- 3 -

Modélisation dans le développement logiciel

UNIVERSITÉ DE LORRAINE

Analyse et Conception de Logiciels

Horatiu CIRSTEA

Exemples cours

Une partie inspirés de

- Software Engineering, 9th ed. - Ian Sommerville
- Object-Oriented Software Engineering: Practical Software Development using UML and Java - Timothy C. Lethbridge, Robert Laganière
- UML 2 par la pratique - Pascal Roques, Eyrolles, 2009

Modélisation système

- le développement de modèles abstraits d'un système
- un modèle ➠ un point de vue (perspective)
- souvent utilisant des notations graphiques (Unified Modeling Language - UML)

Modélisation système

- modéliser le système existant
  - base pour l'analyse des besoins
- modéliser le nouveau système
  - expliquer les services

Modèles (graphiques)

- Faciliter la communication
  - peuvent être incomplets, voire incorrects
- Documenter le système
  - modèles précis
- Générer l'implantation
  - modèles précis et corrects

➥ utilisés tout le long du cycle de développement

Modélisation système

- Un modèle est créé à plusieurs niveaux (avec plus au moins de détails)
- L'abstraction est un principe fondamental de la modélisation
  - ✦ regarder les informations importantes à un moment (niveaux) donné
  - ✦ cacher les détails

Un bon modèle

- utiliser une notation standard
- être compris des clients et utilisateurs
- permettre aux développeurs de bien saisir le système

Modélisation

- Plusieurs points de vues (perspectives)
- Chaque point de vue (view) contient
  - informations uniques
  - informations partagées avec d'autres views
  - informations partagées cohérentes

Perspectives

- *externe*
  - modéliser le contexte (l'environnement)
- *interaction*
  - modéliser les interactions avec l'environnement et entre les composants
- *structurelle*
  - modéliser l'architecture et les données
- *comportementale*
  - modéliser la dynamique

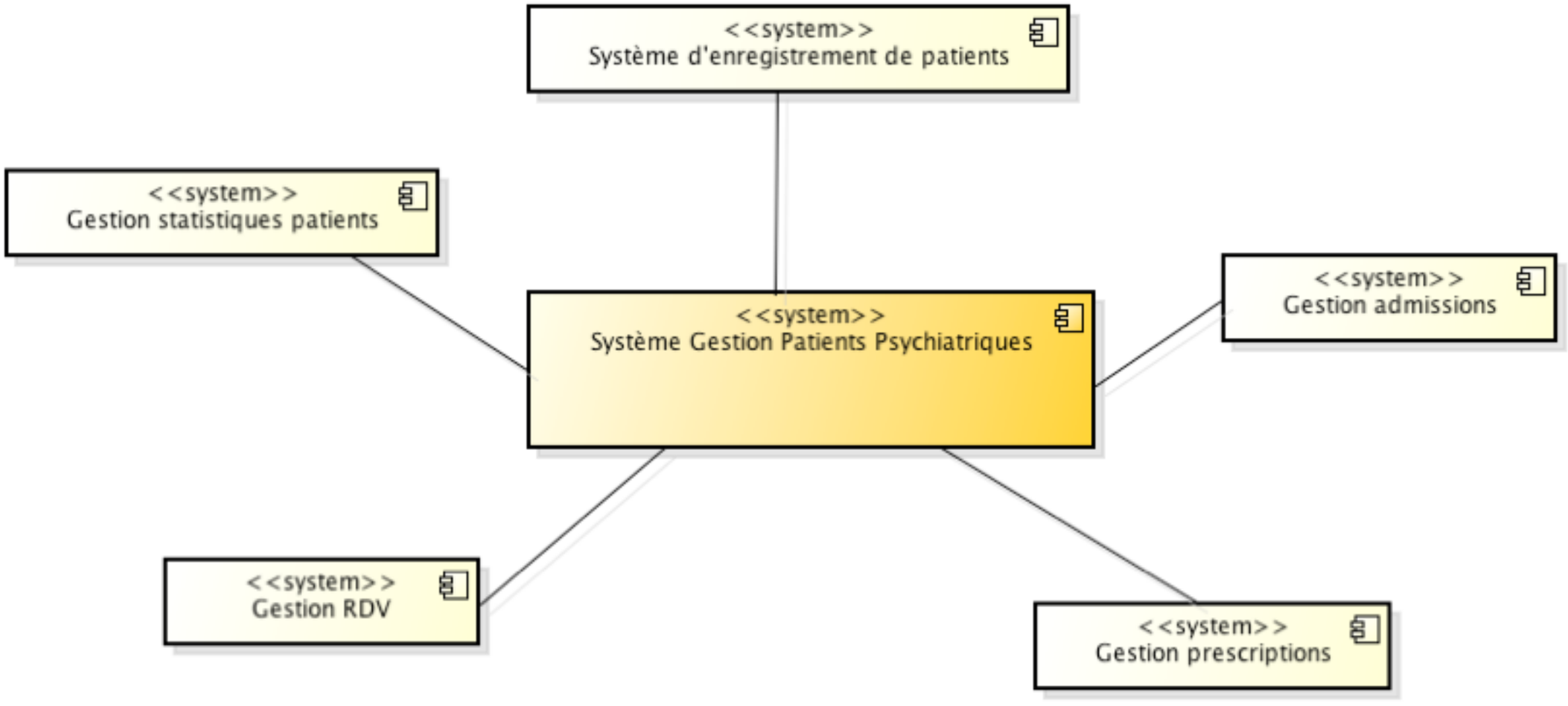
Modélisation du contexte

- illustrer le contexte opérationnel d'un système
- donner des informations sur les frontières du système
- illustrer les relations avec les autres systèmes/acteurs

Frontière système

- délimiter les fonctionnalités et montrer les systèmes utilisés ou utilisant le système
- impact important sur le cahier des charges
- peut être politique

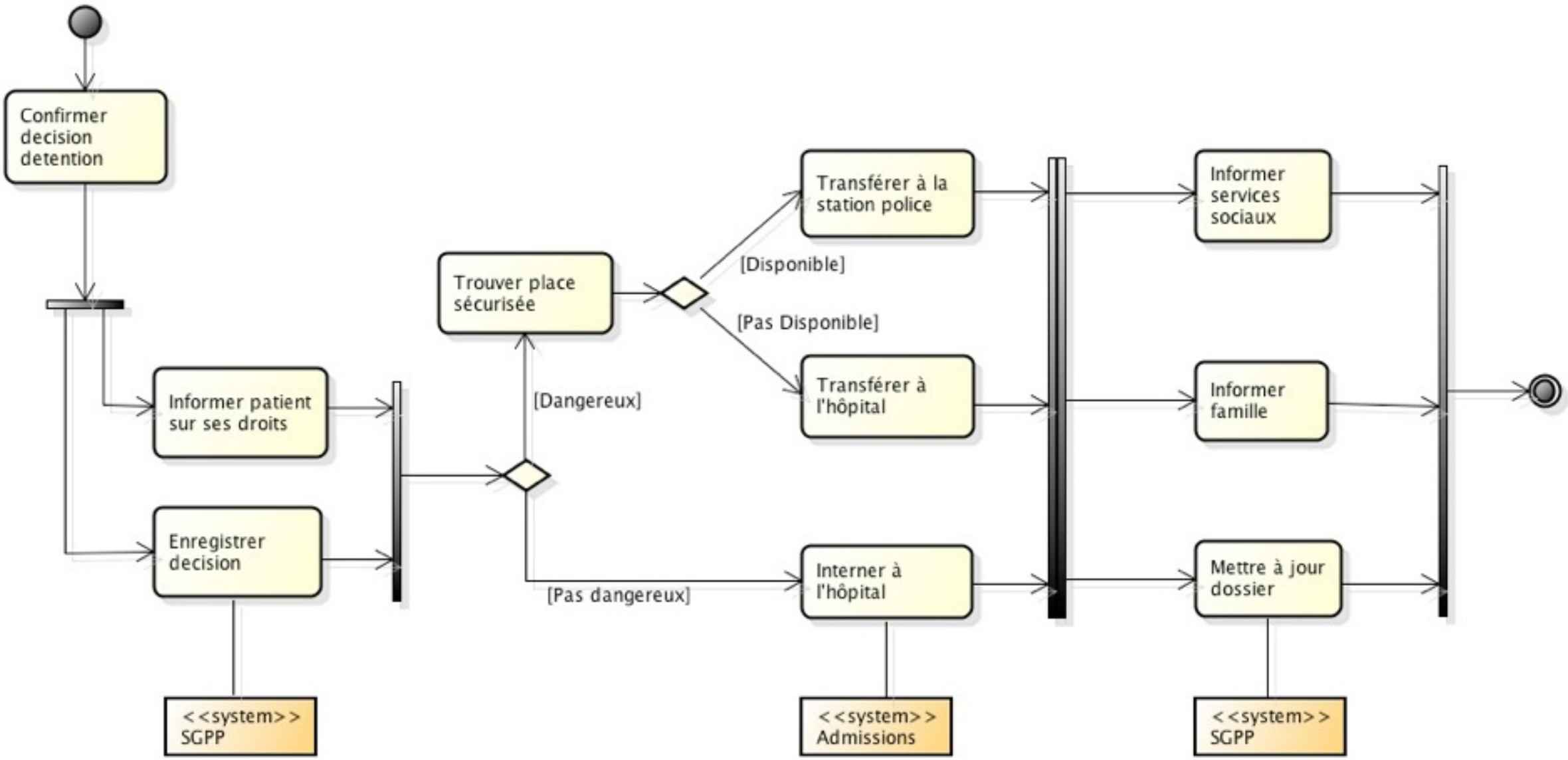
Exemple


<<system>>
Système d'enregistrement de patients
<<system>>
Gestion statistiques patients
<<system>>
Système Gestion Patients Psychiatriques
<<system>>
Gestion admissions
<<system>>
Gestion RDV
<<system>>
Gestion prescriptions

Perspective dynamique

- complète la perspective statique de la modélisation du contexte
- illustre l'utilisation du système
- éventuellement avec des diagrammes d'activité

Système consultations



Modélisation interactions

- permet d'identifier les besoins
- identifier les problèmes de communication
- identifier les problèmes de fiabilité et performance

➠ UML : use-cases, diagrammes d'interaction (diagrammes de séquence)

UML

Unified Modeling Language

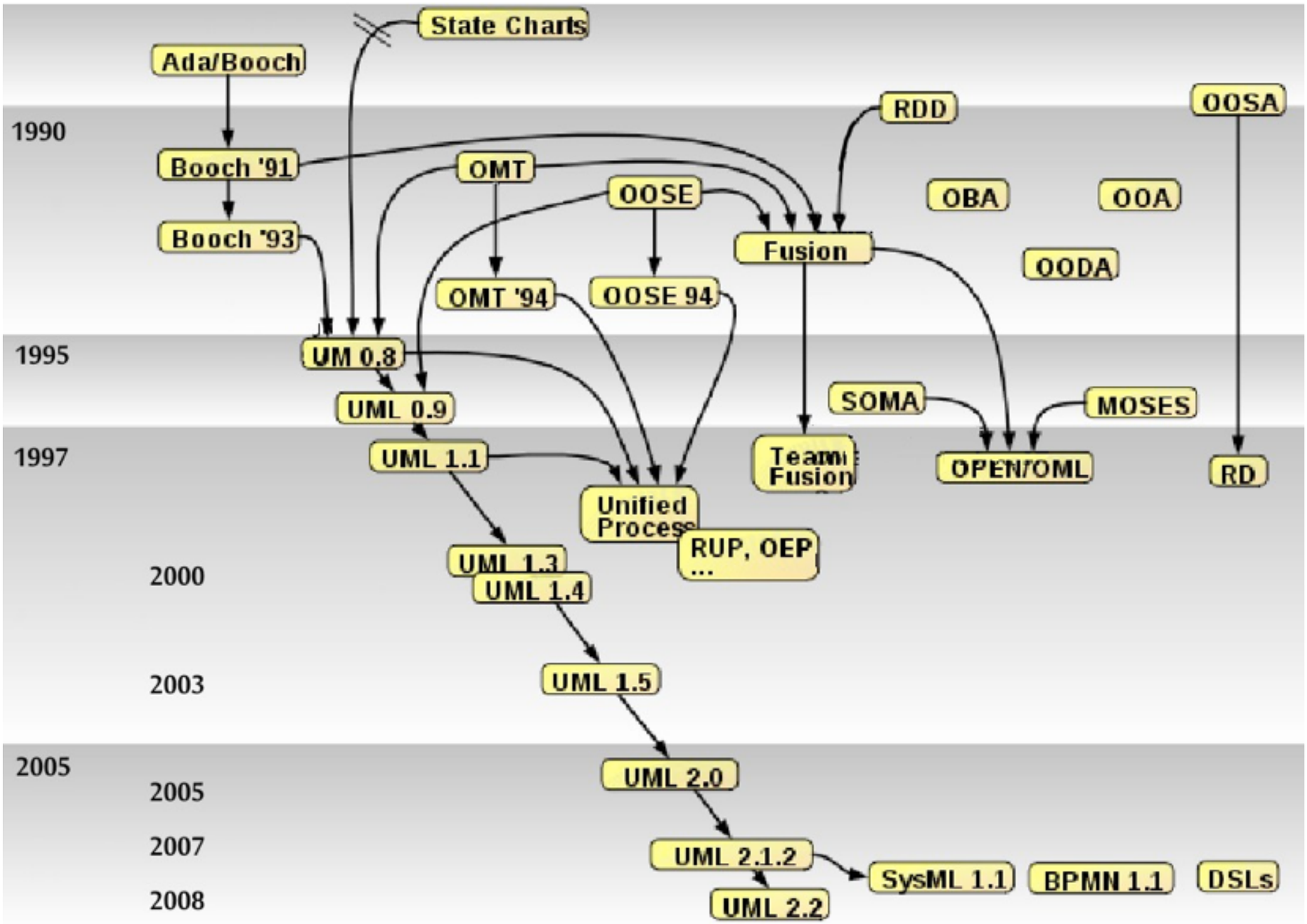
UML

- ancêtres
  - ✦ OMT (James Rumbaugh)
  - ✦ Booch (Grady Booch)
  - ✦ OOSE (Ivar Jacobson)
- Standardisation OMG (Object Management Group) depuis 2000

