Conception des systèmes d'information
Merise et UML

Caroline DEVRED

Université d'Angers

Plan

Introduction

Génie logiciel

- Cherche à fournir des méthodes pour concevoir des programmes qui tiennent la route (un bug peut coûter très très cher).
- Le génie logiciel est donc ensemble de méthodes et outils pour la conception de logiciels de qualité
- Plusieurs méthodes proposées, dont :
  - Merise (vieille méthode ayant fait ses preuves)
  - UML (plus adaptée aux applications objets).
- Malheureusement rien ne peut garantir une application parfaite, restez vigilant.

Merise

Système

Système

Ensemble d'éléments (matériels ou immatériels : hommes, machines, méthodes, règles etc.) en interaction, dont les éléments sont organisés et coordonnés pour atteindre un objectif.

Étudier le système

- Identifier le système et son environnement.
- Décomposer en sous systèmes.
- Mettre en évidence les flux.

- On ne s'intéressera qu'aux systèmes constitués par des organisations (entreprises, associations, services etc.).

Système (suite)

- En général :
- Un système opérant (SO).
- Un système de pilotage (SP).
- Un système d'information (SI).

Système d'information

- Interface entre le SO et le SP.
- Stocke et gère les informations du SO ; et les met à la disposition du système de pilotage.
- Interaction SO – SI :
  - mise-à-jour du SI par le SO ;
  - SO fait une action uniquement si le SI lui donne telle ou telle information.
- SI mémoire de l'organisation : données, structures de données, règles, contraintes, méthodes de l'organisation etc.

Système d'information (suite)

- Un aspect statique : données du monde extérieur, règles et contraintes de l'extérieur.
- Un aspect dynamique : possibilité de mise à jour (données + règles).

On informatise quoi ?

Le SO fait des actions programmées ou obéit à des choix. Le choix reste le propre de l'homme (même si certains choix peuvent être automatisés :-( ). Le SI ne peut que mettre à disposition du SO des informations aidant à la décision.

Vocation de Merise

- Mersise va chercher à modéliser le système.
  - Un modèle pour comprendre le système.
  - Un modèle pour communiquer.
- Proposer une méthode de conception des SI.
- Démarche méthodologique de développement des SI.

Proposer une méthode de conception des SI

- Approche globale du SI menée parallèlement sur les données et les traitements.
- Description du SI par niveau (du plus abstrait au moins abstrait). Appelé aussi cycle d'abstraction.
  - conceptuel (déterminé par les choix de gestion) ;
  - organisationnel (déterminé par les choix d'organisation) ;
  - logique (déterminé par les contraintes techniques) ;
  - physique (ou opérationnel – déterminé par les contraintes techniques).
- Description du SI avec le modèle ≪entité – relation ≫.
- 2 premiers niveaux indépendants de l'organisation physique et informatique.
- Représentation graphique.

Démarche méthodologique de développement des SI

- Découpage du développement en 4 étapes et deux axes.
  - Étude préalable.
  - Étude détaillée.
  - Réalisation/Mise en œuvre.
  - Les données.
  - Les traitements.
- On parle aussi de cycle de vie (naissance, maturité, maintenance). Un bouleversement profond de l'organisation et de son environnement conduit à un nouveau cycle de vie.
- Le cycle de vie et le cycle d'abstraction se mélangent.

Niveau conceptuel – Données

Niveau conceptuel

- Posons nous la question de savoir quelles sont les contraintes pesant sur le système, quels sont les résultats attendus.
- Correspond à la finalité de l'organisation.
- S'agit de décrire le QUOI.
- 2 modèles (2 invariants) :
  - Modèle Conceptuel des Données (MCD). Niveau statique.
  - Modèle Conceptuel des Traitements (MCT) Niveau Dynamique.
- INDÉPENDANT de la manière dont cela sera implenté.

MCD

- Description des données et de leurs interactions.
  - Entité.
  - Relation.
  - Propriété.
- 2 approches :
  - Pragmatique ou intuitive.
  - Technique.

Concept d'entité

Exercice

La société ≪les trois belges ≫ fonctionne de la façon suivante. 2 fois par an, elle dépose dans les boites aux lettres un catalogue des produits qu'elle vend. Chaque personne intéressée va passer auprès de notre société une commande. Cette commande peut bien sûr porter sur plusieurs produits. Après une quinzaine de jours, la commande est envoyée au client (le vert du pull ne correspond absolument pas à celui de la photo) ainsi qu'une facture.
Chercher à identifier les acteurs (matériel ou inmatériel) intervenant (interagissant) lors du fonctionnement de l'entreprise les ≪trois belges≫. N'oubliez pas, on cherche à modéliser le SI et non le SO ou le SP.

Concept d'entité (suite)

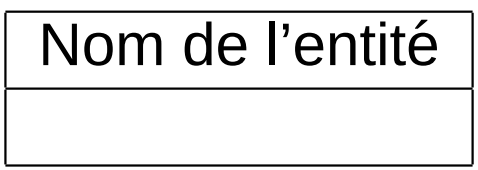
Solution

Il y a quatre acteurs : les clients, les commandes, les produits et les factures.

Formellement

Une entité est un objet (! dans le sens de chose !) pourvue d'une existence propre et conforme aux choix de gestion de l'organisation.

- On la représente de la façon suivante :



- Note pour plus tard : une entité est complètement définie si elle possède des propriétés (des informations).
- Entité représente ce qui est réellement perçu.

Concept de relation

Exercice

Les entités sont liées. Donnez les liens entre les entités des ≪trois belges≫.

Concept de relation (suite)

Solution

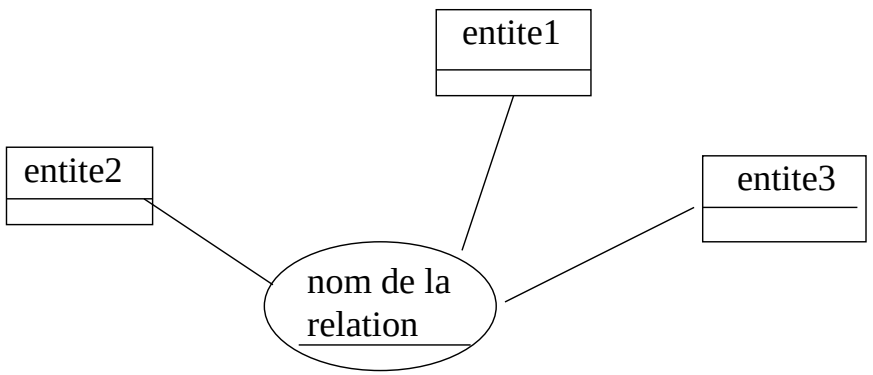
- Un client passe une commande.
- Une commande porte sur des produits.
- Un produit est facturé .
- À une commande est associé une ou plusieurs factures.

Concept de relation (presque fin)

Formellement

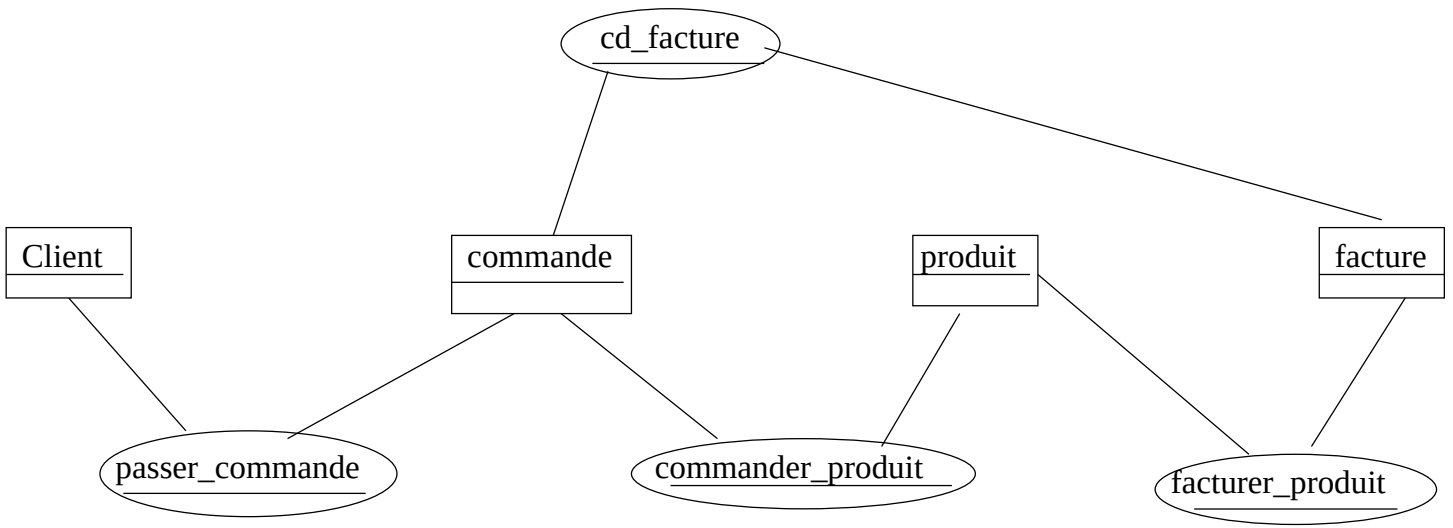
Une relation entre entités est une association perçue dans le réel par une ou plusieurs entités.

- Elle se représente ainsi :



Concept de relation (fin)

Solution



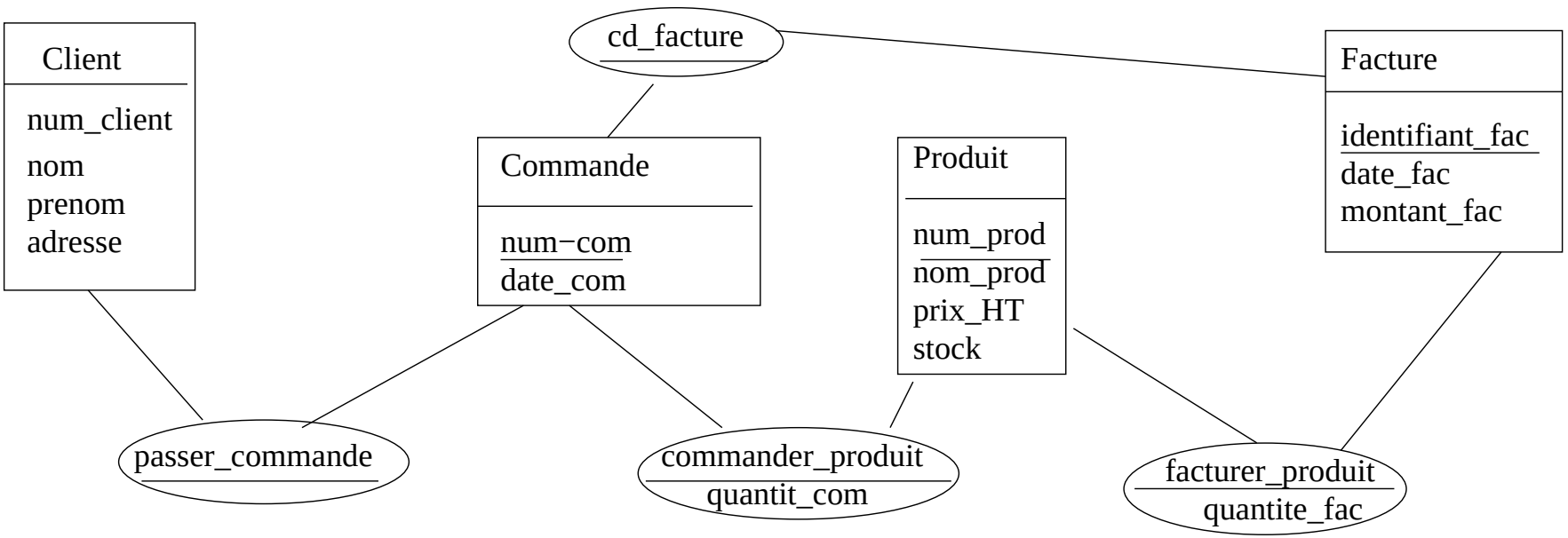
Concept de propriété

- On complète les entités et associations par les informations qui les concernent.
- Quelques idées sur notre exo fil rouge ? (Ce n'est pas une question, juste une figure de style).

Formellement

Une propriété (ou attribut) est une donnée élémentaire que l'on perçoit sur une entité ou sur une relation entre entités.

Propriétés de l'exemple fil rouge



Mais. . .

- Dans la vie réelle jamais les systèmes ne sont aussi simples, et construire les MCD à la volée ne garantit pas d'avoir un MCD valide :'(
- Il faut faire les choses avec méthode :
  - dictionnaire des données,
  - dictionnaire des données élémentaires,
  - dépendances fonctionnelles,
  - graphe épuré des dépendances fonctionnelles,
  - passage au MCD,
  - cardinalités.

Dictionnaire des données

- À la recherche des propriétés perdues.
- Le but : identifier chaque propriété (chaque information), la nommer, lui donner sa signification, son type (voir plus loin), sa longueur (i.e. la place qu'elle va prendre) sa nature et une contrainte d'intégrité ou une règle de calcul.
- Toujours garder une place à table pour les identifiants.

Un peu plus sur les propriétés

- Plusieurs natures (ou encore classes) :
  - Élémentaire (E) ou concaténée (CO).
  - Calculée (C).
  - Paramètre (P).
- Plusieurs types :
  - Numérique (N).
  - Alphabétique (A).
  - Alphanumérique (α).
  - Date (D).
- Contrainte d'intégrité (CI) : ce que la propriété doit vérifier pour être reconnue comme valable.
- Règle de calcul (RC) : calcul à effectuer pour obtenir la valeur de la propriété.

Dictionnaire des données

- Se présente sous la forme d'un tableau :

Nom	Signification	Type	Longueur	Nature	CI ou RC
stock	indique le stock du produit	N	4	E	≥ 0

- Compléter le tableau avec notre exercice fil rouge.

Dictionnaire des données des «trois belges»

Nom	Signification	Type	Longueur	Nature	CI ou RC
num_client	l'id du client	N	4	E	pas 2 pareils
nom	le nom du client	A	20	E	
prénom	le prénom du client	A	20	E	
adresse	l'adresse du client	α	150	CO	
num_com	le numéro de la commande	N	4	E	pas 2 pareils
date	la date de la commande	D	20	E	$\leq$ date du jour
client_com	client passant commande	N	4	E	
num_prod	identifiant du produit	N	4	E	pas 2 pareils
nom_prod	nom du produit	α	50	E	
prix_HT	prix HT du produit	N	4	E	≥ 0
TVA	TVA	N	4	P	$= 0.186$
prix_TTC	prix avec TVA	N	4	C	prix_TTC = prix_HT $\times(1+$ TVA)
stock	indique le stock du produit	N	4	E	≥ 0
prod_com	produit commandé	N	4	E	
prod_fact	produit facturé	N	4	E	
quantité_com	quantité de produit commander	N	4	E	≥ 0
quantité_fac	quantité de produit facturée	N	4	E	$\leq$ quantité_com
identifiant_fac	numéro de facture	N	6	E	pas 2 pareils
date_fac	date de la facture	D	20	E	$\leq$ date du jour
montant_fac	montant de la facture	N	6	CA	somme des prix_TTC facturés

Dictionnaire des données élémentaires

- On épure le dictionnaire des données :
  - Adieu propriété paramètre ou calculée.
  - Adieu synonyme (2 propriétés représentant la même chose).
  - Adieu polysène (2 propriété de même nom mais représentant 2 informations différentes).
  - Morcellement des données concaténées.

Dictionnaire des données élémentaires des «trois belges»

Nom	Signification	Type	Longueur	Nature	CI ou RC
num_client	l'id du client	N	4	E	pas 2 pareils
nom	le nom du client	A	20	E	
prénom	le prénom du client	A	20	E	
num_ad	num de l'adresse du client	*N*	3	E	
rue_ad	rue de l'adresse du client	*A*	20	E	
code_postal	code postal du client	*N*	5	E	5 chiffres
ville	ville du client	*A*	20	E	
num_com	le numéro de la commande	N	4	E	pas 2 pareils
date_com	la date de la commande	D	20	E	$\leq$ date du jour
num_prod	identifiant du produit	N	4	E	pas 2 pareils
nom_prod	nom du produit	α	50	E	
prix_HT	prix HT du produit	N	4	E	≥ 0
stock	indique le stock du produit	N	4	E	≥ 0
quantité_com	quantité de produit commander	N	4	E	≥ 0
quantité_fac	quantité de produit facturée	N	4	E	$\leq$ quantité_com
identifiant_fac	numéro de facture	N	6	E	pas 2 pareils
date_fac	date de la facture	D	20	E	$\leq$ date du jour

Dépendances fonctionnelles

- On cherche les liens entre les éléments du dictionnaire des données élémentaires.

Formellement

- On dit que 2 propriétés sont reliées par une dépendance fonctionnelle si la valeur de l'une détermine au plus une valeur de l'autre.
- On dit que *b* dépend fonctionnellement de *a* si une valeur de *a* n'entraîne qu'une valeur de *b*.
  - On note $a \xrightarrow{df} b$;
  - *a* est la source de la dépendance fonctionnelle ;
  - *b* est le but de la dépendance fonctionnelle.

Classification des dépendances fonctionnelles

- Une dépendance fonctionnelle simple n'a qu'un but et qu'une source (num_client $\xrightarrow{df}$ nom).
- Une dépendance fonctionnelle à partie gauche composée est une dépendance fonction dont la source est composée de plusieurs propriétés ((num_com, num_pro)$\xrightarrow{df}$ quantité_com).
- Aucune sous-partie de la source d'une dépendance fonctionnelle élémentaire ne permet d'en déduire le but ((num_com, num_pro)$\rightarrow$ quantité_com et (num_client, nom) $\xrightarrow{df}$ rue_adresse).
- Une dépendance fonctionnelle directe $a \xrightarrow{df} b$ est une dépendance fonctionnelle telle qu'il n'existe pas de propriété c telle que $a \xrightarrow{df} c$ et $c \xrightarrow{df} b$.

Les dépendances fonctionnelles en vue d'un MCD

- Les dépendances fonctionnelles qui nous intéressent sont les dépendances fonctionnelles élémentaires directes .
- Cherchez les dépendances fonctionnelles des ≪trois belges≫et morcelez-les pour obtenir uniquement des dépendances fonctionnelles élémentaires directes.

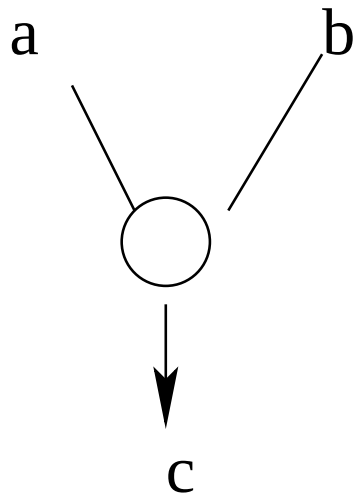
Les dépendances fonctionnelles des «trois belges»

- num_client $\longrightarrow$ nom
- num_client $\longrightarrow$ prénom
- num_client $\longrightarrow$ num_ad
- num_client $\longrightarrow$ rue_ad
- num_client $\longrightarrow$ code_postal
- num_client $\longrightarrow$ ville
- num_com $\longrightarrow$ date_com
- num_produit $\longrightarrow$ nom_prod
- num_produit $\longrightarrow$ prix_HT
- num_produit $\longrightarrow$ stock
- identifiant_fac $\longrightarrow$ date_fac
- identifiant_fac $\longrightarrow$ num_com
- num_com $\longrightarrow$ num_client
- (identifiant_fac , num_produit) $\longrightarrow$ quantité_fac
- (num_com , num_produit) $\longrightarrow$ quantité_com

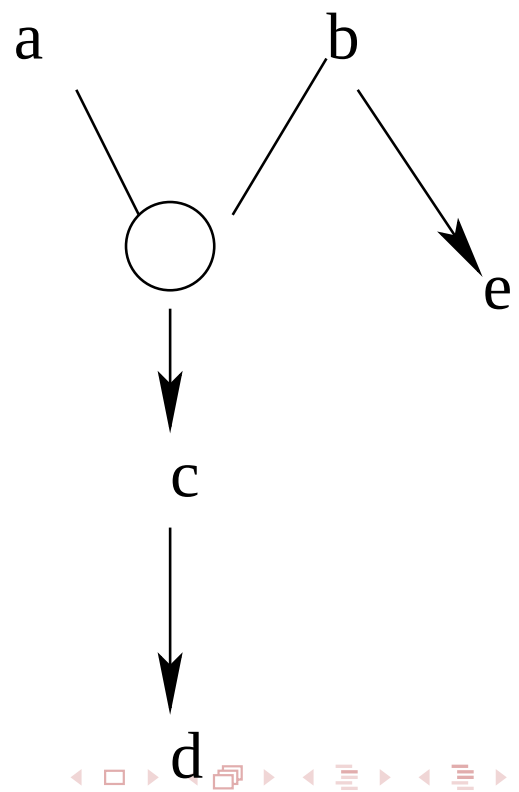
Représentation graphique des dépendances fonctionnelles et graphe des dépendances fonctionnelles

$a \rightarrow b$

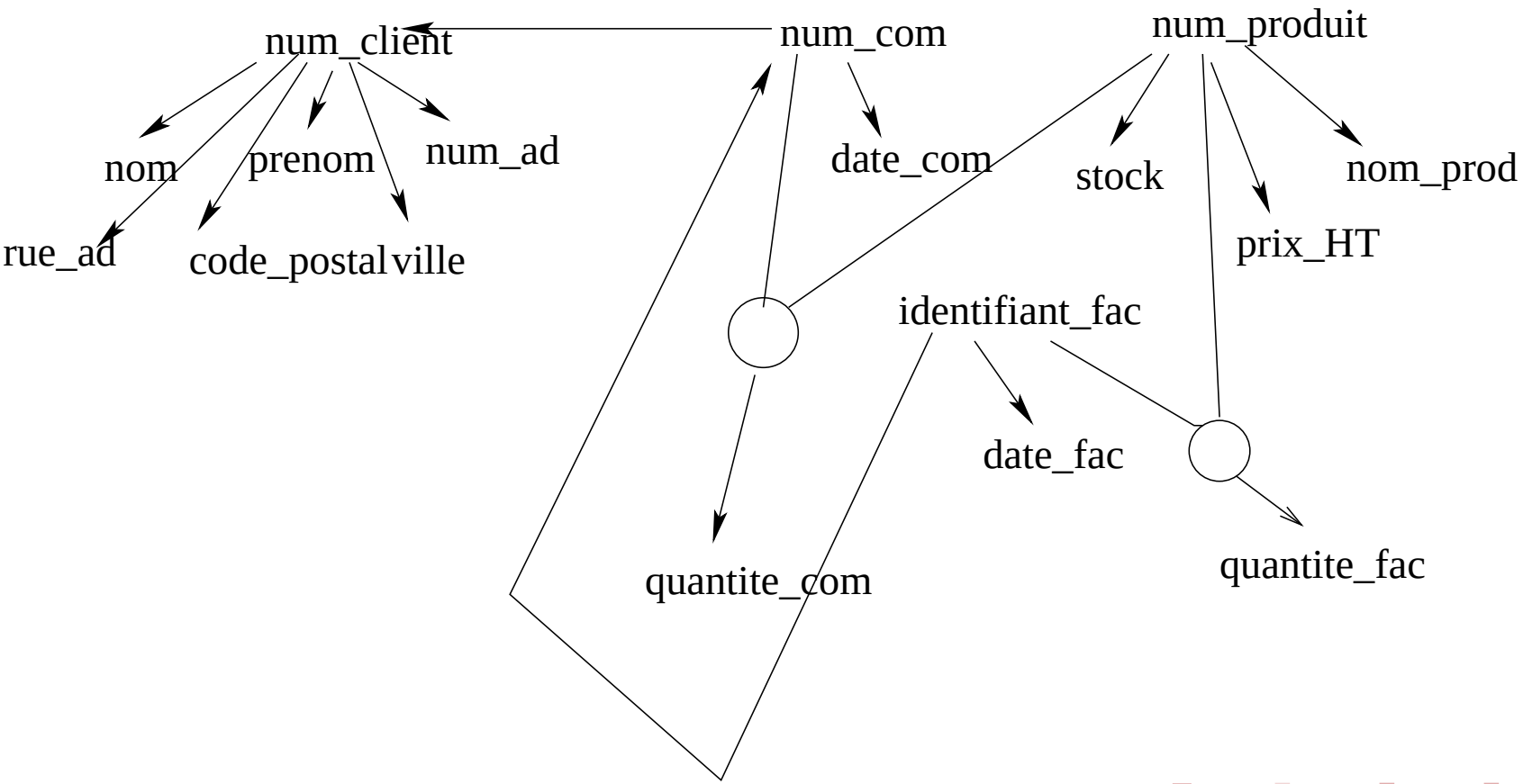
$(a, b) \rightarrow c$



$(a, b) \rightarrow c\,;\, c \rightarrow d\,;\, b \rightarrow e$



Le graphe des dépendances fonctionnelles des «trois belges»

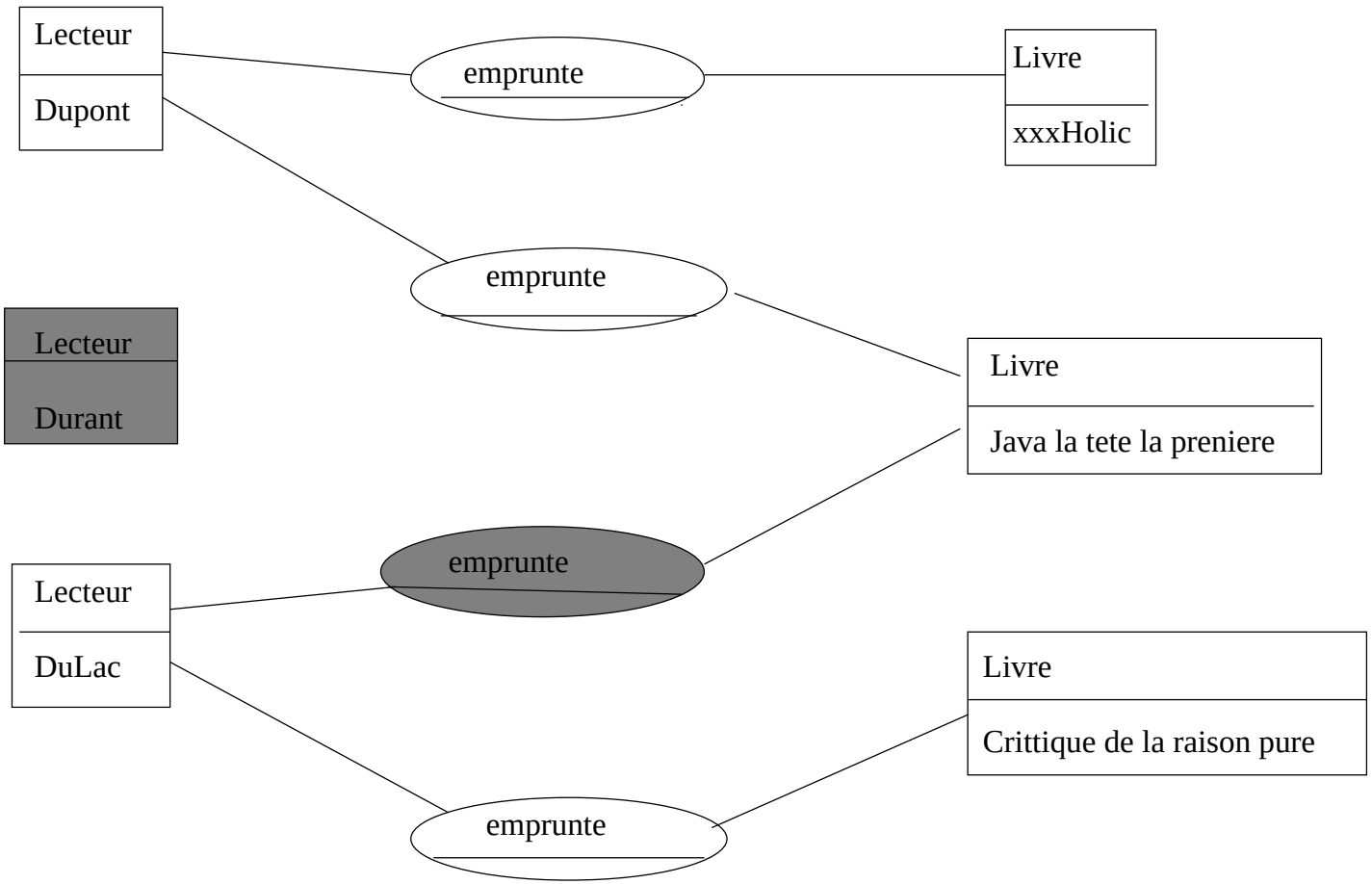


Le passage du graphe des dépendances fonctionnelles au MCD

- Il ne reste plus qu'à passer du graphe des dépendances fonctionnelles au MCD, i.e. quel groupe de propriétés forme une entité et quelles entités sont en relation.
- On rappelle qu'une entité est une ≪chose≫ réelle du SI et qu'une relation représente les interactions entre les entités.

La cardinalité objet-relation

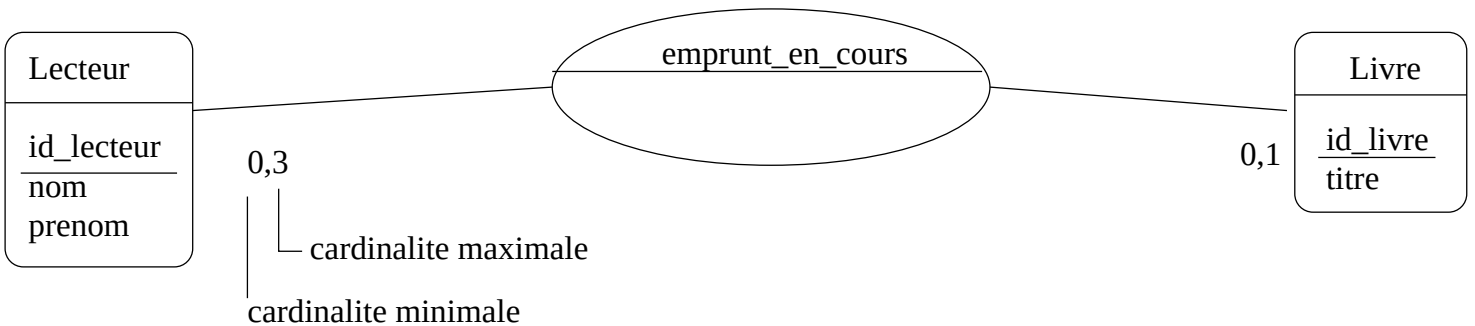
- Le diagramme d'occurrence.



La cardinalité objet-relation

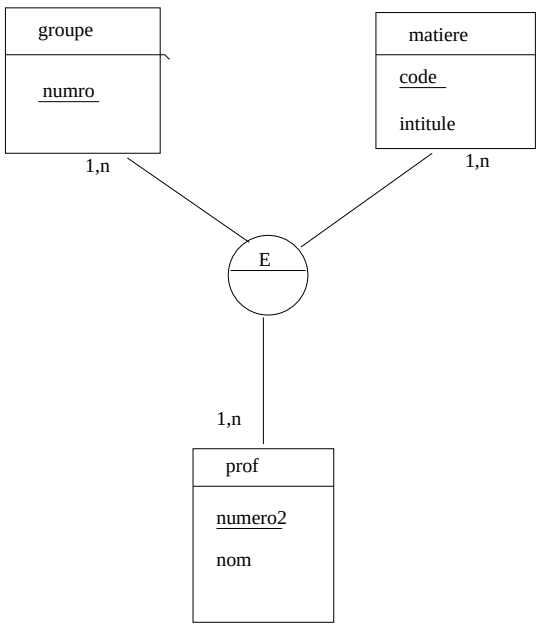
- La cardinalité d'une relation est le nombre minimal ou maximal d'occurrences d'une entité pouvant prendre part à une relation .
  - La cardinalité minimale est le nombre minimum de fois où une occurrence d'une entité peut appartenir aux occurrences de la relation.
  - La cardinalité maximale est le nombre maximum de fois où une occurrence d'une entité peut appartenir aux occurrences de la relation.

Exemple de cardinalité objet-relation binaire



Un Lecteur peut emprunter de 0 à 3 livre. Pour un Lecteur combien de livre au minimum, combien de livre au maximum ?

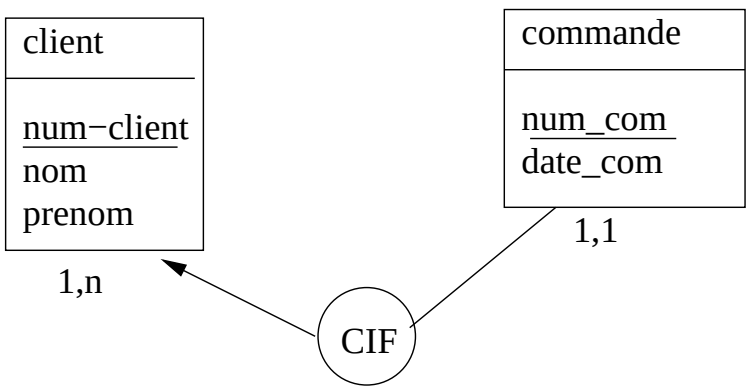
Exemple de cardinalité objet-relation n-aire



Pour un professeur combien de couples (Groupe, Matière) au minimum, combien de couples (Groupe, Matière) au maximum ?

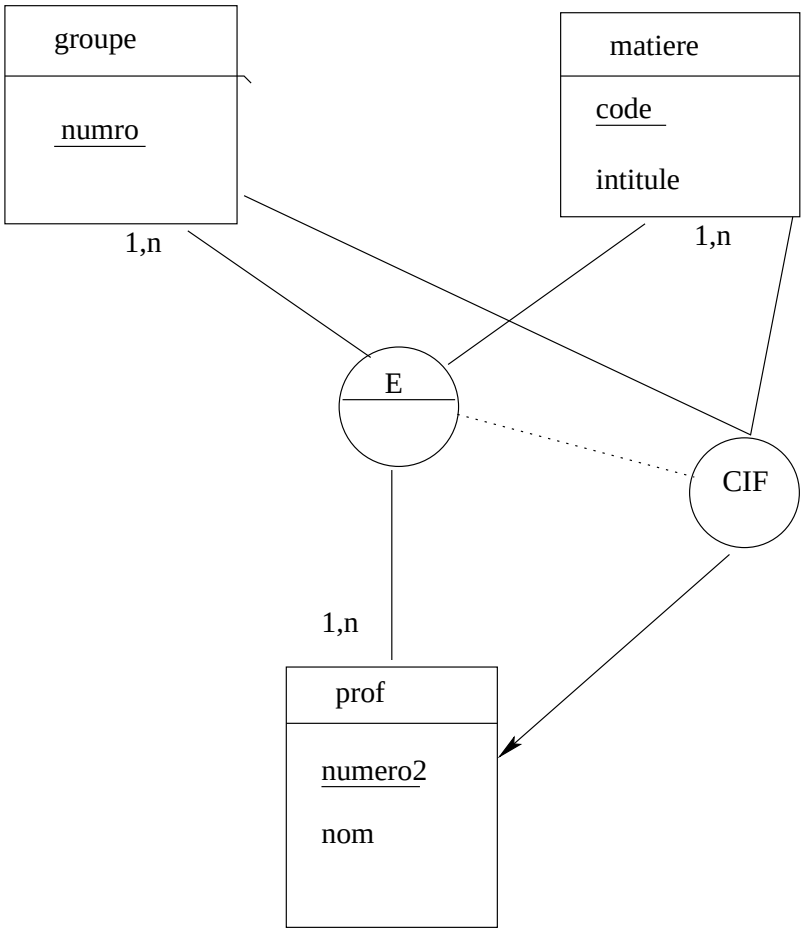
Contrainte d'intégrité fonctionnelle

Certaines cardinalités vont générer des associations particulières : Lorsque la cardinalité maximale d'une entité dans une association binaire est égale à 1, c'est à dire lorsqu'une entité est totalement connue lorsqu'on en connaît une autre.



Contrainte d'intégrité fonctionnelle

Il peut y avoir des CIF sur une association non binaire.



Les identifiants

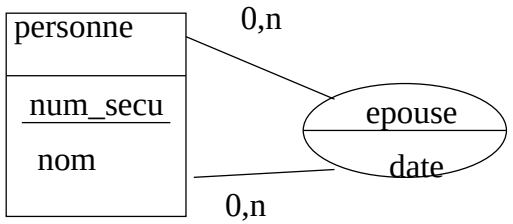
- D'une entité : c'est une propriété particulière de l'entité telle qu'à chaque valeur de la propriété corresponde une unique occurrence de l'entité. On la représente soulignée.
- D'une relation : c'est la concaténation des identifiants des entités appartenant à la relation. Elle a le même but.

