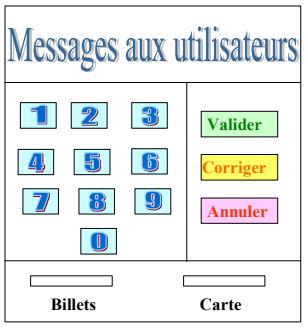
SCENARIO
Exemple 2: le DAB - cas normal

- Le DAB demande à l 'utilisateur d 'insérer la carte. L 'utilisateur insère la carte.
- Le DAB accepte la carte et lit le N° de série.
- Le DAB demande le code secret. L 'utilisateur entre « 1.2.3.4 ».
- Le DAB vérifie le N° de série et le code secret par l 'intermédiaire du consortium. Le consortium vérifie à l 'aide de la banque « 39 » et signifie son acceptation au DAB.
- Le DAB demande à l 'utilisateur de spécifier le montant de son retrait. L 'utilisateur indique 500.
- Le DAB vérifie que le montant ne dépasse pas la somme autorisée et transmet le traitement de la transaction au consortium. Le consortium fait suivre vers la banque qui retourne éventuellement un acquittement avec le nouveau solde.
- Le DAB délivre l 'argent et demande à l 'utilisateur de le prendre. L 'utilisateur le prend.
- Le DAB délivre le reçu, éjecte la carte et demande à l 'utilisateur de les récupérer. L 'utilisateur les récupère.

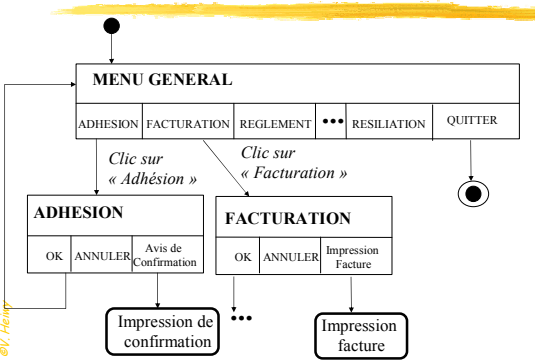
DIAGRAMME D 'ETATS
Exemple 2: le DAB - cas normal



L 'INTERFACE
Exemple 2: le DAB - cas normal



(4.2) Modélisation de l 'enchaînement des fenêtres



EXERCICES D 'APPLICATION
Création d 'interface

1 - Le DAB

- Définir sous forme de scénario un cas d 'erreur qui pourrait se présenter dans le déroulement d 'un « Retrait d 'argent ».
- Compléter le diagramme d 'états pour prendre en compte ce nouveau scénario.
- Comment modifier le diagramme d 'état représentant le fonctionnement du DAB pour prendre en compte le fait que l 'on puisse réaliser d 'autres opérations que des retraits (par exemple dépôt, virement, relevé)

2 - La bibliothèque

- A partir du MOT du cas bibliothèque, Définir sous forme de scénarios (cas normal, cas d 'exception, cas d 'erreur), puis sous la forme d 'un diagramme d 'états l 'interface du logiciel de gestion de la bibliothèque

Exemple 2: le DAB - Un cas d 'erreur

- Le DAB demande à l 'utilisateur d 'insérer la carte. L 'utilisateur insère la carte.
- Le DAB accepte la carte et lit le N° de série.
- Le DAB demande le code secret. L 'utilisateur entre « 9.9.9.9 ».
- Le DAB vérifie le N° de série et le code secret par l 'intermédiaire du consortium. Le consortium rejette la carte après consultation de la banque appropriée.
- Le DAB indique qu 'un mauvais code a été entré et demande à l 'utilisateur de l 'entrer à nouveau. L 'utilisateur entre « 1.2.3.4 » que le DAB vérifie avec succès.
- Le DAB demande à l 'utilisateur de spécifier le montant de son retrait. L 'utilisateur change d 'avis et appuie sur « Annuler ».
- Le DAB éjecte la carte et demande à l 'utilisateur de la récupérer. L 'utilisateur la retire.

