1Cours en Systèmes d'information, méthodes avancées
Chapitre 1 : Notions de base

1. Introduction

Un système d'information (SI) est un ensemble de moyens matériels, logiciels, humains pour gérer l'information au sein de l'entreprise.

Le SI d'une entreprise assure quatre principales fonctions :

- **Acquisition** de l'information de l'environnement interne ou externe.
- **Transformation** de l'information par traitement ou autre
- **Stockage** des données dans des bases de données ou autres
- **Diffusion** de l'information au sein de l'entreprise ou vers l'extérieur.

Le développement des systèmes d'information et de logiciels s'appuie sur une démarche, suivant un processus et en se basant sur un ensemble de modèles. UML offre à ce titre une solution en matière de notation des éléments composant un système. Étant simplement un langage de description, il doit être secondé d'un processus qui met en pratique l'ensemble des modèles qu'il propose. A ce titre, le Processus Unifié (PU ou UP en anglais) exécuté autour d'UML permet d'encadrer l'élaboration des modèles d'UML, de maîtriser les risques et d'assurer la réutilisation des composants du système.

2. Concepts préliminaires

2.1. Génie logiciel

Science de génie industriel qui étudie les méthodes de travail et les bonnes pratiques des ingénieurs qui développent des logiciels.

2.2. Modèles / Méthodes de développement

Pour développer un système d'information, nous avons besoin de méthodes, qui s'appuient sur des modèles.

1.1.1. Modèle

Un modèle est une abstraction de la réalité. Selon Torlone, « *a data model is for a database designer what a box of colors is for a painter: it provides a means for drawing representations of reality* » [Livre : Multidimensional Databases, problems and solutions, Idea group, 2003].

Il s'agit de la représentation des éléments d'une réalité pour n'en retenir que les éléments essentiels pour un besoin donné. Il permet aussi de simplifier une réalité et en réduire la complexité. Un modèle est composé d'un ensemble de concepts de modélisation et de relations entre ces concepts ainsi qu'un ensemble de règles de modélisation. La modélisation est le passage de la réalité au modèle en utilisant un langage de modélisation.

Exemple : le modèle conceptuel de données, le modèle conceptuel de traitements (Merise); le modèle relationnel (niveau logique)

1.1.2. *Méthode*

Guide plus ou moins formalisé, constituée d'un ensemble de concepts de modélisations qui sont mis en œuvre en suivant une succession chronologique d'activité et en respectant un ensemble de règles.

Dans le domaine des SI, et selon Rumbaugh, une méthode est *un ensemble de règles et de directives et se compose des éléments suivants*

- un ensemble de **concepts fondamentaux de modélisation** pour capturer la connaissance sémantique d'un problème et de sa solution. Des exemples de concepts sont l'entité, le processus, l'acteur...
- un ensemble de **vues et de notations** pour présenter la modélisation sous-jacente aux personnes qui les examinent et les modifient. Des exemples de modèles sont le modèle *entité/association*, le *modèle relationnel* (niveau logique) le *réseau de Petri*, ...
- **un processus interactif** pas-à-pas employé pour la construction des modèles et pour leur implantation;
- une collection de **suggestions et de règles** qui conduisent à l'exécution du développement. Par exemple, séparer la modélisation des données de celles des traitements, mener certaines étapes en parallèle...

1.1.3. *Modèle de développement de logiciels*

Un modèle de développement de logiciel spécifie l'ensemble des étapes de développement indépendamment de tout modèle (donc de toute méthode). Il décrit la chronologie des étapes et leur organisation dans le temps.

Il existe un plusieurs modèles de développement de logiciels, dont les plus connus sont :

- le modèle en Cascade
- le modèle en V
- le modèle en Spirale.

2.3. **Chronologie des méthodes de développement**

Il existe une multitude de méthodes de développement de SI. On peut classer ces méthodes chronologiquement comme suit

- *Années 70 :* méthodes cartésiennes. Elles s'appuient sur un découpage fonctionnel et hiérarchique du système à analyser. Elles mettent l'accent sur l'aspect dynamique d'un système tout en modélisant les données. Parmi les méthodes de cette catégorie on retrouve la méthode **SADT** (Structured Analysis and Design Technique).

- *Années 80 :* méthodes systémiques. Elles s'appuient sur la théorie des systèmes, en essayant d'aborder la réalité en tant que système complexe à modéliser. Les méthodes de cette catégorie sont orientée côté statique du système tout en modélisation les traitements mais de manière séparée. Exemple : **Merise**.

