Conception, Projet

Cours 8 - Conception détaillée - UML p.1

Table des matières

Introduction à l'UML

Nous allons passer en revue les différents types de diagrammes UML, les éléments qui les composent, les relations entre ces éléments, et nous allons également explorer les outils et les logiciels disponibles pour la modélisation UML.

Introduction à l'UML

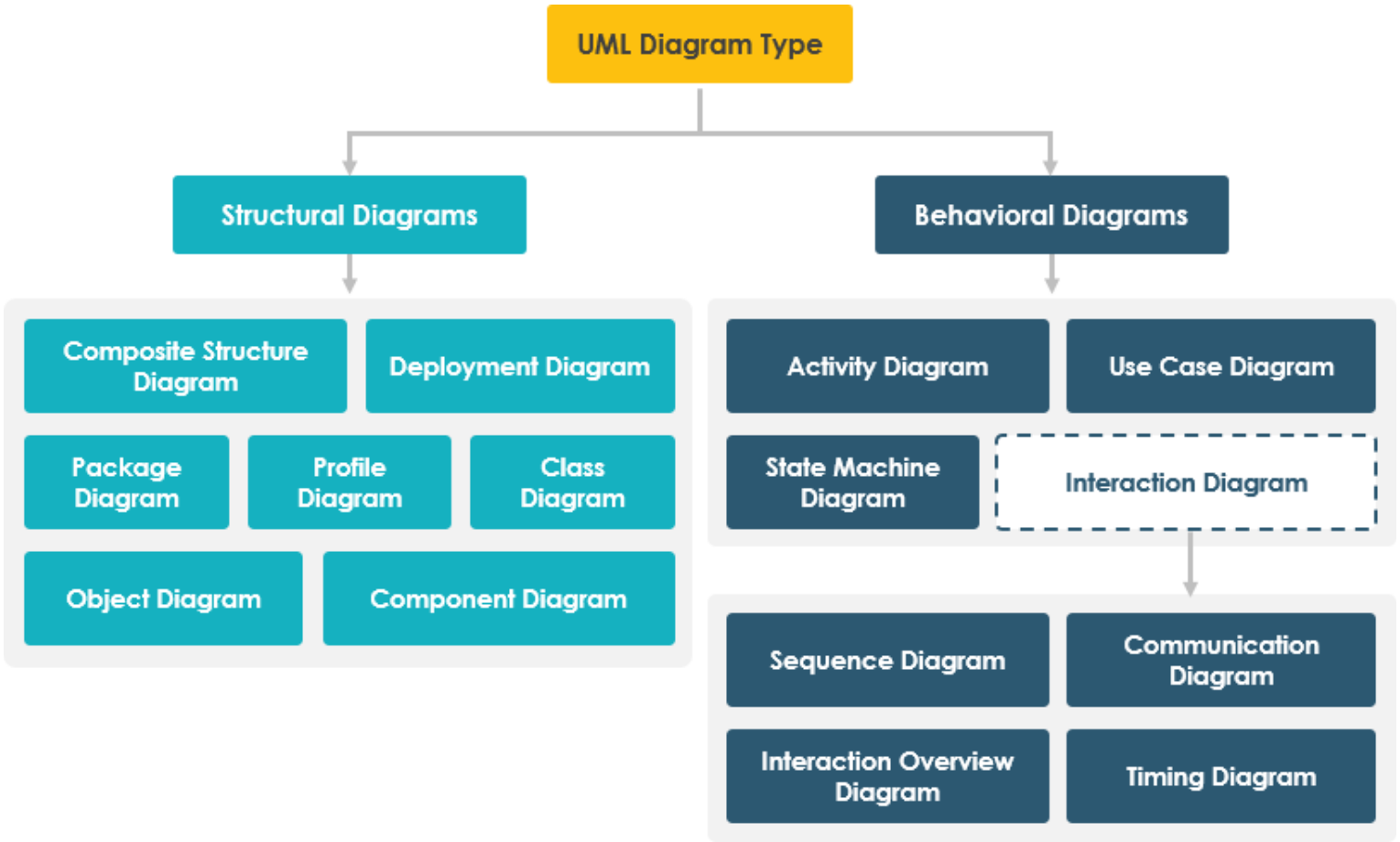

Figure: `https://www.visual-paradigm.com/guide/uml-unified-modeling-language/overview-of-the-14-uml-diagram-types/`

Définition de l'UML

L'UML (Unified Modeling Language) est un langage de modélisation graphique utilisé pour représenter visuellement les différents aspects d'un système logiciel.

Il a été développé pour unifier et standardiser les notations et les méthodes de modélisation utilisées dans le domaine du génie logiciel.

L'UML est largement utilisé pour spécifier, concevoir et documenter des systèmes logiciels complexes.

Il permet de décrire les différents éléments qui composent un système, ainsi que les relations et les interactions entre ces éléments. Il peut également être utilisé pour représenter des processus métier, des systèmes matériels, des réseaux, etc.

Définition de l'UML

Les avantages de l'utilisation de l'UML sont nombreux.

Il permet une meilleure compréhension du système, une communication plus claire entre les différents acteurs du projet, une documentation précise et complète du système, une détection précoce d'éventuelles erreurs ou incohérences, et une meilleure gestion du projet en général.

L'UML a connu une évolution depuis sa première version en 1997, et de nouvelles versions ont été publiées régulièrement pour prendre en compte les évolutions du domaine du génie logiciel et les besoins des utilisateurs.

Objectifs et avantages de l'UML

Les objectifs de l'UML sont les suivants :

- Fournir un langage de modélisation standardisé et commun à l'ensemble de l'industrie du logiciel, afin de faciliter la communication entre les différents acteurs d'un projet (développeurs, architectes, clients, etc.).
- Permettre une compréhension plus facile et plus rapide du système logiciel, grâce à des diagrammes clairs et précis.
- Faciliter la gestion du développement logiciel, en permettant une analyse approfondie des besoins, une conception efficace du système et une validation préalable des choix d'architecture.

