Ingénierie des Systèmes d'Information

Problématique et méthodologie : illustration avec la méthode MERISE.

Chap. 7 : Modèles Logiques. Traitement & Données.

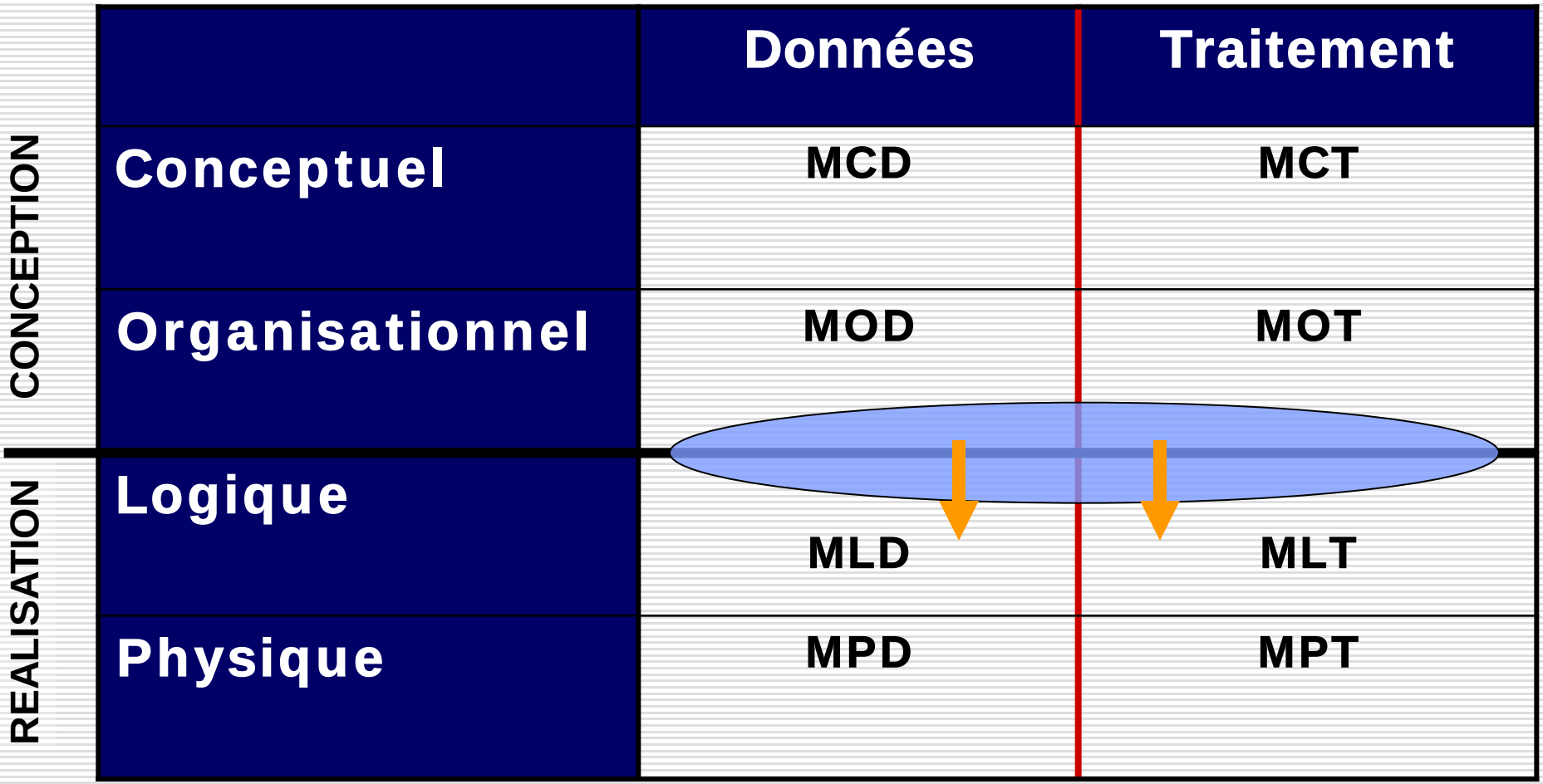
Erwan TRANVOUEZ
erwan.tranvouez@univ-amu.fr

POLYTECH MARSEILLE
Aix*Marseille Université

Plan de la session

- **Transition SIO -> SII**
- **Modèle Logique des Traitements**
  - Objectifs et règles de conceptions
- **Modèle Logique des Données**
  - Objectifs
  - Règles générales de traduction
  - Règles spécifiques (héritage)

Modèles et niveaux d'abstraction



Transition SIO -> SII

- Le SIO a :
  - Modélisé l'existant : fonctionnement organisation, SI existant (ie SIO voire SII)
  - Modélisé le futur : fonctionnement futur de l'organisation, identification/caractérisation des données et des traitements à informatiser …
  - Le périmètre du SIO à informatiser est donc établit
- Mais SIO pas directement implémentable
- Donc le passage au SII requiert de raffiner le SIO pour produire une spécification complète de l'application à développer
- On a décidé dans le SIO QUOI informatiser… reste à établir COMMENT l'informatiser

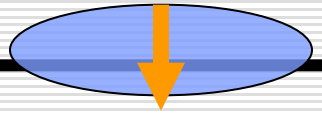
1. Modélisation Logique des Traitements

Objectifs

Règles de conceptions

Modèles et niveaux d'abstraction

		Données	Traitement
CONCEPTION	Conceptuel	MCD	MCT
	Organisationnel	MOD	MOT
REALISATION	Logique	MLD	MLT
	Physique	MPD	MPT



Objectifs des MLT

- Poursuivent le raffinement des MOT au niveau matériel et logiciel.
- Le MOT a (entre autres) :
  - Découpé les opérations en tâches
  - Caractérisé le niveau d'automatisation des tâches
  - Affecté des ressources aux tâches
- Le MLT va préciser leur mise en œuvre au niveau logiciel donc en tenant compte des choix techniques (architecture, capacité du SGBD…) MAIS sans aller au niveau du langage/environnement de programmation…
- On se trouve au niveau du SI Informatisé

Modélisation Logique des Traitements

- Définit comment le MOT sera mis en œuvre informatiquement. Cad en fonction:
  - des choix techniques (SGBD, AGL, …)
  - de l'organisation informatique des traitements (transactions, IHM)
  - des contraintes/règles ergonomiques
  - des principes GL
- Sachant que l'application s'appuiera en général sur un SGBD.

