Devoir Bases UML

**Object Modeling Technique (OMT)**

La méthode OMT a été inventée dans le centre de recherche et développement de Général Electric à la fin des années 80. Le principal ouvrage décrivant la méthode, dont l'auteur est James Rumbaugh, est paru en 1991.

Grady Booch et Ivan Jacobson ont développé d'autres méthodes orientées objet ; ce sont ces trois personnes, ont créé UML (Unified Modeling Language).

OMT fournit trois modèles principaux pour décrire les aspects statique, dynamique et fonctionnel d'un système d'information.

1. [**Le modèle objet**](http://d.martg.pagesperso-orange.fr/le.htm)**:** Il décrit la structure statique des objets, classes et associations. C'est une extension du modèle Entité-Relation vers les concepts d'objet (classe, agrégation, généralisation, héritage).
2. [**Le modèle dynamique**](http://d.martg.pagesperso-orange.fr/le1.htm) décrit les transformations d'un système dans le temps. Il est fondé sur des diagrammes d'états/transitions avec spécification.

Le modèle dynamique est employé pour spécifier les aspects "contrôle" d'un système.

1. [**Le modèle fonctionnel**](http://d.martg.pagesperso-orange.fr/le2.htm)décrit la transformation des valeurs des données dans le système. Il est fondé sur des diagrammes de flux de données classiques (DFD). Ces diagrammes sont utilisés pour spécifier des fonctions générales qui sont opérées sur les objets.

**La démarche d'analyse et de conception d'OMT.**

Comme la plupart des méthodes orientées objet, OMT propose une démarche en trois phases.

**L'étape d'**[**analyse**](http://d.martg.pagesperso-orange.fr/laphase.htm)

Elle concerne l'approche conceptuelle. Elle permet d'élaborer les trois modèles conceptuels (statique, dynamique et fonctionnel).

Cette étape est une description **de ce que** le système doit faire et non du "comment le faire".

**L'étape de** [**conception**](http://d.martg.pagesperso-orange.fr/laphase1.htm)

Elle permet de présenter l'architecture d'ensemble du système, par intégration des contraintes et des soucis de performance. La conception technique des objets donne une spécification détaillée de l'implémentation des objets, indépendante d'un environnement donné, mais compatible avec une technologie choisie;

C'est le niveau logique, il définit les structures de données et les algorithmes d'implémentation des classes.

**L'implémentation du Système.**

Il s'agit de l'écriture du code, en tant qu'extension du processus de conception .Elle varie en fonction des langages cibles et des systèmes de gestion de bases de données choisis, suivant qu'ils sont ou non orientés objet.

**Booch**

La **méthode Booch**[1](https://fr.wikipedia.org/wiki/M%C3%A9thode_Booch#cite_note-book-1) est une méthode de développement de logiciels pour la [programmation orientée objet](https://fr.wikipedia.org/wiki/Programmation_orient%C3%A9e_objet). Elle a été conçue par [Grady Booch](https://fr.wikipedia.org/wiki/Grady_Booch) qui l'a publiée en 1992 puis révisée en 1994. Elle se compose d'un [langage de modélisation](https://fr.wikipedia.org/wiki/Langage_de_mod%C3%A9lisation) graphique, d'un processus itératif de développement, et d'un ensemble de pratiques recommandées.

Booch est, avec [OMT](https://fr.wikipedia.org/wiki/Object_modeling_technique) et [OOSE](https://fr.wikipedia.org/wiki/OOSE), l'une des [méthodes d'analyse et de conception](https://fr.wikipedia.org/wiki/M%C3%A9thodes_d%27analyse_et_de_conception) [orientées objet](https://fr.wikipedia.org/wiki/Orient%C3%A9_objet) à l'origine d'[UML](https://fr.wikipedia.org/wiki/Unified_Modeling_Language), [Grady Booch](https://fr.wikipedia.org/wiki/Grady_Booch), étant l'un des pères du langage [UML](https://fr.wikipedia.org/wiki/Unified_Modeling_Language) avec [Ivar Jacobson](https://fr.wikipedia.org/wiki/Ivar_Jacobson), fondateur de la méthode [OOSE](https://fr.wikipedia.org/wiki/Object_Oriented_Software_Engineering) et [James Rumbaugh](https://fr.wikipedia.org/wiki/James_Rumbaugh), fondateur de la méthode [OMT](https://fr.wikipedia.org/wiki/Object_Modeling_Technique). La méthode Booch comprend six types de diagrammes: la classe, l'objet, la transition d'état, l'interaction, le module et le processus.

