UML – Unified Modeling Language

# Introduction et définitions

* Outil de structuration, organisation et modélisation de la programmation orientée objet, des classes et de leurs relations, sous forme de dessins / diagrammes.
* Utilisé autant du côté client que conception & dev.
* UML est un langage standardisé, regroupant plusieurs méthodologies (qui elles ne peuvent fusionner de par les différents corps de métier).
* Analyse vs Conception : Le diagramme d’analyse reste très épuré, peu détaillé, vision globale. En revanche une fois l’analyse validée, on entre dans le détail pour la conception, beaucoup plus spécifique et détaillée, on se rapproche plus du pseudo-code.
* Il est possible de générer du code à partir d’un diagramme ou, à l’inverse, d’extraire un diagramme à partir du code.

# Différentes vues et diagrammes

* Toutes les vues sont complémentaires, mais comportent des informations et des niveaux de détails différents.
* Cela peut également correspondre à un découpage en fonction des aspects fonctionnels ou architecturaux.

## Vue fonctionnelle : diagrammes de cas d’utilisation (use case) et de séquences

* Plutôt orienté client, voir les interactions/liens entre personnes, systèmes, utilisateurs.
* Nature du lien implicite ou décrite (note)
* Liens système-système, acteur-acteur, acteur-système
* Généralisation, extends, include

## Vue structurelle ou statique : diagrammes de classes et de packages

* Relations entre classes + action + **effectif n..m**
* Associations (agrégation & composition) nommés avec rôle, héritages
* Sens monodirectionnel A – fait 🡪 B, ref de B dans A. Bidirectionnel A 🡨fait 🡪 B, ref de A dans B et ref de B dans A, les deux se connaissent.
* Groupement de classes en package + relations/dépendances entre les packages

## Vue dynamique : diagrammes d’état et d’activité

* Décrit l’évolution des objets complexes, du programme, de leur vie et de leur mort.