

Premiers pas dans la recherche des Cus, des classes et du glossaire métier

Cedric Dumoulin
Cours de COA

Plan

Objectifs

Le document a produire

Introduction au diagramme de CUs

Introduction au diagramme de classes

Pourquoi un glossaire métier

Premiers pas dans l'identification des CU, classes et termes métiers

Objectifs

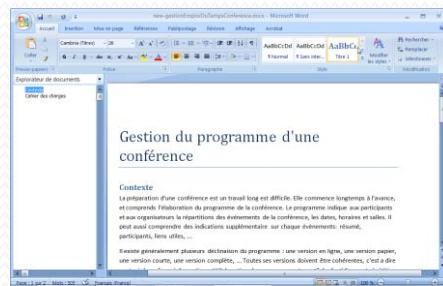
- Produire un document proposant une solution pour l'application
 - Que doit-il contenir ?
 - Comment le remplir ?

Objectifs

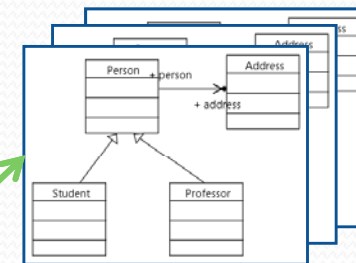
- A partir d'un cahier des charges ou d'un recueil des besoins ...
- Comment identifier ...
 - les fonctionnalités
 - Les premières classes
 - Les termes métiers
- ... d'une application ???

Objectifs

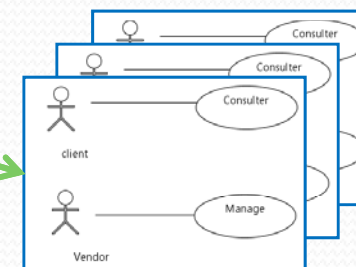
- **Fonctionnalités** → Cas d'Utilisations (**CU**) et diagrammes de CU
- Les **objets métier** → classes et diagrammes de classes
- Les **termes métiers** → glossaire métier



Recueil des besoins



classes



CUs

Terme	Type	Définition
enchère		
vendeur		Personne mettant en vente des articles dans une enchère
article		Objet mis en vente
Acheteur		Personne consultant le catalogue et pouvant faire des offres sur les articles
offre		Montant proposé pour l'achat d'un article
Compte		Données identifiant un vendeur ou un acheteur
Clore une enchère		Terminer une enchère. L'article sera vendu à l'acheteur ayant fait la plus grande offre
Catalogue		Ensemble des articles en vente
Faire une offre		Spécifier un montant pour acheter un article

glossaire
métier

Plan

Objectifs

Le document a produire

Introduction au diagramme de CUs

Introduction au diagramme de classes

Pourquoi un glossaire métier

Premiers pas dans l'identification des CU, classes et termes métiers

Contenu du document

- Glossaire métier
- Glossaire de l'ingénierie des besoins
- Classes composants l'application
 - Overview et details
 - Descriptions, relations
 - Plusieurs diagrammes de classes
- Cas d'utilisation
 - Overview et details
 - Descriptions, relations
 - Plusieurs diagrammes de CU
- Scénario textuels
- Architecture de l'application
- Diagrammes de séquences
- Comportement dynamique de certaines parties
- Maquette

Exemple de structure du document

- 1) Introduction
- 2) Analyse
 - 1) Glossaire métier
 - 2) Les cas d'utilisations: overview puis détails
 - 3) Les scénarios textuel
- 3) Conception
 - 1) Architecture
 - 2) Classes (overview) et paquetages
 - 3) Diagrammes de séquences (raffinement des scénarios)
 - 4) Glossaire ingénierie des besoins
- 4) Maquette
- 5) Conclusion
- 6) Annexes



Autres sources de structures

- Faites des recherches sur le net !!

Comment l'obtenir ?

- Par itérations successives

Plan

Objectifs

Le document à produire

Introduction au diagramme de CUs

Introduction au diagramme de classes

Pourquoi un glossaire métier

Premiers pas dans l'identification des CU, classes et termes métiers

Problématique

- Comment **décrire les besoins d'une application** ?
 - **Quelles fonctionnalités** doit elle avoir ?
 - **Qui** peut déclencher quelle fonctionnalité ?
 - Quelles autres entités sont impliqués dans la fonctionnalités ?

Réponse UML

- Le Cas d'Utilisation (CU)



- L'acteur



Qu'est-ce qu'un cas d'utilisation ?

- Technique permettant d'identifier et de décrire les fonctionnalités d'un logiciel qui sont significatives pour ses utilisateurs (humains, matériels, logiciels)
 - Permet de décrire les interactions du logiciel avec son environnement
 - Expression du comportement du logiciel (actions et réactions) selon le point de vue des utilisateurs
 - Détermination des besoins fonctionnels des utilisateurs cibles
 - Introduit par Ivar Jacobson en 1986

Premier CU

- Que nous indique ce diagramme ?