UML

Langage graphique pour la modélisation

- fin 1980 - début 1990 : premiers processus de développement orienté-objet
- Rumbaugh et Booch décident de fusionner leurs approches en 1994.
- En 1995, Jacobson, se joint à l'équipe
- En 1997, le « Object Management Group » (OMG) lance le processus de standardisation UML

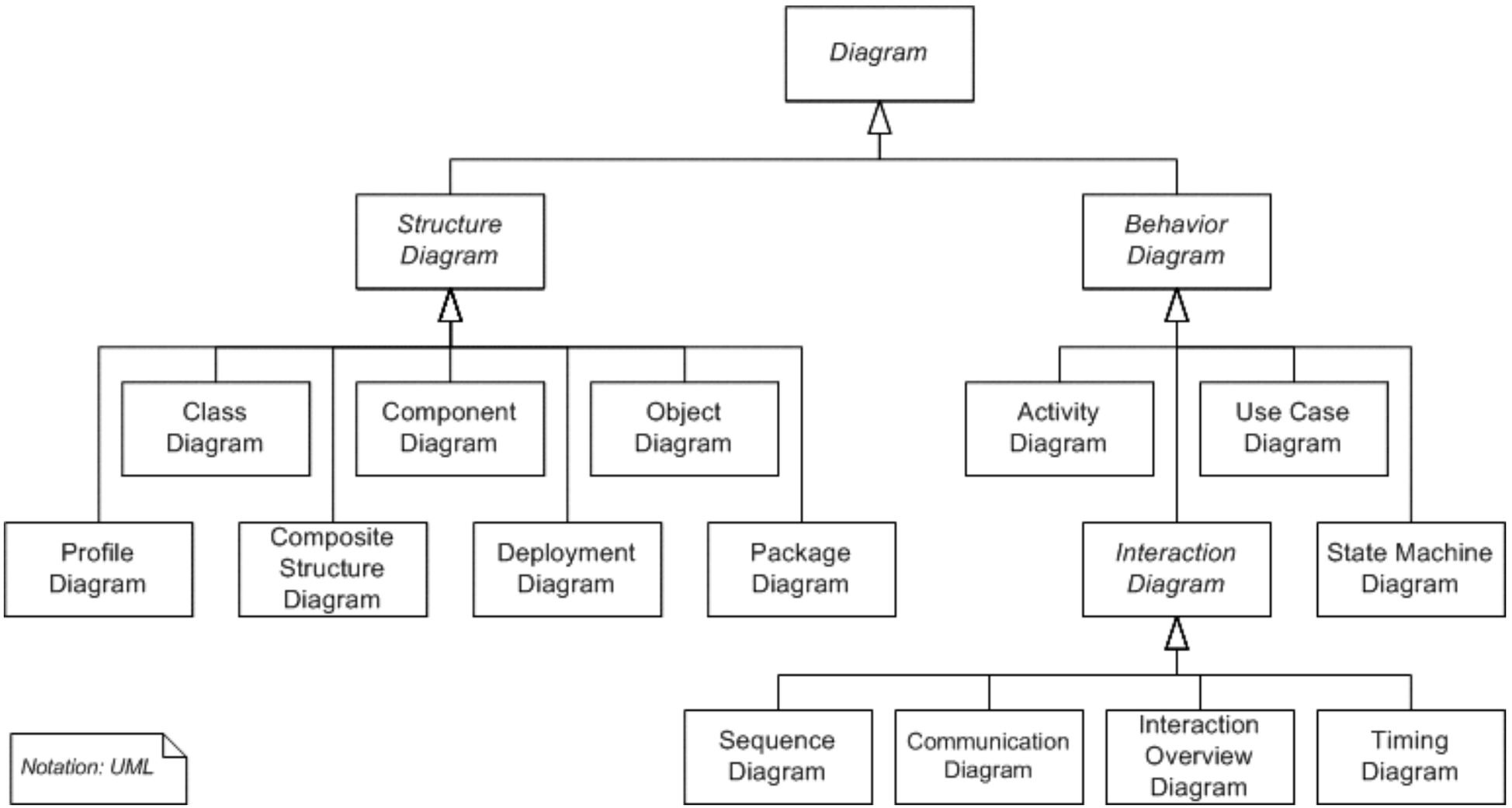


Axel Scheithauer/Marcel Douwe Dekker (WikiMedia)

Utilisation UML

- comprendre et décrire des besoins,
- communiquer,
- spécifier,
- documenter des systèmes,
- esquisser des architectures logicielles,
- concevoir des solutions.

Diagrammes UML



Paulo Merson(WikiMedia)

Avantages

- Indépendant des méthodes de développement
- Indépendant des langages de programmation
- Adapté à toutes les phases du développement
- Peut être utilisé pour modéliser autre chose que des logiciels (mécanisme d'extension)

Caractéristiques de UML

Une *sémantique* détaillée

Un *mécanisme* intégré d'extension

Un langage textuel associé utilisé pour la description des contraintes: *Object Constraint Language* (OCL)

L'objectif de UML est d'assister le développement du logiciel

Ce n'est pas une *méthodologie*

Utilisation des diagrammes dans le processus de développement

Analyse des besoins

- *Use-case diagrams* : décrire les fonctionnalités du système (par rapport au point de vue des utilisateurs)
- *Activity diagrams* : décrire l'activité du système
- *Sequence diagrams* : décrire l'interaction entre utilisateurs et système

Analyse - Conception

- *Class diagrams* : structures de données du système, relations, etc. (faire abstraction de de ce qui est extérieur au système - IHM, SGBD, ...)
- *Sequence diagrams* : interactions entre objets lors de la réalisation de chaque fonctionnalité
- *State diagrams* : le comportement d'une famille d'objets en termes d'états et de transitions entre états

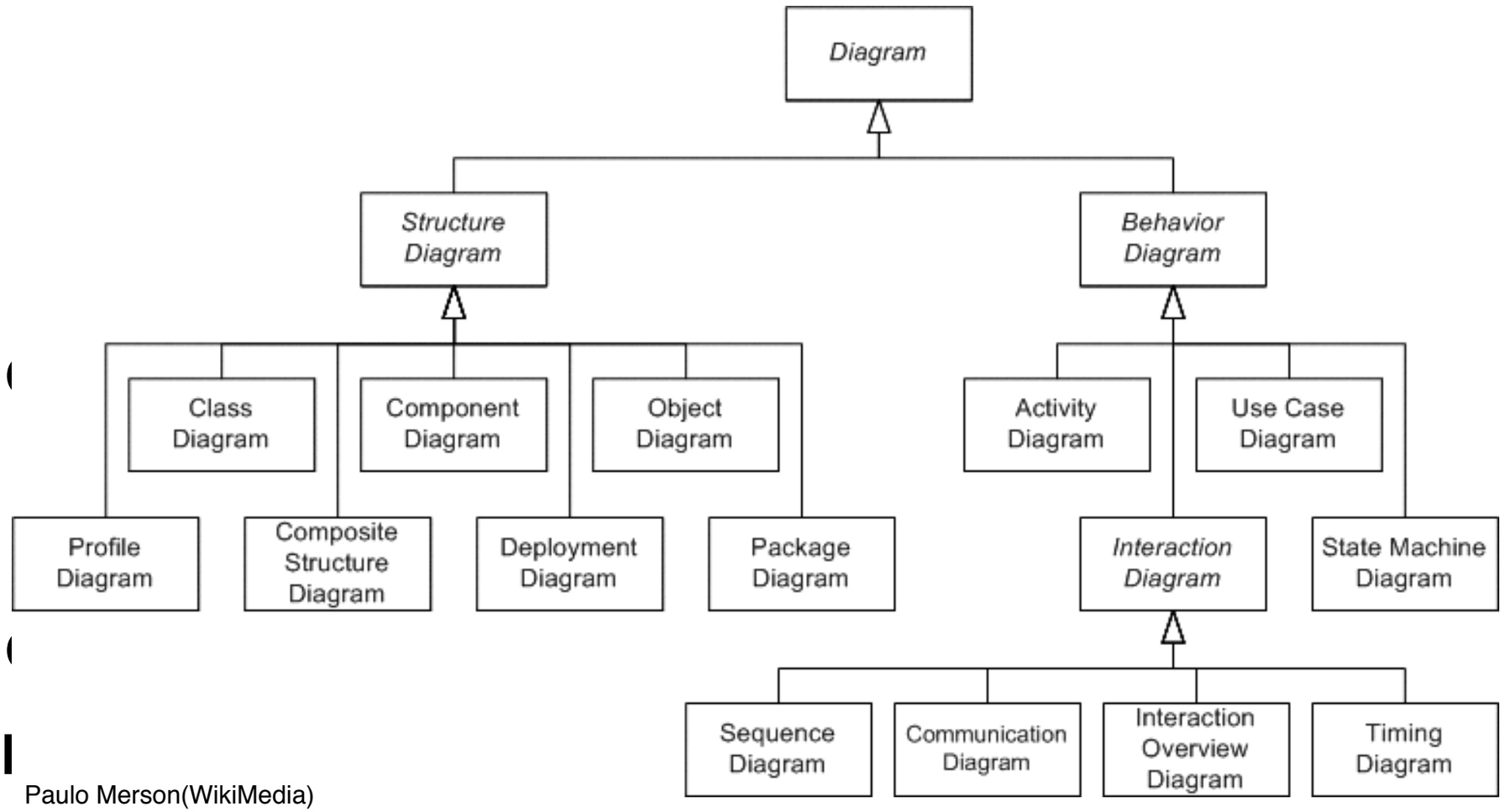
Conception

- *Class diagrams* : relations entre classes (plus détaillées)
- *Sequence diagrams* : interactions entre objets (en incluant les nouveaux objets)
- *Component diagrams* : architecture des composants
- *Deployment diagrams* : déploiement des composants sur les équipements

Implantation

- Génération automatique du code (squelettes de classes)
- Ecrire ***code*** en accord avec les différents diagrammes
- Ecrire ***tests*** en accord avec les différents diagrammes

- *externe*
  - modéliser l
- *interaction*
  - modéliser l
    l'environnei



Paulo Merson(WikiMedia)

- *structurelle*
  - modéliser l'architecture et les données
- *comportementale*
  - modéliser la dynamique

- 3.1 -
Modélisation interactions (1)

