1-**Introduction:**

UML (*Unified Modeling Language*, que l'on peut traduire par "*langage de modélisation unifié*) est une notation permettant de modéliser un problème de façon standard. Ce langage est né de la fusion de plusieurs méthodes existantes auparavant, et est devenu désormais la référence en terme de modélisation objet, à un tel point que sa connaissance est souvent nécessaire pour obtenir un poste de développeur objet.

2-**Présentation d'UML :**

UML est un langage de modélisation graphique et textuel destiné à comprendre et à décrire des besoins, spécifier et documenter des systèmes, esquisser des architectures logicielles, concevoir des solutions et communiquer des points de vue. En effet UML est un langage avec une syntaxe et des règles bien définies qui tentent à réaliser les buts décrits grâce à une représentation graphique formée de diagrammes et une modélisation textuelle qui vient enrichir la représentation graphique.

3-**Historique:**

En Octobre 1994 **G.Booch** et **J.Rumbaugh** ont décidé de travailler ensemble pour unifier leurs méthodes au sein de la société Rational Software un an après, **I.Jacobsoll** a rejoint Rational Software pour travailler sur l'unification les travaux sur ce langage ont continué avec son adoption par de grands acteurs industriels dont Microsoft, Oracle et Unisys. Ce travail a abouti en janvier de l'année 1999 à UML 1.0.

Le langage à été soumis par Rational Software et ses partenaires à l'**OMG** (Object Management Group) comme réponse à un appel d'offre sur la standardisation des langages de modélisation en Septembre 1997, et qui a été accepté à l'unanimité deux mois plus tard en Novembre 1997 dans sa version 1.1 qui devient de ce fait un standard.

Des transformations continuelles ne cessent d'être effectuées pour supprimer les incohérences, apporter des améliorations et ajouter de nouveaux concepts d'où l'apparition de la version 1.2 en Juin 1998 cette dernière n'a introduit aucun ajout d'ordre techniques, ces modifications par rapporta la version 1.1 porte uniquement sur un remaniement de la forme.

La version 1.3 apparue en Juin 1999 a apporté de nombreux changements qu'il s'agisse de corrections ou d'ajouts (modification des associations entre cas d'utilisation, simplification des stéréotypes, changement d'éléments de graphe d'activités et des automates d'états,..).

D'autres versions sont apparues après la version 1.3 en Septembre 2001. La version 1.4 a été publiée, suivie par UML 1.5 en 2003. La version d'UML en cours à la fin 2004 est UML 2.O et les travaux d'amélioration se poursuivent. **[UML2.0]**.

UML est donc non seulement un outil intéressant mais une norme qui s'impose en technologie à objets et à la quelle se sont rangés tous les grands acteurs du domaine, acteurs qui ont d'ailleurs contribué à son élaboration.

4- **Modélisation avec UML :**

Le métamodèle UML fournit une panoplie d'outils permettant de représenter l'ensemble des éléments du monde objet (classes, objets, ...) ainsi que les liens qui les relie.

Toutefois, étant donné qu'une seule représentation est trop subjective, UML fournit un moyen astucieux permettant de représenter diverses projections d'une même représentation grâce aux vues.
Une vue est constituée d'un ou plusieurs diagrammes.

On distingue deux types de vues :

Les vues statiques, c'est-à-dire représentant le système physiquement

- Diagrammes d'objets
- Diagrammes de classes
- Diagrammes de composants
- Diagrammes de déploiement

Les vues dynamiques, montrant le fonctionnement du système

- Diagrammes de séquence
- Diagrammes de communication
- Diagrammes d'états-transitions
- Diagrammes d'activités
- Diagrammes de cas d'utilisation