Exemple 2: le DAB - incluant le cas d 'erreur

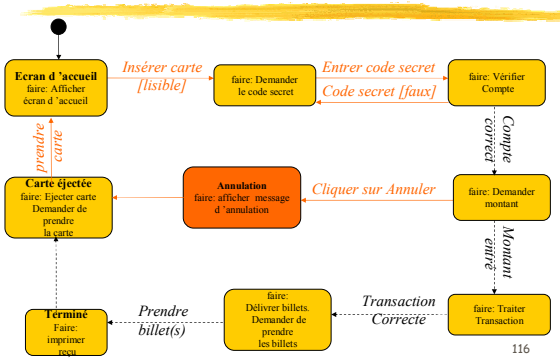
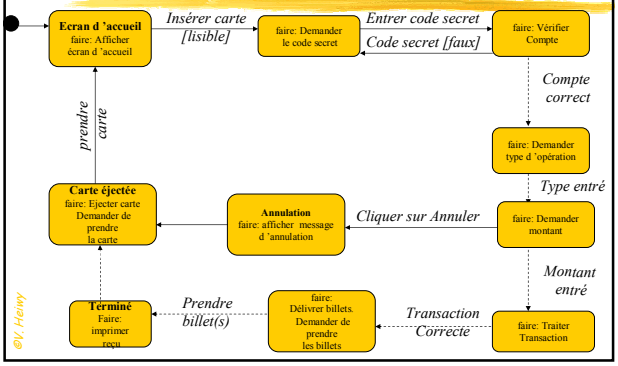


DIAGRAMME D 'ETATS *Exemple 2: le DAB - complété*

Plan



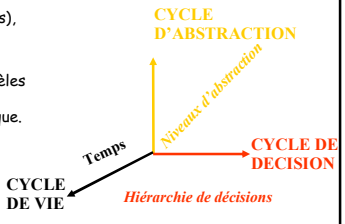
LE DEMARCHE MERISE: *Plan*

- **LES CYCLES**
- **MERISE: UNE DEMARCHE PAR ETAPE**
  - SCHEMA DIRECTEUR
  - ETUDE PREALABLE
  - ETUDE DETAILLEE
  - REALISATION
    - ETUDE TECHNIQUE
    - PROGRAMMATION
  - MISE EN OEUVRE
  - MAINTENANCE

LE DEMARCHE MERISE

LES CYCLES

- La démarche de développement d'un SI doit être conduite suivant 3 axes (les cycles):
- **Le cycle de vie** (échelle de temps),
- **Le cycle de décision** (choix organisationnels et techniques),
- **Le cycle de d 'abstraction** (modèles organisés en niveaux conceptuel, organisationnel/ logique et physique.



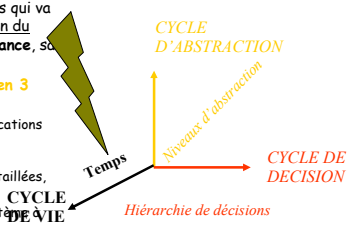
LE DEMARCHE MERISE: *Les cycles*

LE CYCLE DE VIE:

se situe sur une échelle de temps qui va du point de départ à l'exploitation du système, en passant par sa **naissance**, sa **maturité** et sa **maintenance**.

Dans MERISE, il est découpé en 3 périodes:

- La **conception du système** (spécifications fonctionnelles et techniques),
- La **production des programmes** correspondant aux spécifications détaillées, et
- La **maintenance** (adaptation du système à l'évolution de son environnement).



LE DEMARCHE MERISE: *Les cycles*

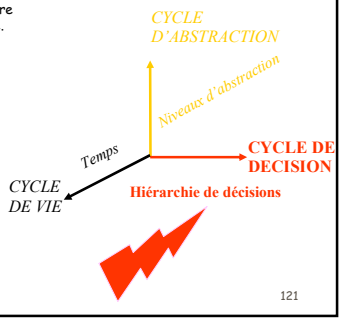
LE CYCLE DE DECISION:

représente l'ensemble des choix à faire durant le déroulement du cycle de vie.

Ces décisions fixent:

- les **objectifs organisationnels** et **techniques**,
- les **priorités de développement**,
- les **ressources** allouées au projet,
- la **mise en place** et le **suivi** de **plannings** d'avancement.

Elles sont prises hiérarchiquement.