Dimension d'une relation

- C'est le nombre d'entités participant à la relation.
- Une relation binaire concerne 2 entités.
- Une relation ternaire concerne 3 entités.
- Une relation n-aire concerne n-entités.

Relation réflexive

- C'est une relation d'une entité avec elle même.



- Un petit mot sur la polygamie. En vrai cela s'appelle règle de gestion.

Du graphe des dépendances fonctionnelles au MCD

- Déterminer les rubriques sources.
- Déterminer les entités.
- Déterminer les CIF binaire.
- Déterminer les CIF multiples.
- Déterminer les autres relations.
- Ajouter les relations non induites par le graphe des DF.
- Répartir les propriétés entre les entités et les associations.
- Réamménager le MCD si besoin est (normalisation).

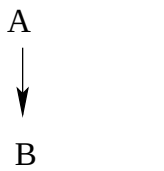
Toutes les CIF sont représentés dans le graphe des DF, c'est la seule garantie en ce qui concerne les relations.

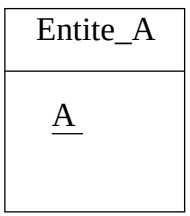
Déterminer les rubriques sources

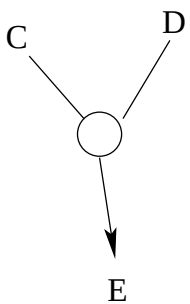
Une rubrique source est :

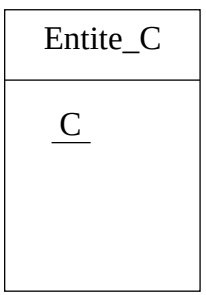
- soit source de dépendance fonctionnelle simple ou d'une dépendance fonctionnelle à partie gauche composée ;
- soit le but de plusieurs dépendances fonctionnelles sans être la source d'une dépendance fonctionnelle ;
- soit une rubrique isolée.

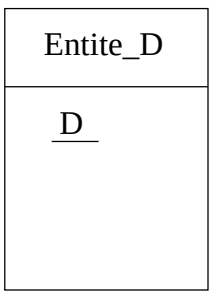
Déterminer les entités



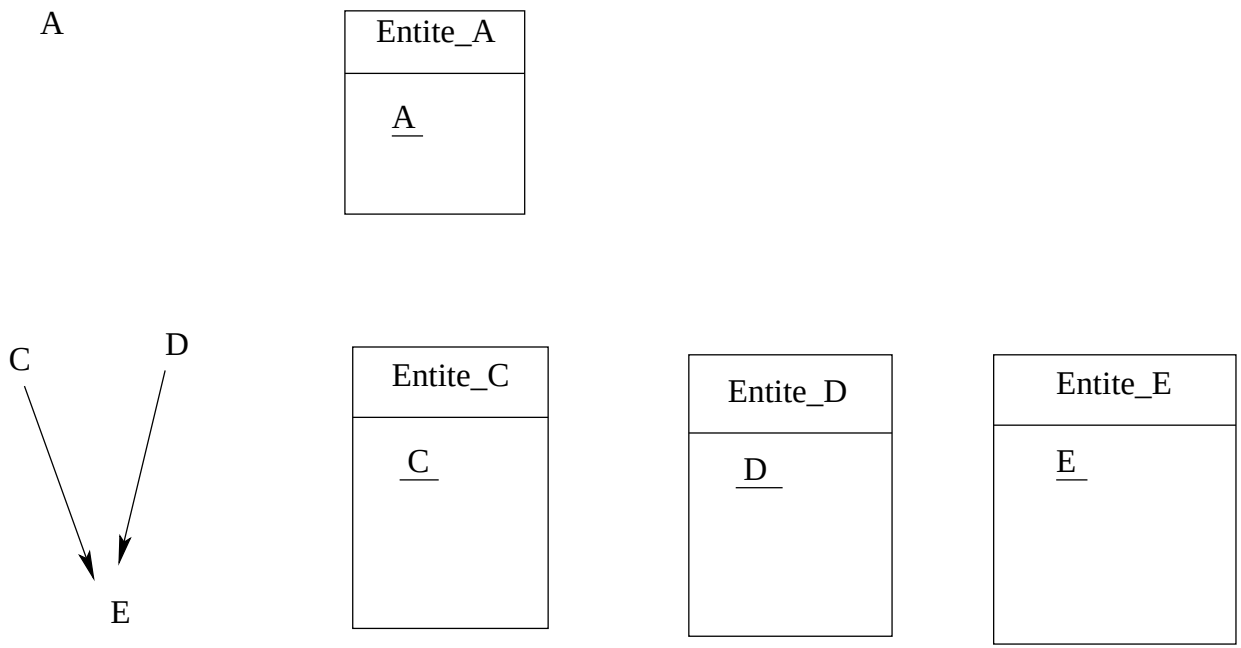






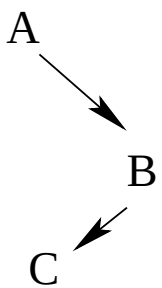


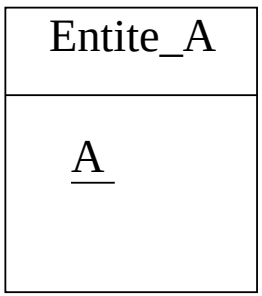
Déterminer les entités (suite)

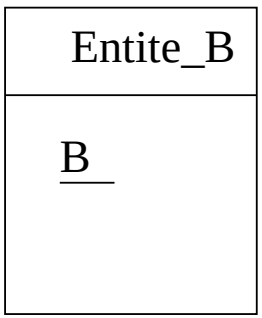


Déterminer les CIF binaires

Une dépendance fonctionnelle simple, dont la source et le but sont des rubriques sources, est une CIF binaire.

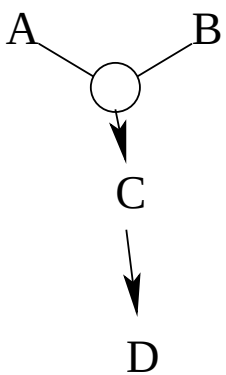


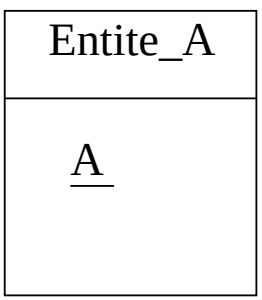


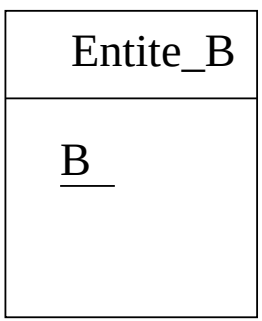


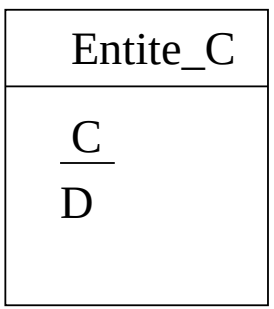
Déterminer les CIF multiples

Une dépendance fonctionnelle à partie gauche composée, dont le but est une rubrique source, est une CIF multiple.



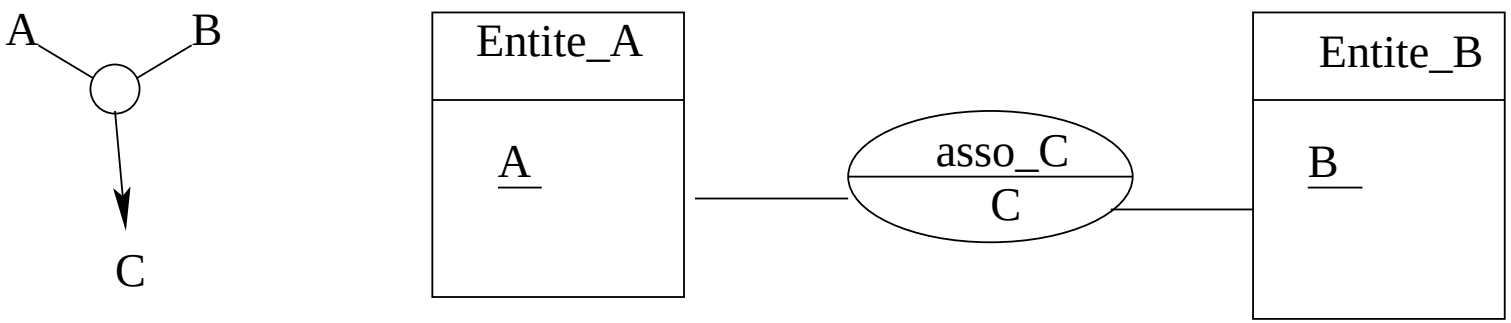






Déterminer les associations

- Une dépendance fonctionnelle à partie gauche composée dont le but n'est pas une rubrique source donne une association multiple.

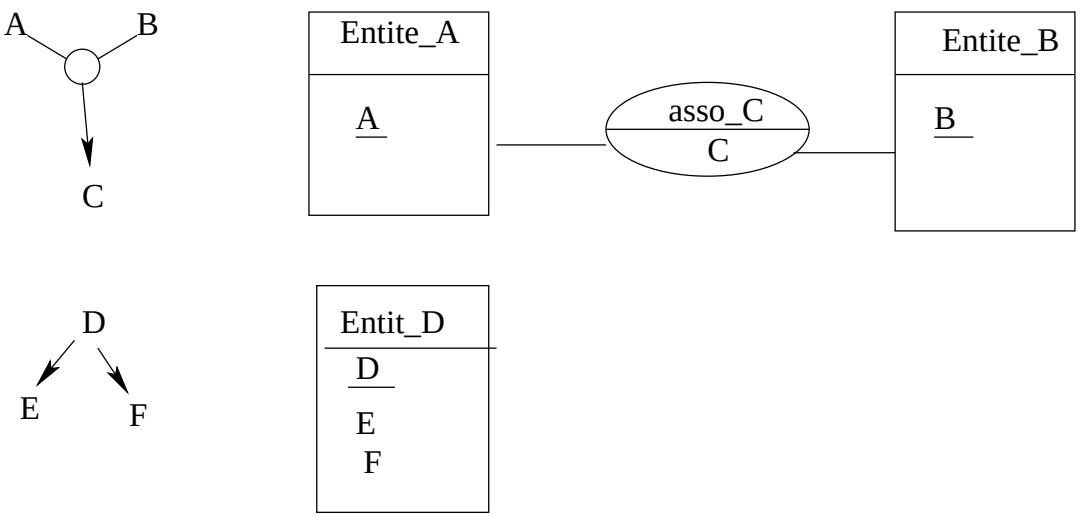


- Attention parfois le contexte peut faire surgir des associations entre deux rubriques sources non reliées par une dépendance fonctionnelle.

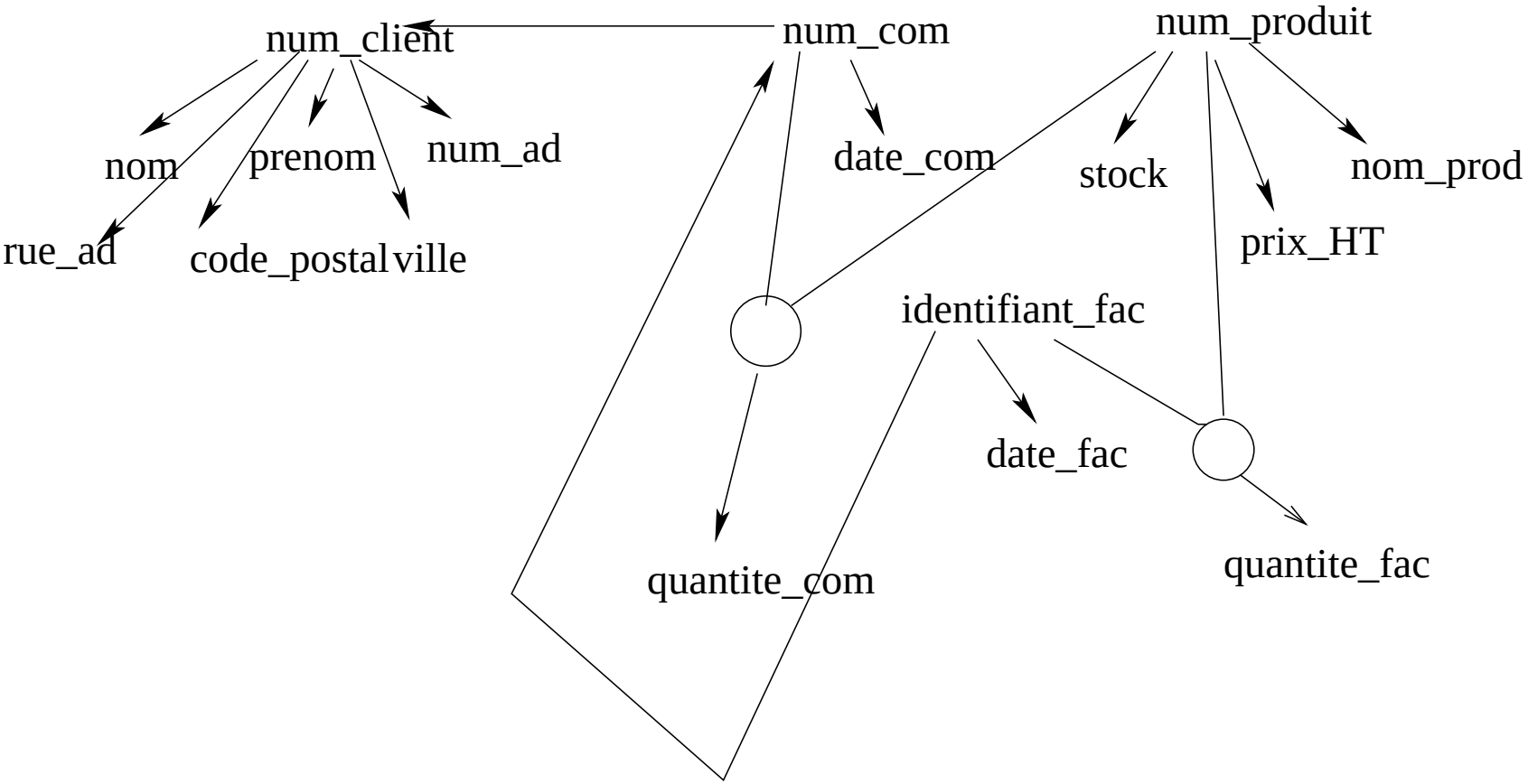
Répartir les propriétés entre les entités et les associations

Les rubriques non-sources sont associées à :

- une entité si elles sont la partie droite d'une dépendance fonctionnelle simple ;
- une association si elles sont la partie droite d'une dépendance fonctionnelle à partie gauche composée.



Hop on applique au graphe des df des «trois belges »



En résumé : règles à se rentrer dans le crâne

- Chaque entité possède un identifiant.
- Toutes les propriétés autres que l'identifiant doivent dépendre fonctionnellement (dépendance élémentaire) de celui-ci ($\Longrightarrow$ elles ne peuvent donc prendre qu'une valeur pour une entité – ex. des enfants – et ne jamais être nulle).
- Toutes les propriétés d'une relation doivent dépendre complètement de l'identifiant de la relation (exemple de date et de la chambre d'hotel) et prend donc une et une seule valeur.
- Pas de doublon de propriétés, pas de propriété calculée.

En résumé : formellement

- Toutes les propriétés doivent être élementaires, c'est-à-dire non décomposable.
- Chaque objet possède un identifiant et un seul.
- Les propriétés d'une entité autre que l'identifant doivent être en dépendance fonctionnelle de cet identifiant.
- Une propriété ne peut qualifier qu'une seule entité ou qu'une seule relation.
- Toute dépendance fonctionnelle transitive doit être écartée.
- Pour chaque occurrence d'une relation, il doit exister une et une seule occurrence par entité participant à la relation.
- Les propriétés doivent dépendre de la totalité de l'indentifiant de la relation. Si ce n'est pas le cas, il faut éclater la relation en autant de relations que nécessaire.

Normalisation

Première forme normale (1FN)

Dans une entité, toutes les propriétés sont élémentaires et il existe au moins une clé caractérisant chaque occurrence de l'objet représenté. Si cette clé est unique elle sera choisie comme identifiant, s'il y en a plusieurs on fait un choix.

- Plus d'entité client avec juste nom et prénom.
- Plus de propriété adresse (on fera une exception pour les dates).
- Dictionnaire élémentaire.
- Source des dépendances fonctionnelles et association à l'entité.
- Gagné.

Normalisation (suite)

Deuxième forme normale (2FN)

Toute propriété d'une entité doit dépendre fonctionnellement de l'identifiant par une dépendance fonctionnelle élémentaire.

- Plus d'entité facture avec le nom du client.
- Source des dépendances fonctionnelles élémentaires et association à l'entité.
- Gagné.

Normalisation (suite)

Troisième forme normale (3FN)

Toute propriété d'une entité doit dépendre fonctionnellement de l'identifiant par une dépendance fonctionnelle élémentaire directe.

- Plus d'entité livre(num_ISBN, id_Style, descriptif_Style).
- Source des dépendances fonctionnelles élémentaires directe et association à l'entité.
- Gagné.

Normalisation (suite)

Forme normale de Boyce-Codd (BCNF)

Si une **entité** a un identifiant concaténé, un des éléments composant l'identifiant ne doit pas dépendre d'une autre propriété.

- Dans un restaurant, un menu est composé d'une entrée, d'un plat principal et d'un dessert. Un client ne peut choisir qu'un seul plat par catégorie. On serait tenté de faire une entité element_repas(id_client, id_catégorie, id_plat) avec id_client, id_catégorie comme identifiant. Mais on a id_plat $\xrightarrow{df}$ id_categorie, ce n'est donc pas BCFN. La bonne décomposition est plat(id_plat, id_catégorie, descriptif_plat, prix) en relation avec l'entité client.
- Là on ne gagne pas, il faut vérifier :'(

Normalisation (le retour du come back)

- On peut encore faire mieux, on peut imposer qu'une propriété dépende fonctionnellement de l'ensemble des identifiants participant à la relation, mais d'aucun sous-ensemble de cet ensemble.
- Ce qui est déjà fait lorsqu'on prend des dépendances fonctionnelles élémentaires.

Normalisation (fin)

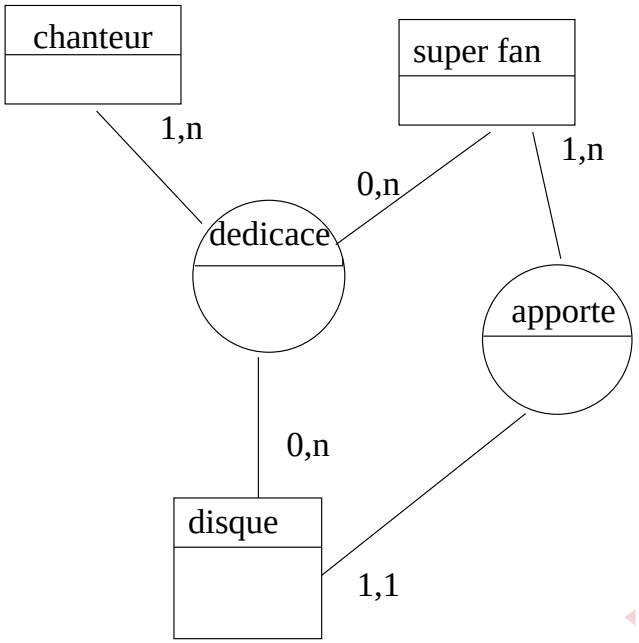
- Un bon MCD est au moins 3FN.
- J'attends de vous des BCNF.
- Il existe des 4FN et 5FN (un tour sur le net vous renseignera).

Décomposition

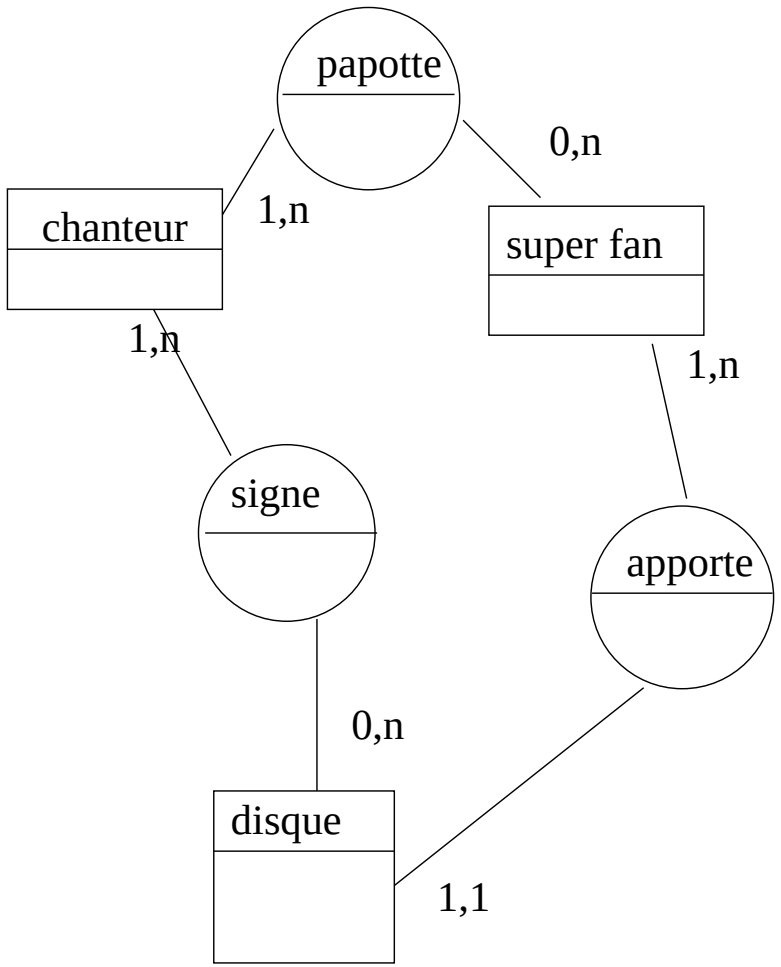
- On peut décomposer une relation n-aire en plusieurs relations de dimension plus petite en utilisant les df que l'on peut trouver.
- Uniquement,
  - si la cardinalité minimum d'une des entités participant à la relation a pour cardinalité minimale 1 ;
  - si la df trouvée dépend d'une relation existant dans notre MCD autre que la relation à décomposer, elle doit concerner les mêmes occurrences d'entités que celle de la relation à décomposer.
- Ce n'est pas une obligation.

Décomposition (exemple)

Une dédicace est organisée pour des chanteurs ou groupes connus. Un fan peut venir faire dédicacer ses disques. Nous imaginerons que le chanteur aura au moins une dédicace à faire (sinon on ne l'aurait pas invité). En revanche à 20 heures tout ferme, un fan peut avoir fait la queue pour rien. On serait tenté de faire le MCD suivant :



Décomposition (exemple – suite)



Extension du formalisme entité/relation

But

Représenter des propriétés liées à certains sous-ensembles d'occurrences d'entités ou de relations. 2 concepts :

- Généralisation/spécialisation.
- Représentation de nouvelles contraintes.

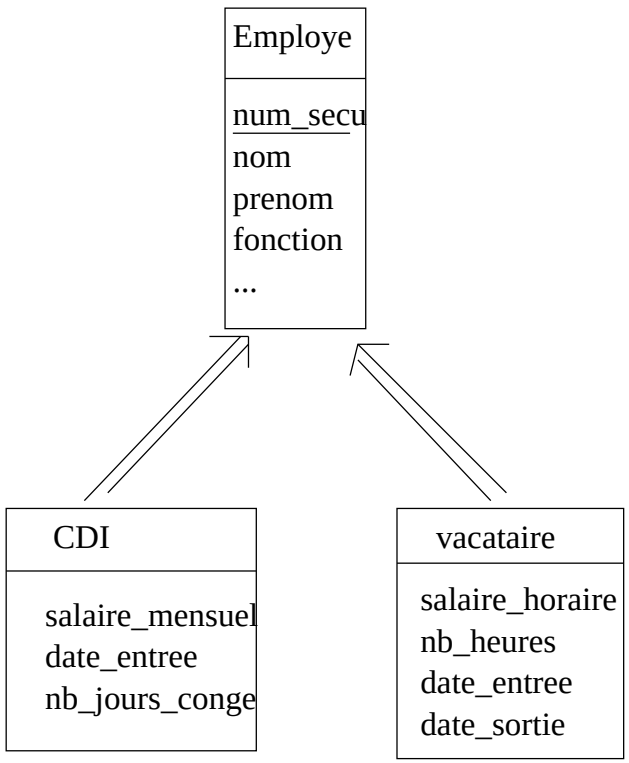
Spécialisation

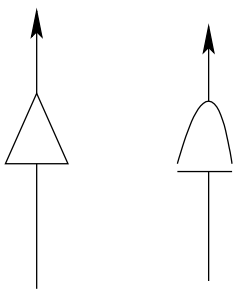
- Exemple : Membres du personnel universitaire. Ils ont tous un nom, un prénom, une adresse, une situation maritale, un nombre d'enfants à charge, une fonction etc.
  - Ceux qui travaillent en CDI possèdent une date d'entrée dans l'Université, un salaire mensuel, un nombre de jours de congés à prendre.
  - Les vacataires eux sont payés à l'heure, ont une date d'entrée dans l'Université et une date de sortie, on conserve aussi le nombre d'heures cumulées de vacation.

Spécialisation (suite)

- Impossible de faire une unique entité : certaines propriétés n'ont pas de sens pour certaines occurrences.
- On peut faire 2 entités distinctes. Dans ce cas, il y a redondance d'information.
- Solution la spécialisation : regrouper dans une entité générique (sur-type) les propriétés communes et créer deux entités spécialisées (sous-type) ne contenant que les propriétés spécifiques.

Spécialisation (suite)

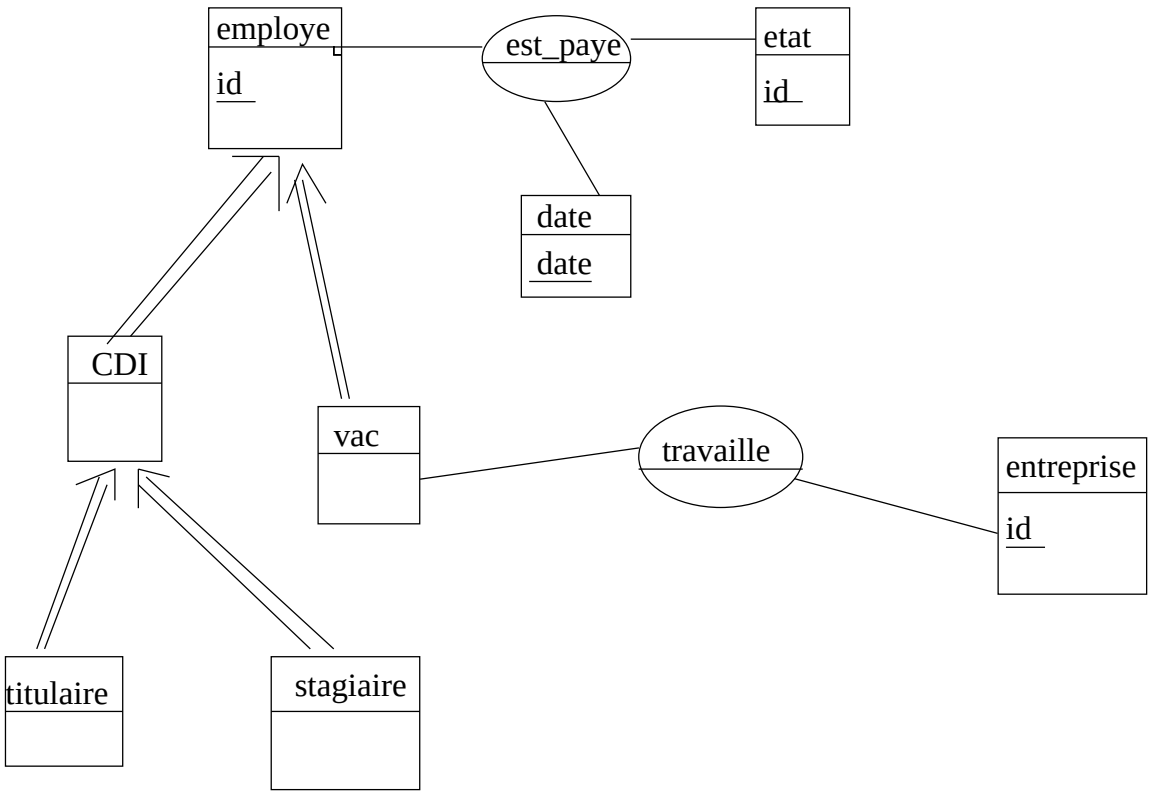




Spécialisation (suite)

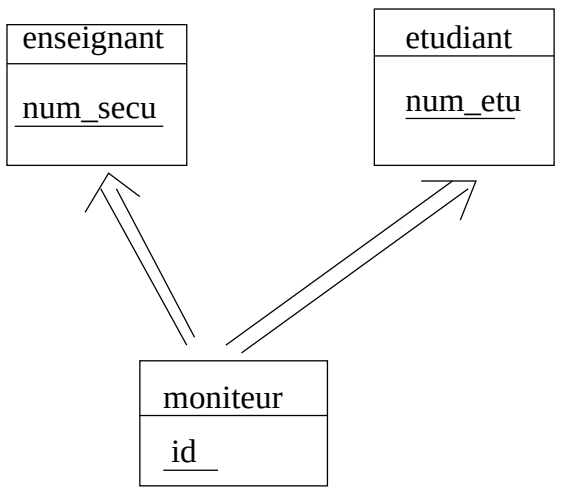
- La spécialisation permet de représenter les entités du monde réel possèdant des propriétés communes.
- L'entité primordiale est le sur-type qui définit l'identifiant.
- Les sous-types héritent des propriétés du sur-type.
- Les occurrences d'un sous-type sont également les occurrences du sur-type.
- Les associations dans lesquelles intervient le sur-type concernent aussi le sous-type.
- Les associations dans lesquelles intervient directement un sous-type ne concernent pas le sur-type.
- Un sous-type peut aussi être un sur-type.

Spécialisation (suite)

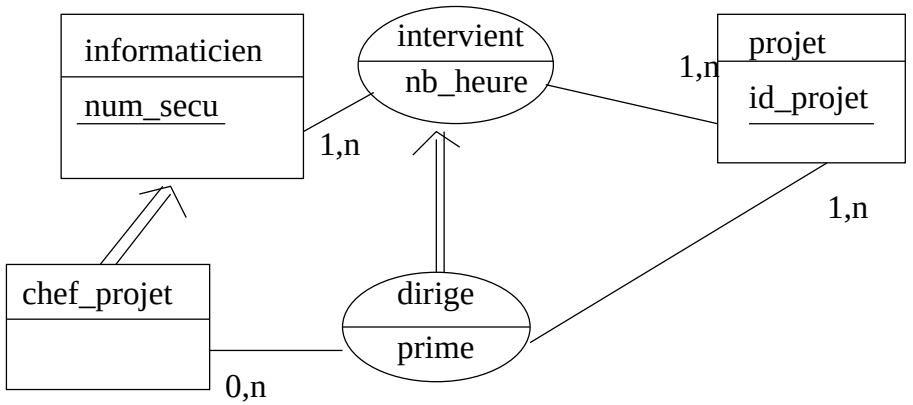


Généralisation

- L'idée est la même, sauf que la conception est l'inverse.
- C'est le sous-type qui est l'entité primordiale.
- Le sous-type a un identifiant propre .
- Ce qui permet une généralisation multiple (i.e. un sous-type qui définit plusieurs sur-types).
- Exemple des moniteurs qui sont des étudiants préparant un doctorat et donnant des cours à l'Université.



Surtype et restriction

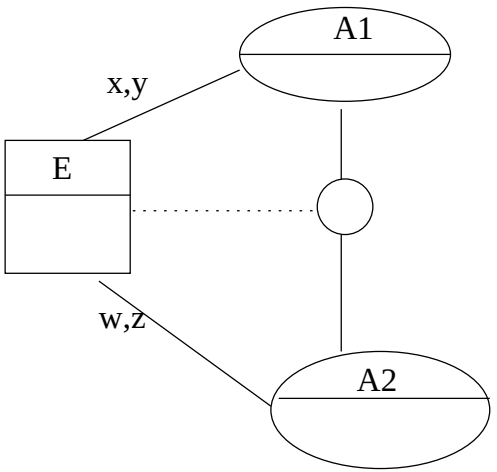


- Un chef de projet est un informaticien particulier.
- Les informaticiens interviennent sur un ou plusieurs projets.
- dirige est une restriction de intervient.
- dirige garde les propriétés de intervient (nb_heure) et possède ses propres propriétés (prime).

Contrainte sur les relations ou entités

- Contrainte d'inclusion (I).
- Contrainte d'exclusion (X).
- Contrainte de totalité (T).
- Contrainte de ou exclusif (+).
- Contrainte d'égalité (=).

Représentation



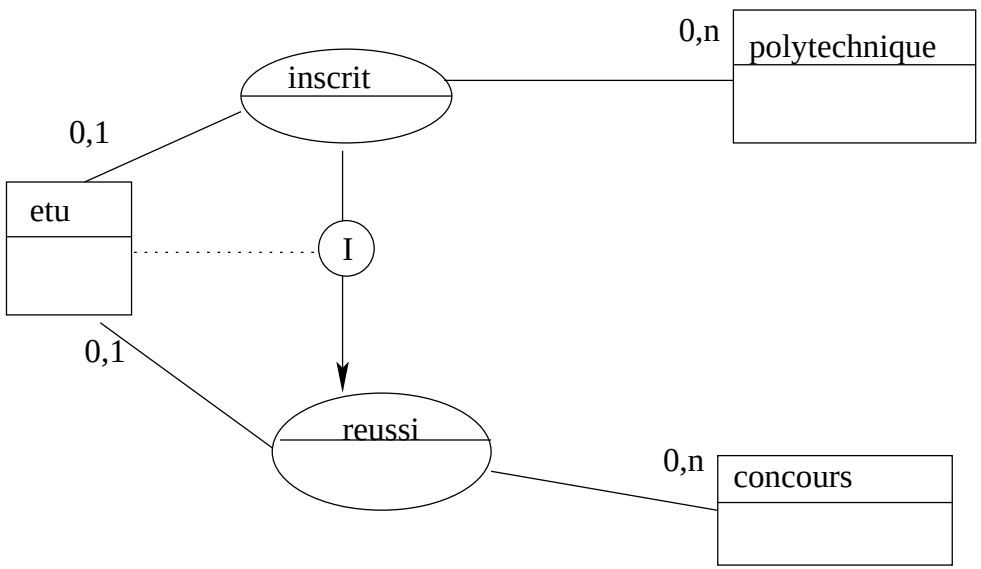
- Le cercle contient le type de la contrainte.
- Le trait en pointillé indique l(es)'entité(s) liée(s).
- Les traits pleins indiquent les relations liées.
- Il peut y avoir une flèche pour indiquer le sens de la contrainte.

La contrainte d'inclusion

- Une contrainte d'inclusion d'une relation 1 vis-à-vis d'une relation 2 exprime que toutes les occurrences des entités concernées par la contrainte qui participent à la relation 1 doivent participer à la relation 2.

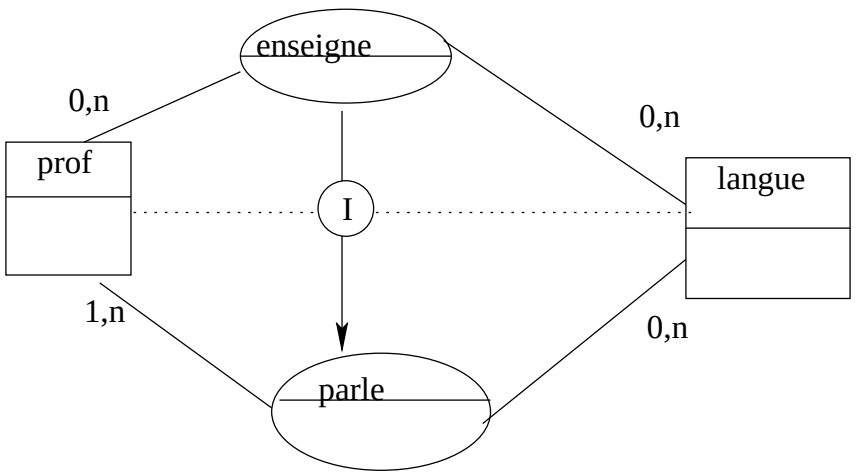
La contrainte d'inclusion (suite)

- Tous les étudiants de polytechnique doivent avoir réussi le concours d'entrée.



La contrainte d'inclusion (suite)

- Un professeur ne peut enseigner qu'une langue qu'il parle.
- Et une langue ne peut être enseignée que par un professeur qui la connaît.