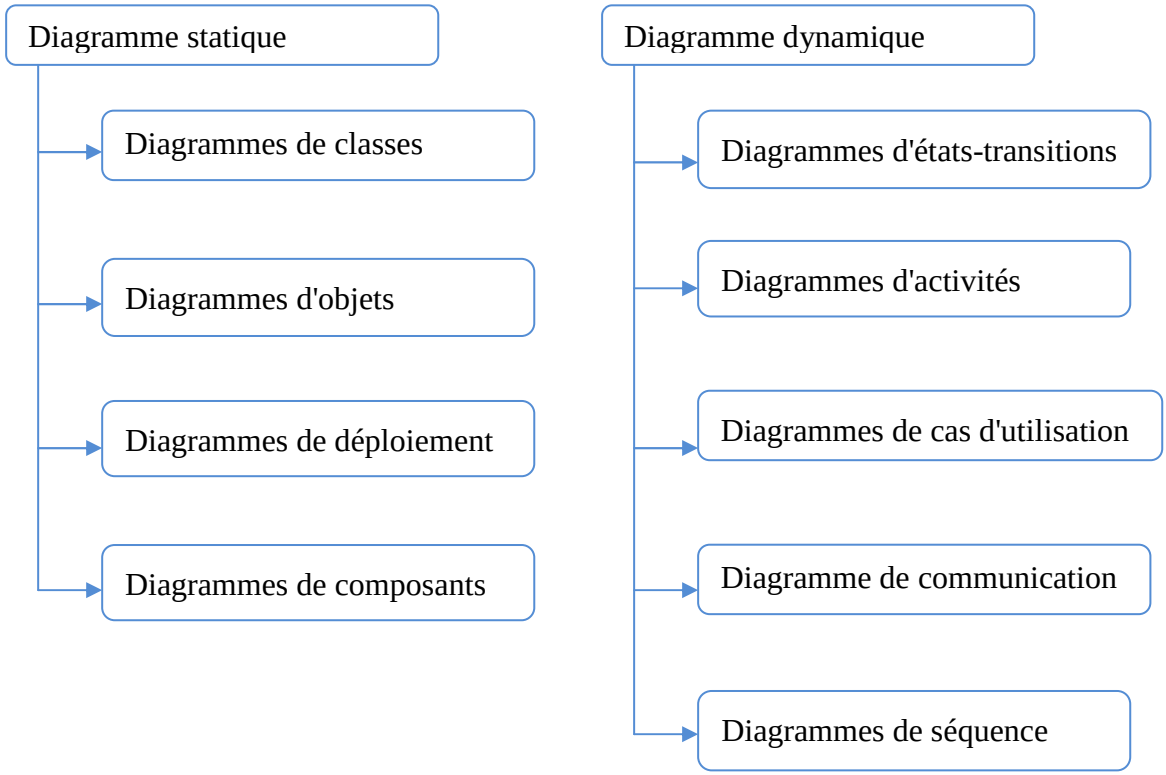


Figure 1.2.1 : Diagrammes constituant d'UML par catégorie

a) <u>Diagramme des classes:</u>

Le diagramme des classes est une représentation statique du système d'information. Il permet aux concepteurs de visualiser les relations inter-classes dans un système.

Une classe décrit un ensemble d'éléments.

Chaque relation entre classes est définit par des associations.

Représentation d'une classe UML.

Les classes en UML sont représentées par un rectangle avec des séparations :

Exemple :

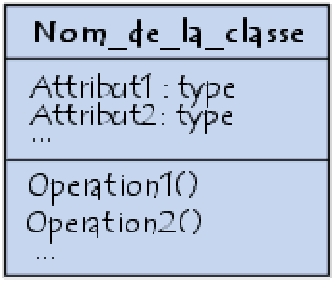


Figure 1.2.2 : Exemple d'une classe UML

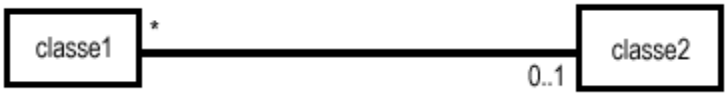
Les associations

Un diagramme des classes est composé de plusieurs classes qui sont reliées entre elles. Des liens sont alors tracés afin de créer ces relations.
On les appelle des associations.

Représentation d'une association entre 2 classes

La multiplicité des relations :

La multiplicité se précise sur les extrémités des associations :



La multiplicité associer à une terminaison d'une association indique le NBR d'objet apparaissant dans l'autre extrémité

a) Diagramme Cas d'utilisation :

L'étude de cas d'utilisation a pour objectif de déterminer ce que chaque acteur attend du système. La détermination des besoins est basée sur la représentation de l'interaction entre l'acteur et le système. A ce prospect de la modélisation, les interactions représentent les principaux événements qui se produisent dans le domaine de l'application. Plus tard, lors de la conception, ces événements sont traduits en messages qui déclenchent des opérations.