LE DEMARCHE MERISE: *Les cycles*

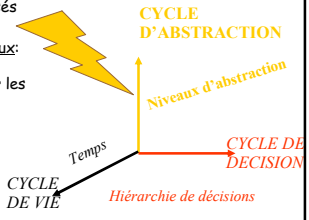
LE CYCLE D'ABSTRACTION:

regroupe l'ensemble des modèles utilisés pour concevoir le système.

Ces modèles sont organisés en 3 niveaux: le **conceptuel**, l'**organisationnel** ou le **logique** et le **technique** (**physique** pour les données et **opérationnel** pour les traitements).

Dans MERISE:

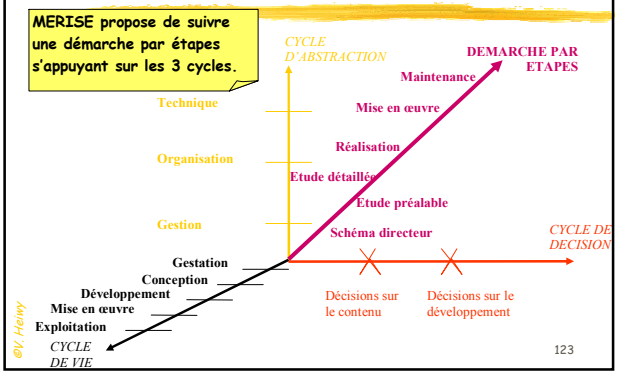
Au niveau le plus élevé, se trouvent les objets les plus stables.



NIVEAU	DONNEES	TRAITEMENTS
CONCEPTUEL	MCD	MCT
ORGANISATIONNEL	MLD	MOT
TECHNIQUE	MPD	MOpT

LE DEMARCHE MERISE *PAR ETAPE* *Introduction*

MERISE propose de suivre une démarche par étapes s'appuyant sur les 3 cycles.



LE DEMARCHE MERISE *PAR ETAPE* *Introduction*

1- LE SCHEMA DIRECTEUR

- Définition des domaines d'étude,
- Planification du développement de chaque domaine,
- Moyens en personnels et en matériel,
- Stratégie de mise en œuvre de la méthode.

2- L'ETUDE PREALABLE

- Porte sur un sous-ensemble du système.
- Doit fournir aux responsables les moyens de prendre les décisions pertinentes sur la globalité du projet.

3- L'ETUDE DETAILLEE

Détermine les spécifications formelles, en accord avec les solutions retenues en (2).

4- LA REALISATION

Étude technique

Production des programmes et consignes d'utilisation.

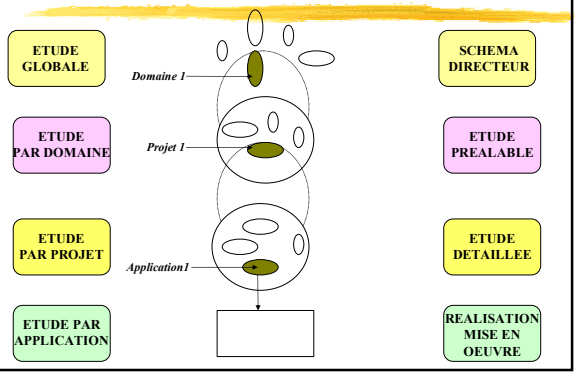
5- LA MISE EN OEUVRE

Mise en place effective du système réalisé dans son environnement réel.

6- LA MAINTENANCE

Adaptation du système exploité aux évolutions de l'entreprise.

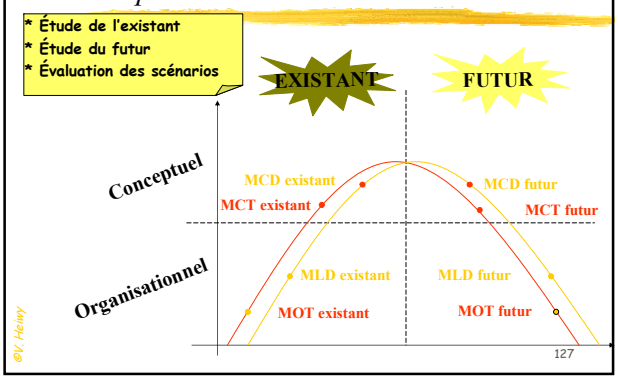
LE DEMARCHE MERISE *PAR ETAPE* *Introduction*