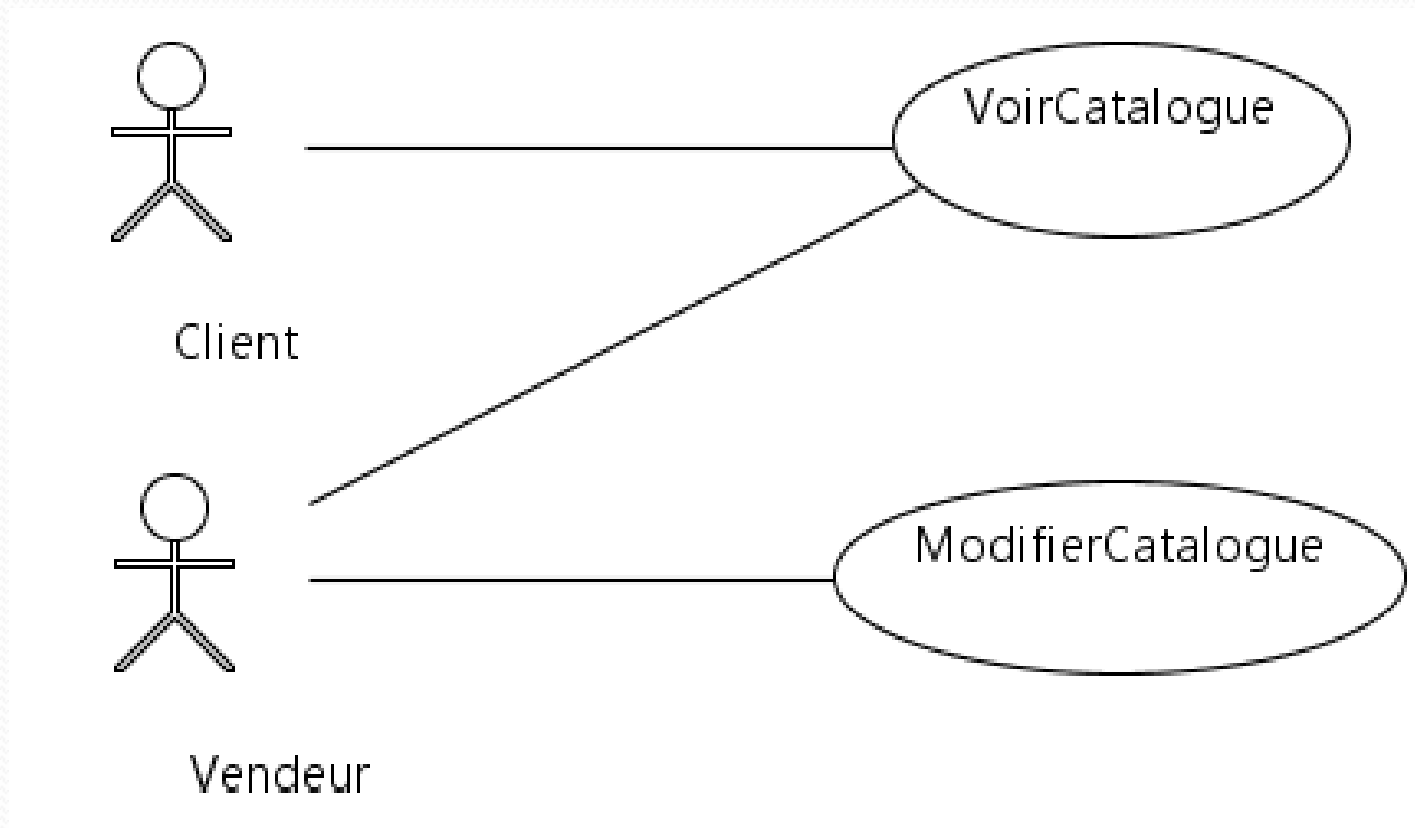
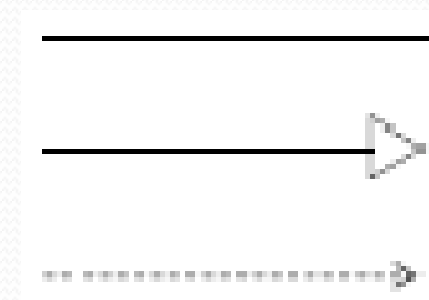
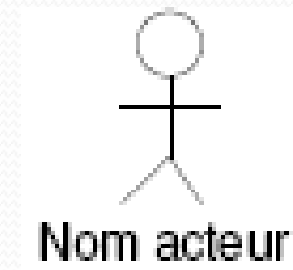


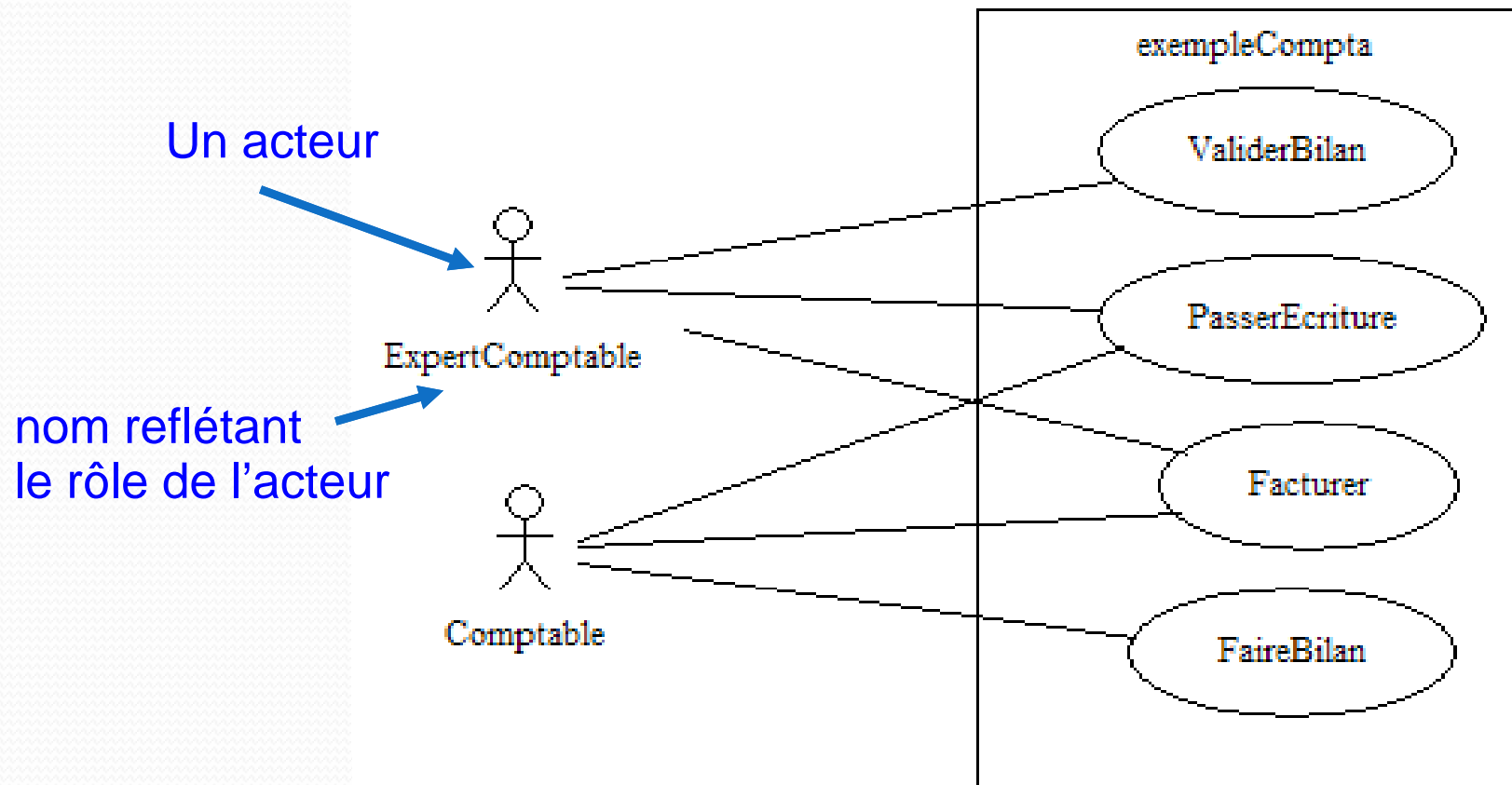
Diagramme de cas d'utilisation

Principaux concepts

- Acteurs
- Cas d'utilisation
- Relations
 - Entre acteurs et cas d'utilisation
 - Entre acteurs
 - Entre cas d'utilisation