UNIVERSITÉ DE LORRAINE

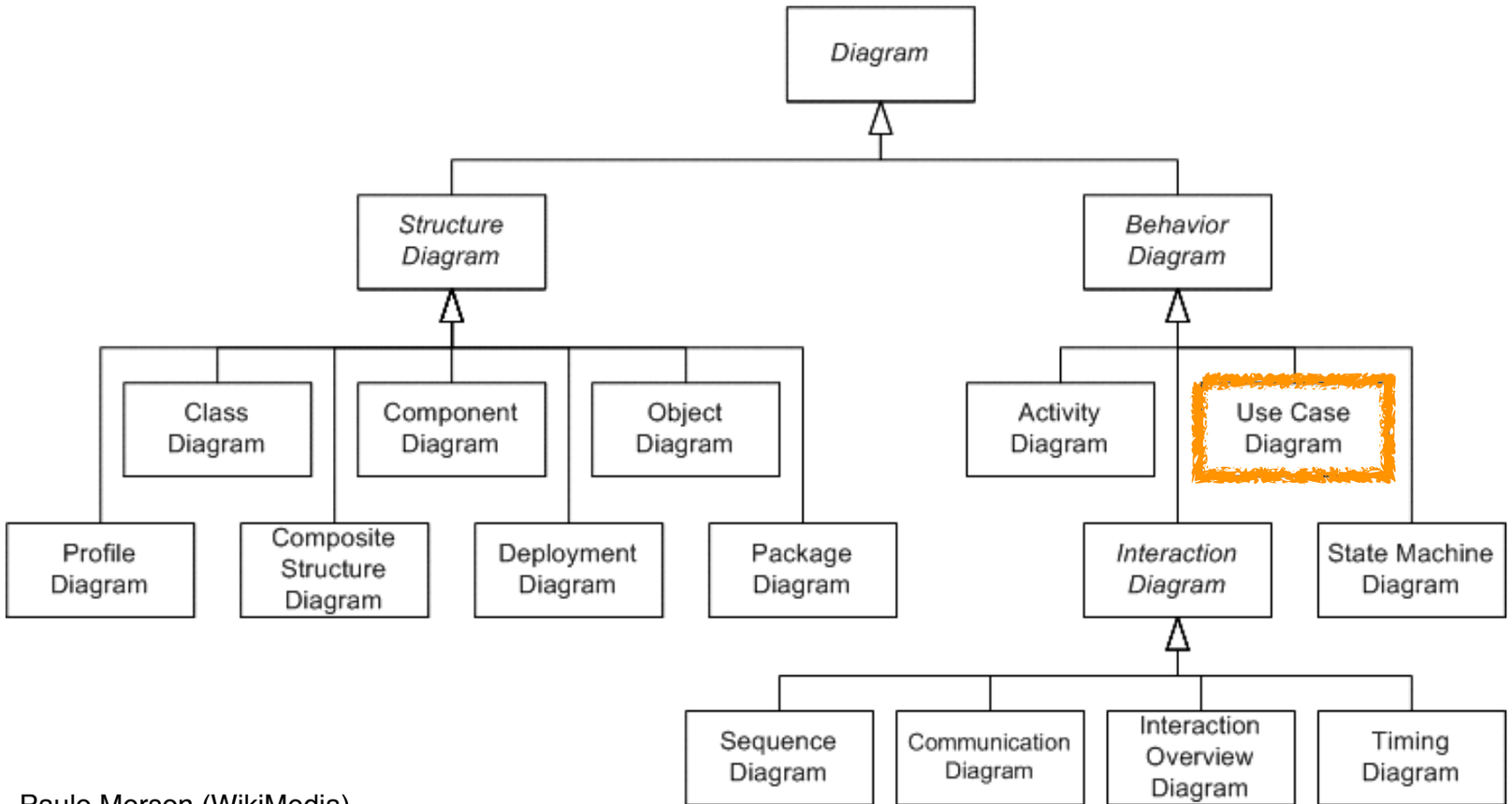
Analyse et Conception de Logiciels

Horatiu CIRSTEA

Perspectives

- ***externe***
  - modéliser le contexte (l'environnement)
- ***interaction***
  - modéliser les interactions avec l'environnement et entre les composants
- ***structurelle***
  - modéliser l'architecture et les données
- ***comportementale***
  - modéliser la dynamique

Diagrammes cas d'utilisation (Use-case diagrams)



Paulo Merson (WikiMedia)

Use-cases

- une séquence typique d'actions qu'entreprend un utilisateur afin d'accomplir une tâche donnée
- **Objectif** : *modéliser le système*
  - … du point de vue de la façon par laquelle l'utilisateur interagit avec le système
  - … afin d'accomplir ses objectifs

Identification use-case

Modélise un service rendu par le système

- regroupe une famille de scénarios d'utilisation selon un critère fonctionnel

Comment construire les use-cases ?

- Pour chaque acteur, identifier les intentions avec lesquelles il communique avec le système.

Use-cases

- un moyen de déterminer les besoins fonctionnels de l'application
  - partie intégrante du cahier des charges
  - formalisme basé sur le langage naturel,
  - compréhensibles par les experts et les informaticiens

Identification use-case

Exemple : **client GAB**

- retirer de l’argent
- consulter le solde
- déposer de l’argent

Exemple


Client
Retirer argent
SI banque

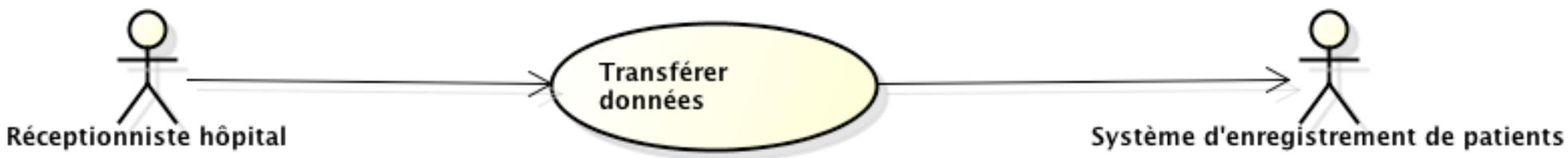
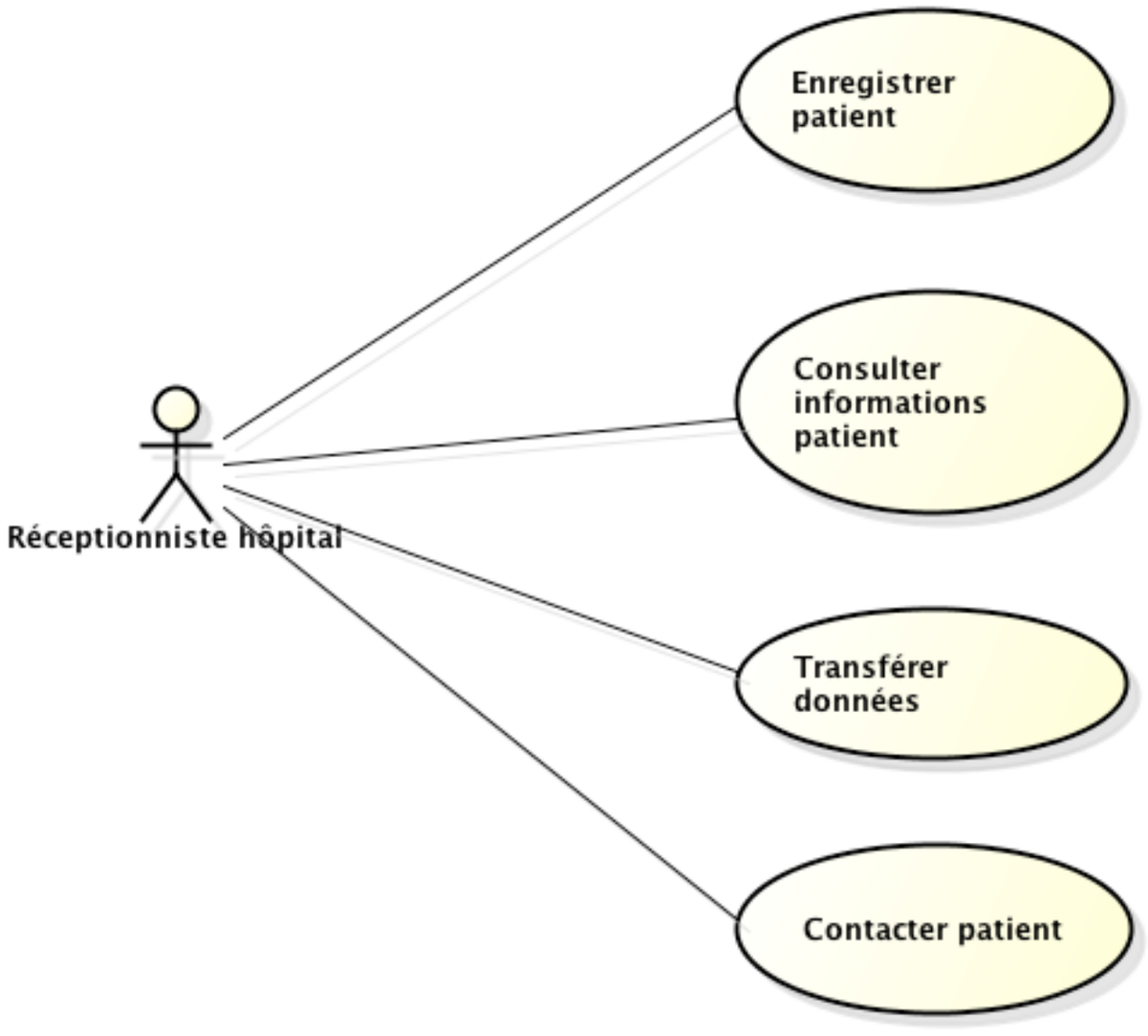

Réceptionniste hôpital
Transférer
données
Système d'enregistrement de patients

Diagramme composite use-cases



Les use-case

- Un ***use-case*** doit couvrir l'ensemble des étapes à suivre dans l'accomplissement d'une tâche donnée
- Le ***use-case*** doit être écrit d'une façon indépendante de toute interface utilisateur
- Un **modèle** avec use-cases contient
  - un ensemble de use-cases
  - un diagramme expliquant comment ils sont reliés

Les use-case

- Le *use-case* doit décrire l'interaction de l'utilisateur avec le système
  - ✦ et **non** les opérations que doit réaliser le système
- Un *use-case* ne doit inclure que les interactions avec le système
  - ✦ et **non** d'autres opérations réalisées par l'utilisateur indépendamment du système

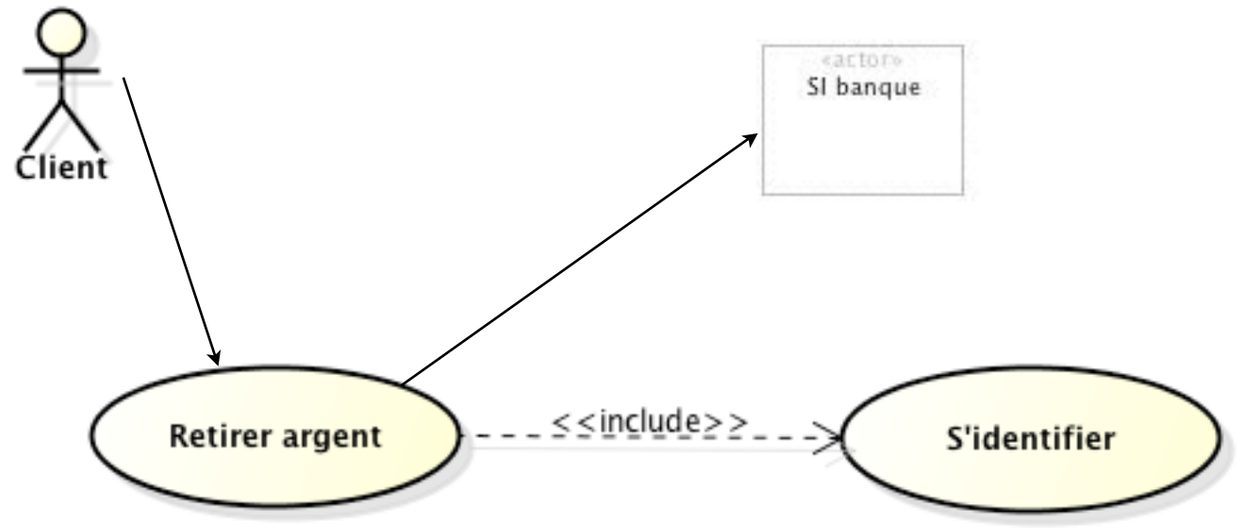
Utilisation use-cases

- décrire le comportement du système selon un point de vue *extérieur* (des utilisateurs)
- décrire les *fonctionnalités attendues* du système
- préciser les *contraintes non fonctionnelles*, concernant par exemple le matériel, le temps de réponse, etc.

