MAI NFE103
Année 2013-2014

# Introduction à UML 2.0

F.-Y. Villemin, CNAM

http://deptinfo.cnam.fr/Enseignement/CycleSpecialisation/MAI/index.html

## UML 2.0

La version 2 de UML a été finalisée par l'OMG en Juillet 2005

Ajouts d'un ensemble de nouvelles fonctionnalités en partie issues des "manques" de la version 1.x pour :

- Rendre plus "exécutable" le langage
- Fournir des mécanismes plus robuste pour la modélisation des workflows et des actions
- Créer un standard pour la communication entre outils
- Fournir un cadre standard de modélisation

UML 2.0 utilise 13 types de diagrammes, contre 9 en UML 1.X

Source : "UML 2.0", Martin Fowler, Pearson Education (2004)




## UML 2.0

Les diagrammes ont été revus pour répondre aux nouveaux besoins (abstraction, automatisation...)

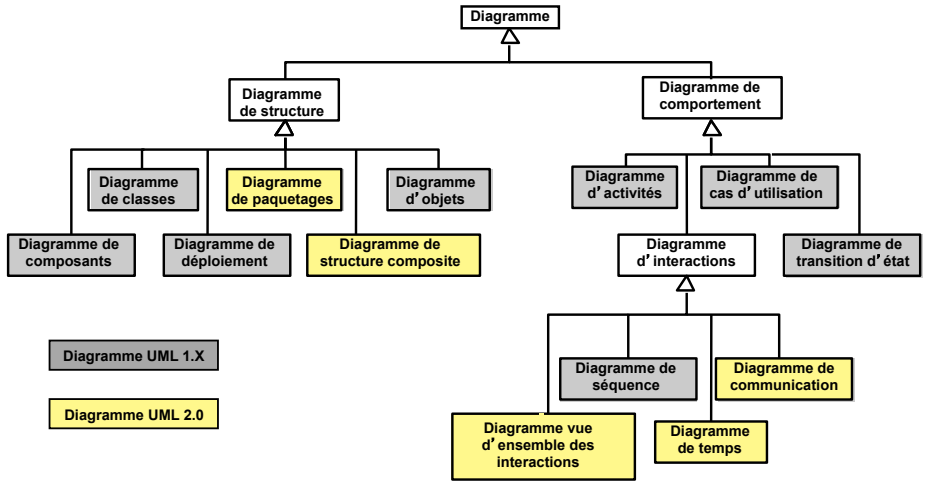
Le diagramme de collaboration d'UML 1.X devient le diagramme de communication

Principaux changements dans les diagrammes de :

- Classes
- Séquences
- De machines-états
- D'activités




## Les diagrammes



## Les diagrammes

### Diagrammes Structurels ou Diagrammes statiques (Structure Diagram)

- Diagramme de classes (Class diagram) : il représente les classes intervenant dans le système
- Diagramme d'objets (Object diagram) : il sert à représenter les instances de classes (objets) utilisées dans le système
- Diagramme de composants (Component diagram) : il permet de montrer les composants du système d'un point de vue physique, tels qu'ils sont mis en œuvre (fichiers, bibliothèques, bases de données...)
- Diagramme de déploiement : il sert à représenter les éléments matériels (ordinateurs, périphériques, réseaux, systèmes de stockage...) et la manière dont les composants du système sont répartis sur ces éléments matériels et interagissent avec eux



## Les diagrammes

### Diagrammes Structurels ou Diagrammes statiques (Structure Diagram)

- Diagramme des paquetages (Package Diagram) : permet de représenter la hiérarchie des paquetages du projet, leur organisation et leurs interdépendances, simplifie les diagrammes (donc plus simple à comprendre)
- Diagramme de structures composites (Composite Structure Diagram) : permet de décrire la structure interne d'un objet complexe lors de son exécution (c'est à dire, décrire l'exécution du programme), dont ses points d'interaction avec le reste du système



## Les diagrammes

### Diagrammes Comportementaux ou Diagrammes dynamiques (Behavior Diagram)

- Diagramme des cas d'utilisation (Use case diagram) : il décrit les possibilités d'interaction entre le système et les acteurs, c'est-à-dire toutes les fonctionnalités que doit fournir le système
- Diagramme états-transitions (State Machine Diagram) : il montre la manière dont l'état du système (ou de sous-parties) est modifié en fonction des événements du système
- Diagramme d'activité (Activity Diagram) : variante du diagramme d'états-transitions, il permet de représenter le déclenchement d'événements en fonction des états du système et de modéliser des comportements parallélisables (multi-threads ou multi-processus)