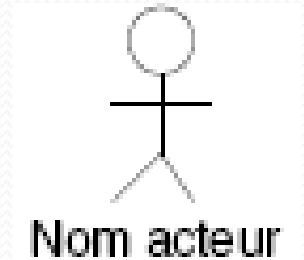
Exemple



- en UML un acteur est une classe stéréotypée <<Actor>>

Les acteurs

- Représentation **idéalisée** d'une personne, d'un logiciel, d'un processus, d'une organisation qui interagit (depuis l'extérieur) avec le logiciel
 - Rôle joué par cette personne, logiciel, etc.
 - Une même personne peut correspondre à plusieurs acteurs
 - Un même acteur peut être joué par plusieurs entités
L'acteur peut **consulter** ou **modifier** l'état du logiciel :
interaction avec le cas d'utilisation par envoi de message
- En réponse à l'action d'un acteur, le logiciel fournit un service : le cas d'utilisation qui correspond à la **fonctionnalité** désirée
- On trouve les acteurs en observant les **utilisateurs directs** du système, ceux qui sont **responsable de sa maintenance**, ainsi que les **autres systèmes qui interagissent** avec le système



Les acteurs

- Un acteur est **externe** à l'entité (il a une existence externe) avec laquelle il interagit.
 - Il s'agit de tout ce qui peut interagir avec le système et qui existe en dehors du système.
 - Ce sont souvent **des êtres humains**,
 - mais ce peut être aussi **des dispositifs mécaniques** ou autres.
- Les acteurs **principaux** ont besoin des services (cas d'utilisation) du système,
- Les acteurs **secondaires** permettent au système de fonctionner (ex: imprimante).
- On parle également d'acteurs **actifs** (déclencheurs) et d'acteur **passifs** (les autres)

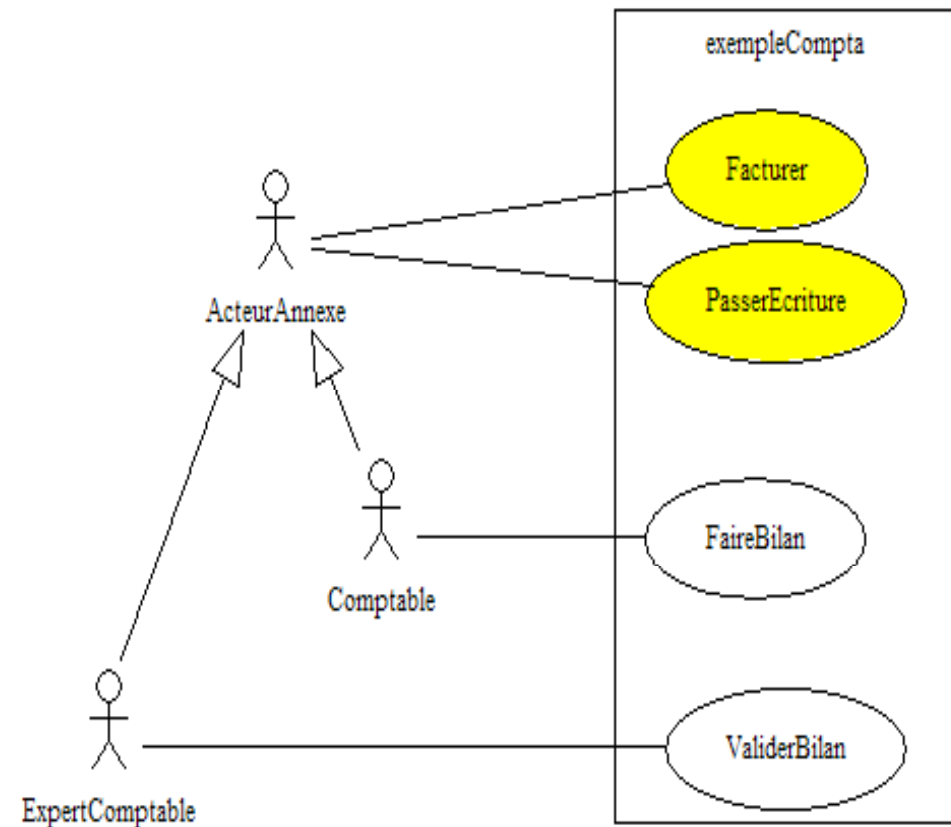
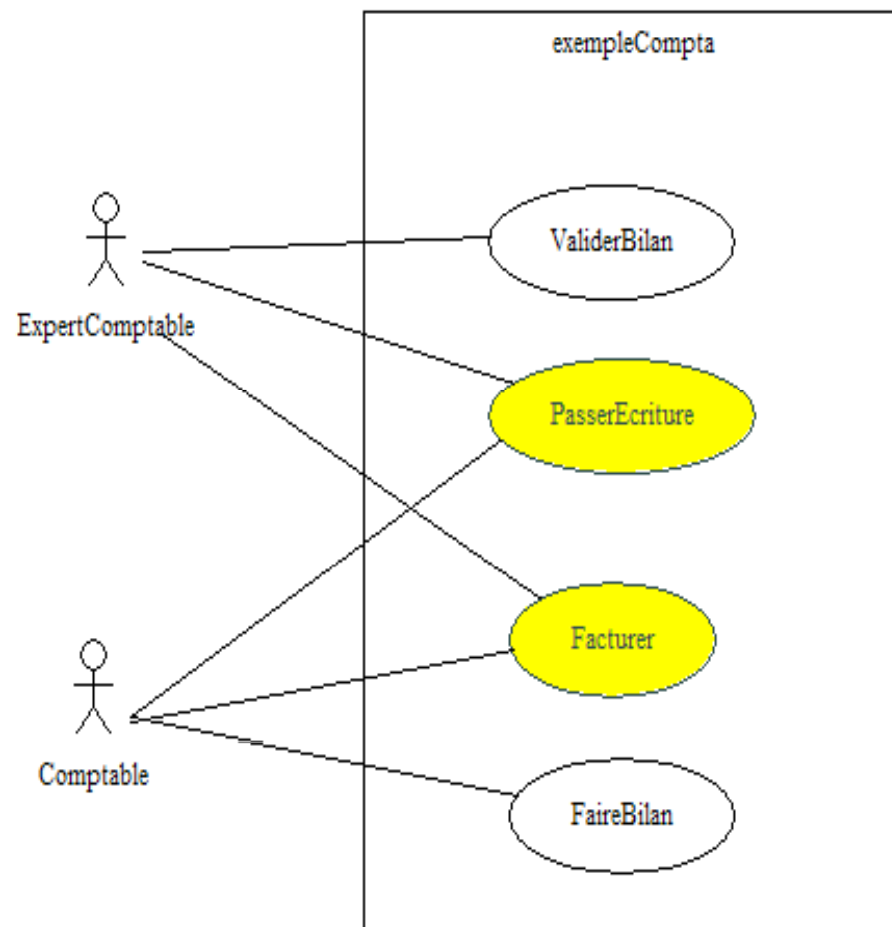
Relation entre acteurs



