



Licence Sciences, Technologies, Santé
Mention SPI, parcours Informatique

Introduction au Génie Logiciel
L3 / 175EN002

C5 – Introduction au langage de modélisation UML

Thierry Lemeunier
thierry.lemeunier@univ-lemans.fr
www-lium.univ-lemans.fr/~lemeunie

Plan du cours

- Le langage UML
- Modélisation des fonctionnalités
 - Les acteurs et les cas d'utilisation
 - Exemple de diagramme de cas d'utilisation
- Modélisation des interactions système
 - Les scénarios d'utilisation
 - Exemple de diagramme de séquence
- Modélisation du domaine
 - Les concepts métier et leurs relations
 - Exemples de diagramme de classe

Le langage UML

- Principes d' UML (*Unified Modeling Language*)
 - Langage de modélisation orienté objet
 - Langage graphique semi-formel (utilise en partie le langage naturel)
 - UML n'est pas une méthode mais un langage !
 - Langage indépendant de toute méthode et de tout langage de programmation
- Origines :
 - Il est issu de l'unification de différents langages de modélisation orientés objet du début des années 1990
 - Langage proposé par l'OMG (*Object Management Group*) depuis 1997, consortium qui regroupe la plupart des industriels de l'informatique (www.omg.org)
- Utilisations :
 - Il permet de couvrir toutes les phases du processus logiciel
 - Avec UML 2.0, il y a jusqu'à 11 types de diagrammes différents
 - Il est supportés par de nombreux ateliers ou d'outils de génie logiciel

Modélisation des fonctionnalités (1/6)

□ Rappel :

- L'analyse des besoins est une étape du processus logiciel
- Objectifs :
 - Comprendre le métier des utilisateurs
 - Définir précisément toutes les fonctionnalités

□ On utilise les cas d'utilisation (*use case*) d'UML :

- Ils décrivent le comportement global et apparent du logiciel selon un point de vue externe
- Ils permettent d'aider à identifier :
 - Les différentes catégories d'utilisateurs : les acteurs
 - Les fonctions systèmes : les fonctionnalités accessibles aux acteurs
- Intérêts :
 - Notion et formalisme « accessibles » aux non informaticiens
 - Aide les utilisateurs à structurer et préciser leurs besoins
 - Oblige à définir les interactions et les informations échangées

Modélisation des fonctionnalités (2/6)

■ Les acteurs :

■ Un acteur est soit une personne soit une chose

- Une personne ou une chose peuvent jouer plusieurs rôles et plusieurs personnes ou choses peuvent jouer le même rôle

- Exemple :

- Thierry est *Administrateur* le matin et *Comptable* l'après-midi
- Thierry, Paul et Jacques sont les trois *Comptable*

- Une chose :

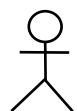
- Les autres systèmes : les systèmes extérieurs qui interagissent avec le système
 - Exemple : un système externe de persistance des données
- Le matériel (dispositifs matériels périphériques) *faisant partie du domaine de l'application* n'est pas modélisé comme acteur
 - Exemple : le clavier, un lecteur de carte à puce, etc.


■ Il joue un rôle vis-à-vis du système et interagit avec le système

- Rôle : l'acteur se comporte selon son métier qui nécessite l'utilisation de certaines fonctionnalités
- Interactions : accès aux fonctionnalités entraînant un échange d'informations entre l'acteur et le système aux cours de ses processus métiers

■ Représentations :

- Le « stickman » nommé
- Le nom de l'acteur doit indiquer son rôle
- Donner une description du rôle de chaque acteur (ses responsabilités vis-à-vis du système)


