

Crémetz Olivier
Duhem Jérôme
Lefort Céline
Lemrabet Youness

*Étude des méthodologies UML
(Unified Modeling Language)*



Polytech'Lille – Département Génie Informatique et Statistiques
Année 2005-2006

Étude des méthodologies UML (Unified Modeling Language)

UML est par définition un langage, ou encore une notation, standardisé par l'OMG. UML n'est donc pas une méthode contrairement à MERISE (qui est un langage assorti d'une méthode). Pour savoir précisément comment utiliser UML dans un cycle de développement, il faut associer au langage une méthode, ou process. Plusieurs process existent désormais et proposent des démarches pour utiliser UML (RUP, XP, 2TUP...).

La démarche de création des diagrammes UML est dépendante du process suivi. On doit donc dans un premier temps identifier quel process conviendrait le mieux à notre problématique de développement et ainsi l'ordre de réalisation des diagrammes.

Il existe différentes méthodes mais nous allons vous présenter les 3 principales méthodes :

2TUP : Two Track Unified Process

RUP : Rational Unified Process

XP : Extreme Programmig

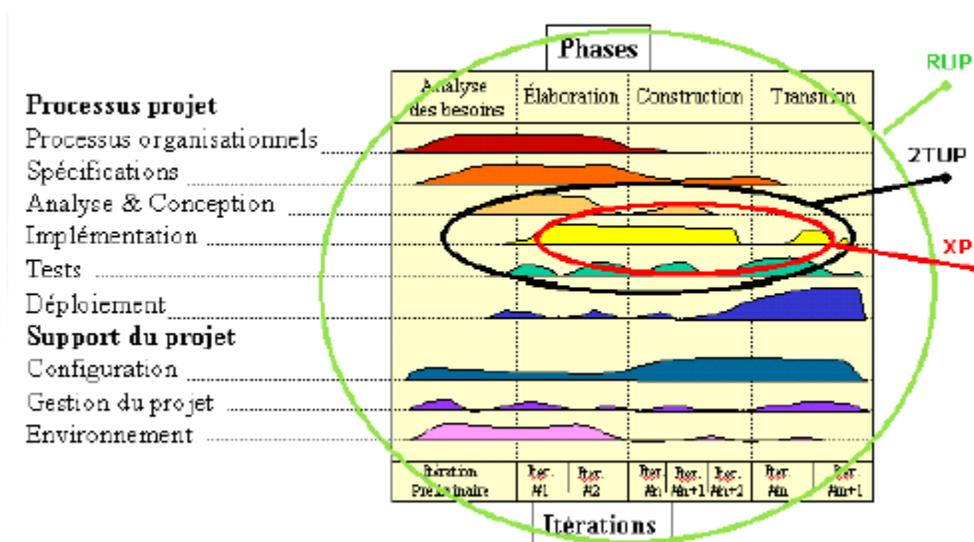


Schéma 1 : Impact des méthodes de développement sur la structure du projet

I) Les principes fondamentaux du Processus Unifié (UP)

Le Processus Unifié (UP) est un processus de développement logiciel « itératif et incrémental, centré sur l'architecture, conduit par les cas d'utilisation et piloté par les risques » :

Il est itératif et incrémental car le projet est découpé en itérations de courte durée (environ 1 mois) . Cela permet de mieux suivre l'avancement global. A la fin de chaque itération, une partie exécutable du système final est produite, de façon incrémentale.

Ce processus est centré sur l'architecture. Tout système complexe doit être décomposé en parties modulaires afin de garantir une maintenance et une évolution facilitées. Cette architecture (fonctionnelle, logique, matérielle, etc.) doit être modélisée en UML et pas seulement documentée en texte.

Il est conduit par les cas d'utilisation : le projet est mené en tenant compte des besoins et des exigences des utilisateurs. Les cas d'utilisation du futur système sont identifiés, décrits avec précision et priorités.

Il est piloté par les risques : les risques majeurs du projet doivent être identifiés au plus tôt mais surtout levés le plus rapidement possible. Les mesures à prendre dans ce cadre déterminent l'ordre des itérations.

La gestion d'un tel processus est organisée suivant les quatre phases suivantes : initialisation, élaboration, construction et transition :

- La phase d'initialisation conduit à définir la « vision » du projet, sa portée, sa faisabilité, son « business case », afin de pouvoir décider au mieux de sa poursuite ou de son arrêt.
- La phase d'élaboration poursuit trois objectifs principaux en parallèle :
 - * Identifier et décrire la majeure partie des besoins utilisateurs,
 - * Construire (et pas seulement décrire dans un document !) l'architecture de base du système,
 - * Lever les risques majeurs du projet.
- La phase de construction consiste surtout à concevoir et implémenter l'ensemble des éléments opérationnels (autres que ceux de l'architecture de base). C'est la phase la plus consommatrice en ressources et en effort.
- Enfin, la phase de transition permet de faire passer l'application des développeurs aux utilisateurs finaux. (comme la formation des utilisateurs, le déploiement, et les tests).

