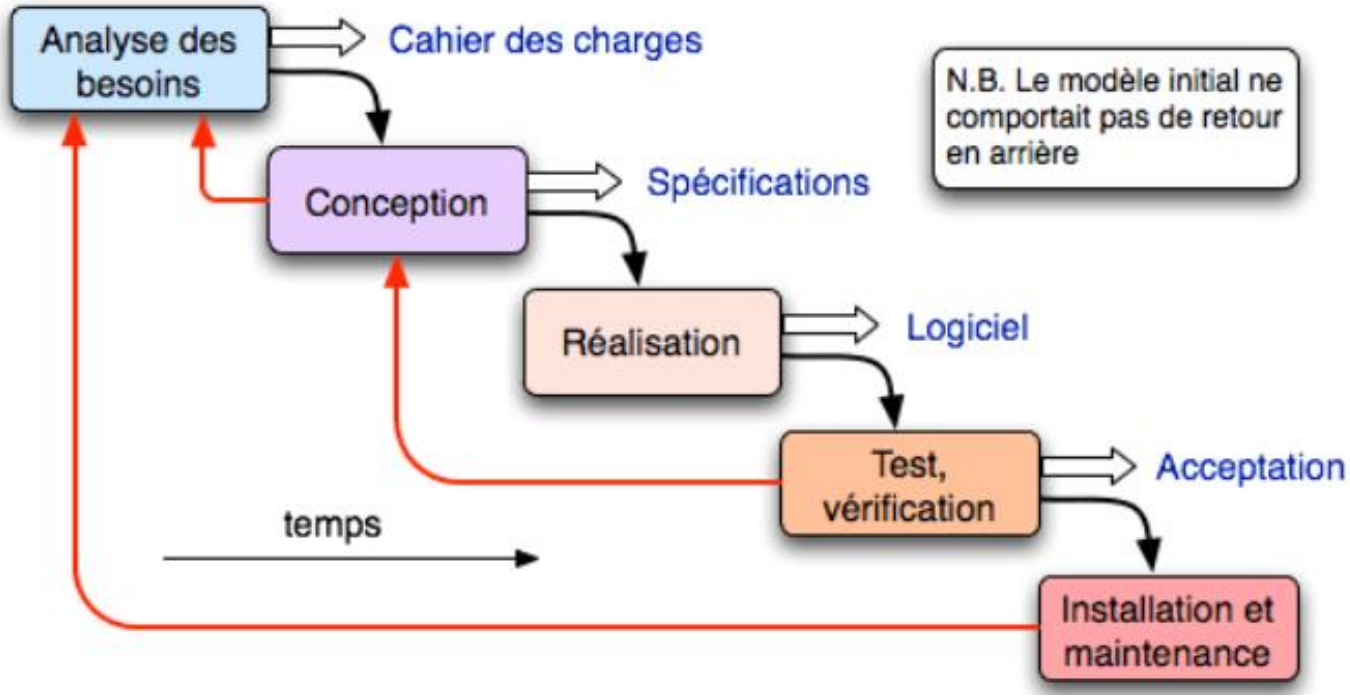
Génie Logiciel

- Dispensé par Dr. Msc. Ir. **MWAMBA KASONGO Dahouda**
  Docteur en génie logiciel et systèmes d'information
  Machine and Deep Learning Engineer

- Assisté Master Student, Ir. Jason MUSA

Heure : 08H00 – 12H00



MODELISATION AVEC UML

CHAPITRE 2. MODELISATION AVEC UML

2.1. Introduction

2.2. Historique

2.3. Elements et mecanismes generaux

2.4. Les diagrammes UML

CHAPITRE 2. MODELISATION AVEC UML

2.1. Introduction

UML est un langage de modélisation unifié orienté objet. Il s'agit d'une norme principalement utilisée pour la construction de normes de documentation orientées objet significatives pour tout système logiciel dans le monde réel.