Comptable


Un acteur humain

« actor »
Un acteur chose

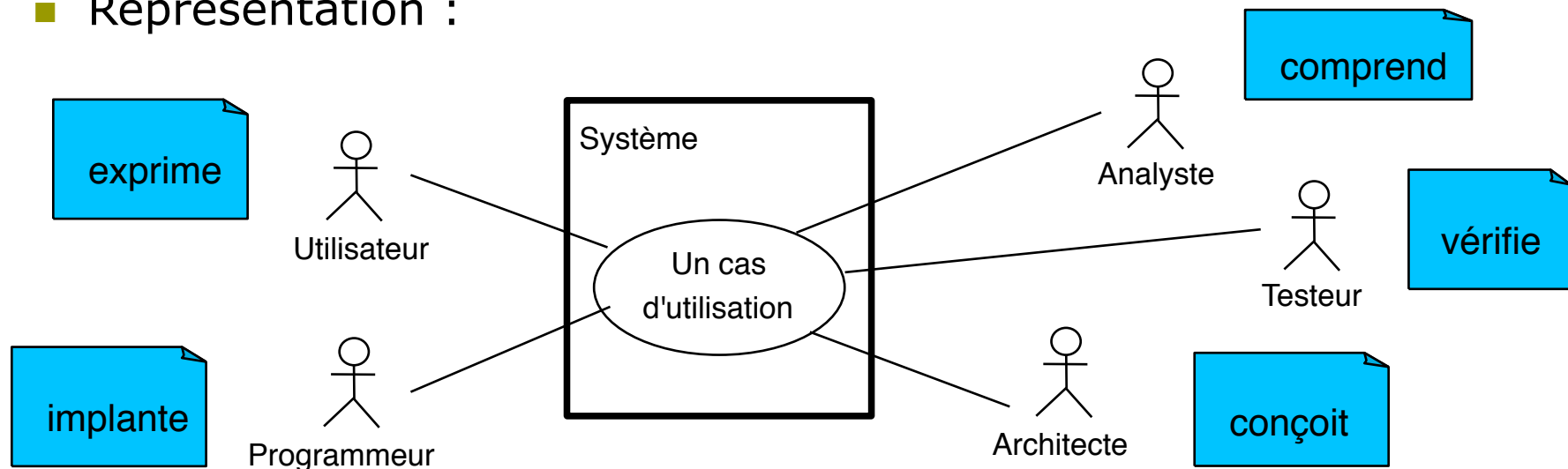
Modélisation des fonctionnalités (3/6)

□ Les fonctions systèmes :

■ Principes :

- Un cas d'utilisation modélise un dialogue entre l'acteur et le système menant à la réalisation d'une fonctionnalité
- La réalisation d'un cas d'utilisation s'effectue par un échange de messages entre le système et les acteurs impliqués dans le cas : c'est un scénario d'interaction
- Il y a généralement plusieurs scénarios possibles (plusieurs chaînes d'interactions) par cas d'utilisation représentables par des diagrammes de séquence
- L'ensemble des cas d'utilisation représente toutes les façons d'utiliser le système

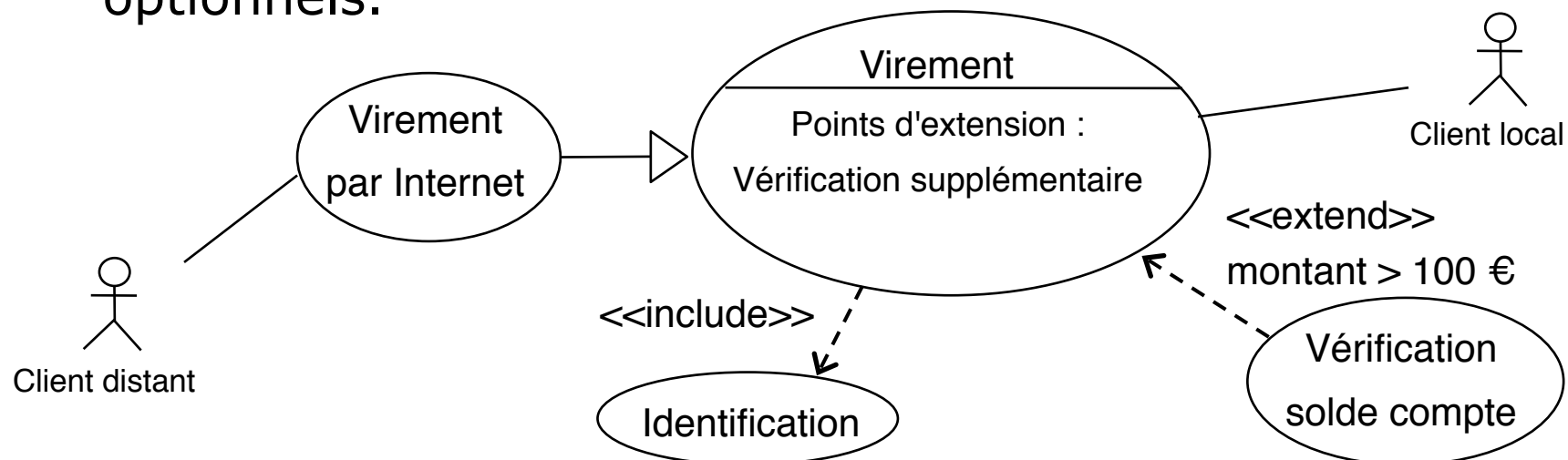
■ Représentation :



Modélisation des fonctionnalités (4/6)

□ Les relations entre cas d'utilisation :

- La relation de **généralisation** : le cas d'utilisation enfant est une spécialisation du cas d'utilisation parent.
- La relation **d'inclusion** : le cas d'utilisation source nécessite le comportement décrit par le cas d'utilisation destination. Cette relation permet de définir des comportements partageables en modélisant les cas d'utilisation communs.
- La relation **d'extension** : le cas d'utilisation source ajoute son comportement au cas d'utilisation destination. L'extension peut être soumise à une condition. Cette relation permet de définir des cas d'utilisation exceptionnels ou optionnels.