- *Années 90.* Elles sont caractérisées par l'apparition du paradigme objet en termes de méthodes d'analyse, puisqu'il existait depuis les années 60 en termes de programmation.

- Méthodes récentes : méthodes itératives ou semi-itératives (agiles). Ces méthodes mettent l'accent sur l'adaptation au changement et la délivrance rapide de produits logiciels. Des exemple de telles méthodes sont Scrum et XP (eXtreme Programming).

2.4. **La démarche Objet**

Les concepts de l'orienté objet ont débuté dans la programmation. Par la suite, durant les années 90, l'intérêt a été porté à ces concepts en matière de développement de systèmes d'information pour l'entreprise. Un des concepts-clé de l'introduction de l'objet dans le développement des systèmes étant la notion de réutilisation. En effet, la démarche objet permet de fabriquer des composants réutilisables dans d'autres contextes similaires, ce qui rentabilise l'effort du développement des SI.

Durant les années 90, on comptait déjà environ 50 méthodes différentes, mais les plus distinguées sont les méthodes OMT (Rumbaugh), BOOCH, et OOSE (Jacobson). La fusion des trois méthodes ayant donné lieu en 1996 à UML, qui est soumise à l'OMG, et depuis, UML est devenu une notation reconnue dans le développement des systèmes.

Le développement des systèmes suit un cycle de vie qui définit les étapes, leur enchaînement, et leur interdépendance. La diversité des méthodes a conduit à la diversité des cycles de vie des systèmes orientés objet. Cependant, il a été ressorti trois phases principales

- *L'analyse Orientée Objet* : il s'agit de l'abstraction des objets du monde réel en utilisant un modèle conceptuel ou spécification d'une application.

- *La conception Orientée Objet* : il s'agit d'organiser la solution informatique (la concevoir) avant de passer à l'implantation, en s'appuyant sur les spécifications de la phase d'analyse.

- *L'implantation (ou implémentation) Orientée Objet* : il s'agit ici de mettre en œuvre le logiciel en terme de programmes et base de données basés sur le paradigme objet.

Remarque : en général, les méthodes orientées-objet suivent un cycle de développement en spirale.

2.5. **UML**

1.1.4. *Présentation*

UML (Unified modeling language) est le résultat de la fusion, en 1996, de trois méthodes de développement orientées-objet (OMT, OOD et OOSE) (voir figure 1). UML devient une spécification de l'object management group en 1997 [5] (https://www.omg.org/spec/UML/). Actuellement, UML est en version 2.5.1. UML n'est pas une méthode au sens propre de méthode mais plutôt un ensemble de notations graphiques (modèles) qui peuvent être utilisées par une méthode.

