



# MODÈLE CONCEPTUEL DES TRAITEMENTS

Analyse 3

2020-2021

## PLAN

### MCD

**Modèle conceptuel des données**

Diagramme de classes  
(rappels) Documentation

### MCT

**Modèle conceptuel des traitements**

Diagramme de Use Cases (UC)  
Documentation

Conceptuel

### UC Specification

Documentation de UC  
Interface utilisateur  
Diagramme d'activité (rappel)

### PTFE

**Plan de tests fonctionnels élémentaires**  
Documentation

Fonctionnel

### MTD-MTT

### UC Realization

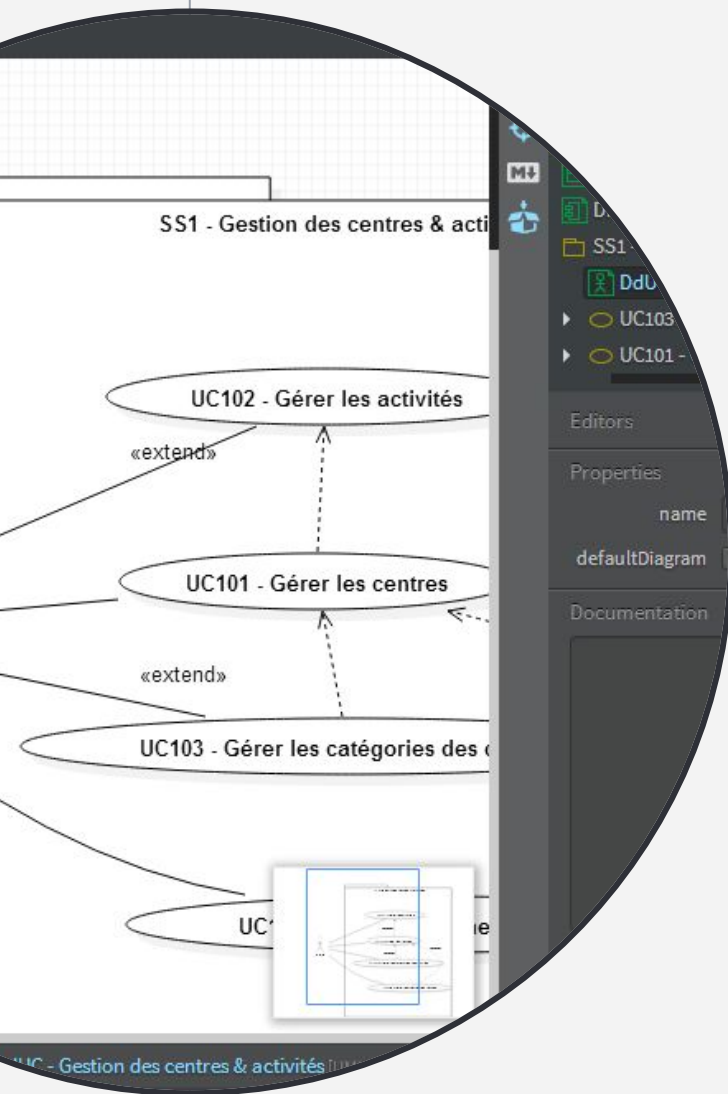
Diagramme de séquence  
Diagramme de classes techniques

### Design Pattern

Technique

## Méthodes

## OBJECTIFS



**Modéliser les traitements** du domaine qui permettront de construire les classes et leurs méthodes.

## STRUCTURE DU MCT

Descriptions  
textuelles

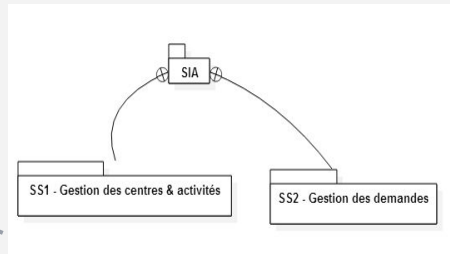
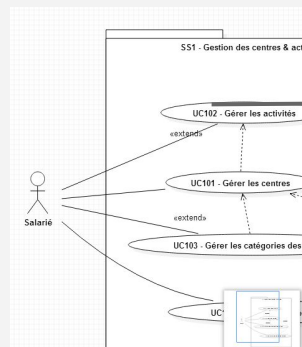


Diagramme de  
sous-systèmes



Diagrammes de use cases



## SUITE DU COURS

- Diagramme de Cas d'Utilisations (Use Case (UC))