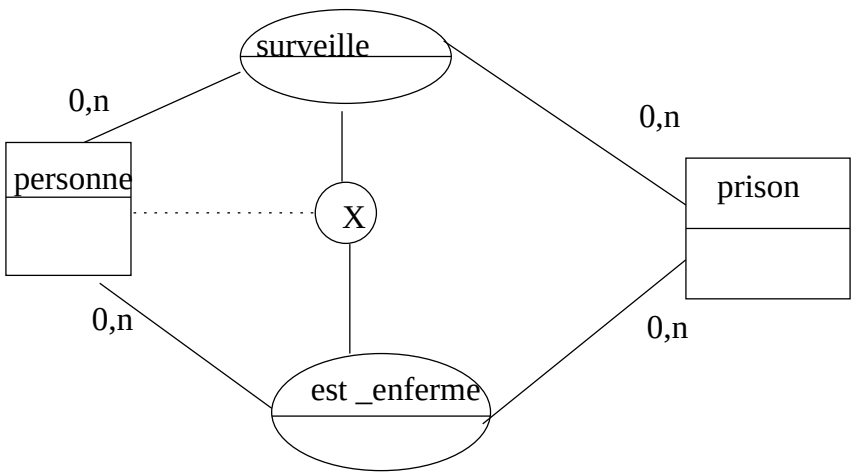


La contrainte d'exclusion

- La contrainte d'exclusion d'une relation 1 vis-à-vis d'une relation 2 exprime que toutes les occurrences des entités concernées par la contrainte qui participent à la relation 1 (resp. 2) ne peuvent pas participer à relation 2 (resp. 1).

La contrainte d'exclusion (suite)

- Un prisonnier ne peut pas être un gardien de prison en activité.

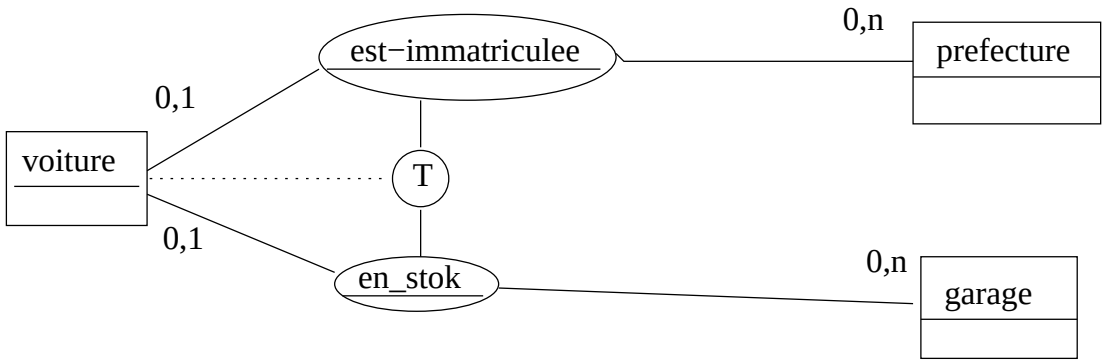


La contrainte de totalité

- La contrainte de totalité d'une relation 1 vis-à-vis d'une relation 2 exprime que toutes les occurrences des entités concernées par la contrainte participent soit à la relation 1, soit à la relation 2, soit aux deux relations, mais au moins une.
- On l'appelle aussi contraite de ou inclusif ou contrainte couverture .

La contrainte de totalité (suite)

- Suivant la loi française un véhicule doit être immatriculé, sinon c'est qu'il n'a pas encore été vendu.
- Il existe des immatriculations de garage.

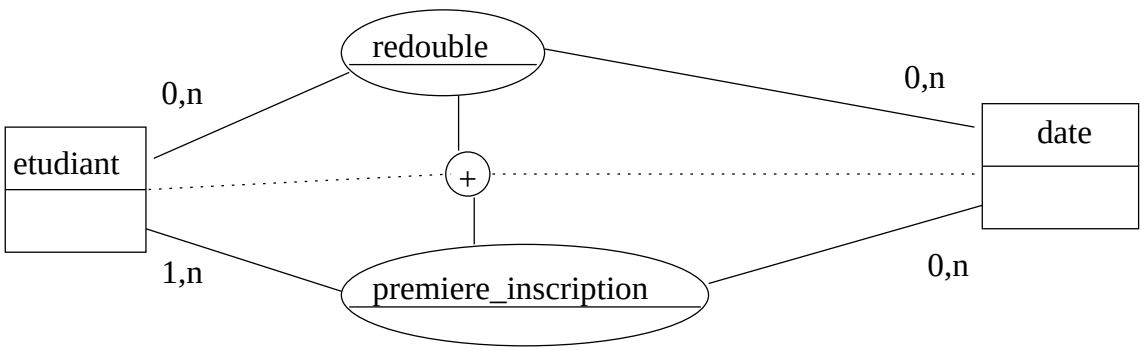


La contrainte de ou exclusif

- La contrainte de ou exclusif d'une relation 1 vis-à-vis d'une relation 2 exprime que toutes les occurrences des entités concernées par la contrainte participent soit à la relation 1, soit à la relation 2, pas au 2, mais à au moins une.
- On l'appelle aussi la contrainte d'exclusivité et de totalité, elle est aussi noté XT.

La contrainte de ou exclusif (suite)

- Un étudiant dans une filière est forcement inscrit.
- Ou c'est sa première année ou il redouble.
- Mais il ne peut pas faire les deux en même temps.
- On considère qu'il n'y a pas d'inscription en ≪enjambement ≫.

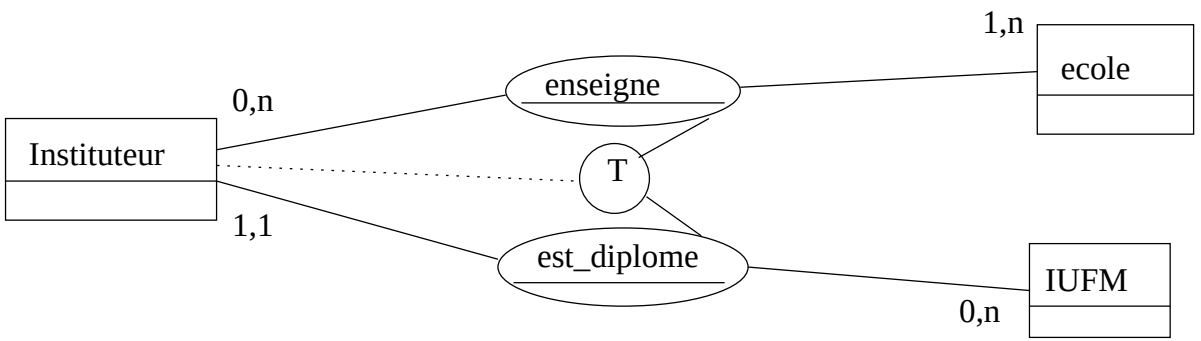


La contrainte d'égalité

- La contrainte d'égalité d'une relation 1 vis-à-vis d'une relation 2 exprime que toutes les occurrences des entités concernées par la contrainte participant à l'une des relations participent aussi à l'autre.
- On l'appelle aussi la contrainte de simultanéïté, elle est aussi notée S.

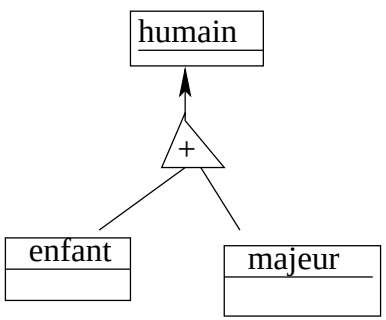
La contrainte d'égalité (suite)

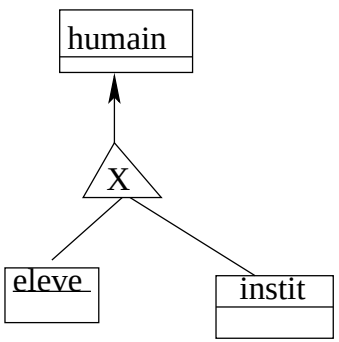
- Tout instituteur d'une école française est diplomé d'un des IUFM de France.

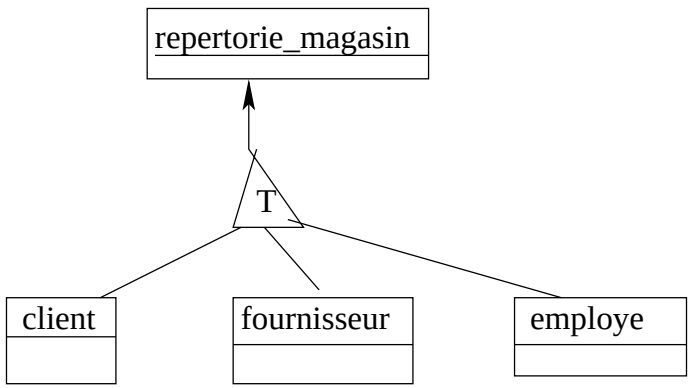


Spécialisation et contrainte

- On peut appliquer les contraintes de ou exclusif (+), d'exclusion (X) et de totalité (T) aux spécialisations d'entités.





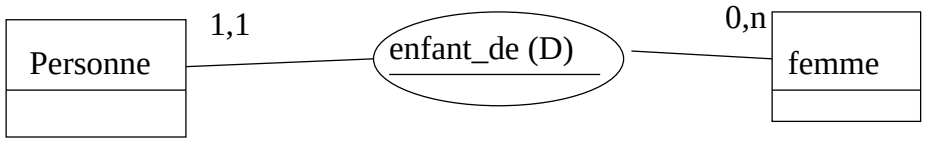


Contrainte de stabilité sur une relation

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

Contrainte de stabilité sur une relation (suite)

- On est indéfiniment l'enfant de sa mère.

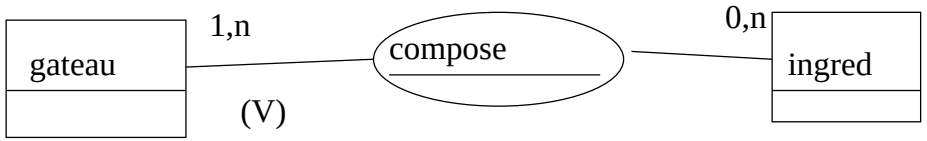


Contrainte de stabilité sur une patte

- Cette contrainte se nomme verrouillage (noté (V)). Soit une entité E, une relation R et une patte P reliant E à R. P est verrouillé si toute les occurences de R dans les quelles intervient une occurence occ de E doivent être crée en même temps que occ.

Contrainte de stabilité sur une patte

- Un gateau n'existe que si des ingrédients le compose.

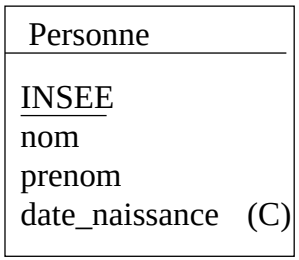


Contrainte de stabilité sur une propriété

- Une propriété peut-être déclarée constante (noté (C)) si pour une occurence donnée, sa valeur ne varie jamais.

Contrainte de stabilité sur une propriété (suite)

- Notre date de naissance est immuable.



Et le MCD dans tout ça ?

- Avant d'utiliser les extensions, on fait son MCD normal.
- Ensuite, on cherche les héritages et autres propriétés.
- On transforme alors notre MCD, qui devient un MCD analytique (MCDA).

Niveau organisationnel – Données

Niveau organisationnel

- Définir l'organisation que m'on doit mettre en place pour atteindre les objectifs décrits au niveau conceptuel.
- Prise en compte de l'organisation.
- Qui fait QUOI, QUAND et OÙ.
  - Répartition homme-machine.
  - Mode de fonctionnement (temps réel ou temps différé).
  - affectation des données et des traitements.

Niveau organisationnel (suite)

- 2 modèles
  - Modèle Organisationnel des Données (MOD) – représentation de l'ensemble des données par type de site (Même formalise que le MCD).
  - Modèle Organisationnel des Traitements (MOT) – représentation par des procédures les phases et les taches effectuée par chaque poste de travail.

MOD

- Au commencement des temps . . . cela n'existait pas.
- En effet, tout était centralisé, c'était donc inutile. Depuis, grand développement de la technologie client-serveur. On a donc besoin de savoir qui fait quoi.

Les grands principes

- Données retenues au niveau organisationnel. On ne conserve que ce qui subira un traitement automatique.
- Droit d'accès aux données. On doit définir les droits d'accès et de mise à jour des données par type d'utilisateur.
- Visibilité des données par site organisationnel. En cas de multicité des sites, qui voit quoi.
- Volumétrie des données. Volume des données actives (mises à jour), volume et conditions d'archivage des données passives (non mises à jour).

- Ce qui nous donne un MOD global et des MOD par site (dans le cas de système réparti sur plusieurs sites)
- Même formalise que pour le MCD.

Identification des types de site et des types d'acteur

- Comprendre l'organisation de l'entreprise. En particulier, sa structuration par site organisationnel et la typologie des acteurs par site.
- Type d'acteur : ensemble d'occurrence d'acteurs travaillant sur un même type d'activité dans une organisation donnée, par exemple le service comptabilité.
- type de site : ensemble de type d'acteurs regroupés selon un critère fonctionnel et/ou organisationnel, par exemple une agence régionale.

Exemple PME

Soit une PME spécialisée dans la mise à disposition de personnes pour le compte de ses clients.
Chaque intervention donne lieu à un contrat par le client ; les principales informations du contrat sont : la description succincte de l'intervention, la date de début de l'intervention, la qualification précise de chaque intervenant (il existe une vingtaine de qualifications possibles), le nombre de jours $\times$ hommes prévus.

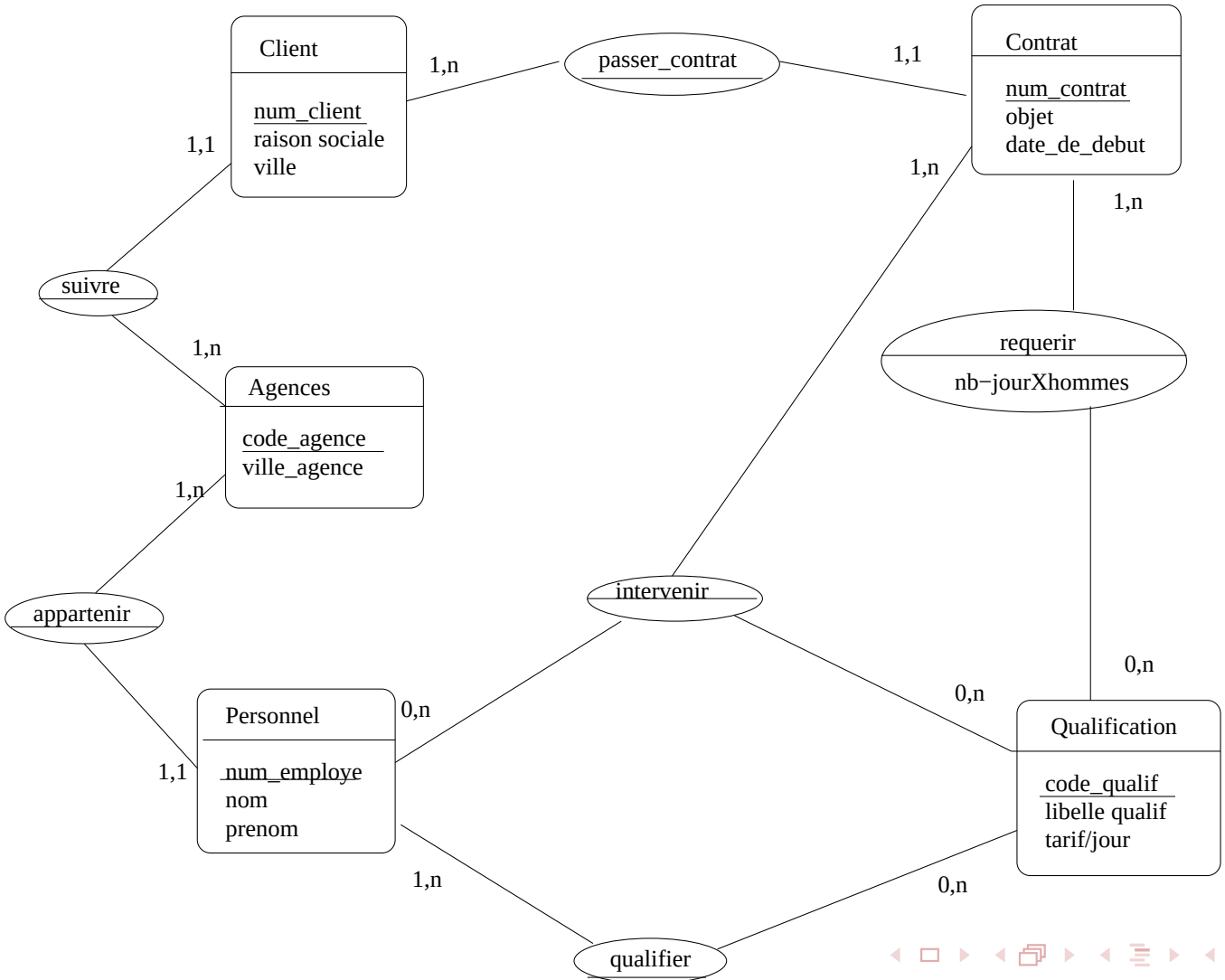
Exemple PME (suite)

À chaque qualification correspond un tarif journalier. La PME s'accorde en interne une certaine souplesse sur la détermination précise de la qualification de son personnel en procédant de la manière suivante : Chaque personne possède a priori une qualification de base. À chaque intervention, il est possible de réajuster la qualification dite d'intervention par rapport à la qualification de base. La qualification d'intervention est déterminée pour un contrat donné. La qualification retenue doit toujours appartenir à l'ensemble des qualifications standard.

Exemple PME (encore)

La PME possède un siège et des agences. Le siège est chargé de l'éllaboration des contrats à partir des propositions des agences. Le siège comprend lui-même un service des contrats et un service financier. Les agences sont chargées d'une part de la relation clientèle gérée par le service clientèle et d'autre part d'un service recrutement des personnels réalisé par la direction de l'agence. Il est précisé que les clients sont répartis par agences. Enfin, le siège assure un suivi statistique des contrats soldés en ne tenant plus compte des clients, mais seulement des agences concernées.

Exemple PME (MCD)



Exemple PME : sites et acteurs

Type de site	Type d'acteur
Siège	Service des contrats Service financier
Agence	Service clientèle Direction de l'agence

Exemple PME : données

Données existantes au niveau conceptuel	Données retenues au niveau organisationnel
num_client	X
raison_sociale	X
ville	X
num_contrat	X
objet	X
date_de_debut	X
nb-jourXhommes	X
code_qualif	X
libelle_qualif	X
tarif/jour	X
num_employe	X
nom	X
prenom	X
code_agence	X
ville_agence	X
	Nombre de contrat par agence

Droit d'accès

- En général, large accès en consultation des données. Sauf :
  - données sensibles (par exemple, données nominatives – s'en référer à la CNIL) ,
  - raison stratégique (par exemple, information salariales, certaines informations financières)
- Droits d'accès en mise à jour représente un enjeu fondamental.
  - S'assurer de l'unicité de la responsabilité de la mise à jour d'une donnée.
  - Souvent utile de séparer la création d'une occurrence à sa mise à jour.

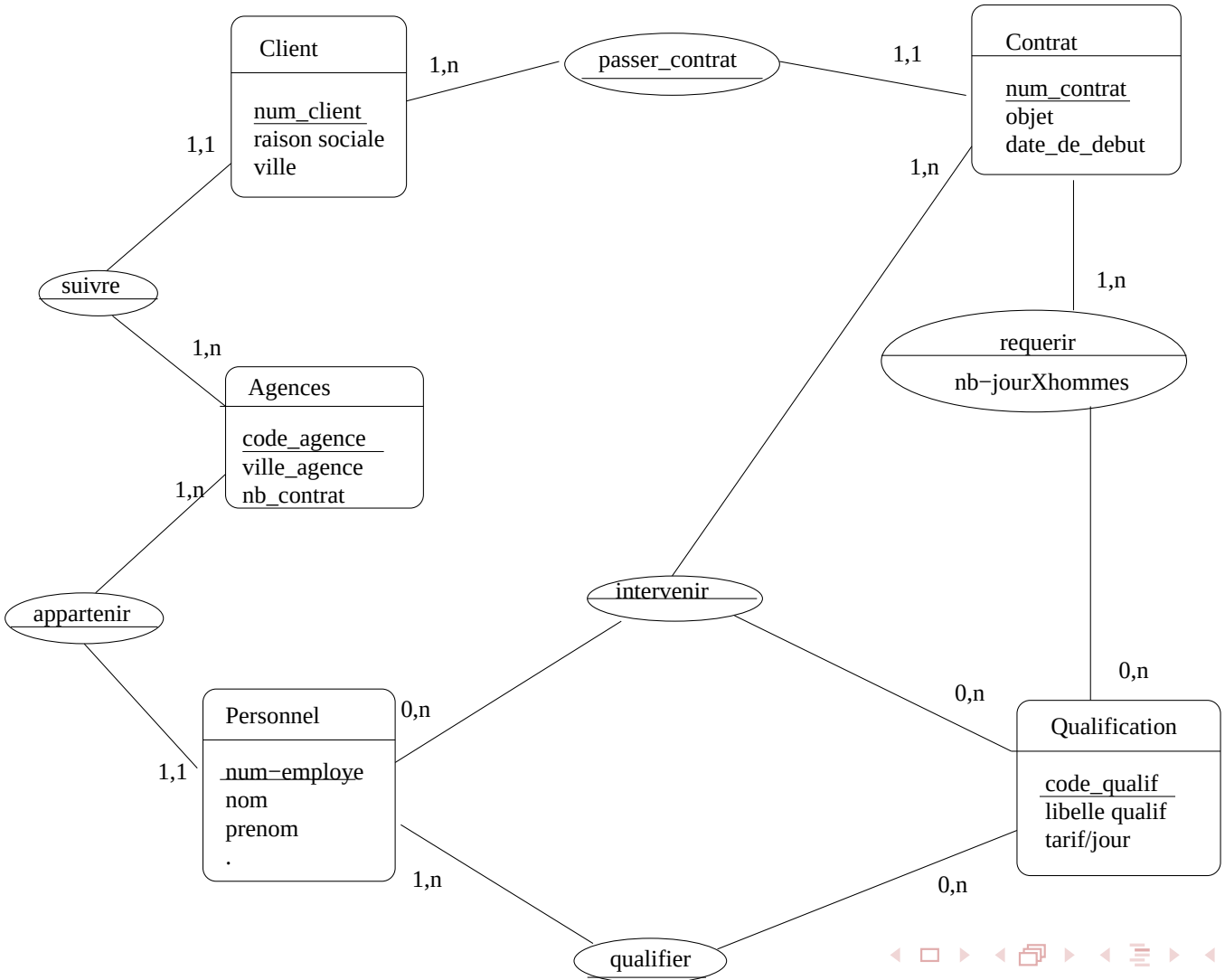
Exemple PME : droits d'accès

Entité-Relation	Service contrat		Service financier		Service clientèle		Direction agence	
	consul.	M.A.J.	consul.	M.A.J.	consul.	M.A.J.	consul.	M.A.J.
Client	X	X	X		X		X	X
Contrat	X	X	X		X		X	
Agence	X	X	X					
requerir	X	X	X		X		X	
intervenir	X	X	X		X		X	
passer_contrat	X	X	X		X		X	
Qualifiaction	X		X		X		X	X
Personnel	X		X		X		X	X
qualifier	X		X		X		X	X
suivre	X	X	X					
appartenir	X		X	X				

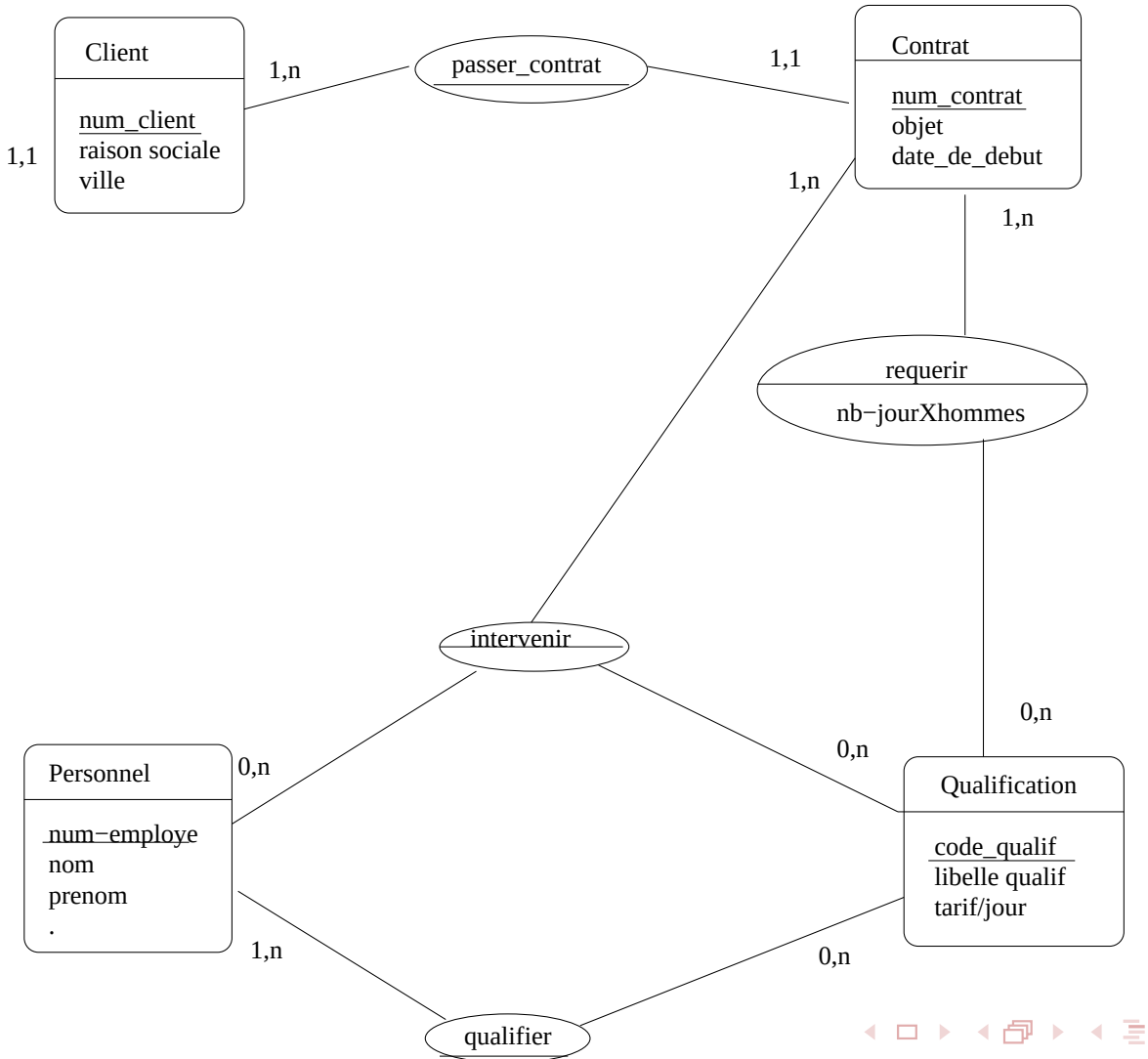
Visibilité des données par site organisationnel

- Dans le cas d'un SI destiné à plusieurs sites organisationnels, la visibilité de chaque type de site doit être définie. On découpe le MOD générale (i.e. le MCD avec les nouvelles données et relations) en MOD par type de site en précisant :
  - les entité et relation en consultation ou mise à jour,
  - les occurrences concernées dans le cas ou seul un sous ensemble d'occurrence est spécifique à un site. Cette opération s'appelle la fragmentation .

Exemple PME : MOD du siège



Exemple PME : MOD d'une agence



Exemple PME : MOD d'une agence (suite)

- La fragmentation :
  - la restriction des occurrences des entités Client, Contrat et personnel aux occurrences propres à une agence donnée,
  - la restriction des occurrences des relations passer_contrat, requérir, intervenir et qualifier aux occurrences concernées par les occurrences des objets intervenant dans chaque relation.

Niveau logique – Données

Niveau logique

- prise en compte des choix logiques.
- Hourra on passe au niveau de l'≪informatique≫.
  - Pour les données : prendre en compte la structuration technique propre au stockage informatisé. Merise adapté au modèle relationnel $\Longrightarrow$ bases de données relationnelles.
  - Pour les traitement : structuration en unité de traitement (temps réel, temps différé).

MLD

- Plusieurs formalismes :
  - Le formalisme CODASYL et ses dérivés pour les bd hierarchiques, réseaux ou fichiers.
  - Le formalisme Codd pour les bd relationnelles.
- Nous nous intéresserons au formalisme de Codd.

MLD – Présentation des concepts du modèle relationnel

- Domaine : l'ensemble de valeurs que peut prendre une donnée. Par exemple, la donnée marque sera définie sur le domaine D_m = Renault, Peugeot ; couleur sur D_c = Blanc, bleu, vert.
- Relation ou encore table : un sous-ensemble du produit cartésien de domaines. Notion de tuple .
- Attribut représente une colonne d'une table caractérisée par un nom. Par exemple, marque et couleur sont des attributs de voiture.
- Clé d'une relation : une clé est constituée d'un ensemble d'attributs dont les valeurs définissent de manière unique les tuples de la relation. Ce sont nos identifiants.

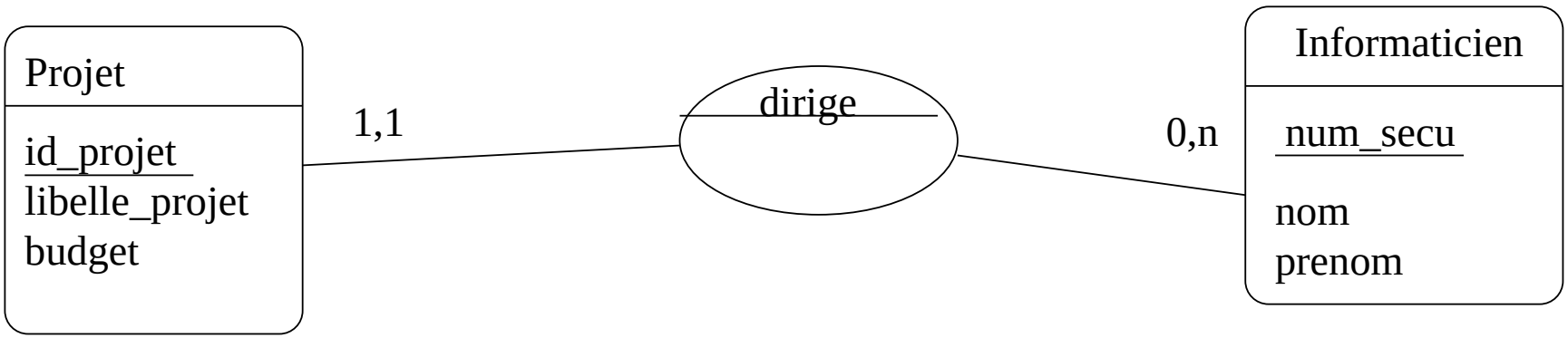
MLD – Règles pour les entités du MCD

- L'entité se transforme en une table .
- L'identifiant devient la clef primairede la table.
- Les propriétés deviennent des attributs de la table.

MLD – Règles pour les relations du MCD

- Cas de la relation type père-fils (cardinalié entité ≪père≫ 0,n ou 1, n, cardinalité ≪fils≫ 1,1) :
  - L'entité ≪père≫ devient la table ≪père≫.
  - L'entité ≪fils≫ devient la table ≪fils≫.
  - L'identifiant de l'entité ≪père≫ devient attribut de la table ≪fils≫. Cet attribut est la clé étrangère de la table ≪fils≫.

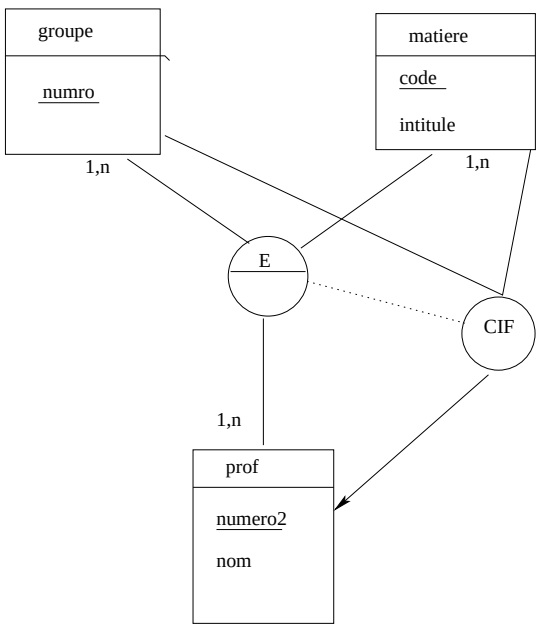
MLD – Règles pour les relations du MCD (suite)



- L'entité «père» est l'entité informaticien.
- L'entité «fils» est l'entité projet.
- Ce qui ne donne le MLD suivant : Informaticien (num_secu, nom, prenom) ; Projet (id_projet,libelle_projet, budget, # *num_secu*).

MLD – Règles pour les relations du MCD

- Les CIFs multiples par l'exemple.

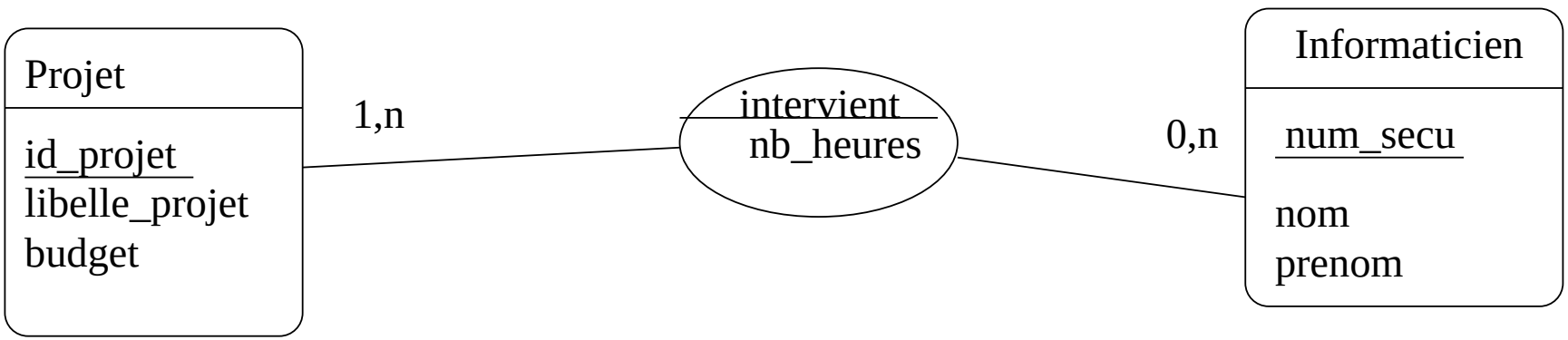


groupe(numero) ; matière(code, intitule) ; prof(numero2, nom) ;
E (# numero,# code,# numero2).

MLD – Règles pour les relations du MCD (suite)

- Autres cas :
  - Une entité devient une table, l'identitifiant de l'identé devient la clé de la table.
  - Une relation devient une table.
  - L'identifiant de la relation (à savoir les identifiants des entités intervenant dans la relation) devient la clé primaire de la table.