## Les diagrammes

### Diagrammes Comportementaux ou Diagrammes dynamiques (Behavior Diagram)

- Diagramme d'interactions (Interaction Diagram) :
  - Diagramme de séquence (Sequence Diagram) : la représentation séquentielle du déroulement des traitements et des interactions entre les éléments du système et/ou des acteurs
  - Diagramme de communication (Communication Diagram) : la représentation simplifiée d'un diagramme de séquence se concentrant sur les échanges de messages entre les objets
  - Diagramme global d'interaction (Interaction Overview Diagram) : variante du diagramme d'activité où les noeuds sont des interactions, permet d'associer les notations du diagramme de séquence à celle du diagramme d'activité, ce qui permet de décrire une méthode complexe

## Les diagrammes

Diagrammes Comportementaux ou Diagrammes dynamiques (Behavior Diagram)

- Diagramme de temps (Timing Diagram) : la représentation des interactions où l'aspect temporel est mis en valeur; il permet de modéliser les contraintes d'interaction entre plusieurs objets, comme le changement d'état en réponse à un évènement extérieur



## Diagramme de classes

Attribut et association multi-directionnelles sont maintenant 2 notations différentes (concept de propriété)

Les multiplicités discontinues (ex [2,4]) sont abandonnées

La propriété frozen n'existe plus, certains mots-clés ont disparus ("parameter" et "local")

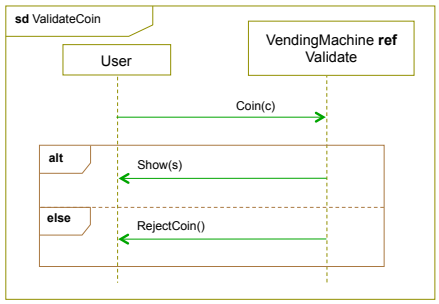


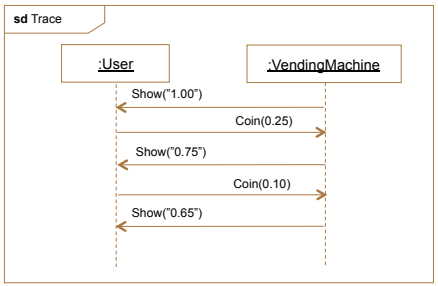
## Diagramme de séquences

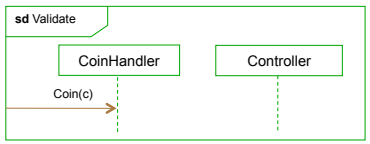
- Notation des cadres d'interaction pour gérer les structures itératives, conditionnelles ou autres (mots-clés alt, opt, par, loop, region…)
- Les en-têtes des lignes de vie ne sont plus obligatoirement des instances mais des participants (notion plus complexe qu'en UML 1.x)
- Suppression des gardes et des marqueurs d'itération



## Diagrammes de séquence

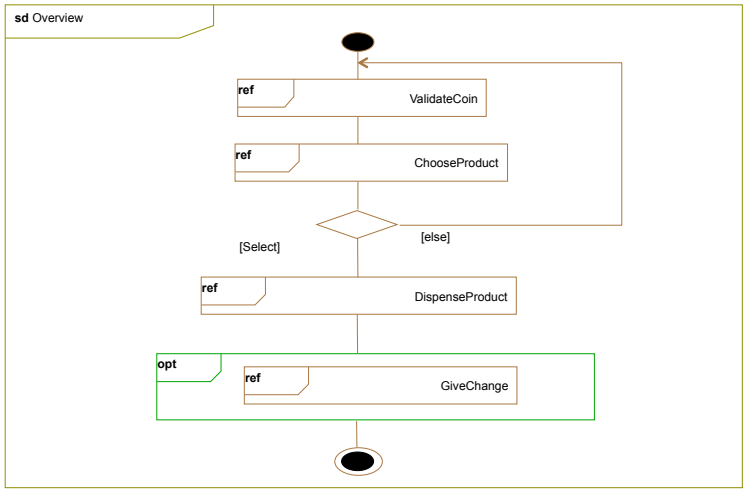


## Diagramme de vue d'ensemble des interactions

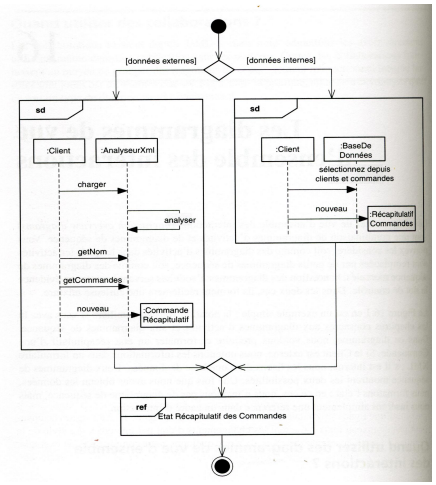
Combinaison de diagrammes d'activités et de diagrammes de séquences