Evolution des concepts

- MOT : Evénement = stimuli au niveau SIO ie message d'un acteur externe au domaine
- MLT : Evénement = stimuli du SII donc tout flux d'information externe au système informatique donc provenant
  - d'un acteur interne du SIO (notion d'opérateur)
  - d'un autre système (SII) de l'entreprise ou du domaine (ex. architecture Client /Serveur)
- Par ailleurs, ces stimuli n'ont pas d'émetteur identifié (analogie d'une fonction qui joue un rôle mais dont on ne spécifie pas l'appelant).

Evolution des concepts

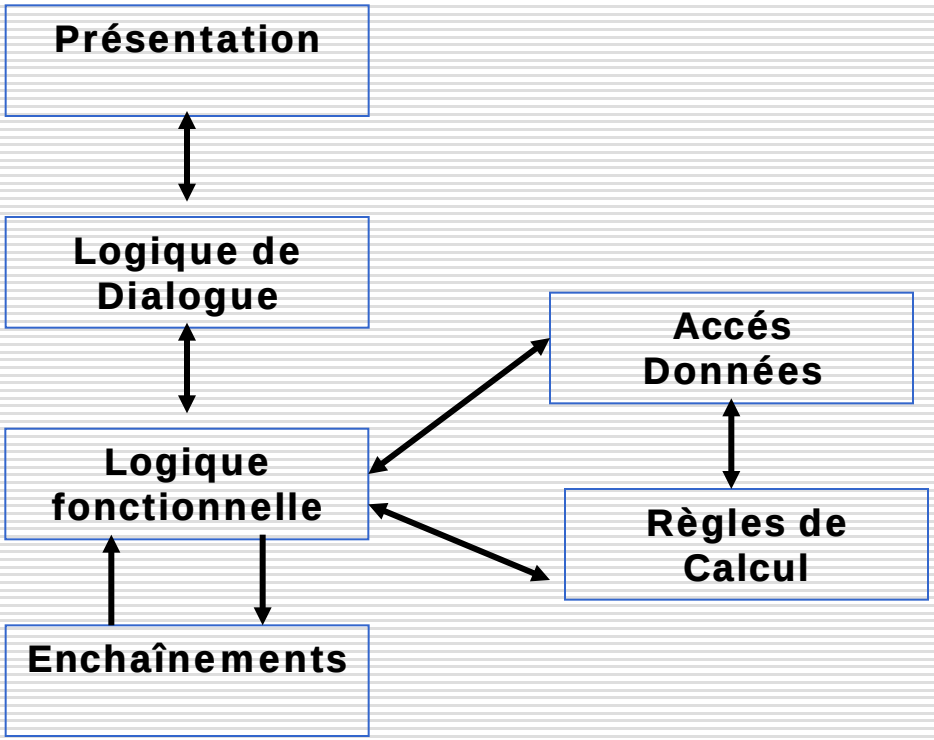
- MOT : Tâche = regroupement de fonctions homogènes
- MLT : Unité Logique de Traitement = regroupement de traitements informatiques regroupés de manière homogène permettant de passer d'un état stable et cohérent à un autre.
- Décomposition du MOT

$$(\text{Tâche}_{(i)})_{i=1..n} \rightarrow (\text{ULT}_{(j)})_{j=1..m}$$

- Cette notion d'ULT peut être apparentée à celle de grosse fonction du SII qui se déclinera en :
  - Fenêtres, Transactions (au sens BD ie regroupement de requête), groupes de traitements/fonction etc…
- Ce nouveau découpage doit préparer la réalisation du logiciel et donc assurer la classification des traitements comme par exemple en MVC, Couche Métier/Logique/Applicative …

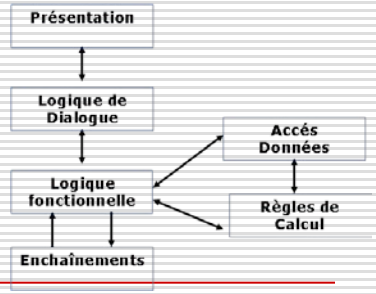
Conception des ULT

- Décomposition des tâches MOT en fonction de la nature des traitements à réaliser (ie Interface Traitement, Données)
- 6 grands types d'ULT :



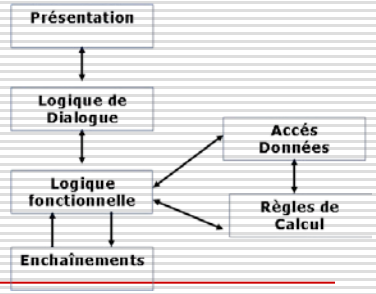
Largement inspiré de [Nancy & Espinasse]

Caractérisation des ULT



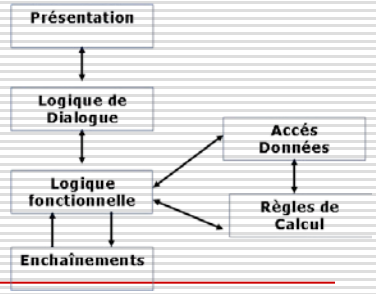
- Interface (ie IHM) basée sur la Présentation et la logique Dialogue (à l'exclusion des accès données, calculs etc…)
- **Présentation** :
  - ULT chargée de communiquer avec l'extérieur (ie IHM)
  - Support « physique » de la communication
    - Écran, Imprimante, Haut-Parleur, …
    - Clavier, Scanner, caméra, Micro …
  - Doit respecter des règles ergonomiques ie le message doit être bien compris et la présentation ne dois pas être source d'erreur de manipulation (respects normes…)
  - Ex: schématisation d'une fenêtre et ses menus/boutons ou conception via outils dédiés

Caractérisation des ULT



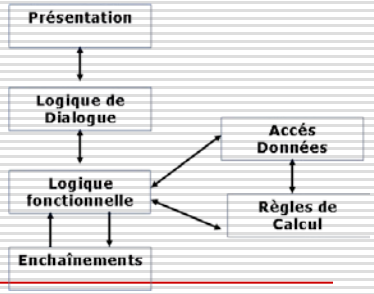
- **Logique de dialogue** définit les règles de gestion de la présentation ie:
  - Réaction par rapport aux événements souris en fonction des boutons
  - Enchaînement des fenêtres possibles
  - Vérification des données…
- Exemples de séparation de rôle dans l'Interface
  - IDE :
    - outil de conception graphique de la fenêtre (positionnement géographique, libellé, forme et couleur…)
    - génération automatique du code PRODUISANT L'INTERFACE (ie forme seulement) avec des zones « *mettre le code correspondant à l'évènement clic souris ICI* »
  - Page web :
    - les champs de formulaires (zone de texte, bouton, …) structure les informations le code
    - Le code Javascript vérifie les données d'un formulaire (format date, adresse email valide …) AVANT que les informations soient envoyées au serveur