MLD – Règles pour les relations du MCD (fin)

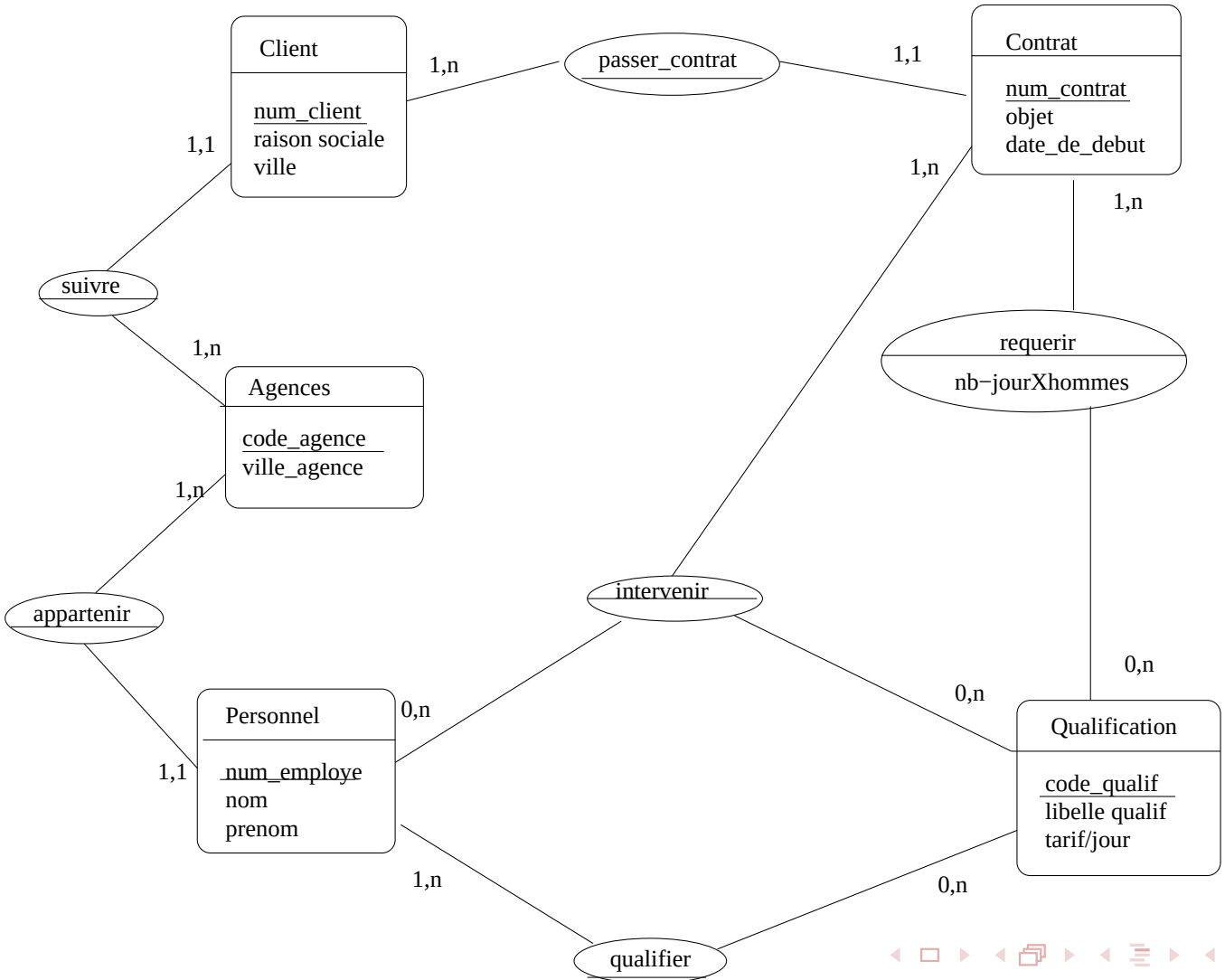


- Ce qui ne donne le MLD suivant : Informaticien (num_secu, nom, prenom) ; Projet (id_projet,libelle_projet, budget) ; intervient (# num_secu,# id_projet, nb_heures).

MLD – Optimisation

- L'objectif poursuivi par l'optimisation est de diminuer le nombre d'accès logique au MLD pour les traitements décrits.
- Le travail d'optimisation consistera à créer le minimum de redondance dans le MLD pour obtenir un nombre minimum d'accès logiques.
- Exemple : ajout d'un attribut, d'une table de lien.

MLD – Exemple


Client
num_client
raison sociale
ville
passer_contrat
Contrat
num_contrat
objet
date_de_debut
suivre
Agences
code_agence
ville_agence
requerir
nb−jourXhommes
appartenir
intervenir
Personnel
num_employe
nom
prenom
Qualification
code_qualif
libelle qualif
tarif/jour
qualifier
1,n
1,1
1,n
1,1
1,n
1,n
1,n
1,n
0,n
0,n
0,n
1,1
0,n
1,n
0,n

MLD – Exemple (suite)

- Client (num_client, raison_sociale, ville, # *code_agence*) ;
- Agence (code_agence, ville_agence) ;
- Personnel (num_employe, nom, prenom, # *code_agence*) ;
- Contrat (num_contrat, objet, date_de_debut, # *num_client*) ;
- Qualification (code_qualif, libellequalif, tarif/jour) ;
- intervenir (# num_contrat, # num_employe, # code_qualif) ;
- qualifier (# num_employe, # code_qualif) ;
- requerir (# num_contrat, # code_qualif, nb-jourXhommes).

MLDA

- Le côté obscure de la force ?
- Parlons-en...

Niveau physique ou opérationnel

Niveau physique

- Répond au COMMENT.
- Met en oeuvre les choix techniques.
- Pour les données (description physique des données –DPD), il s'agit de bases de données (CREATE TABLE etc.).

Niveau conceptuel – Traitements

MCT

- Description de la partie dynamique.
  - Processus.
  - Opération (évènement, résultat et synchronisation).

Les acteurs et les fluxs

- Acteur : entité agissant sur le système d'information (transmet des factures etc.).
- Flux : ce qui est échangé par les acteurs (matière, finance, personnel, matériel ou savoir-faire, information).
- Diagramme de flux :

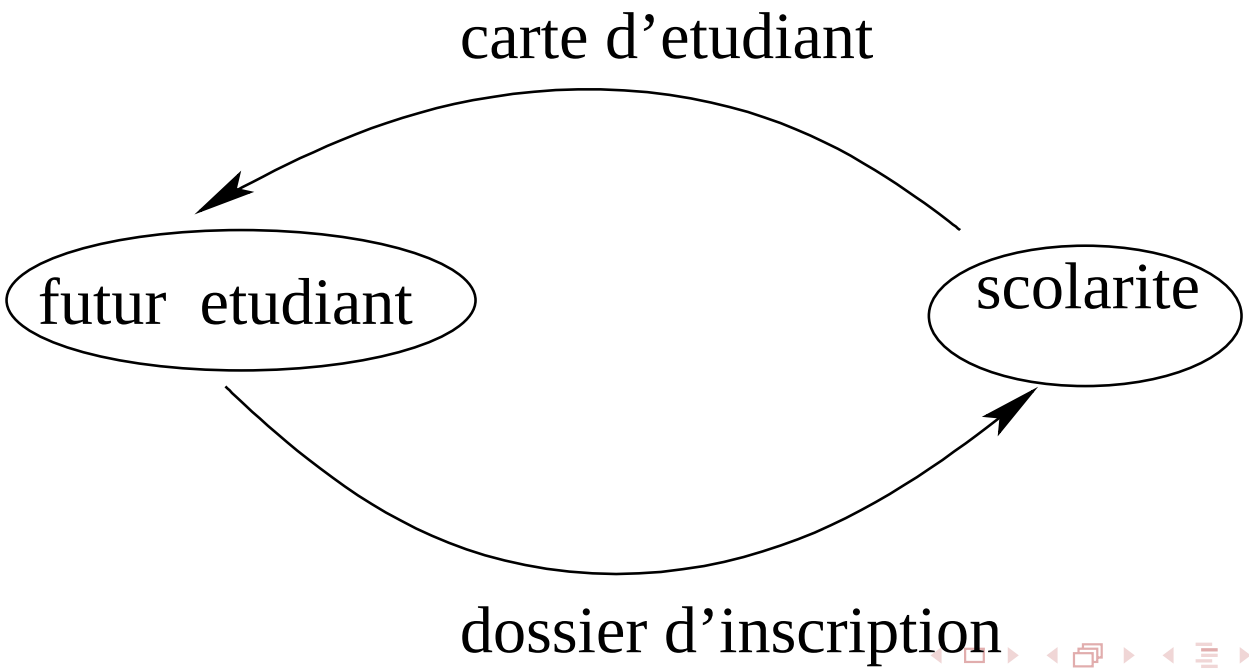


Diagramme de flux : plein en 1

- D'abord on identifie les acteurs externes et leurs Échanges de flux avec l'organisation.
- Ensuite, on décompose l'organisation interne en granularité de plus en plus petite.

Les traitements

- Les traitements représentent les diverses manipulations qu'il faut appliquer aux données pour obtenir les informations recherchées.
- Première étape de description des traitements : le modèle conceptuel des traitements (MCT).
- Le MCT doit permettre de modéliser les différents traitements et leur réalisation dans le temps, sans tenir compte des contraintes d'organisation, ni des moyens matériels et logiciels.
- Le MCT exprime le QUOI, mais ni le QUI, ni le QUAND, ni le OÙ, ni le COMMENT.

Les concepts

- Le MCT se base sur les concepts suivants :
  - Les évènements.
  - Les opérations.
  - La synchronisation.
  - Les résultats.

Les événements

- Quelque chose s'est passé dans l'univers extérieur ou dans le SI lui-même.
- Le système d'information a été sollicité et doit réagir en conséquence.
- 2 types d'événements :
  - Les événements externes (sollicitation externe au système) doivent entraîner une réaction du SI sous forme d'une opération.
  - Les événements internes (dûs au fonctionnement du système) doivent soit entraîner une réaction du SI sous forme d'une opération soit constituer un résultat pour l'univers extérieur.

Les événements (suite)

- L'événement peut être porteur de propriétés . Ces propriétés constituent un mouvement (interne ou externe). Contrairement aux entités les mouvements n'ont pas besoin d'identifiant.
- Si on veut garder trace d'un mouvement, une des opérations en résultant doit en être la mémorisation de ses propriétés.

Exemple

- Les événements externes : une personne dépose un dossier d'inscription à l'université (l'Université subit, elle ne peut pas menacer pour qu'on s'inscrive en ses rangs).
- Les événements internes : on accepte l'inscription de la personne X (C'est l'Université qui décide. Le SI de l'Université génère cet événement. Cet événement est le résultat d'un traitement interne dont l'origine est la réception d'un dossier d'inscription).

Les opérations

- Une opération est définie comme une suite ininterrompue d'actions (i.e. ne nécessitant pas l'arrivée de nouveaux événements) et correspond à un traitement portant sur les données.
  - Elle fait suite à un événement déclencheur (ou une conjonction d'événements) et a pour but d'y apporter une réponse appropriée.
  - Une opération produit en sortie de nouveaux événements.
  - Un ensemble d'opérations peut être regroupé sous l'appellation processus s'il concerne un ensemble cohérent d'activités (gestion des inscriptions, gestion des payes du personnel etc.).

Exemple

- «Envoi d'un accusé de réception» fait suite à l'évènement externe «réception d'un dossier d'inscription».
- «Édition de la carte d'étudiant» fait suite à l'évènement interne «acceptation du dossier d'inscription».
- À un événement peut correspondre plusieurs opérations indépendantes.

La synchronisation

- La synchronisation est une condition logique portant sur les événements, et qui doit être satisfaite pour permettre le déclenchement d'une opération donnée.
- Ceci permet de synchroniser une opération en fonction de plusieurs événements.
- Elle est formée d'une imbrication de ET et de OU.
- La première opération peut se faire sans attente de l'arrivée de l'événement déclencheur, il doit y avoir attente entre les autres, sinon c'est une même opération.

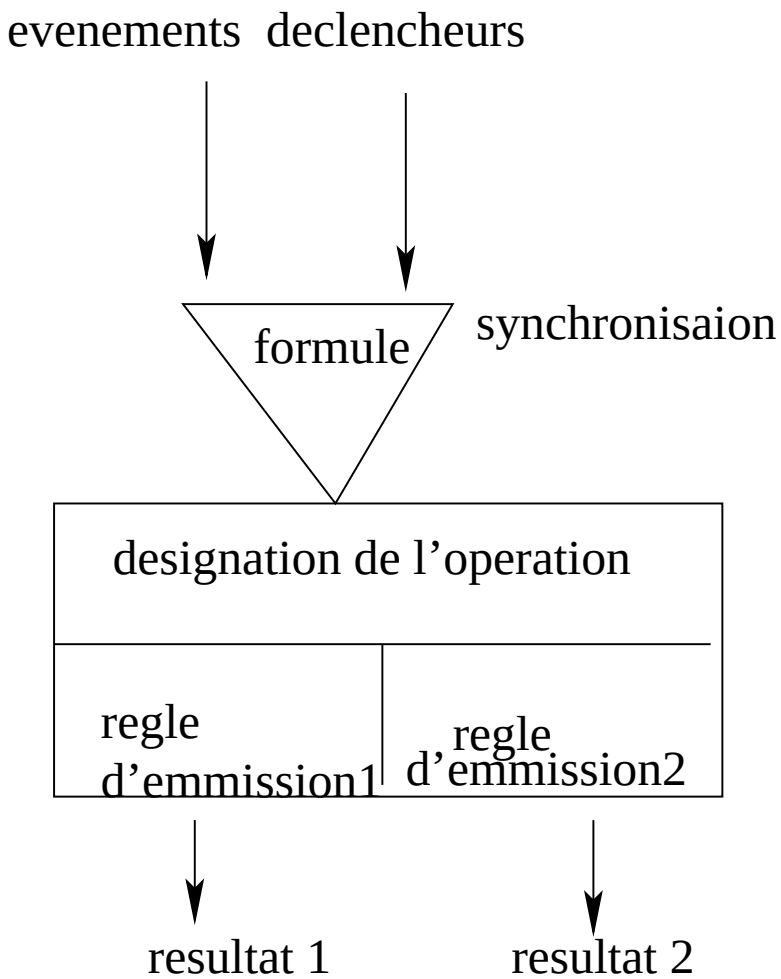
Exemple

- On peut accorder une équivalence de diplôme à un étudiant si ≪on a reçu un dossier ≫ et si ≪le jury a donné son accord ≫.

Le résultat

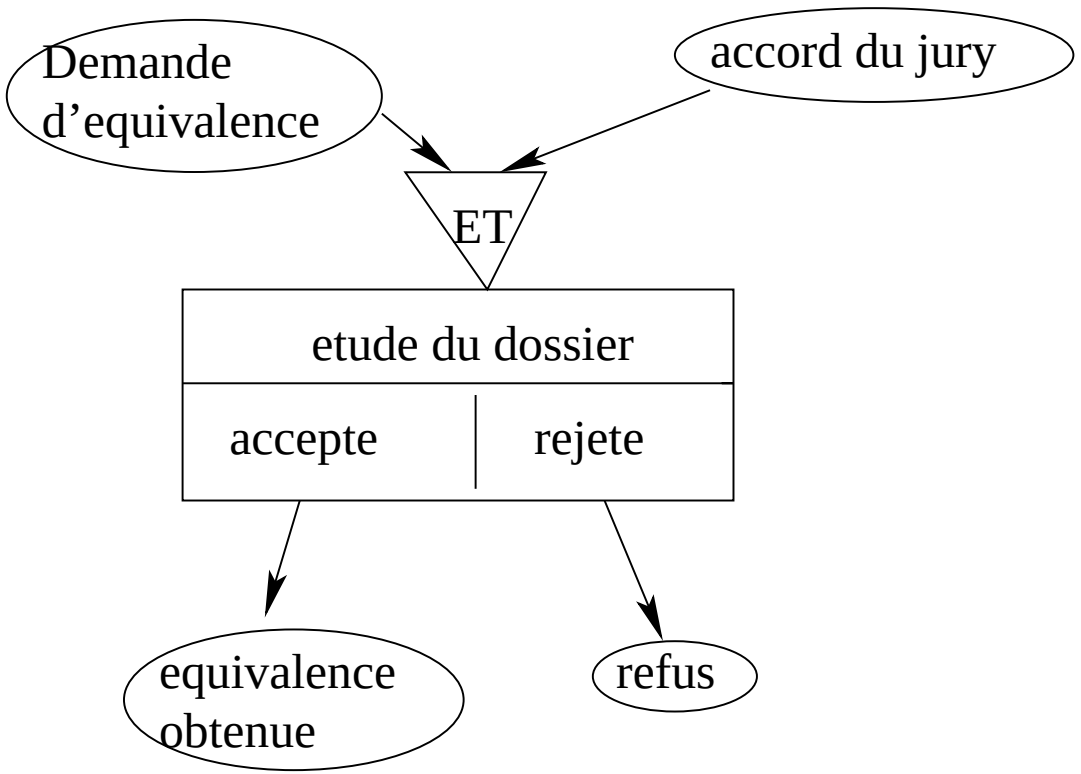
- Le résultat est le produit d'une opération.
- Il peut devenir le déclencheur pour une ou plusieurs opérations
- ≪Édition de la carte d'étudiant ≫ est le résultat de l'acceptation d'un dossier d'inscription.

Formalisme

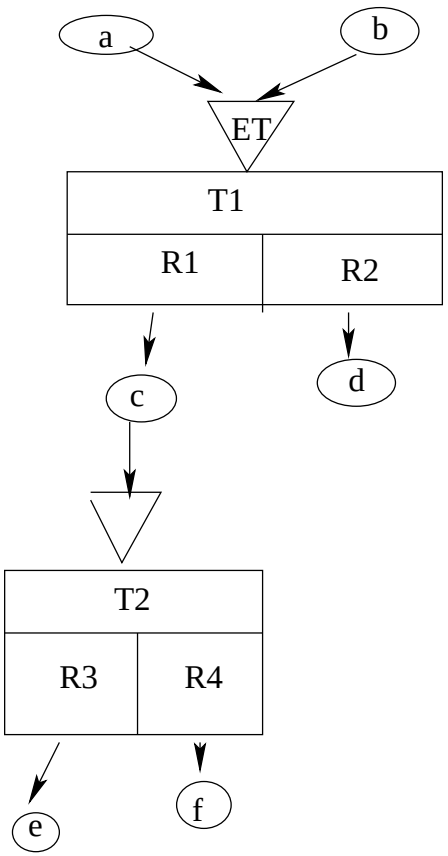

evenements
declencheurs
formule
synchronisaion
designation de l'operation
regle d'emmission1
regle d'emmission2
resultat 1
resultat 2

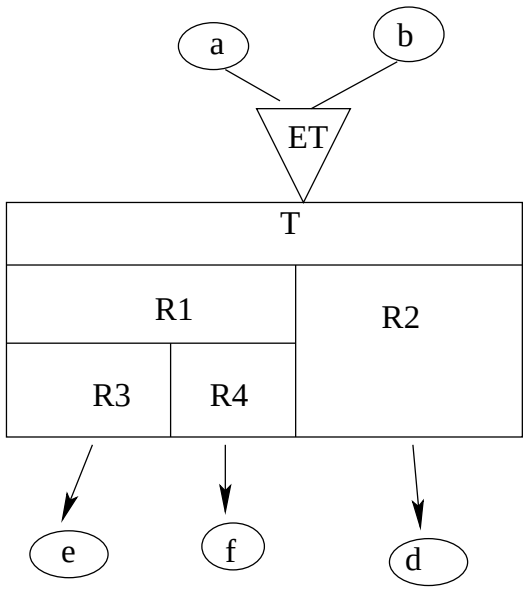
Exemple

- On peut accorder une équivalence de diplôme à un étudiant si ≪on a reçu un dossier ≫ et si ≪le jury a donné son accord ≫.



Concept d'opération



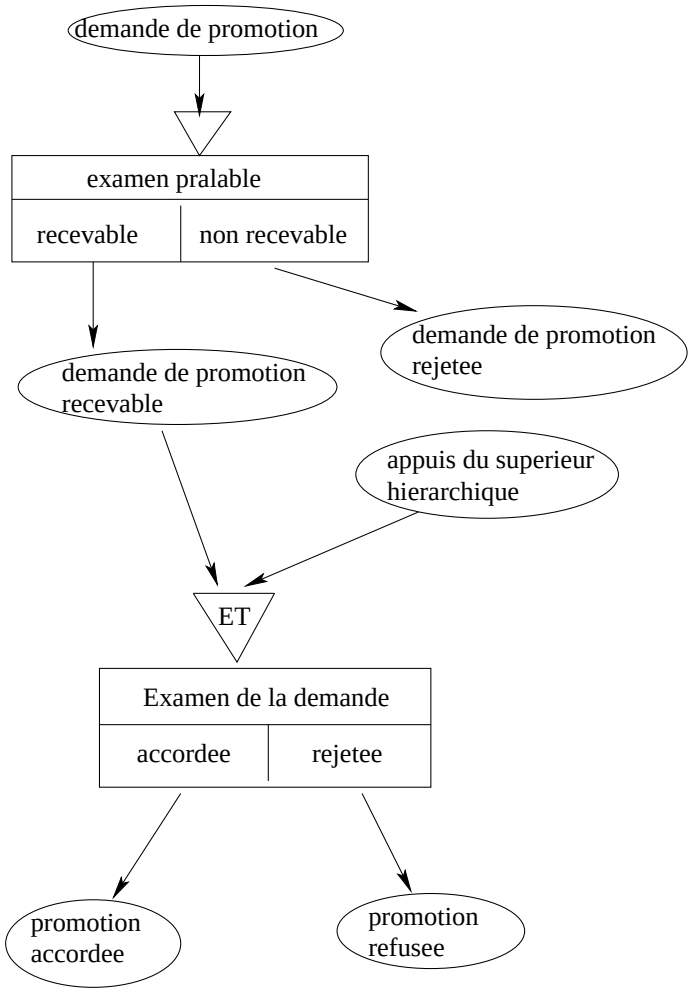


Exercice : La promotion

Dans une grande entreprise, les demandes de promotion sont traitées selon les règles de gestion suivantes :

- Toute demande de promotion doit subir un examen préalable permettant de déterminer si elle est recevable ou pas.
- L'examen du dossier d'une demande recevable ne peut se faire qu'après rapport du supérieur hiérarchique.
- Après examen du dossier par l'autorité compétente, la promotion est accordée ou refusée.

Exercice : La promotion (correction)

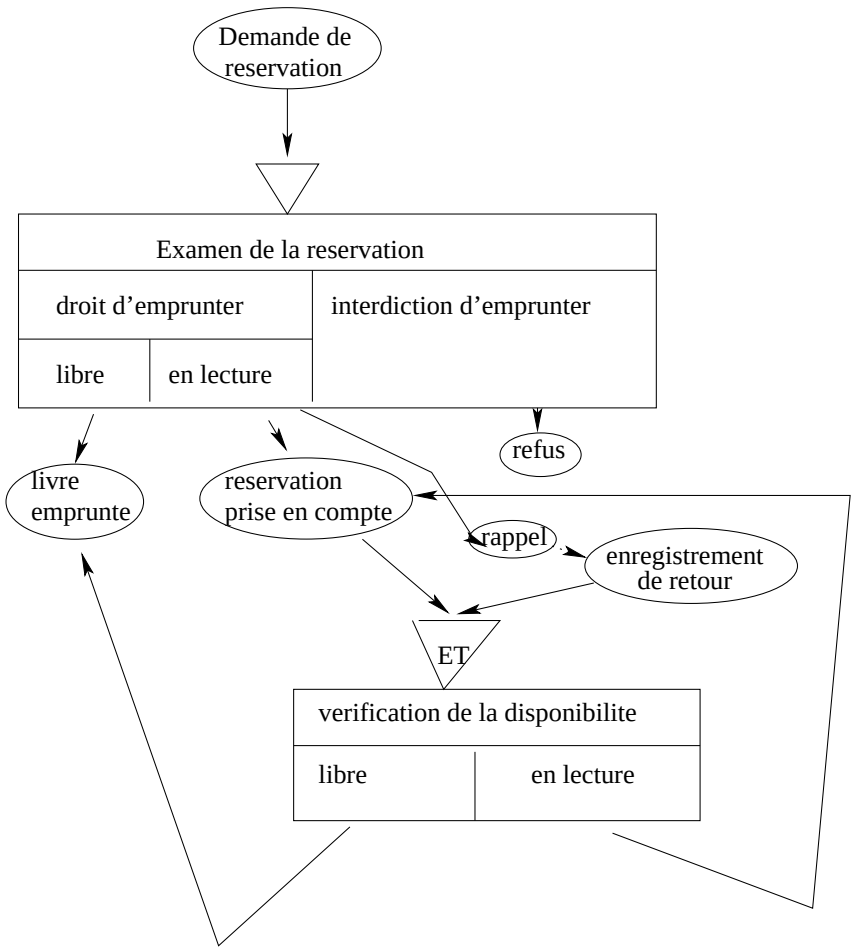


Exercice : La réservation

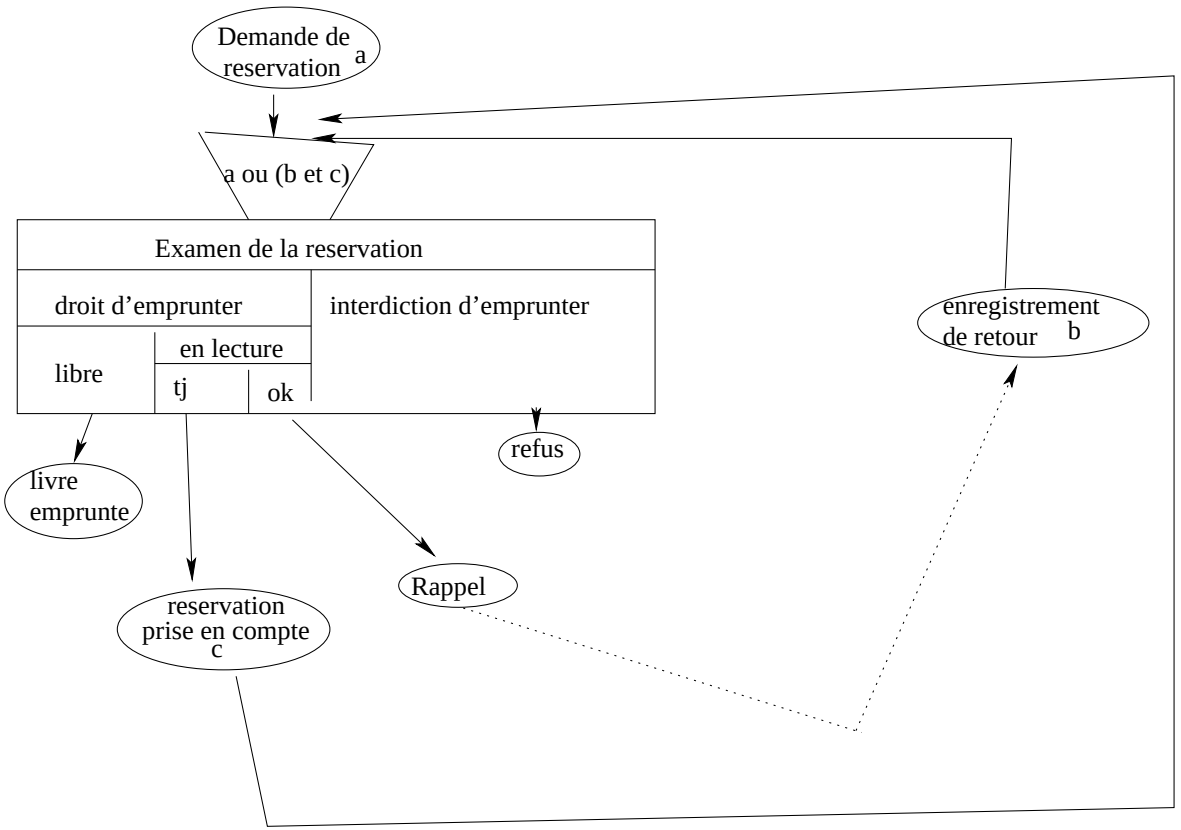
A la bibliothèque ≪Les rats de biblio≫ une réservation se déroule de la façon suivante :

- On remplit un formulaire.
- Si on n'est pas interdit de prêt (pour livres rendus en retard) on examine la demande.
  - Soit le livre est disponible et on l'obtient.
  - Soit il faut attendre qu'il soit rendu par le lecteur précédent et réenregistré comme disponible.

Exercice : La réservation (première idée)



Exercice : La réservation (sans redondance)



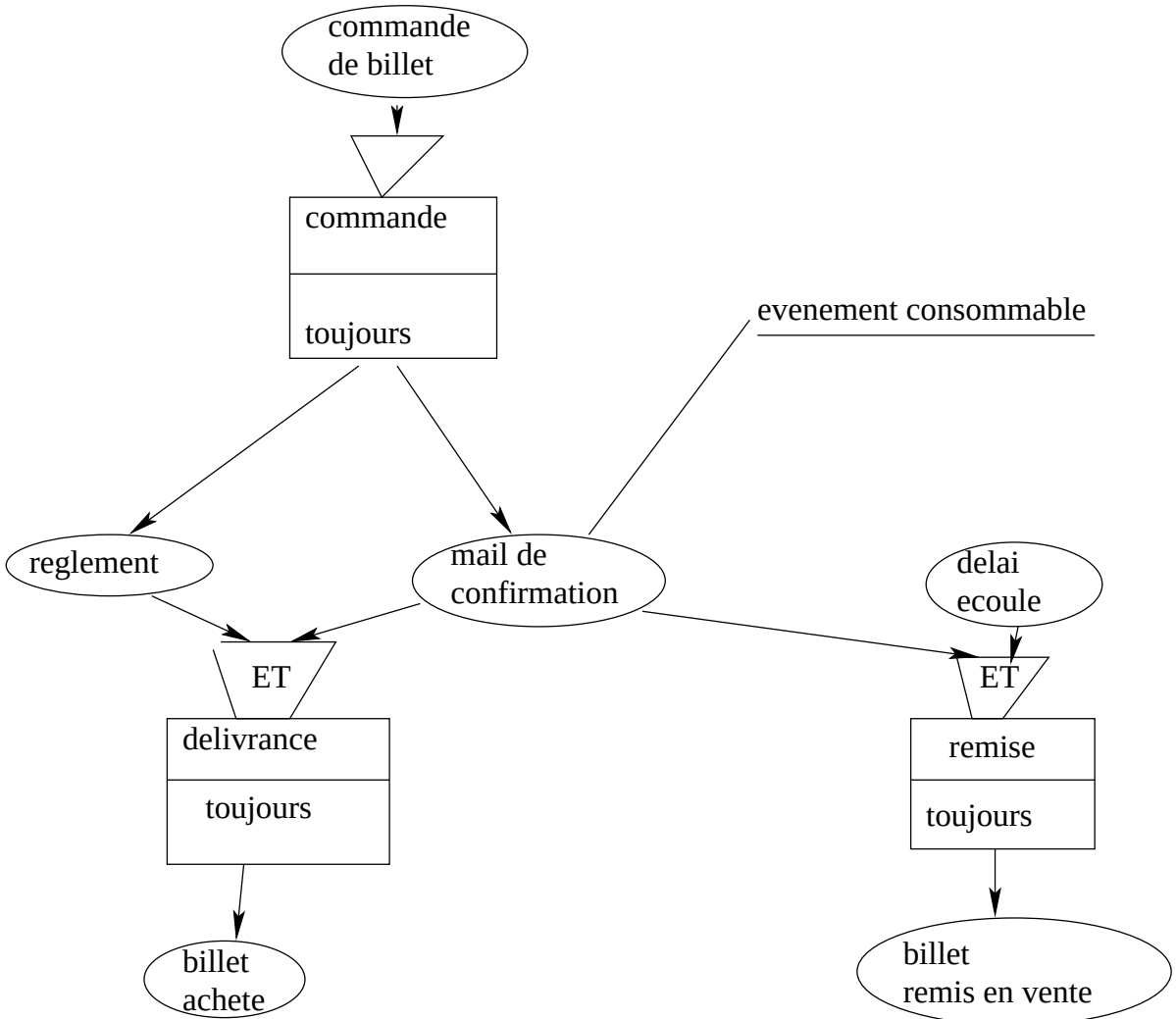
Evènement consommable

- Il se peut qu'un évènement puisse déclancher plusieurs opérations, mais ne doive en déclancher qu'une (ou exclusif).
- Un tel évènement est appelé évènement comsommable.

Évènement consommable : exemple

- On commande un billet de concert sur internet avec retrait du billet en magasin. À la commande du billet un mail de confirmation nous est envoyé. C'est avec une copie de cet émail et le réglement que l'on peut récupérer son billet. Passer un certain délais si le client n'est pas passé en magasin, le billet est remis en vente.

Évènement consommable : exemple (suite)



Niveau organisationnel – traitements

MOT

- Au plan de la description des traitements, le Modèle Organisationnel des Traitements (MOT) intègre les notions de temps et de durée (déroulement), de ressources, de lieu et de responsabilité (poste de travail) et de nature des traitements (manuels ou automatiques).

MOT – Concepts

- Poste de travail : est caractérisé par :
  - Un type de lieu : représente l'ensemble des lieux où les actions d'une opération pourront s'effectuer (ex : magasin, service achat etc.).
  - Un responsable : personne (ou ensemble de personnes) ayant la responsabilité de certaines actions d'une opération (ex. directeur du magasin)
  - Des ressources : moyens permettant de réaliser certaines actions d'une opération (personnel, logiciels, matériel, système homme-machine, des fichiers).

Procédures fonctionnelles (pf)

- À chaque processus du MCT correspondra une ou plusieurs procédures fonctionnelles.
- Procédure fonctionnelle : Succession de tâches appartenant à un même processus et affecté à un poste de travail à un moment déterminé. Une pf est ininterruptible.
- Acteur : Entité pouvant exécuter des pf.
- Poste de travail : Acteur muni de caractéristiques organisationnelles.

Procédures fonctionnelles – détail

- Période.
- Type : Indication du degré d'automatisation : manuelle, interactive (ou conversationnelle), différée (ou batch).
- Tâches.
- Évènements contributifs et synchronisation.
- Événements émis et règles d'émission.

PF	Déroulement		Action	Nature	Poste de travail		
	début	durée			lieu	responsable	ressources
				M/AC/AB			

MOT – Exemple – Règles de gestion tenue de stock

- Le SI concerne une entreprise de distribution qui achète des produits aux fournisseurs pour les revendre aux clients.
- Un produit peut être en stock dans plusieurs magasins.
- Un produit en magasin peut être mouvementé plusieurs fois par diminution ou augmentation du stock.
- Un produit est vendu par un seul fournisseur pour tous les magasins.
- Une commande de réapprovisionnement concerne un fournisseur.

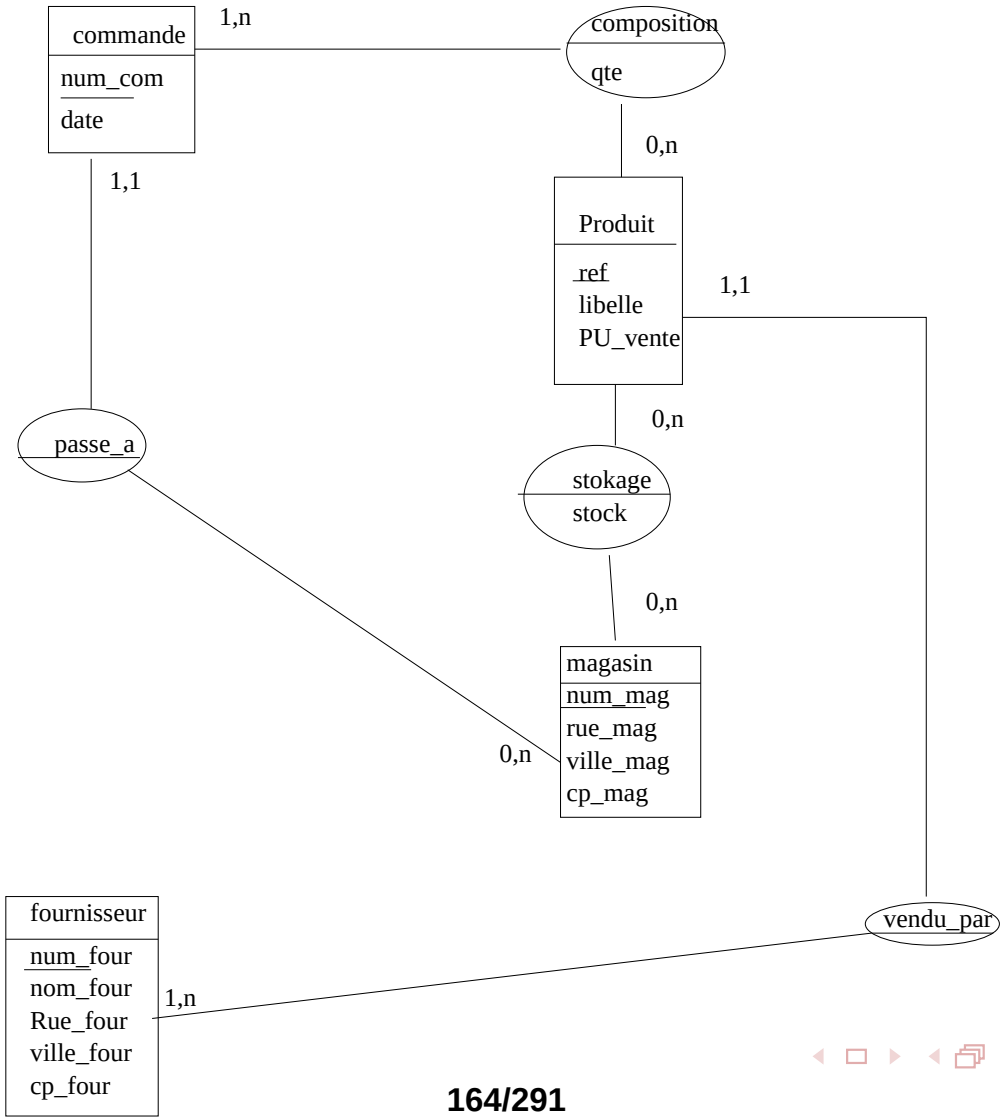
MOT – Exemple – Règles de gestion tenue de stock (suite)

- On passe commande à un fournisseur dans l'un des deux cas suivants : Un produit commandé dans un magasin est en rupture de stock dans ce magasin. Dans un magasin, on a pour un produit : Stock + total commandé aux fournisseurs $<$ stock mini. On commande alors Stock max $-$ (stock + total commandé au fournisseurs).
- Les livraisons des fournisseurs sont contrôlées par comparaison avec les commandes. Toute livraison non conforme est refusée.
- À période fixe on fait un inventaire pour déterminer les écarts entre le stock physique et le stock théorique calculé par le SI.

