

---

## UML: Unified modeling Language

---

**Luigi DILILLO - LIRMM / UM2**  
[dilillo@lirmm.fr](mailto:dilillo@lirmm.fr)

## Plan

- **Cours 1**
  - Introduction*
  - Présentation d'UML*
  - Modèle fonctionnel (utilisation)*

## Introduction


- **Modélisation**

*Produire une représentation simplifiée du monde réel pour :*

- accumuler et organiser des connaissances,
- décrire un problème,
- trouver et exprimer une solution,
- raisonner, calculer.

## Introduction

- En informatique,  
Résoudre le hiatus entre :



**le réel**

- Évolutif
- Ambiguïté

**le monde informatique**

- Langages codifiés
- Sémantique unique

## Introduction

### Difficultés de la modélisation

- *Problèmes des spécifications*
  - parfois imprécises, incomplètes, ou incohérentes
- *Taille et complexité des systèmes importantes et croissantes*
  - les besoins et les fonctionnalités augmentent
  - la technologie évolue rapidement
  - les architectures se diversifient
  - assurer l'interface avec le *métier* (domaine d'application)

## Introduction

### Difficultés de la modélisation

- *Évolution des applications*
  - évolution des besoins des utilisateurs
  - réorientation de l'application
  - évolution de l'environnement technique (matériel et logiciel)
- *Problèmes liés à la gestion des équipes*
  - taille croissante des équipes
  - spécialisation technique
  - spécialisation *métier*

## Introduction

### Les méthodes = des guides structurants

- *Décomposition du travail*
- *Organisation des phases*
- *Concepts fondateurs*
- *Représentations semi-formelles*

Assurent une démarche reproductible  
pour obtenir des résultats fiables

## Introduction

### Décomposition du travail

- Phases  
*analyse, conception, codage, validation, etc.*
- Niveaux d'abstraction  
*conceptuel (besoins)*  
*logique (solution informatique abstraite)*  
*physique (solution informatique concrète)*

## Introduction

### Décomposition du travail

- Phases  
*analyse, conception, codage, validation, etc.*
- Niveaux d'abstraction  
*conceptuel (besoins)*  
*logique (solution informatique abstraite)*  
*physique (solution informatique concrète)*

## Introduction

### Organisation du travail

- Processus de développement  
Phases séquentielles  
Itération sur les phases

## Introduction

### Concepts fondateurs

- Fondent l'approche du problème et l'expression de la solution  
Classe, signal, état, fonction, etc.

## Introduction

### Représentations semi-formelles

- Représentations partiellement codifiées basées sur les concepts fondateurs  
diagrammes, formulaires, etc.
- Support de différentes activités  
réflexion, spécification, communication, documentation, mémorisation (trace)

## Introduction

### Pour résumer ...

*Une méthode d'analyse et de conception*

- propose une démarche qui distingue les étapes du développement dans le cycle de vie du logiciel (modularité, réduction de la complexité, réutilisabilité éventuelle, abstraction)
- s'appuie sur un formalisme de représentation qui facilite la communication, l'organisation et la vérification

**Le langage de modélisation**

- produit des documents (modèles) qui facilitent les retours sur conception et l'évolution des applications

## Plan

### • Cours 1

*Introduction*

*Présentation d'UML*

*Modèle fonctionnel (utilisation)*

## UML - Unified Modeling Language

- Langage de modélisation véhiculant en particulier les concepts des approches par objets

**classe, instance, classification, etc.**

mais intégrant d'autres aspects

**associations, fonctionnalités, événements, états, séquences, etc.**

## UML = Bénéficier des qualités des approches par objets

- Simplicité
- Facilité pour coder et réutiliser
- Modèle plus proche de la réalité
  - description plus précise des combinaisons (données, opérations)
  - décomposition basée sur “classification naturelle”
  - facile à comprendre et à maintenir
- Stabilité
  - de petites évolutions peuvent être prises en compte sans changements massifs

## La portée d'UML s'explique par l'importance de l'approche par objets

- *Omniprésence technique de l'Objet*  
dans les langages de programmation, les bases de données, les interfaces graphiques, ... et les méthodes d'analyse et de conception.
- *Universalité de l'Objet*  
la notion d'objet, plus proche du monde réel, est compréhensible par tous et facilite la communication entre tous les intervenants d'un projet.

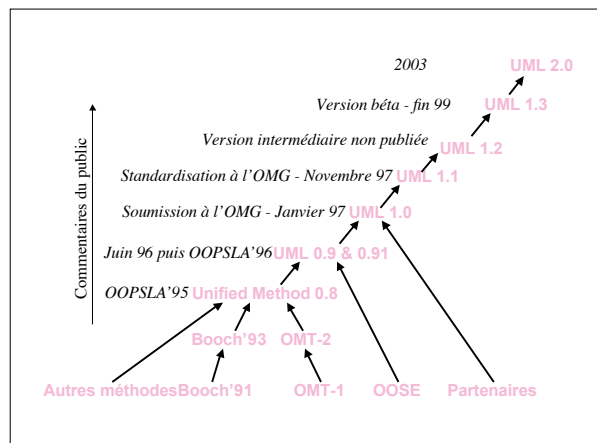
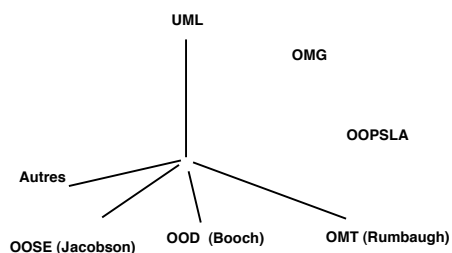
## Genèse d'UML