## Diagramme de vue d'ensemble des interactions

Mélange du diagramme d'activités et du diagramme de séquence






## Diagramme de machines-états

Les actions (momentanées) et les activités (continues) sont regroupées sous le même terme : activité

Les activités sont alors appelées activité "do"



## Diagramme d'activités

Différenciation forte entre diagramme d'activités et diagramme d'états (à l'inverse de UML 1.x)

Nouvelles notations telles que :

- les signaux temporels,
- l'acceptation,
- les paramètres,
- les spécifications de jonction,
- les connecteurs,
- les râteaux de sous-diagrammes,
- ...

## Diagramme de communication

Idem diagramme de séquence

Insiste sur les relations entre les objets

Permet de montrer un scénario particulier

Chronologie indiquée par la numérotation des messages

Possibilité de faire apparaître des boucles, des conditions (à éviter)



## Diagramme de communication

Les objets :

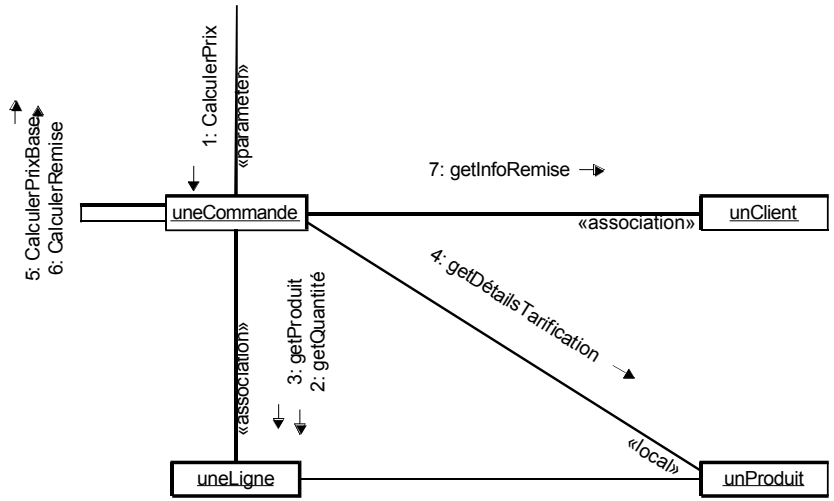
- pas de ligne de vie
- associé à d'autres objets
- nature de l'association est décrite par un stéréotype :
  - "association" : l'objet est un attribut
  - "global" : l'objet est une variable globale
  - "local" : l'objet est une variable locale
  - "parameter" : l'objet est un paramètre d'un message

Les messages :

