Chapitre 2 : Modélisation avec UML

1. Introduction

1.1. Un modèle… c'est quoi ?

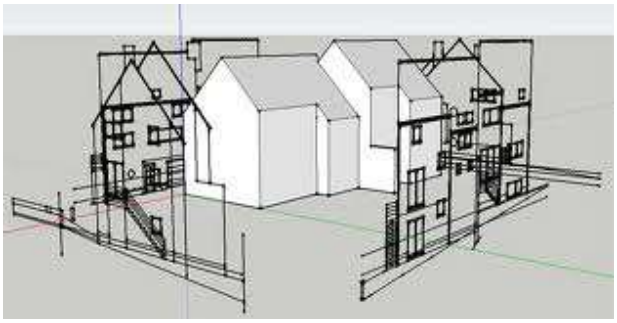
Un modèle est une représentation abstraite et simplifié d'une entité du monde réel en vue de le décrire, de l'expliquer ou de le prévoir.

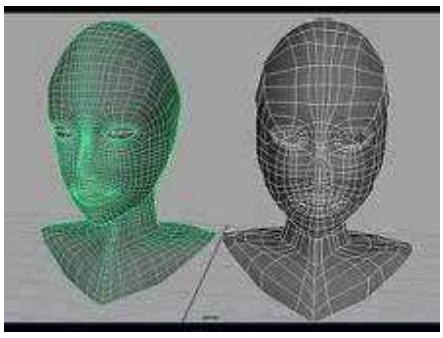
- Cette représentation peut être sous forme physique, graphique, mathématique ou verbale

$$MS_i = \frac{A}{\sum A}$$

$$Avec\ A_i = f(X_i, \beta)$$

$$et\quad X_{it} = f(X_{j,t-k}, \delta)$$





1.2. Pourquoi ? ...Intérêt de la modélisation

- Les modèles aident à visualiser un système existant ou futur (tel que l'on souhaite qu'il devienne)
- Les modèles permettent de spécifier la structure et le comportement d'un système
- Les modèles documentent les choix effectués

1.3. Comment ?... Langage de modélisation

Un langage de modélisation doit définir :

- La **sémantique** des concepts ;
- Une **notation** pour la représentation de concepts ;

- Des **règles** de construction et d'utilisation des concepts.

1.4. Quel langage ?... niveau de formalisation

Les langages sont différents au niveau de formalisation :

- **Langages formels** (Z, B, VDM) : le plus souvent mathématiques, au grand pouvoir d'expression et permettant des preuves formelles sur les spécifications ;
- **Langages semi-formels** (MERISE, UML...) : le plus souvent graphiques, au pouvoir d'expression moindre mais plus faciles d'emploi.

2. UML

UML (*Unified Methode Language*) est un langage de modélisation graphique à base de pictogrammes conçu pour fournir une méthode normalisée pour visualiser la conception d'un système. Il est couramment utilisé en développement logiciel et en conception orientée objet.

Elle résulte de la fusion de précédents langages de modélisation objet : Booch, OMT, OOSE. Principalement issu des travaux de Grady Booch, James Rumbaugh et Ivar Jacobson, UML est à présent un standard adopté par *l'Object Management Group* (OMG).