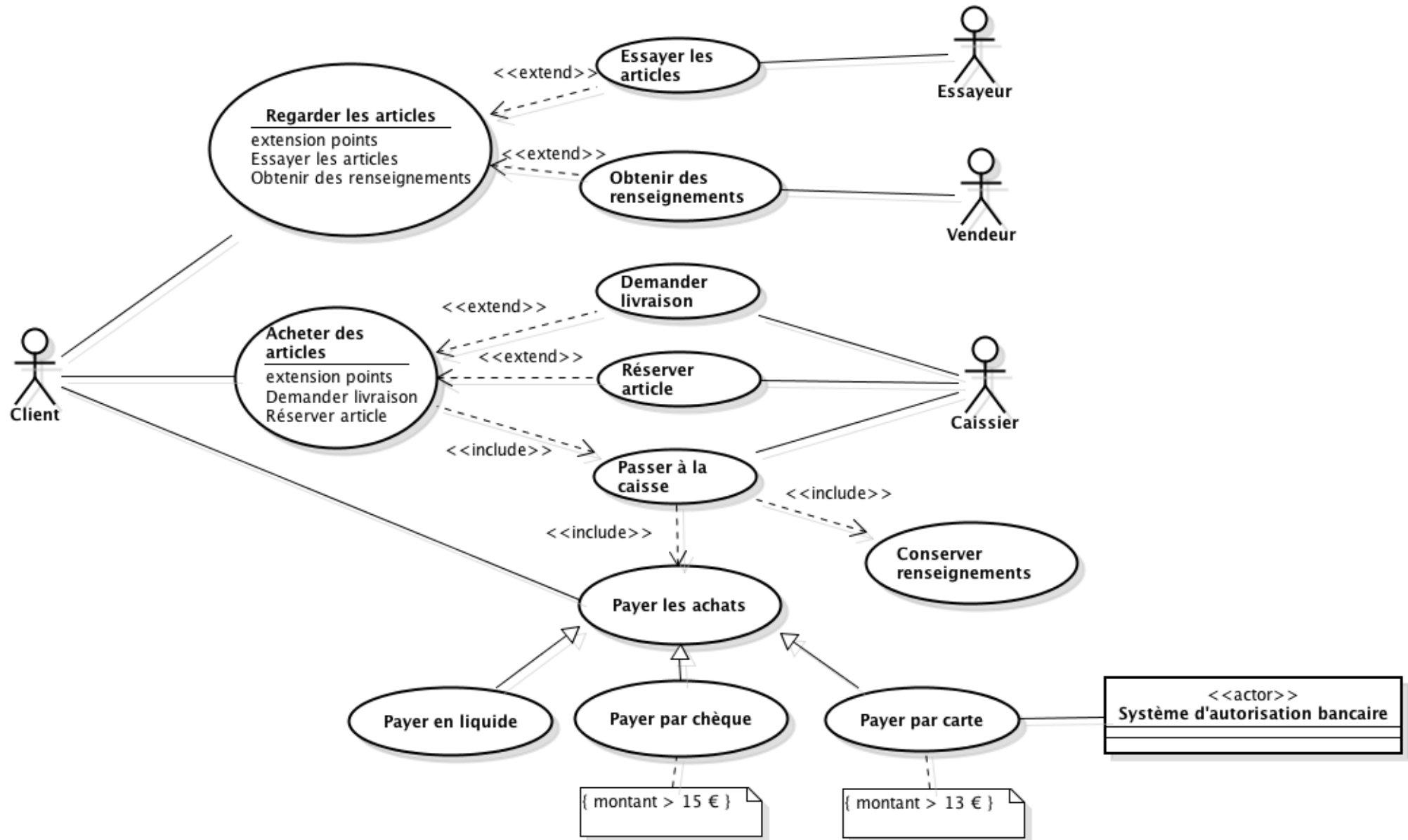


Modélisation des fonctionnalités (5/6)

□ Exemple du **processus de vente** :

- Dans la plupart des magasins, le processus de vente est le suivant : le client entre dans le magasin, passe dans les rayons, demande éventuellement des renseignements ou procède à des essais, prend des articles (ou les réserve si le stock est insuffisant), passe à la caisse où il règle ses achats.
- Il peut bénéficier d'une réduction ou d'un avoir.
- Il peut régler en liquide, par chèque (pour un montant supérieur à 15 €) ou par carte (pour un montant supérieur à 13 €).
- Aucune référence n'est attribuée au client, même si des informations sont conservées en cas de réclamation.
- Une livraison est possible pour les achats encombrants.