Le diagramme UML est une description picturale des classes, des objets et des relations. Il s'agit d'une norme qui décrit un e partie d'un système. Il présente un plan pour développer des modèles riches qui définissent le fonctionnement de tout système matériel ou logiciel.

- Les années 1990 ont été la période de développement des langages orientés objet tels que C++.

- Ces langages orientés objet ont été utilisés pour construire des systèmes complexes mais convaincants.

- Il a été développé par les ingénieurs logiciels Grady Booch, Ivar Jacobson et James Rumbaugh de Rational Software en 1994 et 1995. Il a été en développement jusqu'en 1996.

Tous les créateurs d'UML, à savoir Grady Booch, Ivar Jacobson et James Rumbaugh ont eu une idée fabuleuse pour créer un langage qui réduira la complexité.

CHAPITRE 2. MODELISATION AVEC UML

2.1. Introduction

2.1.1 Modélisation

- L'action de concevoir un **modèle** dans un langage de modélisation dédié.

- Modéliser consiste à identifier les caractéristiques intéressantes ou pertinentes d'un système dans le but de pouvoir l'étudier du point de vue de ses caractéristiques.

Qu'est ce qu'un modèle ?:

- Une abstraction permettant de mieux comprendre un objet complexe (représenter lesystème selon des degrés différents de détails)
- Une représentation simplifié d'une réalité (logiciel, bâtiment, … etc.)
- Les modèles sont souvent graphiques (un petit dessin c'est vaut mieux qu'un long discours)

Bon modèle:

- Il doit faciliter la compréhension du phénomène (système) étudié.
- Il doit permettre de simuler le phénomène (système) étudié.

CHAPITRE 2. MODELISATION AVEC UML

2.1. Introduction

2.1.2 Modélisation

Pourquoi modéliser ?

- ✓ Faciliter la compréhension d'un système
- ✓ Permettent également une bonne communication avec le client
- ✓ Permettent d'écrire avec précision et complétude les besoins sans connaitre les détails du systèmes
- ✓ Permettent de se focaliser sur des aspects spécifiques d'un système (offres différentes visions)
- ✓ Permettent de tester une multitude de solutions à moindre coût et dans des délais réduits

Méthode de modélisation:

Une méthode définit une démarche reproductible pour obtenir des résultats fiable. Donc, une méthode d'élaboration d'un l ogiciel décrit comment modéliser et construire des systèmes logiciels de manières fiable.

Les méthodes de modélisation fonctionnelles:

Les méthodes fonctionnelles et systémiques sont imposées les premiers (les années 70_80), ces méthodes s'inspirent direct ement de l'architecture des ordinateurs. La séparation entre les données et le traitement telle qu'elle existe physiquement dans le matériel à transposée vers les méthodes.

CHAPITRE 2. MODELISATION AVEC UML

2.1. Introduction

2.1.1 Modélisation

Inconvénients:

- Une simple mise à jour du logiciel à un point données peut impacter en cascade une multitude d'autres fonctions
- L'évolution majeur du logiciel multiple les points de maintenance (le logiciel doit être retouché en globalité)

Solutions:

✓ Centraliser les données d'un type et les traitements associés dans une même entiténphysique permet de limiter les points de maintenance dans le code'

Solution: Les méthodes orienté objet (OO)

CHAPITRE 2. MODELISATION AVEC UML

2.1. Introduction

2.1.1 Modélisation

Concepts importants de l'approche objet

- **L'objet**: est une entité autonome, qui regroupe un ensemble de propriétés cohérentes et de traitements associés.

Une telle entité constitue le concept fondateur de l'approche du même nom.

- **L'encapsulation**: consiste à masquer les détails d'implémentation d'un objet, en définissant une interface.

L'interface définit les services accessibles (offerts) aux utilisateurs de l'objet.

L'encapsulation facilite l'évolution d'une application car elle stabilise l'utilisation des objets : on peut modifier l'implémentation des attributs d'un objet sans modifier son interface.

