UML

Conclusions

Vue d’un système

- **Statique**
  - **Décrit les états observables du système**
  - **Pris en compte par les données (MCD, NIAM)**
- **Dynamique**
  - **Représente les interactions entre les différents constituants du système**
  - **Pris en compte par les contraintes de temps, d 'exclusion, de synchronisation (MCI, RdP, Automate à états)**
- **Fonctionnelle**
  - **Définit les relations entre les entrées et les sorties du système**
  - **Pris en compte par un diagramme de flux (DFD, SADT)**

Rappel sur les méthodes

- Les méthodes analytiques ou cartésiennes
  - Basées sur le principe de décomposition hiérarchique
  - Exemple de DFD, SADT
  - Permet la compréhension de système complexe mais offre une vision parcellaire des constituants du système
- Les méthodes systémiques
  - Met l'accent sur la communication des constituants du système Exemple de Merise
  - Conserve une vision globale du système sans exclure une approche analytique
- Les méthodes objets
  - Basées sur la notion d 'objet provenant du système réel
  - Intègre dans une même entité les 3 aspects :
    - Statique
    - Dynamique
    - fonctionnel
  - Offre un compromis entre les méthodes analytiques et systémiques

Rappel sur les méthodes

- Une méthode est un processus dont les activités constituent des phases du cycle de développement du logiciel
- Les phases du cycle de développement sont ordonnées pour former un processus définissant la démarche méthodologique
  - Cycle en cascade et cycle en V
  - Cycle en spirale
  - Cycle itératif
- Chaque phase du cycle de développement est caractérisée par:
  - des activités
  - des ressources en entrée et produits en sortie
  - des formalismes supportant les entrées / sorties

Modèles formels

- Un modèle formel permet d'appliquer un procédé automatique
- Le modèle de données
  - Basé sur la théorie des ensembles
  - Offre une procédure de normalisation
  - Modèle rendu opérationnel avec les SGBDR
- Le modèle dynamique
  - Basé sur les graphes d'états du système
  - Offre des procédures de calcul des propriétés du système (état franchissable, réseau vivant, système équivalent, ...)
- Le modèle fonctionnel
  - Basé sur le principe des langages fonctionnels
  - Peut être décrit par un organigramme, un pseudo code, une formule

Cycle itératif

- On divise le problème en sous problèmes
- On applique le cycle complet à chaque sous problème
- On intègre le résultat précédent à chaque itération
- Nécessite de définir l'architecture qui peut être mener avec l'analyse des besoins

Méthode OOA (Coad et Yourdon)

- Le sujet permet de regrouper certains éléments du modèle
- Résumé de la démarche
  - Analyse
    - Identifier les classes d 'objets
    - Identifier les structures de généralisation / spécialisation, agrégation / composition
    - Identifier les sujets regroupant des éléments du modèle
    - Définir les attributs et opérations
    - Décrire les opérations en montrant les interactions entre les objets
  - Conception
    - prototypage
    - prise en compte des profils utilisateurs et des détails d 'interaction
    - Prise en compte du temps réel
    - Prise en compte des systèmes de gestion de données

Méthode OOSE d’Ivar Jacobson

- Introduit la notion de Use Case
- Repose sur 5 modèles et 3 types d'objets :
  - Besoins, analyse, conception, implémentation et test
  - Entités, contrôles et Interfaces
- Résumé de la démarche
  - Définir les Use Cases
  - Créer un modèle d 'analyse objet
  - Créer un modèle de conception objet
  - Créer un modèle d 'interaction pour chaque Use Case
  - Créer un diagramme d 'états / transitions pour chaque cycle de vie d 'objet

Méthode OMT de Rumbaugh

- Repose sur 3 modèles et 4 phases :
  - Modèles statique, dynamique et fonctionnel
  - Analyse, conception du système, conception objet et implantation
- Résumé de la démarche :
  - Créer un modèle de classes
  - Identifier les attributs et opérations
  - Créer un diagramme d 'états de Harel pour représenter le comportement dynamique du système
  - Créer un modèle de processus de style Yourdon/DeMarco pour représenter les transformations de données
  - Créer un modèle de tâche pour représenter les tâches concurrentes et les processeurs
  - Créer un diagramme d 'interaction des objets pour détailler les mécanismes de conception

Methode BOOCH

- introduit les notions de module, processus, des types de message et d'architecture du système
- Distingue 2 niveaux :
  - Logique : Diagramme de classes et d 'instances
  - Physique : Diagramme d 'état transition et de temps
- Résumé de la démarche :
  - Créer un modèle de classes
  - Identifier les attributs et opérations
  - Créer un diagramme d'états de Harel pour représenter le comportement dynamique du système
  - Créer un diagramme d'interaction des objets pour détailler les mécanismes de conception
  - Créer un scénario pour montrer les relations temporelles entre les objets
  - Créer un diagramme de module pour représenter la conception physique du système
  - Créer un diagramme de tâches montant les processeurs et périphériques

- UML prend le meilleur de chacune des 3 méthodes
  - OOSE (Jacobson) : Use Cases
  - OMT (Rumbaugh) : Analyse
  - Booch : conception Architecture

Qu'est ce qu'UML?

- C'est un langage semi formel qui permet de décrire tous les aspects d'une application informatique.
- Il est issu de la programmation orientée objet. A ce titre il est surtout adapté aus applications réalisées à l'aide d'un langage OO (C++,Java,Eiffel,Smalltalk..)
- UML n'est pas une méthodologie. Il ne préconise en effet aucune démarche.

Les principaux diagrammes UML

- Statiques :
  - Diagrammes de classe (et d'acteurs)
  - Diagrammes d'objet
  - Diagrammes de cas d'utilisation
  - Diagrammes de composants
  - Diagrammes de déploiement
- Dynamiques
  - Diagrammes de séquences
  - Diagrammes de collaboration
  - Diagrammes d'état
  - Diagrammes d'activité

Démarche

- Inception : définit la portée du projet et les processus métiers
- Elaboration : planification du projet, spécification des caractéristiques et des grandes lignes de l'architecture, des ressources et spécificités
- Construction : réalisation du produit en une suite d'itérations incrémentales
- Transition : définition des livrables et formations, livraison du produit aux utilisateurs

Processus	Inception	Elaboration	Construction	Transition
Modélisation métier	XXX	XXX		
Exigences	XX	XXXX		
Analyse et Conception		XXXXX		
Réalisation		XX	XXXX	
Tests	X	X	XX	X
Déploiement			X	XXXX
Gestion configuration		X	XXX	XX
Conduite de projet	XX	XX	XX	XX
Environnement	XX			