Caractérisation des ULT



- Logique fonctionnelle:
  - ie algorithmes, transformations, coordination entre Interface & Données
  - Correspond au Contrôleur du modèle MVC de développement d'IHM.
  - Caractérise et contrôle donc toutes les interactions avec les autres types d'ULT.
- Règle de calcul:
  - garantie que les règles de gestions seront respectées
  - Considère des informations valides issues de l'interface (ces vérifications auront déjà été réalisées au niveau logique de dialogue)
  - Par ex. : Si on dispose des infos de sommes à débiter d'un compte (IHM), du solde du compte (Données) on peut vérifier que le compte est suffisamment abondé

Caractérisation des ULT



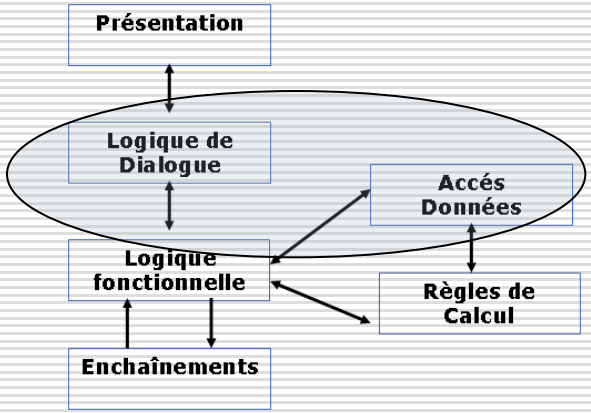
- Accès au données comprends:
  - Le sous-schéma logique de données: ie sous-ensemble du MLD caractérisant les informations nécessaires
  - Opérations de manipulation de l'information (lecture, calculs, etc… => future reqûete SQL)
  - Contrôle de cohérence et intégrité des données
- Enchaînement:
  - Décrit les liaisons entre les ULT et les liens d'activation possibles (avec éventuellement des conditions)
  - Assez naturellement représentable avec un diagramme de séquence UML (cf cours UML prochaines séances).

Pourquoi décomposer aussi finement ?

- Approche classique voire basique de Génie Logiciel : comment préparer au mieux le développement pour réduire les risques : *diviser pour régner !*
  - Arriver à des objectifs simples et précis (atomique) : 1 fonction/1 objet résoud 1 problème à la fois pour éviter de mélanger les problèmes techniques (ex. tâche de compilation décomposée en différentes phases d'analyse)
  - Rendre les parties le plus indépendantes possibles les unes des autres => mise à jour/correctif de l'une sans conséquence (grave) sur les autres
  - Distribution et implémentation parallèle des ULT « séparés » puis intégration des solutions (suppose tests à tous les niveaux : unitaire et intégration)
  - Réutilisation possible des ULT (cf ci après)

Conception des MLT

- A noter : possibilité de regrouper certains types de tâche
  - Principe: « objet métier » miroir Objet d'une occurrence d'information stockée dans une BD fournissant les méthodes d'accès au données et de présentation à une interface
  - Ex: JDO, Hibernate …



- Réutilisation de ULT
  - Ex. recherche de produits apparaît dans +sieurs tâches. L'ULT est définie 1 fois et appelée plusieurs fois dans différents contexte)
  - S'appuie sur une logique de bibliothèque de fonctions (API) ou de composants (ex. Windev avec des fenêtre de saisies de commandes prédéfinies).
  - Peut cependant complexifier l'ULT (ex. dans Windev la même fenêtre fait la saisie, le parcours et la modification de données => Code non optimisé et plus lourd à maintenir ou a débugger…)