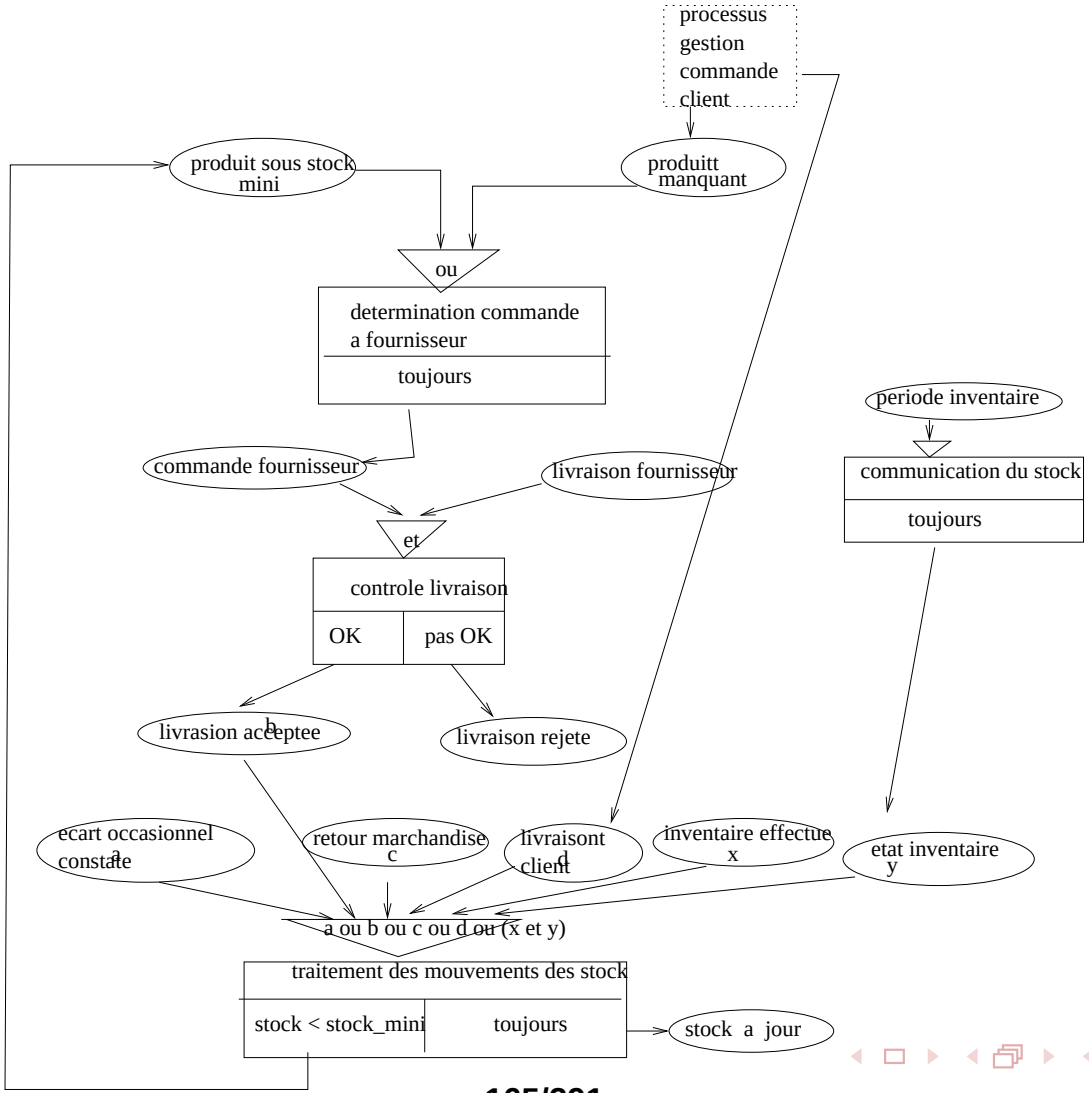
MOT – Exemple – Règles de gestion tenue de stock (fin)

- Les mouvements du stock sont :
  - Hors période inventaire :
    - Livraison fournisseur : stock $=$ stock $+$ quantité livrée.
    - Bon de livraison client : stock $=$ stock $-$ quantité livrée.
    - Retour marchandise client : stock $=$ stock $+$ quantité retournée.
  - Hors ou pendant période inventaire ; ajustement suite à un inventaire ou un écart occasionnel constaté : stock $=$ stock $\pm$ écart entre les stocks réel et théorique.

MOT – Exemple – MCD tenue de stock



MOT – Exemple – MCT tenue de stock



MOT – Exemple – Règles d'organisation

- Le service achats, le service commercial et les magasins sont équipés de micro-ordianateurs communiquant par mail.
- Pour la détermination des commandes à passer aux fournisseurs, le micro du service achats édite des propositions de commandes qui sont analysées par le responsable en vue d'une validation ou de modifications. Ces opérations doivent être faites le matin.
- Les commandes validés sont ≪mailées ≫ aux fournisseurs concernés et aux magasins concernés.
- À chaque livraison fournisseur, le magasinier contrôle la marchandise livrée en la comparant à la marchandise comandée figurant sur la demande fournie au fournisseur.

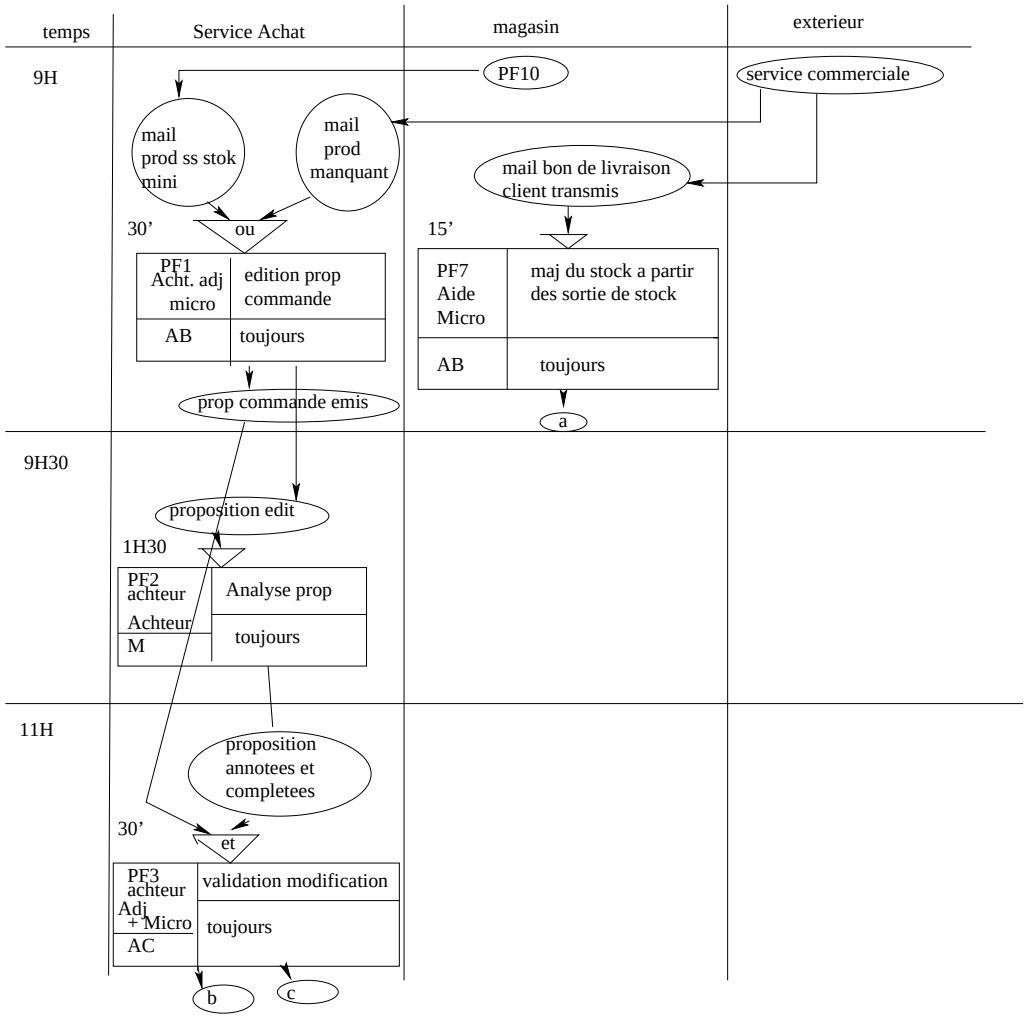
MOT – Exemple – Règles d'organisation (fin)

- La mise à jour du stock s'effectue :
  - Le matin à 9H pour les sorties de stock. Celles-ci (double des bons de livraison à clients) proviennent du processus gestion des commandes clients et sont transmises au magasin concerné par mail.
  - En temps réel à tout autre moment de la journée de travail pour les autres mouvements. Les anomalies sont immédiatement recyclées.
- L'inventaire est annuel. La veille il y a édition de l'état des stocks. L'inventaire dure 2 jours. On recherche les écarts entre le stock du SI et le stock réel.

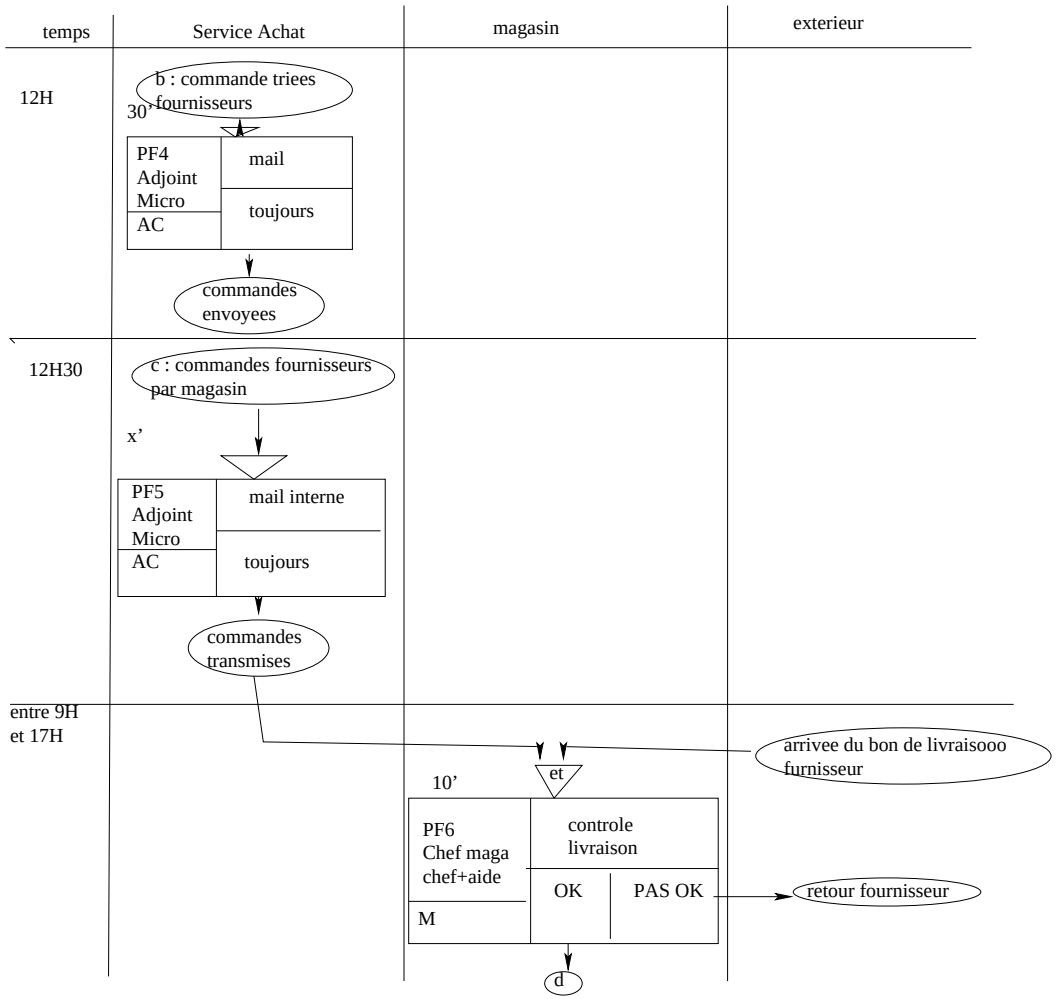
MOT – Exemple – Procédures fonctionnelles

PF	Déroulement		Action	Nature	Poste de travail		
	début	durée			lieu	responsable	ressources
PF1	9H	30'	Édition des propositions de commande	AB	SA	Achteur adjoint	Micro
PF2	10H	1H30	Analyse des propositions	M	SA	Achteur	Achteur
PF3	11H	30'	Validation	AC	SA	Achteur	Adjoint + Micro
PF4	12H	30'	Envoi cdes fournisseurs	AC	SA	Adjoint	Micro
PF5	12H30	x'	Envoi cdes aux mag	AC	SA	Adjoint	Micro
PF6	9H $\leq t \leq$ 17H	10'	contrôle livraison	M	Mag	Chef magasinier	chef + aide
PF7	9H	15'	maj par sortie du stock	AB	Mag	Aide	Micro
PF8	9H $\leq t \leq$ 17H	5'	maj par autre mouvement de stock	AC	Mag	chef ou aide	chef ou aide + micro
PF9	17H	x'	détermination produit ss stock mini	AB	Mag	Aide	Micro
PF10	17H10	x'	transmission par mail des produit ss sock mini	AC	Mag	Aide	Micro
PF 11	jo 17H	1H	Edition état du stock	AB	Mag	Aide	Micro
PF 12	jo +1, 7H	2 jours	détermination des écarts	M	Mag	chef	chef + aide

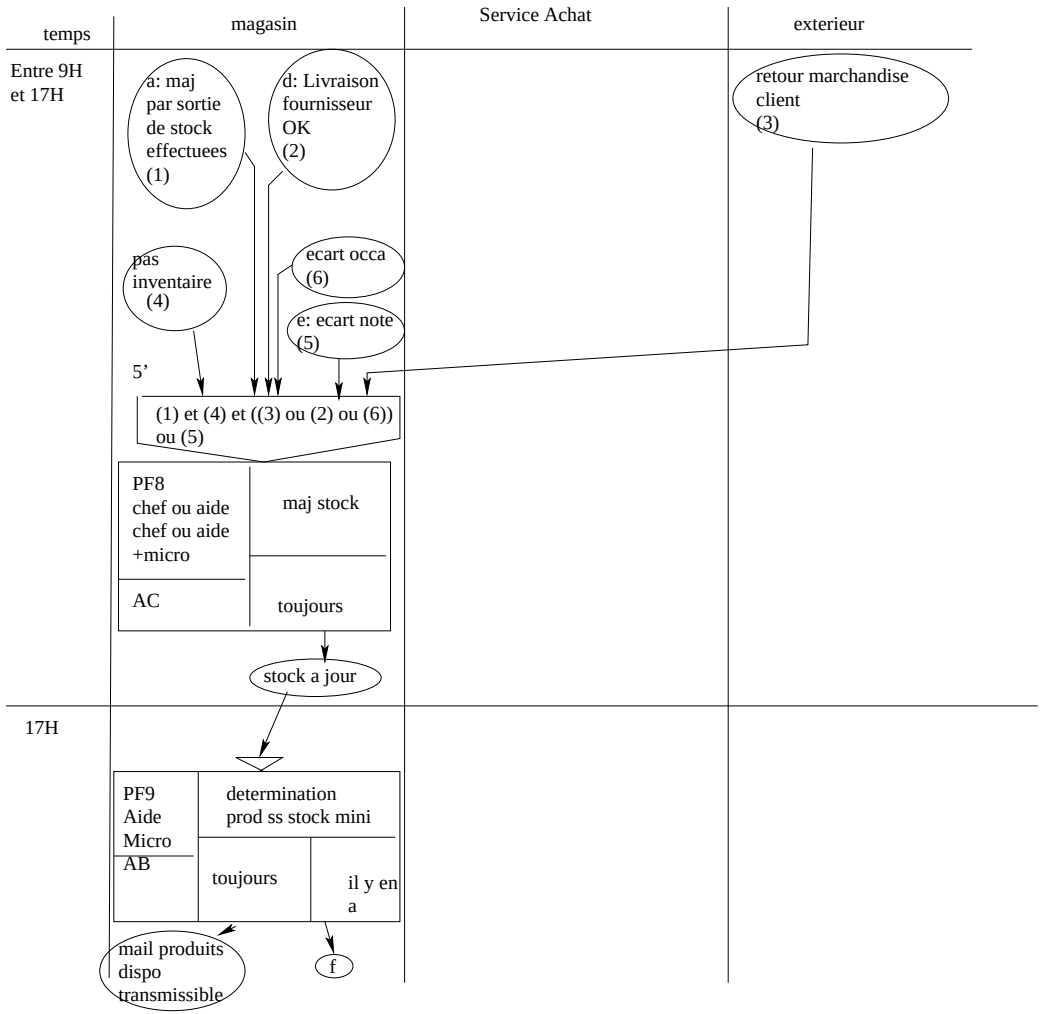
MOT – Exemple


temps
Service Achat
magasin
exterieur
9H
PF10
service commerciale
mail prod ss stok mini
mail prod manquant
mail bon de livraison client transmis
30'
ou
15'
PF1 Acht. adj micro
edition prop commande
AB
toujours
PF7 Aide Micro
maj du stock a partir des sortie de stock
AB
toujours
prop commande emis
a
9H30
proposition edit
1H30
PF2 achteur
Achteur M
Analyse prop
toujours
11H
proposition annotees et completees
30'
et
PF3 achteur Adj + Micro AC
validation modification
toujours
b
c

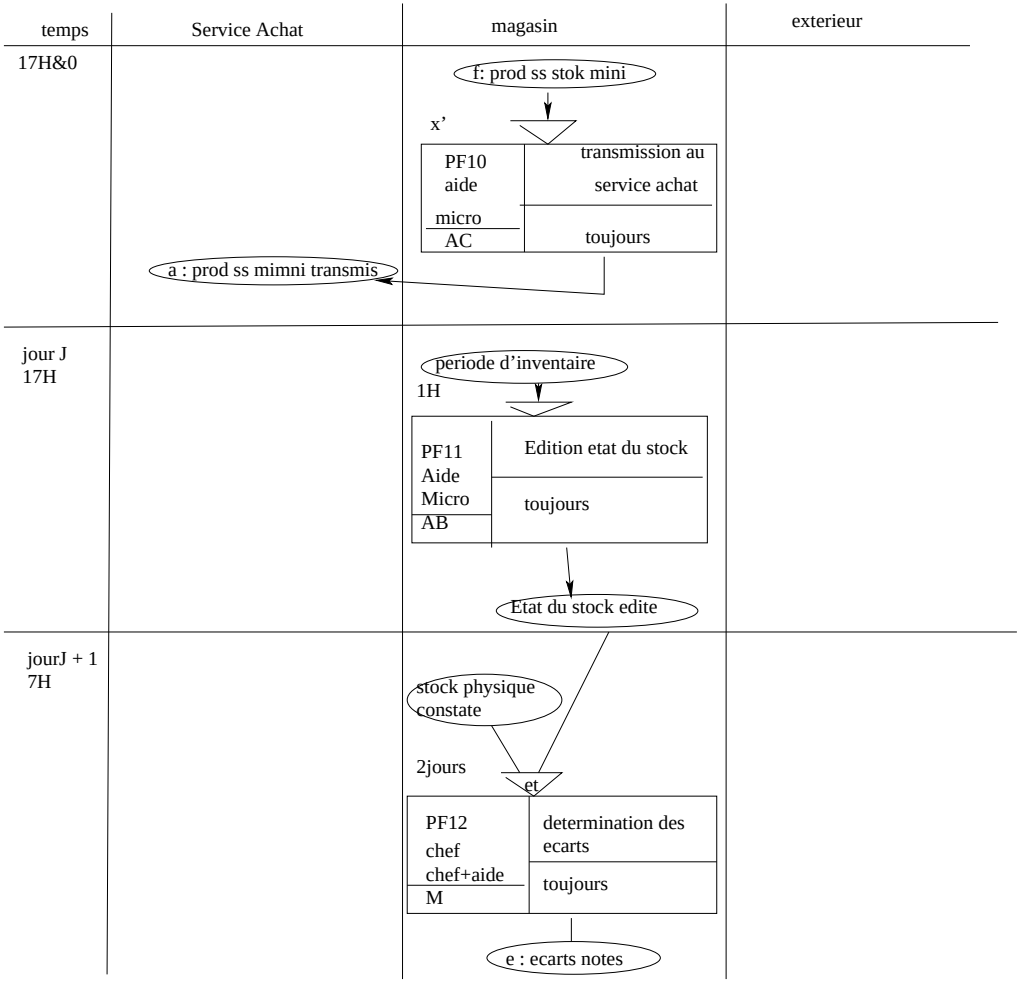
MOT – Exemple (suite)



MOT – Exemple (suite) Attention inversion



MOT – Exemple (fin)



MOT – À noter

- Vous pouvez aussi trouver le MOT en plusieurs schémas :
  - Un schéma par processus représentant les enchaînements de procédures fonctionnelles.
  - Un graphe de circulation. Mise en évidence de la circulation d'information avec code de dessin pour le texte, les disquettes etc.

Niveau logique – Traitements

La description logique des traitements (DLT)

- Dépend :
  - du type de poste de travail du client,
  - les réseaux utilisés (locaux, nationaux et internationaux),
  - les serveur de données et de traitements,
  - les outils de développement,
  - les SGBD,
  - etc.

UML

Approche objet

- Fournir une solution au problème de la séparation données / traitements.
- Un type de données contient aussi les traitements qui lui sont propres.
- Un objet est une entité clairement définie qui possède une identité (qui permet de le distinguer des autres objets, indépendamment de son état).

Approche objet (suite)

- L'état de l'objet est caractérisé par la valeur de ses attributs.
- Le comportement de l'objet est caractérisé par ses méthodes(ensemble d'actions (appelées opérations lorsqu'elles ne sont pas implémentées) que l'objet est à même de réaliser).
- Tout objet est une instance d'une classe. Une classe est un type de données abstrait caractérisé par des propriétés (attributs et méthodes) communes à toute une famille.

Approche objet (fin)

- Encapsulation : Consiste à masquer les détails d'implémentation d'un objet, en définissant une interface. L'interface définit les services accessibles aux utilisateurs de l'objet.
- Agrégation : Relation entre classe : les instances d'une classe seront composés d'instances d'autres classes.
- Héritage : Relation de spécialisation, facilite la réutilisation en ajoutant de nouvelles propriétés à une classe existante. Mecanisme de transmission des propiétés d'une classe vers une sous-classe. Mélange de : plus d'attributs, plus de méthodes, restriction des domaines de définition, redéfinition.
- Polymorphisme : Utilisation d'une méthode sur des instances de plusieurs classes.

Une rapide introduction

- UML inventé pour faire face à la conception de logiciels de plus en plus complexes.
- Résultat de la fusion de trois méthodes de modélisation objet : OMT (James Rumbaugh), OOD (Grady Booch) et OOSE (Ivar Jacobson).
- Domaine d'utilisation :
  - une conception logiciel (formaliser toutes les modalités de réalisation et d'implémentation),
  - modélisation d'un processus entreprise.
- UML est un langage pas une méthode.
- UML est un langage, donc il convient d'en respecter scrupuleusement la syntaxe et la sémantique.

UML – architecture

- Vue logique : Modélisation des mécanismes principaux du système.
- Vue de réalisation : Modules qui implémentent les classes de la vue logique.
- Vue des processus : Décomposition du système en processus, intéractions et synchronisations.
- Vue de déploiement : Resources matérielles et logicielles, environnement distribué.
- Vue des cas d'utilisation : Fournit un guide pour les autres vues : définition des besoins des utilisateurs.

Niveau d'abstraction

- Conceptualisation : À partir de l'expression des besoins, définir les fonctionnalités principales du système.
- Analyse du domaine : À partir des cas d'utilisation, définir les éléments du domaine et leurs interactions.
- Analyse applicative : Détermination des interfaces des éléments de modélisation, relation entre les éléments.
- Conception : Les éléments de modélisation sont complêtement détaillés.

Diagrammes

- Élément fondamental de modélosation UML : le diagramme.
- Un diagramme est une représentation graphique d'objets et de relations entre eux.
- Il en existe plusieurs sortes.
- Modèle structurel : Diagramme de classes, diagramme d'objet.
- Modèle dynamique : Diagramme des cas d'utilisation, diagramme d'activité, diagramme des collaborations, diagramme des séquences, diagramme d'états-transitions.
- Architecture : diagramme de composant, diagramme de déploiement.

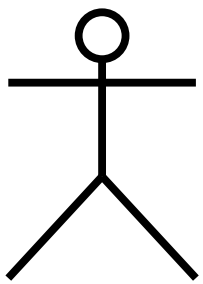
Diagramme de cas d'utilisation

Diagramme de cas d'utilisation (Use cases)

- Spécification des besoins des utilisateurs.
- Description du comportement du système pour l'utilisateur.
- Objectif : Spécification de la fonctionnalité offerte à l'utilisateur.

Acteur

- Identification des acteurs . Rôle joué par des entités externes au système et qui ont des intercations avec le système. Exemple : utilisateur, autre système.
- Description des acteurs . À quoi sert chaque acteur ? nombre d'utilisateurs et fréquence d'utilisation, connaissances de l'acteur.
- Pompiste et Client, Adm bal et utilisateur bal



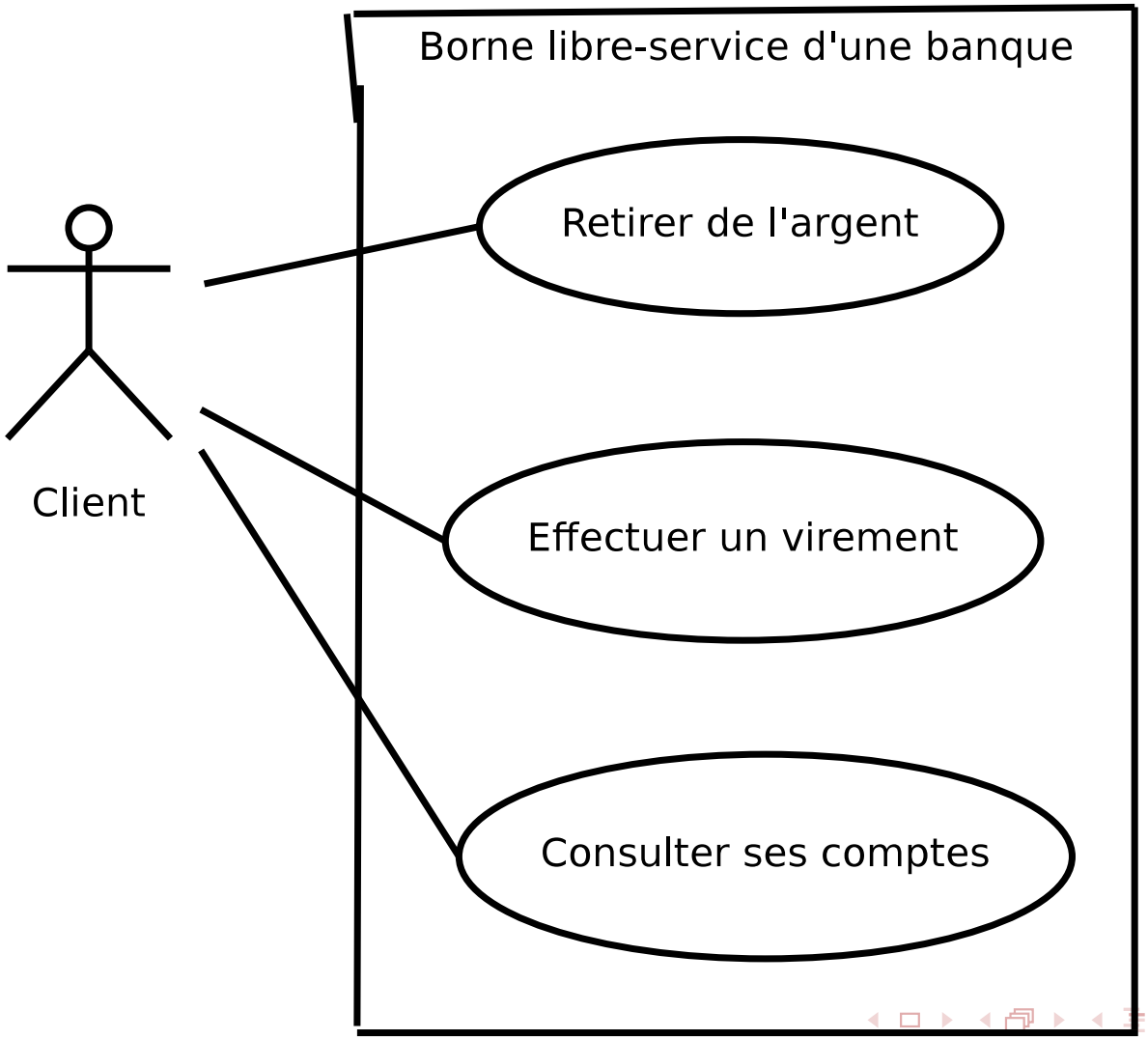
Client

Cas d'utilisation

- Identification des cas d'utilisation. Cas d'utilisation : modélisation d'un service fourni par le système. Exprime des interactions avec les acteurs. Séquence d'actions réalisé par le système.
  Quels sont les traitements que chaque acteur attend du système ?

- Organisation des cas d'utilisation . Structuration en paquetages. Dans chaque paquetage, mettre en relation les cas d'utilisation.

Cas d'utilisation (suite)

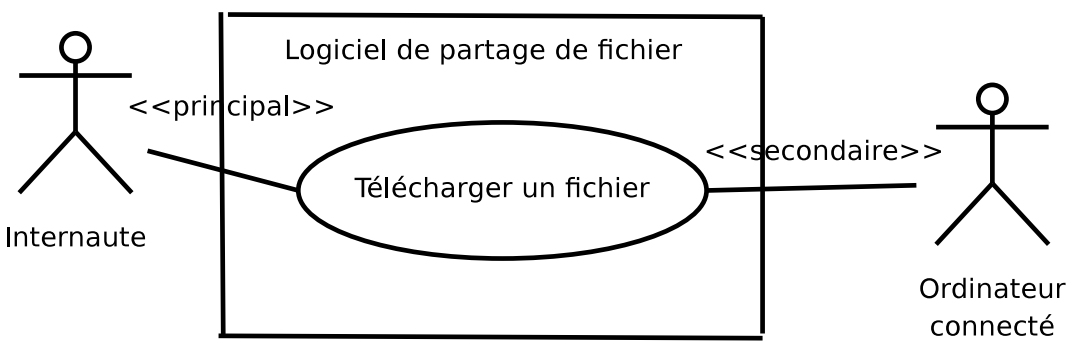


Multiplicité

- Lorsqu'un acteur peut interagir plusieurs fois avec un cas d'utilisation, il est possible d'ajouter une multiplicité sur l'association du côté du cas d'utilisation.
- * signifie plusieurs.
- n signifie exactement n.
- n..m signifie entre n et m.

Acteurs principaux et secondaires

- Acteur principal, acteur à qui un cas d'utilisation rend service.
- Les autres acteurs sont appelés acteurs secondaires.
- Un seul acteur principal par cas d'utilisation.
- Acteur principal obtient un resultat observable.
- Acteur secondaire est solicité pour des informations suplémentaires.
- En général, l'acteur principal initie le cas d'utilisation par ses solliciations.

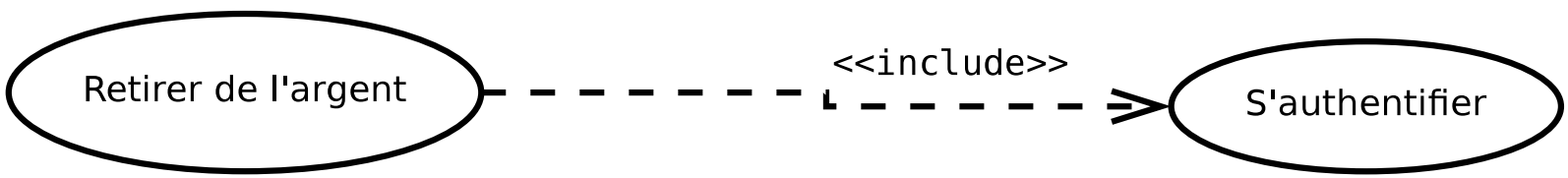


Relation entre cas d'utilisation

- Cas d'utilisation interne : Lorsqu'un cas d'utilisation n'est pas relié à un acteur.
- 2 types de relations entre les cas d'utilisation :
  - Les dépendances stéréotypés (les plus utilisés : l'inclusion et l'extension).
  - La généralisation/spécialisation.

La relation d'inclusion entre cas

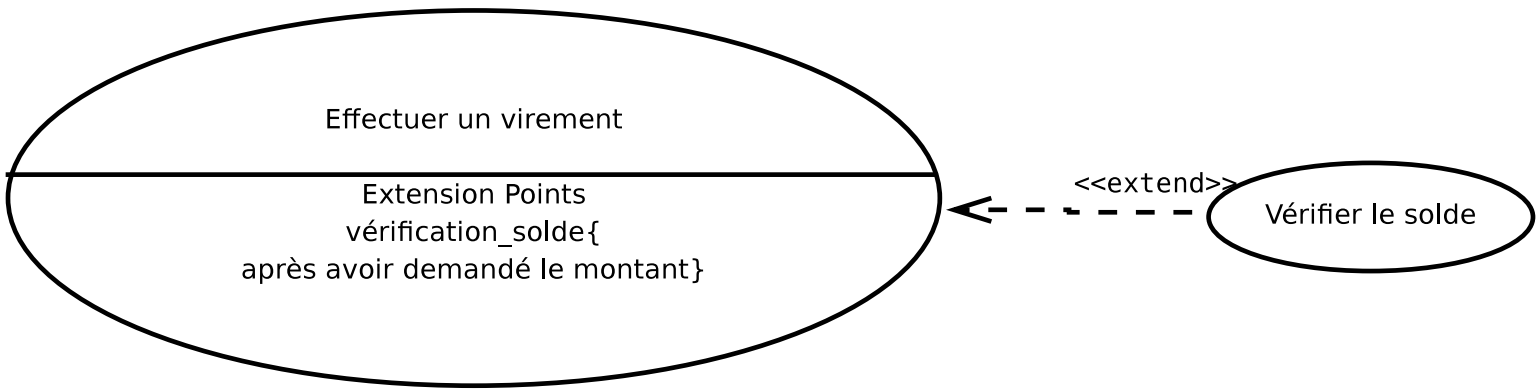
- Un cas d'utilisation incorpore un autre cas.
- Le cas qui incorpore l'autre ne peut pas être utilisé seul.
- Si A inclut B, lorsque A est sollicité, B l'est obligatoirement. Attention, les cas d'utilisation ne s'enchaînent pas. Il n'y a pas de représentation temporelle.
- Si A inclue B, une flèche pointillé part de A vers B.



- *Retirer de l'argent* inclut (incorpore) *s'authentifier*.

La relation d'extension entre cas

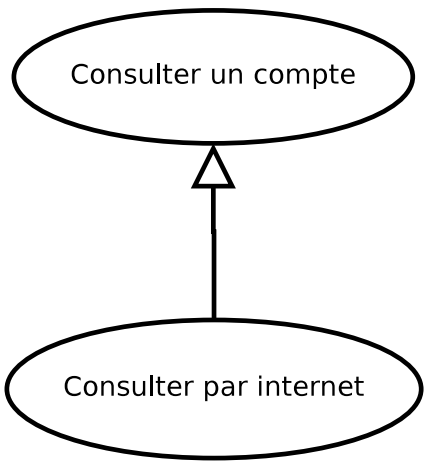
- A étend B, lorsque A peut être appelé au cours de B. Attention ce n'est pas obligatoire.
- L'extension peut intervenir à un point précis du cas étendu. Ce point s'appelle le point d'extension.



- *Vérifier le solde* étend *effectuer un virement*.
- On vérifie le solde, lorsque la somme dépasse 200 euros.