Modélisation des fonctionnalités (6/6)



Modélisation des interactions système (1/5)

□ Rappel :

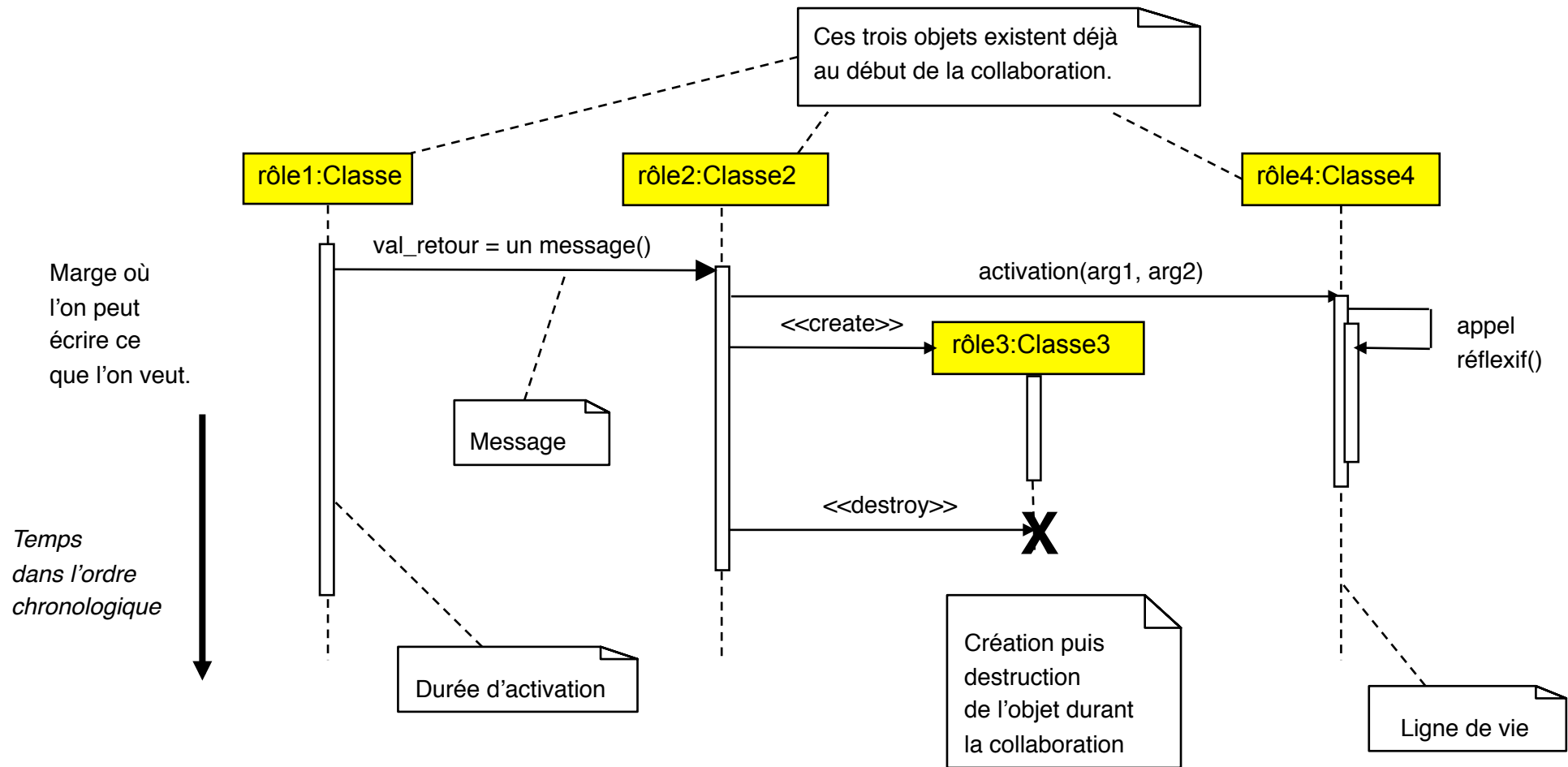
- Il y a plusieurs scénarios possibles par cas d'utilisation
- L'ensemble des scénarios modélisent toutes les interactions

□ On utilise les diagrammes de séquence d'UML :