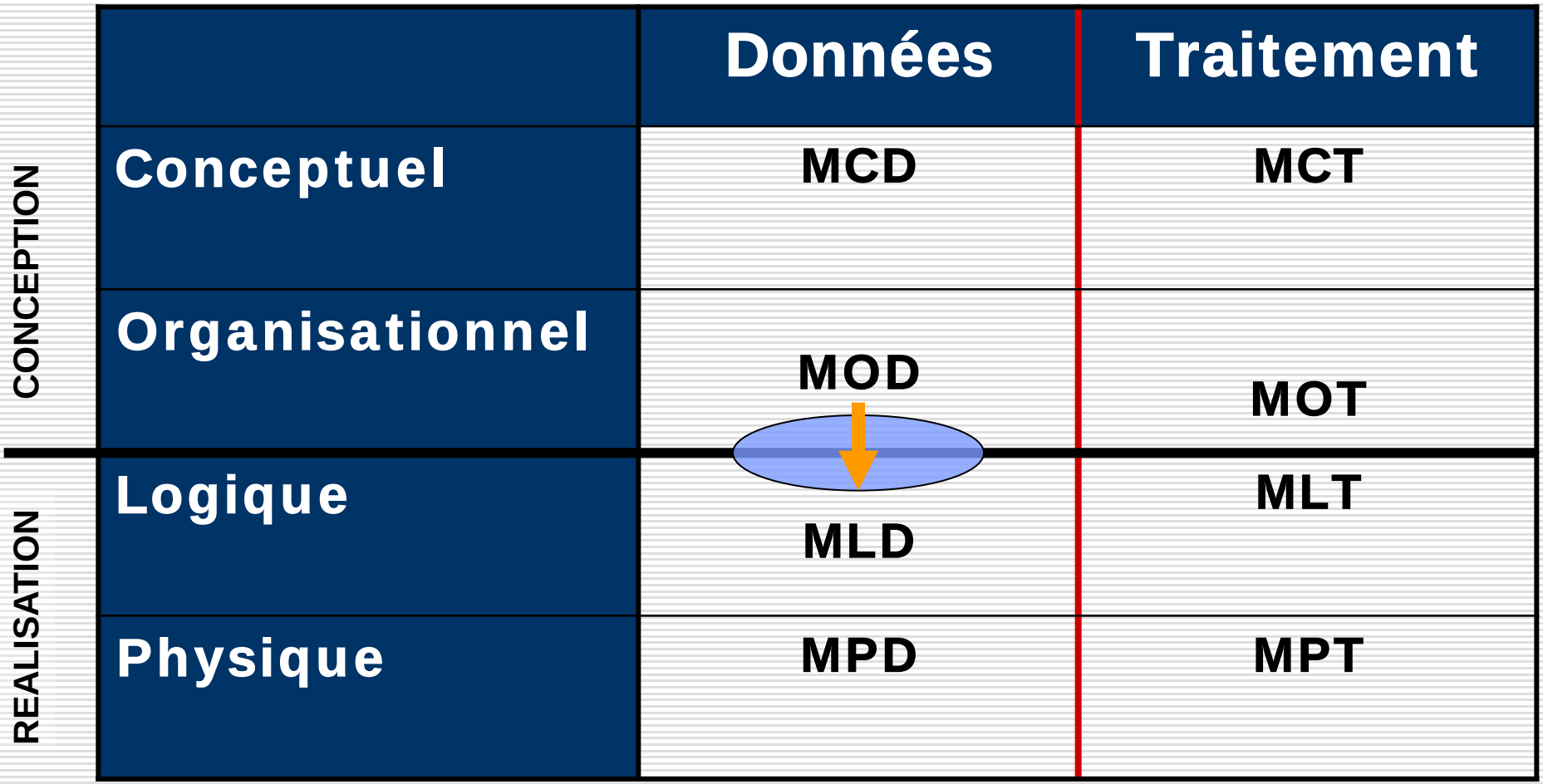
2. Modélisation Logique des Données

Objectifs

Règles générales de traduction

Règles spécifiques (héritage)

Modèles et niveaux d'abstraction



Objectifs des MLD

- De même que le MLT, le MLD intègre la dimension informatique du SI et prépare la mise en œuvre du MOD dans un Système informatique ... en général un SGBD ou une application construite autour d'un moteur de SGBD.
- Le MLD tient donc compte du couple logiciel/matériel cible...
  - S'il s'agit d'un SGBD relationnel => le MLD sera de fait un schéma relationnel.

Modélisation Logique des Données

- La suite du cours considère le cas d'un modèle relationnel des données, exprimé selon
  - Des relations : <-> entités du MOD. Ce sont les futures tables...
  - Des attributs : <-> propriétés des entités du MOD
    - Parmis lesquels :
      - Des **clés primaires** <-> identifiant(s) du MOD
      - Des **clés étrangères** : non exprimées dans le cadre du SIO, mais résultent des relations (au sens MCD) entre les entités.
    - Soumises à des contraintes sur les valeurs qu'elles peuvent prendre (unicité des clés primaires, existence et cohérence des valeurs des clés étrangères, ...)

Modélisation Logique des Données

- Ces relations doivent être normalisés (1NF, 2NF, 3NF ...) afin d'assurer la structuration la plus optimale des données (pas de redondance).
- Cela étant dit, les MCD et MOD ont normalement été construit dans cette logique...

Conception des MLD

- La construction du MLD par rapport au MLT présente l'avantage de s'appuyer sur des règles de transformations systématiques

- Cela facilite le développement d'outils/fonctions de traduction MCD/MOD -> MLD (WinDesign par ex.)

- Les règles peuvent être simples (transformation d'une relation 1..n ou n..) mais peuvent se complexifier et requérir l'expertise de l'analyste programmeur dans les cas plus complexes (héritage, relation n-aires, ...)

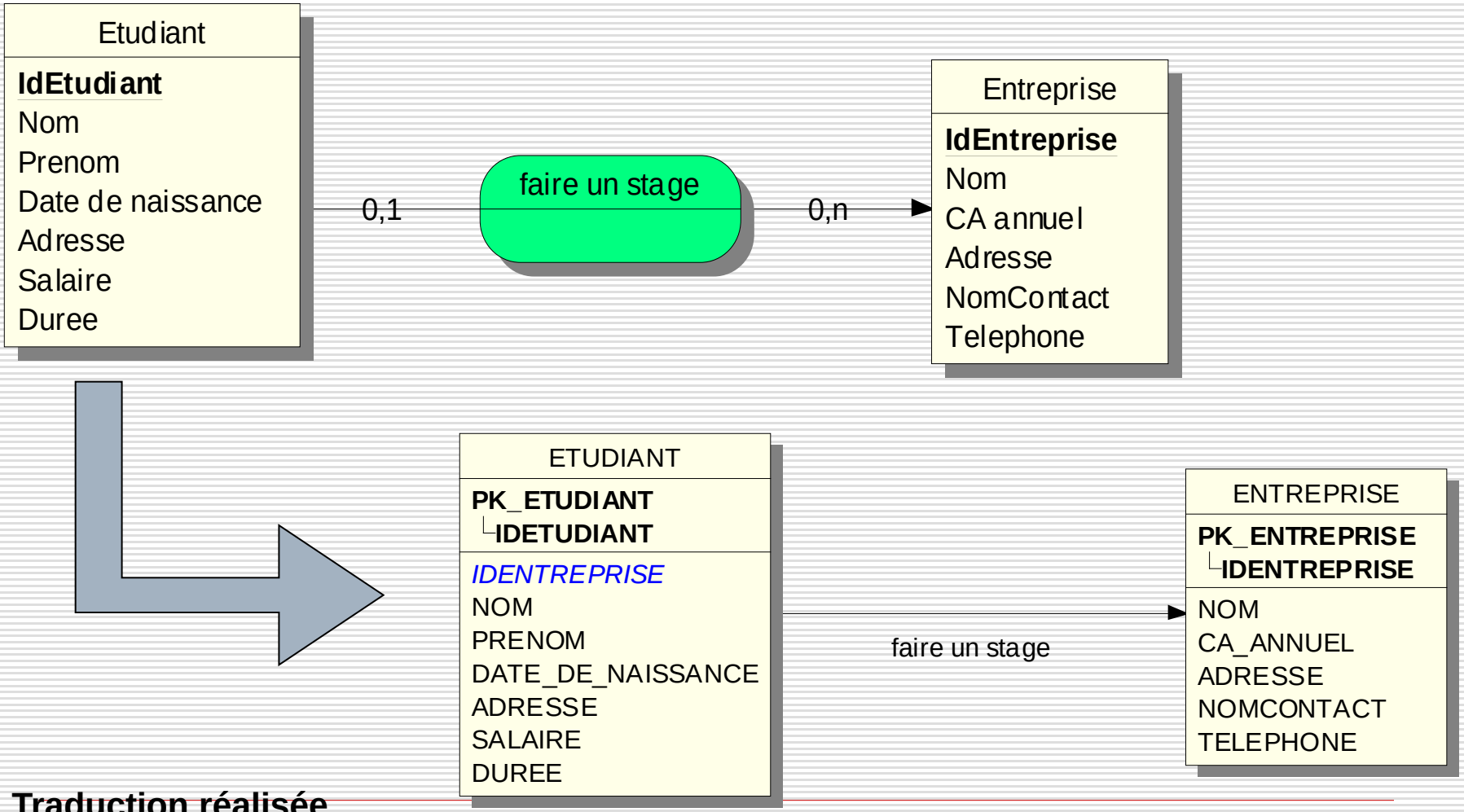
Relation binaire 1 .. N

Principe de traduction

- C'est-à-dire une relation (x .. 1) -> (x .. n)
- Ce type de relation traduit une dépendance fonctionnelle.
  - Ex. Un chien a 1 propriétaire
    - => On marque son nom sur son collier.
    - Si x vaut 0 (donc pas d'obligation d'avoir un propriétaire), alors le chien n'a pas de collier.
- Se traduit au passage du MLD
  - par la création d'une **clé étrangère** sur la relation **dépendante**.
  - Si x vaut 0 (donc participation pas obligatoire) cela signifie que l'attribut portant cette clé étrangère peut ne pas être renseigné (***null*** autorisé au niveau implémentation). Une 2ème solution est possible sur le principe des relations n..n.