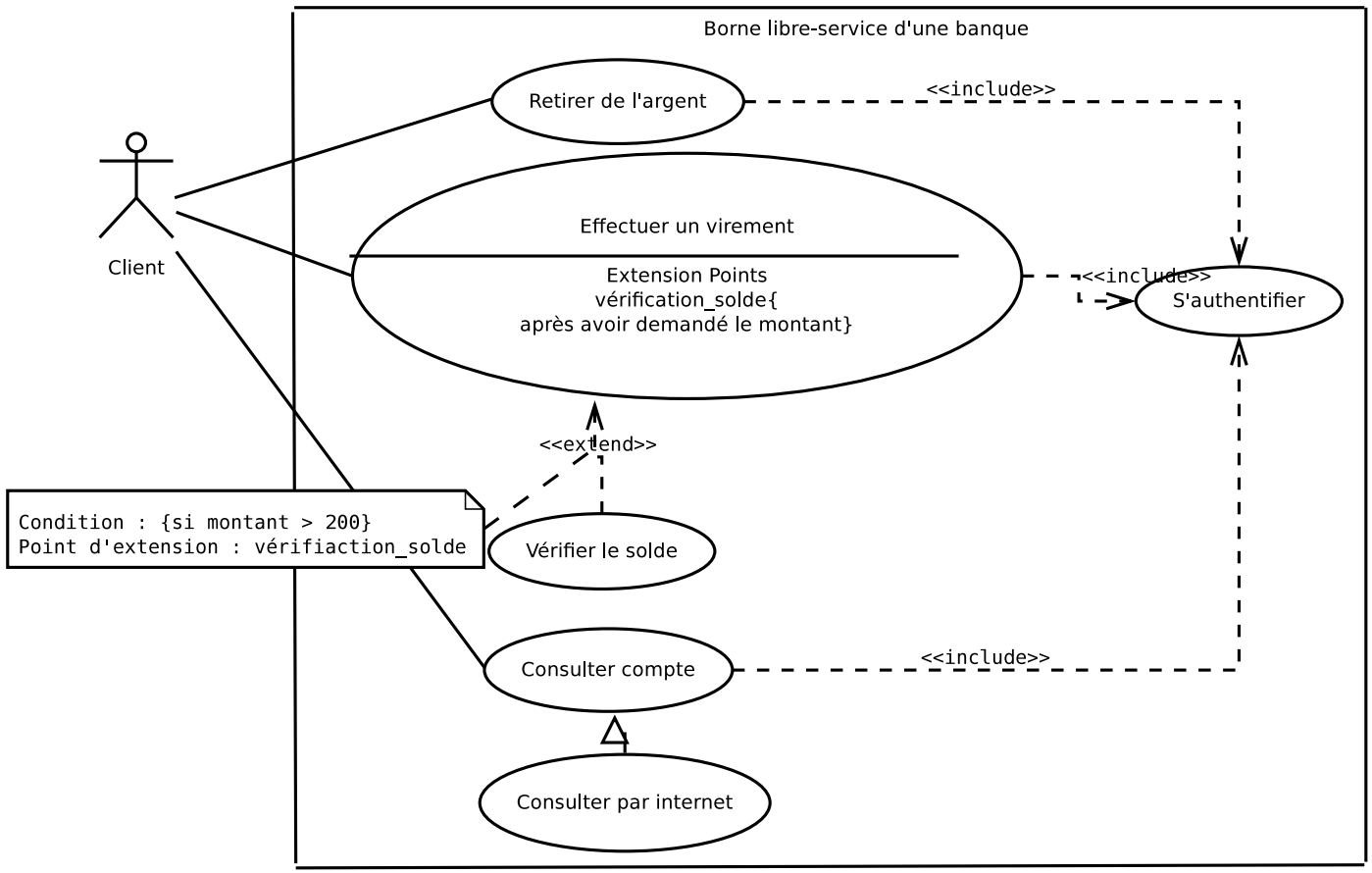
Relation de généralisation

- Un cas A est une généralisation de B, si B est un cas particulier de A.



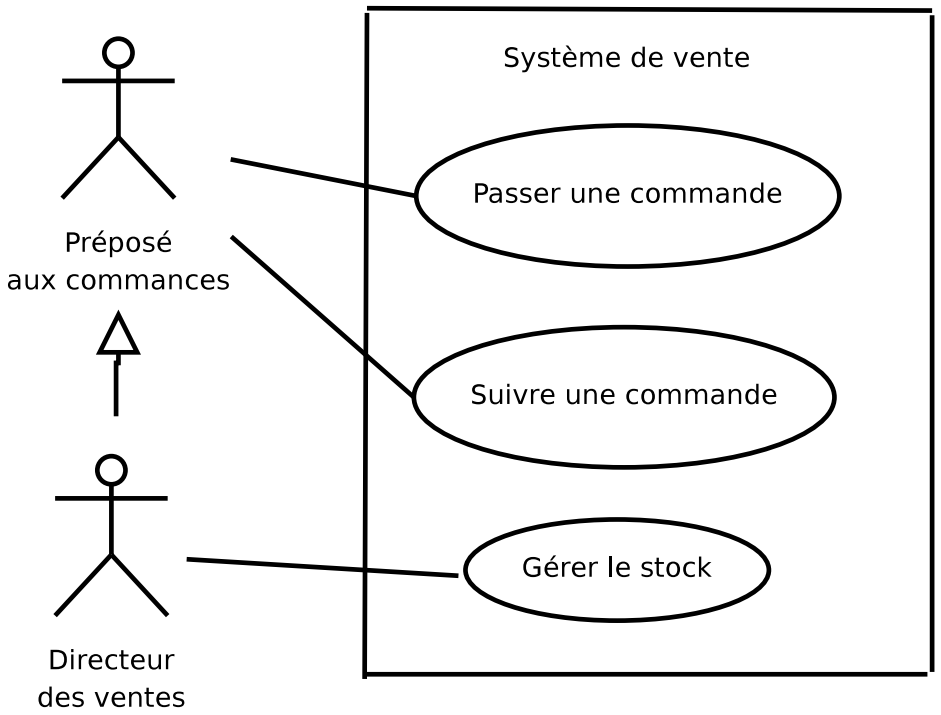
- *Consulter par internet* est un cas particulier *consulter compte*.

Les relations entre cas d'utilisation



Les relations entre acteurs

- L'acteur A est une généralisation de l'acteur B si A peut être remplacé par B.
- Tous les cas d'utilisation accessible par A sont accessibles par B. L'inverse n'est pas vrai.



Description textuelle des cas d'utilisation

- Pour les acteurs : Nombre d'utilisateurs, fréquence d'utilisation, connaissances.
- Pour les cas d'utilisation : Titre, but, résumé, acteurs, préconditions, déclenchement, enchaînement, fin, exceptions, postconditions.

Notions en langage UML

- Une note. Contenue dans un rectangle au coin supérieur droit replié. Doit être reliée à son sujet par un trait en pointillé.
- Un stéréotype. Annotation s'appliquant à un élément du modèle. Représenté dans une chaîne de caractères entre guillemets. Contenu ou placé à côté du symbole de l'élement du modèle.
- Paquetage. Groupement d'éléments du modèle et de diagramme. Éléments d'un paquetage doivent représenter un ensemble fortement cohérent. Sont généralement de même nature et de même niveau sémantique.
- Attention aux frontières des acteurs. Exemple hotesse de caisse, caisse et client.

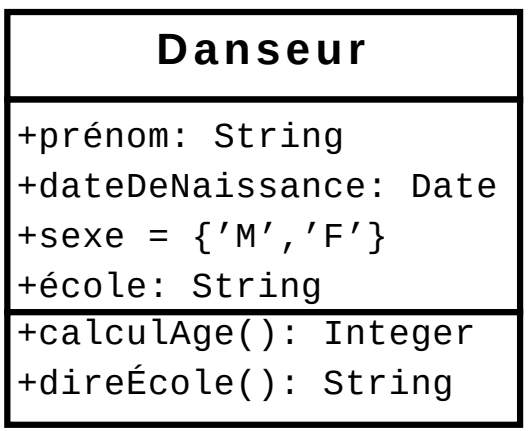
Diagramme de classes

Diagramme de classes

- Le diagramme de classes est considéré comme le plus important en conception objet.
- Il montre la structure interne du système.
- Il contient principalement des classes.
- Une classe est consitué d'attributs et d'opérations.
- Il ne montre pas comment utiliser les opérations. C'est une description statique du système.

La classe

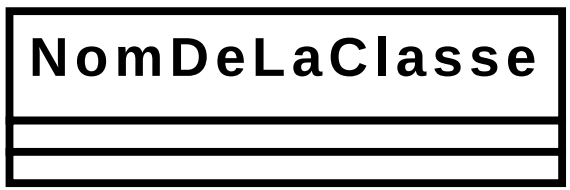
- Une classe est une description d'un ensemble d'objets ayant une sémantique, des attributs, des opérations et des relations en commun.
- Un objet est une instance d'une classe.
- Un objet d'une classe doit avoir des valeurs uniques pour tous les attributs définis dans la classe et doit pouvoir exécuter toutes les méthodes de la classe sans exception.

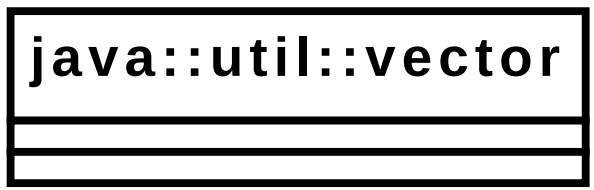


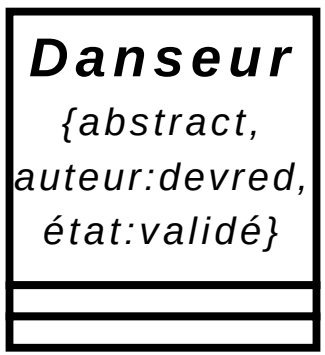
- On dit que l'âge est un attribut indirect.

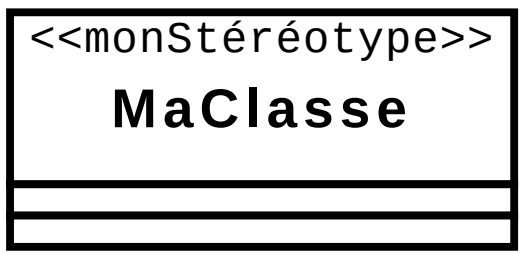
Nom de la classe

- [≪stéréotype ≫]
  [<NomDuPaquetage1 > : : . . . : : <NomDuPaquetageN>] : :
  <NomDeLaClasse>[{[abstract],[auteur],[etat],. . .}]





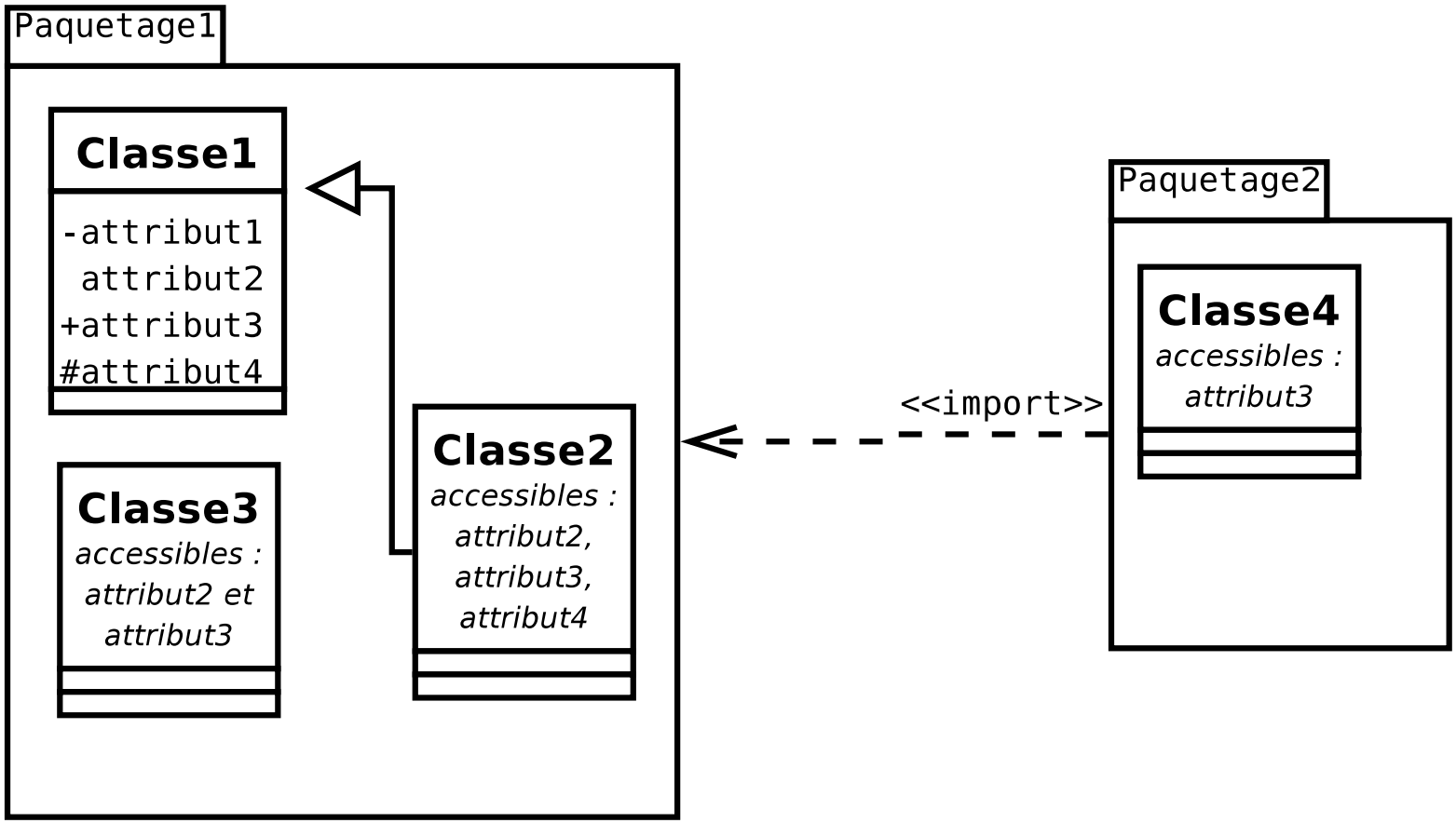




Encapsulation

- Principe du coffre-fort.
- Une propriété visible partout est dite publique , on note public ou + .
- Une propriété visible qu'à l'intérieur d'une classe est dite privée , on note private ou - .
- Une propriété visible partout seulement des descendants est dite protégée , on note protected ou # .
- La visibilité d'une propriété se limite au paquetage dans lequel elle est définie. Pour l'imposer ni mot-clé ni caractère.

Encapsulation (suite)

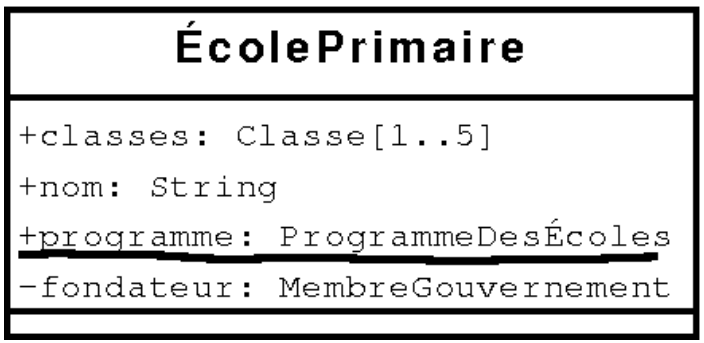


Attributs de la classe

- Attributs définissent les informations qu'une classe ou un objet doivent connaître.
- Ils sont définis par un nom (unique dans la classe), un type de données et une visibilité.
- En UML, les attributs correspondent à des instances de classe déjà définies.

Attributs de la classe

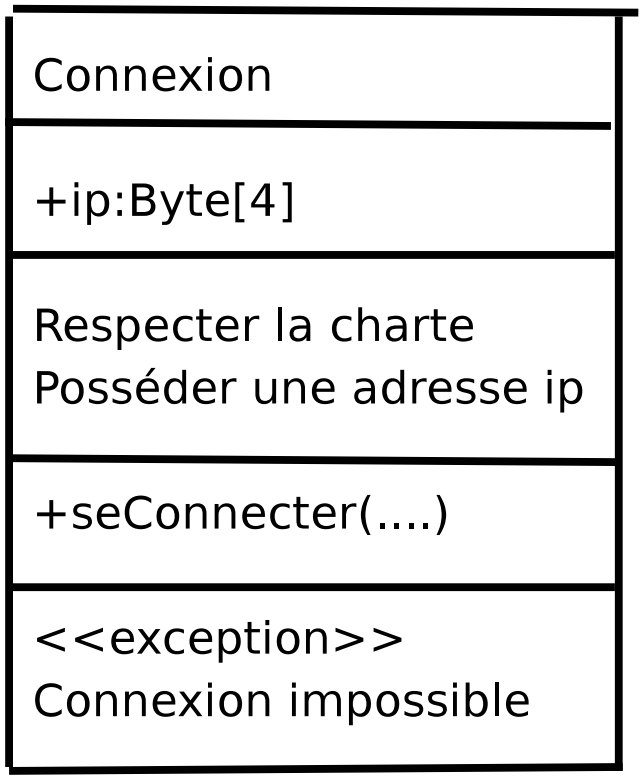
- <modificateur d'accès>[/]<nomAttribut > : <NomClasse >['[' multiplicite ']'] [= valeur(s) initiale(s)]



- Les programmes sont les mêmes pour toutes les écoles, il sagit d'un attribut de classe. ! On n'en hérite pas !
- Un attribut dérivé / est calculé en fonction des autres propriétés. Il sert de marqueur jusqu'à ce qu'il soit déterminé.

Compartiments complémentaires d'une classe

- Processus de modélisation incrémental.
- D'où 2 compartiments supplémentaires : Responsabilité et Exception.



Relation entre classes

- Classes éléments de base du diagramme de classe.
- Classes liées entre elles.
- Les relations le plus utilisées : l'héritage, l'association, l'aggrégation, la composition et la dépendance.
- La plupart du temps ces relations sont binaires.
- Même si les relations sont décrites dans le diagramme de classe, elles expriment souvent des liens entre objets.

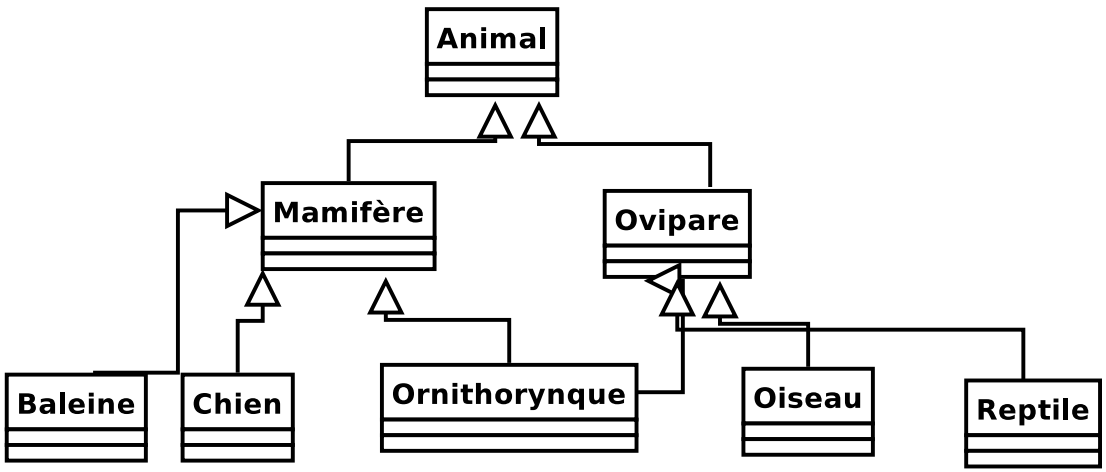
Généralisation et héritage

- La généralisation décrit une relation entre une classe générale (classe de base, sur-classe ou classe parent) et une classe spécialisée (sous-classe).
- La sous-classe est cohérente avec la classe parent, mais comporte des propriétés supplémentaires.
- Un objet de la sous-classe peut être utilisé partout où un objet de la classe parent peut être utilisé.

Généralisation et héritage (suite)

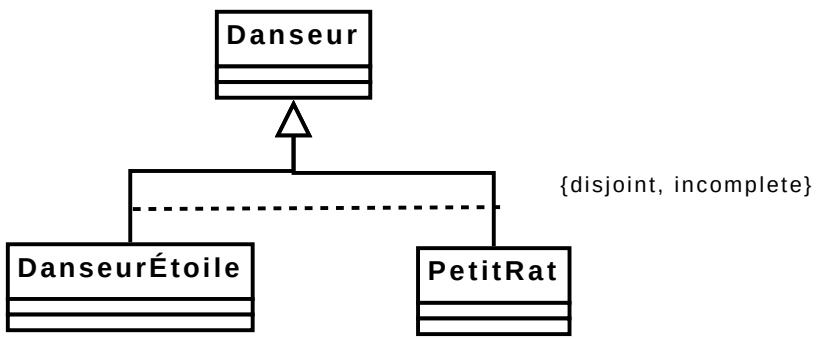
- Propriété :
  - Possède toutes les propriétés de ses classes parents mais ne peut accéder aux propriétés privées de celle-ci.
  - Une classe enfant peut redéfinir les méthodes de la classe de base (par défaut on utilise les méthodes les plus spécifiques).
  - Toutes les associations de la classe de base s'appliquent aux classes dérivées.
  - Une classe peut avoir plusieurs parents (le java ne le permet pas).

Généralisation et héritage(suite)



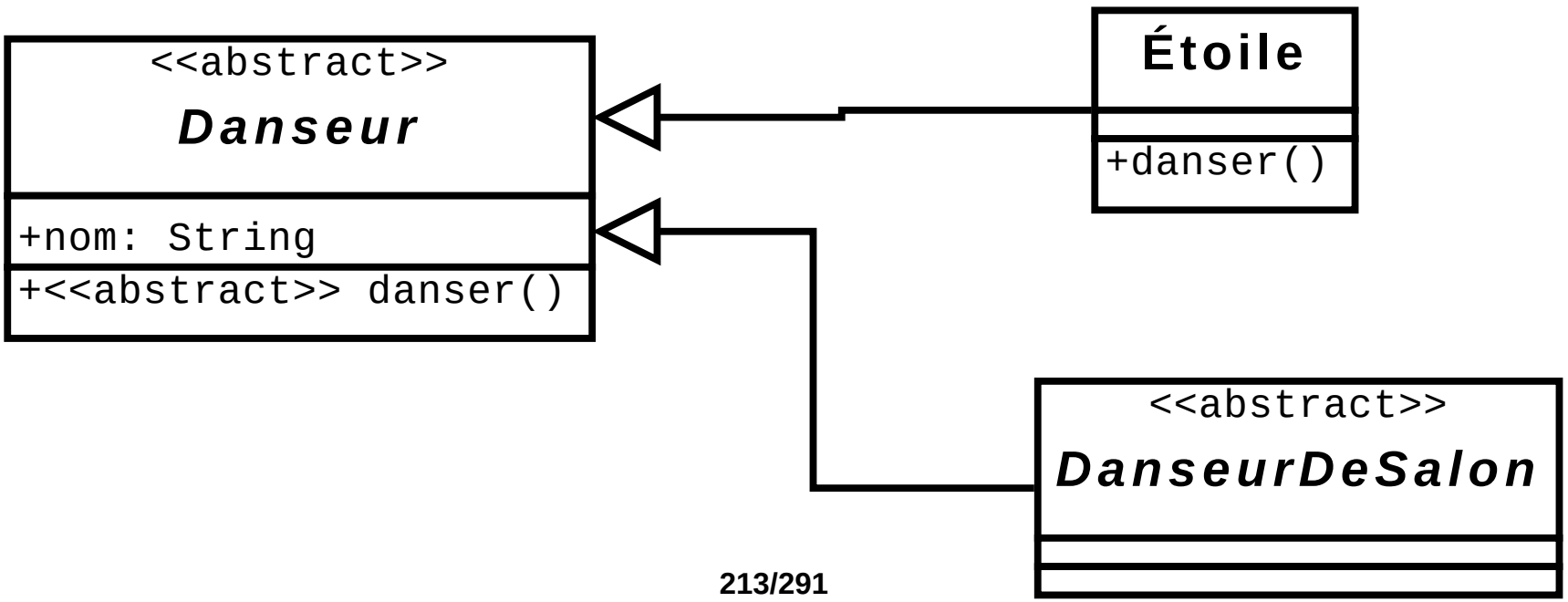
Contraintes sur la relation d'héritage

- 4 contraintes :
  - Sur les classes :
    - incomplet : { incomplete}
    - complet : { complete}
  - Sur les instances :
    - disjoint : { disjoint }
    - possibilité d'instance en commun : { overlapping }



Les classes abstraites

- Une méthode est dite abstraite lorsqu'on connaît son entête mais pas la manière dont elle peut être réalisée.
- Une classe est dit abstraite lorsqu'elle contient une méthode abstraite ou qu'une classe parent contient une méthode abstraite non réalisée.

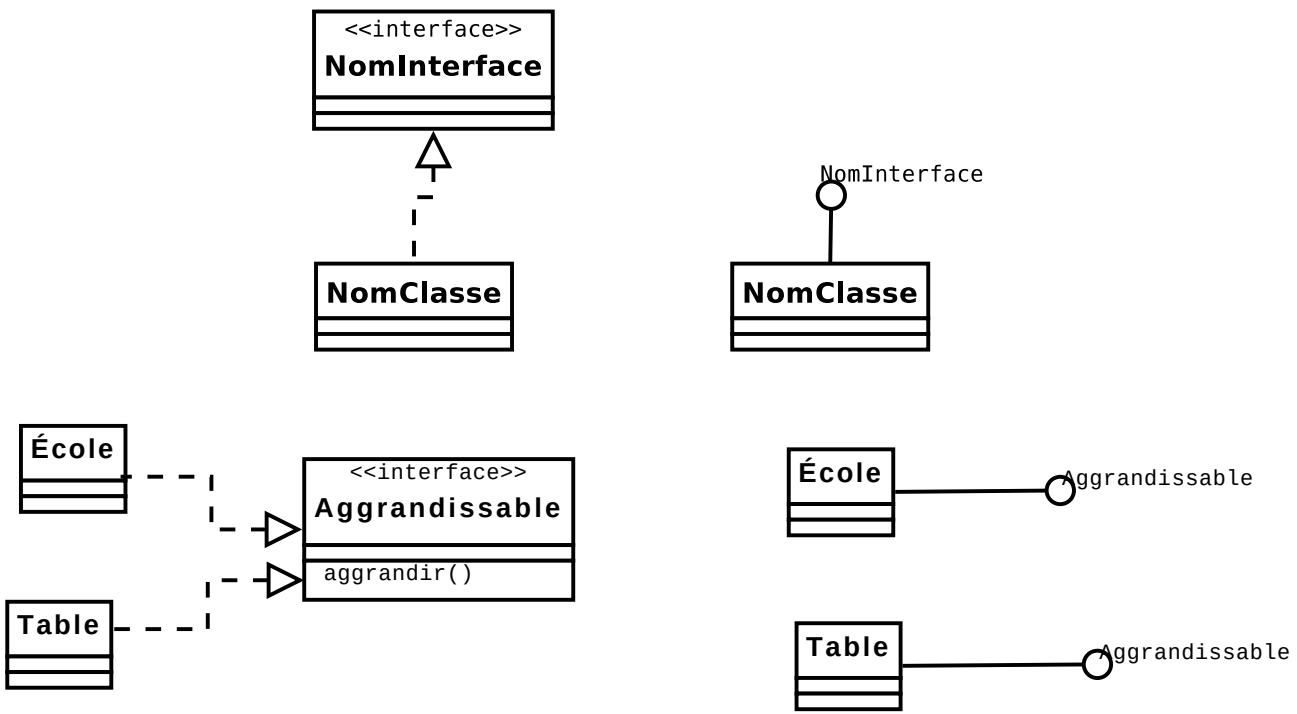


Interface

- Rôle des interfaces : regrouper un ensemble d'opération assurant un service cohérent offert entre autres par une classe.
- Concepte d'interface utilisé essentiellement pour classer les opérations en catégorie sans préciser la manière dont elles sont implémentées.
- Ne spécifie ni une structure interne, ni des algos de réalisation.
- Peut être spécialisée ou généralisée par d'autre interface.
- Toutes ses méthodes sont par définition abstraites.
- Doit être mis en oeuvre par une classe.
- Pas d'attribut.

Interface (suite)

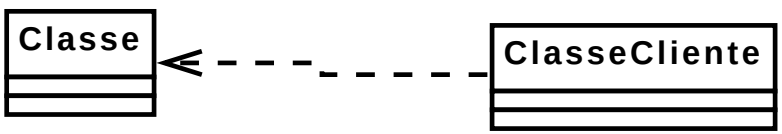
- Classe qui réalise une interface.

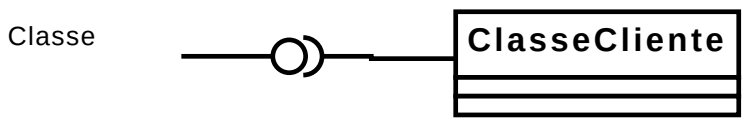


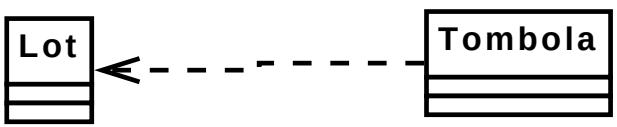
Classe cliente

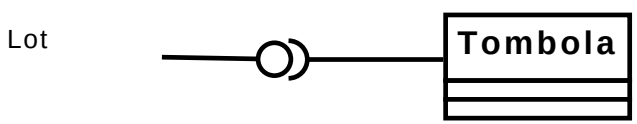
- Une classe peut dépendre d'une interface pour réaliser ses opération. Une classe qui dépend d'une interface en est sa cliente .
- La relation de dépendance existe aussi entre deux classes. En effet, pour réaliser ses opérations, une classe peut dépendre d'une autre classe et pas seulement d'une interface.

Classe cliente (suite)



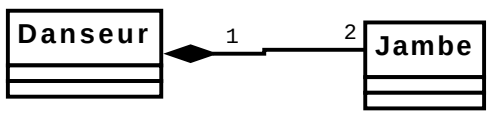






Multiplicité

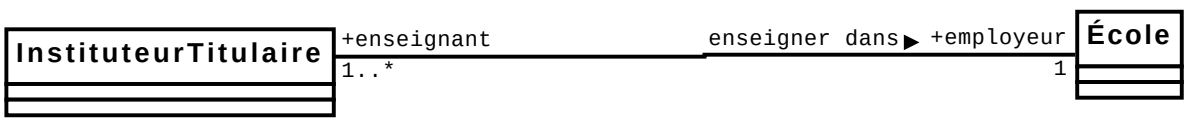
- En cas de lien entre objets, la multiplicité permet de contrôler le nombre d'objets intervenant dans chaque instance de la relation.
- Apparaît à chaque extrémité de la relation et indique le nombre d'objets de la classe appartenant à cette extrémité pouvant s'associer à un seul objet de la classe de l'autre extrémité.



- Danseur possède (*composition*) 2 jambes.

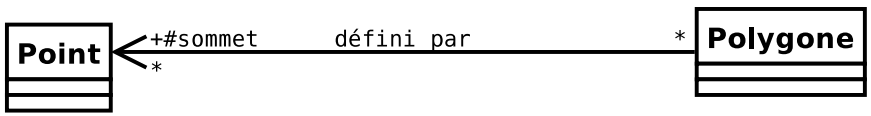
Association

- Association relation sémantique.
- Association entre classe indique que des propriétés sont échangées ou partagées par les classes reliées (indication pour le codage futur).
- Représentée par un trait plein entre les classes. Possède un nom (verbe à l'infinitif), des multiplicités, des rôles, parfois un sens, et parfois une non-navigabilité.



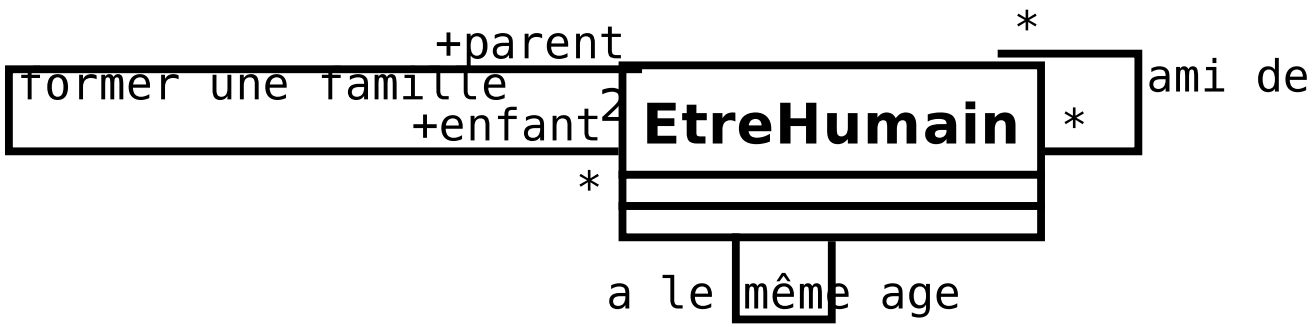
- Une école reçoit l'enseignement d'au moins un instituteur titulaire, un instituteur titulaire enseigne dans une unique école.

Association (suite)



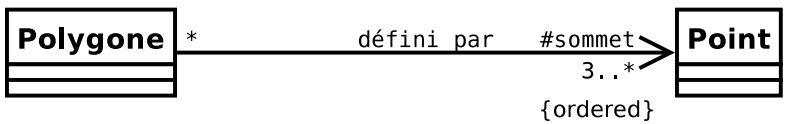
- Un polygone est formé de plusieurs points. Inutile de faire un lien des points vers les figures auquelles ils participent (pas de classe Polygone dans la classe Point).

Association (réflexive)

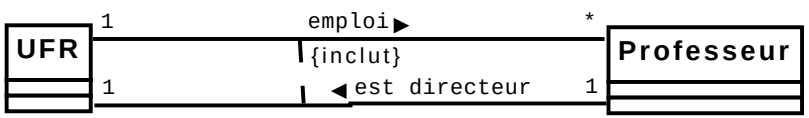


Association avec contraintes

- Présence d'une association indique qu'il y a des propriétés échangées ou partagées.
- Présence d'une contrainte apporte plus d'information.



- Les points sont ordonnés.



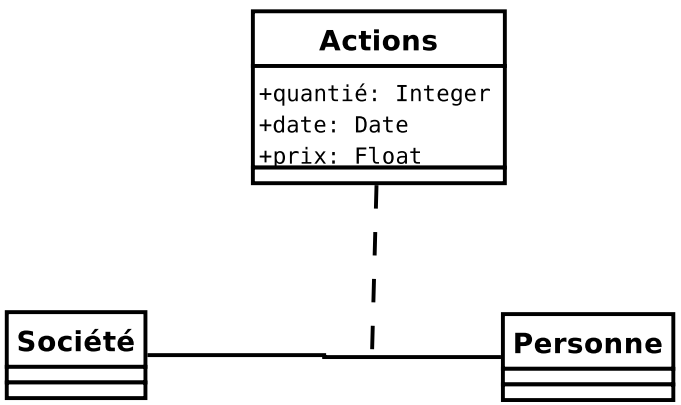
- Le directeur de l'UFR est un des professeurs de l'UFR.

Qualification

- Généralement une classe peut être décomposée en sous-classes ou posséder plusieurs propriétés. Quand on relie deux associations on peut choisir de restreindre la portée de l'association à quelques éléments ciblés (comme un ou plusieurs attributs). Ces éléments ciblés sont appelés un qualificatif .
- Exemple avec un objet référencé une seule fois dans un vecteur.
- Exemple de la banque.

Classe-association

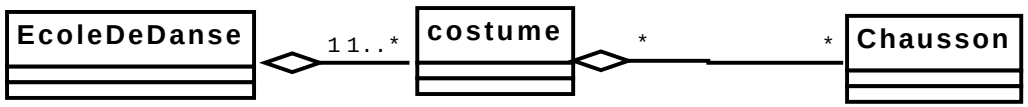
- Une association peut être raffinée et avoir ses propres propriétés, qui ne sont dans aucune des classes qu'elle lie. Dans le modèle objet, seule les classes ont des propriétés, cette association devient donc une classe, appelée classe-association . Classe qui est traitée comme les autres classes.
- Peut avoir des opérations.



Relation d'agrégation

- Une agrégation est une forme particulière d'association.
- Elle représente la relation structurelle ou comportementale d'un élément dans un ensemble.
- Contrairement aux associations l'agrégation est transitive.
- Durée de vie de l'agrégat (le contenant) indépendant des objets qui le constitue).
- Un composant peut apparaitre dans plusieurs agrégats.

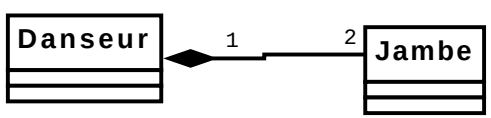
Relation d'agrégation (exemple)



- Un costume appartient à une unique école. Il y a au moins 1 costume par école. Chaque costume possède des chaussons.
- La destriction du costume n'impose pas la destruction des chaussons.
- Une instance de chausson peut apparaître dans plusieurs costumes (on n'aime pas partager ses chaussures).