**Object Oriented Software Engineering (OOSE)**

**OOSE** est un langage de [modélisation objet](https://fr.wikipedia.org/wiki/Mod%C3%A9lisation_objet) créé par [Ivar Jacobson](https://fr.wikipedia.org/wiki/Ivar_Jacobson). OOSE est une méthode pour l’analyse initiale des usages de logiciels, basée sur les « cas d’utilisation » et le cycle de vie des logiciels.

La méthode OOSE est basée sur cinq modèles et trois types d’objets fondés par Ivar Jacobson en 1992

Les cinq modèles sont :

* le modèle des besoins est un modèle qui se situe au niveau de l’interaction entre l’individu et l’environnement ;
* le modèle d’analyse permet de définir les différentes relations entre les objets, les acteurs et le système ;
* le modèle de conception permet de modéliser ces relations à travers différents diagrammes ;
* le modèle d’implémentation permet le passage au code ;
* le modèle de test basé sur les essais d’actions et d’interactions du système.

Les trois types d’objets sont :

* les entités représentant un objet matériel ou immatériel ;
* des contrôles permettant différentes vérifications ;
* des interfaces représentant des classes abstraites, sans attribut et dont les méthodes sont abstraites et publiques.

**Différentes versions d’UML et diagrammes**

| Version | Année | Diagrammes structurels disponibles | Diagrammes comportementaux disponibles | Nombre total de diagrammes |
| --- | --- | --- | --- | --- |
| 0.8 | 1995 | * Diagramme de classes ; * Diagramme d’objets ; * Diagramme de déploiement ; * Diagramme de composants. | * Diagramme de cas d’utilisation ; * Diagramme d’Etats-transitions ; * Diagrammes d’activités ; * Diagrammes de séquence ; * Diagramme de collaboration. | 9 |
| 0.9 | 1996 | * Diagramme de classes ; * Diagramme d’objets ; * Diagramme de déploiement ; * Diagramme de composants. | * Diagramme de cas d’utilisation ; * Diagramme d’Etats-transitions ; * Diagrammes d’activités ; * Diagrammes de séquence ; * Diagramme de collaboration. | 9 |
| 0.9.1 | 1996 | * Diagramme de classes ; * Diagramme d’objets ; * Diagramme de déploiement ; * Diagramme de composants. | * Diagramme de cas d’utilisation ; * Diagramme d’Etats-transitions ; * Diagrammes d’activités ; * Diagrammes de séquence ; * Diagramme de collaboration. | 9 |
| 1.0 | 1997 | * Diagramme de classes ; * Diagramme d’objets ; * Diagramme de déploiement ; * Diagramme de composants. | * Diagramme de cas d’utilisation ; * Diagramme d’Etats-transitions ; * Diagrammes d’activités ; * Diagrammes de séquence ; * Diagramme de collaboration. | 9 |
| 1.1 | 1997 | * Diagramme de classes ; * Diagramme d’objets ; * Diagramme de déploiement ; * Diagramme de composants. | * Diagramme de cas d’utilisation ; * Diagramme d’Etats-transitions ; * Diagrammes d’activités ; * Diagrammes de séquence ; * Diagramme de collaboration. | 9 |
| 1.2 | 1998 | * Diagramme de classes ; * Diagramme d’objets ; * Diagramme de déploiement ; * Diagramme de composants. | * Diagramme de cas d’utilisation ; * Diagramme d’Etats-transitions ; * Diagrammes d’activités ; * Diagrammes de séquence ; * Diagramme de collaboration. | 9 |
| 1.3 | 2000 | * Diagramme de classes ; * Diagramme d’objets ; * Diagramme de déploiement ; * Diagramme de composants. | * Diagramme de cas d’utilisation ; * Diagramme d’Etats-transitions ; * Diagrammes d’activités ; * Diagrammes de séquence ; * Diagramme de collaboration. | 9 |
| 1.4 | 2001 | * Diagramme de classes ; * Diagramme d’objets ; * Diagramme de déploiement ; * Diagramme de composants. | * Diagramme de cas d’utilisation ; * Diagramme d’Etats-transitions ; * Diagrammes d’activités ; * Diagrammes de séquence ; * Diagramme de collaboration. | 9 |
| 1.5 | 2003 | * Diagramme de classes ; * Diagramme d’objets ; * Diagramme de déploiement ; * Diagramme de composants. | * Diagramme de cas d’utilisation ; * Diagramme d’Etats-transitions ; * Diagrammes d’activités ; * Diagrammes de séquence ; * Diagramme de collaboration. | 9 |