- Mise en pratique des diagrammes de Use Cases dans StarUML

- Rédaction de la documentation dans StarUML

# MCT

UML : Diagramme de cas d'utilisation

- Modèle de cas d'utilisation

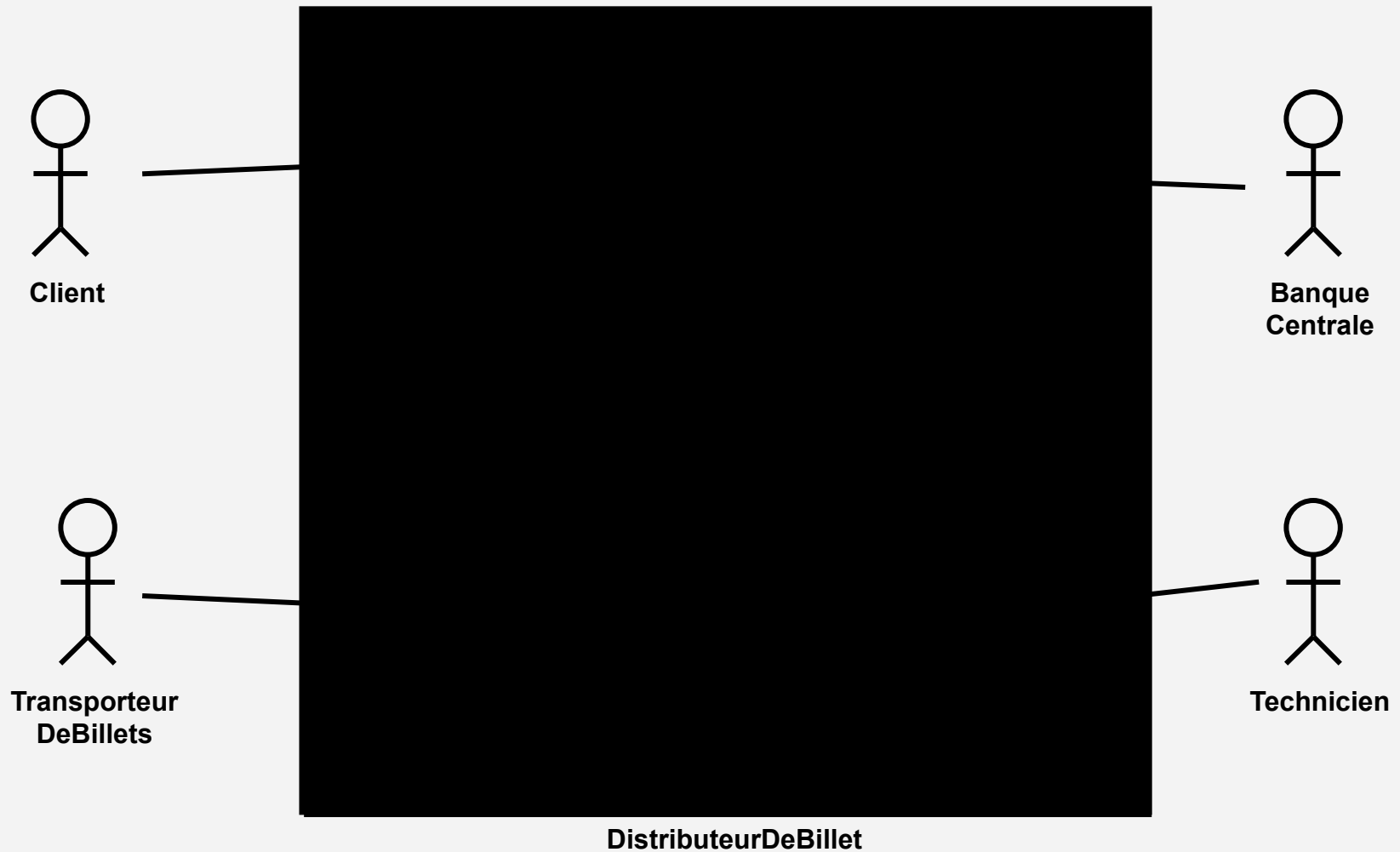
# Exemple



Client

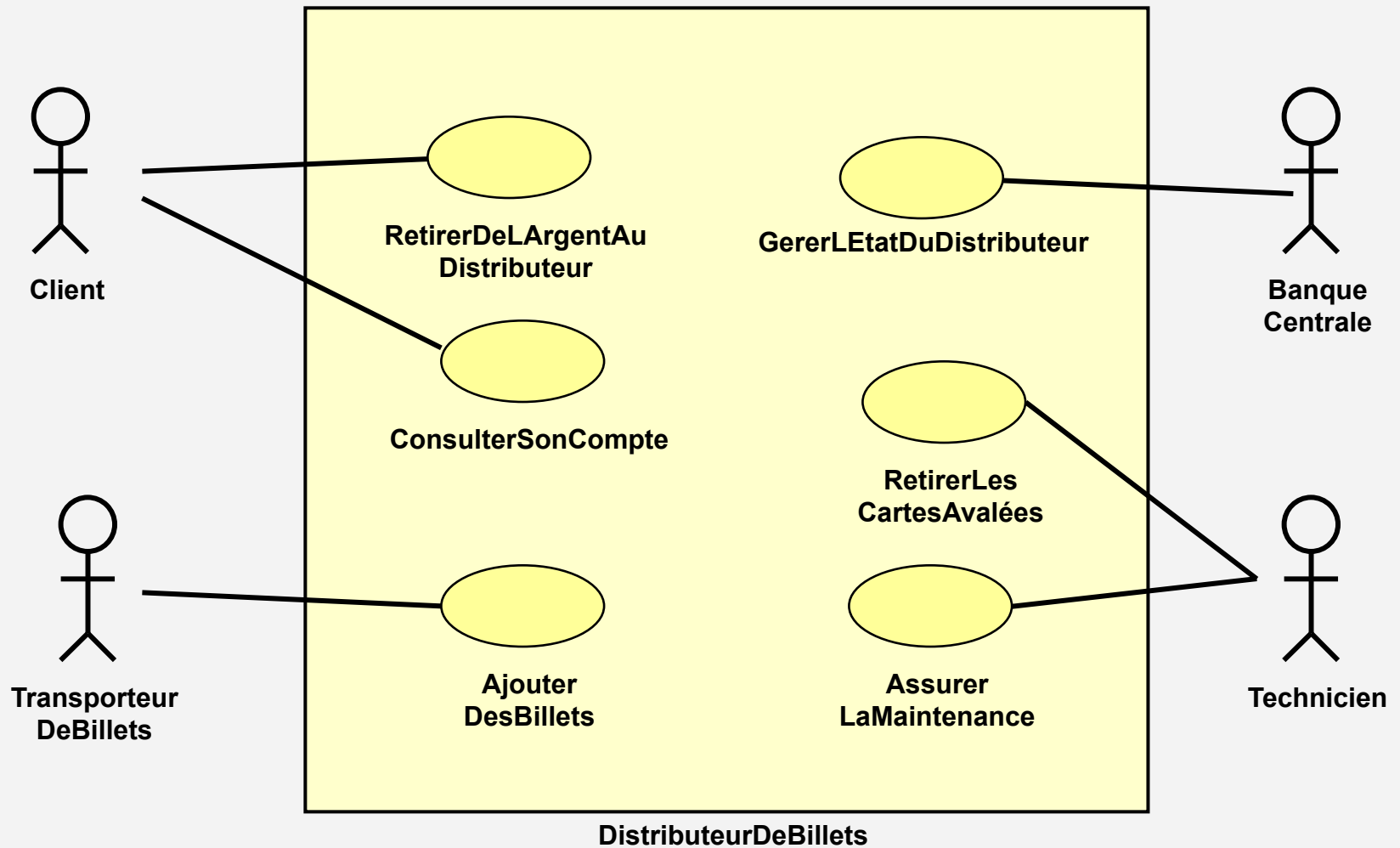


# Technique dite de “boite noire”





# Diagramme de cas d'utilisation



# Modèle des cas d'utilisation

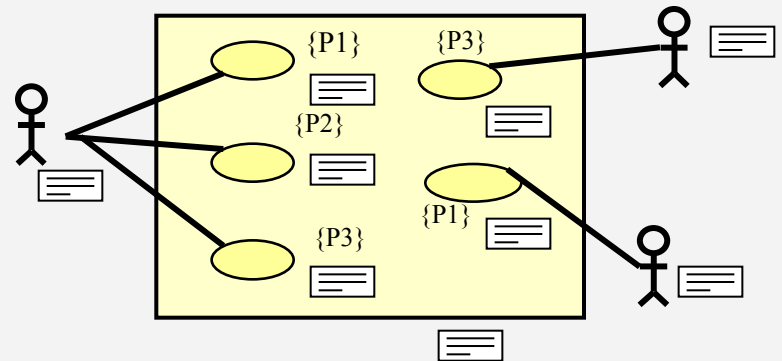
## Objectifs

Description des fonctionnalités du système

- Définir les limites précises du système

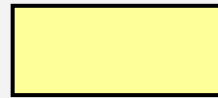
### **Structurer :**

- les besoins
- le projet



# Diagramme de Cas D'utilisation

## Éléments de bases



SystèmeN



CasDUtilisationN

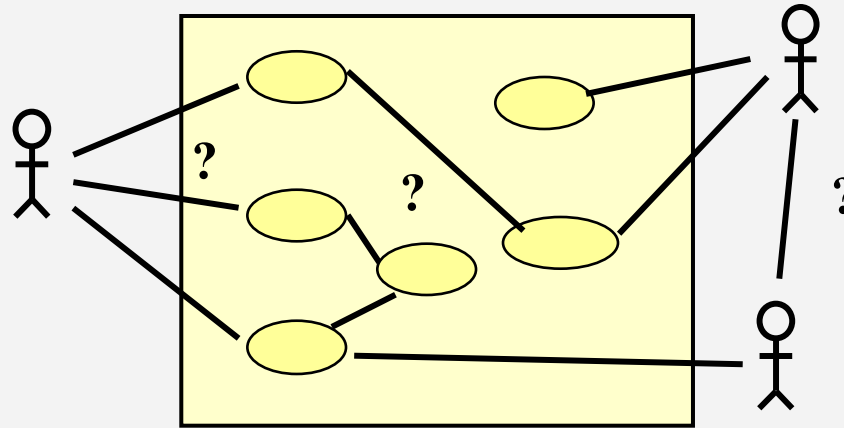


ActeurN

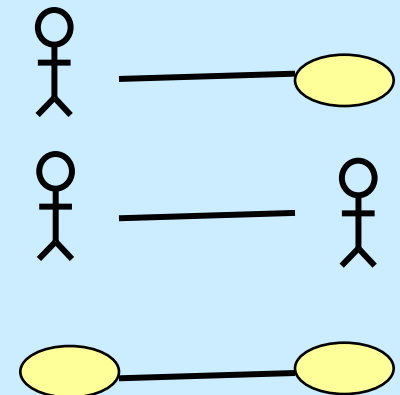
# Diagramme de Cas D'utilisation

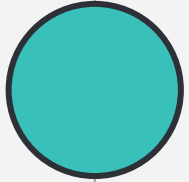
## Relations entre éléments de base

## Relations entre éléments de base



- Relations acteurs  $\leftrightarrow$  cas d'utilisation ?
- Relations acteurs  $\leftrightarrow$  acteurs ?
- Relations cas d'utilisation  $\leftrightarrow$  cas d'utilisation ?





Acteur



# Différents types d'acteurs

Liste pour ne pas oublier d'acteurs :

- Utilisateurs principaux  
ex: client, guichetier
- Utilisateurs secondaires  
ex: contrôleur, directeur, ingénieur système, administrateur...
- Périphériques externes  
ex: un capteur, une horloge externe, ...
- Systèmes externes  
ex: système bancaires



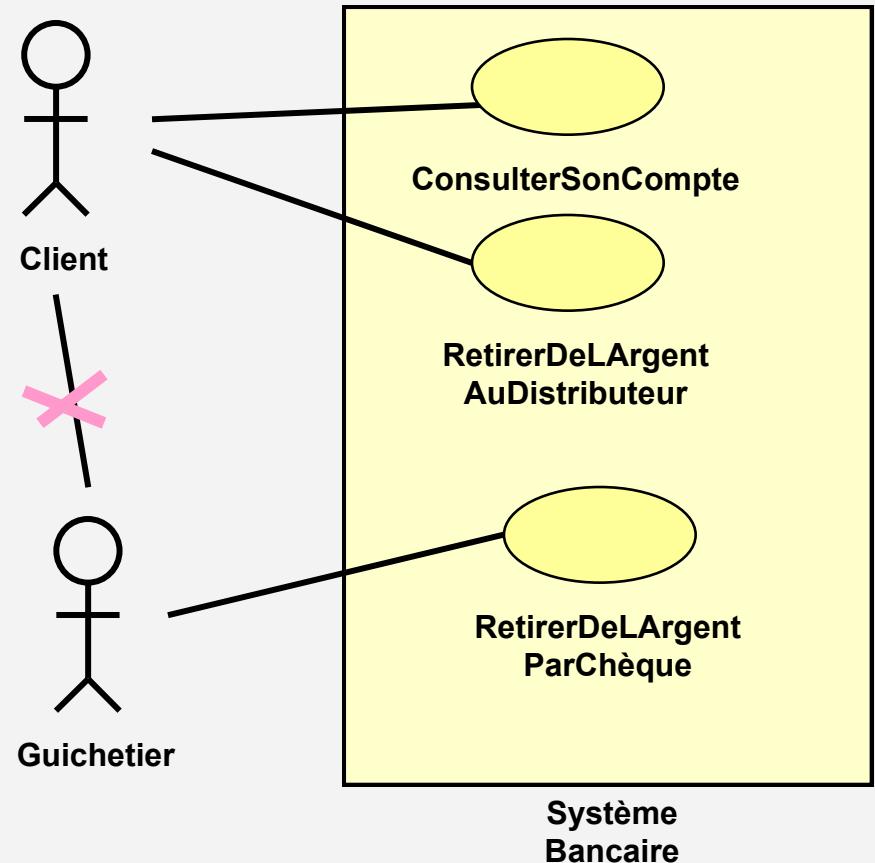
Client

# Acteurs vs. utilisateurs

- Une même personne peut jouer plusieurs rôles  
ex: Maurice est directeur mais peut jouer le rôle de guichetier
- Plusieurs personnes peuvent jouer un même rôle  
ex: Paul et Pierre sont deux clients
- Un rôle par rapport au système plutôt que position dans l'organisation  
ex: PorteurDeCarte plutôt qu'Enseignant
- Un acteur n'est **pas forcément** un être **humain**  
ex: un distributeur de billet peut être vu comme un acteur

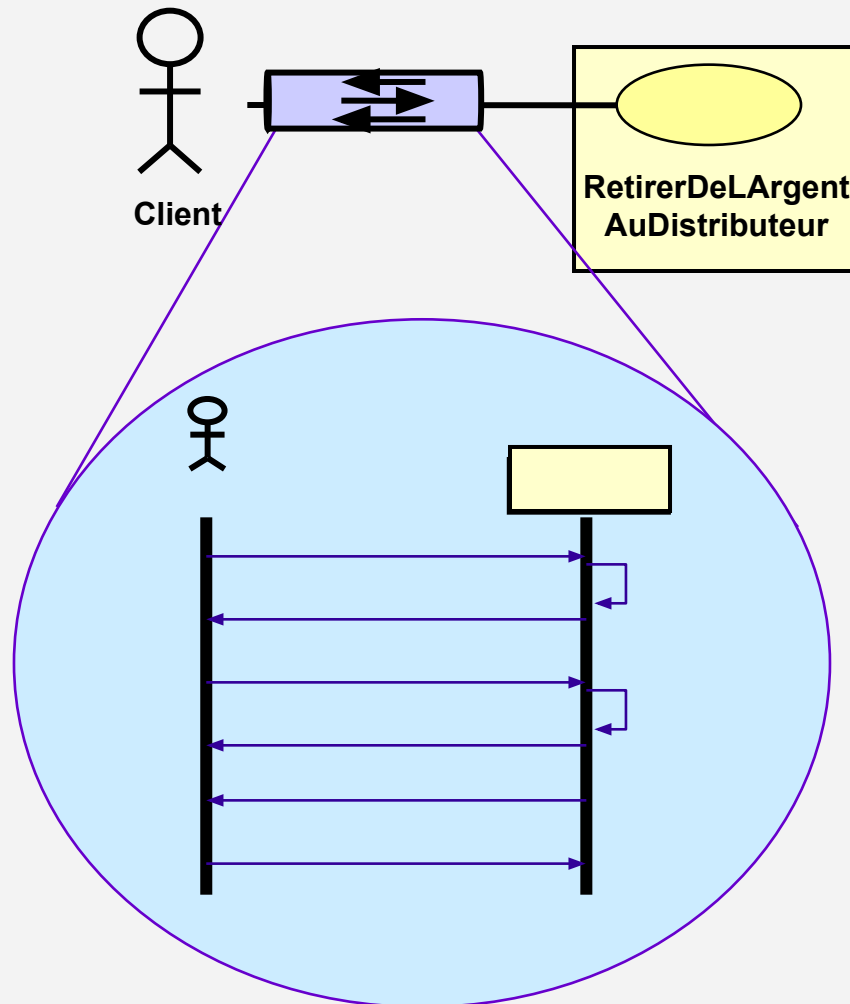
# Relation de communication acteur-acteur

- **Communications externes non modélisée**
- UML se concentre sur la description du système et de ses interactions avec l'extérieur





# Description de l'interaction

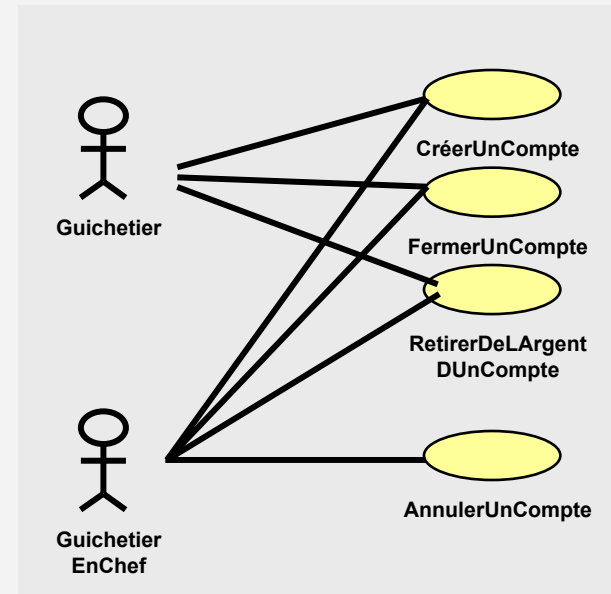
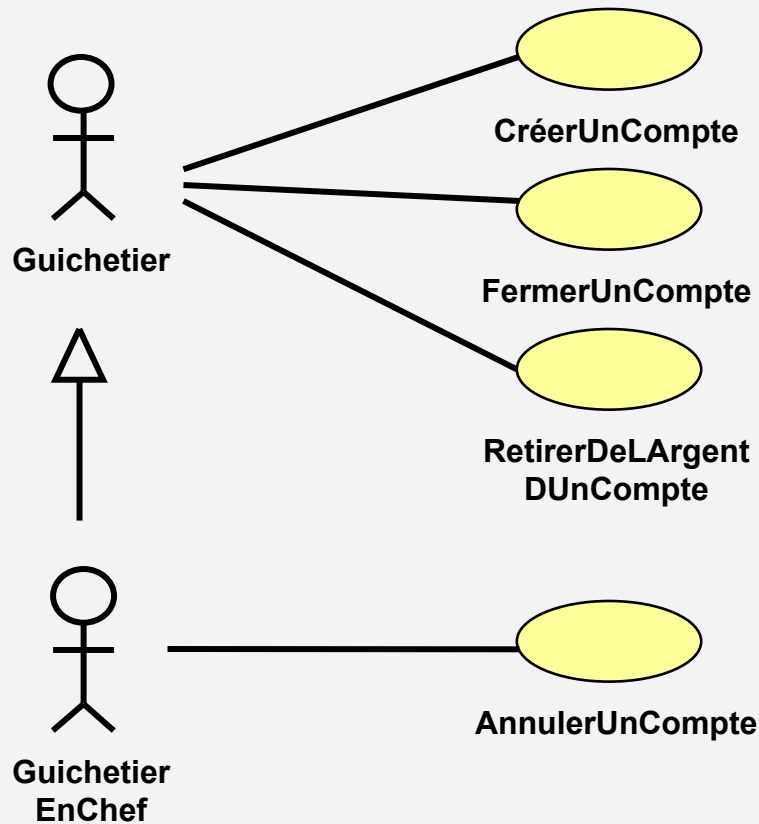


Diagrammes de  
séquences  
"système"

Plus tard dans le cours ...