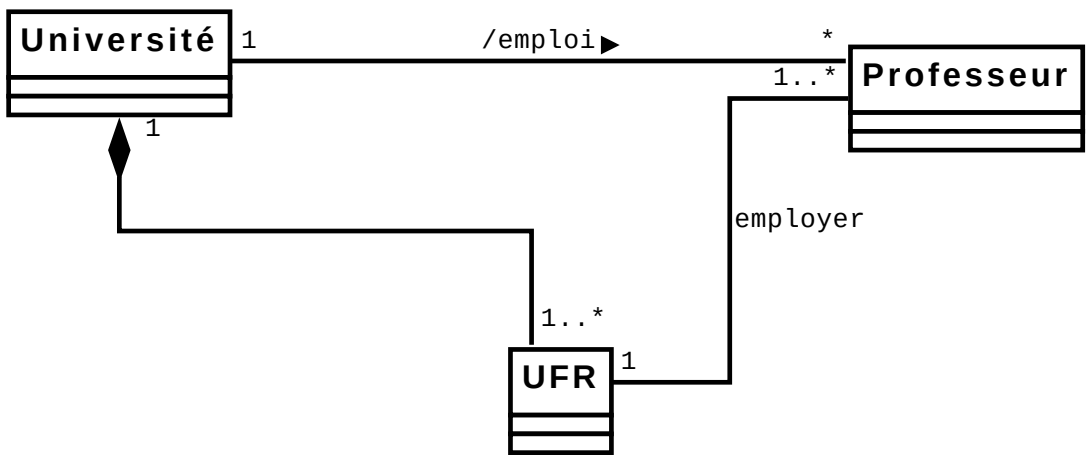
Relation de composition

- Appelée également agrégation composite
- Contenance structurelle entre instances.
- L'élément composite est responsable de la création, de la copie et de la destruction des composants.
- La destruction (resp. la copie) de l'objet composite entraine la destruction (resp. la copie) des composants.
- Une instance de la partie appartient toujours au plus à une instance de l'élement composite.



Association dérivée

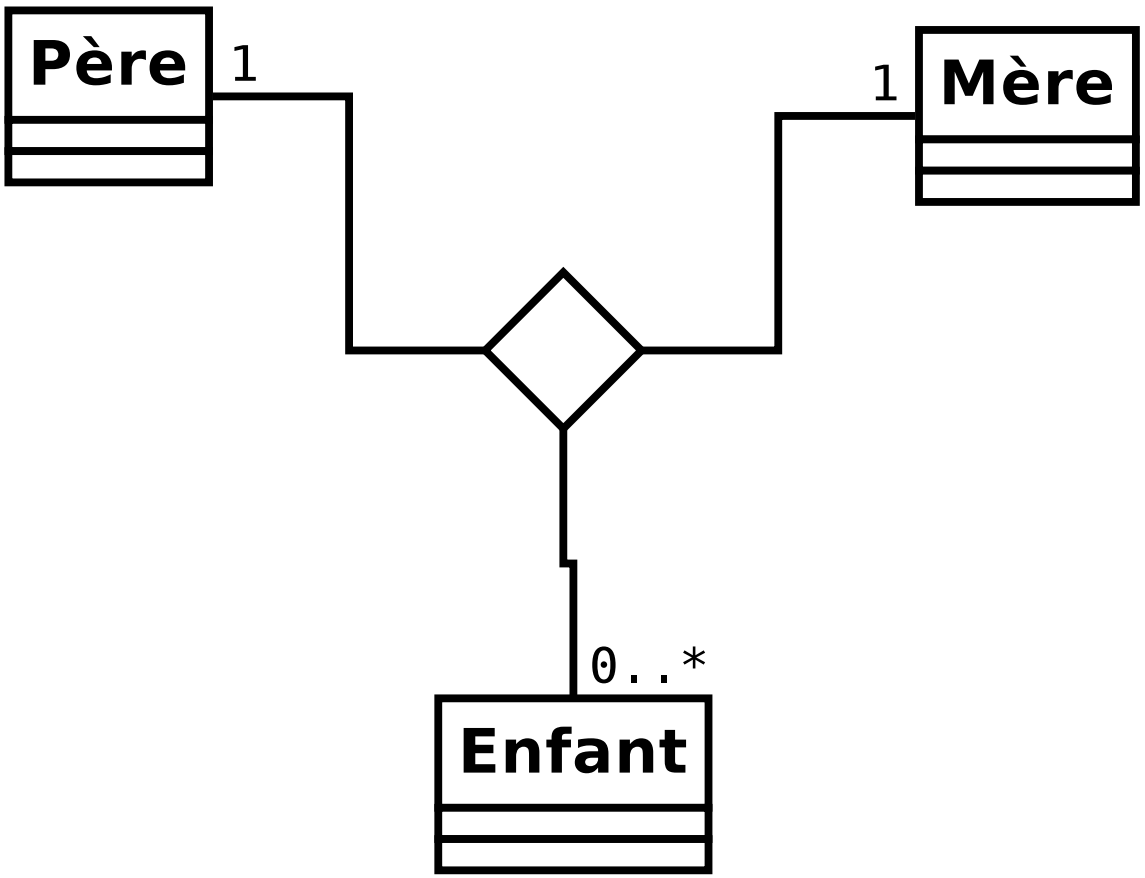
- Qui peut être déduite d'autres association.
- Son nom est précédé d'un /



- Un professeur employé par un UFR et *de facto* employé par l'Université dont dépend l'UFR.

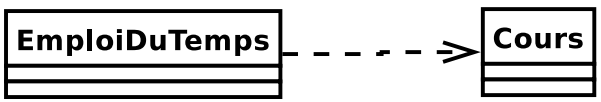
Relation n-aire

- Mettant en cause plus de deux relations.
- À éviter.



Relation de dépendance

- Relation unidirectionnelle expriment une dépendance sémantique.
- La modification de la cible implique le changement de la source.



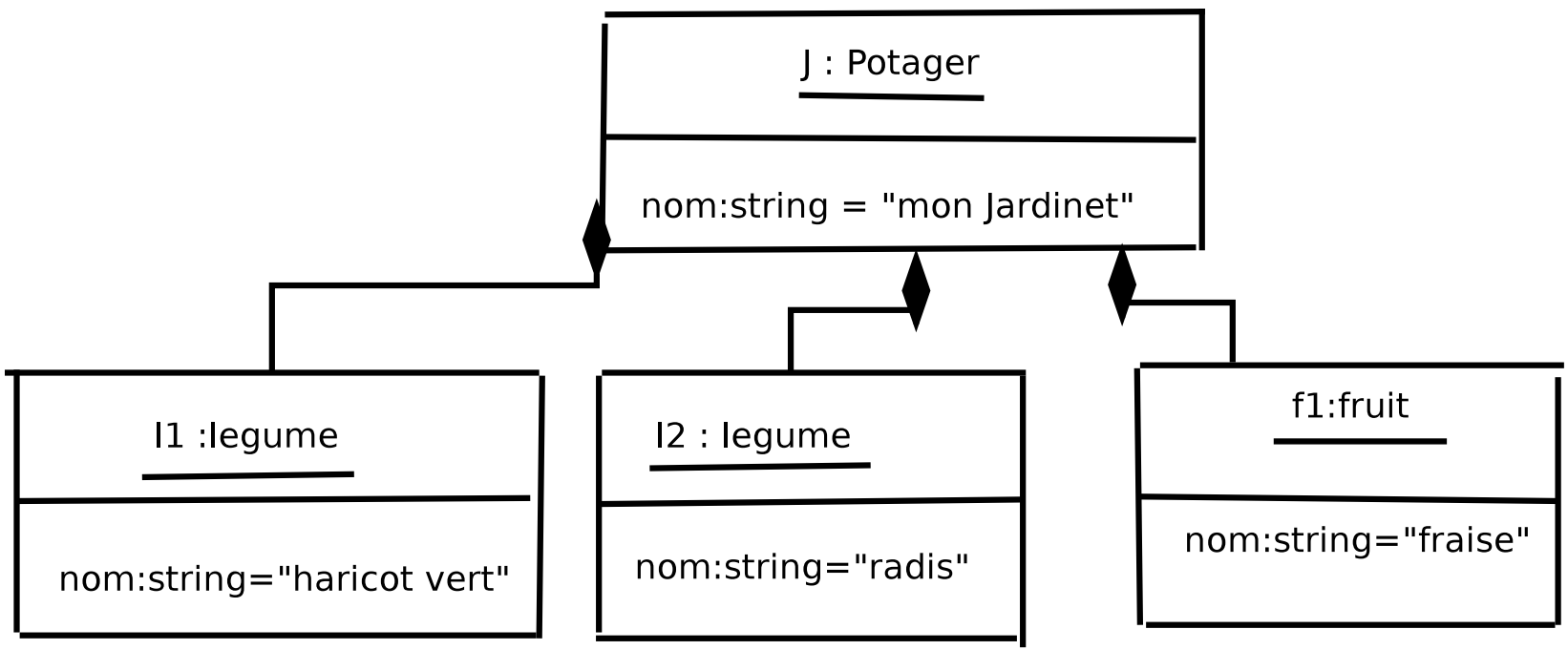
- Le déroulement des cours dépend de l'emploi du temps.

Relation de dépendance (suite)

- Souvent accompagnée d'un stéréotype.
- friend on accorde une visibilité spéciale à la classe source dans la cible.
- dérive la source est calculée à partir de la cible.
- appel appel d'une opération par une autre.
- instanciate une opération de la source crée une instance de la cible.
- send une opération envoie un signal vers l'élément cible.
- trace la cible est une version précédente de la source.
- refine.
- copie.

Diagramme d'objets

- Un objet est une instance de classe.
- Le diagramme de classes représente des règles, le diagramme d'objet représente des faits.



Relation de dépendance d'instanciation

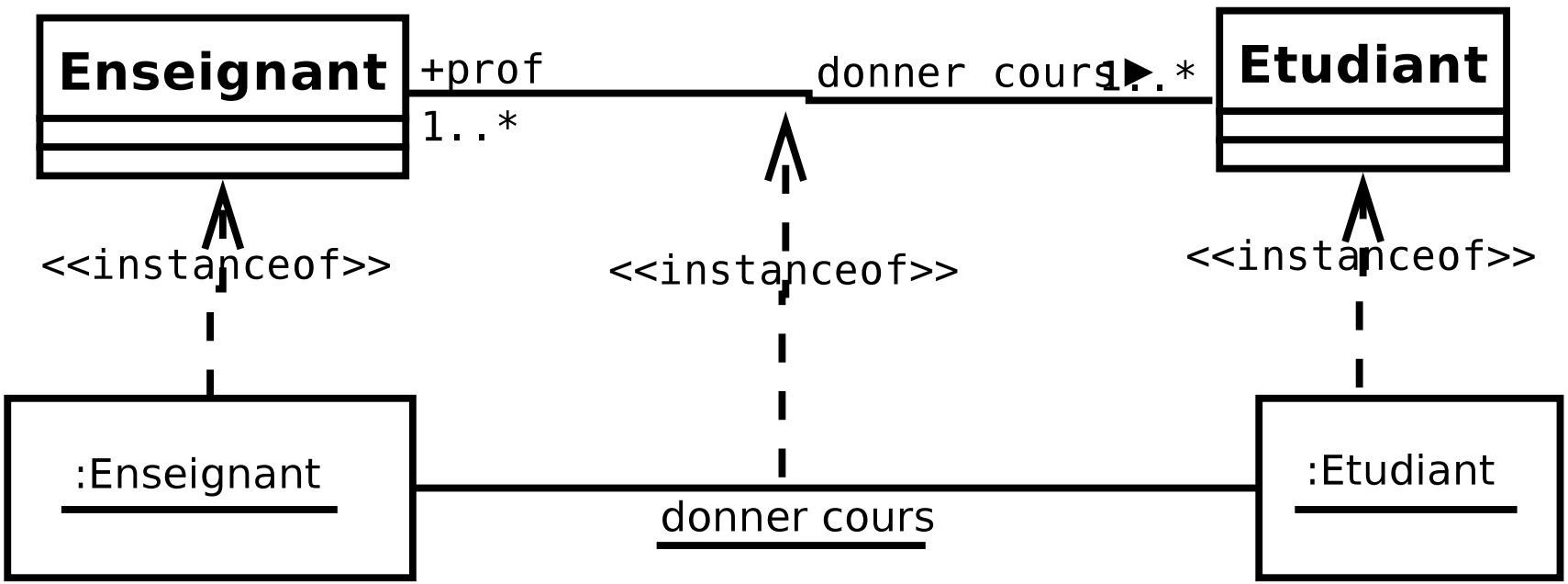


Diagramme de séquence

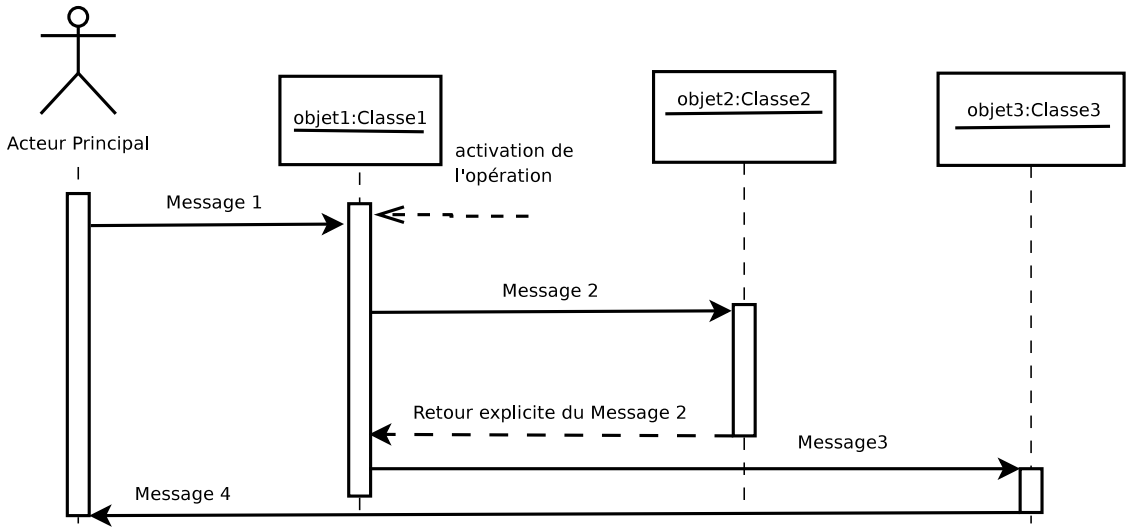
Présentation

- Représentation d'interactions entre objets (instances de classes et acteurs).
- Représentation des échanges de messages dans un ensemble d'objets (pour accomplir une action).
- Envoi de message : communication (unidirectionnelle) entre objets pour susciter une réaction chez le récepteur.
- Objectif : Représentation de l'ordonnancement des intercations entre objets. Utilisé pour détailler un sénario (instance d'un cas d'utilisation).

Présentation (suite)

- Utilisé principalement pour des systèmes ≪temps réel≫, avec des interactions complexes faisant intervenir peu d'objets.
- Un diagramme de séquence est formé :
  - des objets qui intéragissent (instances des classes et acteurs) ;
  - et d'une description des messages échangés.

Formalisme



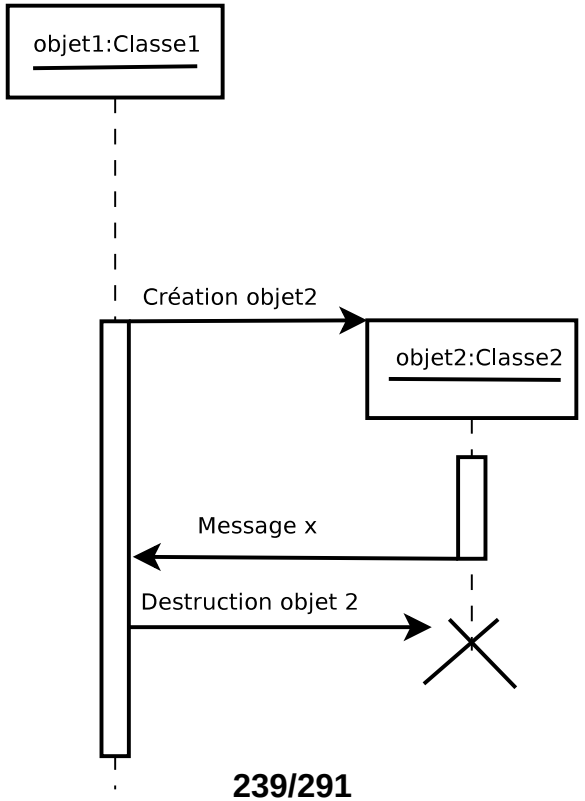
- Acteur principal à gauche, système au centre, acteur secondaire à droite. Normalement, on donne leur rôle, et non leur nom.
- Ligne de vie, axe du temps.
- Activation d'un objet (partie ligne de vie pendant laquelle il effectue une action).

Un message c'est quoi ?

- Généralement :
  - Un signal (es. une interruption)
  - Un appel de fonction. Lors de la reception du message, le destinataire devient actif et exécute la méthode du même nom que le message.
  - La création ou la destruction d'une instance.
- Remarque : une instance peut s'envoyer un message à elle-même.

Création et destruction d'objet dans un scénario

- Si un objet est crée au cours de l'exécution d'un scénario, celui-ci n'apparaît qu'au moment où il est crée.
- Si un objet est détruit dans un scénario, on barre sa ligne de vie à l'aide d'une croix.

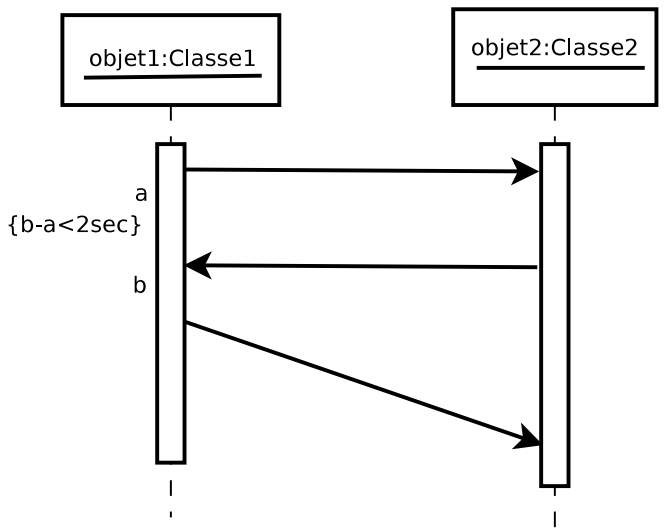


Message synchrone et asynchrone

- Message synchrone : Dans ce cas l'émetteur reste en attente de la réponse à son message avant de poursuivre ses actions. Le message retour peut ne pas être représenté, il est en fait inclus dans la fin d'exécution de l'opération déclenchée dans l'objet destinataire du message.
- Message asynchrone : Dans ce cas l'émetteur n'attend pas la réponse à son message, il poursuit l'exécution de ses opérations. Dans ce cas si l'exécution de l'opération sollicité produit un résultat en retour, un trait en pointillé est utilisé pour représenter le retour vers lobjet appelant. (flêche simple : $\rightarrow$).

Contrainte temporelle

- Des contraintes de chronologie entre les messages peuvent être spécifiées.
- Lorsque l'emission d'un message requiert une certaine durée, il se représente sous la forme d'un trait oblique.



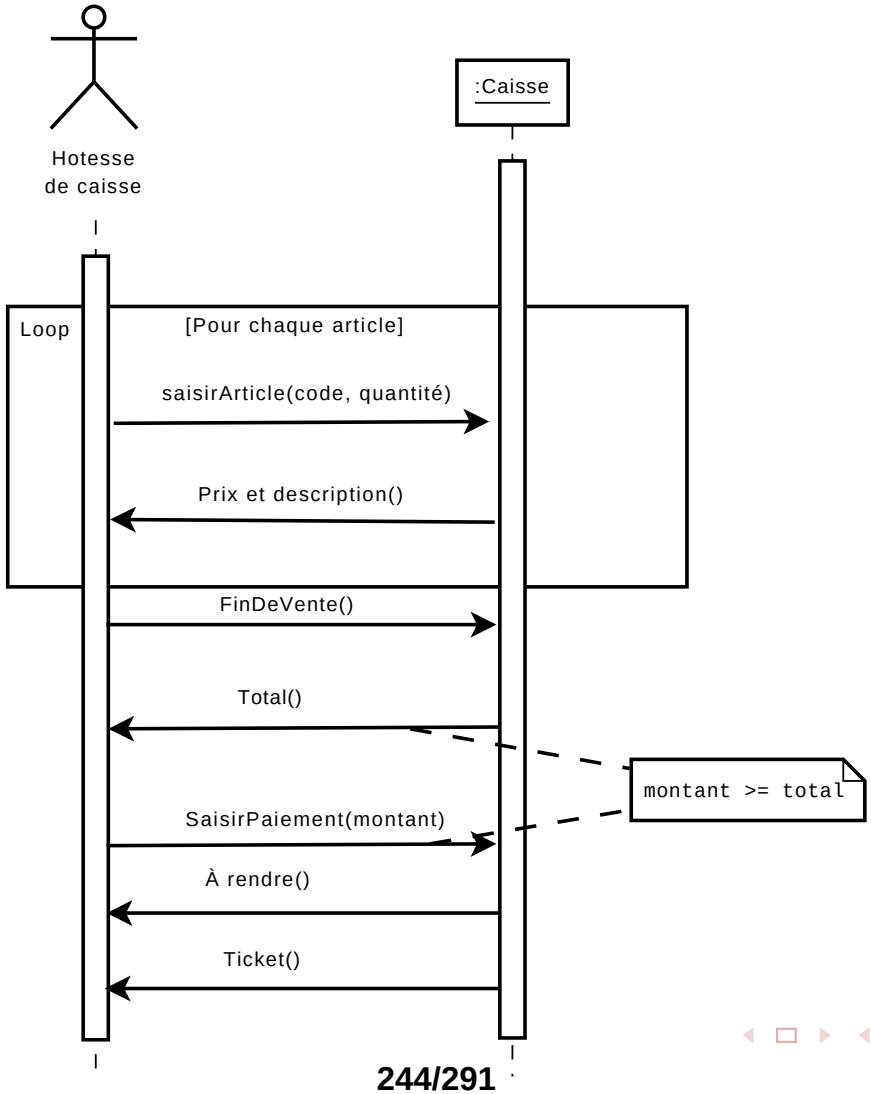
Les cadres d'interaction

- Avec UML 2, arrivée des cadres d'interaction.
- Entre autre : la répétiton, l'alternative, l'obligation, la négation et le reférencement.

La répétition

- Opérateur : loop [min, max, contion]
- On effectue ce qu'il y a dans le cadre min fois, puis tant que la condition est réalisé et pas plus de max fois.
- min, max et condition sont optionnels.

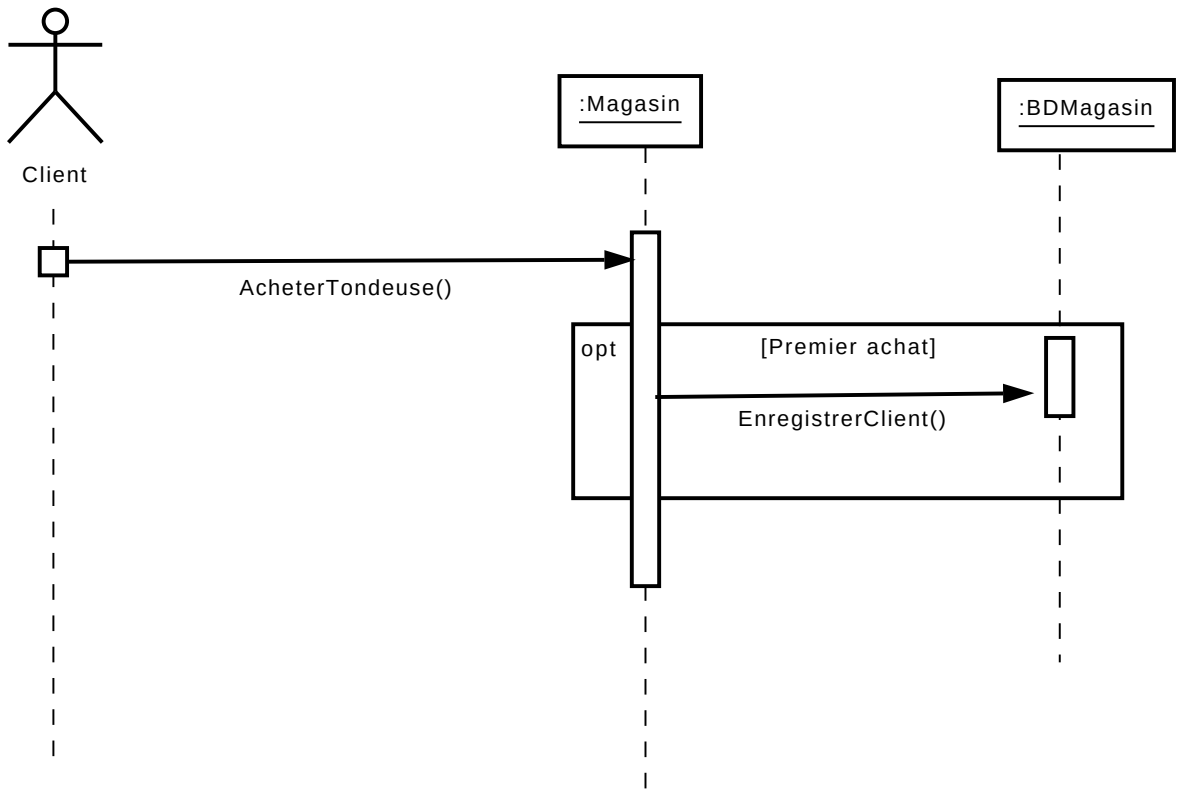
Exemple : la caisse enregistreuse



L'alternative simple

- Opérateur : opt [condition]
- Les messages contenus dans le cadre ne s'envoient que si la condition est réalisé.

Exemple : Centrale d'achat

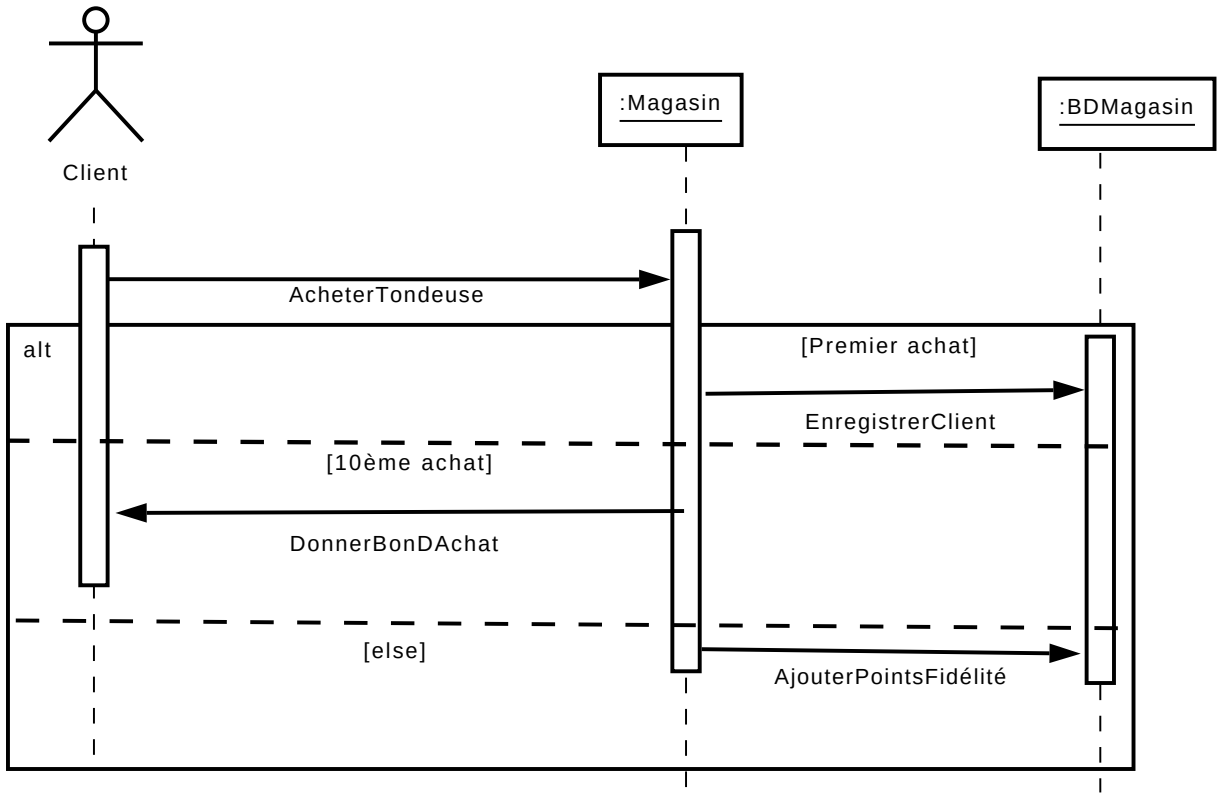


- Si c'est le premier achat du client, on l'enregistre.

L'alternative multiple

- Opérateur : alt [condition][else]
- Les messages contenus dans le cadre ne s'envoient que si la condition est réalisé.

Exemple : Centrale d'achat (le retour)



- Si c'est le premier achat du client, on l'enregistre.
- Au 10 ème achat, on lui donne un bon d'achat.
- Sinon, on lui ajoute des points de fidélité.

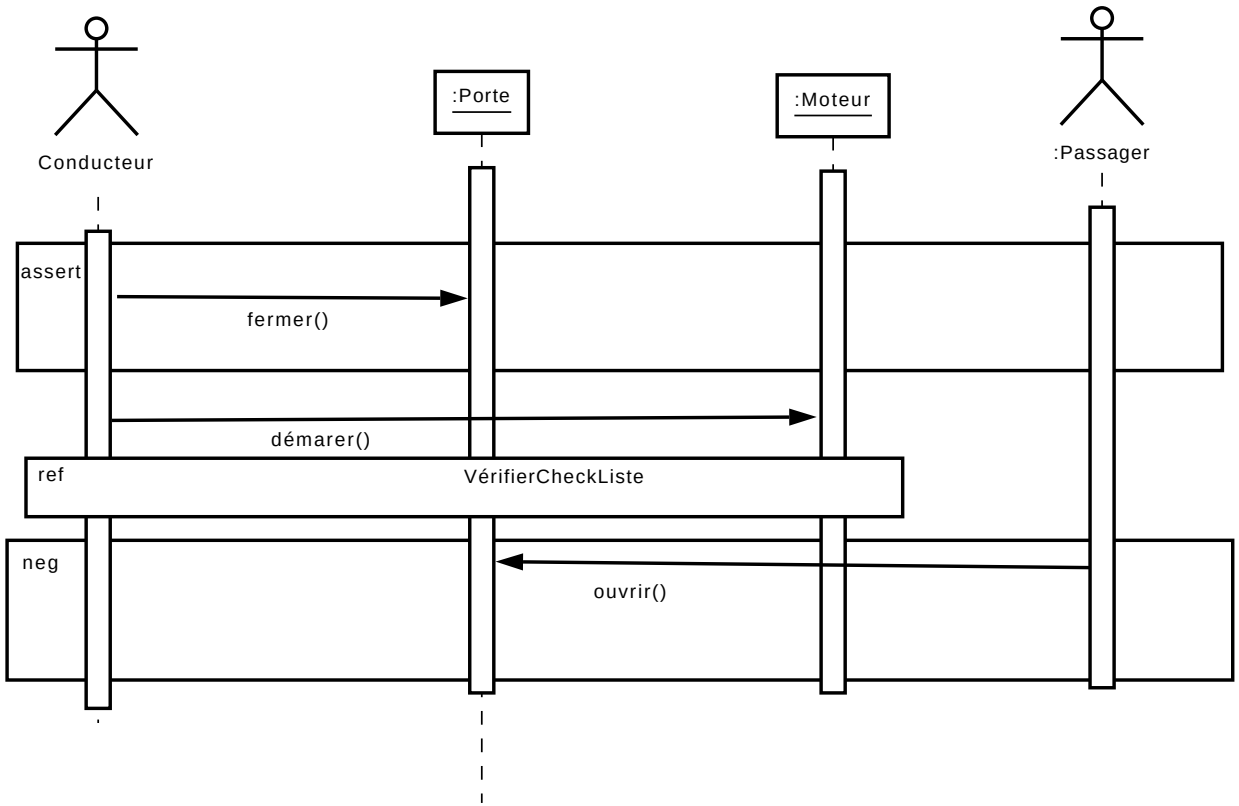
«On doit» ou «on peut pas»

- L'obligation :
  - Opérateur : assert
  - Les messages contenus dans le cadre doit obligatoirement être envoyés avant que l'on passe à la suite.
- La négation :
  - Opérateur : neg
  - Les messages contenus dans le cadre ne peuvent pas être envoyés.

Le référencement

- Opérateur : ref
- Ce qu'il a dans le cadre est référencé dans un autre diagramme de séquence.

Exemple : le train



- Le train ne démare que si les portes sont fermés.
- Pendant que le train roule, on ne peut pas ouvrir les portes.

Diagramme d'états-transitions

Introduction

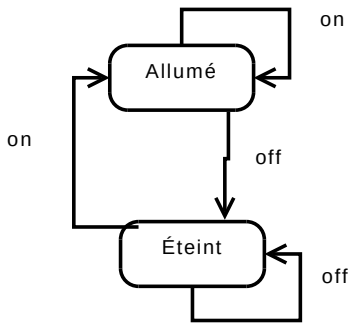
- Les diagrammes d'états-transitions décrivent le comportement interne d'un objet à l'aide d'automates à états finis.
- Représente les séquences possible d'états et d'actions que peut traiter une instance de classe au cours de son cylce de vie en réaction à des évènements discret (signaux, invocation de méthode). Une instance de classe et une seule.
- Vision partielle du système.
- Automate à états finis, le comportement des sorties dépend de l'états des entrées et de l'historique.

Utilité

- Spécifier le comportement d'un objet :
  - en définissant explicitement les évènements auxquels peut répondre un objet en fonction de l'état dans lequel il se trouve,
  - et donc en définissant les événements auxquels il ne peut pas répondre dans certains états.

Notion d'automate à états finis

- Historique des sollicitations passés est représenté par un état global.
- État global : jeu de valeurs d'un objet, pour une classe donnée, produisant la même réponse face aux évènement.
- État représenté par un rectangle aux coinx arrondis.



- Interrupteur du vidéoprojecteur.

État et état global

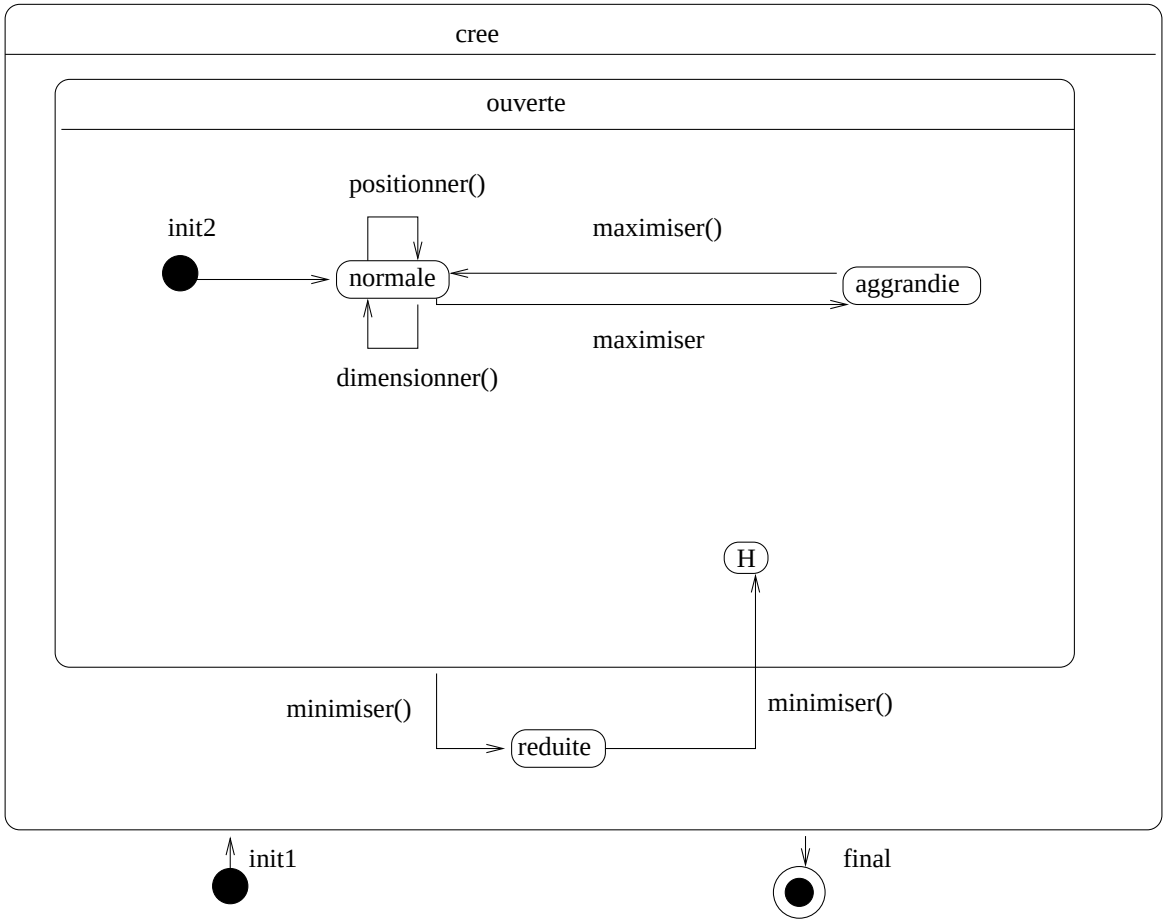
- État : un état de l'automate à états finis. Représente une période de la durée de vie de l'objet (attente d'un évènement, ou action).
- Un état peut être un état composite (i.e. contenir des états – cf exemple plus loin).
- État global : Le jeu des états (de l'automate d'états finis) actifs à un moment donné.