INCEPTION

- Définir la vision globale du projet
- Définir les processus métiers impliqués avec le système
- Délimiter les frontières du projet
- Identifier les entités externes qui interagissent avec le système
- Identifier et décrire les principaux cas d 'utilisation et scénarios
- Identifier les risques
- Définir Iles bornes pour chaque incrément

ELABORATION

- Définition des exigences
- Analyse du domaine
- Définition d'un glossaire du domaine
- Création d'une ossature de l'architecture (framework)
- Planification du projet
- Définition des scénarios supportés par chaque incrément
- Eliminer les facteurs de risques les plus élevés

CONSTRUCTION

- Définition des cas d'utilisation restants
- Description des scénarios pour les cas d'utilisation
- Développement d'un modèle objet complet avec les attributs et méthodes
- Détailler les diagrammes de séquence Implémenter les classes
- Créer les scénarios de test basés sur les cas d'utilisation et sur l'architecture

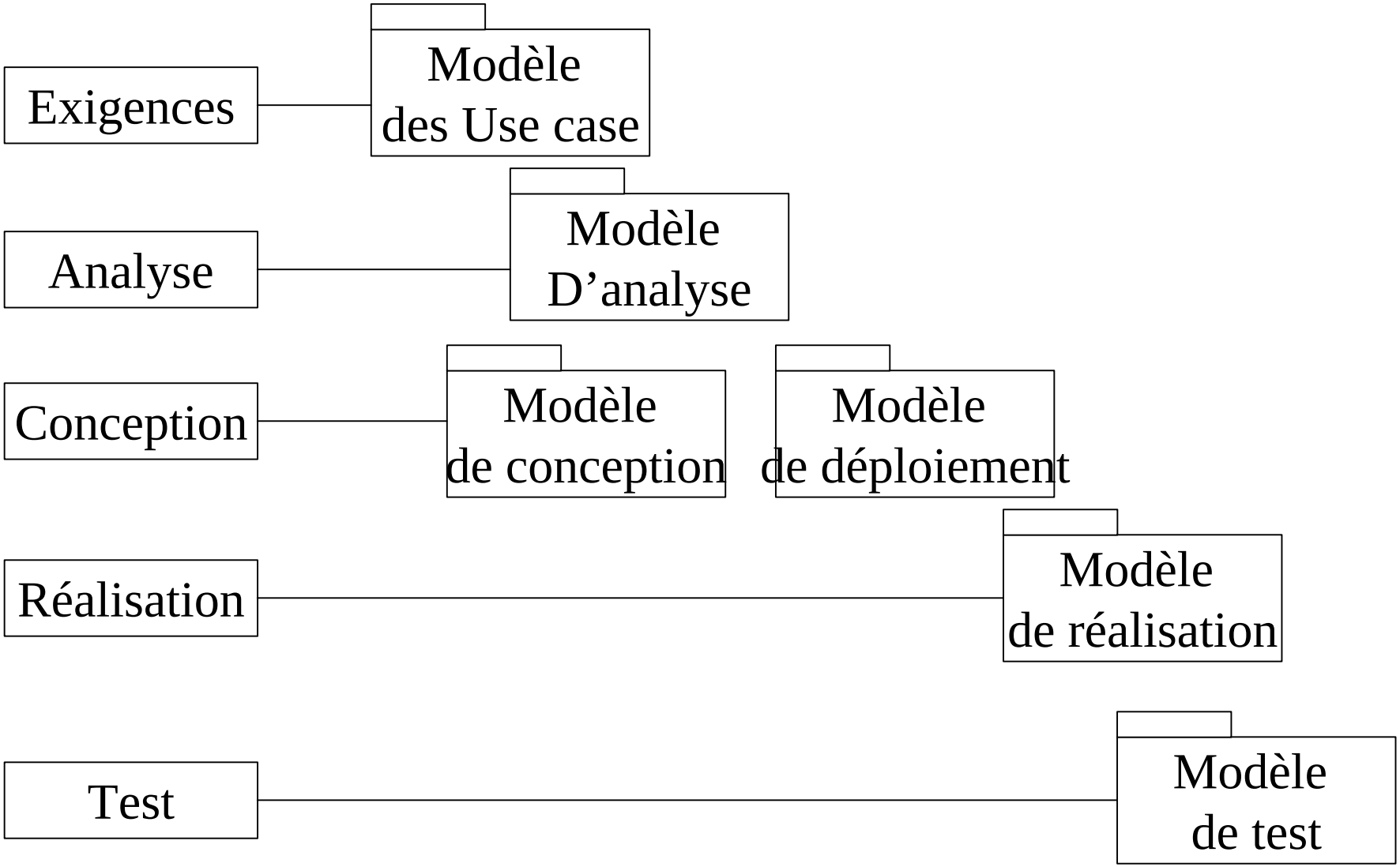
TRANSITION

- Confier le logiciel aux utilisateurs
- Préparer les plans de formation pour les utilisateurs
- Description des livrables
- Mises au point et résoudre besoins mal compris
- Définition de la portée de la prochaine mise à jour