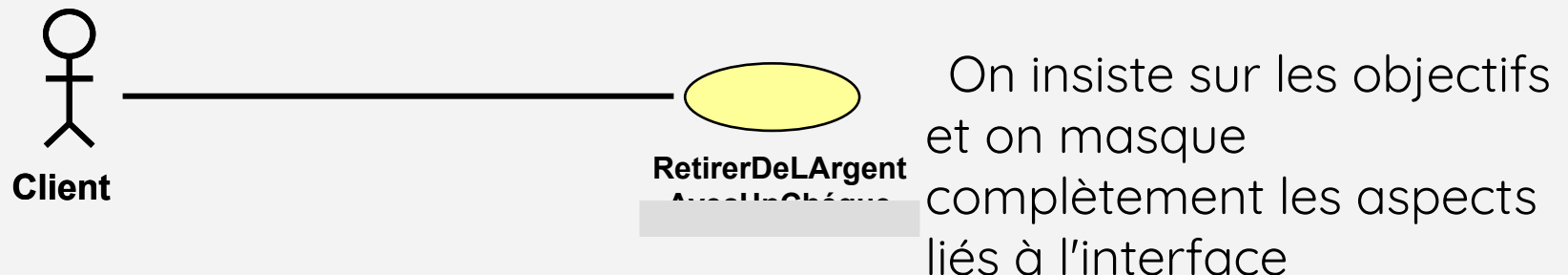
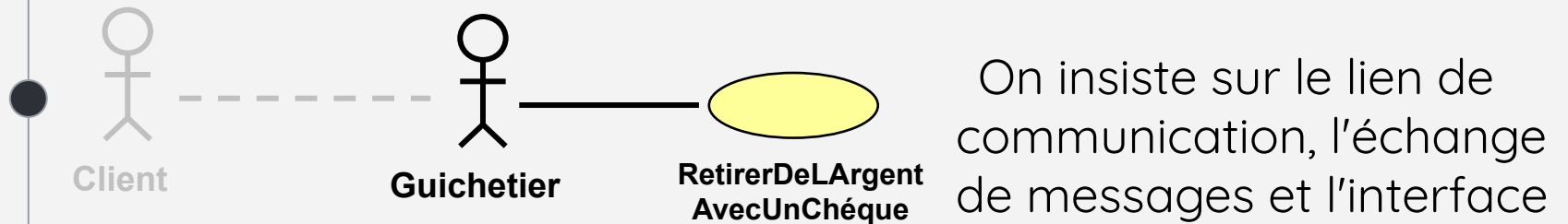
# Relation acteur - acteur : généralisation

La seule relation entre acteurs est la relation de généralisation

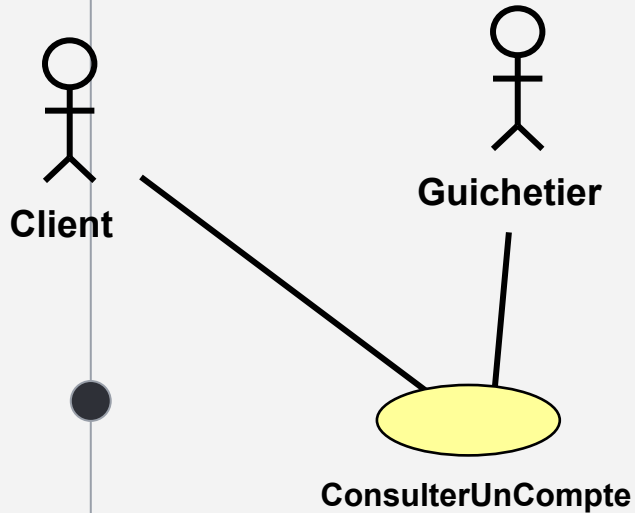


# Problème des intermédiaires

- Représentation des intermédiaires entre le système et l'intéressé ?
- **Différents points de vue**

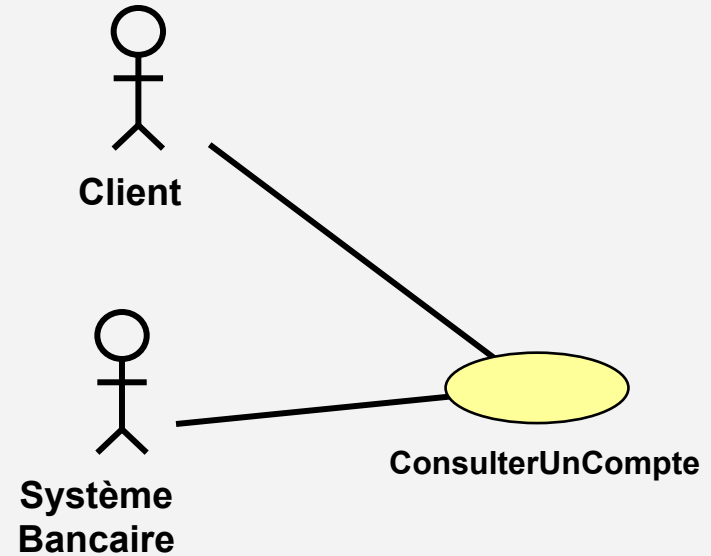


# Une notation mais deux interprétations



## (1) CAS D'UTILISATION "PARTAGE"

Deux acteurs peuvent réaliser le cas d'utilisation mais pour répondre à des objectifs qui leur sont propres



## (2) CAS D'UTILISATION "COLLABORATIF"

Deux acteurs collaborent à la réalisation d'un objectif. Le système interagit avec les deux acteurs.

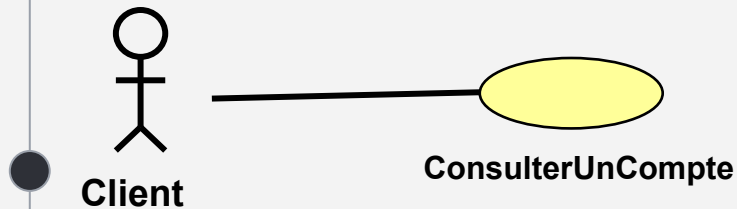
# Différents styles dans la pratique

- **STYLE "primaire":**  
Ne représenter que les acteurs primaires dans les diagrammes
- **STYLE "décoré":**  
Utiliser une décoration particulière (e.g. auxiliaire ou initiator)
- **STYLE "gauche/droite":**  
Positionner les acteurs primaires à gauche, secondaires à droite
- **STYLE "fléché":**  
Utiliser une flèche pour indiquer l'acteur primaire (à éviter)

# Différents styles

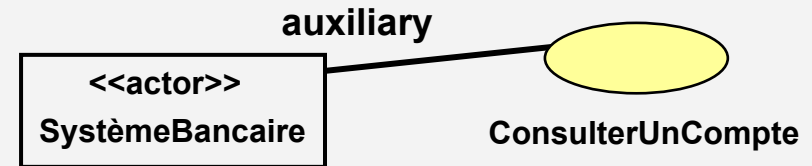
## STYLE "primaire":

Ne représenter que les acteurs primaires dans les diagrammes



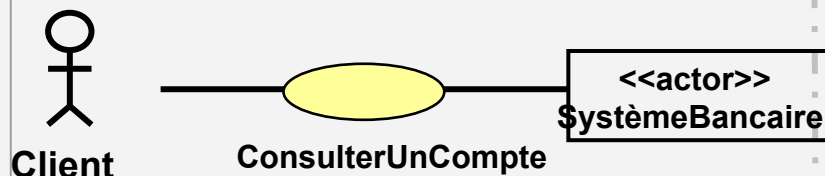
## STYLE "décoré":

Utiliser une décoration particulière (e.g. auxiliaire ou initiator)



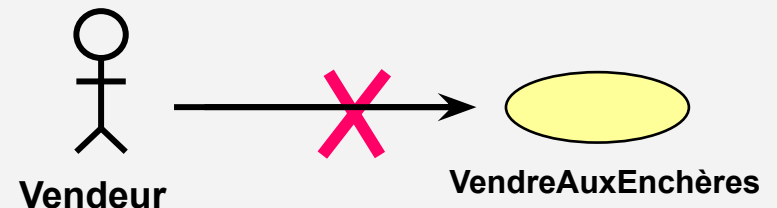
## STYLE "gauche/droite":

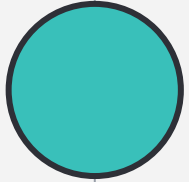
Positionner les acteurs primaires à gauche, secondaires à droite



## STYLE "fléché":

Utiliser une flèche pour indiquer l'acteur primaire (à éviter)





## Cas d'utilisation



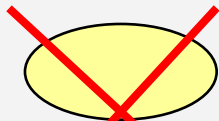
CasDUtilisation

# Cas d'utilisation (CU)

- Cas d'utilisation (CU)
  - une manière d'utiliser le système
  - une suite d'interactions entre un acteur et le système
- Correspond à une fonction du système **visible par l'acteur**
- Permet à un acteur d'atteindre **un but**
- Doit être **utile en soi**
- Regroupe un **ensemble de scénarii** correspondant à un même **but**



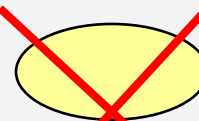
Entrer



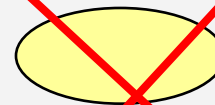
EnregistrerEntrée



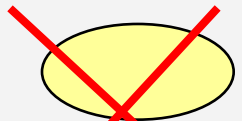
RetirerDeLArgentAu  
Distributeur



Sidentifier



EntrerPendant  
LesHeuresDOuverture

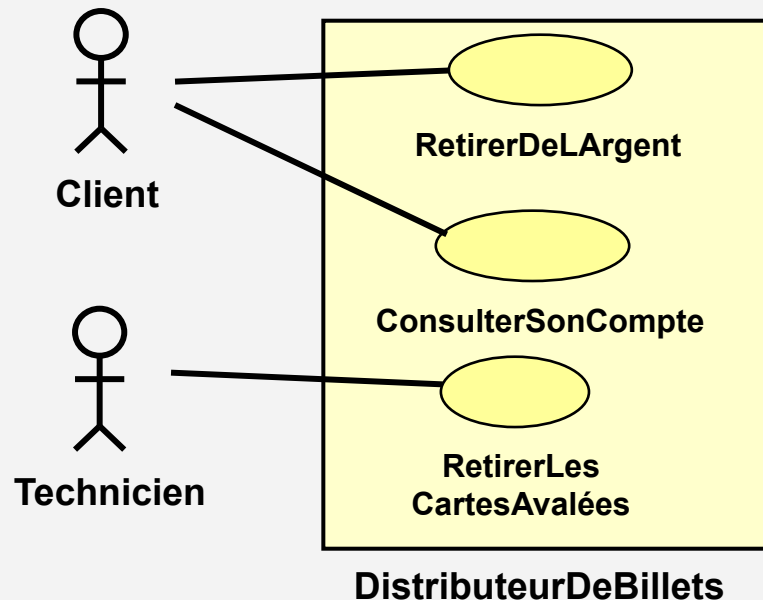


TaperSonCode



# Problème des cas d'utilisation orientés-solution

- Décrire les buts et les besoins des acteurs, les interactions mais **pas l'interface (concrète)**
- Le POURQUOI, POUR QUI, pas le COMMENT

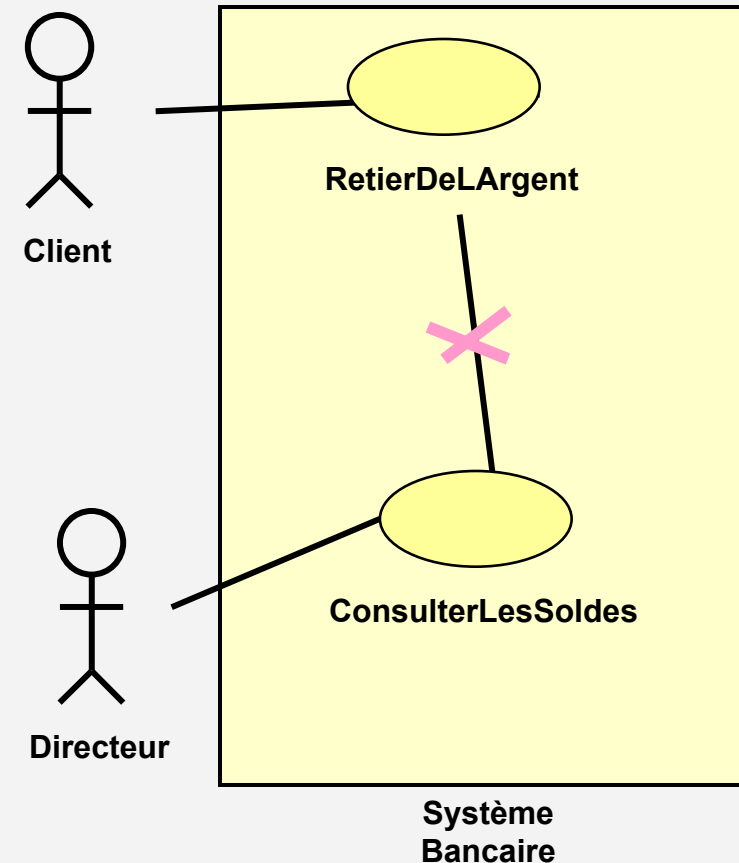


Se concentrer  
sur l'essentiel

=> cas d'utilisation "essentiels"