Relation binaire 1 .. N

Exemple



Traduction réalisée avec WinDesign

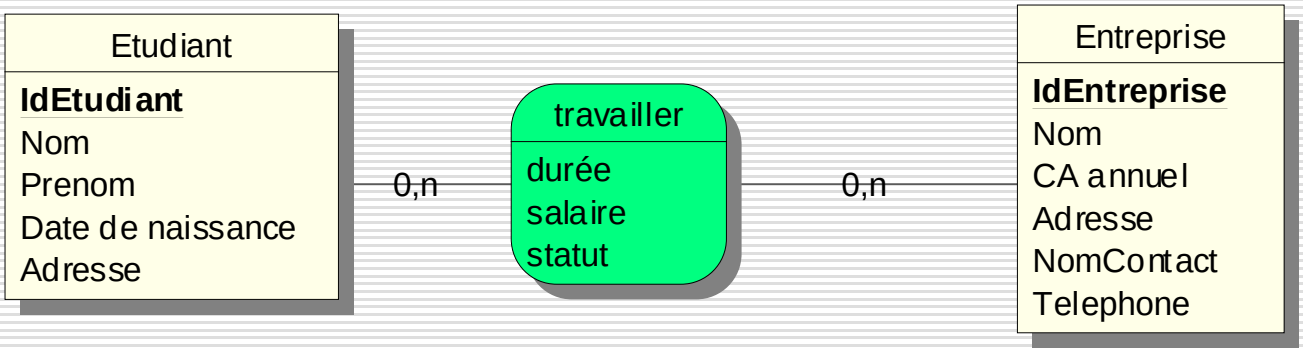
Relation binaire N .. N

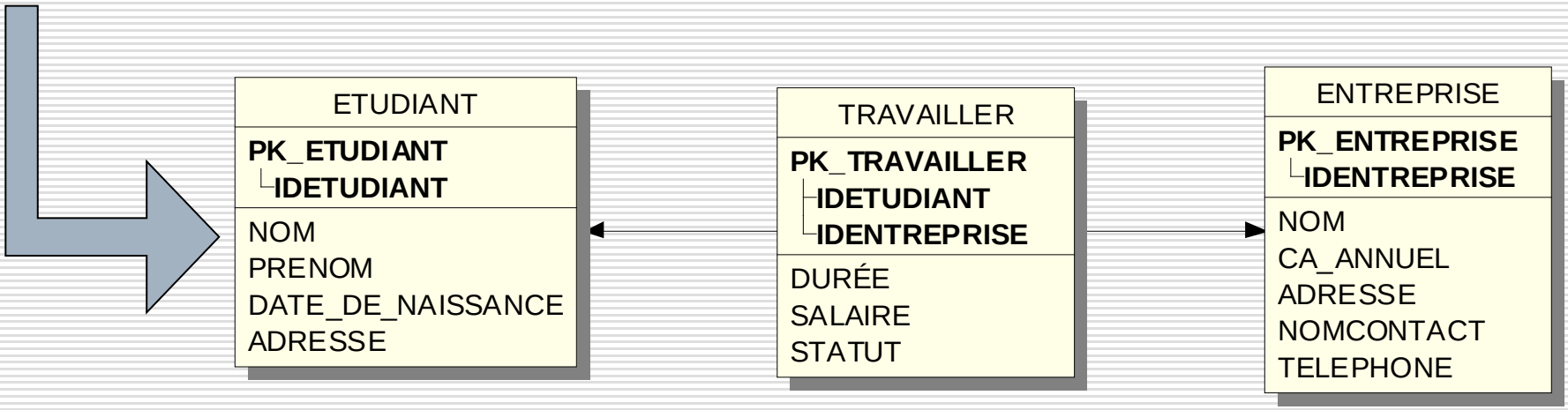
Principe de traduction

- C'est-à-dire une relation (x .. n) -> (x .. n)
- Cas précédent inapplicable.
- Utilise une 3ème table qui associera les occurrences participant à la relation (MCD)
- Si l'occurrence a de A et b de B participe a la relation (A,B) alors il existe un couple de valeur (a,b) dans la table AB.
- A noter, que le même principe peut être appliqué dans le cas des relation 0..1 – 0..n afin d'éviter