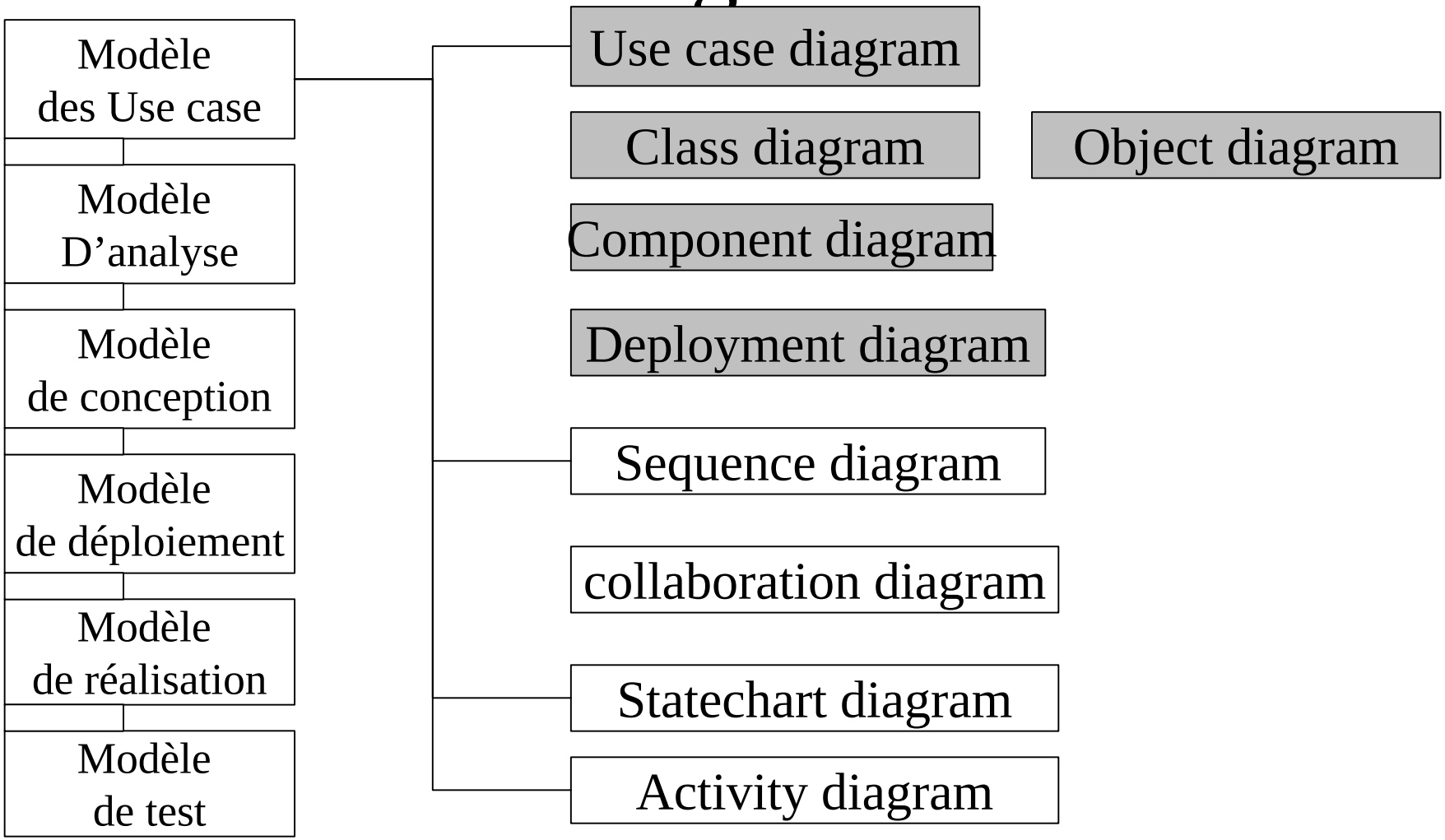
Incrément et Itération

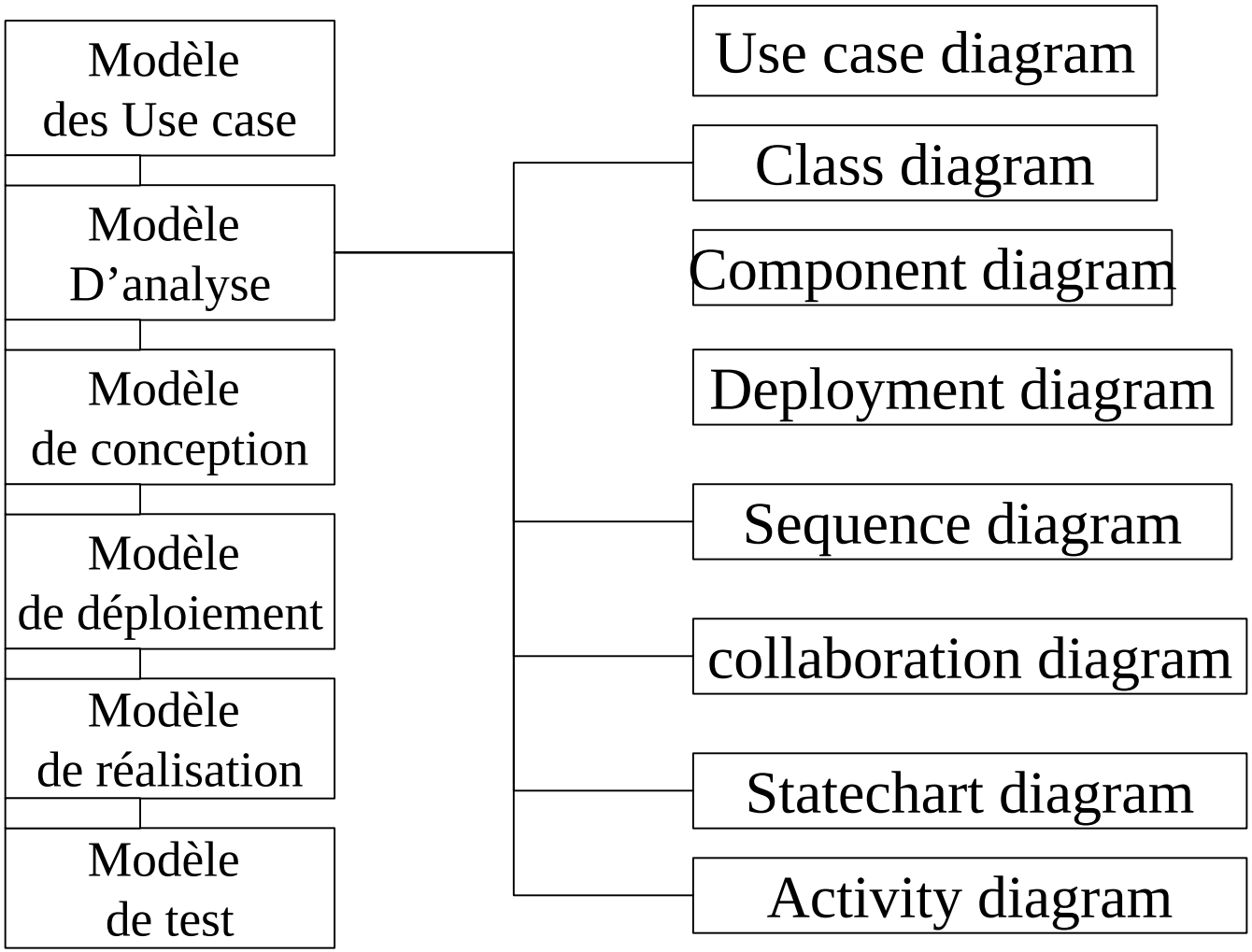
- Chaque étape du processus est divisé en incréments :
  - - Un incrément est un ensemble de cycles de développements conduisant à certaines fonctionnalités compréhensibles et utilisables du logiciel
  - - A la fin d'un incrément, certains scénarios et exigences ont été conçus et réalisés
  - - Chaque incrément comporte plusieurs itérations
- Chaque incrément est subdivisé itérations :
  - - Une itération est un cycle de développement d'un ou plusieurs cas d'utilisation
  - - A la fin d'une itération, plusieurs scénarios ou exigences ont été conçus et réalisés
  - - Chaque itération comprend l'analyse du domaine, la conception et la réalisation
  - - Une itération se déroule le plus souvent dans un temps limité