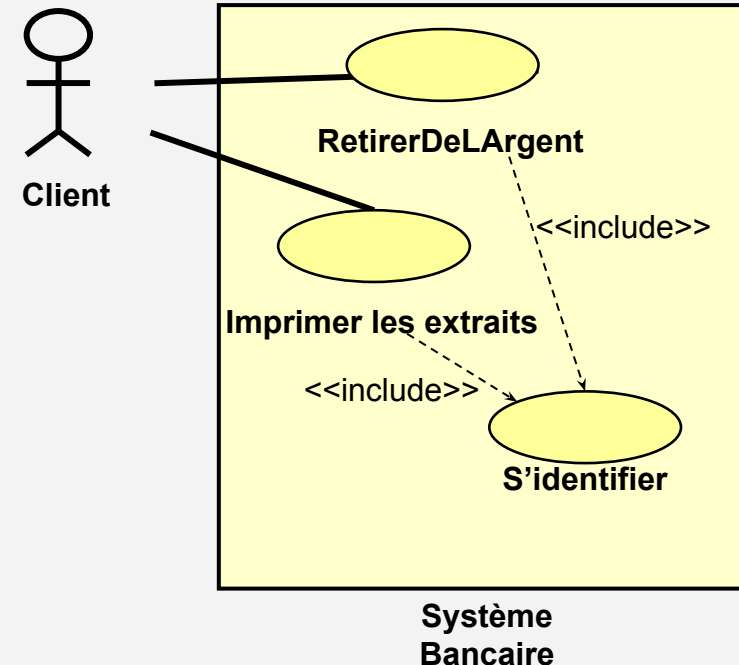
# Relation cas d'utilisation - cas d'utilisation

- **Communications internes non modélisées.**
- Interactions système ↔ extérieur
- Formalisme bien trop pauvre pour décrire l'intérieur du système. Utiliser les autres modèles UML pour cela.
- Autres relations possibles entre CU (concepts avancés)



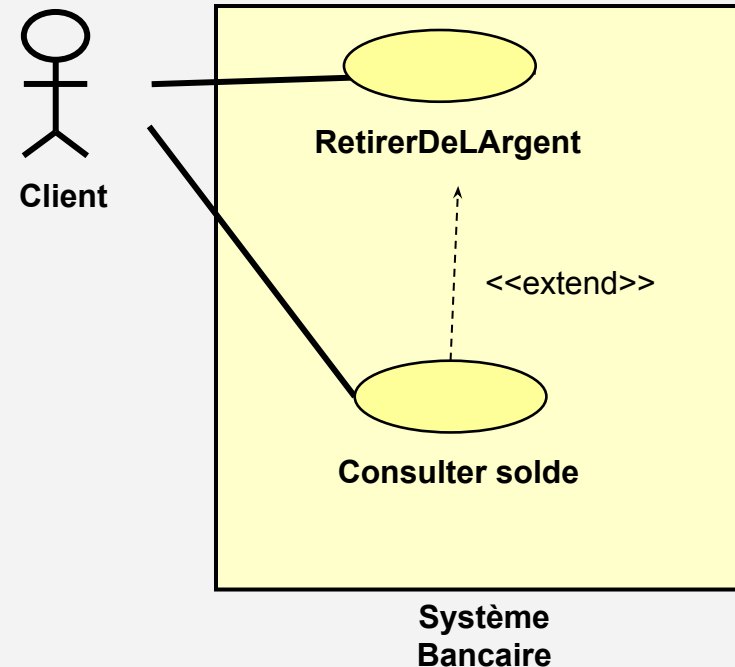
# Relation “include”

- Si plusieurs CU ont des sous-ensembles communs d’actions
- Permet la factorisation des CU
- Une instance de RetirerDeLArgent va engendrer une instance de S’identifier
- RetirerDeLArgent dépend de S’identifier
- S’identifier n’existe pas seul
- La relation d’inclusion suppose une **obligation** d’exécution des interactions



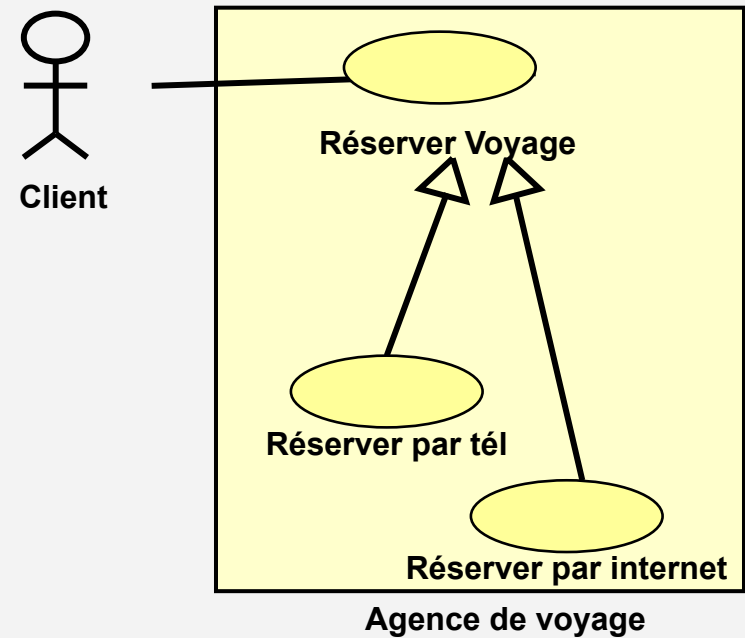
# Relation “extend”

- Permet d'étendre les fonctionnalités d'un CU
- Ce CU peut fonctionner seul mais peut également être complété par un autre, sous certaines conditions et à certains moments précis
- La relation d'extension suppose une **option** d'exécution des interactions (pas d'obligation)



# Relation de généralisation

- Un CU peut hériter d'un autre CU
- À voir comme un *polymorphisme* de cas
- Le but est le même mais les interactions pour y arriver ne sont pas les mêmes



# Niveaux d'abstractions

Clouds  
Level



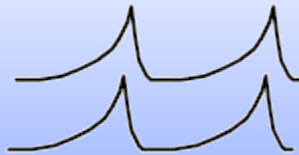
- Trop haut

Kite  
Level



- Niveau résumé : décrit un regroupement correspondant à un objectif plus global

Sea  
Level



- **Niveau normal** : décrit un but de l'acteur qu'il peut atteindre via une interaction avec système

Fish  
Level



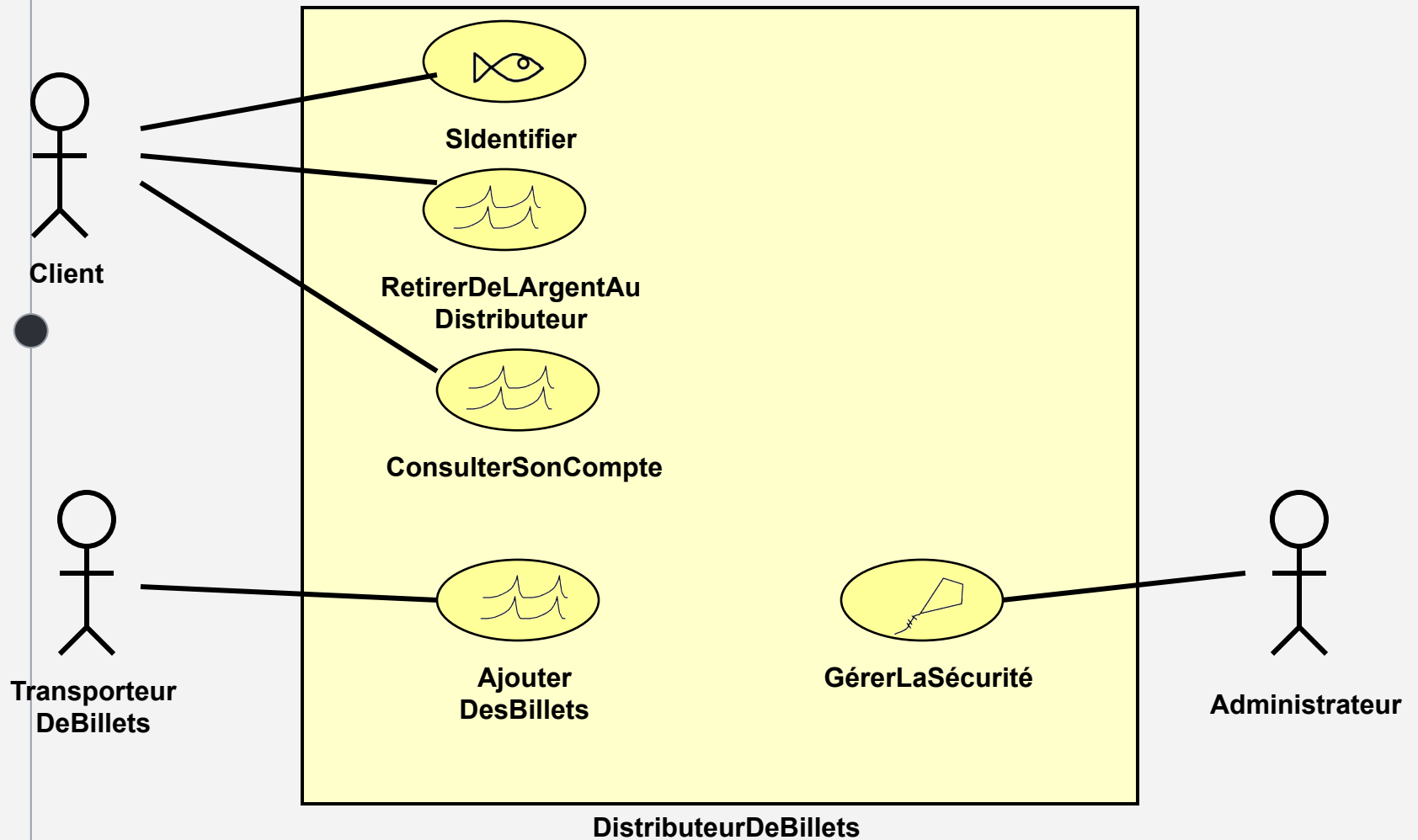
- Niveau détaillé : décrit une interaction avec le système, pas un but en soi

Clam  
Level



- Trop bas

# Exemple de marquage



## ● Unicités des cas d'utilisation

### ● Doit respecter les 4 unicités :

**Unicité d'objectif** Le UC doit fournir un résultat appréciable à l'acteur avec lequel il interagit.

**Unicité de responsabilité** Pour un UC interactif, un seul acteur direct

#### **Unicité de temps**

**Unicité d'exécution** Un enchaînement d'actions qui se suivent sans interruption

**Unicité de périodicité** Exécution à la même fréquence

#### **Unicité de mode**

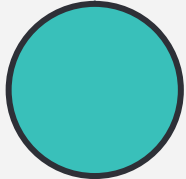
manuel, interactif ou automatisé  
unitaire ou par lot



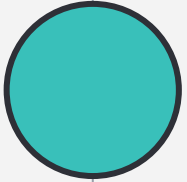
## Mode de cas d'utilisation

	Unitaire	Par lot
Manuel	<b>Remplir un formulaire</b>	<b>Vérifier manuellement tous les formulaires reçus</b>
Interactif	<b>Enregistrer un passager sur un vol</b>	<b>Encoder les 1000 livres achetés par la bibliothèque</b>
Automatisé	<b>Traitement d'un signal d'alarme</b>	<b>Envoyer au siège central les opérations bancaires d'un distributeur de billets toutes les heures</b>