Relation binaire N .. N

Exemple

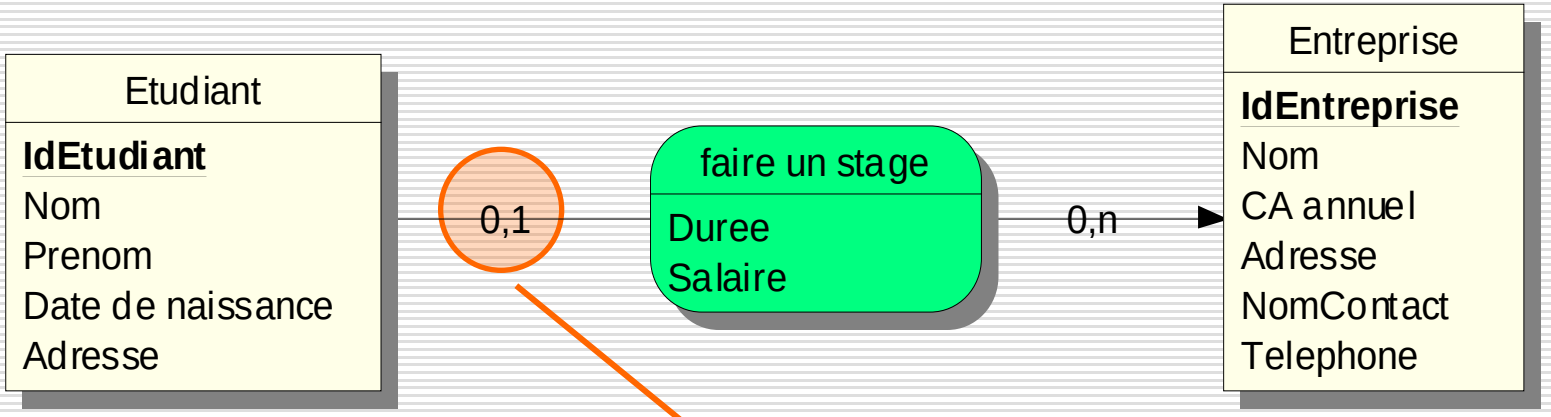




Traduction réalisée avec WinDesign

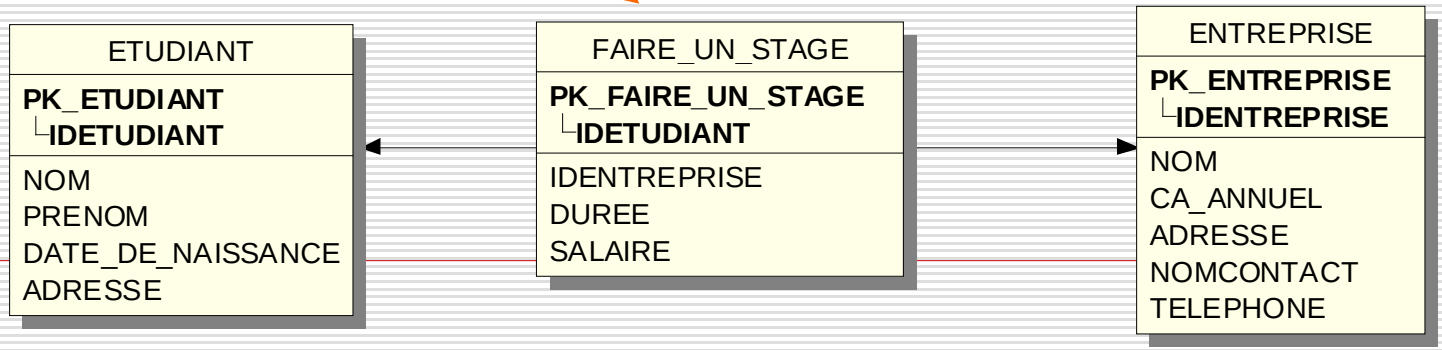
Retour sur la Relation binaire 1 .. N

Autre solution



Ainsi, pas de valeur d'attribut pouvant être nulle mais « surcoût » en terme de stockage d'information