Chaque phase est elle-même décomposée séquentiellement en itérations limitées dans le temps (entre 2 et 4 semaines). Le résultat de chacune d'elles est un système testé, intégré et exécutable.

L'approche itérative est fondée sur la croissance et l'affinement successifs d'un système par le biais d'itérations multiples, feed-back et adaptation cycliques étant les moteurs principaux permettant de converger vers un système satisfaisant. Le système croît avec le temps de façon incrémentale, itération par itération, c'est pourquoi cette méthode porte également le nom de développement itératif et incrémental. Il s'agit là du principe le plus important du Processus Unifié.

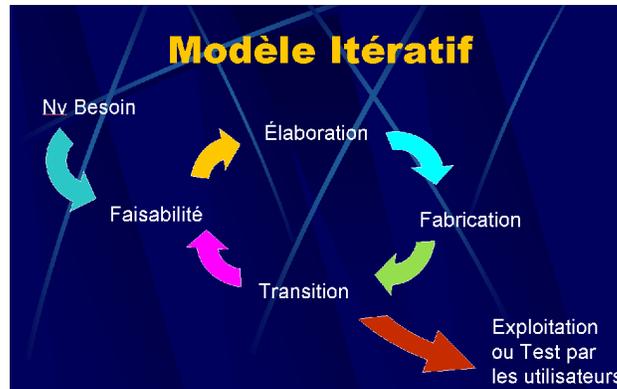


Schéma 2. Schéma du Modèle Itératif

Les activités de développement sont définies par cinq disciplines fondamentales qui décrivent la capture des exigences, l'analyse et la conception, l'implémentation, le test et le déploiement. La modélisation métier est une discipline amont optionnelle et transverse aux projets. Enfin, trois disciplines appelées de support complètent le tableau : gestion de projet, gestion du changement et de la configuration, ainsi que la mise à disposition d'un environnement complet de développement incluant aussi bien des outils informatiques que des documents et des guides méthodologiques.

Contrairement au processus en cascade (souvent appelé cycle en V en France), le Processus Unifié ne considère pas que les disciplines sont purement séquentielles. En fait, une itération comporte une certaine quantité de travail dans la plupart des disciplines. Mais la répartition de l'effort relatif entre celles-ci change avec le temps. Les premières itérations ont tendance à mettre plus l'accent sur les exigences et la conception, les autres moins, à mesure que les besoins et l'architecture se stabilisent grâce au processus de feed-back et d'adaptation.

UP doit donc être compris comme une trame commune des meilleures pratiques de développement, et non comme l'ultime tentative d'élaborer un processus universel.

Les méthodes basées sur UP :

1) La Méthode RUP

Le RUP (Rational Unified Process) est à la fois une méthodologie et un outil prêt à l'emploi (documents types partagés dans un référentiel Web). Cette méthodologie est couramment utilisée dans les projets de plus de 10 personnes.

La méthode RUP est promue par Rational. Ce logiciel donne des documents types, exemples de ce qu'il faut fournir à chaque étape du projet.

Enfin, Cette méthode englobe toutes les phases d'un projet, de l'analyse à la transition, ce qui convient à des projets où "tout est à faire".

a) Ses Points forts :

Ce processus est itératif et cette méthode spécifie le dialogue entre les différents intervenants du projet : les livrables, les plannings, les prototypes... et propose des modèles de documents, et des

canevas pour des projets types

b) Ses Points faibles :

Cette méthode est coûteuse à personnaliser : Il faut une batterie de consultants. Elle est très axé processus, au détriment du développement ce qui laisse peu de place pour le code et la technologie

2) La Méthode 2TUP

La méthode 2TUP (2 track unified process) est un processus de développement logiciel qui implémente le processus unifié UP. Elle s'adapte à des projets de toutes tailles.

Le 2TUP propose un cycle de développement en Y, qui dissocie les aspects techniques des aspects fonctionnels.

