Introduction à UML

Quasiment un extrait de l'article de wikipédia sur UML

Quasiment un extrait de l'article de wikipédia sur UML

UML (de l'anglais *Unified Modeling Language*), ou Langage de modélisation unifié, est un langage de modélisation graphique à base de diagrammes. Il est utilisé en développement logiciel, et en conception orientée objet.

UML est couramment utilisé dans les projets logiciels.

UML est l'accomplissement 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 défini par l'Object Management Group (OMG).

La dernière version diffusée par l'OMG est UML 2.5 bêta 2 depuis septembre 2013

Histoire

Les méthodes objets ont commencé à émerger au début des années 80, ces méthodes avaient pour but de remplacer les méthodes structurée et fonctionnelles.

Plusieurs méthodes objets (une cinquantaine) ont fait leur apparition au début des années 90. Ces méthodes s'orientant sur l'abstraction des composants matériels, se basent sur des notions de classe, d'association, de partition en sous-système et autour de l'étude de l'interaction entre utilisateur et le système. Les principaux auteurs de ces méthodes sont James Rumbaugh, Grady Booch et Ivar Jacobson. Parmi ces méthodes, deux se sont principalement établies : la méthode de Booch et la méthode OMT (Object Modeling Technique) de James Rumbaugh,

James Rumbaugh et Grady Booch s'unissent en 1994 afin d'unifier leur travaux. Ils proposent ainsi « Unified Method », un an plus tard, Ivar Jacobson (créateur des « cas d'utilisation »), les rejoindra. Les auteurs de la méthode unifiée sortent, en 1995, un document intitulé Unified Method V0.8.

En 1996, la méthode change de nom et se transforme en UML (Unified Modeling Language for Object-Oriented Development). Un consortium de grandes entreprises se crée (Microsoft, IBM, Oracle, etc.) permettant de faire évoluer la méthode à la version 1.0 d'UML. En janvier 1997, UML est devenu un standard OMG.

UML

Les **14** diagrammes UML sont dépendants hiérarchiquement et se complètent, de façon à permettre la modélisation d'un projet tout au long de son cycle de vie.

Diagrammes structurels ou statiques

Les diagrammes structurels ou statiques (Structure Diagram) rassemblent :

- **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** (Deployment diagram) : 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.
- **Diagramme des paquetages** (Package diagram) : 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.
- **Diagramme de structure composite** (Composite Structure Diagram) : depuis UML 2.x, permet de décrire sous forme de boîte blanche les relations entre composants d'une classe.
- **Diagramme de profils** (Profile diagram) : depuis UML 2.2, permet de spécialiser, de personnaliser pour un domaine particulier un meta-modèle de référence d'UML.

Diagrammes comportementaux

Les diagrammes comportementaux (Behavior Diagram) rassemblent :

- **Diagramme des cas d'utilisation** (use-cases ou Use Case Diagram) : 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.
- **Diagramme états-transitions** (State Machine Diagram) : permet de décrire sous forme de machine à états finis le comportement du système ou de ses composants.
- Diagramme d'activité (Activity Diagram) : permet de décrire sous forme de flux ou d'enchaînement d'activités le comportement du système ou de ses composants.

Diagrammes d'interaction ou dynamiques

Les diagrammes d'interaction ou dynamiques (Interaction Diagram) rassemblent :

- **Diagramme de séquence** (Sequence Diagram) : représentation séquentielle du déroulement des traitements et des interactions entre les éléments du système et/ou de ses acteurs.
- **Diagramme de communication** (Communication Diagram) : depuis UML 2.x, représentation simplifiée d'un diagramme de séquence se concentrant sur les échanges de messages entre les objets.
- **Diagramme global d'interaction** (Interaction Overview Diagram) : depuis UML 2.x, permet de décrire les enchaînements possibles entre les scénarios préalablement identifiés sous forme de diagrammes de séquences (variante du diagramme d'activité).
- **Diagramme de temps** (Timing Diagram) : depuis UML 2.3, permet de décrire les variations d'une donnée au cours du temps.