CHAPITRE 2. MODELISATION AVEC UML

2.1. Introduction

2.1.2 Modélisation

Concepts importants de l'approche objet

- **Héritage**: est un mécanisme de transmission des propriétés d'une classe (ses attributs et méthodes) vers une sous-classe.

L'héritage évite la duplication et encourage la réutilisation.

- **Polymorphisme**: représente la faculté d'une même opération de s'exécuter différemment suivant le contexte de la classe où elle se trouve.

Ainsi, une opération définie dans une superclasse peut s'exécuter de façon différente selon la sous-classe où elle est héritée.
Le polymorphisme augmente la généricité du code.

- **L'agrégation**: il s'agit d'une relation entre deux classes, spécifiant que les objets d'une classe sont des composants de l'autre classe.

L'agrégation permet d'assembler des objets de base, afin de construire des objets plus complexes.

CHAPITRE 2. MODELISATION AVEC UML

2.2. Historique

- **La méthode de Booch** était flexible pour travailler tout au long de la conception et de la création d'objets.

- **La méthode de Jacobson** a contribué à une excellente façon de travailler sur les cas d'utilisation.

Elle offre en outre une excellente approche pour la conception de haut niveau.

- **La méthode de Rumbaugh** s'est avérée utile lors de la manipulation de systèmes sensibles.

Les modèles comportementaux et les diagrammes d'état ont été ajoutés à l'UML par David Harel.

UML a été reconnu comme norme par l'Object Management Group (OMG) en 1997.
L'Object Management Group est responsable de la maintenance régulière d'UML depuis son adoption en tant que norme.

En 2005, l'Organisation internationale de normalisation a établi UML comme norme ISO. Il est utilisé dans de nombreux secteurs pour la conception de modèles orientés objet.
La dernière version d'UML est la 2.5.1, qui a été publiée en décembre 2017.

CHAPITRE 2. MODELISATION AVEC UML

2.2. Elements et mécanismes généraux

Pourquoi UML?

- ✓ Graphique et simple
- ✓ Uml est un standard
- ✓ Représenter des systèmes entiers
- ✓ Langage commun qui est utilisable par toutes les méthodes et adapté à touts les phases de développement

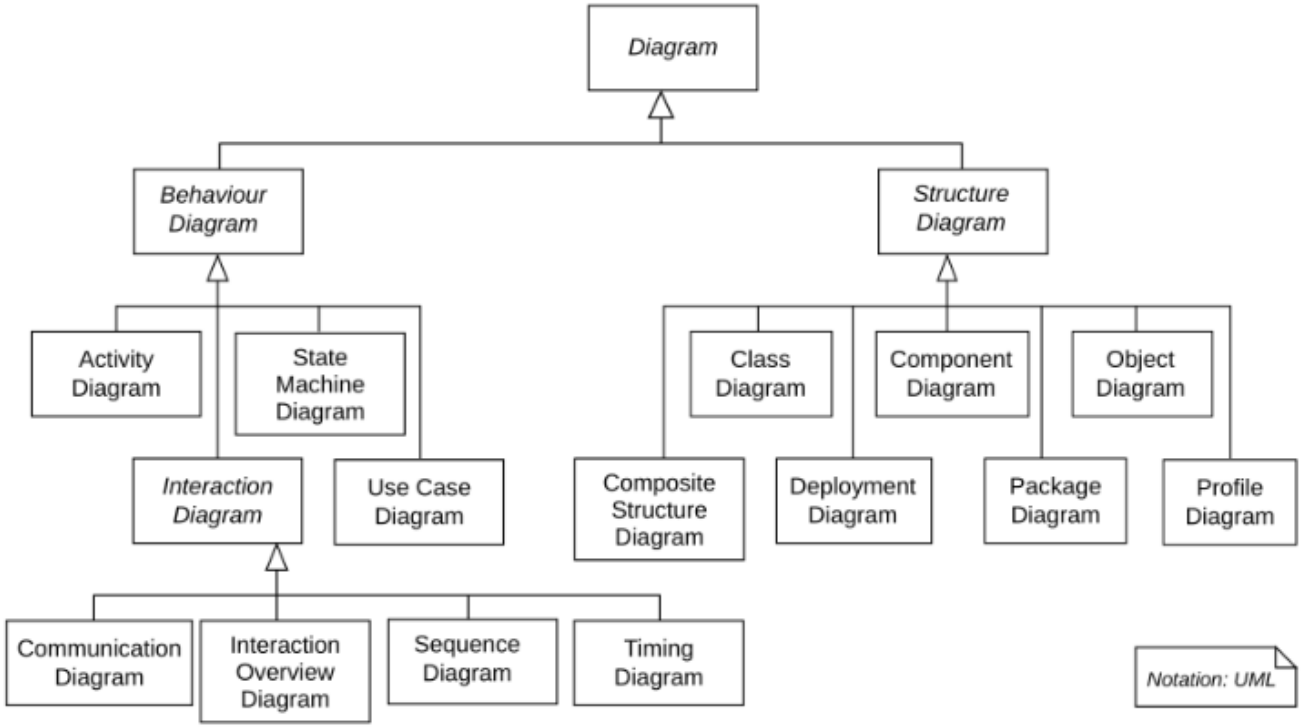
UML n'est pas une méthode, c'est un langage de modélisation objet