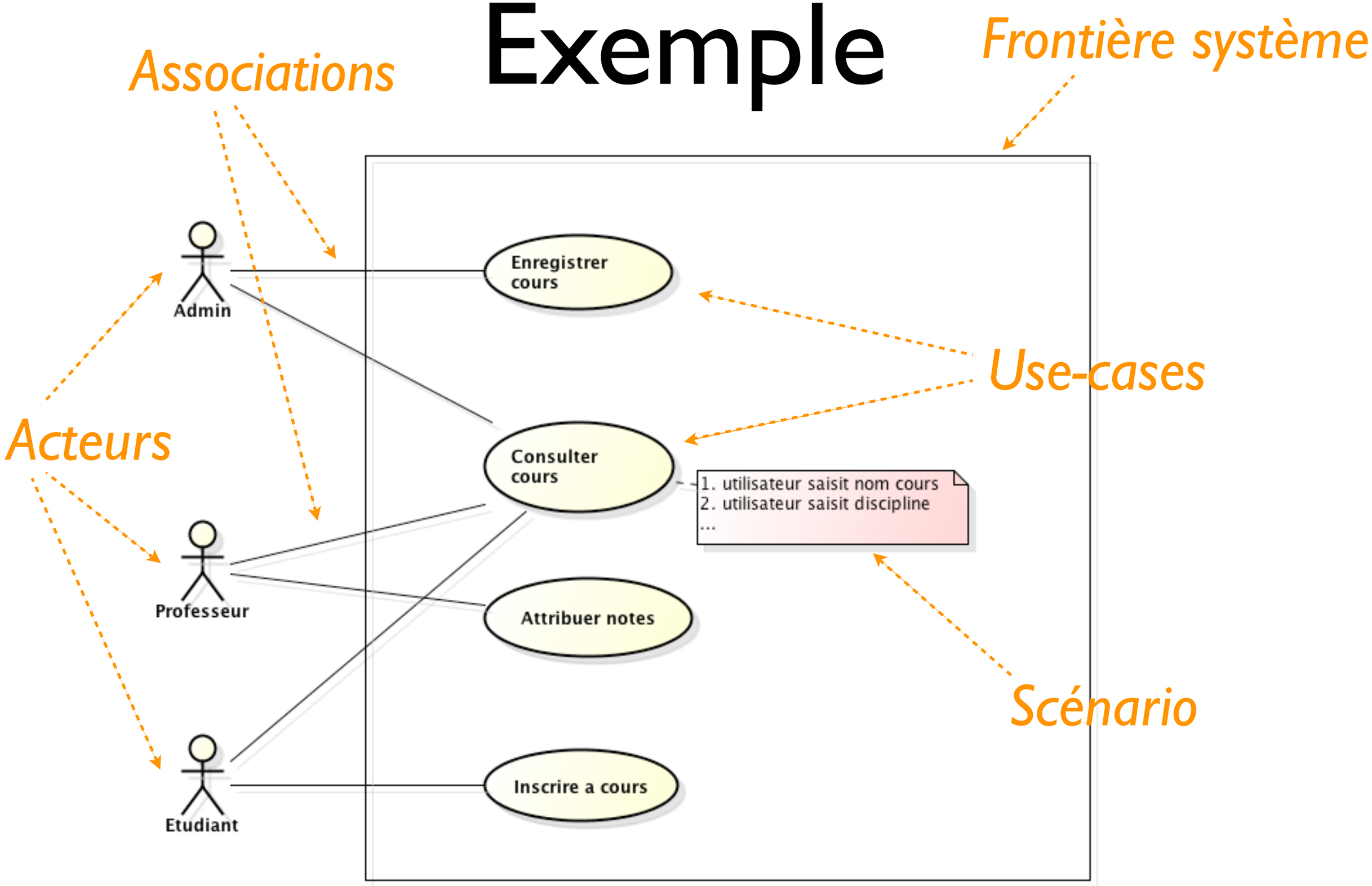
➡ Décrire le « **quoi** » et non le « **comment** »

Concepts

- Acteurs
- Use-case (et relations)
- Associations
- Scénarios (documents textuels pour les cas)
- Frontières système, packages
- prototypes IHM (optionnel)
- diagrammes d'activités (optionnel)



Exemple


Associations
Frontière système
Acteurs
Use-cases
Scénario
Admin
Professeur
Etudiant
Enregistrer cours
Consulter cours
Attribuer notes
Inscrire a cours
1. utilisateur saisit nom cours
2. utilisateur saisit discipline
...

L’acteur

- représente un **rôle** joué par un utilisateur (extérieur) qui interagit avec le système
  - ➡ personne, système, entité externe, le temps
- lié à un ou plusieurs use-cases
  - ➡ permet de grouper les use-cases
- n’a pas d’autres propriétés que les associations aux use-cases

Utilisateur vs. Acteur

- un ***utilisateur*** peut jouer plusieurs **rôles** (un professeur peut aussi participer à un cours)
- plusieurs ***utilisateurs*** peuvent jouer le même **rôle**

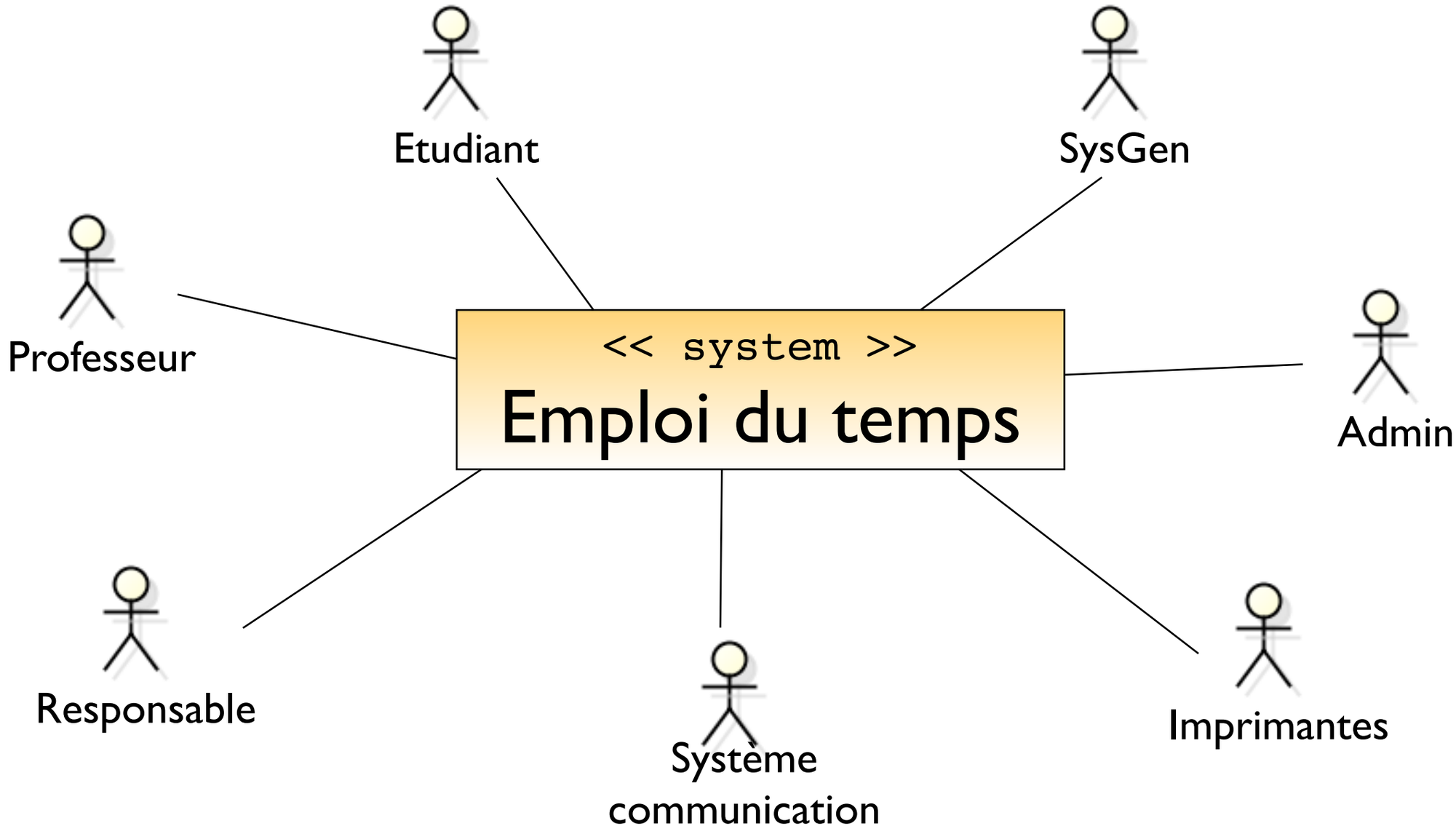
Les acteurs

- Les acteurs communiquent directement avec le système et non pas par un intermédiaire.
  - le caissier est un acteur, mais pas le client
- Les échanges entre acteurs ne concernent pas la modélisation du système !

Acteur principal/secondaire

- l'acteur **principal** utilise/modifie le système
  - ➡ le réceptionniste hôpital
  - ➡ le client d'un GAB
- l'acteur **secondaire** est sollicité pour obtenir une information complémentaire
  - ➡ système d'enregistrement de patients
  - ➡ le système d'autorisation de CB

Contexte statique



➠ résumé de la liste des acteurs

Scénarios

Un scénario est une instance d'un use-case

- un acteur spécifique ...
- à un moment spécifique ...
- avec des données spécifiques

➥ Décrit les étapes d'interaction entre un utilisateur et le système

Use-case

- collection de scénarios de base avec alternatives
  - *nom* : le but de l'acteur qui le déclenche
    - exemple: *consulter solde*, *inscrire étudiant*
  - une procédure complète et self-contained

Use-case

A task performed by one person in one place at one time, in response to a business event, which adds measurable business value and leaves the data in a consistent state.

C. Larman, Applying UML and Patterns

Décrire un use-case

- **Nom** : Une appellations courte et descriptive
- **Acteurs** : Énumérer les acteurs impliqués
- **Buts** : Expliquer ce que les acteurs veulent accomplir
- **Pré-conditions** : Décrire l'état du système avant l'exécution de ce use-case
- **Description** : Une courte description informelle
- **Référence** : Autre use-case apparentés
- **Déroulement** : Décrire chacune des étapes
- **Post-conditions** : Décrire l'état du système après l'exécution de ce use-case

Use-case

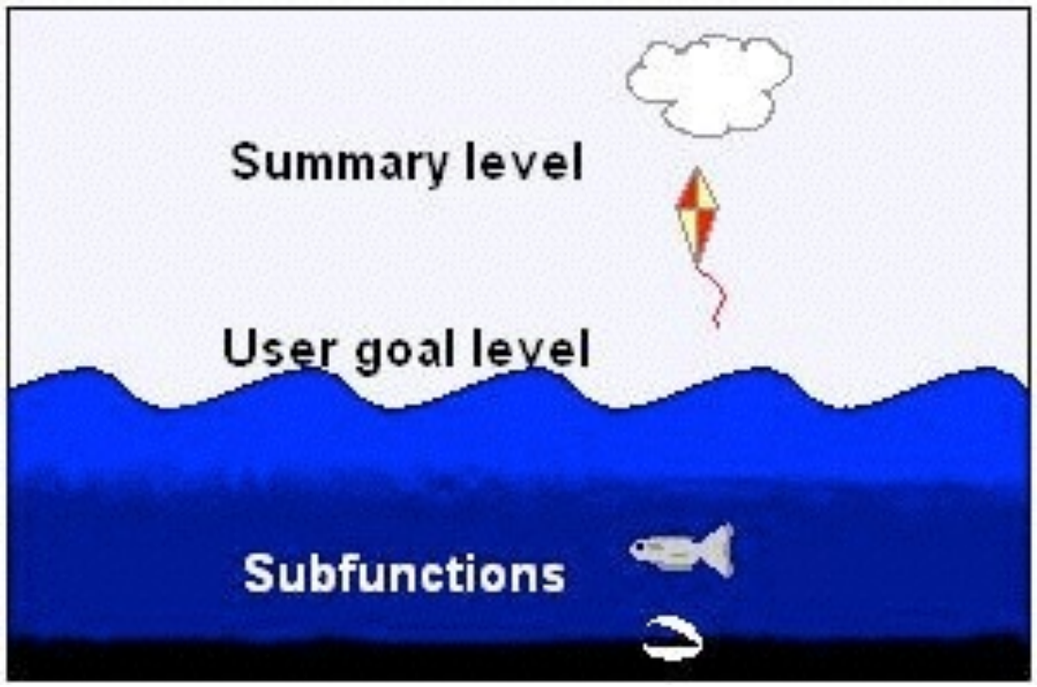
- facile à comprendre
- non ambigu
- complet - contient toutes les alternatives
- confirmé par les utilisateurs du système
- mappé vers un prototype d'interface
- ~~un morceau de (pseudo-)code~~
- ~~une liste de fonctions~~

Niveaux use-cases

kite-level

sea-level

fish-level



Business use-case

System use-case

A. Cockburn, Writing Effective Use Cases

Organisation use-cases

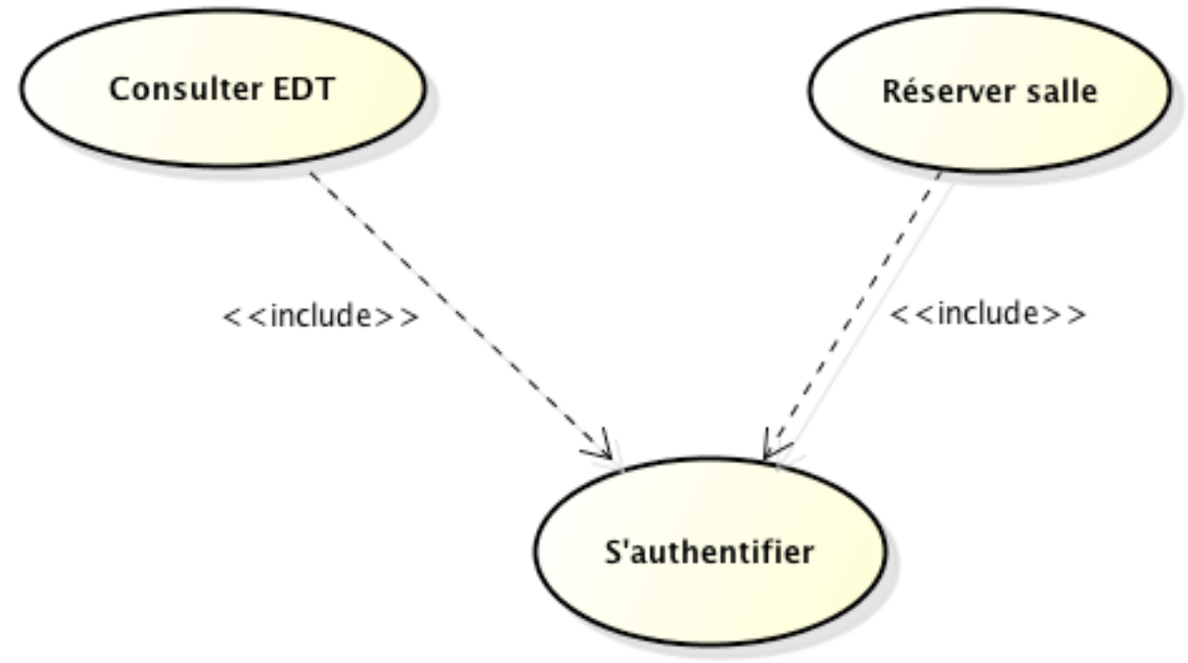
Il est possible (et souhaitable) d'organiser les use-cases

- en regroupant les use-cases en packages, par acteur et/ou par use-case,
- en utilisant les relations d'*inclusion*, d'*extension* et de *généralisation* entre use-cases.