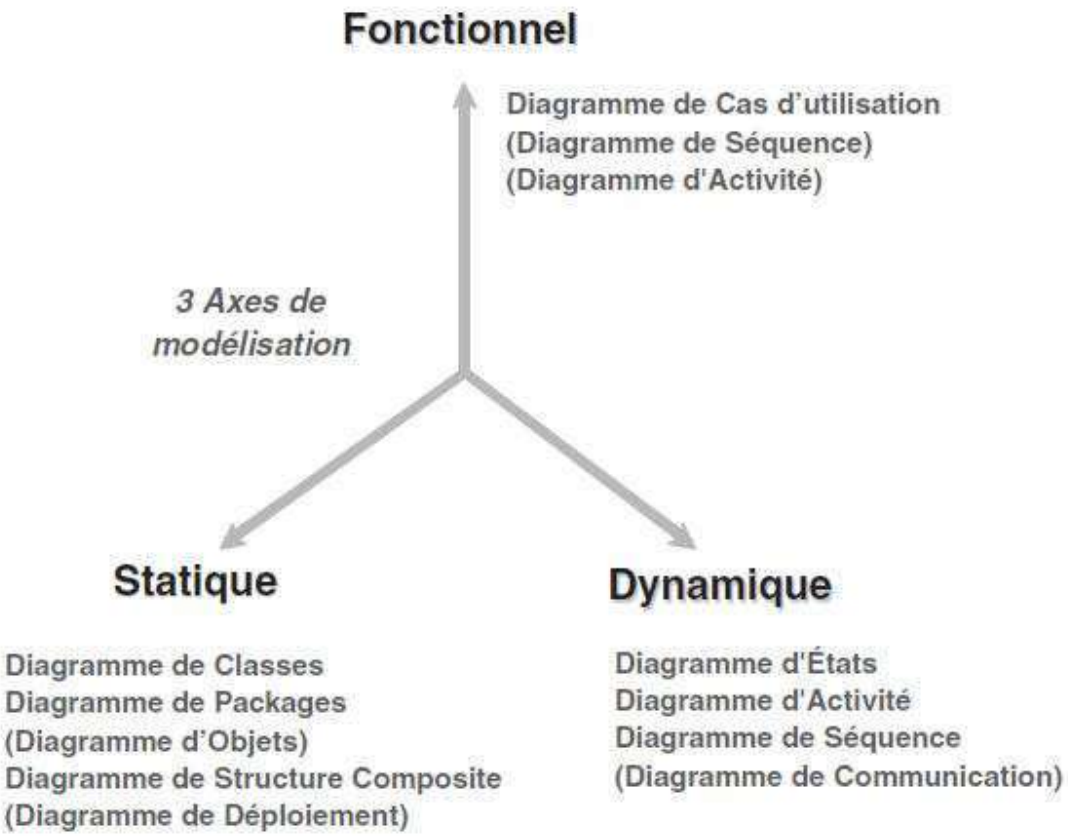
2.1. Historique

Au début des années 90, il existait une cinquantaine de méthodes d'analyse et de conception objet et la volonté de converger vers une méthode unifiée était déjà bien réelle. Le rapprochement entre ces méthodes a commencé par l'alliance entre James Rumbaugh et Grady Booch qui se sont retrouvés au sein de la société *Rational Software* et ont été ensuite rejoints par Ivar Jacobson en se donnant comme objectif de fusionner leur méthode et créer ainsi **UML**

- **1991** : première édition de Modélisation et conception orientées objet basée sur *OMT*,
- **1994** : *James Rumbaugh* rejoint *Rational* et travaille avec *Grady Booch* à la fusion des notations *OMT* et *Booch*.
- **1995** *: Ivar Jacobson* rejoint *Rational* et intègre *Objectory* au travail d'unification.
- **1997** : l'*OMG* accepte UML, proposé par *Rational*, comme standard de modélisation objet.
- **2001** : révision par l'*OMG* d'UML 1.
- **2004** : adoption d'UML 2.0

- **OMT** : Object Modeling Technique, une méthode de conception objet issue de la R&D de General Electric.
- **Rationnel software** : l'entreprise qui a développé UML.
- **Grady Booch** : l'auteur de la méthode d'analyse et conception objet qui porte son nom au début des 90.
- **Ivar Jacobson** : l'auteur de la méthode d'analyse et conception objet OMT.
- **Objectory** : la société crée par Ivar Jacobson pour commercialiser une méthode de développement orientée objets appelée Objectory Process dont le concept fort est l'utilisation des cas d'utilisation.
- **OMG** : Object Management Group, association américaine à but non lucratif créée en 1989 dont l'objectif est de standardiser et promouvoir le modèle objet sous toutes ses formes. L'OMG est notamment à la base des spécifications UML.