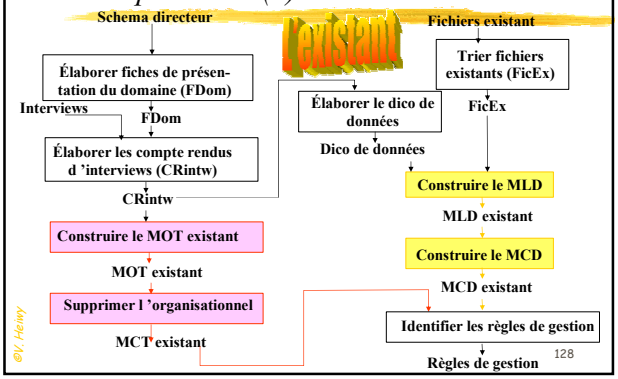
LE DEMARCHE MERISE *PAR ETAPE* *Le schema directeur*

- Étude menée pour la direction générale par des organisateurs et des responsables informatiques.
- **Objectif**: Fixer les grandes orientations pour les années à venir. Il relève de la planification stratégique.
- Établissement d'un **PLAN INFORMATIQUE** pour prévoir l'évolution en terme de matériel et logiciels.
- Il peut être établit à priori ou au fur et à mesure de l'avancement de l'étude préalable.
- Le SD est une **étape continue** qui s'affine au fur et à mesure des étapes préalables.
- Le SI est **découpé en domaines** (*ensemble de processus de SI utilisant des données communes et présentant peu d'échanges avec les autres processus*).

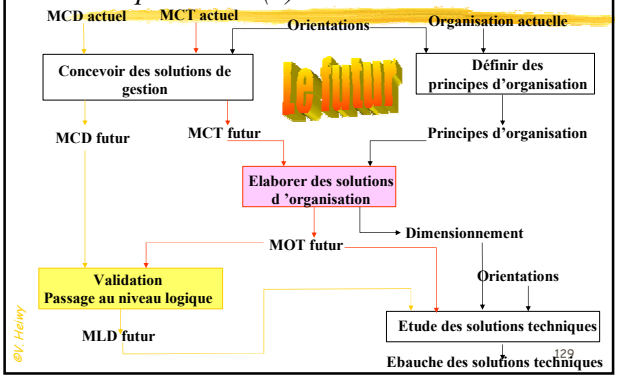
LE DEMARCHE MERISE *PAR ETAPE*
L'étude préalable



LE DEMARCHE MERISE *PAR ETAPE*
L'étude préalable (1)



LE DEMARCHE MERISE *PAR ETAPE*
L'étude préalable (2)



LE DEMARCHE MERISE *PAR ETAPE*
L'étude préalable (3)

- Pour chaque SCENARIO, on rédige un **rapport d'évaluation du scénario** comportant:
  - EVALUATION DES COUTS (Matériel, stockage, logiciel, traitement, communication, exploitation, personnel, formation),
  - EVALUATION DES AVANTAGES (quantifiables et chiffrables au plan financier, quantifiables et non chiffrables (*gain de temps*), et non quantifiables (*aide à la décision*)),
  - EVALUATION DE L'IMPACT SUR L'ORGANISATION (impact sur les postes de travail, acceptabilité du personnel)
  - EVALUATION DE LA FAISABILITE (en matériel, logiciels, personnel),
  - EVALUATION APPROXIMATIVE DES DELAIS (de livraison des matériels, de programmation des logiciels, de recrutement et de formation),
  - EVALUATION DE LA MISE EN ŒUVRE (cadencement du lancement des applications, périodes transitoires).