Inclusions

- décrire les points communs contenus dans plusieurs use-cases différents (et éviter la répétition de ces détails dans chacun des use-cases)
- inclus dans d'autres use-cases
- représente l'exécution d'une tâche de plus bas niveau pour accomplir un but aussi de plus bas niveau

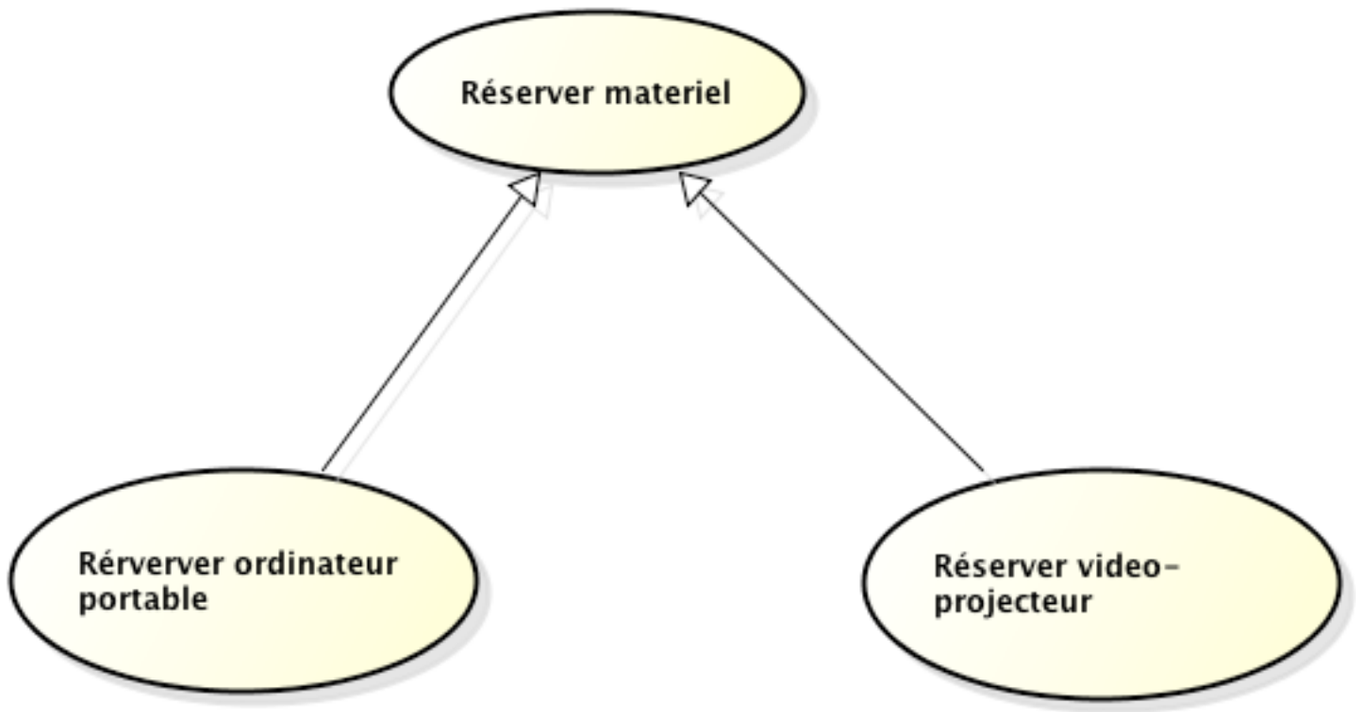
Inclusion


Consulter EDT
Réserver salle
<<include>>
<<include>>
S'authentifier

Généralisations

- Similaire aux super-classes dans les (diagrammes de) classes
- Un use-case généralisé représente plusieurs use-case similaires
- La spécialisation d'un use-case fournit les détails spécifiques relatifs à un de ces use-cases similaires
  - peut remplacer un ou plusieurs enchaînements d'actions du use-case hérité

Généralisations

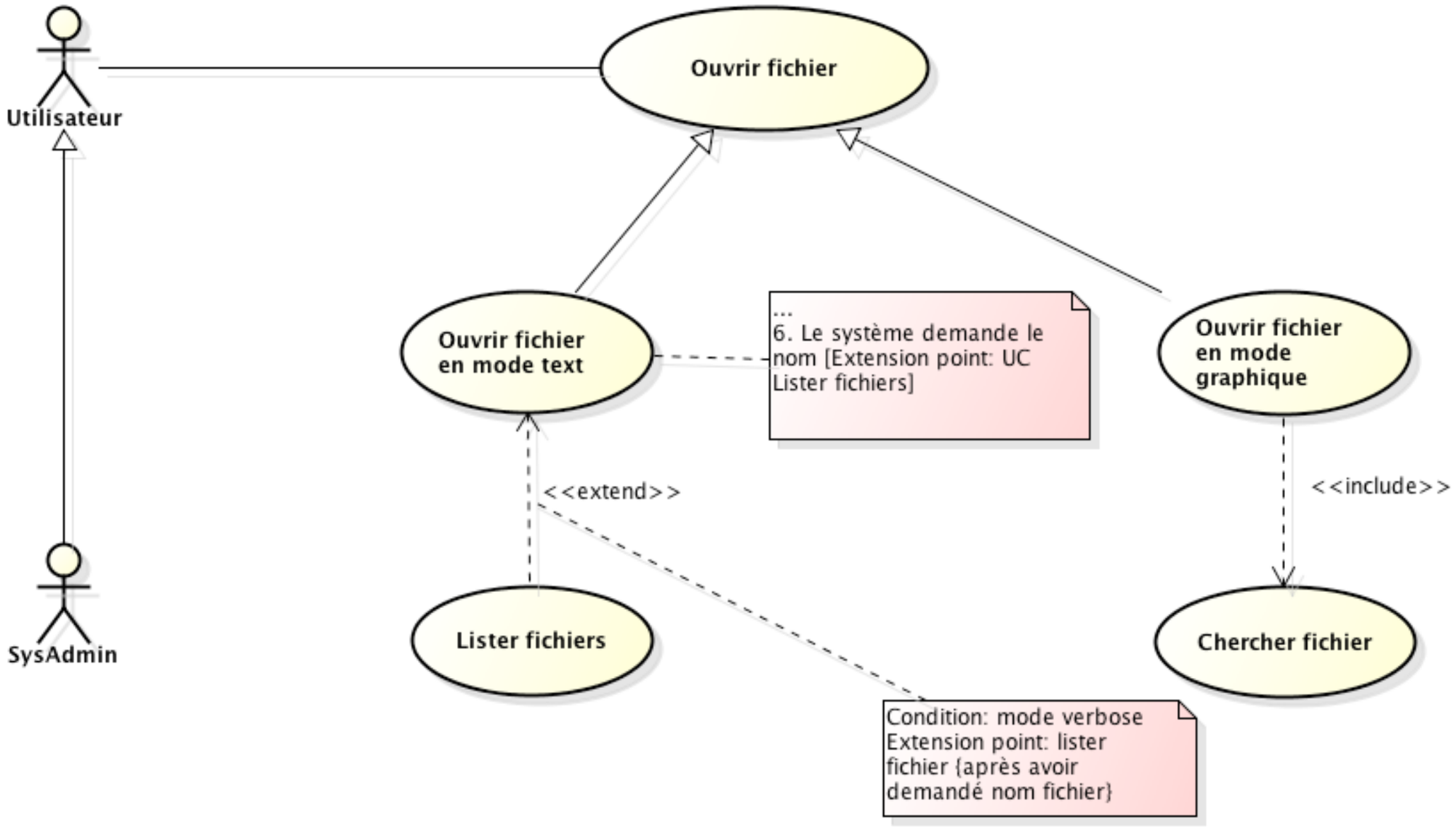


➠ Le cas fils hérite du comportement et de la signification du cas père (il peut le compléter ou le remplacer

Extensions

- **Utilisées afin de spécifier des interactions optionnelles et permettent de modéliser**
  - des partie vues comme optionnelles par l'acteur
  - des flux conditionnels
  - l'insertion de flux conditionnés par l'interaction avec l'acteur
- **En créant un use-case d'extension, la description principale peut demeurer simple**

Exemple

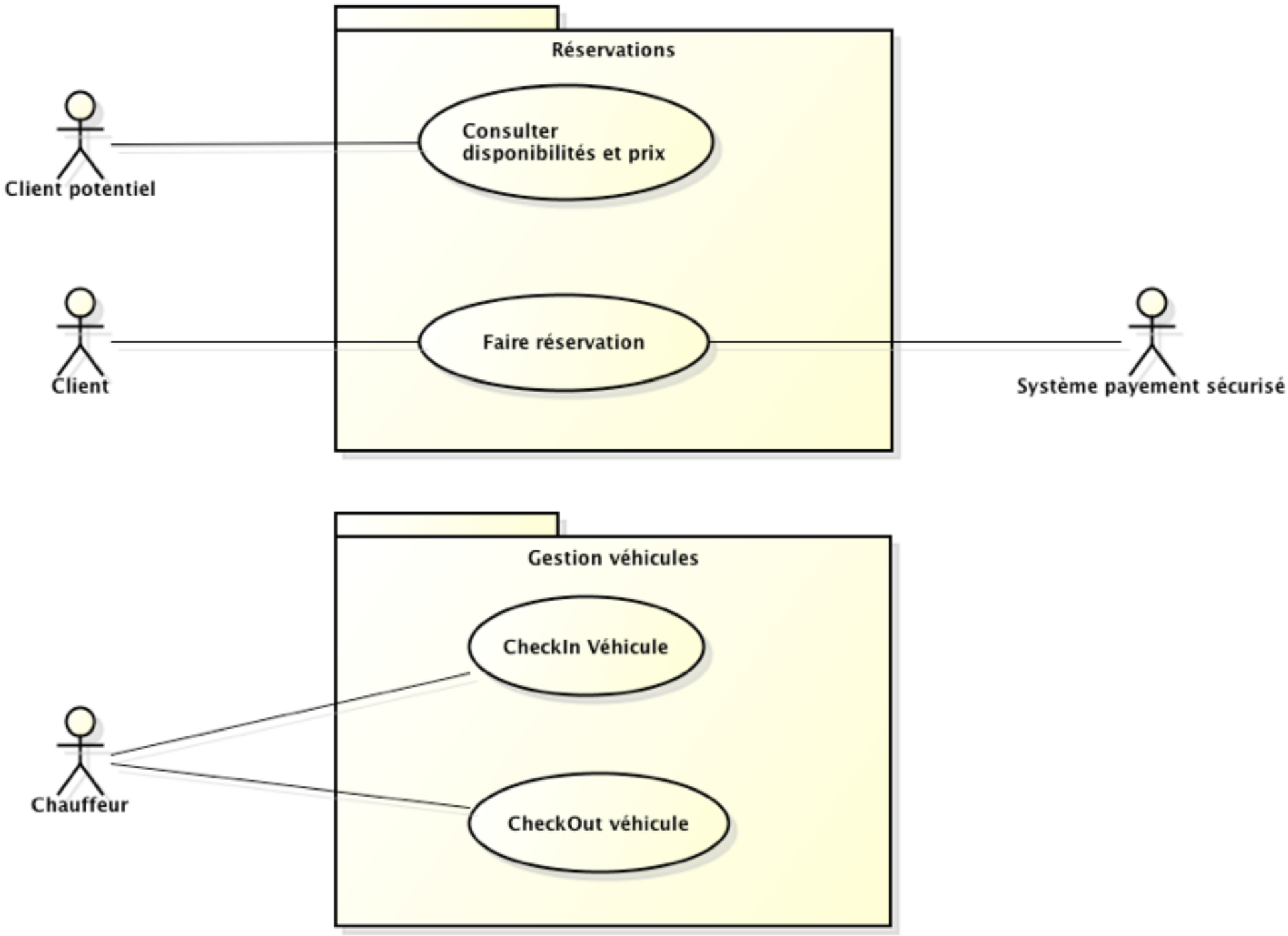

Utilisateur
Ouvrir fichier
Ouvrir fichier en mode text
...
6. Le système demande le nom [Extension point: UC Lister fichiers]
Ouvrir fichier en mode graphique
<<extend>>
<<include>>
SysAdmin
Lister fichiers
Chercher fichier
Condition: mode verbose
Extension point: lister fichier {après avoir demandé nom fichier}