- Itératif et incrémental
- Guidé par les besoins des utilisateurs
- Centré sur l'architecture

CHAPITRE 2. MODELISATION AVEC UML

2.2. Elements et mécanismes généraux

UML comporte de nombreux types de diagrammes, qui sont divisés en deux catégories.
Certains types représentent des informations structurelles, et les autres représentent des types généraux de comportement, dont quelques-uns qui représentent différents aspects des interactions.
Ces diagrammes peuvent être classés hiérarchiquement comme indiqué dans le diagramme de classes suivant :



- ✓ Diagrammes de structure et
- ✓ Diagrammes de comportement

Statiques

Diagramme de Classes
Diagramme d'Objets
Diagramme de Composants
Diagramme de Déploiement
Diagramme de paquetages
Diagramme de structure composite (UML 2.x)

Fonctionnels

Diagramme de Use Case

Dynamiques

Diagramme de Séquence
Diagramme de communication (UML 2.x)
Diagramme global d'interaction (UML 2.x)
Diagramme de temps (UML 2.x)
Diagramme d'Etats-Transitions
Diagramme d'activités

CHAPITRE 2. MODELISATION AVEC UML

2.2. Elements et mécanismes généraux

UML représente un système, en se basant dut plusieurs diagrammes parmi lesquels on cite:

- **Diagrammes de structure (4 pour la structure statique)**
  - Diagramme d'objets
  - Diagramme de classes
  - Diagramme de composants
  - Diagramme de déploiement

- **Diagrammes de comportement (5 pour le comportement dynamique)**
  - Diagramme de cas d'utilisation
  - Diagramme de séquences
  - Diagramme d'activité
  - Diagramme de collaboration
  - Diagramme d'états transitions