- Un diagramme de séquence modélise une collaboration :
 - Collaboration : échanges de messages entre **objets** permettant de réaliser une fonction système
 - Une collaboration peut traverser différents modules de l'architecture du logiciel, par exemple, depuis un module d'IHM jusqu'à un module technique
- Notion de message :
 - Un message est la représentation d'une interaction entre objets
 - Il transporte des informations et a pour but de déclencher une activité
- Notion d'interaction système :
 - Interactions entre les acteurs et le système
 - Le système est vu comme une « boîte noire »
 - En analyse, les acteurs et le système peuvent manipuler des instances de concepts métier

Modélisation des interactions système (2/5)

□ Représentation générale :



Modélisation des interactions système (3/5)

□ Un message suit la syntaxe suivante :

[séquence ':'] [résultat '='] nom_message ['(' arguments ')']

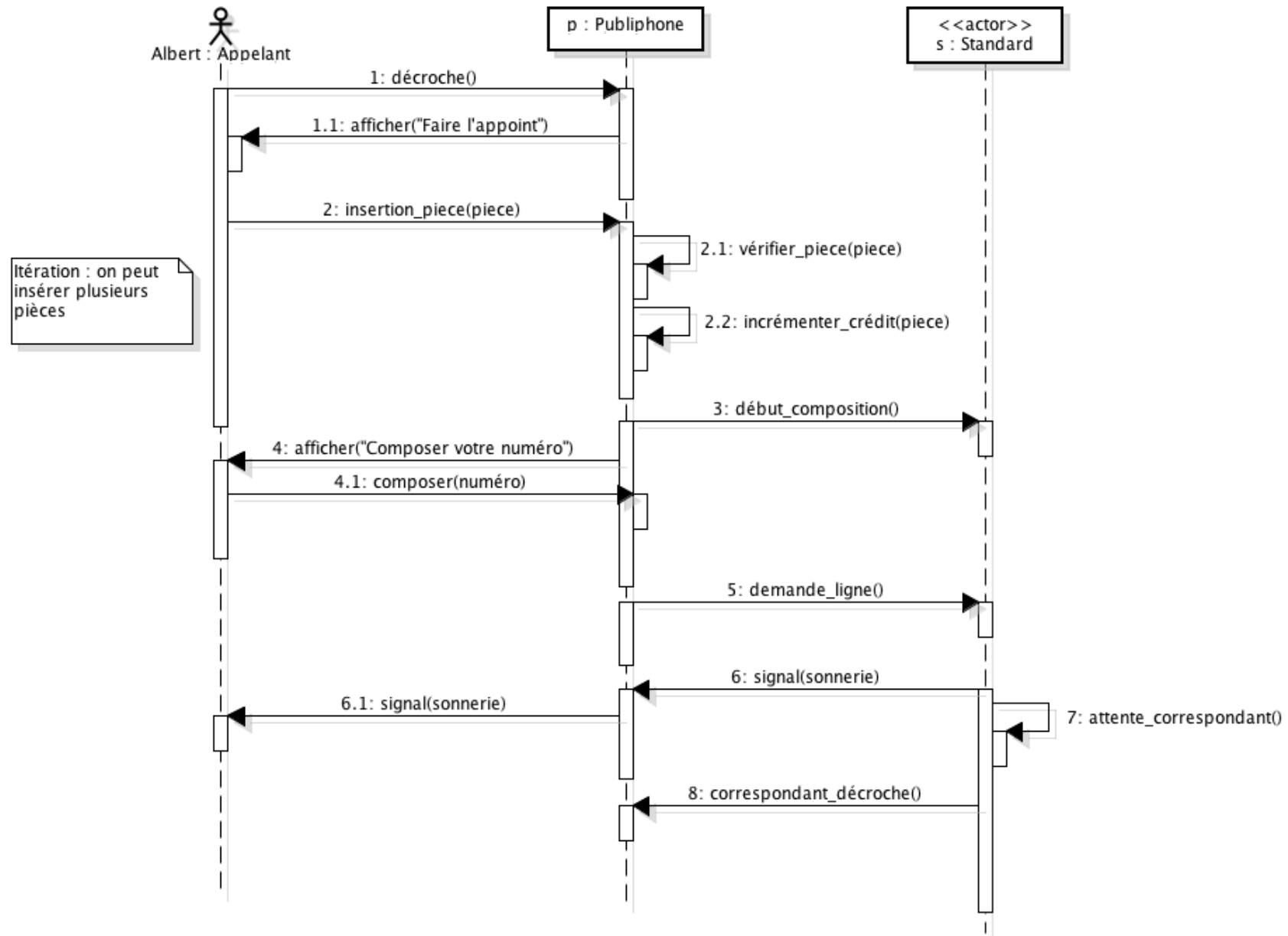
□ La séquence (optionnelle) :