Extends vs. Generalises

- distinguer les variantes d'un use-case
- acteur lié au use-case de base
- acteur exécute le use-case et les extensions

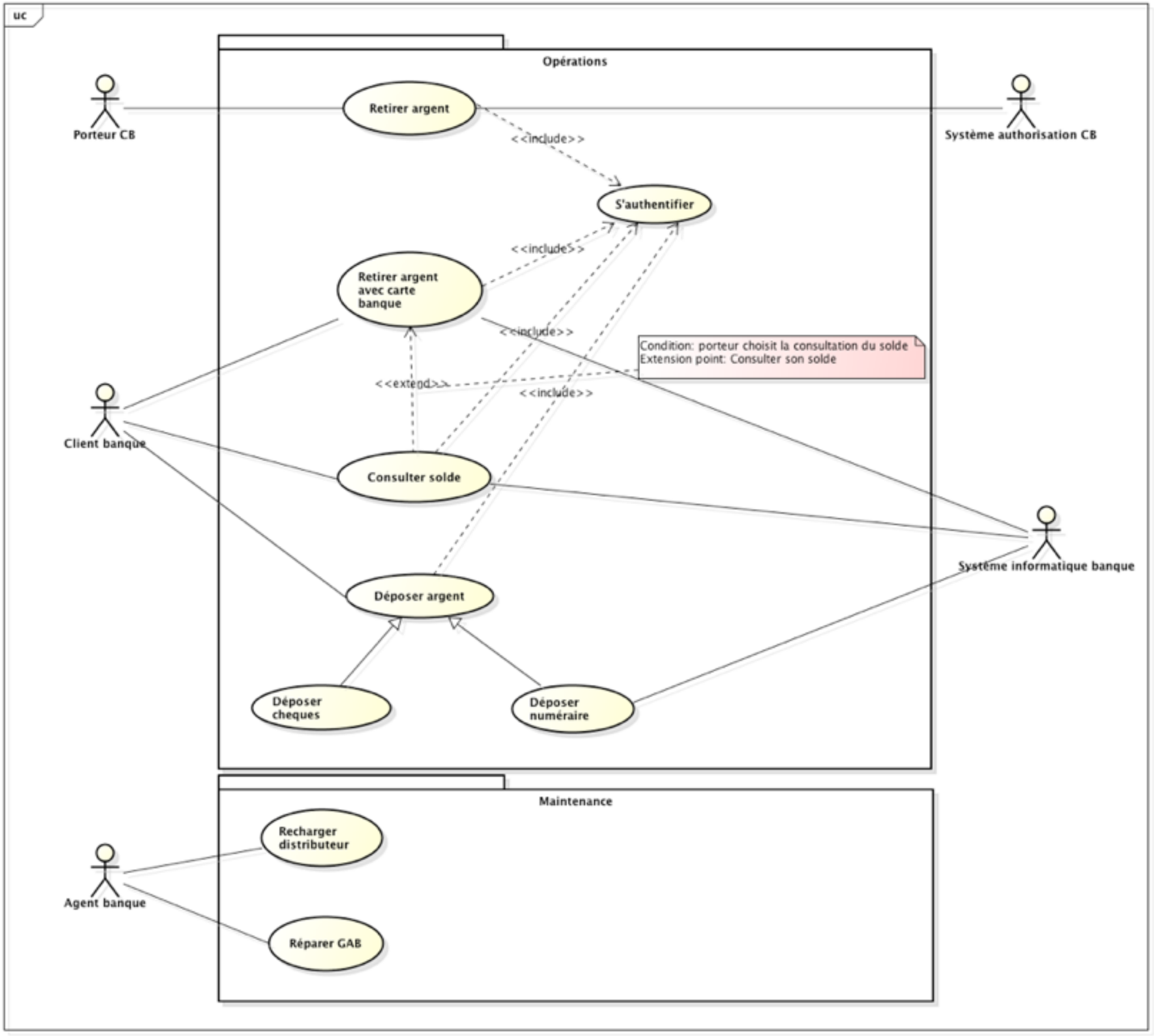
- identifier comportements communs
- pas d'acteur lié au use-case de base (en général)
- acteurs différents pour les généralisations

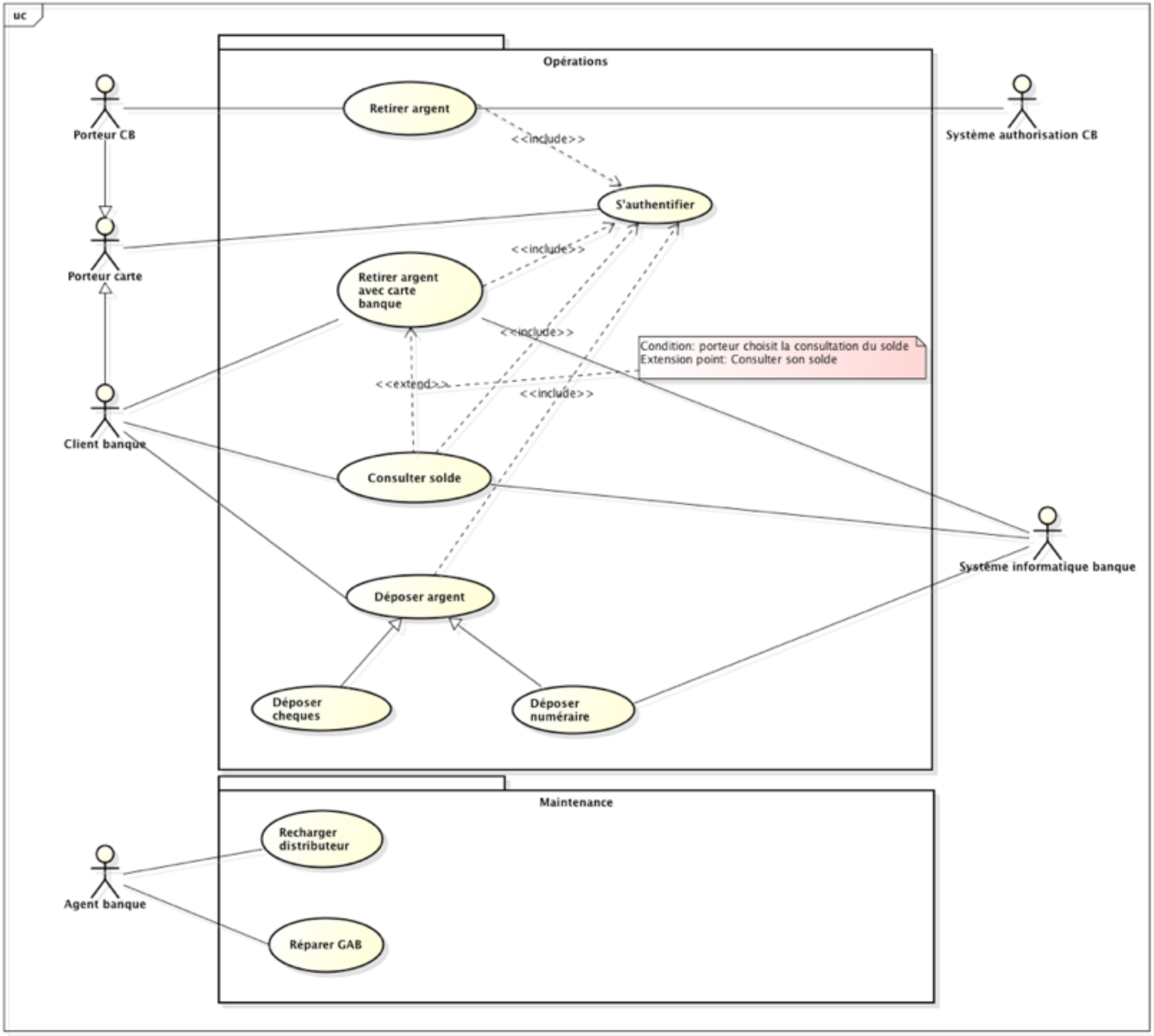
Structurer en package



Modélisation Guichet Automatique de Banque (GAB)

- Ses principales fonctions :
  - distribution d'argent à tout porteur d'une carte de la banque (autorisation d'un certain montant par le Système d'Information de la banque) ou d'une carte bancaire (autorisation à distance par le Système d'Autorisation CB),
  - consultation du solde, dépôt en numéraire et de chèques pour les possesseurs d'une carte de la banque.
- **Toutes les transactions sont sécurisées (code personnel vérifié avec le code enregistré sur la puce de la carte ; la carte est avalée après trois échecs). Il faut parfois recharger le GAB et retirer du numéraire et des chèques.**


uc
Opérations
Porteur CB
Retirer argent
<<include>>
S'authentifier
Système authorisation CB
<<include>>
Retirer argent avec carte banque
<<include>>
Condition: porteur choisit la consultation du solde
Extension point: Consulter son solde
<<extend>>
<<include>>
Client banque
Consulter solde
Déposer argent
Déposer cheques
Déposer numéraire
Système informatique banque
Maintenance
Recharger distributeur
Agent banque
Réparer GAB


uc
Opérations
Porteur CB
Retirer argent
Système authorisation CB
<<include>>
S'authentifier
Porteur carte
<<include>>
Retirer argent avec carte banque
<<include>>
Condition: porteur choisit la consultation du solde
Extension point: Consulter son solde
<<extend>>
<<include>>
Client banque
Consulter solde
Système informatique banque
Déposer argent
Déposer cheques
Déposer numéraire
Maintenance
Recharger distributeur
Agent banque
Réparer GAB

Détailler les use-case

- Un diagramme de use-cases doit être détaillé
  - Détailler la dynamique
  - Décrire le flot d'événements qui le constitue

Description textuelle

- Identifier des scénarios possibles pour un use-case.
  - Scénario nominal
  - Scénario(s) alternatif(s)
  - Scénario(s) d'erreur/exceptionnel(s)
- Aucune normalisation UML

Exemple : GAB

P. Roques

- titre : retirer de l'argent avec une CB
- résumé : ce use-cases permet à un porteur de CB non client de la banque, de retirer de l 'argent, si son crédit hebdomadaire le permet.
- date de création : 02/03/09 de mise à jour : 05/05/09
- version 1.7
- acteurs concernés : porteur de CB, SA CB
- responsables : P. Roques

Exemple :

l'intention de l'acteur
(pas comment)

P. Roques

qui fait quoi

Scénario nominal

1. Le porteur de CB introduit sa carte dans le lecteur du GAB.
2. Le GAB vérifie que la carte introduite est bien une CB.
3. Le GAB demande au porteur de saisir son code.
4. Le porteur de CB saisit son code.
5. Le GAB compare le code saisi avec celui inscrit sur la puce.
6. Le GAB demande une autorisation au SA CB.
7. Le SA CB donne son accord et indique le solde.

Exemple : GAB

P. Roques

Scénario nominal

1. Le porteur de CB introduit sa carte dans le lecteur du GAB.
2. Le GAB vérifie que la carte introduite est bien une CB.
3. Le GAB demande au porteur de saisir son code.
4. Le porteur de CB saisit son code.
5. Le GAB compare le code saisi avec celui inscrit sur la puce.
6. Le GAB demande une autorisation au SA CB.
7. Le SA CB donne son accord et indique le solde.

...

Exemple : GAB

P. Roques

8. Le GAB demande au porteur de CB de saisir le montant.
9. Le porteur de CB saisit le montant désiré du retrait.
10. Le GAB contrôle le montant par rapport au solde.
11. Le GAB demande au porteur de CB s'il veut un ticket.
12. Le Porteur de carte demande un ticket.
13. Le GAB rend sa carte au Porteur de carte.
14. Le porteur de CB reprend sa carte.
15. Le GAB délivre les billets et un ticket.
16. Le Porteur de carte prend les billets et le ticket.

Exemple : GAB

P. Roques

Scénario nominal

1. Le porteur de CB introduit sa carte dans le lecteur du GAB.
2. Le GAB vérifie que la [illegible] e CB.
3. Le GA[illegible]de au [illegible]
4. Le [illegible]isit son code.
5. Le G[illegible] code saisi avec celui inscrit sur la puce.
6. Le GAB demande u[illegible]
7. Le SA CB donne so[illegible]
8. ...

alternatives ?