| 2.0 | 2005 | * Diagramme de classes ; * Diagramme d’objets ; * **Diagramme de paquetage ;** * Diagramme de déploiement ; * Diagramme de composants * **Diagramme de structure composite.** | * Diagramme de cas d’utilisation ; * Diagramme d’Etats-transitions ; * Diagrammes d’activités ; * **Diagramme global d’interaction ;** * **Diagramme de temps ;** * Diagrammes de séquence ; * **Diagramme de communication** | 13 |
| 2.1.1 | 2007 | * Diagramme de classes ; * Diagramme d’objets ; * Diagramme de paquetage ; * Diagramme de déploiement ; * Diagramme de composants * Diagramme de structure composite. | * Diagramme de cas d’utilisation ; * Diagramme d’Etats-transitions ; * Diagrammes d’activités ; * Diagramme global d’interaction ; * Diagramme de temps ; * Diagrammes de séquence ; * Diagramme de communication | 13 |
| 2.1.2 | 2007 | * Diagramme de classes ; * Diagramme d’objets ; * Diagramme de paquetage ; * Diagramme de déploiement ; * Diagramme de composants * Diagramme de structure composite. | * Diagramme de cas d’utilisation ; * Diagramme d’Etats-transitions ; * Diagrammes d’activités ; * Diagramme global d’interaction ; * Diagramme de temps ; * Diagrammes de séquence ; * Diagramme de communication | 13 |
| 2.2 | 2009 | * Diagramme de classes ; * Diagramme d’objets ; * Diagramme de paquetage ; * **Diagramme de profils ;** * Diagramme de déploiement ; * Diagramme de composants * Diagramme de structure composite. | * Diagramme de cas d’utilisation ; * Diagramme d’Etats-transitions ; * Diagrammes d’activités ; * Diagramme global d’interaction ; * Diagramme de temps ; * Diagrammes de séquence ; * Diagramme de communication | 14 |
| 2.3 | 2010 | * Diagramme de classes ; * Diagramme d’objets ; * Diagramme de paquetage ; * Diagramme de profils ; * Diagramme de déploiement ; * Diagramme de composants * Diagramme de structure composite. | * Diagramme de cas d’utilisation ; * Diagramme d’Etats-transitions ; * Diagrammes d’activités ; * Diagramme global d’interaction ; * Diagramme de temps ; * Diagrammes de séquence ; * Diagramme de communication | 14 |
| 2.4 | 2011 | * Diagramme de classes ; * Diagramme d’objets ; * Diagramme de paquetage ; * Diagramme de profils ; * Diagramme de déploiement ; * Diagramme de composants * Diagramme de structure composite. | * Diagramme de cas d’utilisation ; * Diagramme d’Etats-transitions ; * Diagrammes d’activités ; * Diagramme global d’interaction ; * Diagramme de temps ; * Diagrammes de séquence ; * Diagramme de communication | 14 |
| 12.4.1 | 2011 | * Diagramme de classes ; * Diagramme d’objets ; * Diagramme de paquetage ; * Diagramme de profils ; * Diagramme de déploiement ; * Diagramme de composants * Diagramme de structure composite. | * Diagramme de cas d’utilisation ; * Diagramme d’Etats-transitions ; * Diagrammes d’activités ; * Diagramme global d’interaction ; * Diagramme de temps ; * Diagrammes de séquence ; * Diagramme de communication | 14 |
| 2.5 | 2015 | * Diagramme de classes ; * Diagramme d’objets ; * Diagramme de paquetage ; * Diagramme de profils ; * Diagramme de déploiement ; * Diagramme de composants * Diagramme de structure composite. | * Diagramme de cas d’utilisation ; * Diagramme d’Etats-transitions ; * Diagrammes d’activités ; * Diagramme global d’interaction ; * Diagramme de temps ; * Diagrammes de séquence ; * Diagramme de communication | 14 |
| 2.5.1 | 2017 | * Diagramme de classes ; * Diagramme d’objets ; * Diagramme de paquetage ; * Diagramme de profils ; * Diagramme de déploiement ; * Diagramme de composants * Diagramme de structure composite. | * Diagramme de cas d’utilisation ; * Diagramme d’Etats-transitions ; * Diagrammes d’activités ; * Diagramme global d’interaction ; * Diagramme de temps ; * Diagrammes de séquence ; * Diagramme de communication | 14 |