Objectifs et avantages de l'UML

Les avantages de l'utilisation de l'UML sont multiples :

- Amélioration de la qualité du code produit, grâce à une modélisation précise et rigoureuse.
- Réduction des coûts et des délais de développement, grâce à une meilleure compréhension des besoins et à une conception plus efficace du système.
- Facilitation de la communication entre les différents acteurs du projet, en particulier entre les équipes techniques et les équipes métiers.
- Possibilité de documenter et de maintenir plus facilement le système logiciel, grâce à une documentation graphique claire et précise.
- Facilitation de l'évolution et de la maintenance du système logiciel, grâce à une modélisation précise de l'architecture et des interactions entre les différents éléments du système.

Le design driven development (DDD)

Le design driven development (DDD) est une approche de développement logiciel qui met l'accent sur la conception et la modélisation dès les premières étapes du projet.

Le DDD reconnaît l'importance de la modélisation en tant qu'outil pour comprendre, organiser et optimiser le système.

Dans cette approche, les développeurs créent d'abord des modèles de conception pour représenter les aspects clés du système, tels que la structure, les comportements, et les interactions.

Ces modèles servent ensuite de base pour le développement du code source.

Le design driven development (DDD)

Le DDD favorise la qualité et la maintenabilité du logiciel, car il encourage une réflexion approfondie sur l'architecture du système avant de commencer à écrire le code.

De plus, la modélisation dans le cadre du DDD permet de détecter les problèmes potentiels en amont et d'assurer une meilleure communication entre les membres de l'équipe de développement.

UML permet de fournir des outils afin d'adopter des apparoches DDD au cours du développement.

Les différents types principaux de diagrammes UML

L'UML propose deux grandes familles de diagrammes : les diagrammes structurels et les diagrammes comportementaux.

Les différents types principaux de diagrammes UML

Les diagrammes structurels sont utilisés pour modéliser la structure statique d'un système, c'est-à-dire sa composition, ses entités, leurs propriétés et leurs relations.

Les différents types principaux de diagrammes UML

Les diagrammes comportementaux sont utilisés pour modéliser le comportement dynamique d'un système, c'est-à-dire les interactions entre les éléments et les processus qui s'exécutent en leur sein.

Les différents types secondaires de diagrammes UML - Structurels

Voici les types secondaires de diagrammes UML pour ceux de type structurels :

- Composite (Composite)
- Deploiement (Deployment)
- Packet (Package)
- Profil (Profile)
- Classe (Class) [le plus populaire]
- Objet (Object)
- Composant (Component)

Les différents types secondaires de diagrammes UML - Comportementaux

Voici les types secondaires de diagrammes UML pour ceux de type comportementaux :

- Activité (Activity)
- Cas (Use Case)
- Machine d'état (State Machine)
- Intéraction (Interaction)
  - Séquence (Sequence)
  - Communication (Communication)
  - Intéraction (Interaction)
  - De moment (Timing)

Les éléments graphiques (morceaux/items) de UML

Il existe des items qu'on peut glisser, copier/coller et utiliser pour représenter les diagrammes.

Voici quelques items :
Classe : Représente un type d'objet avec des attributs et des méthodes. Les classes sont généralement représentées par des rectangles avec trois compartiments : le nom de la classe, les attributs et les méthodes.

Objet : Représente une instance d'une classe. Les objets sont souvent représentés par des rectangles avec le nom de l'objet souligné, suivi du nom de la classe entre deux points.

Interface : Représente un ensemble d'opérations déclarées mais non implémentées. Les interfaces sont généralement représentées par un cercle avec le nom de l'interface à l'intérieur ou un rectangle avec le mot-clé "interface".

Les éléments graphiques (morceaux/items) de UML

Association : Représente une relation entre deux classes ou objets. Les associations sont représentées par des lignes reliant les éléments concernés.

Généralisation (héritage) : Représente une relation “ est un “ entre deux classes, où une classe hérite des attributs et des opérations d'une autre classe. Les généralisations sont représentées par des lignes avec des flèches triangulaires pointant vers la classe parente.

Agrégation : Représente une relation “ a un “ entre deux classes, où une classe est composée d'autres classes. Les agrégations sont représentées par des lignes avec un losange vide du côté de la classe qui contient les autres classes.

Les éléments graphiques (morceaux/items) de UML

Composition : Semblable à l'agrégation, mais avec une relation plus forte entre les classes. Les compositions sont représentées par des lignes avec un losange plein du côté de la classe qui contient les autres classes.

Dépendance : Représente une relation où un élément dépend d'un autre élément. Les dépendances sont représentées par des lignes en pointillés avec une flèche pointant vers l'élément dont dépend l'autre.

État : Représente une condition particulière d'un objet pendant son cycle de vie. Les états sont généralement représentés par des rectangles arrondis avec le nom de l'état à l'intérieur.

Les éléments graphiques (morceaux/items) de UML

Transition : Représente le changement d'un état à un autre. Les transitions sont représentées par des lignes avec des flèches pointant vers l'état suivant.

Activité : Représente une action ou un ensemble d'actions effectuées par un objet. Les activités sont généralement représentées par des rectangles arrondis avec le nom de l'activité à l'intérieur.

Acteur : Représente une entité externe qui interagit avec le système, comme un utilisateur ou un autre système. Les acteurs sont généralement représentés par des bonshommes simplifiés avec le nom de l'acteur sous eux.

Bibliographie

- https://www.visual-paradigm.com/guide/uml-unified-modeling-language/overview-of-the-14-uml-diagram-types/