- Généralisation (héritage)
 - Toute personne empruntant des journaux peut aussi jouer le rôle d'emprunteur de livres.



Exemple - Généralisation d'acteurs (héritage)

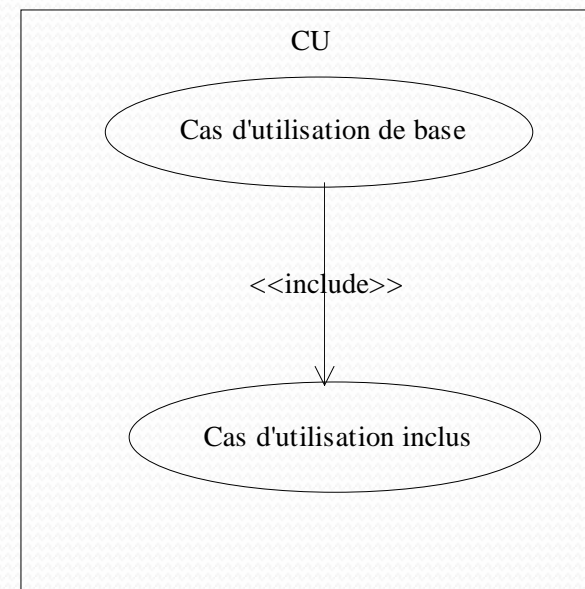


- Permet de factoriser et de simplifier

Relations entre cas d'utilisation

Relation <<include>>

- Certaines étapes dans un cas d'utilisation sont simples, d'autres sont plus complexes et font référence à d'autres cas d'utilisation, ces cas sont dit inclus.

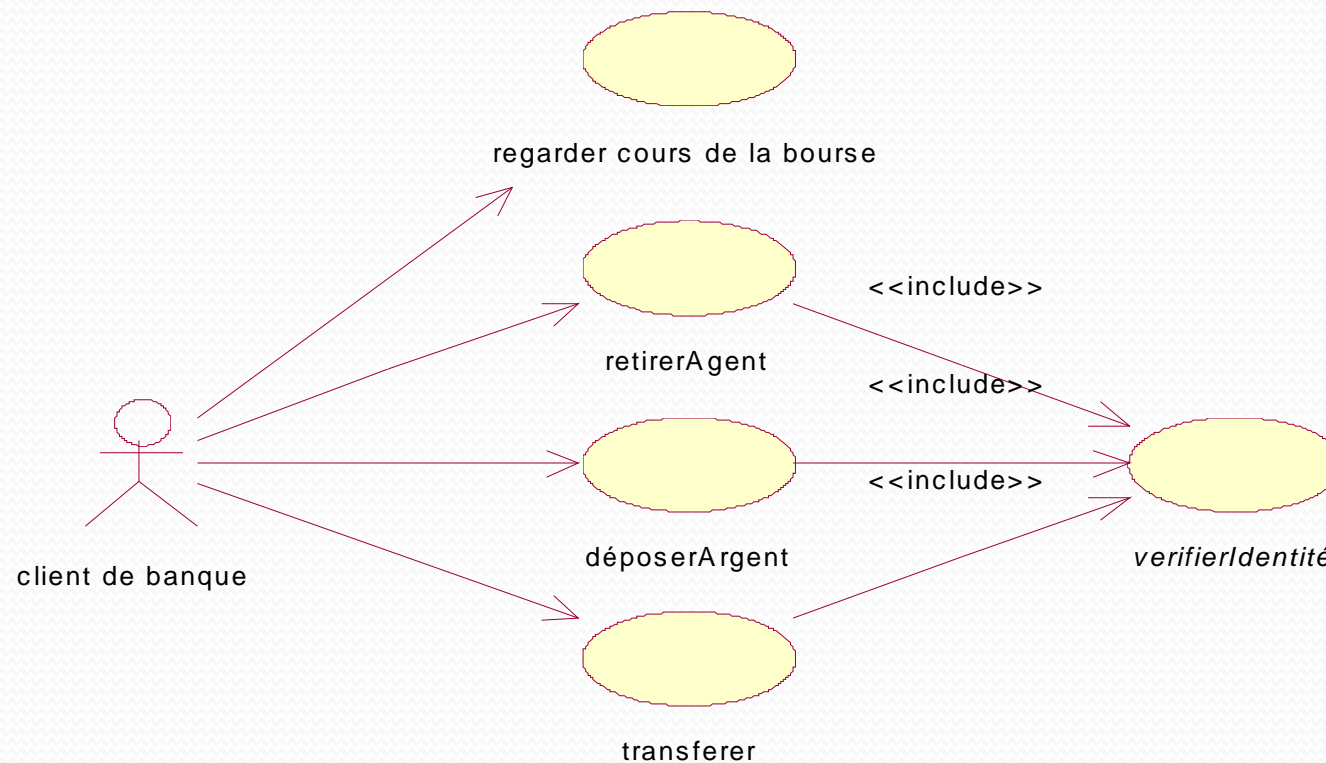


- Notion naturelle pour un développeur (~ appel)

Cas d'utilisation

Relation d'inclusion

- Rôle 1 : Mettre en commun des comportements communs à plusieurs CU
- Rôle 2 : Encapsuler un comportement complexe, pour avoir une vue plus globale.
 - Le cas inclus peut ne pas être déclenchable par un acteur.



Relations entre cas d'utilisation

La relation <<extend>>



Cas d'utilisation « Emprunter un exemplaire »

Scénario principal : ...