Pour chaque étape

dysfonctionnements ?

GAB

P. Roques

- Pour chaque étape
  - quelles sont les alternatives ?
  - quelles sont les dysfonctionnements possibles ?

Exemple : GAB

P. Roques

Scénario alternatif

A1 : *code erroné*

Cet enchaînement démarre au point 5 du scénario nominal.

6.Le GAB indique au client que le code est erroné, pour la première ou deuxième fois.

7.Le GAB enregistre l'échec sur la carte.

Le scénario nominal reprend au point 3.

Exemple : GAB

P. Roques

Scénario exceptionnel

E1 : *carte non valide*

Cet enchaînement démarre au point 2 du scénario nominal.

3.Le GAB indique au porteur que la carte n'est pas valide et la confisque.

Le use-cases est terminé.

Exemple : GAB

P. Roques

Contraintes

Temps de réponse : l'interface du GAB doit réagir en moins de 2 secondes ; une transaction nominale de retrait ne doit pas durer plus de 2 minutes.

Disponibilité : le GAB est accessible 7 jours sur 7, 24 heures sur 24 ; il ne doit pas être immobilisé plus d'une heure par semaine par l'opérateur de maintenance.

Intégrité : l'interface du GAB est très robuste.

Confidentialité : l'authentification du code doit être fiable (marge d'erreur tolérée de 1 pour 10 6).

Descriptions alternatives

- colonnes par acteur
- scénarios alternatifs/exceptionnels implicites
- description des relations
- **utilisation de diagrammes**

Description alternative

1.Le porteur de CB introduit sa carte dans le lecteur du GAB.	2.Le GAB vérifie que la carte introduite est bien une CB. 3.Le GAB demande au porteur de saisir son code.
4.Le porteur de CB saisit son code.	5.Le GAB compare le code saisi avec celui inscrit sur la puce. 6.Le GAB demande une autorisation au SA CB.
7.Le SA CB donne son accord et indique le solde.	8. Le GAB demande au porteur de CB de saisir le montant.
9. Le porteur de CB saisit le montant désiré du retrait.	10. Le GAB contrôle le montant par rapport au solde. 11. Le GAB demande au porteur de CB s'il veut un ticket.
12. Le Porteur de carte demande un ticket.	13. Le GAB rend sa carte au Porteur de carte.
14. Le porteur de CB reprend sa carte.	15. Le GAB délivre les billets et un ticket.
16. Le Porteur de carte prend les billets et le ticket.	

Description alternative

Martin Fowler, UML Distilled

Buy a Product
Goal Level: *Sea Level*

représentation possible des « includes »

Main Success Scenario:

1. Customer browses catalog and selects items to buy
2. Customer goes to check out
3. Customer fills in shipping information (address, ...)
4. System presents full pricing information, including shipping.
5. Customer fills in credit card information
6. System authorizes purchase
7. System confirms sale immediately
8. System sends confirming e-mail to customer

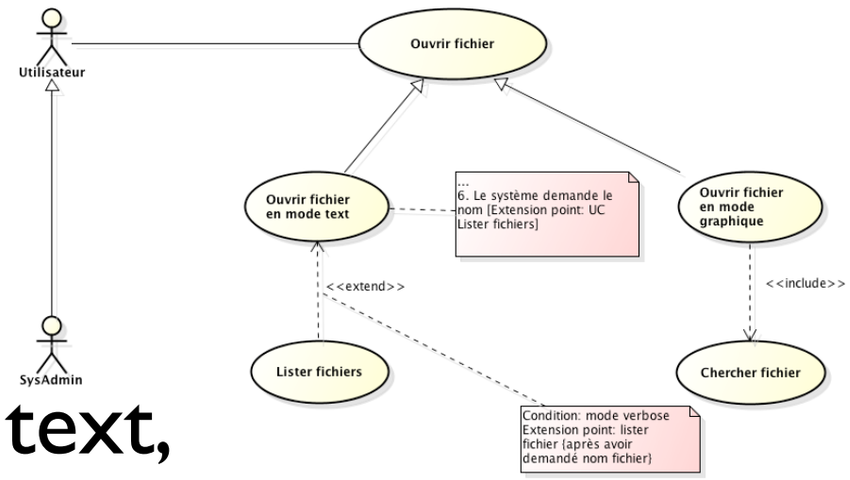
Description alternative

Martin Fowler, UML Distilled

Buy a Product
Goal Level: *Sea Level*

Exceptional:
3a. Customer is a regular customer
.1. System displays current shipping, pricing, and billing information.
.2. Customer may accept or override these defaults, returns to MSS at step 6.
6a. System fails to authorize credit purchase
.1. Customer may reenter credit card information or may cancel

Description relations



UC: **Ouvrir fichier**

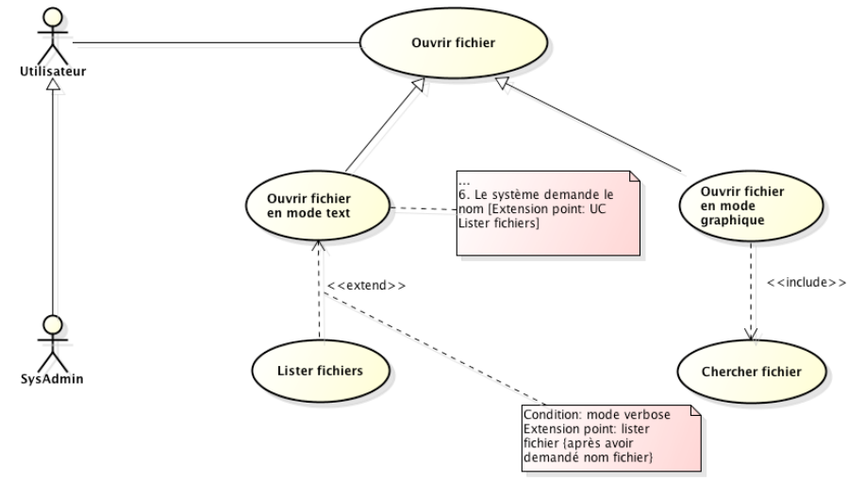
Use-case liés:

Généralisation de: **Ouvrir fichier en mode text,**
Ouvrir fichier en mode graphique

Scénario nominal:

1. L'utilisateur choisit l'option "Open".	2. Le système déclenche le dialogue pour l'ouverture de fichier.
3. L'utilisateur spécifie le nom.	
4. L'utilisateur confirme la sélection.	5. Le système ferme le dialogue pour l'ouverture de fichier.

Description relations



UC: **Ouvrir fichier en mode text**

Use-case liés:

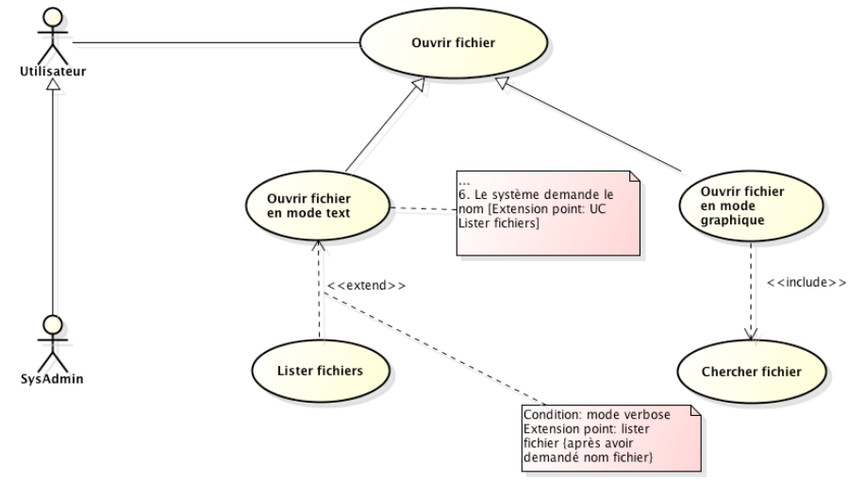
Spécialisation de: **Ouvrir fichier**

Extension: **Lister fichiers**

Scénario nominal:

1. L'utilisateur choisit l'option "Open".	2. Le système déclenche le dialogue pour l'ouverture de fichier.
3a. L'utilisateur sélectionne le champ "Nom". 3b. L'utilisateur indique s'il souhaite le mode verbose [Extension point: UC Lister fichiers] 3c. L'utilisateur saisit le nom du fichier.	
4. L'utilisateur confirme la sélection (en appuyant sur "Open").	5. Le système ferme le dialogue pour l'ouverture de fichier.

Description relations



UC: **Ouvrir fichier en mode graphique**

Use-case liés:

Spécialisation de: **Ouvrir fichier**

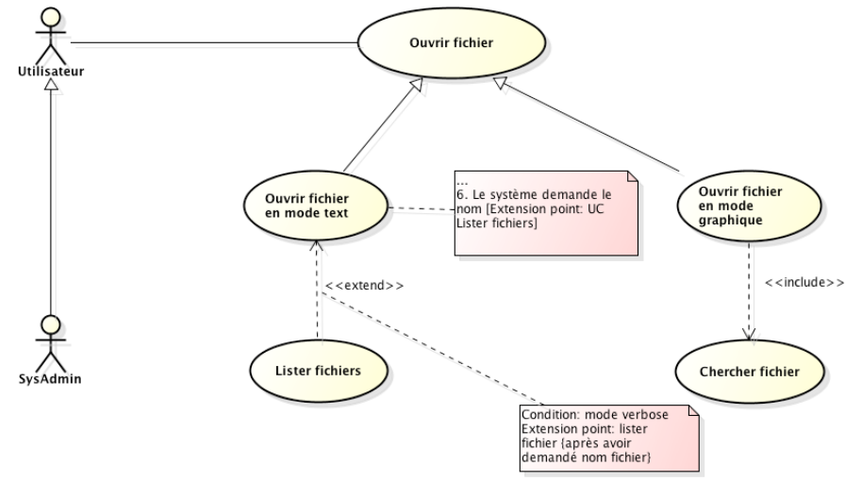
Inclusions: **Chercher fichier**

Scénario nominal:

1. L'utilisateur choisit l'option "Open".	2. Le système déclenche le dialogue pour l'ouverture de fichier.
3a. L'utilisateur sélectionne le champ "Browse". 3b. L'utilisateur cherche le fichier [UC Chercher fichiers]	
4. L'utilisateur confirme la sélection (en appuyant sur "Open").	5. Le système ferme le dialogue pour l'ouverture de fichier.

Description relations

UC: **Lister fichiers**



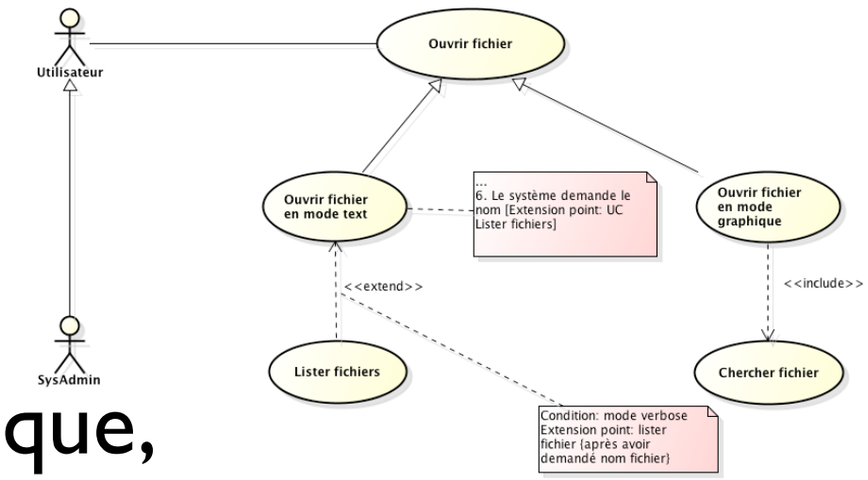
Use-case liés:

Extension de: **Ouvrir fichier en mode text**

Scénario nominal:

1. L'utilisateur choisit les options du listing. 2. L'utilisateur confirme la sélection	3. Le système déclenche la recherche. 4. Le système affiche la liste des fichiers.