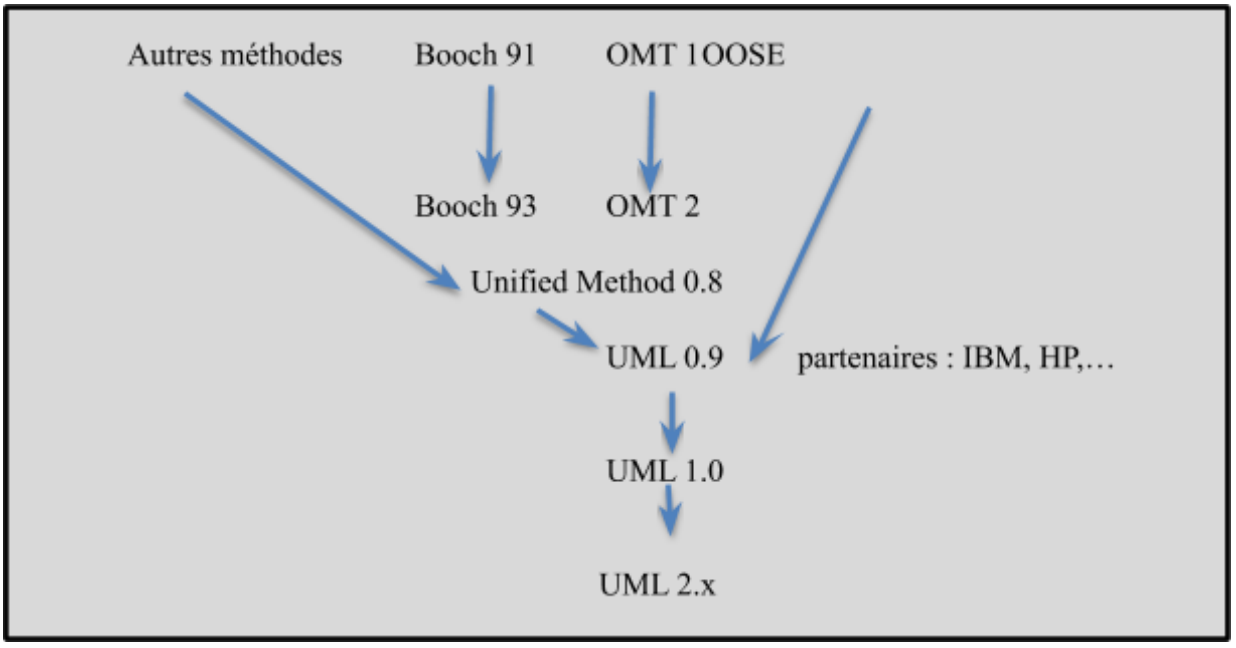

Figure 1. Evolution de UML

UML est un langage unifié qui offre un ensemble de diagrammes représentant différentes vues du système à développer. UML n'a pas été accompagné d'une méthode de développement pour respecter les spécificités des entreprises. Mais, une bonne pratique de développement nécessite d'intégrer l'élaboration des modèles dans un processus afin de:

- **guider** le développement : savoir d'où on vient et où on va, ce qu'on a fait, ce qui reste à faire
- **maîtriser les coûts** du développement
- **maîtriser les délais** associés à chaque étape
- **gérer les risques** liés au projet (ex : rejet par les utilisateurs du système)
- assurer l'**évolutivité** du système en tenant compte de nouveaux besoins ou des changements qui se produisent dans l'environnement de l'entreprise.

Les efforts déployés pour fournir un processus accompagnant UML se sont soldés par la proposition du processus unifié.

1.1.5. *Processus Unifié (Unified Process) :*

Processus de développement de logiciels construit autour d'UML. Les activités de développement autour de ce processus sont énumérées comme suit :

(1) Capture des besoins des utilisateurs,
(2) Analyse des besoins
(3) Conception des solutions
(4) Implémentation des solutions par des outils informatiques
(5) Mise en œuvre (codage, tests, etc.).

1.1.6. *Caractéristique de UP*

Le processus unifié se caractérise par ce qui suit

- **Orienté utilisateur** : il répond aux besoins métiers à travers les cas d'utilisation (use cases). Le cas contraire est de développer un système générique qui ne tient pas compte ni des spécificités de l'entreprise ni des besoins des utilisateurs.
- **Centré sur l'architecture**: c'est-à-dire il prend en compte l'architecture de l'entreprise (deux-tiers, trois-tiers, multi-tiers).
- **Incrémental** : il permet d'élaborer des résultats intermédiaires qui peuvent être évalués par rapports aux spécifications et donc de maîtriser les risques, les coûts et les délais. Le cas contraire est de développer un système comme un seul bloc indivisible.
- **Piloté par les risques** : il permet de prendre en compte les besoins des utilisateurs et les spécificités et contraintes de mise en œuvre.
- **Orienté modèle** : à travers les étapes du processus, les diagrammes UML sont utilisés (classes, cas d'utilisation, ...) afin de capturer les besoins ou de concevoir le système.
- **Permet la réutilisation** car centré sur le développement de composants réutilisables.

1.1.7. *Le processus 2TUP*

Le processus unifié est une spécification générale qui a été implémentée par plusieurs processus concrets (ex : 2TUP, RUP, Agile UP…). Le processus 2TUP (2-Track Unified Process) est un

processus respectant le modèle en Y du développement et qui sépare les aspects techniques des aspects fonctionnels.

1.1.8. *Schéma général du processus 2TUP*

Le schéma général du processus 2TUP est le suivant (figure 2)