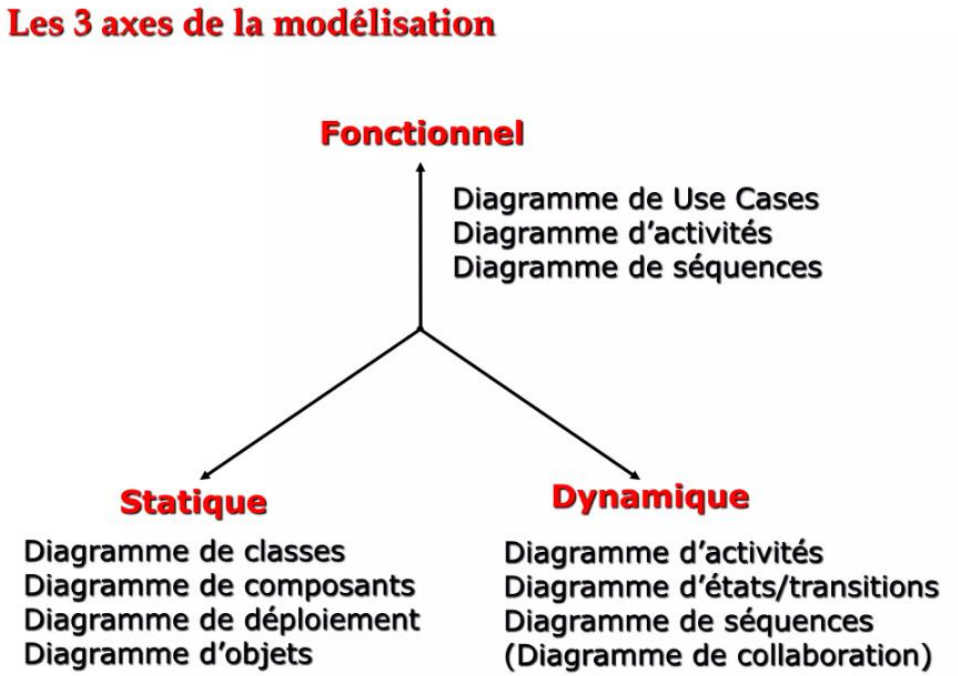
CHAPITRE 2. MODELISATION AVEC UML

2.2. Elements et mécanismes généraux

2.2.1 Les axes de la modélisation

Les concepteurs orientent leurs modélisations selon trois axes sur lesquels ils repartissent les diagrammes:

- **L'axe fonctionnel qui est utilise pour décrire ce que fait le système a réaliser,**
- **L'axe structurel et statique qui est relatif a sa structure,**
- **L'axe dynamique qui est relatif a la construction.**



CHAPITRE 2. MODELISATION AVEC UML

2.3. Les Diagrammes UML

1. Diagramme de cas d'utilisation

Il permet d'identifier les possibilités d'interaction entre le système et les acteurs (intervenants extérieurs au système), c 'est-à-dire toutes les fonctionnalités que doit fournir le système.

2. Diagramme de classe

Il représente les classes intervenant dans le système.

3. Diagramme d'objet

Il sert à représenter les instances de classes (objets) utilisées dans le système

4. Diagramme de composant

Il permet de montrer les composants du système d'un point de vue physique, tels qu'ils sont mis en œuvre (fichiers, bib liothèques, bases de données...)

5. 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 entre eux.

CHAPITRE 2. MODELISATION AVEC UML

2.3. Les Diagrammes UML

6. Diagramme de paquetages

Un paquetage étant un conteneur logique permettant de regrouper et d'organiser les éléments dans le modèle UML.
Le diagramme de paquetage sert à représenter les dépendances entre paquetages, c'est-à-dire les dépendances entre ensembles de définitions.

7. Diagramme de structure composite

Permet de décrire sous forme de boîte blanche les relations entre composants d'une classe.

8. Diagramme de séquence

Représentation séquentielle du déroulement des traitements et des interactions entre les éléments du système et/ou de ses acteurs.

9. Diagramme de communication

Représentation simplifiée d'un diagramme de séquence se concentrant sur les échanges de messages entre les objets

CHAPITRE 2. MODELISATION AVEC UML

2.3. Les Diagrammes UML

6. Diagramme de paquetages

Un paquetage étant un conteneur logique permettant de regrouper et d'organiser les éléments dans le modèle UML.

10. Diagramme global d'interaction

Permet de décrire les enchaînements possibles entre les scénarios préalablement identifiés sous forme de diagrammes d e séquences (variante du diagramme d'activités).

11. Diagramme de temps:

Permet de décrire les variations d'une donnée au cours du temps.

12. Diagramme d'états-transitions

Permet de décrire sous forme de machine à états finis le comportement du système ou de ses composants.

13. Diagramme d'activité

Permet de décrire sous forme de flux ou d'enchaînement d'activités le comportement du système ou de ses composants.