État final, état initial

- État initial : pseudo-état de départ. L'objet est crée, il rentre dans l'état initial.

- État final : pseudo-état qui indique que le diagramme d'état transition est fini.

Des états dans tous leurs états



- État composite : Comportement d'une fenêtre.

Évènement

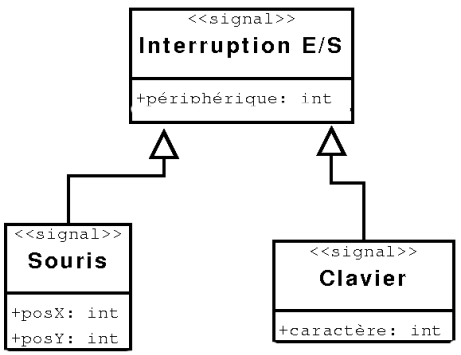
- Quelque chose se produit pendant l'exécution d'un système et mérite d'être modéliser.
- Évènement : Entité reçue par un objet qui déclenche une transition vers un autre état (c'est instantané).
- Évènement se produit à un instant précis et est dépourvu de durée.
- 4 types d'évènements : signal , appel , changement et temporel .

Évènement de type signal (*signal*)

- La réception d'un signal asynchrone, explicitement émis par un autre objet, génère un évènement de type signal .
- Un signal est un type de classeur destiné explictement à véhiculer une communication asynchrone à sens unique.
- L'objet expéditeur crée et initialise explicitement une instance de signal, et l'envoie à un objet (qui peut être lui même) ou à tout un groupe d'objets.
- L'expéditeur n'attend pas la réponse, il continue à ≪vivre sa vie≫.
- La réception du signal est un évènement pour l'objet destinataire.

Un signal, ça a quelle tête ?

- Classe sans opération stéréotypé par le mot signal.
- Les attributs sont concidérés comme des paramêtres.



- Syntaxe : <nom_évèment>([<paramètre> : <type > [; <paramètre > :<type>. . .]]).
- Un signal, quand ? Quand plusieurs processus ou objets actifs sont en concurrence. En particulier lorsque l'instance du receveur est muni d'un contrôle indépendant de celui de l'appelant.

Évènement d'appel (*call*)

- Un appel de méthode sur l'objet courant génére un évènement de type appel.
- Les paramètre de la méthode sont ceux de l'évènement.
- La synthaque de call et la même que celle de signal.
- En revanche, call est une méthode (et donc aussi dans le diagrame de classes).

Évènement de type changement (*change*)

- Le passage de faux à vrai (la satisfaction) de la valeur de vérité d'une condition booléenne, sur des valeurs attributs, génère implicitement un évènement de type changement.
- Syntaxe : when(<condition_booléenne>)
- Un évènement de changement est évalué continuellement jusqu'à ce qu'il devienne vrai, et c'est à ce moment-là que la condition se déclenche.

Évènement temporel (*after* ou *when*)

- Les évènements temporels sont générés par le passage du temps.
- Soit spécification de manière absolue. Arrive à une date précise.
- Soit spécification de manière relative. Une certaine durée vient de sécouler.
- Par défaut le temps commence à s'écouler dés l'entrée dans l'état courant.
- Syntaxe relative : after (<durée >).
- Syntaxe absolue : when (date = <date >).

Transition

- Une transition définit la réponse d'un objet à l'occurence d'un évènement.
- Elle lie un état 1 à un état 2.
- Elle indique qu'un objet peut passer d'un état 1 à un état 2 et éxécuter certaines activité, si un évènement déclangeur se produit et si la condition de garde est vérifiée.
- Syntaxe : [<évènement>] ['[' <garde ']'][' / '<activité>].
- Un évènement peut être le déclencheur de plusieurs transitions. Elles doivent donc avoir des condition de garde différentes, car une seule s'éxécutera.
- Si deux transitions sont activées en même temps, une seule se déclanche et le choix n'est pas prévisible (i.e. pas déterministe).

Condition de garde

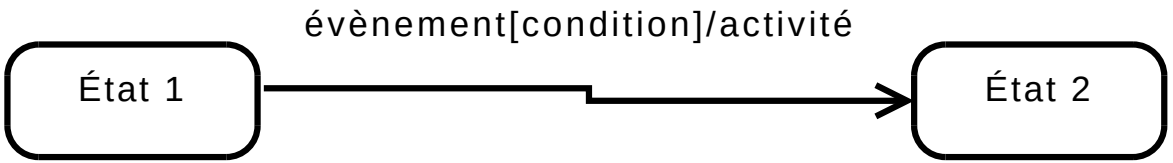
- La condition est évaluée une fois que l'évènement déclencheur de la transition a lieu et que le destinataire le traite. Si elle est fausse, la transition ne se déclenche pas et la condition n'est pas réévaluée. Sinon, elle se déclenche et ses effets se produisent.
- Ne pas confondre avec l'évènement de changement !

Effet d'une transition

- Lorsqu'une transition se déclenche (on parle également de tir d'une transition) son effet (son activité) s'éxécute.
- Cela peut être :
  - une opération primitive (comme une instruction d'assignation) ;
  - l'envoi d'un signal ;
  - l'appel d'une opération ;
  - une liste d'activité, etc.

Transition externe

- Transition qui modifie l'état actif.



Transition d'achèvement

- Transition dépourvue d'évènement déclencheur explicite.
- Se déclenche à la fin de l'activité contenue dans l'état source.
- Peut contenir une condition de garde évalue uniquement quand l'activité s'achève.

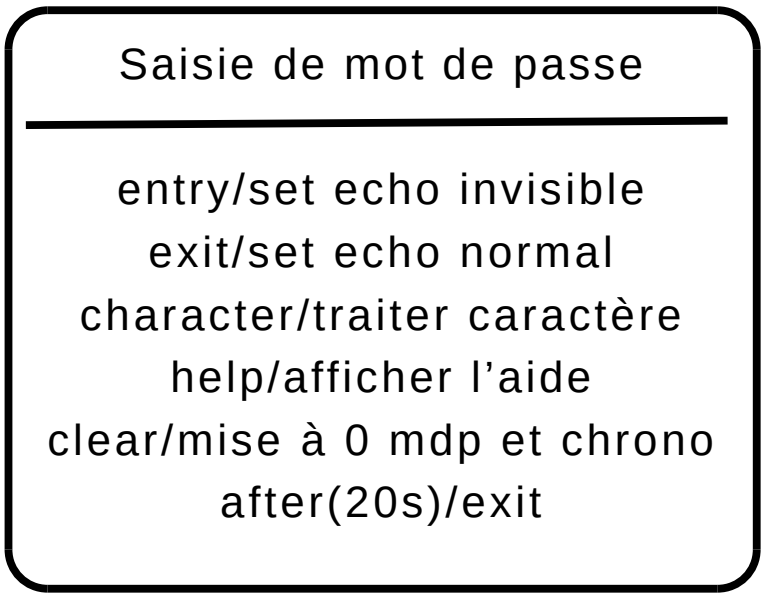
Transition interne

- Transition ne possède pas d'état cible et que l'état actif reste le même à la suite de son déclenchement.
- Les règles de déclenchement des transistions internes sont les mêmes que celles des transistions externes.
- Ne sont pas représentées par une flêche, mais par un compartiment spécial.
- On parle d'action pour une transition externe. C'est inintéruptible et rapide. On parle d'activité pour une transition interne. Cela peut être intérompu.

Transition interne (suite)

- Possède des noms d'évènement prédéfinis correspondant à des déclancheurs particuliers : entry , exit , do et include.
  - entry : permet de spécifier une activité qui s'accomplit lorsqu'on rentre dans l'état.
  - exit : permet de spécifier une activité qui s'accomplit lorsqu'on sort de l'état.
  - do : Une activité do commence dès que l'activité entry est terminé. Lorsque cette activité termine, une transition d'achèvement peut être déclenchée, après l'exécution de l'activité exit. Si une transition se déclenche pendant l'activité do, cette dernière s'intérrompt et l'activité exit est déclenchée.
  - include : permet d'invoquer un sous diagramme d'états-transitions.

Transition interne (suite)



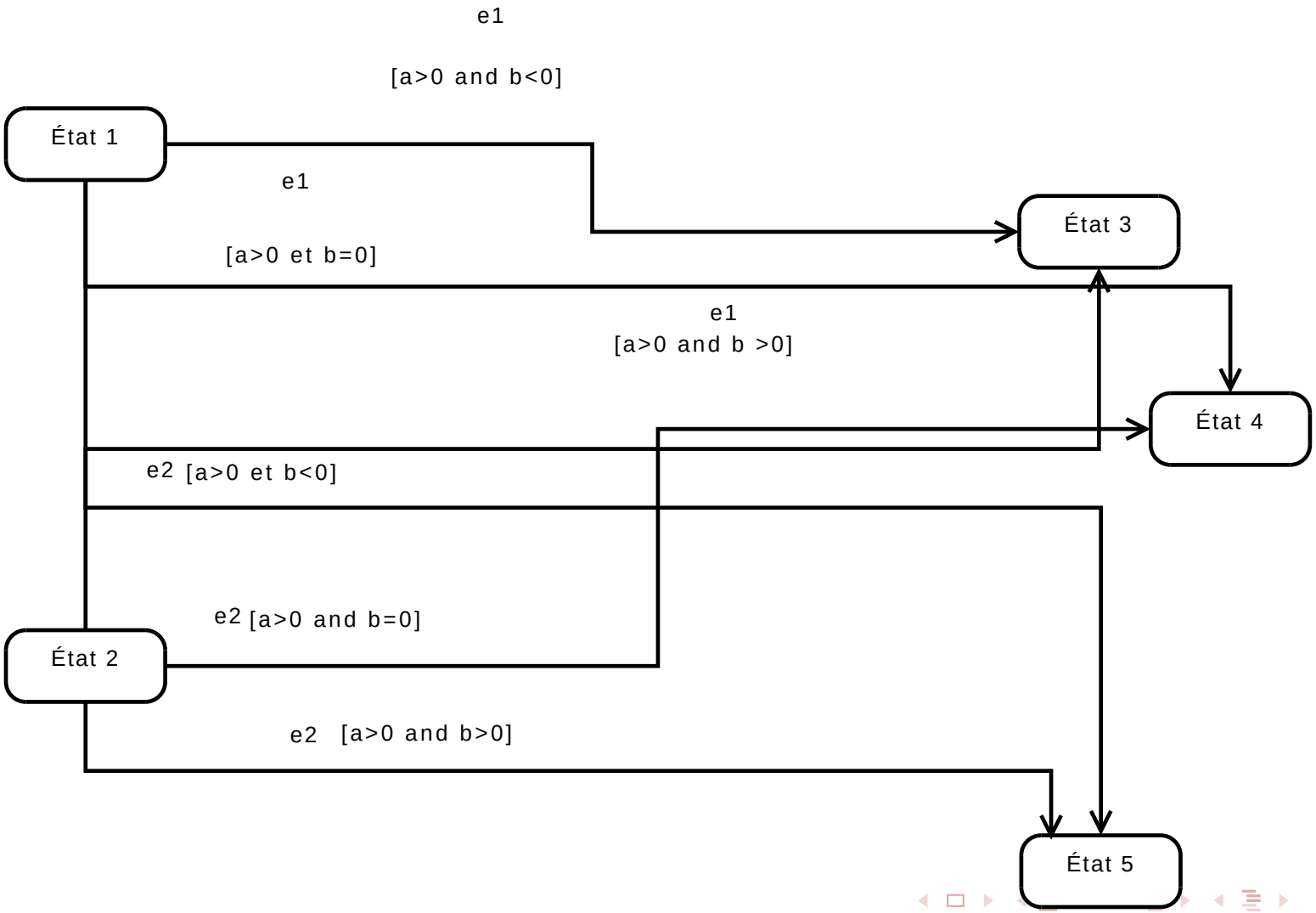
Point de décision

- Il est possible de représenter des alternatives pour le franchissement d'une transition.
- Points de jonction (cercle plein noir).
- Point de choix (petit losange).

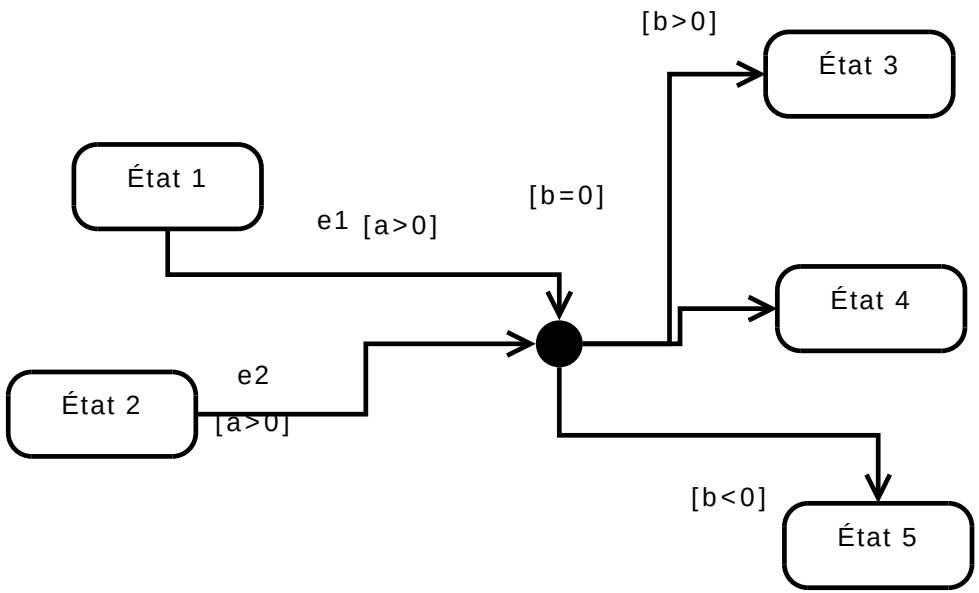
Point de jonction

- Artefact graphique pour segmenter un segment de transition.
- Plusieurs transition peuvent arriver/partir d'un point de jonction.
- Tous les chemins qui passe par un point de jonction sont valides.
- Ce n'est que du sucre syntaxique toutes les gardes du chemin doivent être vraies dés le franchissement du premier chemin.

Point de jonction – puissance



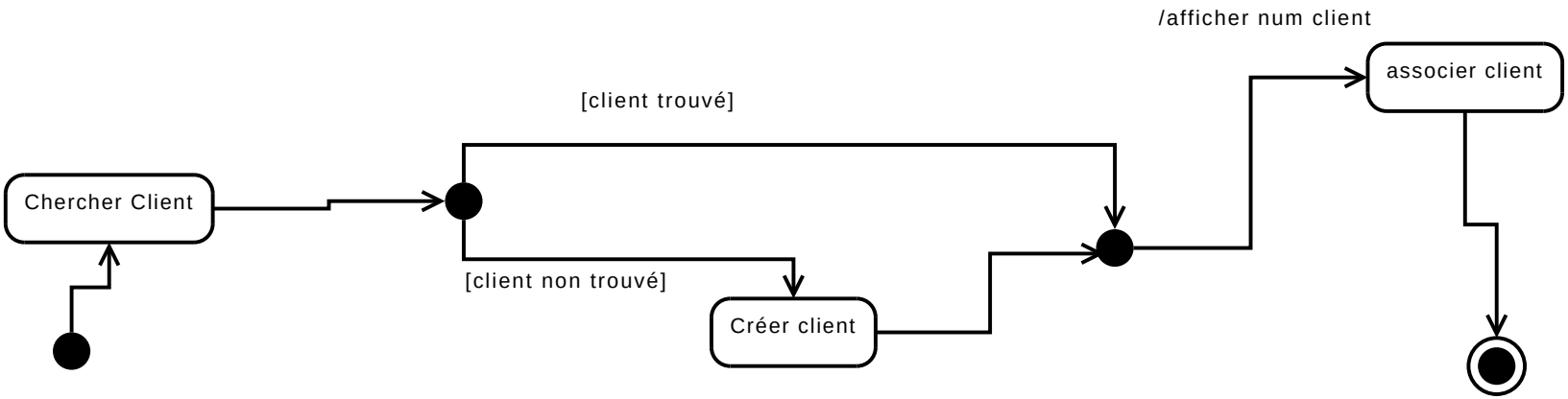
Point de jonction – puissance



Point de jonction – alternative

- On peut utiliser un point de jonction pour effectuer une alternative.
- Exercice : On veut associer un client à une commande. Pour ce faire, on recherche le client. S'il n'existe pas, on le crée. Une fois le client trouvé ou crée, on affiche son numéro, puis on passe à l'étape ≪associer client ≫.

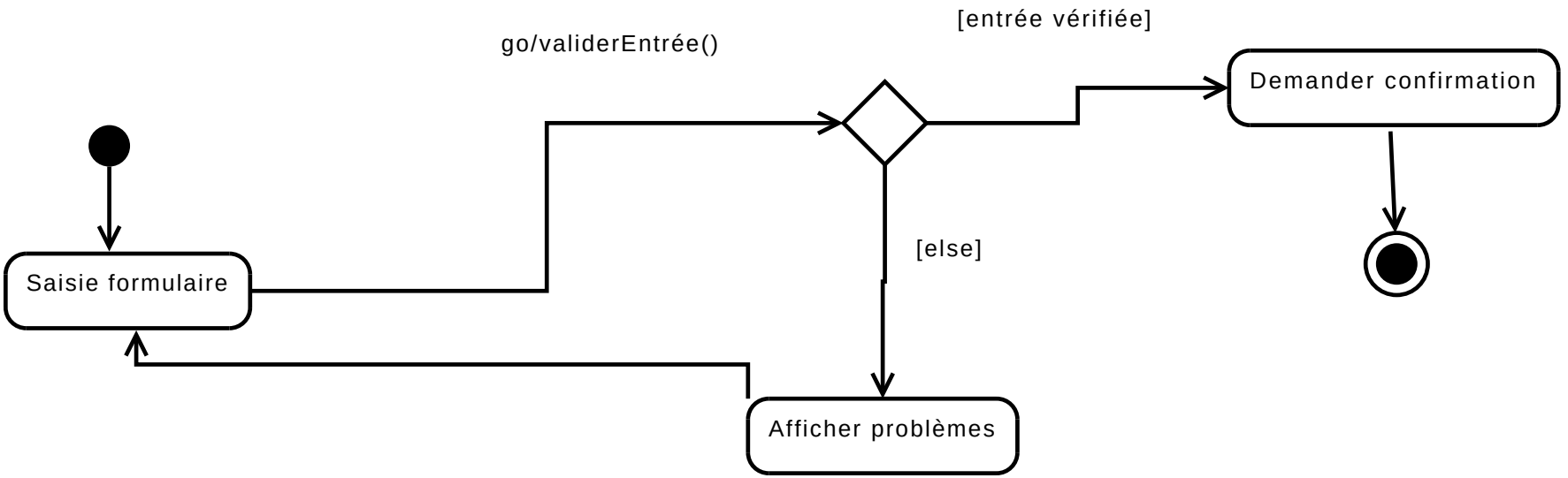
Point de jonction – alternative – correction



Point de choix dynamique

- Contrairement au point de jonction, les gardes sont évaluées au moment du franchissement du point de choix.
- Cela permet de baser ses choix sur des résultats obtenus en parcourant les segments précédents.

Point de choix – Exemple

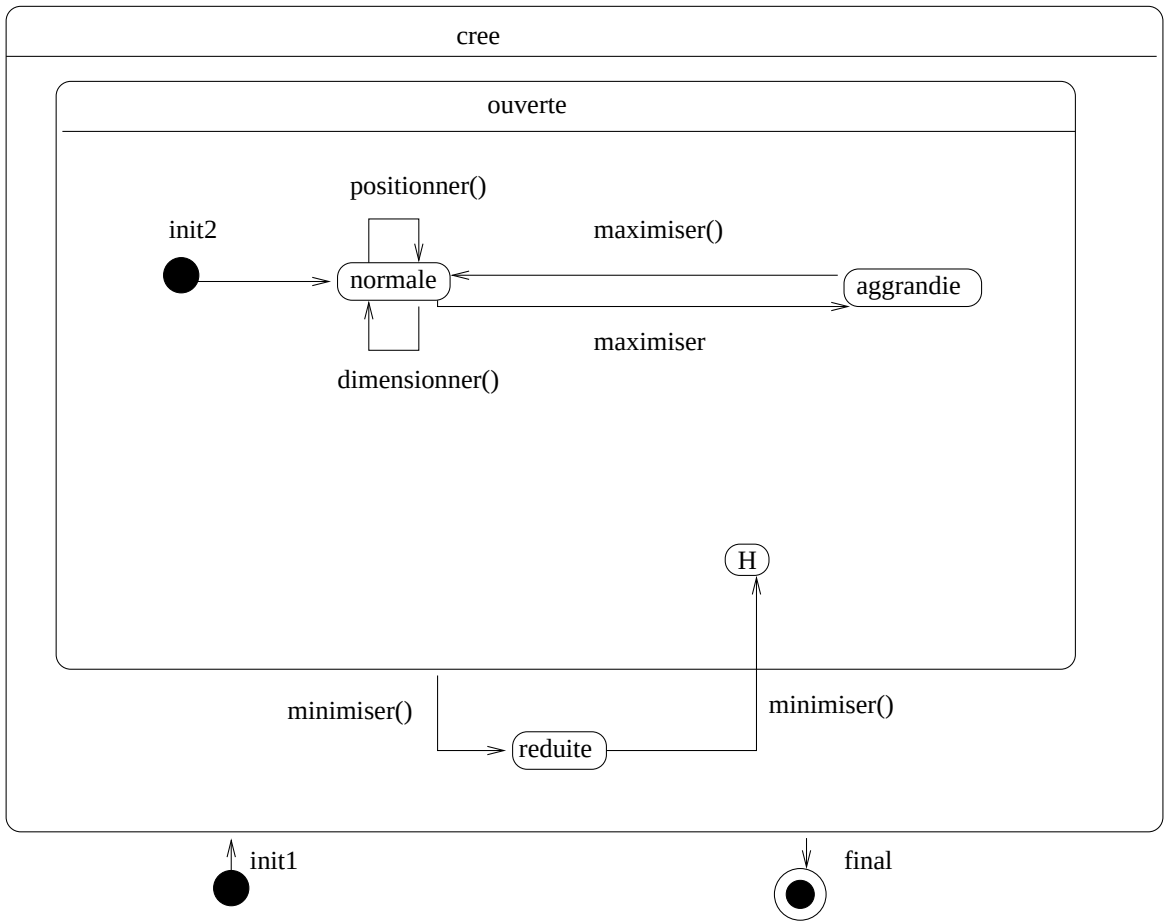


- La vérification de la cohérence des données fournies est réalisée par validerEntrée().

Un peu plus loin dans les états composites

- Un état composite , par opposition à un état simple , est un état comprenant des sous-états.
- Chaque sous-état peut être décomposé en sous-état sans limite de profondeur.
- Une transition qui atteint l'état final d'un sous-diagramme correspond à la fin de l'action associé à l'état emcapsulant.
- La fin d'une activité interne d'un état peut être associée au déclanchement d'un changement d'état, représenté par une transition sans étiquette qui quitte l'état externe courant.

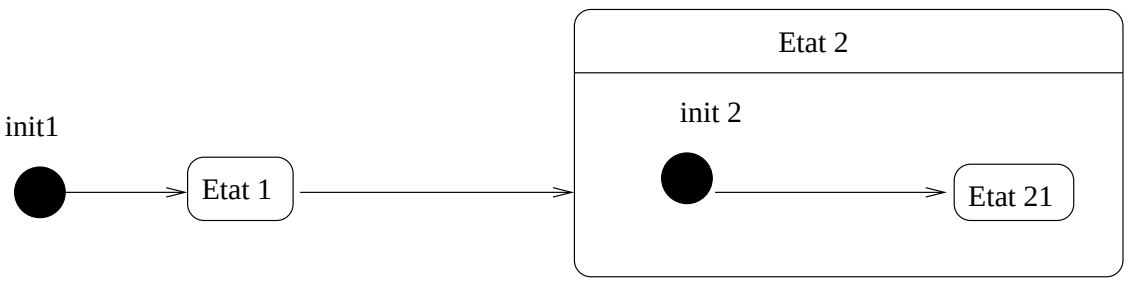
État composite



- État composite : Comportement d'une fenêtre.

Transition et état composite

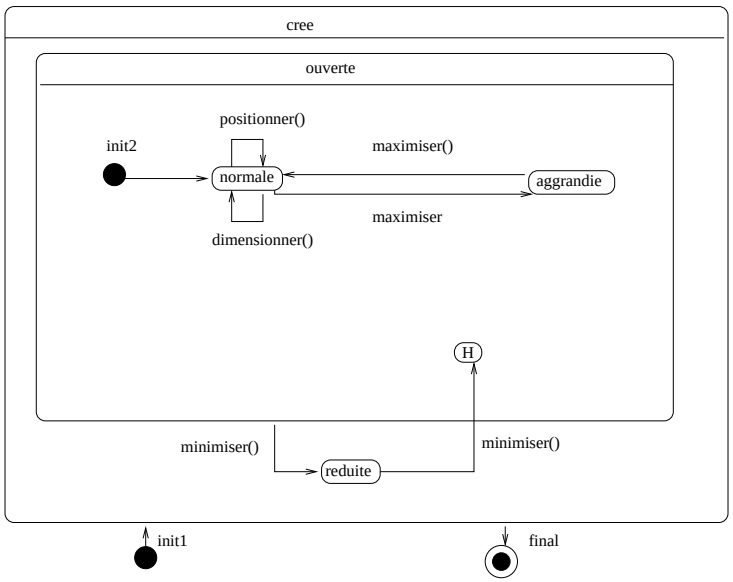
- Une transition, dont la cible est la frontière d'un état composite, est équivalente à une transition ayant pour cible l'état initiale de l'état composite.



- On part de l'état 1 pour aller à init2.

Transition et état composite (suite)

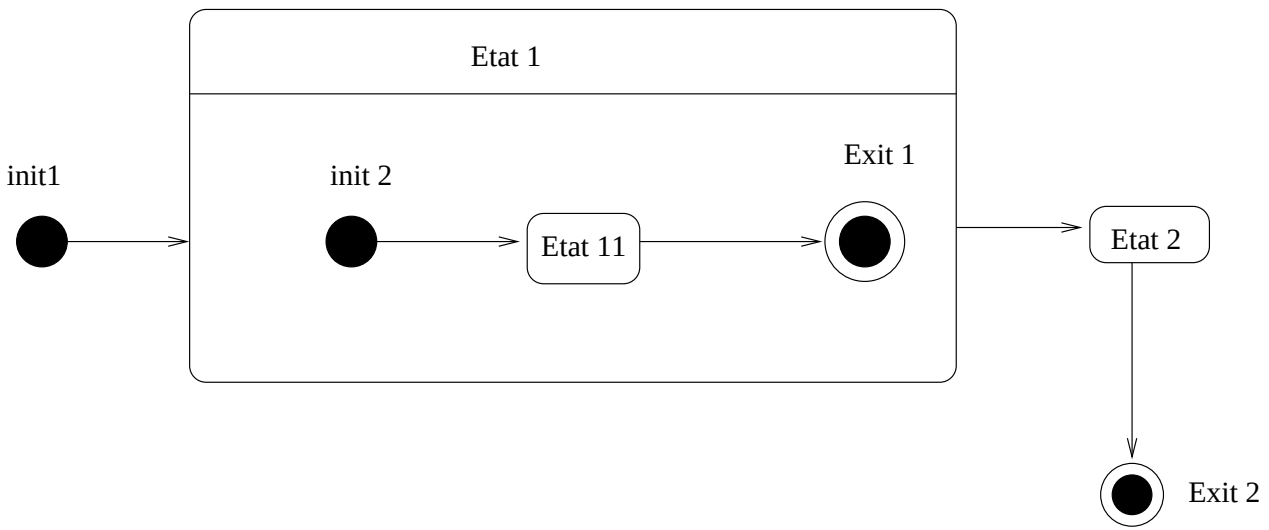
- Une transition, dont la source est la frontière d'un état composite, est équivalente à une transition qui s'applique à tout sous-état de l'état composite source.



- On peut minimiser une fenêtre normale ou une fenêtre aggrandie.

Transition et état composite (suite)

- Si cette transition ne comporte pas d'évènement, elle est franchissable quand l'état final de l'état composite est atteint.

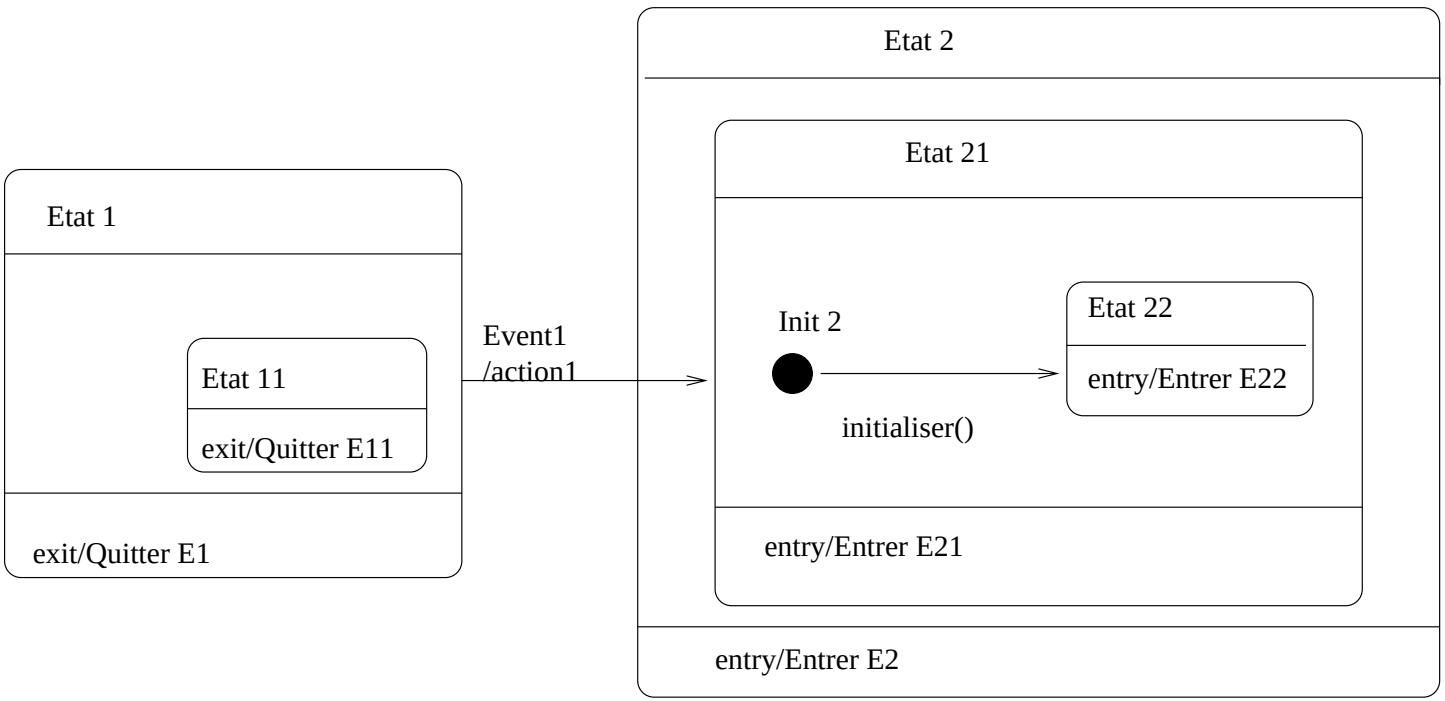


- Lorsqu'on arrive à Exit 1, on passe à Etat 2.

Transition et état composite (suite)

- On peut franchir les frontières, i.e. entrer dans un sous-état directement sans avoir de transition vers l'état englobant.
- Dans tous les cas, les activités entry et exit sont éxécutées. Si on entre ou sort d'un sous-état, attention à ne pas oublier les état englobants.

Transition et état composite (fin)



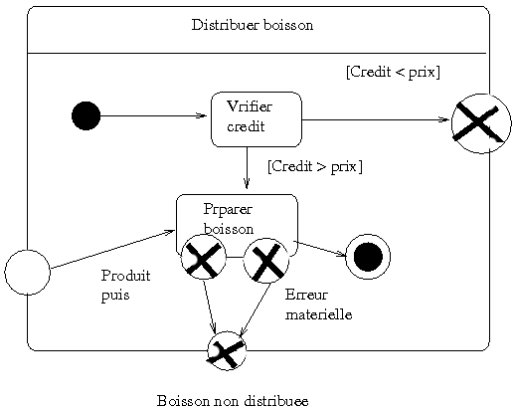
- Depuis l'Etat 11, la reception event1, la séquence d'action QuitterE11, QuitterE1,action1,EntrerE2, entrerE21, initialiser, EntrerE22 et nous place dans l'Etat 22.

Historique et état composite

- Un cercle contenant H est l'état histoirque qui mène au dernier état visité de ce niveau.
- Un cercle contenant H* est l'état historique profond qui mène au dernier état visiter dans la région quelque soit son niveau d'imbrication.

Interface : Les points de connexions

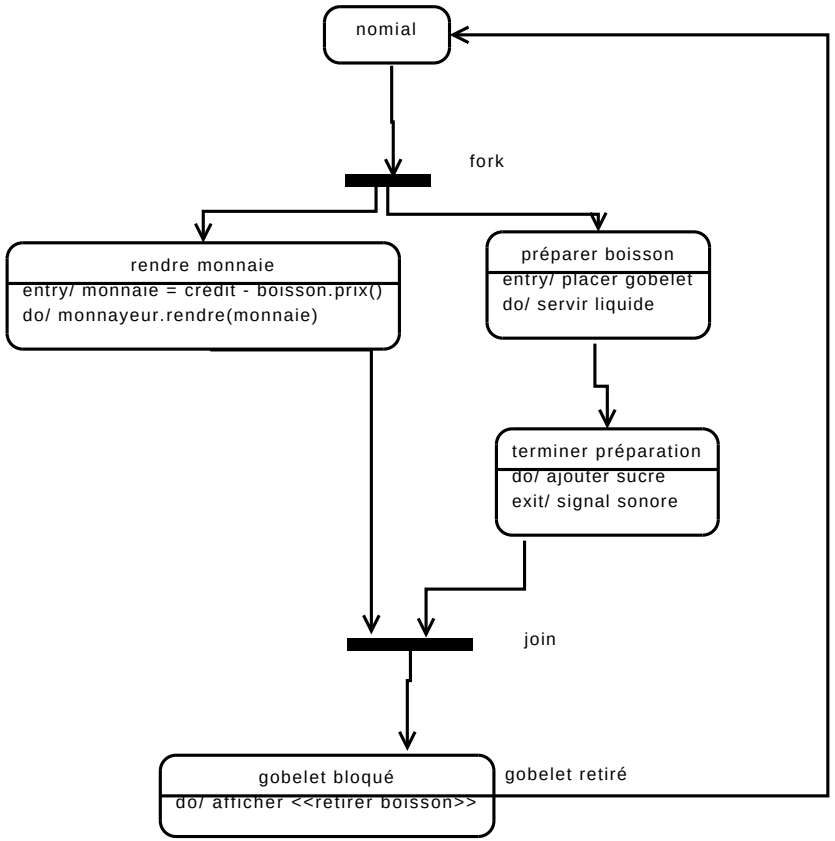
- Il est possible de masquer les états composites.
- Mais que faire lorsqu'un état composite a plusieurs état d'entrée ou plusieurs état de sortie ?
- On utilise les points de connexion.
- Un cercle vide pour les points d'entrée.
- Un cercle barré pour les points de sortie.



Gestion de la concurrence

- Un état composite doté de sous-états peut avoir un comportement concurrent, à travers l'utilisation de régions concurrentes .
- Celles-ci permettent de représenter des zones où l'action est réalisée par des flot d'exécution parallèles.
- Quand un état présente des région concurrentes, elles doivent toutes atteindre leur état final pour que l'action soit considéré comme terminée

Gestion de la concurrence (fin)



- Distributeur de boisson. Tansition concurrente fork , transition concurrente joint .