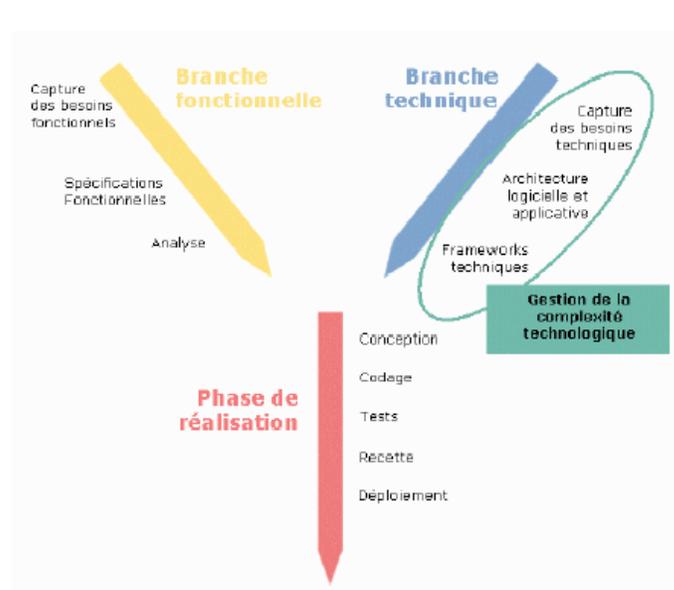


Schéma 2 : structure

en Y du 2TUP

Ce processus en Y s'articule autour de 3 phases :

- une branche technique,
- une branche fonctionnelle,
- et une phase de réalisation.

a) Ses Points forts :

Ce processus est Itératif et fait une large place à la technologie et à la gestion du risque. Il définit les profils des intervenants, les livrables, les plannings, les prototypes.

b) Points faibles :

Cette méthode reste plutôt superficielle sur les phases situées en amont et en aval du développement : capture des besoins, support, maintenance, gestion du changement... Elle ne

propose pas comme RUP des documents types

3) L'eXtreme Programming (XP)

L'Extreme Programming (XP) est une méthode agile de gestion de projet informatique, dont l'objectif est de permettre de gérer des projets de manière simple et efficace. Une méthode agile consiste à impliquer le client au maximum pour favoriser la réactivité.

Elle est un ensemble de pratiques qui couvre une grande partie des activités de la réalisation d'un logiciel - de la programmation proprement dite à la planification du projet, en passant par l'organisation de l'équipe de développement et les échanges avec le client. Ces pratiques n'ont rien de révolutionnaire : il s'agit simplement de pratiques de bon sens mises en œuvre par des développeurs ou des chefs de projet expérimentés, telles que :

- * Avoir un utilisateur à plein-temps dans la salle projet. Ceci permet une communication intensive et permanente entre les clients et les développeurs, aussi bien pour l'expression des besoins que pour la validation des livraisons.

- * Écrire le test unitaire avant le code qu'il doit tester, afin d'être certain que le test sera systématiquement écrit et non pas négligé.

- * Programmer en binôme, afin d'homogénéiser la connaissance du système au sein des développeurs, ainsi que de permettre aux débutants d'apprendre des experts. Le code devient ainsi une propriété collective et non individuelle que tout les développeurs ont le droit de modifier.

- * Intégrer de façon continue, pour ne pas retarder à la fin du projet le risque majeur de l'intégration des modules logiciels écrits par des équipes ou des personnes différentes. La communication avec le client se fait par métaphore ce qui permet de se comprendre entre informaticiens et les personnes extérieures au domaine. Cela facilite la communication.

a) Ses Points forts :

Cette méthode est itérative et simple à mettre en œuvre. Elle fait une large place aux aspects techniques : mise en place de prototypes, de règles de développement, de tests. Cette méthode est d'ailleurs innovante de par sa programmation en binôme.

b) Points faibles :

Cette méthode ne couvre pas les phases en amont et en aval au développement : capture des besoins, support, maintenance, tests d'intégration...

Elle se dispense de la phase d'analyse et reste assez floue sur sa mise en œuvre (Quels sont les intervenants du projet, quels sont les livrables ?)

Pour résumer, on peut dire que XP est une méthodologie légère qui met l'accent sur l'activité de programmation et qui s'appuie sur les valeurs suivantes : communication, simplicité et feedback. Elle est bien adaptée pour des projets de taille moyenne où le contexte (besoins utilisateurs, technologies informatiques) évolue en permanence. Cette méthode fait intervenir le client dans le projet.

II) Choix de la méthode :

Étant donné l'étendue de notre projet et vu sa forte composante technique, on peut d'ores et déjà éliminer la méthode RUP car cette méthode, très axée sur les processus au détriment du développement, laisse peu de place pour le code et la technologie, ce qui n'est pas intéressant vu la forte composante technique du projet. La méthode XP est centrée sur le développement logiciel et est donc quant à elle trop restreinte au niveau des phases en amont et en aval du projet. Enfin, le double aspect fonctionnel et technique du 2TUP est intéressant à prendre en compte car le projet contient une forte composante technique. La parallélisation des tâches s'avère très intéressante sur ce projet.