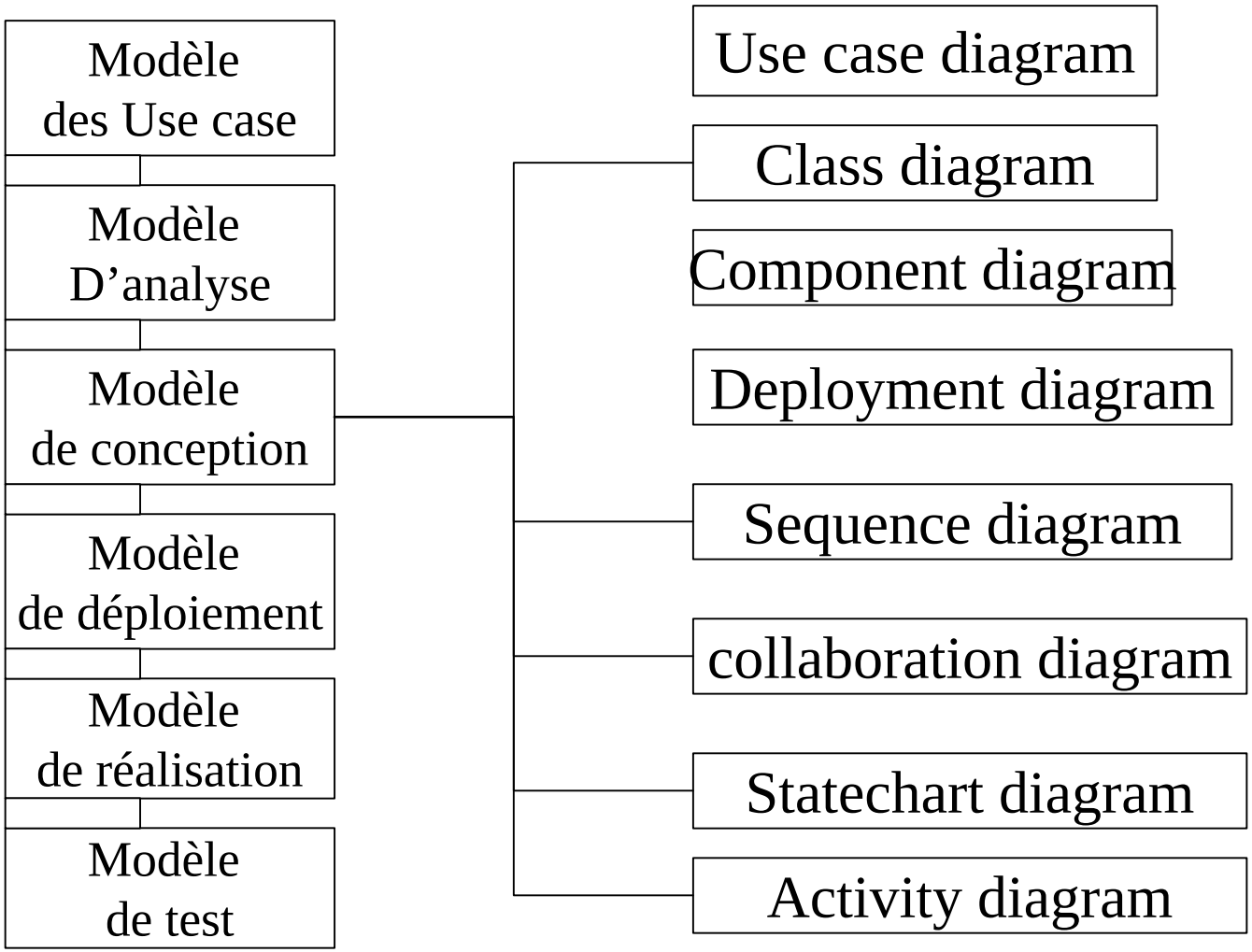
Processus et modèles

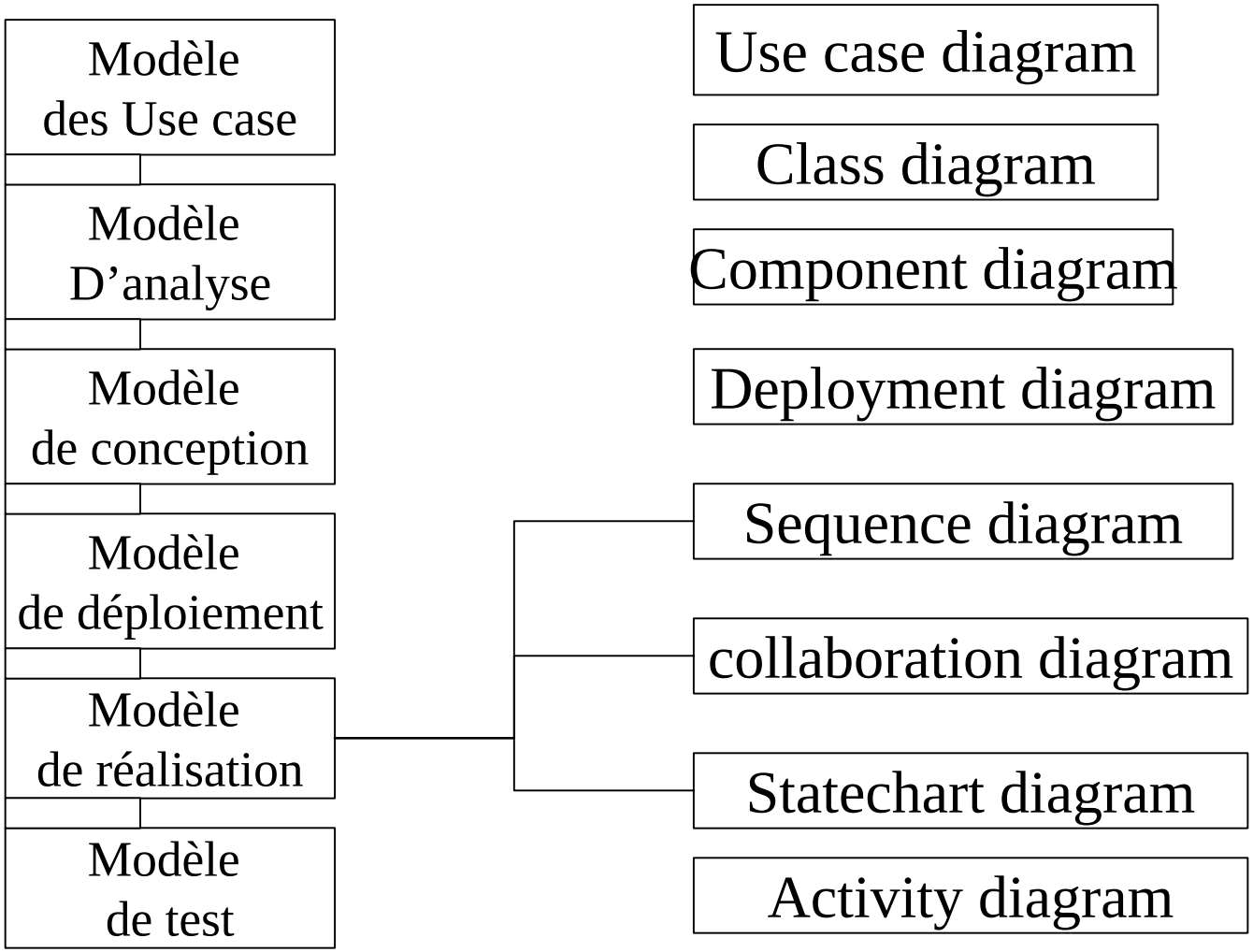


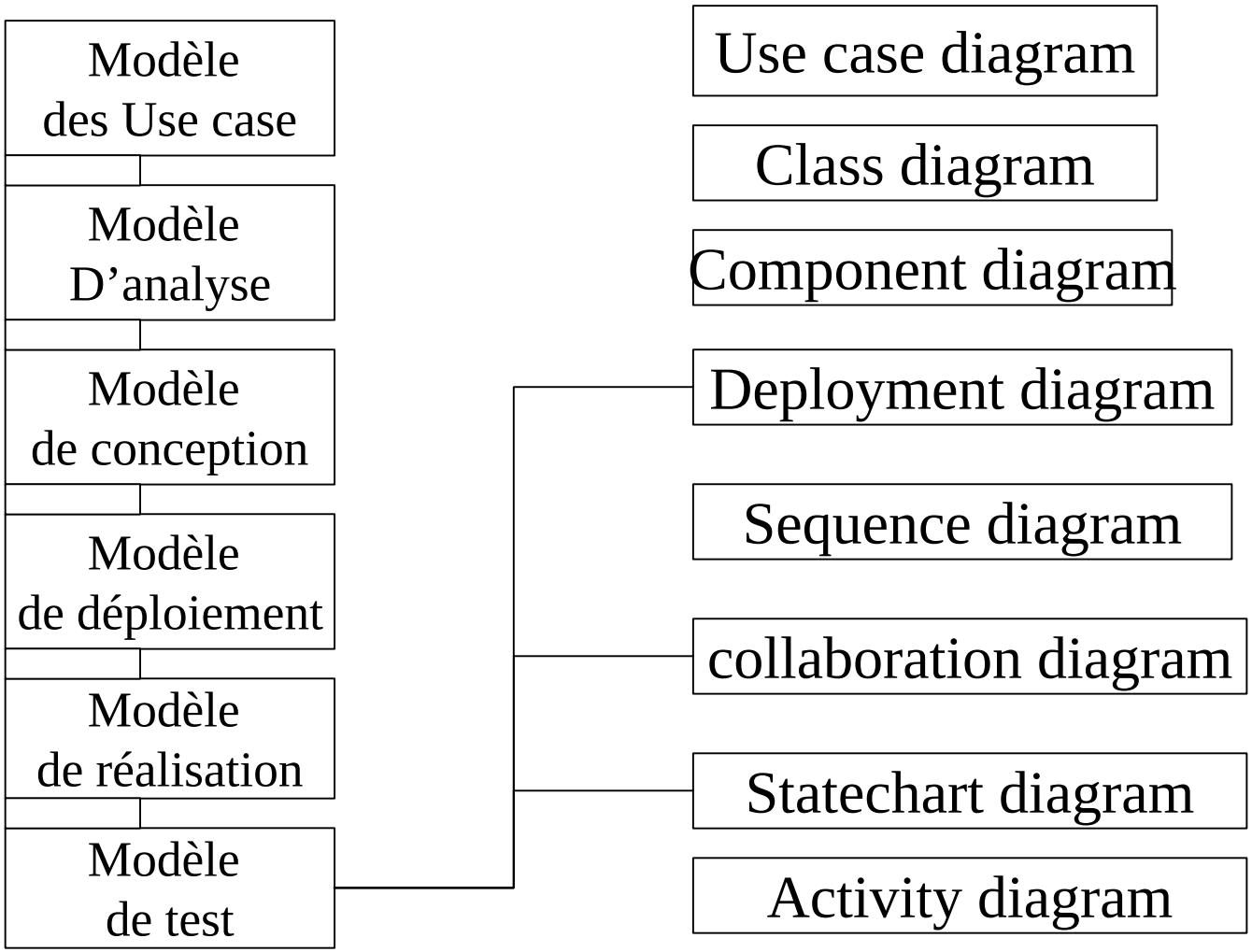
Modèles et diagrammes UML



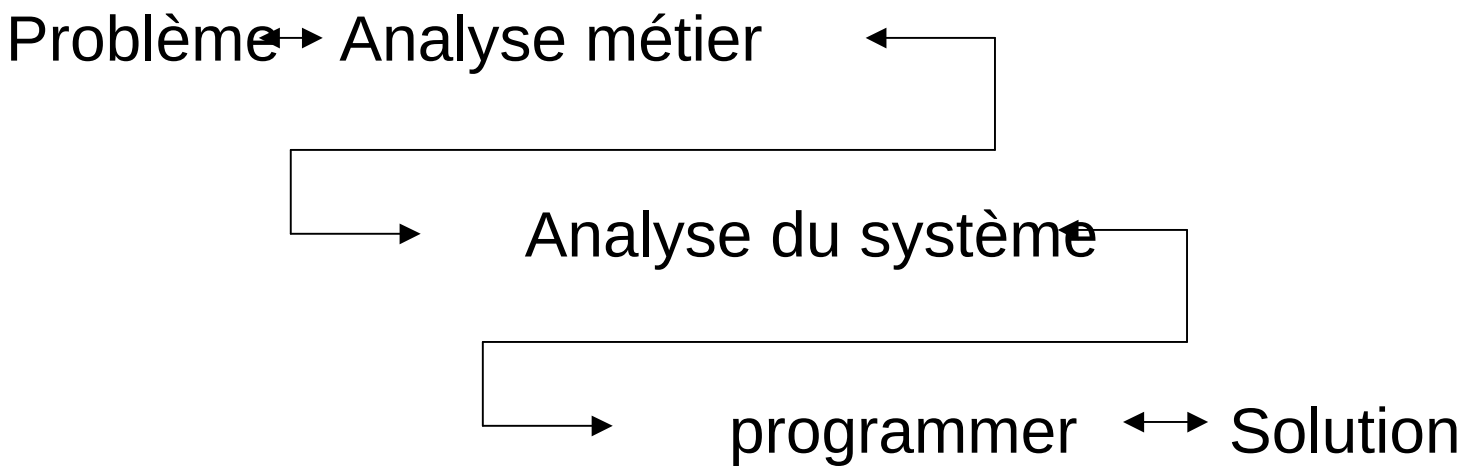

Modèle
des Use case
Modèle
D’analyse
Modèle
de conception
Modèle
de déploiement
Modèle
de réalisation
Modèle
de test
Use case diagram
Class diagram
Component diagram
Deployment diagram
Sequence diagram
collaboration diagram
Statechart diagram
Activity diagram