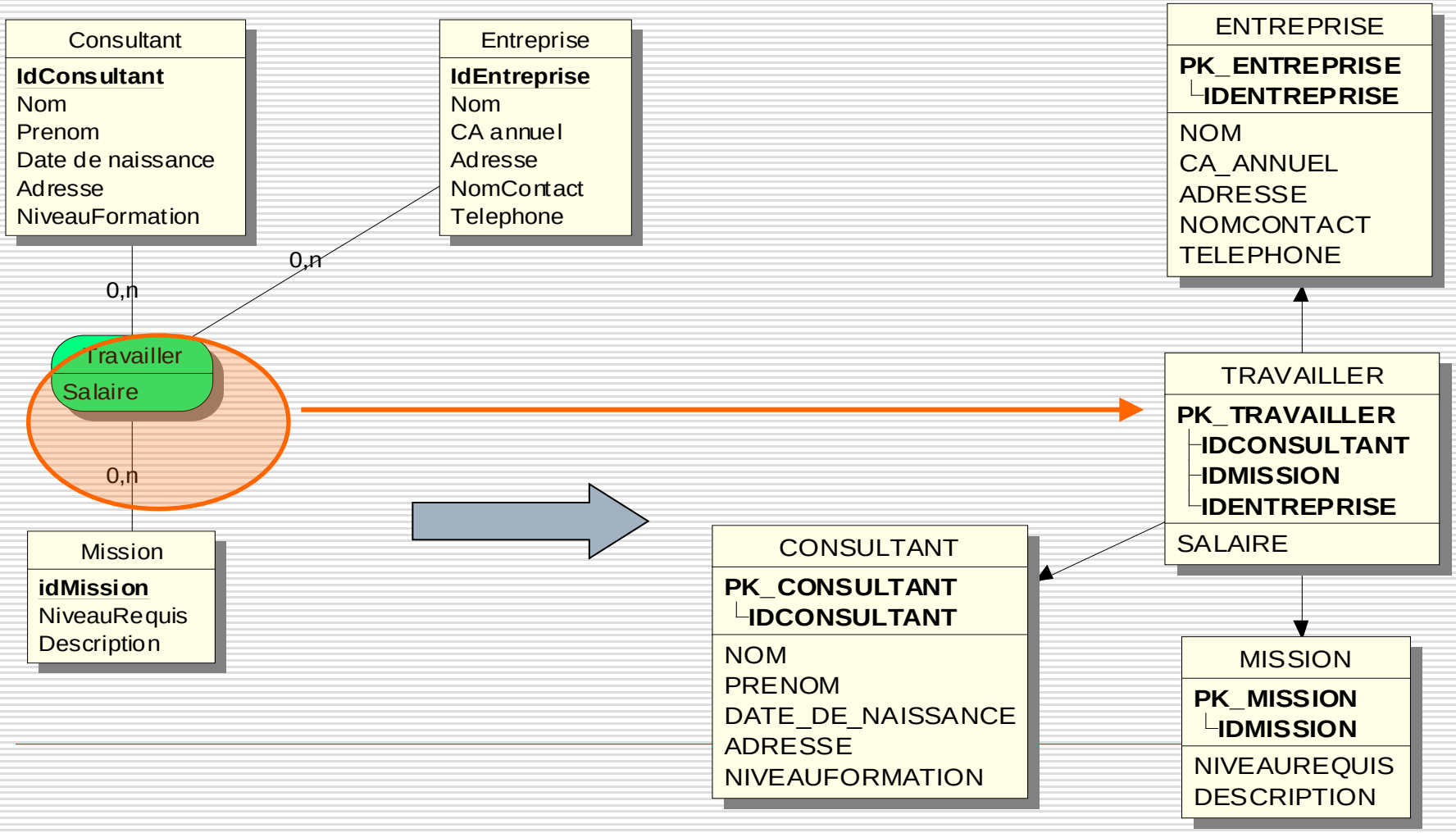
Traduction réalisée avec WinDesign



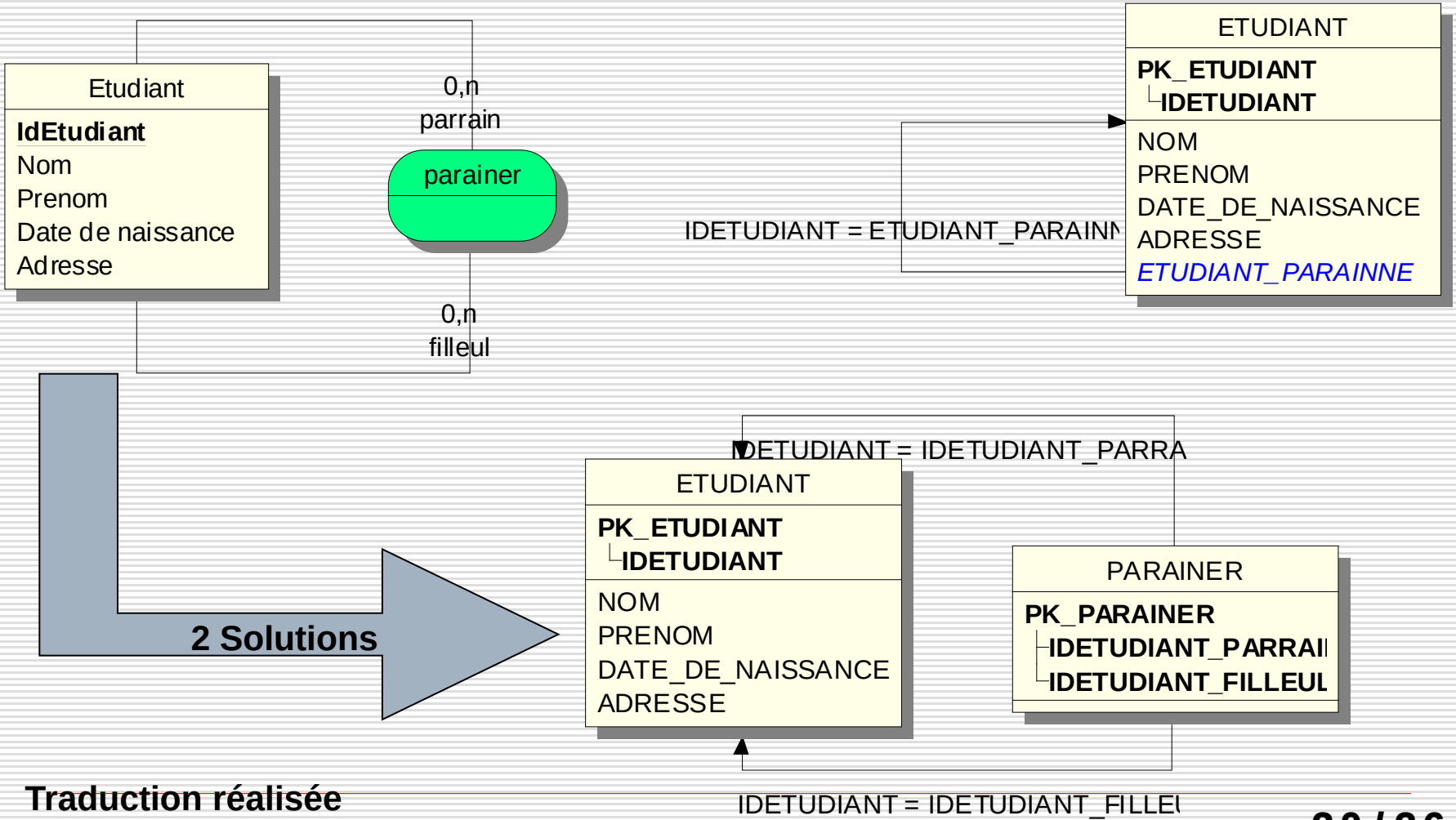
Relation ternaire

Principe de traduction

- Même principe que binaire n..n.