*Hors SIA*

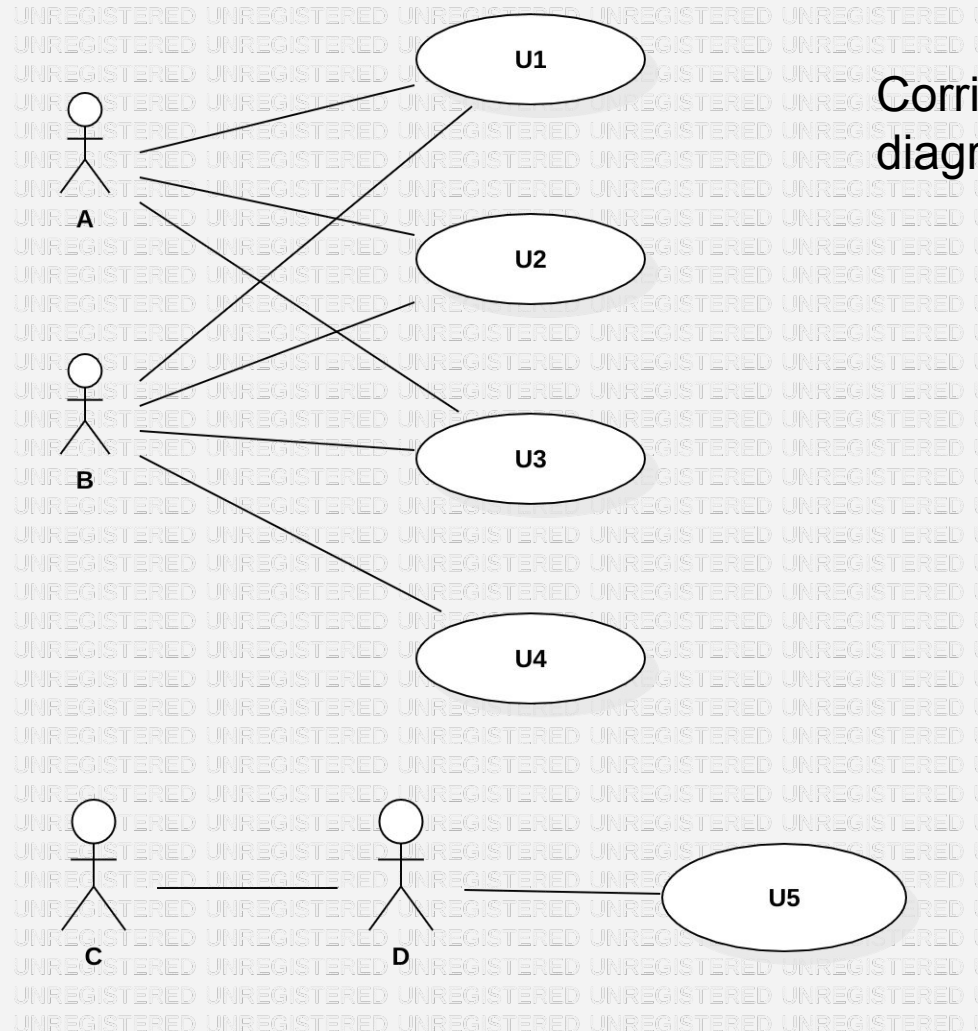


## Démo: Diagramme UC



# Exercices

# Exercice1

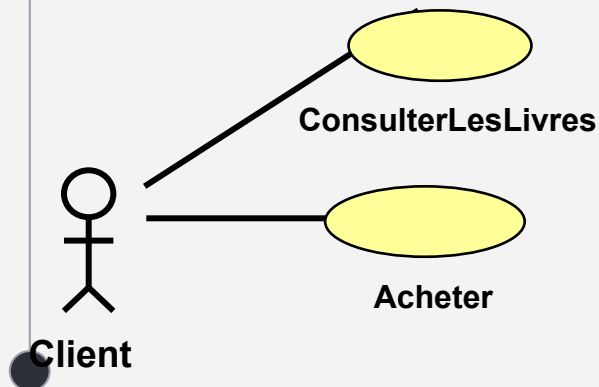


Corrigez ce diagramme!

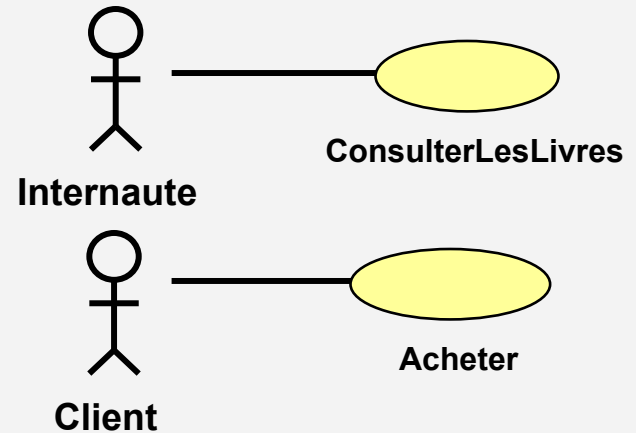
## Exercice 2

Décrivez les SIA de ces diagrammes en mettant en avant leurs différences.

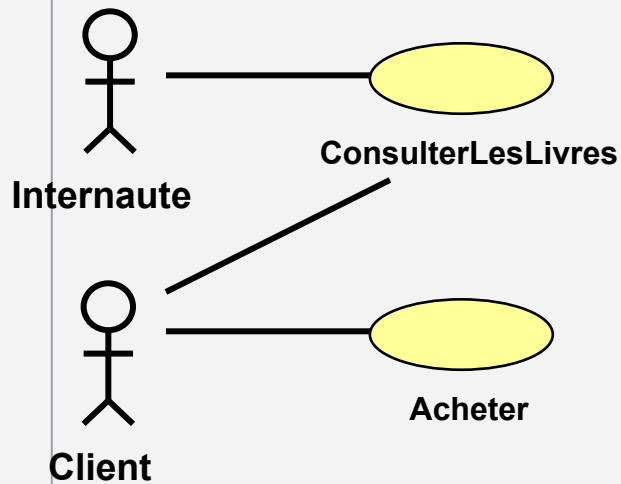
A



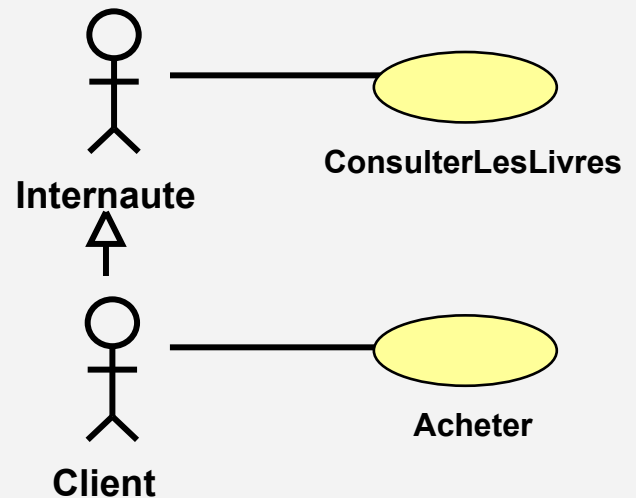
B



C

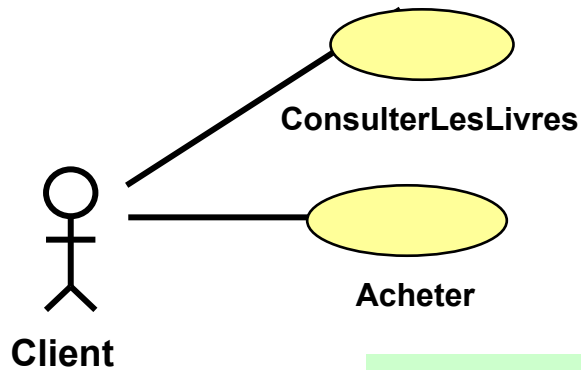


D

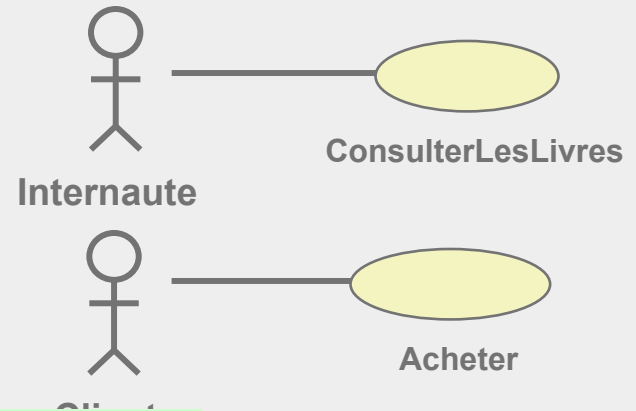


# Problèmes des cas d'utilisation partagés

A

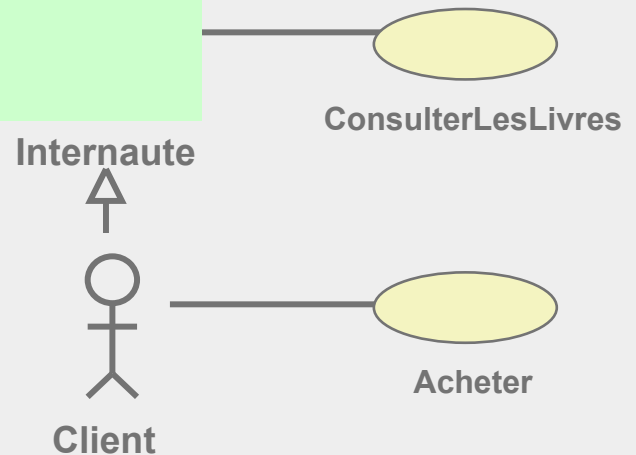
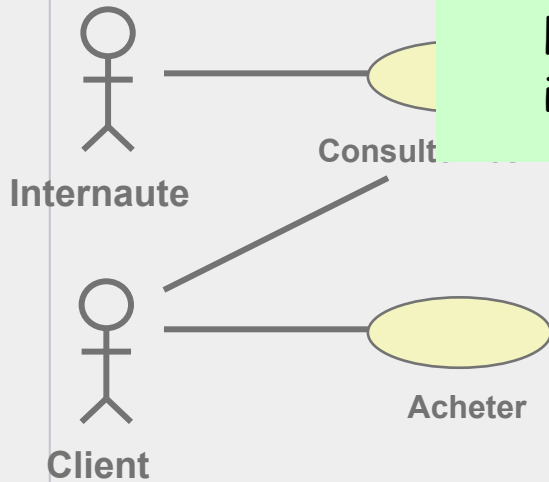


B



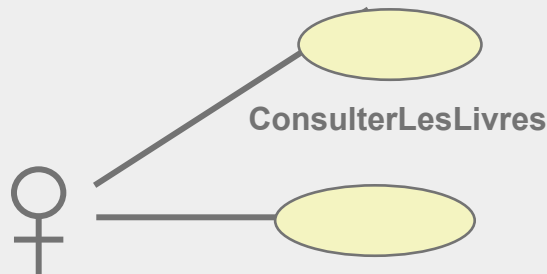
SYSTEME DE VENTE EN LIGNE  
Un client peut consulter  
la liste des livres et  
il peut en acheter

C



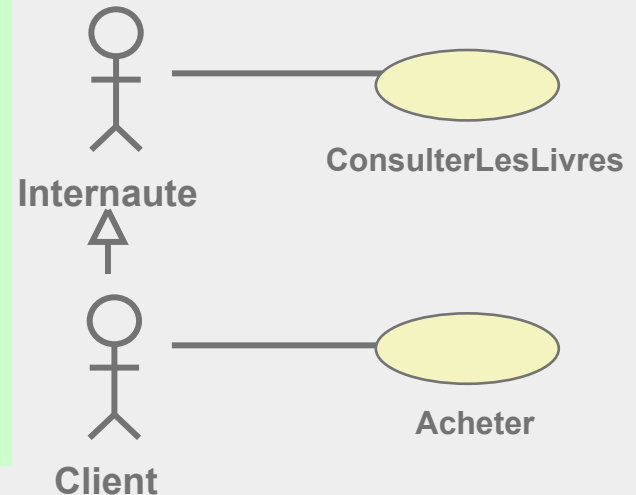
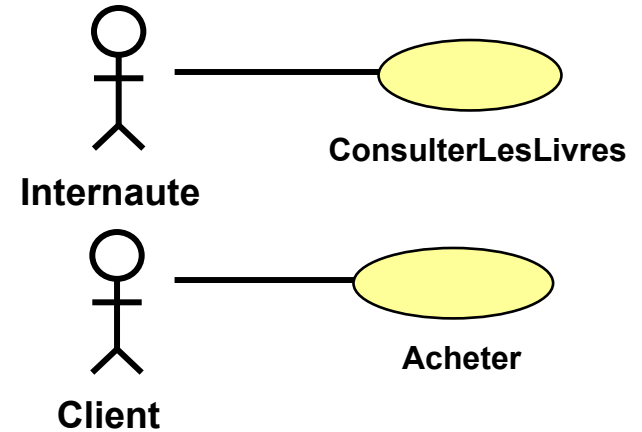
# Problèmes des cas d'utilisation partagés

A



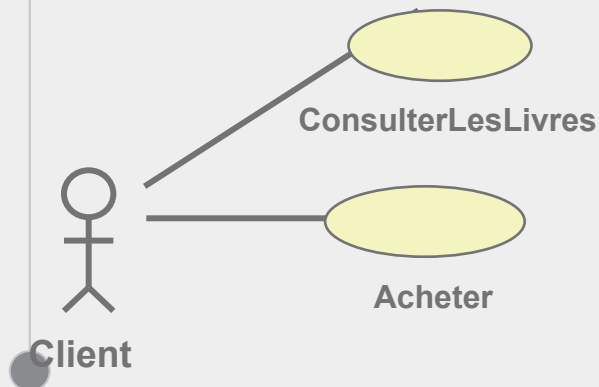
- On insiste sur le fait que l'une des fonctions importantes est d'accueillir des internautes quelconques et de leur permettre de consulter la liste des livres sans que leur objectif soit d'acheter
- La différence est faite entre un internaute et un client (potentiellement habitué)
- Une personne peut changer de rôle dynamiquement en jouant le rôle internaute puis de client.
- Ce changement de rôle est une caractéristique extérieure au système