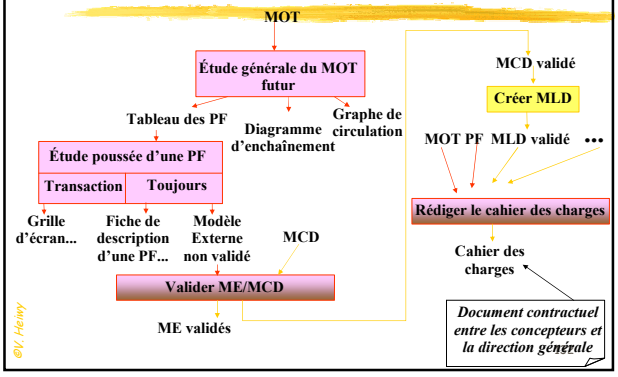
LE DEMARCHE MERISE *PAR ETAPE*
L'étude préalable (4)

Le **dossier de choix** est le document résultant de l 'étude préalable.

- Il comprend tous les documents générés lors de:
  - L 'ETUDE DE L'EXISTANT
  - L 'ETUDE DU FUTUR, et
  - LA DESCRIPTION DE CHAQUE SCENARIO.

- Le Directeur Général devra **trancher** entre les différents scénarios proposés.
- C'est à partir du scénario retenu que sera menée l'étude détaillée.

LE DEMARCHE MERISE *PAR ETAPE*
L'étude détaillée



LE DEMARCHE MERISE *PAR ETAPE*
Réalisation - L'étude technique

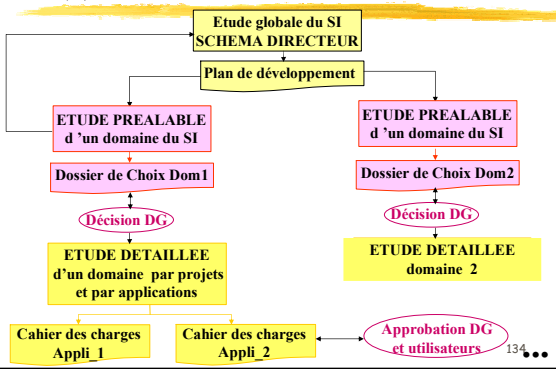
ETUDE TECHNIQUE

- Organisation physique des données
  - Validation des volumes avec l'organisation
- Organisation physique des traitements
  - ARCHITECTURE TECHNIQUE DES PROGRAMMES DIFFERES
  - ARCHITECTURE TECHNIQUE DES PROGRAMMES EN TEMPS REEL
  - Validation par l'exploitation

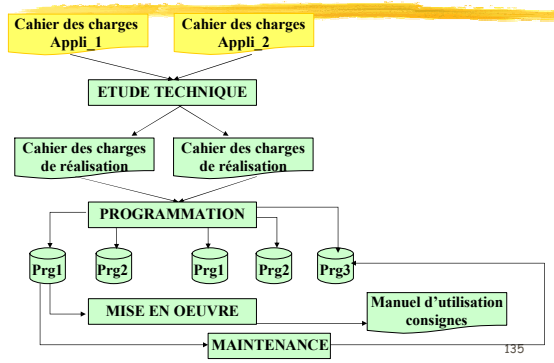
PROGRAMMATION

- Planning de production (stratégie de production et de réception interne des programmes)
  - Validation par le client
- Production des programmes et mise au point
- Test d 'intégration et réception interne des programmes.

LE DEMARCHE MERISE *PAR ETAPE*
Bilan (1)



LE DEMARCHE MERISE *PAR ETAPE*
Bilan (2)



BIBLIOGRAPHIE

- **L 'ESSENTIEL SUR MERISE; D. DIONISI, Ed. Eyrolles; 1994.**
- **COMPRENDRE MERISE, outils conceptuels et organisationnels; J-P. MATHERON, Ed. Eyrolles, 1996.**
- La méthode MERISE Tome 1. Principes et outils; H. TARDIEU, A. ROCHFELD, R. COLETTI; Editions d 'organisation; 1986.
- La méthode MERISE Tome 2. Démarche et pratiques; H. TARDIEU, A. ROCHFELD, R. COLETTI; Editions d 'organisation; 1988.
- La méthode MERISE Tome 3. Gamme opératoire; A. ROCHFELD, J. MOJERON; Editions d 'organisation; 1989.
- **Interfaces graphiques ergonomiques, Conception et Modélisation; Jean-Bernard CRAMPES; Ed. Ellipses;1997.**