- Nombre qui indique le niveau d'emboîtement de l'envoi des messages
- Syntaxe : suite de termes séparés par des points
séquence ::= rang [récurrence]
rang ::= entier ['.' rang]
récurrence ::= '*' '[' clause d'itération ']'
ou
récurrence ::= '[' clause de condition ']'

Modélisation des interactions système (4/5)

- Exemple du publiphone :
 - L'utilisateur peut payer par carte ou par pièces.
 - Le prix minimal d'une communication interurbaine est de 0,35 euro.
 - Après l'introduction de la monnaie, l'utilisateur a 2 minutes pour composer son numéro (ce délai est décompté par le standard).
 - La ligne peut être occupée ou libre.
 - Le correspondant peut raccrocher le premier.
 - Le publiphone consomme de l'argent dès que l'appelé décroche et à chaque unité de temps (UT) générée par le standard.
 - On peut ajouter des pièces à tout moment.
 - Lors du raccrochage, le solde de monnaie est rendu.

Modélisation des interactions système (5/5)



Modélisation du domaine (1 / 9)

□ Rappel :

- L'analyse des besoins implique de comprendre le métier des utilisateurs :
 - Identifier les concepts métiers
 - Exemple : **les fiches de paiement**
 - Identifier les relations entre les concepts métiers
 - Exemple : chaque fiche de paiement **concerne** un employé
 - Identifier les processus métiers (utilise plusieurs fonctionnalités)
 - Exemple : chaque fin de mois, le comptable **édite** les fiches de paiement puis les **envoie par publipostage**

□ Les processus métiers sont modélisés par diagrammes de séquences ou des diagrammes d'activités (non vu en cours !)

□ Les concepts et les relations entre concepts sont modélisés par les diagrammes de classe d'UML :

- Les concepts métiers sont représentés par des classes
- Les relations entre concepts sont représentés par des associations

Modélisation du domaine (2/9)

- Une classe est représentée par un rectangle et comprend plusieurs compartiments :
 - Un nom obligatoire : identifie de façon unique le concept
 - Une suite optionnelle d'attributs : les caractéristiques du concept
 - Une suite optionnelle d'opérations : les comportements du concept

Nom de la classe

Nom de la classe

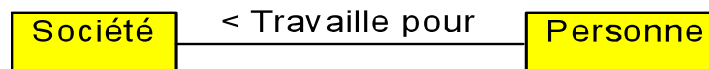
Nom de la classe
attribut1
attribut2
opération1()
opération2()

Fiche de paiement
montant net
montant brut
editer()
publiposter()

- Les types d'associations :
 - L'association simple : lien métier entre deux concepts
 - L'agrégation et la composition : association complexe du type « tout-partie »
 - L'héritage : relation de généralisation/spécialisation entre deux concepts

Modélisation du domaine (3/9)

- Nommage des associations, des agrégations et des compositions :
 - Cela permet de préciser la sémantique du lien entre les classes de l'association
 - Par convention, on utilise une forme verbale



- Multiplicité des associations, des agrégations et des compositions :
 - Les extrémités peuvent porter une indication de **multiplicité**.
 - La multiplicité indique le nombre d'objets que la classe source peut associer à la classe destination.

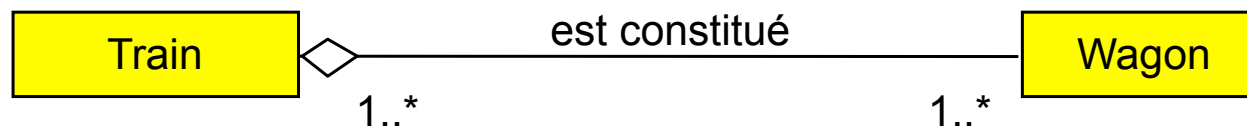


- UML définit les multiplicités suivantes :
 - 1 : un et un seul (*multiplicité par défaut*)
 - 0..1 : zéro ou 1
 - N : exactement N (*entier naturel*)
 - M..N : de M à N (*M et N entiers naturels et M < N*)
 - * : zéro ou plus
 - 0..* : zéro ou plus
 - 1..* : au moins 1

Modélisation du domaine (4/9)

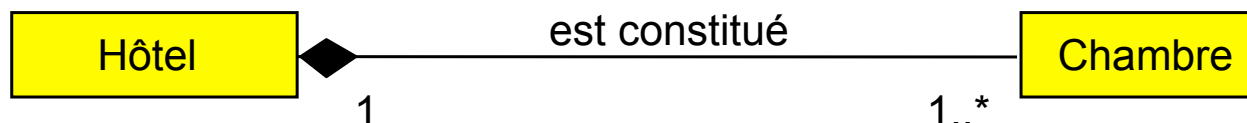
□ Agrégation et composition :