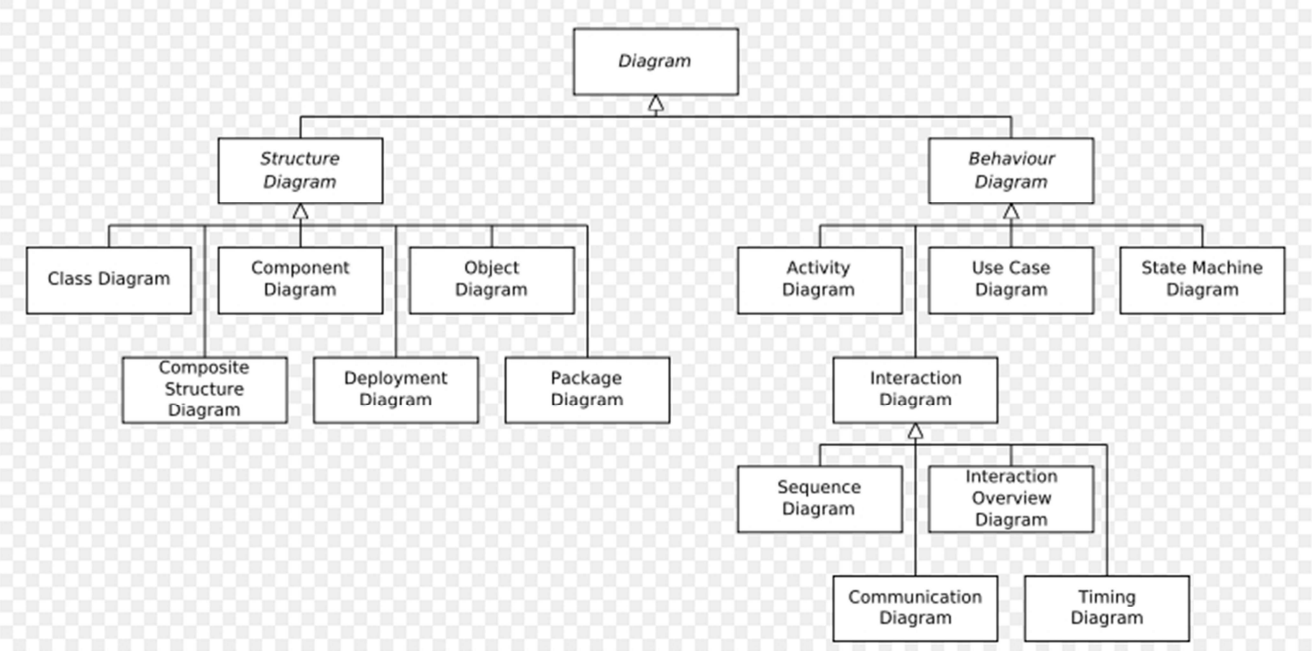
2.2. Les vues d'UML



UML concentre la modélisation autour de 3 axes :

- Fonctionnel : ce qui le système **fait**.
- Statique : ce qui le système **est**.
- Dynamique : comment le système **évolue**.

2.3. Les digrammes d'UML



UML 2.0 comporte 13 types de diagrammes représentant autant de *vues* distinctes pour représenter des concepts particuliers du système d'information.

Ces diagrammes, d'une utilité variable selon les cas, ne sont pas nécessairement tous produits à l'occasion d'une modélisation. En effet **UML est un langage** mais non pas une méthode (avec une description normative des étapes de la modélisation) : ses auteurs ont en effet estimé qu'il n'était pas opportun de définir une méthode en raison de la diversité des cas particuliers. Ils ont préféré se borner à définir un langage graphique qui permet de représenter et de communiquer les divers aspects d'un système d'information.

Les 13 diagrammes d'UML 2.0 se répartissent en deux grands groupes :

- **Les diagrammes structurels** : Ces diagrammes ont vocation à représenter l'aspect statique d'un système (classes, objets, composants…).
  - Diagramme de classe : Ce diagramme représente la description statique du système en intégrant dans chaque classe la partie dédiée aux données et celle consacrée aux traitements. C'est le diagramme pivot de l'ensemble de la modélisation d'un système.

- Diagramme d'objet : Le diagramme d'objet permet la représentation d'instances des classes et des liens entre instances.
- Diagramme de composant : Ce diagramme représente les différents constituants du logiciel au niveau de l'implémentation d'un système.
- Diagramme de déploiement : Ce diagramme décrit l'architecture technique d'un système avec une vue centrée sur la répartition des composants dans la configuration d'exploitation.
- Diagramme de paquetage : Ce diagramme donne une vue d'ensemble du système structuré en paquetage. Chaque paquetage représente un ensemble homogène d'éléments du système (classes, composants…).
- Diagramme de structure composite : Ce diagramme permet de décrire la structure interne d'un ensemble complexe composé par exemple de classes ou d'objets et de composants techniques. Ce diagramme met aussi l'accent sur les liens entre les sous-ensembles qui collaborent.

- **Les diagrammes de comportement** : Ces diagrammes représentent la partie dynamique d'un système réagissant aux événements et permettant de produire les résultats attendus par les utilisateurs. Sept diagrammes sont proposés par UML :
  - Diagramme des cas d'utilisation : Ce diagramme est destiné à représenter les besoins des utilisateurs par rapport au système. Il constitue un des diagrammes les plus structurants dans l'analyse d'un système.
  - Diagramme d'état-transition (machine d'état) : Ce diagramme montre les différents états des objets en réaction aux événements.
  - Diagramme d'activités : Ce diagramme donne une vision des enchaînements des activités propres à une opération ou à un cas d'utilisation. Il permet aussi de représenter les flots de contrôle et les flots de données.
  - Diagramme de séquence : Ce diagramme permet de décrire les scénarios de chaque cas d'utilisation en mettant l'accent sur la chronologie des opérations en interaction avec les objets.