B

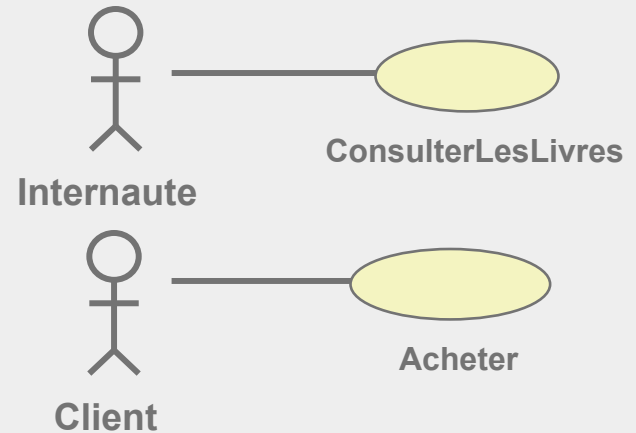


# Problèmes des cas d'utilisation partagés

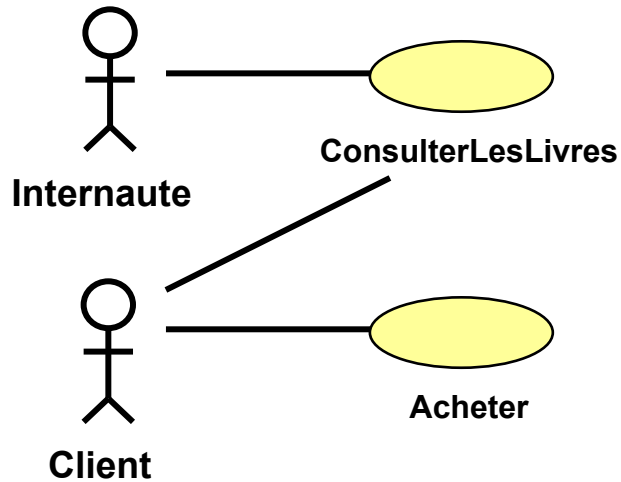
A



B

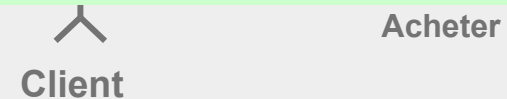


C



- Il est considéré comme important de séparer les clients des internautes
- ConsulterLesLivres est un cas d'utilisation normal pour un client
- Acheter aussi

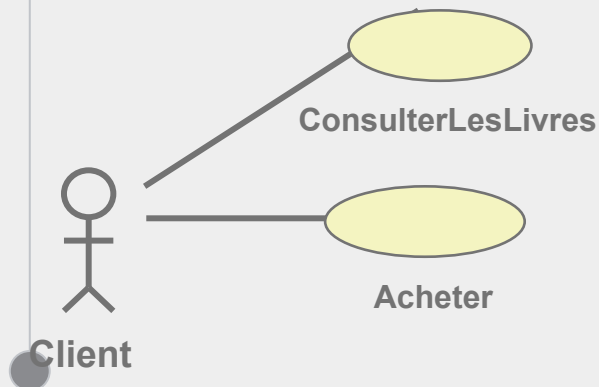
D



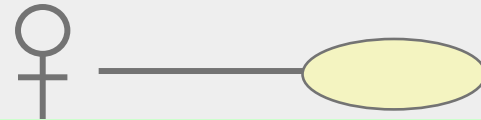


# Problèmes des cas d'utilisation partagés

A

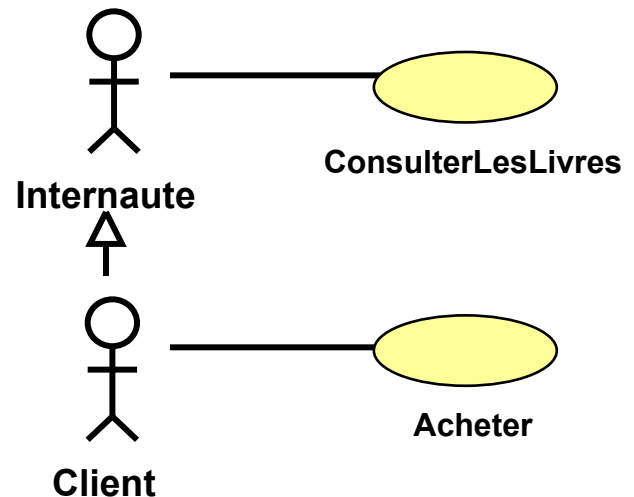
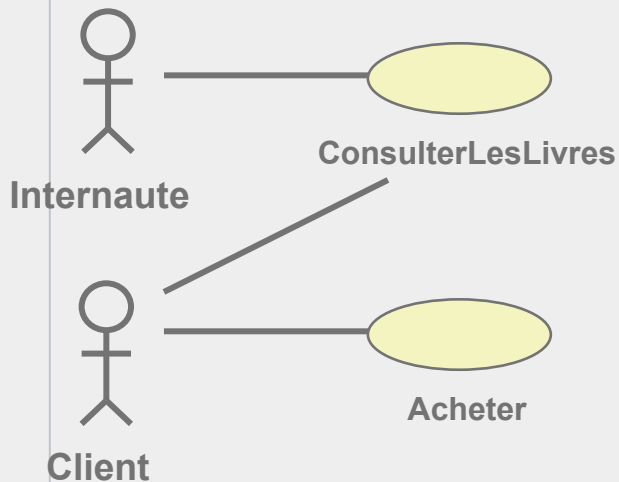


B



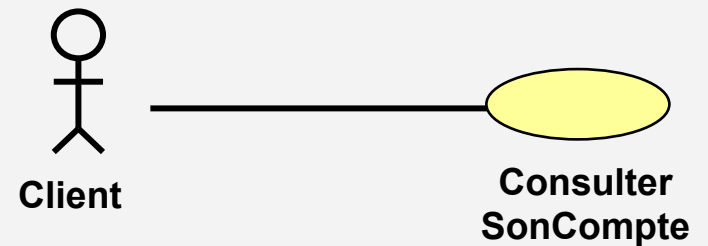
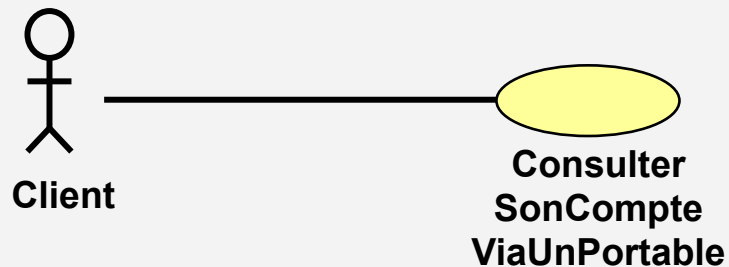
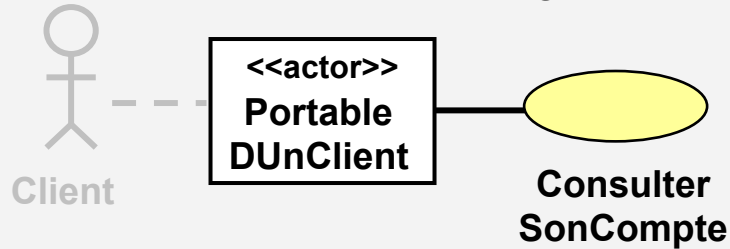
- Un client peut tout faire ce que peut faire un internaute (héritage des cas d'utilisation)
- Un client est un cas particulier d'internaute (spécialisation)
- **La dernière règle doit être respectée**

C



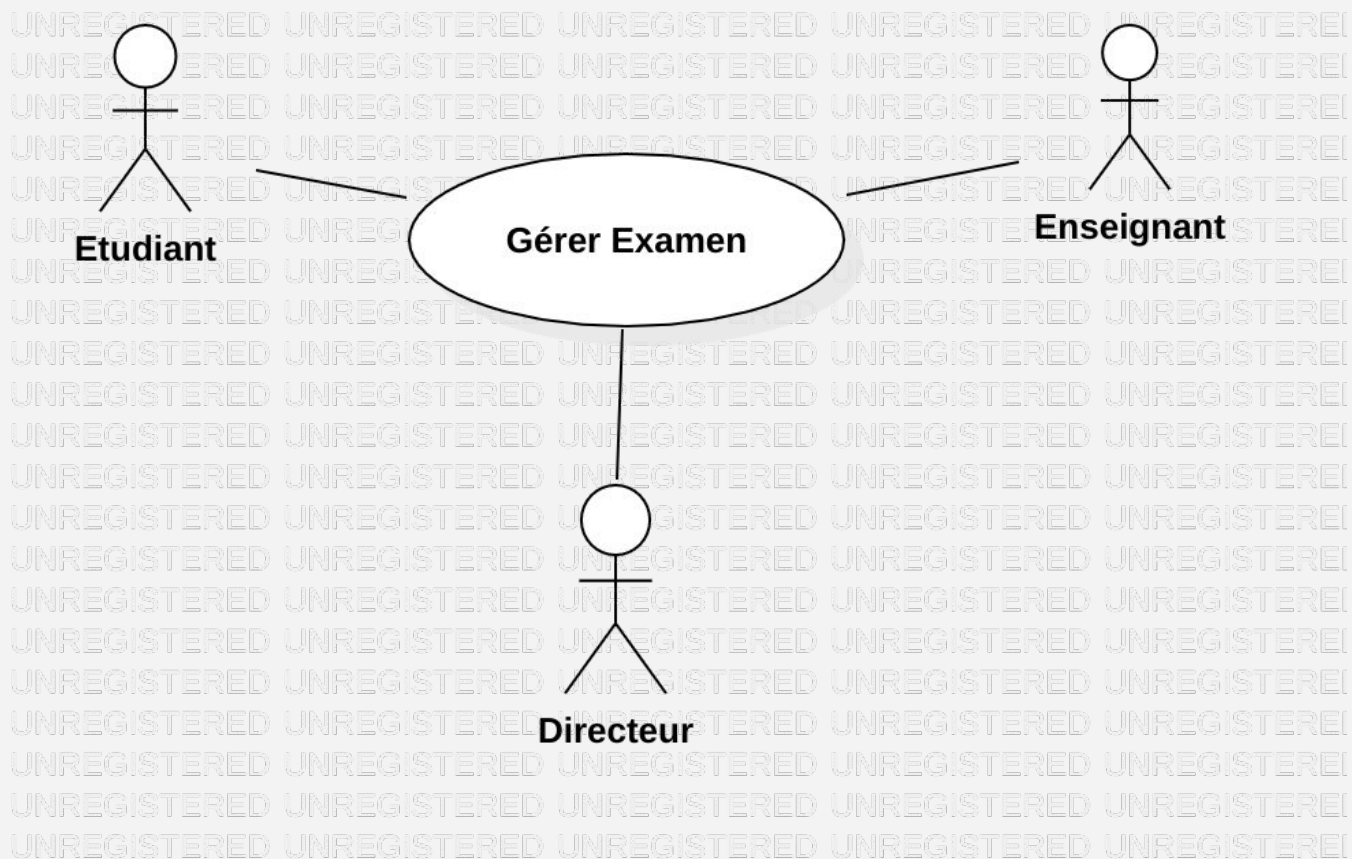
## Exercice 3

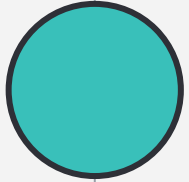
Décrivez les SIA de ces diagrammes en mettant en avant leurs différences.



## Exercice 4

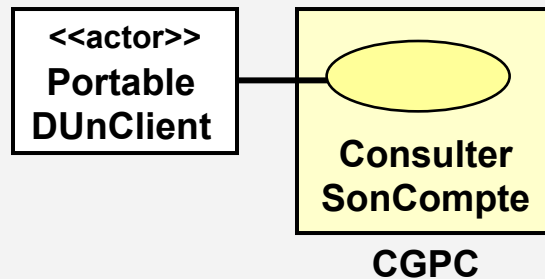
Améliorez ce diagramme!