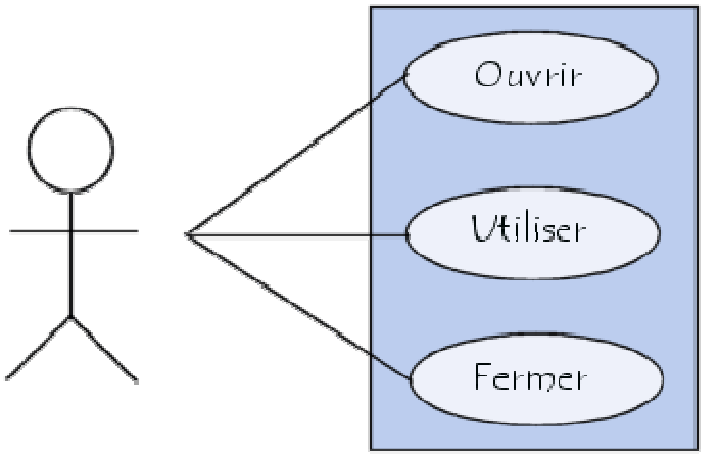


Figure 1.2.3 : Exemple « Diagramme de cas d'utilisation UML »

b) Diagramme séquence :

IL permet d'étudier les interactions entre les objets et constitue une partie dynamique de système d'information.
IL montre d'une façon séquentielle les envois des messages qui interviennent entre les objets, il peut également montrer les flux des données :

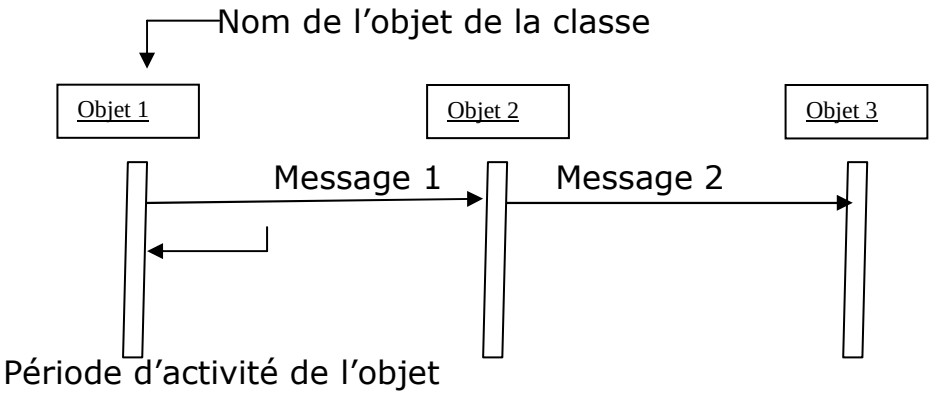


Figure 1.2.4 : représentation diagramme de séquence

c) Diagramme d'état transition :

Le diagramme d'état transition décrit la vie d'un objet en terme d'événement déclenchant des modifications de l'état de celui-ci, il identifie à la fois les événements externe et interne.

Le passage d'un état à un autre s'appelle transition

d) Les diagrammes de déploiement :

Qui représentent le déploiement des composants sur les dispositifs matériels,

e) Les diagrammes d'activités :

Qui représentent le comportement d'une opération en terme d'action.

f) Les diagrammes de communication :

Les diagrammes de communication qui donne une représentation spatiale des objets, des liens des interactions, il est équivalent au diagramme au diagramme de séquence

g) Les digrammes d'objets :

Qui représentent les objets et leur relation et correspondent à des diagrammes de collaboration simplifiés, sans représentation des envois de message.

h) Les diagrammes de composants :

Qui représentent la partie physiques d'une application,

Dans notre projet on a utilisé les diagrammes suivants :

- Diagramme de cas d'utilisation
- Diagramme de classe
- Diagramme de déploiement
- Diagramme de communication
- Diagramme de séquence
- Diagramme état/ transition
- Diagramme de composant
- Diagramme Activité