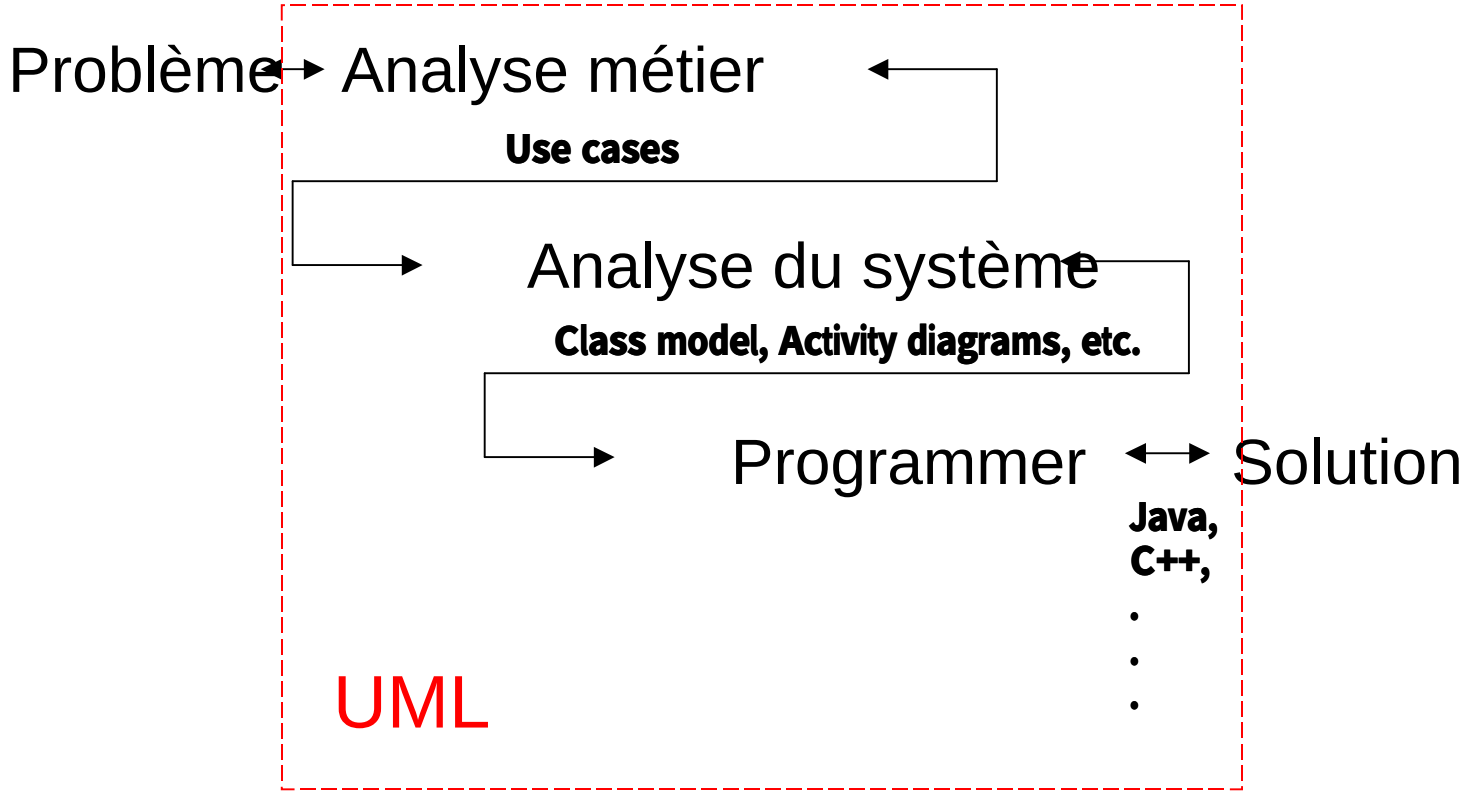

Modèle
des Use case
Modèle
D’analyse
Modèle
de conception
Modèle
de déploiement
Modèle
de réalisation
Modèle
de test
Use case diagram
Class diagram
Component diagram
Deployment diagram
Sequence diagram
collaboration diagram
Statechart diagram
Activity diagram


Modèle
des Use case
Modèle
D’analyse
Modèle
de conception
Modèle
de déploiement
Modèle
de réalisation
Modèle
de test
Use case diagram
Class diagram
Component diagram
Deployment diagram
Sequence diagram
collaboration diagram
Statechart diagram
Activity diagram


Modèle
des Use case
Modèle
D’analyse
Modèle
de conception
Modèle
de déploiement
Modèle
de réalisation
Modèle
de test
Use case diagram
Class diagram
Component diagram
Deployment diagram
Sequence diagram
collaboration diagram
Statechart diagram
Activity diagram

UML offre une structure d’accueil de bout en bout pour spécifier et développer un système




Problème
Analyse métier
Use cases
Analyse du système
Class model, Activity diagrams, etc.
Programmer
Solution
Java,
C++,
.
.
.
UML

Etude préliminaire

- Eléments en entrée :
  - But de l'application
  - Résultats de pré étude.
  - Cahier des charges.
- Eléments en sortie :
  - Acteurs
  - Utilisation faite par les acteurs
  - Elements de volumétrie.
  - Cahier des charges

Analyse

- Eléments en entrée :
  - Cahier des charges
  - Bilan des acteurs
  - Rôle joué par l'application
- Eléments en sortie
  - Classes candidates :
    - Réalisent les use cases identifiés.
    - Responsabilités identifiées.
    - Regroupées en catégories.

Démarche.

- Décrire les principaux processus métier.
- Identifier les cas d'utilisation :
  - Décrire les principales interactions acteur/système.
  - Décrire les activités.
- Consolider les use cases.
- Etablir leurs relations eventuelles
- Identifier les classes :
  - Boundary
  - Control
  - Business.
- Valider la cohérence des diagrammes.