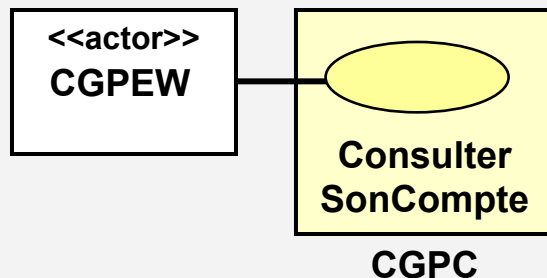


## Le système

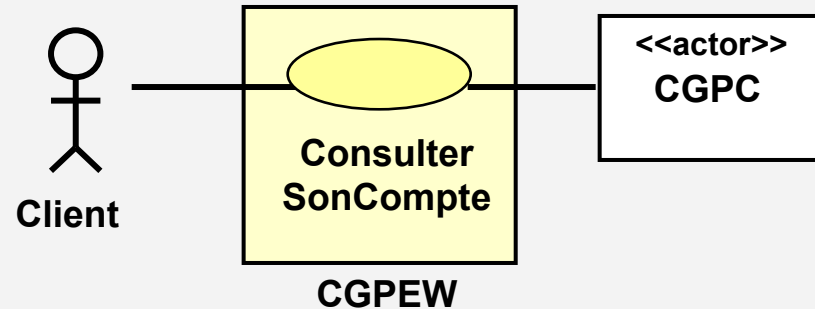
# Ne pas confondre les systèmes...



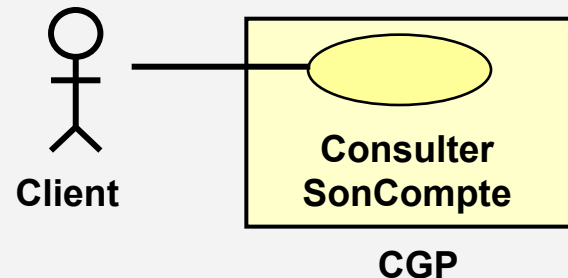
Projet: développer le système centralisé accessible à partir d'un portable



Projet: développer le système centralisé accessible à partir du système embarqué CGPEW



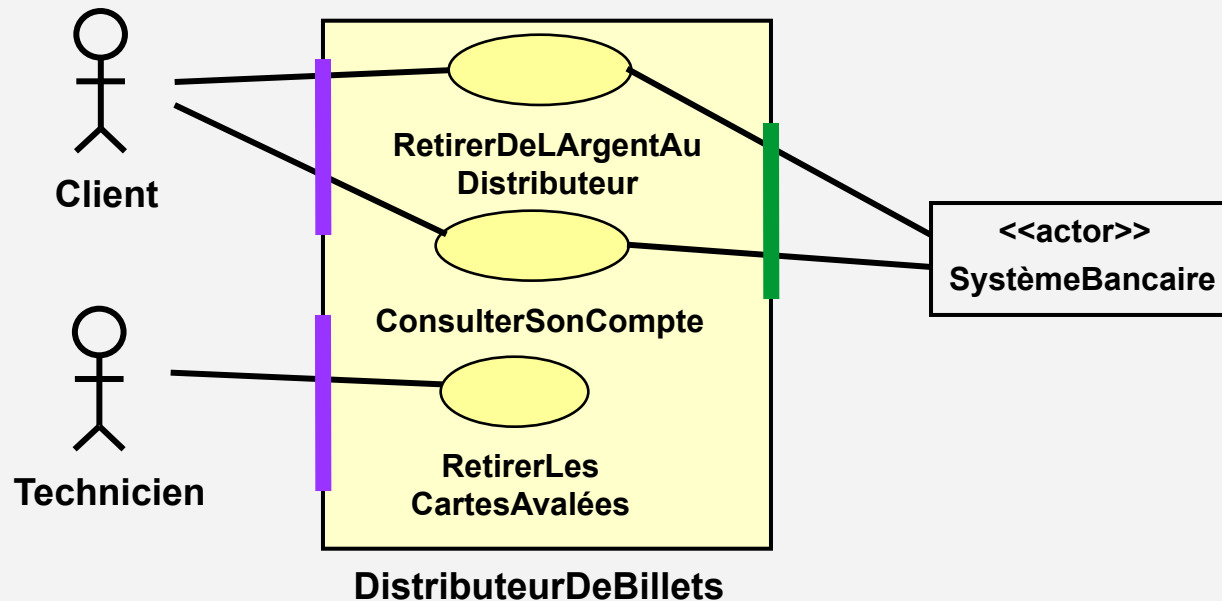
Projet: développer le système embarqué dans un portable pour accéder au système centralisé

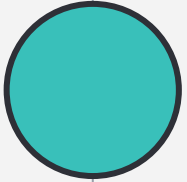


Projet: développer le système global

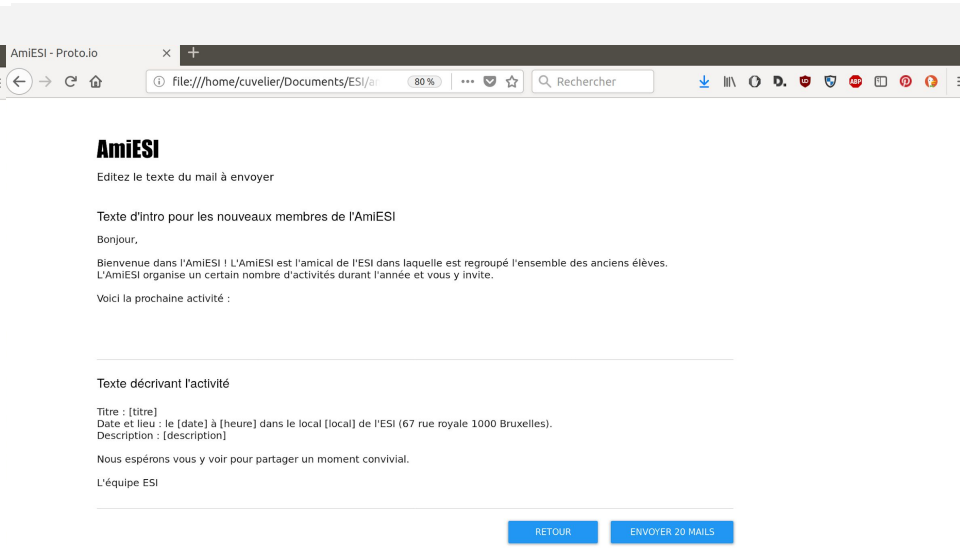
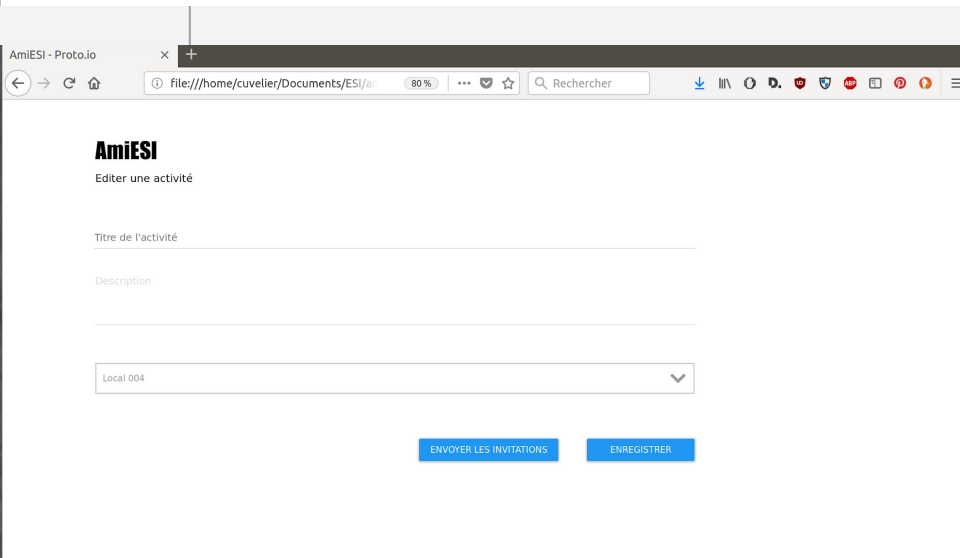
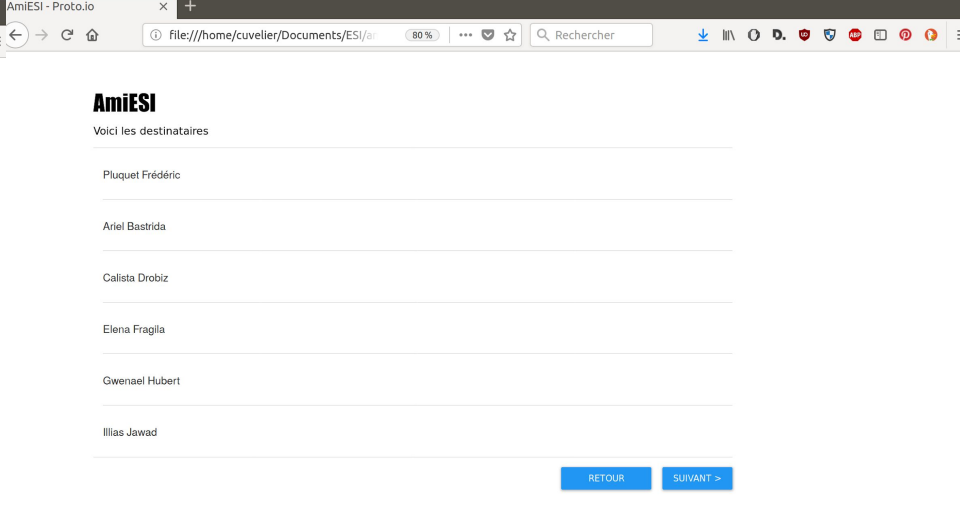
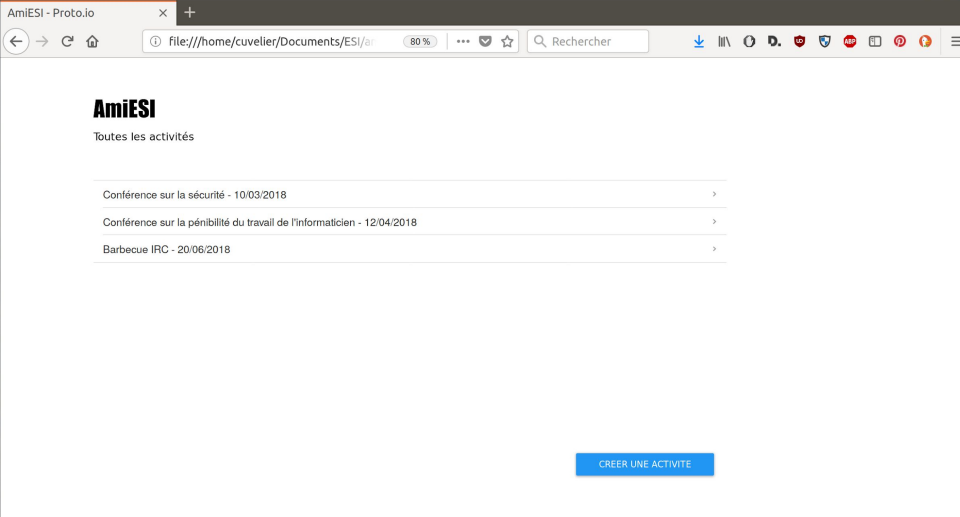
# Limites du système et interfaces

- humain → IHM, interface homme - système
- logiciel → API, interface système - système

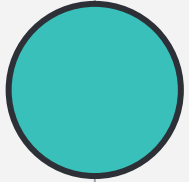




## Démo: IHM





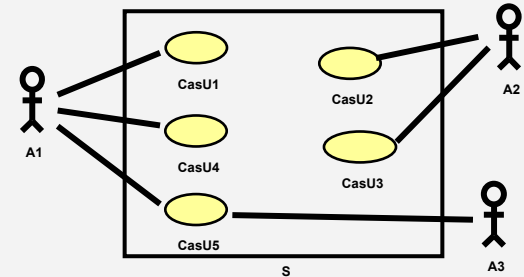


# Modèle de Cas d'Utilisation

# Diagramme vs. Modèle

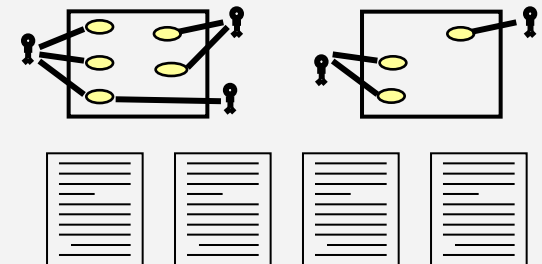
- **Diagramme de cas d'utilisation**

- notation graphique



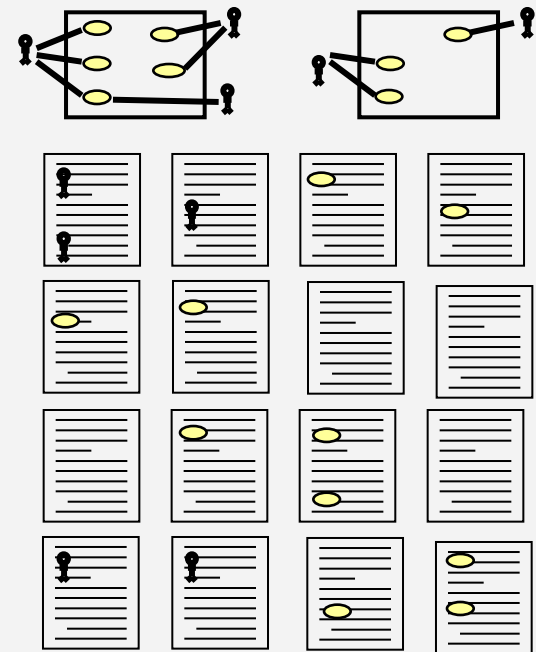
- **Modèle de cas d'utilisation**

- descriptions textuelles,
- diagrammes de cas d'utilisation
- diagrammes de séquences
- ...