*Au début des années 90, une cinquantaine de méthodes objet, liées uniquement par un consensus autour d'idées communes (objets, classes, sous-systèmes, ...)*

Recherche d'un langage commun unique utilisable par toute méthode objet

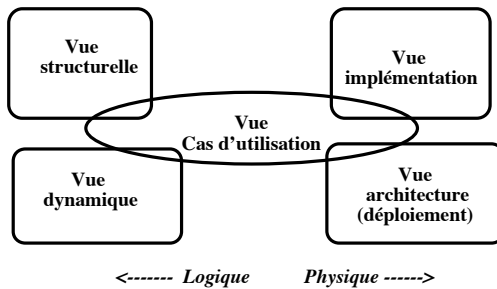
- dans toutes les phases du cycle de vie,
- compatible avec les techniques de réalisation actuelles.

## UML (Unified Modeling Language)



## Concepts généraux

*Points de vue sur le système*



## Concepts généraux

*Quatre modèles pour concrétiser ces points de vue*

<b>Modèle structurel</b> Types d'objets et leurs relations	<b>Modèle Dynamique</b> Stimuli des objets et leurs réponses
<b>Modèle d'utilisation</b> Fonctionnalités	<b>Modèle Implémentation</b> Composants Fichiers BD Projection sur le matériel

## Concepts généraux

**Chaque modèle est une représentation abstraite d'une réalité, il fournit une image simplifiée du monde réel selon un point de vue.**

Il permet :

- de comprendre et visualiser (en réduisant la complexité)
- de communiquer (à partir d'un « langage » commun à travers un nombre restreint de concepts)
- de valider (contrôle de la cohérence, simuler, tester ...)

## Concepts généraux

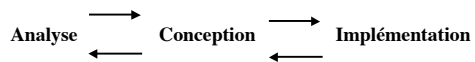
*Diagrammes (représentations graphiques de modèles)*

Diagrammes de classes d'instances	Diagrammes de collaboration de séquences d'états, d'activités
Diagrammes de cas d'utilisation	Diagrammes de déploiement de composants

## Concepts généraux

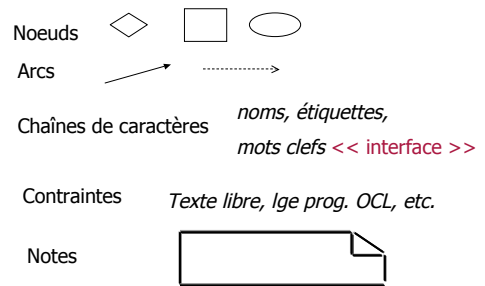
*Démarche uniforme sur le cycle de vie*

*Même notation*



## Aspects du langage

*Les diagrammes sont majoritairement des graphes*



## Plan

- **Cours 1**
  - Introduction*
  - Présentation d'UML*
  - Modèle fonctionnel (utilisation)*

## Modèle d'utilisation

- *Les cas d'utilisation, ou « USE CASE »*
- Fonctionnalités externes
- Modèles descriptifs du point de vue des utilisateurs
- Interactions avec les acteurs extérieurs

*la manière d'utiliser le système*

## Modèle d'utilisation

On part de l'analyse des besoins ....

Deux concepts

-Acteur

toute entité extérieure au système et interagissant avec celui-ci.  
acteurs humains, acteurs « machine » (système extérieur communiquant avec le système étudié)

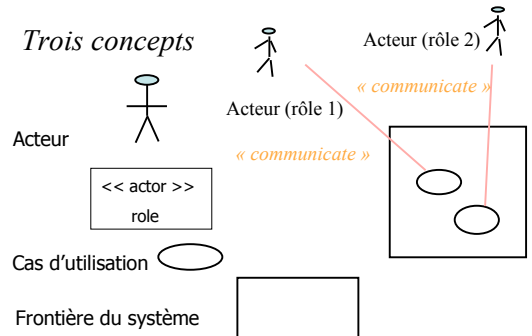
- Cas d'utilisation

toute manière d'utiliser le système

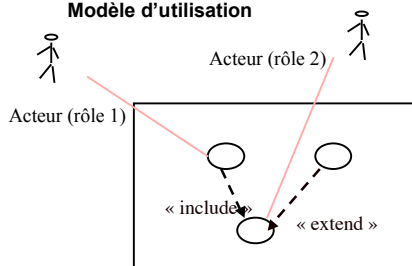
suite d'événements notable du point de vue de l'utilisateur

## Modèle d'utilisation

Trois concepts



## Modèle d'utilisation



Les cas d'utilisation peuvent être liés par des relations :

- d'utilisation « include » (le cas origine contient obligatoirement l'autre)
- de raffinement « extend » (le cas origine peut être ajouté optionnellement)
- de généralisation/spécialisation « generalize »

## Modèle d'utilisation

Diagramme du « contexte statique »