- Association complexe représentant une relation tout-partie :
 - Le tout est constitué de parties
 - Le tout ne peut pas exister sans ses parties
- L'agrégation
 - L'agrégat (le tout) est constitué d'agrégés (les parties)
 - Les parties sont partageables ou échangeables
 - Exemple : un train (le *tout*) est constitué de wagons (les *parties*), mais ces wagons peuvent être utilisés pour former d'autres trains à un autre moment



■ La composition

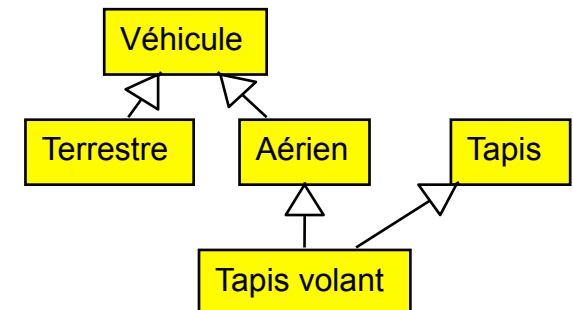
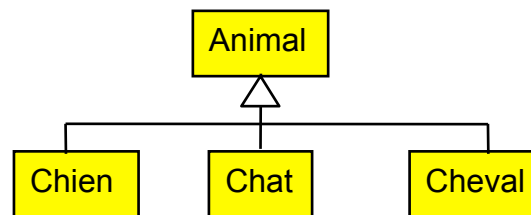
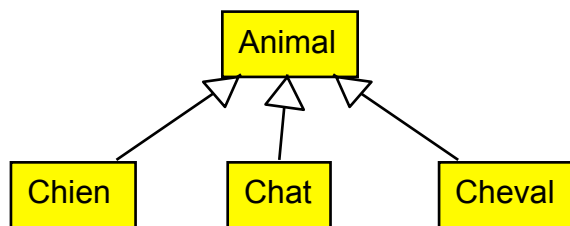
- Le composite (le tout) est constitué de composants (les parties)
- Les parties ne sont pas partageables ou échangeables
- Exemple : les chambres (les *parties*) d'un hôtel (le *tout*) ne sont pas partageables ou échangeables entre plusieurs hôtels



Modélisation du domaine (5/9)

□ Association d'héritage :