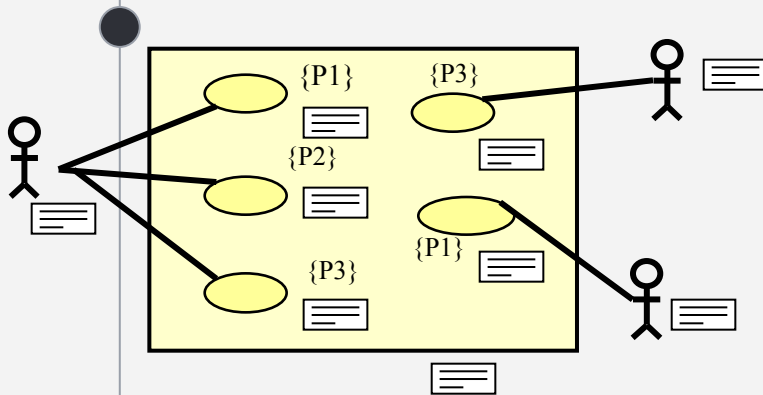
# Modèle "communicationnel"

- Modèle informel
- Multiple intervenants (stakeholders)
- Texte structuré + diagrammes + ref...
- Diagrammes
  - pour les réunions de "brainstorming"
  - pour simplifier la communication
  - pour structurer les documents
  - pour structurer le développement

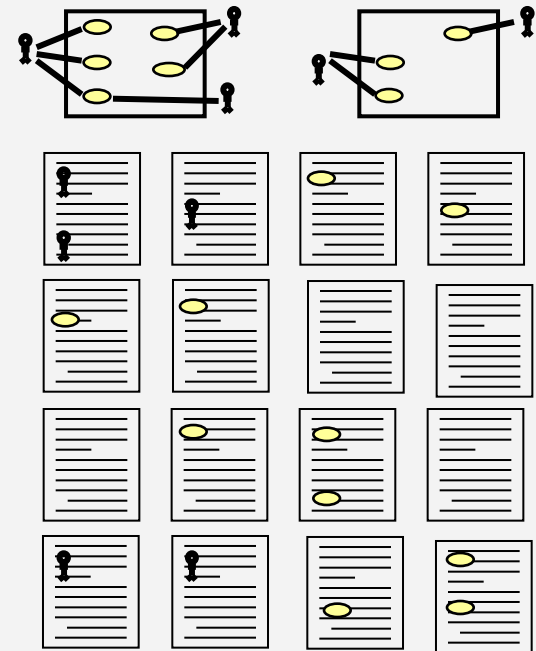
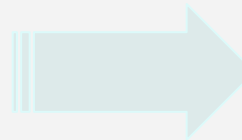


# Modélisation par “Consensus Grandissant”

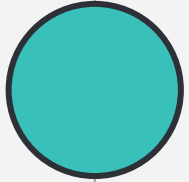
- Modèle informel
- Différents styles
- Différentes interprétations
- Raffinements successifs



Modèle Conceptuel  
des traitements

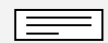
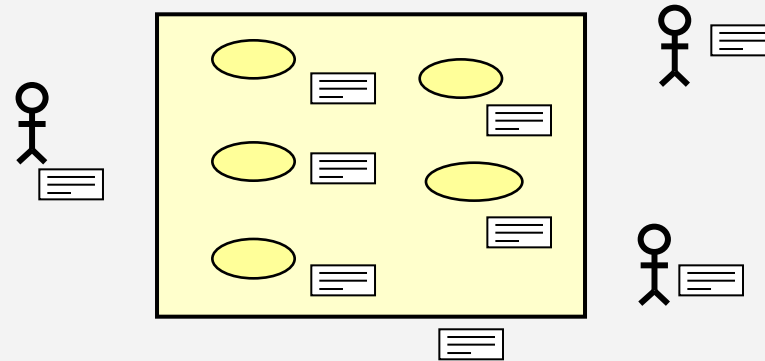


Modèle détaillé  
(UC Specifications)



# Modèle Conceptuel des Traitements

# Description préliminaire de chaque élément



Quelques lignes

→ Éviter les incompréhensions

→ Collaboration

# Description préliminaire du système

- **Identificateur**

- Baptiser le système le plus tôt possible
- Risque d'être référencé dans toute la vie future de l'entreprise

- **Brève description textuelle** (quelques lignes max.)

Le système logiciel **CGDR24/7** ("Crédit Grenoblois Dans la Rue, 24h/24, 7j/7"), déployé sur un distributeur de billets de la gamme DB600, a pour but de contrôler l'ensemble des fonctions associées au distributeur en incluant son fonctionnement normal, mais aussi sa sécurité et sa maintenance.

**CGDR24/7**



Client



# Identifier un Acteur

## Règles

- forme nominale
- vocabulaire métier
- style CamelCase

## Importance :

- réunions avec le client
- manuels utilisateurs
- IHM
- Code





# Description des acteurs



Un **guichetier** est un employé de la banque chargé de faire l'interface entre le système informatique et les clients qu'il reçoit au comptoir. Le guichetier peut réaliser les opérations courantes : création d'un compte, dépôt et retrait d'argent, etc.



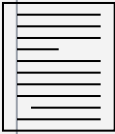
# Identifier un Cas d'Utilisation

## Règles

- forme verbale décrivant une action
  - l'acteur est généralement le sujet
  - éviter les connecteurs (et, ou, puis, ...)
  - vocabulaire métier
  - style CamelCase ou littéral
  - terme générique comme "Gérer" en cas de besoin seulement
    - Gérer = Créer, Supprimer, Ajouter, Modifier, ...
- Exemple: GérerLesDroits

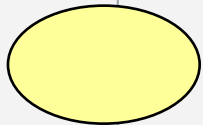
## Importance :

→ réunions avec le client



# Description brève des cas d'utilisation

- **description textuelle simple**
- se concentrer sur le **scénario nominal**
- compréhensible par tous
- pas trop de détails

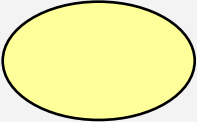


**Retirer  
DeLArgent  
AuDistributeur**

Lorsqu'un *client* a besoin d'argent liquide il peut en utilisant un distributeur retirer de l'argent de son compte. Pour cela :

- le *client* insère sa carte bancaire
- le *système* demande le code
- le *client* choisit le montant du retrait
- le *système* vérifie qu'il y a suffisamment d'argent
- si c'est le cas, le *système* distribue les billets et débite le compte du client
- le *client* prend les billets et retire sa carte

# Réécriture dans un style essentiel

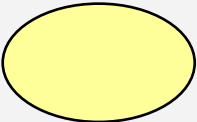


**Retirer  
DeLArgent  
AuDistributeur**

- le *client* insère sa carte bancaire dans le distributeur
- le *système* demande le code pour l'identifier
- le *client* tape le montant du retrait sur le clavier
- le *système* vérifie qu'il y a suffisamment d'argent
- le *système* affiche un message de confirmation
- ...



Extraction de l'essentiel



**Retirer  
DeLArgent  
AuDistributeur**

- le *client* s'identifie
- le *système* vérifie l'identification
- le *client* détermine le montant du retrait
- le *système* vérifie qu'il y a suffisamment d'argent

# Le Processus

## (1) Définir le modèle de cas d'utilisation

(1.1) Trouver les acteurs

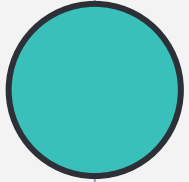
(1.2) Décrire brièvement chaque acteur

(1.3) Trouver les cas d'utilisation

(1.4) Décrire brièvement chaque cas d'utilisation

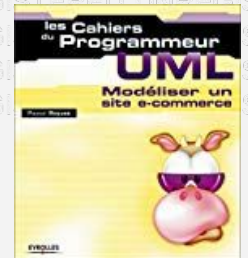
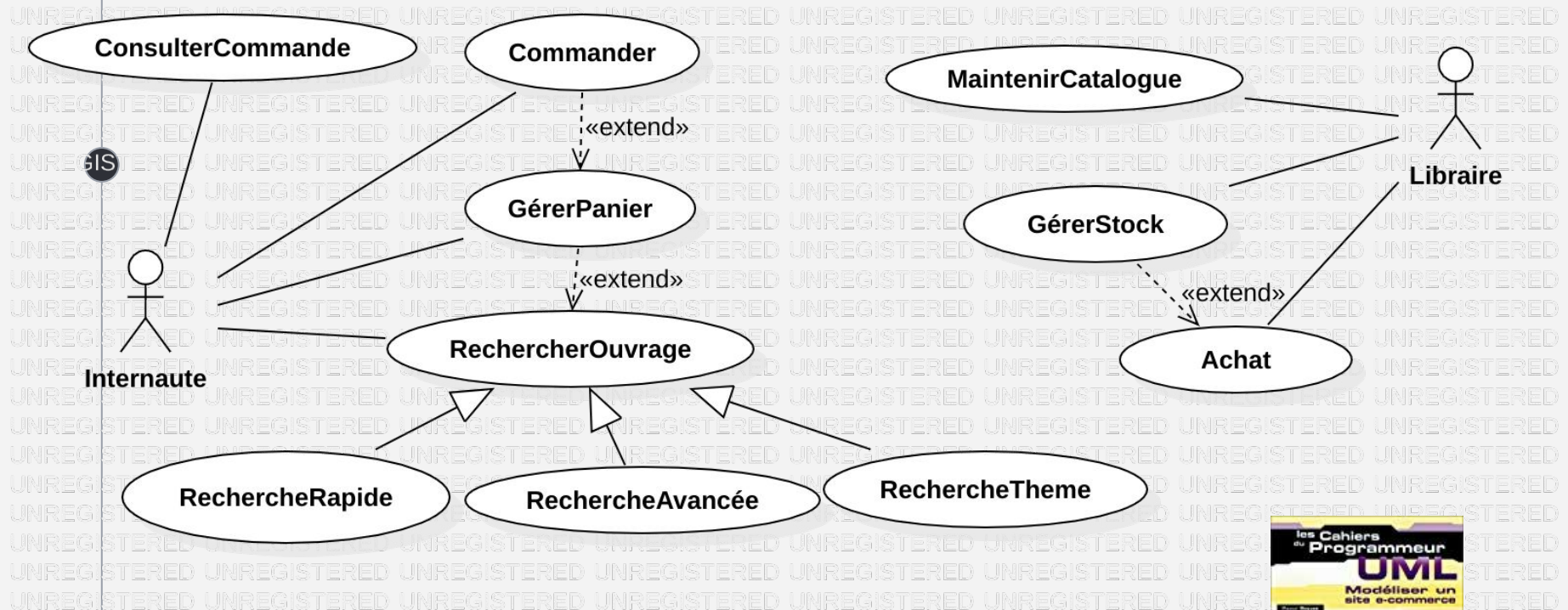
(1.5) Décrire le modèle comme un tout

## (2) Détailler chaque CU



## Démo: MCT

# Découpe en package



# Découpe en package

Catalogue

Vente

MaintenirCatalogue

GérerStock

«extend»

Achat

Libraire

ConsulterCommande

Commander

GérerPanier

«extend»

RechercherOuvrage

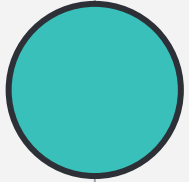
RechercheTheme

RechercheRapide

RechercheAvancée

Internaute





# Conclusion

# ATTENTION

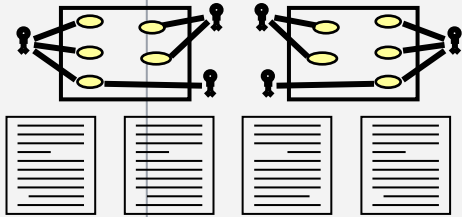
*"Congratulations: Use Cases Have Been Written, and Are Imperfect"*

[Applying UML and Patterns, Craig Larman]

*"A big danger of use cases is that people make them too complicated and get stuck. Usually you'll get less hurt by doing too little than by doing too much".*

[UML Distilled, Martin Fowler]

# Modèle conceptuel des traitements



- Équivalent à définir une **table des matières** et des résumés pour chaque chapitre
- Pas de règles strictes
- Effectuer les meilleurs regroupement possibles
- Rester simple !
- Structuration possible en termes de paquetages
- **Culture d'entreprise**

Stabilisation du modèle par **consensus grandissant**