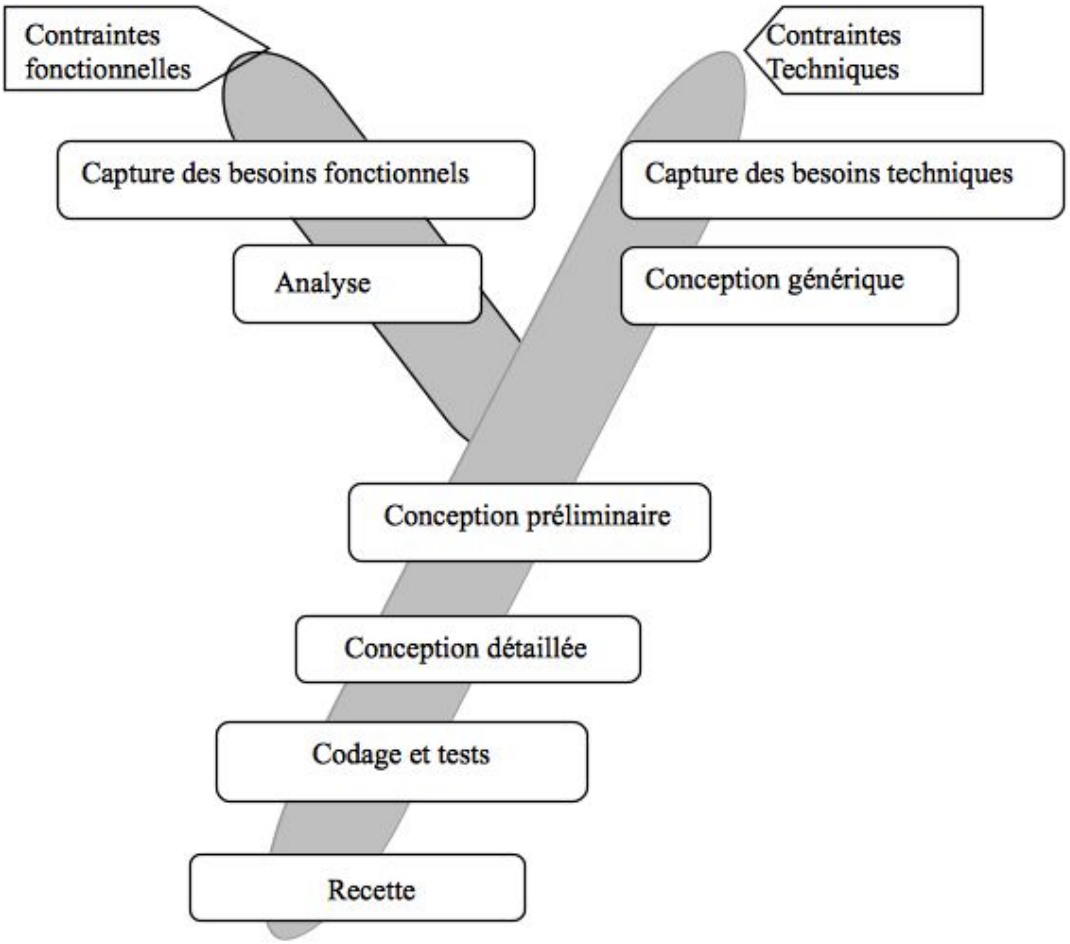

Figure 2. Processus 2TUP

Remarque : les étapes du processus 2TUP sont précédées par une étape de prise de contact avec le domaine et l'environnement de l'entreprise, appelée étude préliminaire. L'étude préliminaire permet de connaître les contraintes fonctionnelles et les contraintes opérationnelles.

- **Contraintes fonctionnelles** : limites et conditions à respecter qui sont liées au métier des utilisateurs.

- **Contraintes techniques** : limites et conditions à respecter qui ne sont liées au métier des utilisateurs mais sont d'ordre technique.

Les étapes, à proprement parler, du processus 2TUP sont les suivantes

- **Capture des besoins fonctionnels :** recensement des besoins liés au métier que les utilisateurs veulent qu'ils soient pris en compte dans le système. Ces besoins sont capturés à l'aide des diagrammes de cas d'utilisation.
- **Capture des besoins techniques :** recensement des besoins à caractère technique non liés au métier des utilisateurs. Ces besoins sont capturés à l'aide des diagrammes de composants et de déploiement.
- **Analyse :** étude des besoins des utilisateurs et leur modélisation. On utilise essentiellement les diagrammes de classes et le diagramme de séquences.
- **Conception générique :** proposition de solution technique indépendante du métier
- **Conception préliminaire :** croisement de l'analyse et de l'architecture générique du système
- **Conception détaillée :** approfondissement de la conception (typage de données, détail des algorithmes, etc.)
- **Codage et tests :** création des bases de données, implémentation des programmes, dans des environnements IDE ou autres.
- **Recette :** remise du système aux utilisateurs finaux.

Dans ce cours, nous allons traiter les premières étapes de la capture des besoins jusqu'à la conception détaillée.

3. Conclusion

Dans ce chapitre, nous avons rappelé la notion de système d'information et passé en revue les différentes approches de développement des systèmes d'informations. Nous avons également présenté l'approche orientée objet pour le développement ainsi que le processus unifié. Enfin, nous avons introduit le processus 2TUP qui sera la base de ce cours.