Scénarios secondaires (extensions) :

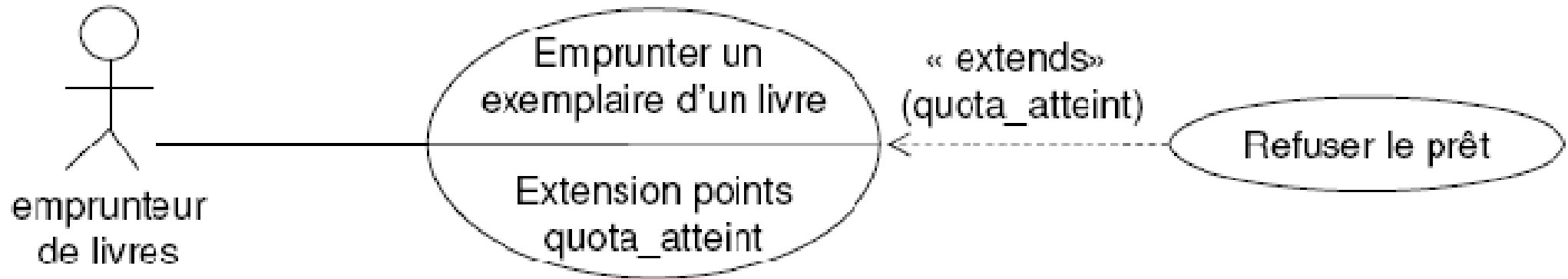
...

4a : Le nombre de livres empruntés supérieur à la limite :
il faut ... et effectuer la procédure **refuser le prêt**

...

Relations entre cas d'utilisation

La relation <<extend>>



Cas d'utilisation « Emprunter un exemplaire »

Scénario principal : ...

Scénarios secondaires (extensions) :

...

4a : Le nombre de livres empruntés supérieur à la limite :

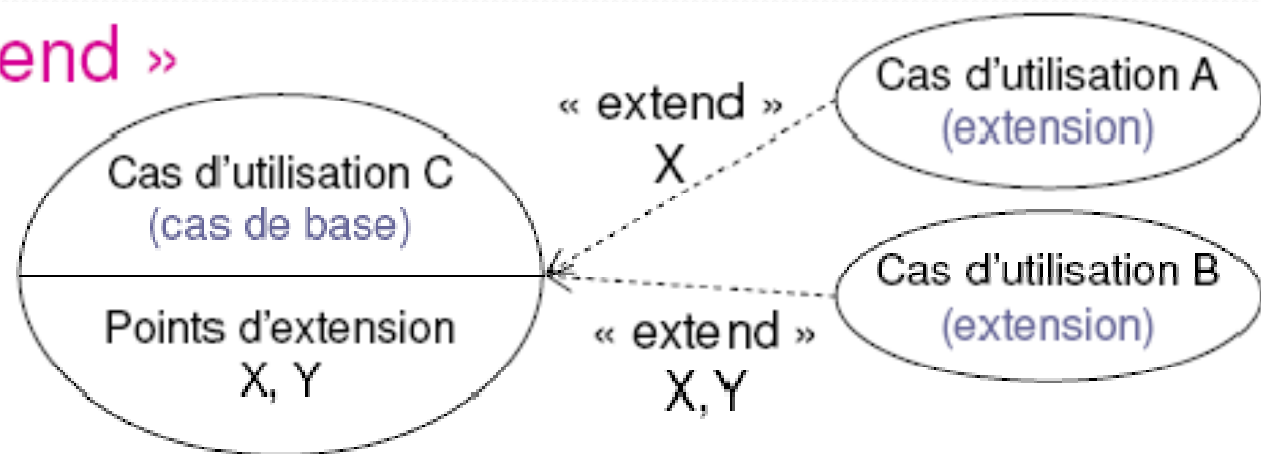
point d'extension « quota_atteint »

...

Relations entre cas d'utilisation

La relation <<extend>>

La relation « extend »



- Permet d'étendre, de façon structurée, le comportement d'un cas d'utilisation de base en utilisant un autre cas d'utilisation à un point d'extension spécifique.
- Les points d'extension sont définis à l'intérieur d'un cas d'utilisation de base, là où un comportement particulier est nécessaire
- Lorsqu'on atteint un point d'extension, on se branche sur le ou les cas d'utilisation traitant ce point d'extension.
- Si plusieurs cas d'utilisation extension sont activés par un même point d'extension, l'ordre de leur exécution est non déterministe.

Relations entre cas d'utilisation

La relation <<extend>>

- Le cas d'utilisation d'extension
 - Est appelé à un endroit explicite (point d'extension) dans le cas d'utilisation de base (cf. Scénarios secondaires)
 - **N'est pas nécessairement exécuté** même si le cas de base est instancié
 - Exécuté si un des points d'extension correspondants est atteint
- On utilise la relation « extend » pour mettre en évidence un scénario secondaire suffisamment complexe pour en faire un cas d'utilisation.

Comparaison <<include >> et <<extend>>

■ Relation « include »

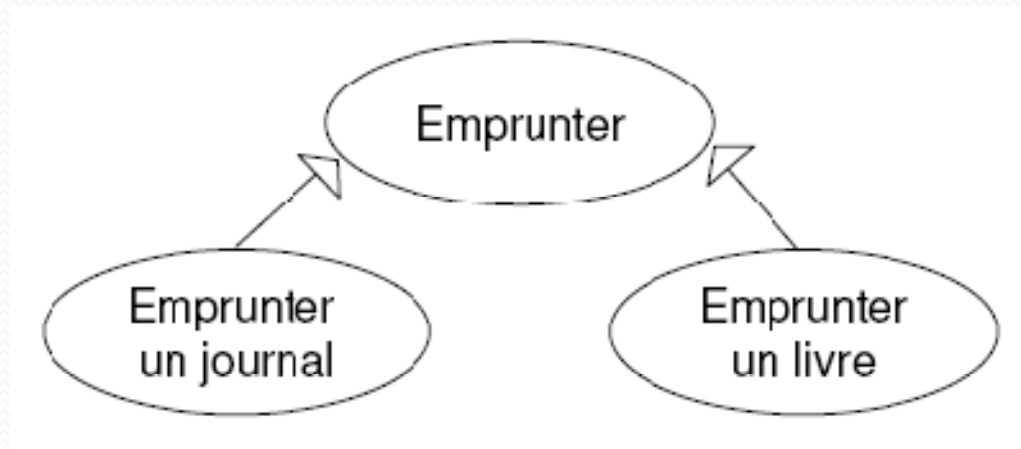
- Sens de la relation : le cas de base vers le cas inclus
- Exécution : **inconditionnelle**, le cas inclus est toujours exécuté par le cas de base (scénario principal)
- Dépendance : le cas de base n'est pas autonome sans le cas inclus