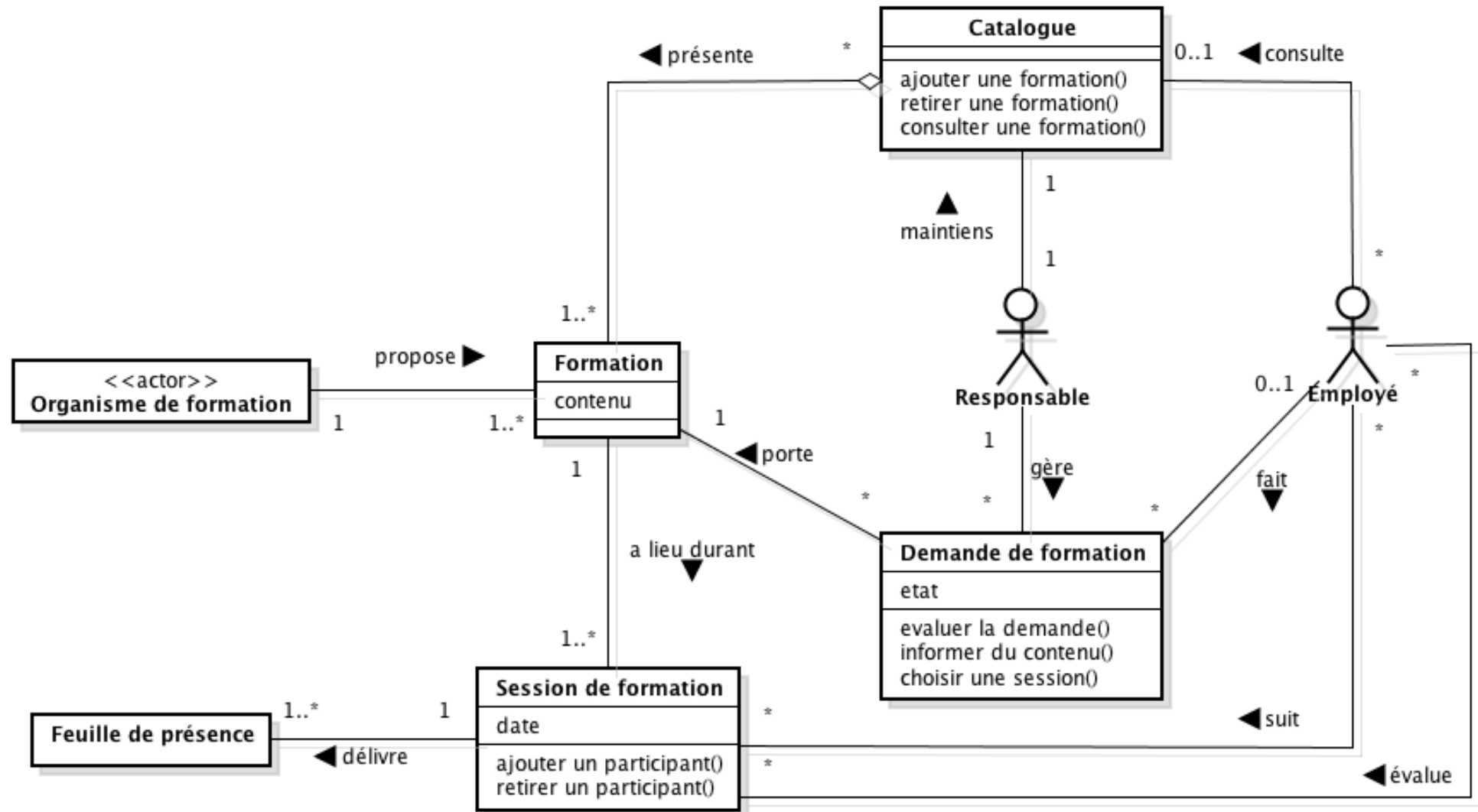
- Désigne une relation de classification du type « est une sorte de » entre un élément général (classe mère) et un élément spécifique (classe fille)
- Les classes filles héritent des attributs, des opérations, des relations et définies dans la classe mère
- La classe fille peut redéfinir des attributs ou des opérations hérités
- La classe fille peut avoir des attributs ou des opérations supplémentaires
- Deux types d'héritage : simple ou multiple
 - Héritage simple : la classe fille hérite d'une classe mère
 - Héritage multiple : la classe fille hérite de plusieurs classes mères
- Représentation :



Modélisation du domaine (6/9)

- ❑ Exemple du service formation d'une grande entreprise :
 - Le processus de formation est initialisé quand le responsable formation reçoit une demande de formation d'un employé.
 - Cet employé peut éventuellement consulter le catalogue des formations offertes par les organismes agréés par l'entreprise.
 - Cette demande est instruite par le responsable qui transmet son accord ou son refus à l'employé.
 - En cas d'accord, le responsable cherche la formation adéquate dans le catalogue des formations agréées qu'il tient à jour. Il informe l'employé du contenu de la formation et lui soumet la liste des prochaines sessions prévues.
 - Lorsque l'employé a fait son choix il inscrit l'employé à la session retenue auprès de l'organisme de formation concerné.
 - En cas d'empêchement l'employé doit avertir au plus vite le responsable formation pour que celui-ci demande l'annulation de l'inscription.
 - A la fin de la formation l'employé évalue la formation et transmet un document attestant sa présence. Le responsable formation contrôle la facture envoyée par l'organisme de formation.

Modélisation du domaine (7/9)



Modélisation du domaine (8/9)

□ Exemple du prêt-à-porter :

- Une entreprise de confection fabrique des jupes et des chemisiers prêts-à-porter à partir de modèles.
- Un modèle est une forme de jupe ou de chemisier créée pour une collection (été ou hiver) qui change tous les ans.
- Une jupe (ou un chemisier) d'un certain modèle est fabriquée dans un tissu référencé et décrit par sa nature principale (coton par exemple) et par sa composition précise (par exemple 80% coton, 20% polyester). Pour un même tissu, plusieurs motifs peuvent exister selon la couleur et le type de dessin.
- Une jupe (ou un chemisier) dans une taille donnée, d'un certain modèle, dans un tissu de motif déterminé est appelée article et identifiée en fabrication par une référence article.
- Un assortiment est constitué d'une jupe + un chemisier de modèles, tailles et tissus éventuellement différents, mais un modèle de jupe ne peut être assorti qu'à un certain nombre de modèles de chemisiers et réciproquement.
- Un assortiment, un modèle de jupe ou de chemisier individuel dans une taille et un motif donnés fournissent une référence catalogue. Chacune d'elles est caractérisée par une image numérique, un texte descriptif, un message publicitaire, un n° de page, une position dans la page et un prix de vente.

Modélisation du domaine (9/9)