Conception

- Eléments en entrée :
  - Use cases
  - Diagrammes de séquence
  - Diagrammes d'activité
  - Description textuelles des use cases et des scénarios.
  - Classes candidates.
- Eléments en sortie :
  - Diagramme de classes.
  - Diagrammes de séquence
  - Statecharts

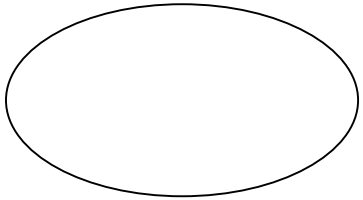
Diagramme de cas d’utilisation

- Où est utilisé le système dans l’entreprise ?
- Qui utilise les fonctions du système?
- Quels sont les rôles ou fonctions joués par les individus ?
- Qui accède (CRUD) aux informations du système ?
- Quels sont les matériels reliés au système ?
- Est-ce qu le système accède à des informations externes ?
- Qui administrera le système ?

- Un acteur représente une entité qui interagit et échange des informations avec le système
- Il peut désigner
  - Un être humain
  - Le rôle d’un utilisateur
  - Une machine
  - Un système externe
- Il est utilisé pour
  - Faciliter la définition du système
  - Définir le système
  - Définir les besoins en test

Cas d’utilisation

- Unité cohérente et homogène de fonctionnalités fournies par le système
- Caractérisé par des interactions entre le système et les utilisateurs
- Initié par un utilisateur et produit des résultats observables par son environnement
- L’ensemble des cas d’utilisation définit tous les moyens possibles d’utiliser le système



Documentation d’un cas d’utilisation

- La documentation d’un cas d’utilisation est effectuée dans le langage des utilisateurs sans se préoccuper des techniques informatiques
- Elle comporte
  - Un identifiant
  - Résumé
  - Pré-condition
  - Contexte : fréquence, contrainte de volume
  - Description des flots d’évènements principaux et alternatifs
  - Post condition
  - execptions

Identifier les cas d’utilisation

- Quelles sont les tâches d’un acteur ?
- Quelles sont les informations accédées (CRUD) par les acteurs ?
- Est-ce que les évènements perçus par le système sont traités ?
- Expression du problème et spécification du système ?
- Interview des experts du domaine ?
- Textes réglementaires ?