■ Relation « extend »

- Sens de la relation : le cas d'extension vers le cas de base
- Exécution : **conditionnelle**, le cas d'extension n'est pas nécessairement exécuté par le cas de base
- Dépendance : le cas de base est nécessairement autonome et bien formé même en l'absence du cas d'extension

Relations entre cas d'utilisation

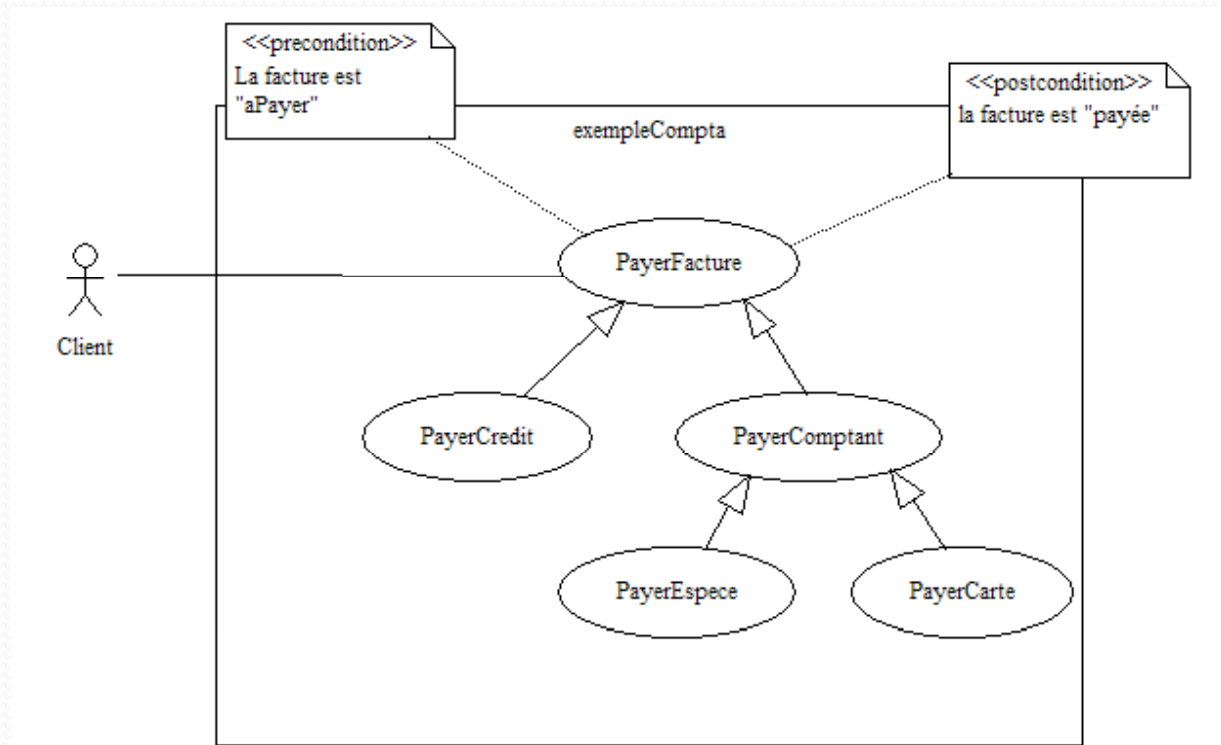
Généralisation (héritage)



- Permet à un sous cas d'utilisation de spécialiser le comportement d'un cas d'utilisation de base (qui peut être abstrait)

Relation de généralisation entre UC

- Un UC peut être spécialisé en un ou plusieurs cas d'utilisation. Les sous cas héritent des caractéristiques du sur cas d'utilisation (acteurs, conditions,...)



- Remarque : Ces relations ne correspondent pas au déroulement, ce sont bien des relations de structuration.

Cas d'utilisation et Scenario

- Un scénario représente **une** exécution possible d'un CU
- Un CU peut s'exécuter de **plusieurs** façons
 - ➔ ils a donc **plusieurs scénario**
- Scénario
 - Un scénario est une **séquence d'actions**, généralement déclenchée par un acteur.
 - {Pré - condition} scénario {post - condition}
 - Un scénario est une instance du cas d'utilisation

Plan

Objectifs

Le document a produire

Introduction au diagramme de CUs

Introduction au diagramme de classes

Pourquoi un glossaire métier

Premiers pas dans l'identification des CU, classes et termes métiers

Que nous indique ce diagramme ?