Description relations



UC: **Chercher fichier**

Use-case liés:

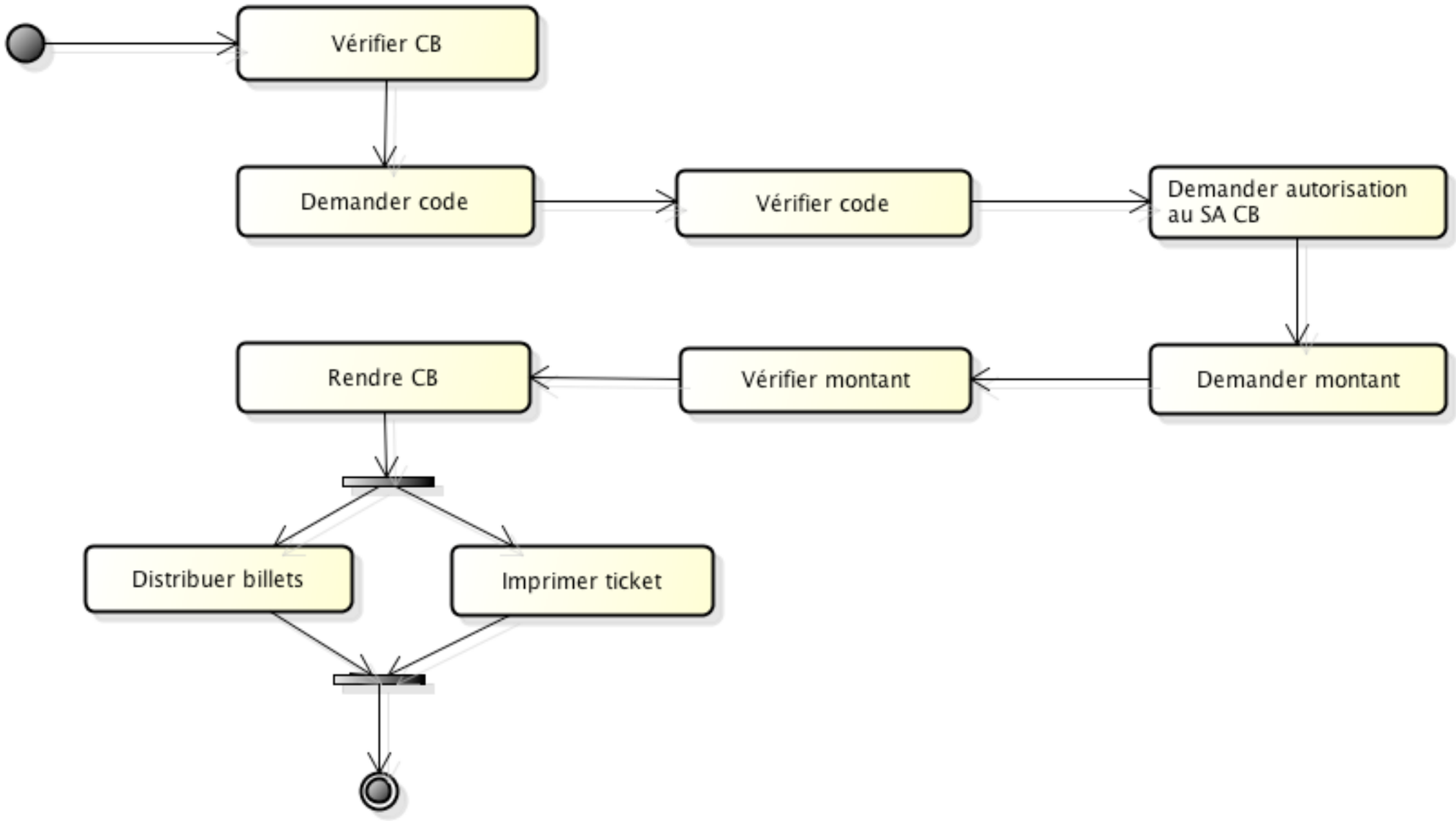
Inclus dans: **Ouvrir fichier en mode graphique,**

...

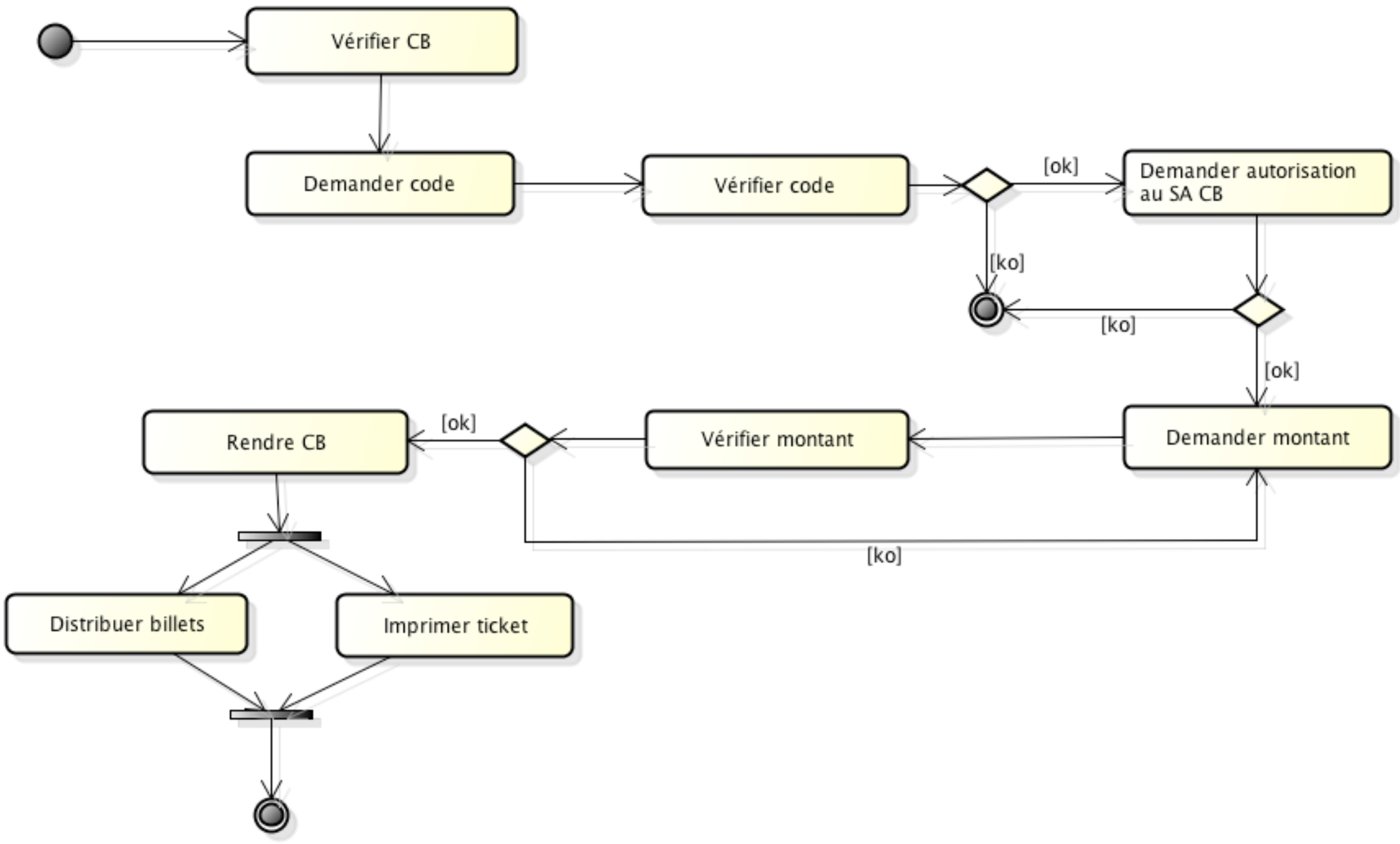
Scénario nominal:

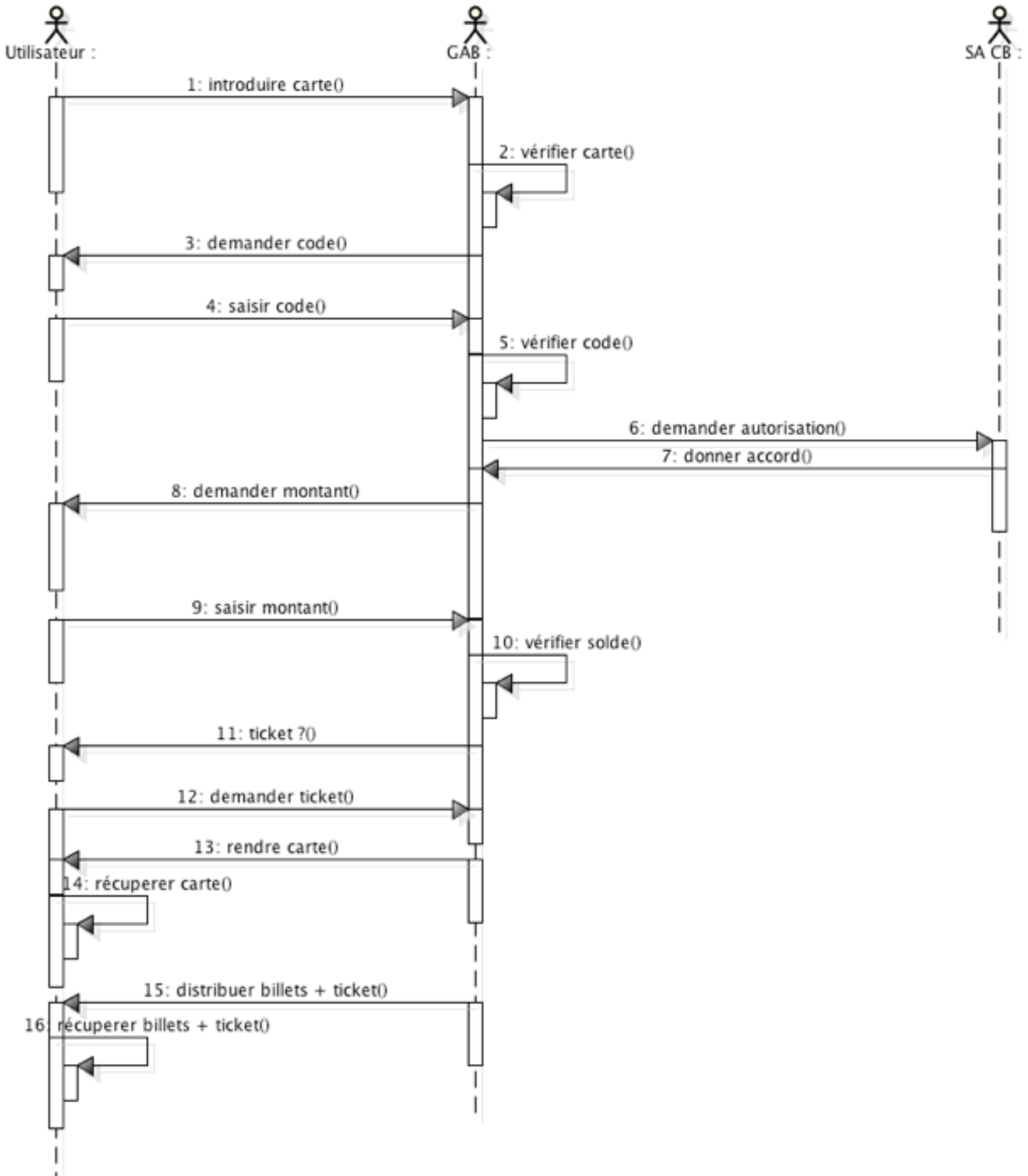
1. L'utilisateur sélectionne un répertoire.	2. Le système affiche le contenu du répertoire.
3. L'utilisateur répète la sélection d'un répertoire tant que le fichier n'est pas affiché.	
4. L'utilisateur sélectionne le fichier.	

Description avec diagrammes



Description avec diagrammes




Utilisateur :
GAB :
SA CB :
1: introduire carte()
2: vérifier carte()
3: demander code()
4: saisir code()
5: vérifier code()
6: demander autorisation()
7: donner accord()
8: demander montant()
9: saisir montant()
10: vérifier solde()
11: ticket ?()
12: demander ticket()
13: rendre carte()
14: récuperer carte()
15: distribuer billets + ticket()
16: récuperer billets + ticket()

Les bénéfices d'une modélisation avec use-cases

- Aide à mieux définir la portée du système
- Permet de définir et de valider les exigences
- Permet de définir les scénarios de tests
- Aide à mieux planifier le développement du logiciel
- Permet de mieux structurer les manuels d'utilisation

Inconvénients

- Ne prennent pas toujours en compte les besoins non fonctionnels
- Apprentissage nécessaire pour les lire
- ...

Les questions à se poser

- Les utilisateurs sont satisfaits de la séquence d'interactions entre l'acteur et le système.
- Il existe une brève description qui donne une image du use-case.
- Les conditions de démarrage et d'arrêt de chaque use-case sont clairement définies.
- Les mêmes termes utilisés dans des use-cases différents ont le même sens.

Les questions à se poser

- Les étapes communes à plusieurs use-cases sont factorisées.
- Les use-cases doivent autant que possible rester indépendants les uns des autres.

➡ on laisse en suspens les problèmes d'interférence, de concurrence et de conflits entre use-cases.

Résumé use-cases

- Le(s) diagramme(s) de use-cases ne représentent qu'une (petite) partie de la description des besoins.
  - *acteurs*
  - *use-cases*
  - *relation entre use-cases*
- La description textuelle est absolument indispensable car le diagramme lui-même est peu instructif.

Questions ?