Périmètre de l'application

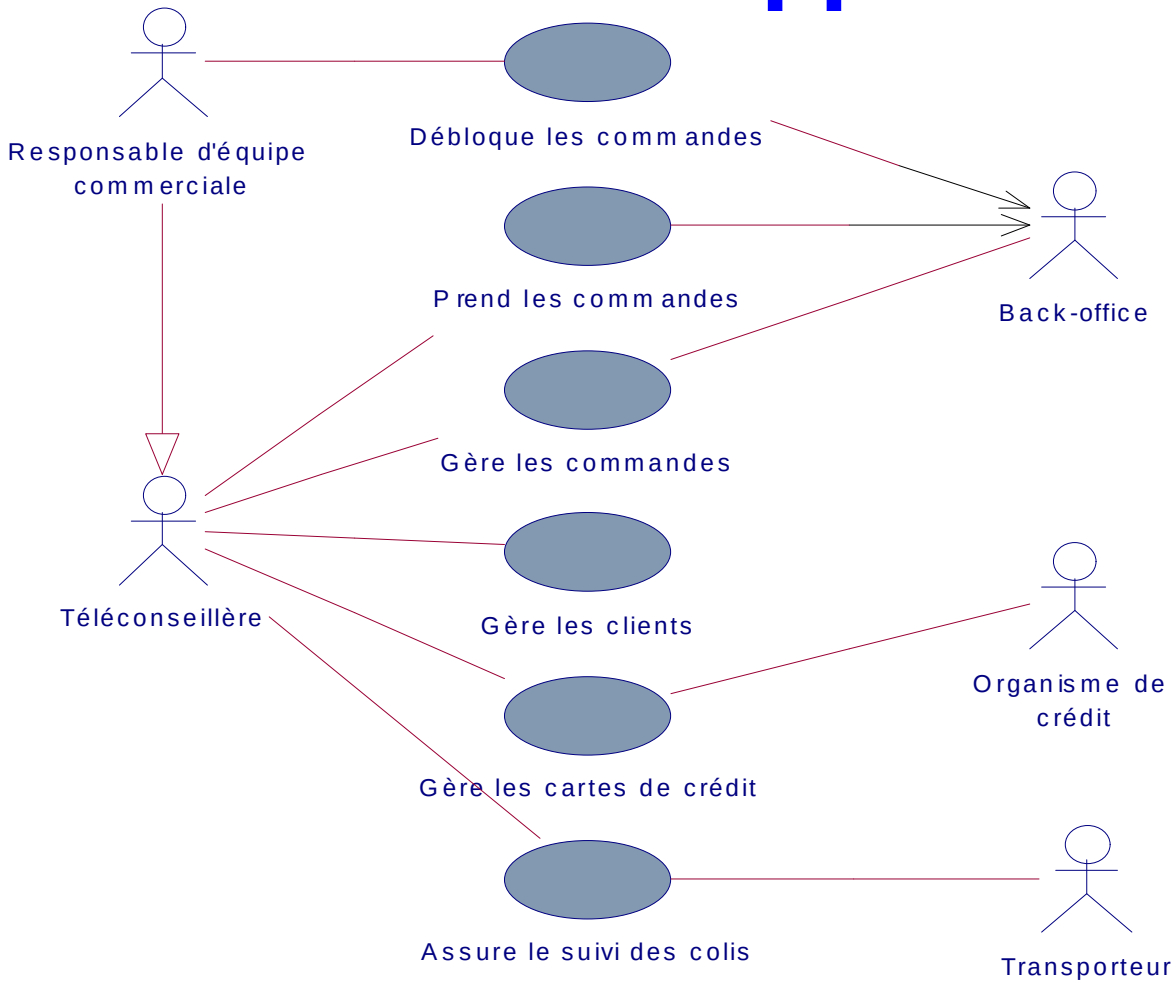
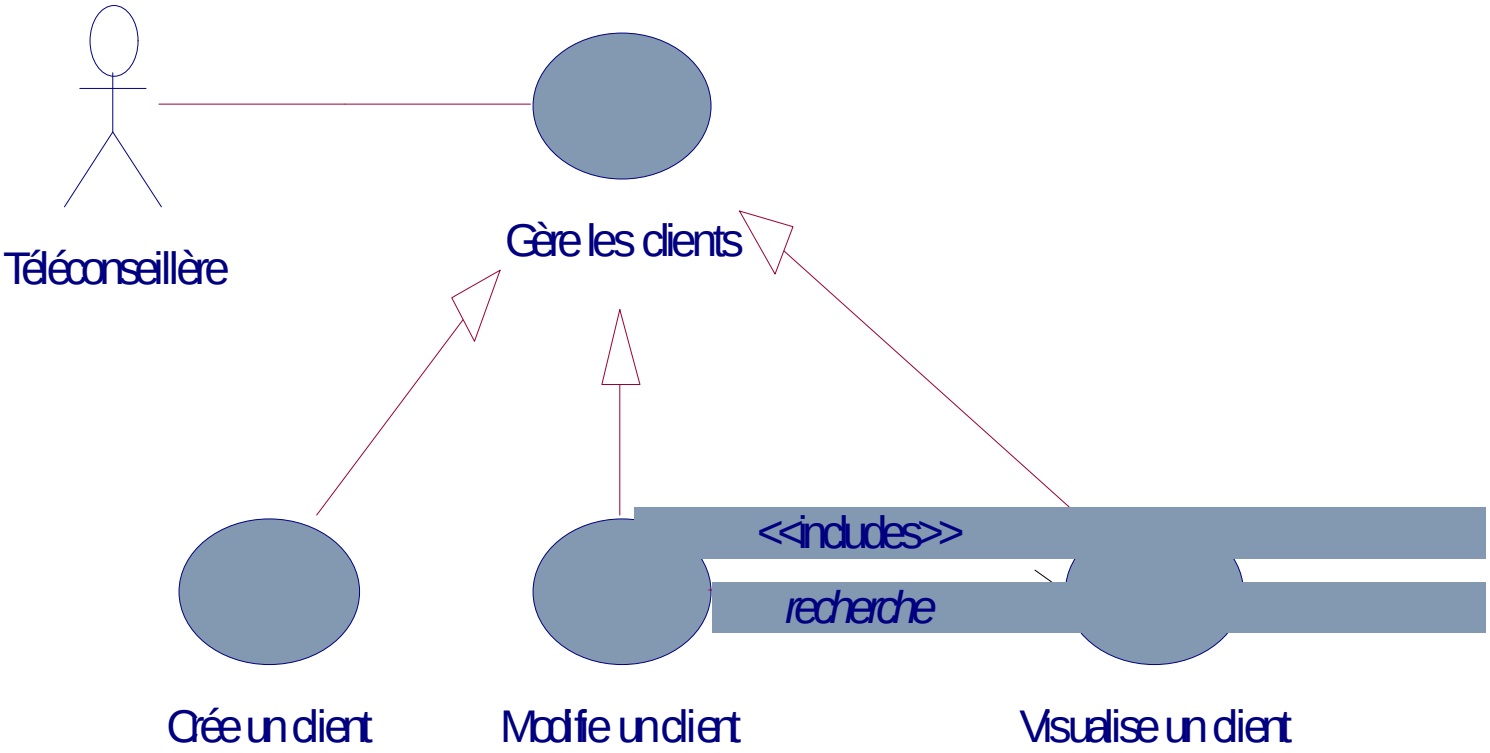
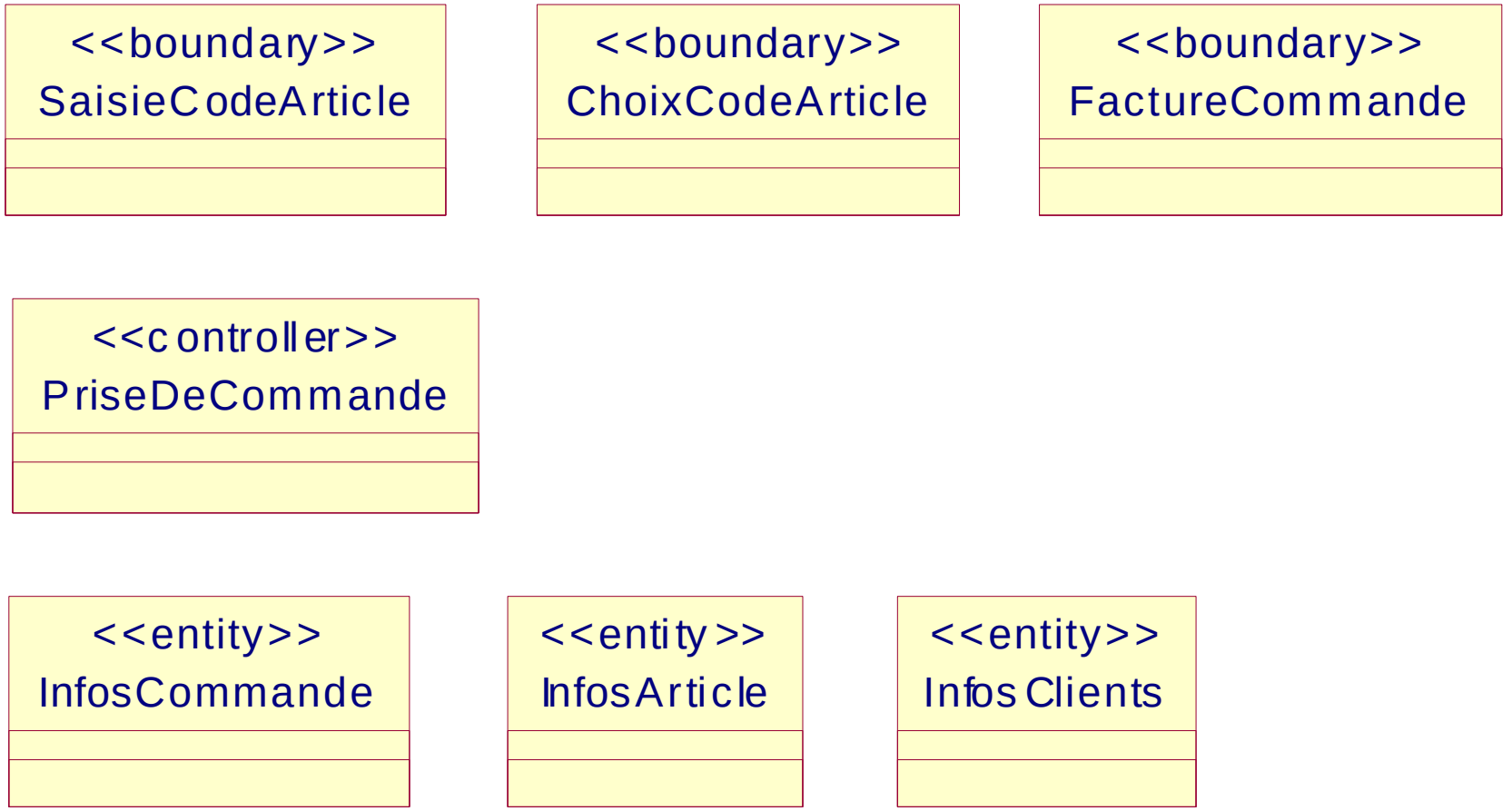


Diagramme de use case
Gérer le client



Classes qui réalisent Prendre une commande



Effets de la réalisation