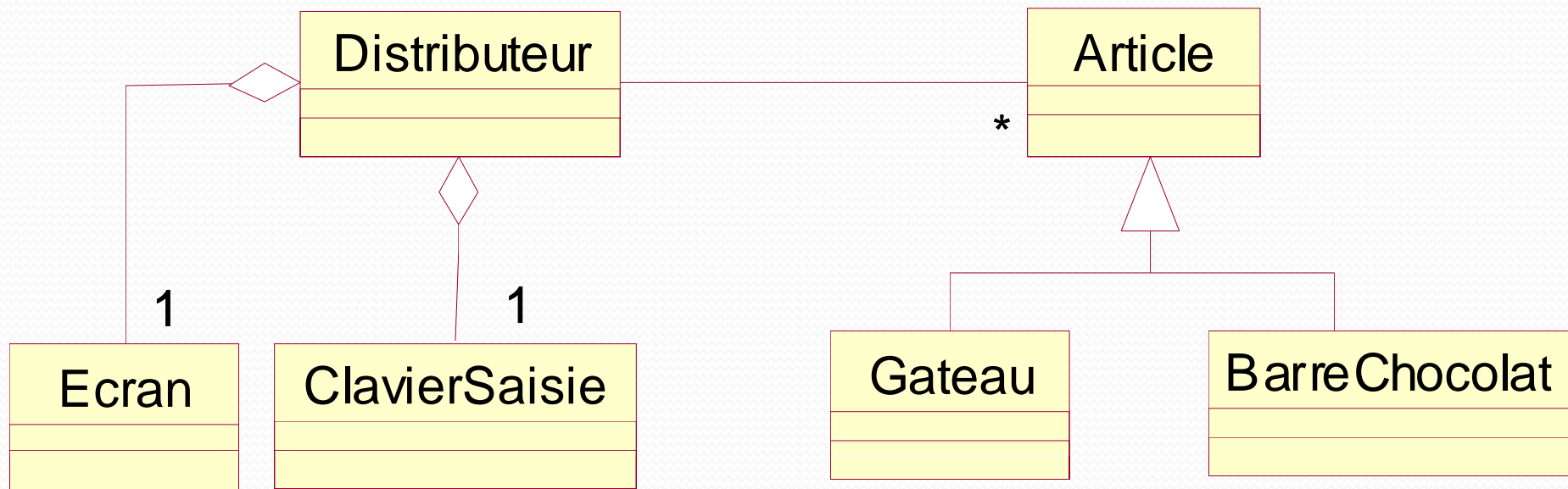
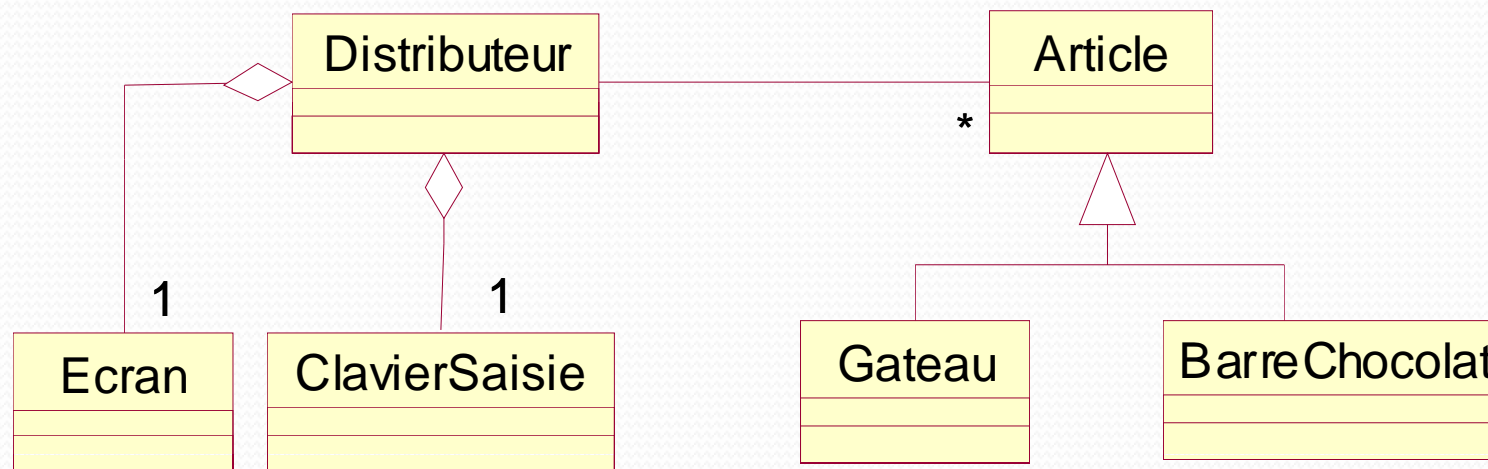
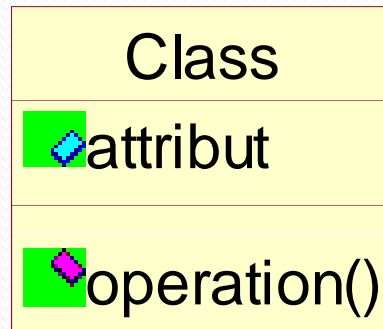


Diagramme de classe

- Permet de décrire la structure statique du logiciel avec des :
 - Classes (attributs, opérations)
 - Relations : associations, agrégations, compositions, héritage, dépendance
 - Paquetages, notes ...



Classe



- Remarque : UML distingue opérations et méthodes:
 - Une **méthode** est une **implémentation** d'une **opération**.
 - Plusieurs méthodes peuvent implémenter une même opération.

Attributs

<visibilité> <portée> <nomAttribut> : <type> [= <valeurInitiale>]

- **NomAttribut** : choisir un nom significatif
- **Type**
 - Types primitifs :
 - boolean, real, integer, string, date,
 - les types énumérés (définis par l'utilisateur)
 - et les datatypes (type dont chaque instance est définie par sa valeur)
 - Types définis par une classe ou une interface
- **ValeurInitiale** : valeur par défaut
- **Portée** : **instance** ou **classe** (\Leftrightarrow **static** java)
 - static indiqué par un **\$** ou en soulignant le nom.
- **Visibilité** : public, protected ou private

Opérations d'une classe





<visibilité> <portée> <nomOp> (<listePara>) : <typeRésultat>

- **NomOp** : choisir un nom significatif
- **ListeParamètres**

<direction> <nom_paramètre> : <type> = valeur_initiale

- <direction> := in | out | inout
- **TypeRésultat** et **type** : primitif ou classe
- **Portée** : opération d'instance ou de classe
(\Leftrightarrow static java)
 - static indiqué par un \$ ou en soulignant le nom.
- **Visibilité** : public, protected ou private

Quel est l'équivalent Java ?

Compteur	
	valeur : int
	<u>count : int</u>
	incrémenter()
	decrémenter()

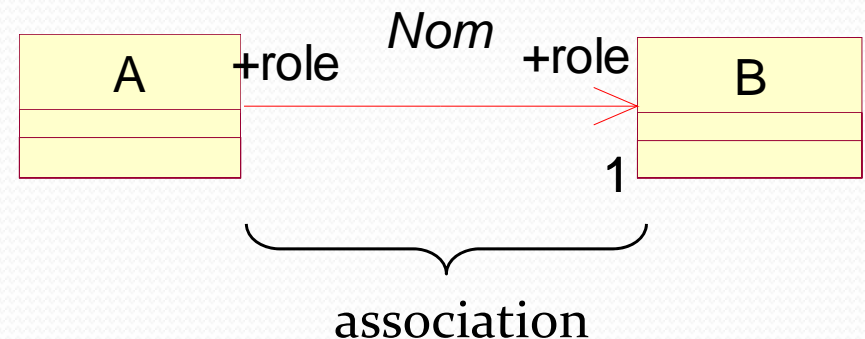
```
public class Compteur {  
  
    protected int valeur;  
    protected static int count;  
  
    public void incrémenter()  
        {...}  
    public void decrémenter()  
        {...}  
}
```