- Un message ne peut être envoyé que si une association existe

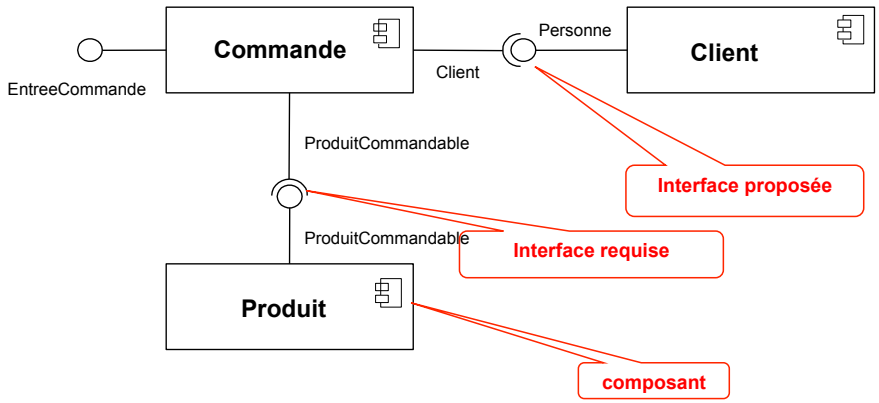


## Diagramme de communication






## Diagramme de composants

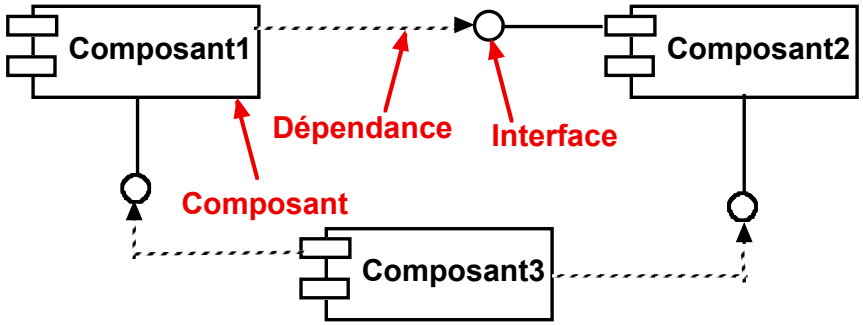


# Diagramme de composants

Montre la répartition physique du code

Souvent répartis dans des paquetages

Les relations s'expriment à travers leurs interfaces et des dépendances






# Diagramme de déploiement

Montre la répartition des processus qui composent le système

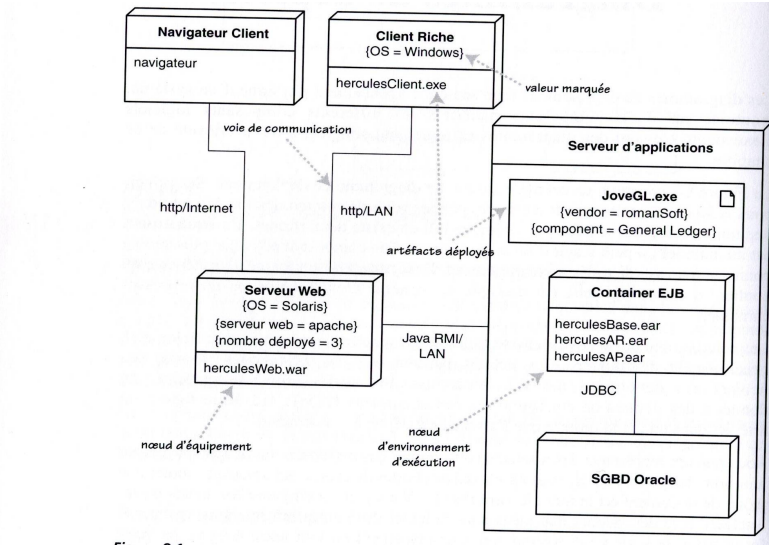
Important dans le cas d'architecture distribuée

Concepts fondamentaux:

- Les nœuds : dispositifs capables d'héberger une partie du logiciel
- Les arcs : liens physiques connectant les noeuds



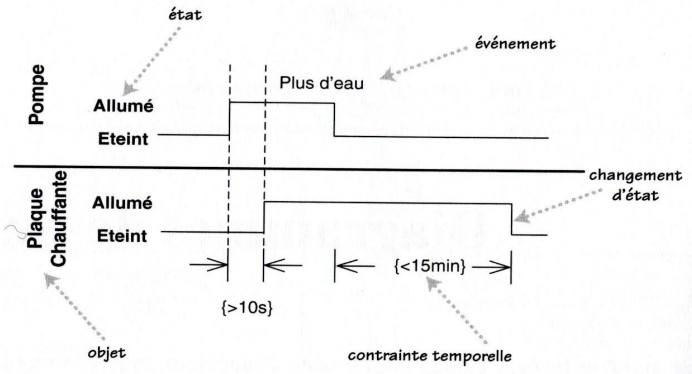
# Diagramme de déploiement





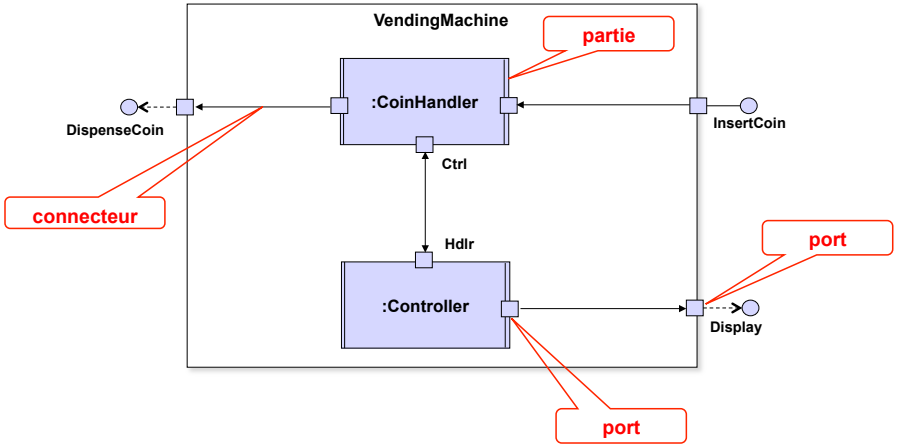

# Diagramme de temps

Montre l'évolution de l'état d'un objet ou d'un groupe d'objets en fonction d'évènements temporels



# Diagramme de structure composite

**Permet de décomposer hiérarchiquement une classe en une structure interne**






# Diagramme de structure composite

**Exemple :**