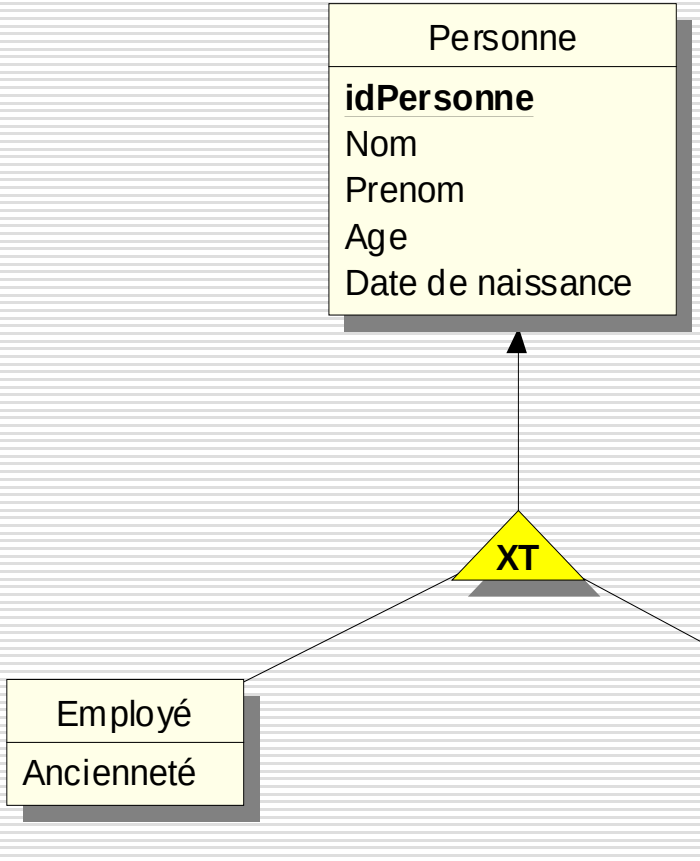
Relation réflexive

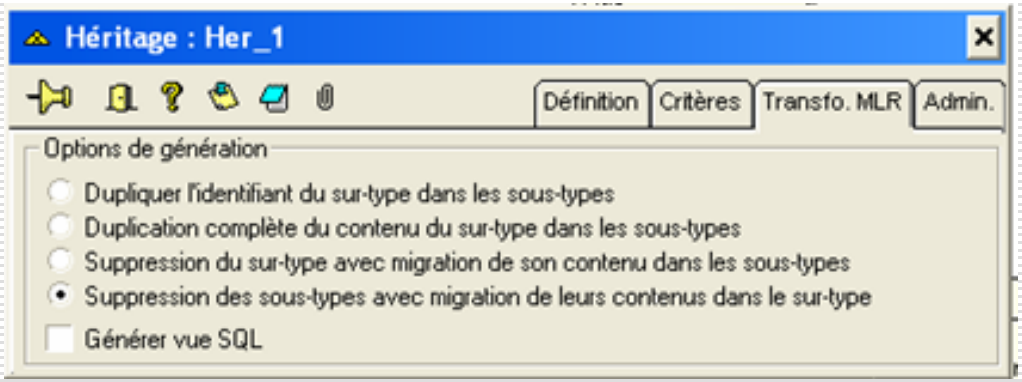


Traduction réalisée avec WinDesign

Héritage

Cas de départ





Traduction réalisée avec WinDesign

Héritage

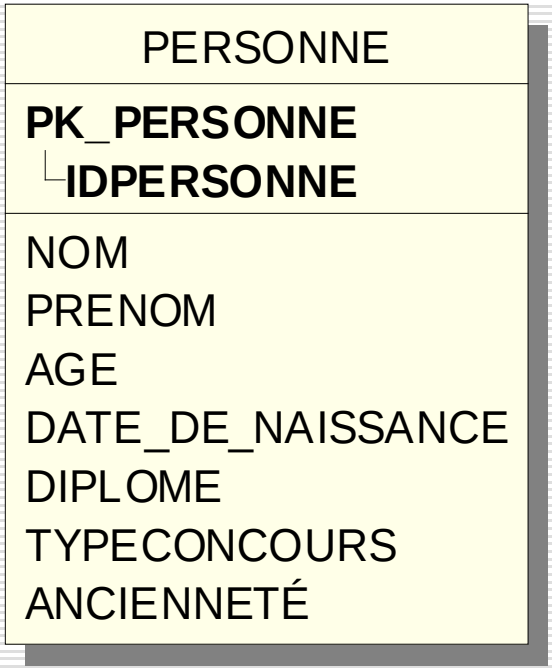
Solution 1

- Fusion de toutes les informations et ajout d'une propriété pour identifier le type.

☺ Simplifie la traduction

☹ Nécessite de gérer le fait que des attributs peuvent être nuls.

⇒Utilisation de vues (SQL) pour traduire le fait que voir une personne comme un employé revient à afficher les infos d'une personne + ancienneté mais pas le diplôme ou le type de concours (d'autant plus si la relation d'héritage est exclusive)



Héritage

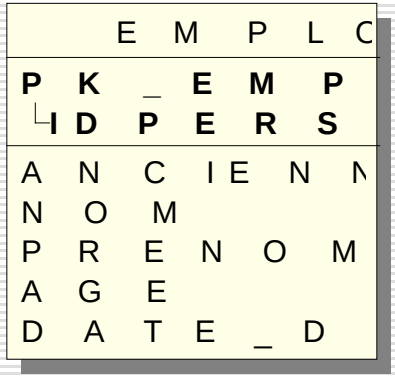
Solution 2

- Duplication des infos générales sur les relations spécialisées

☺ Simplifie la traduction

☹ Redondance des données

☹ Perd la possibilité de faire des relations sur personne. Ex. une personne habite dans une maison.



Traduction réalisée avec WinDesign

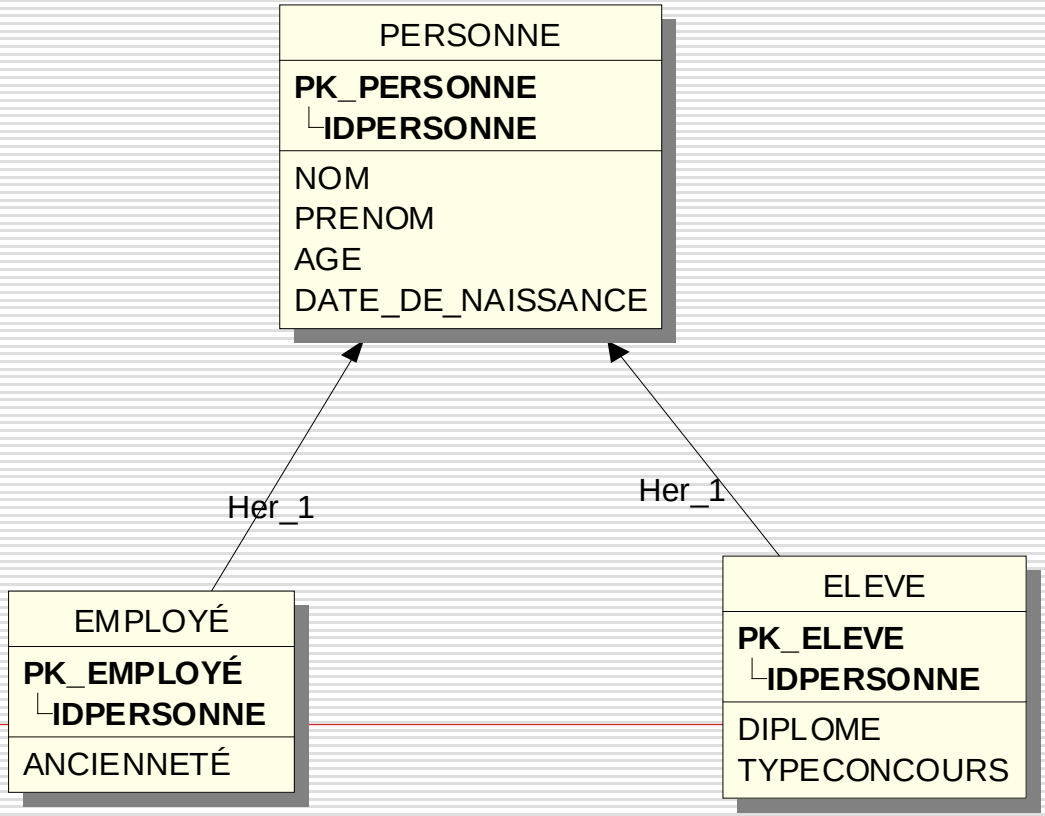
Héritage

Solution 3

- Les entités spécialisées sont considérées comme des relations. Les mêmes valeurs de clés sont utilisées…

☺ Respecte la logique

☹ Manipulation des clés



Traduction réalisée avec WinDesign

Héritage

Solution 4

- Les entités spécialisées sont considérées comme des relations ET le contenu est dupliqué

☺ Respecte la logique

☹ Gestion des duplicats

=> Utilisation de Triggers pour propager les modifications d'une entité à une autre.