Relations entre classes

- L'association
- L'agrégation
- La généralisation
- La dépendance

Association

- **Lien** = Connexion (physique ou conceptuelle) entre deux instances de classes
- **Association** = Ensemble de liens ayant une sémantique commune
 - Réflexive, binaire ou n-aire (peu fréquent)
- Spécification d'une association
 - **Cardinalité**
 - **Nom** (optionnel)
 - **Rôles** (optionnel)



Association

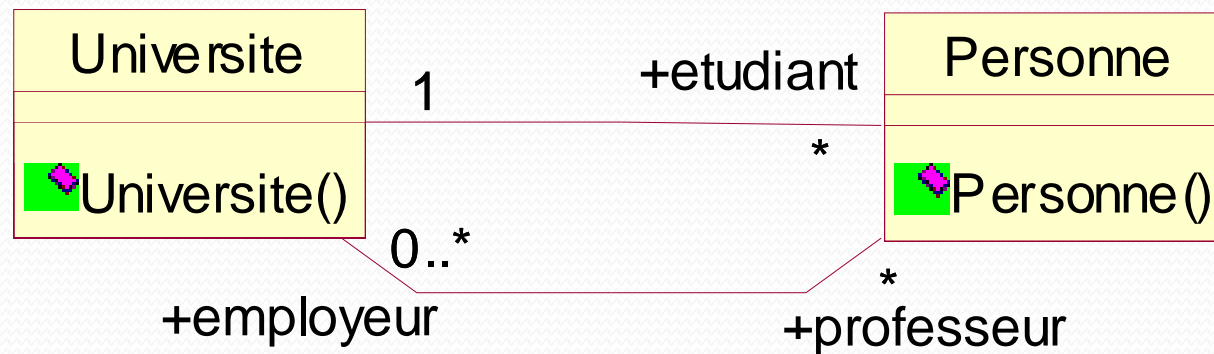
Cardinalité

- Multiplicité (ou cardinalité)
 - Précise le nombre d'instances participantes
 - *min..max*
 - Où *min* et *max* sont des entiers
 - *max* peut être non borné: *
 - Exemples
 - 1 Une et une seule
 - 5 Exactement 5 instances (entiers naturel)
 - 0..1 Zéro ou une
 - 4..12 De 4 à 12 (entiers naturel)
 - * implique plusieurs instances
 - 0..* De zéro à plusieurs
 - 1..* D'une à plusieurs

Association

Nommage des rôles

- Le rôle décrit une extrémité d'une association
 - L'université à 0 ou plusieurs étudiants



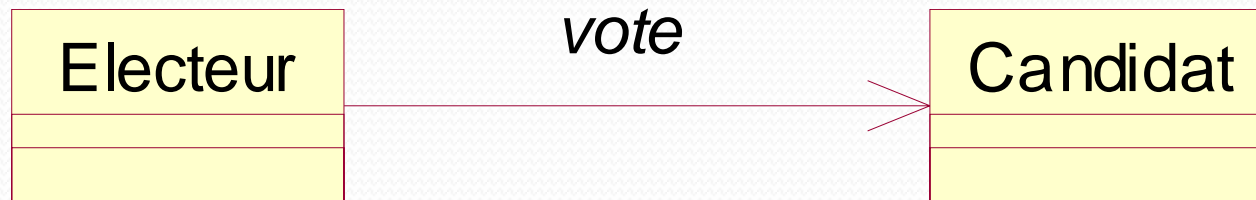
```
public class Universite
{
    public Collection<Personne> etudiant;
    public Collection<Personne> professeur;
    ...
}
```

```
public class Personne
{
    public Collection<Universite> employeur;
    ...
}
```

Association

Navigabilité

- Par défaut, une association est bidirectionnelle
- Une flèche sur l'association restreint la navigabilité à un seul sens : celui de la flèche



- Un électeur connaît un candidat
- Un candidat ne connaît pas un électeur : on ne peut pas naviguer du candidat vers l'électeur.

Association

Agrégation et Composition

- Agrégation

- Association qui exprime qu'une instance possède une autre classe



- Composition

- Agrégation forte entre deux instance
- relation composite – composant
- Exprime qu'une classe est composé d'une autre classe
 - quand le composite est détruit, le composant l'est aussi



A quoi ça sert ?

- Classes
 - Représentes les objets du domaine
- Relations
 - Représentent les relations entre les objets
 - Structurelles
 - Et/ou pour échanger des messages

Plan

Objectifs

Le document à produire

Introduction au diagramme de CUs

Introduction au diagramme de classes

Pourquoi un glossaire métier

Premiers pas dans l'identification des CU, classes et termes métiers

Glossaire métier

Problématique

- Comment être sûr de parler de la même chose ?
- De bien se comprendre ?



Glossaire métier

- Définit tout les termes métier rencontrés
 - Permet de s'assurer que l'on a bien compris les termes.
 - Permet de découvrir les synonymes
 - Permet de se comprendre !
- Représentation graphique (non UML !):

Terme	Définition	auteur
enchère		CD
vendeurs	Personne mettant an vente des articles dans une enchère	CD
article	Objet mis en vente	CD
Acheteur	Personne consultant le catalogue et pouvant faire des offres sur les articles	CD

Comment l'obtenir ?

- A partir du recueil des besoins
 - Tous les termes métiers
- Possibilité de demander à la MOA de clarifier les définitions
- Des synonymes peuvent apparaître
 - Les identifier

Plan

Objectifs

Le document à produire

Introduction au diagramme de CUs

Introduction au diagramme de classes

Pourquoi un glossaire métier

**Premiers pas dans l'identification des CU, classes
et termes métiers**

Identifier les Cus, les classes, les termes métier

- Part des documents à votre disposition
- Verbe identifiant une action
 - → CU
- Sujet du verbe identifiant une action
 - → acteur
- Cible d'une action
 - Acteur passif
- Objets métiers
 - → classes
- Description d'une règle, d'une méthode, d'un processus
 - → règle métier

Identifier les Cus, les classes, les termes métier

- Recense tout les termes
- Supprime les doublons par la suite
 - La définition des termes facilite la découverte des doublons

Conclusion

Qu'avons-nous vu aujourd'hui ?

- Contenu et structure du document présentant une solution
- Introduction aux Cus
- Rappel sur les classes
- Le glossaire métier

● Pour la prochaine séance

- Réfléchir aux deux sujets
 - Gestion d'une bibliothèque universitaire
 - WebPizza – Vente de Pizza par internet. Commande par internet, visualisation des commandes par les cuistos, les livreurs, le manager, ...
- Savoir utiliser Google Doc pour partager un document à plusieurs
 - *docs.google.com*
 - Besoin de quelques portables avec wifi