Université Abdelhamid Mehri – Constantine 2

2020/2021. Semestre 5

Génie Logiciel 2

– Cours 1 –

Chapitre 1 : Processus Unifé (UP)

Staff pédagogique			
Nom	**Grade**	**Faculté/Institut**	**Adresse e-mail**
Dr. Sahar SMAALI	MCB	Nouvelles technologies	sahar.smaali@univ-constantine2.dz

Étudiants concernés			
Faculté/Institut	**Département**	**Niveau**	**spécialité**
Nouvelles technologies	TLSI	Licence 3	GL

Objectifs du cours 1

- Familiariser avec le processus unifié (UP), ses phases, activités et caractéristiques.
-
-

Sommaire

1 Introduction

UML (Unified Modeling language) est un langage de modélisation graphique destiné à comprendre et décrire des besoins, spécifier et documenter des systèmes, esquisser des architectures logicielles, concevoir des solutions et communiquer des points de vue. Il est devenu un standard couramment utilisé en développement logiciel et en conception orientée objet. UML propose 14 types de diagrammes divisés selon 3 axes permettant de modéliser tous les aspects d'un système:

- L'axe fonctionnel utilisé pour décrire ce que fait le système à réaliser (diagramme de cas d'utilisation);
- L'axe structurel et statique relatif à la structure du système (diagrammes de classes, d'objets, de composants et de déploiment);
- L'axe dynamique relatif à la construction des fonctionnalités du système (diagrammes de séquence, de collaboration, d'activités et d'états/transitions).

Cependant, UML n'étant pas une méthode, l'utilisation des diagrammes est laissée à l'appréciation de chacun. Il ne préconise aucune démarche à suivre et ne définit pas le processus d'élaboration des modèles. Ainsi, il ne précise pas l'ordre dans lequel les diagrammes doivent être établis et le moment de la conception auquel doivent intervenir. Seul un processus de développement peut répondre à ces questions.

Remarque

Un processus de développement définit une séquence d'étapes, partiellement ordonnées, qui concourent à l'obtention d'un système logiciel ou à l'évolution d'un système existant. Son but est de produire des logiciels de qualité qui répondent aux besoins de leurs utilisateurs dans des temps et des coûts prévisibles. Plus simplement, un processus doit permettre de répondre à la question fondamentale : « Qui fait quoi et quand ? »

2 Processus Unifié

Le processus unifié noté UP (Unified Process) a été proposé par les créateurs originels de UML. Il s'agit d'un processus de développement logiciel moderne, très complet et construit basé sur UML. Il regroupe les activités à mener pour transformer les besoins d'un utilisateur en système logiciel de bonne qualité. UP peut être adapté à une large classe de systèmes logiciels, à différents domaines d'application, à différents types d'entreprises, à différents niveaux de compétences et à différentes tailles de l'entreprise.
UP définit ainsi quatre éléments primaires de modélisation à savoir les artefacts, les rôles, les activités et leur enchaînement.

- **Travailleurs et Rôles (Qui?):** définissent le comportement et les responsabilités des intervenants dans un projet en terme d'activités et de tâches que l'intervenant doit assurer. Exemples de rôles: architecte, analyste, développeur, testeur et manager.
- **Activités (Comment?):** Une activité est une unité de travail effectuée par un individu dans un rôle bien précis. L'activité a un but clairement établi et produit un résultat exprimé en terme de création ou de mise à jour d'artefacts. Exemples d'activités : Expression des besoins, Analyse, Conception, Implémentation, Test.
- **Artefacts (Quoi?):** Les artefacts sont les produits tangibles du projet : les choses qui sont produites par le projet ou utilisées pour travailler sur le produit final ou pour documenter le projet. Les artefacts représentent l'input et l'output des activités. Exemples d'artefacts: Architecture, Code source, Modèles, Exécutables.
- **Workflow (Quand?):** Il décrit l'enchaînement des activités qui a pour but de produire un résultat de valeur.

3 Phases et activités de UP

Le cycle de vie de la méthode UP se décompose en 4 phases principales au cours desquelles un ensemble d'activités d'ingénierie sont mises en oeuvre. En effet, UP gère le développement selon deux axes:

- **Axe vertical** qui représente les principaux enchaînements d'activités. Cette dimension rend compte de l'aspect statique du processus qui s'exprime en termes de composants, de processus, d'activités, d'enchaînements, d'artefacts et de travailleurs.
- **Axe horizontal** qui représente le temps et montre le déroulement du cycle de vie du processus. Cette dimension rend compte de l'aspect dynamique du processus qui s'exprime en terme de cycles, de phases, d'itérations et de jalons (points de repère).

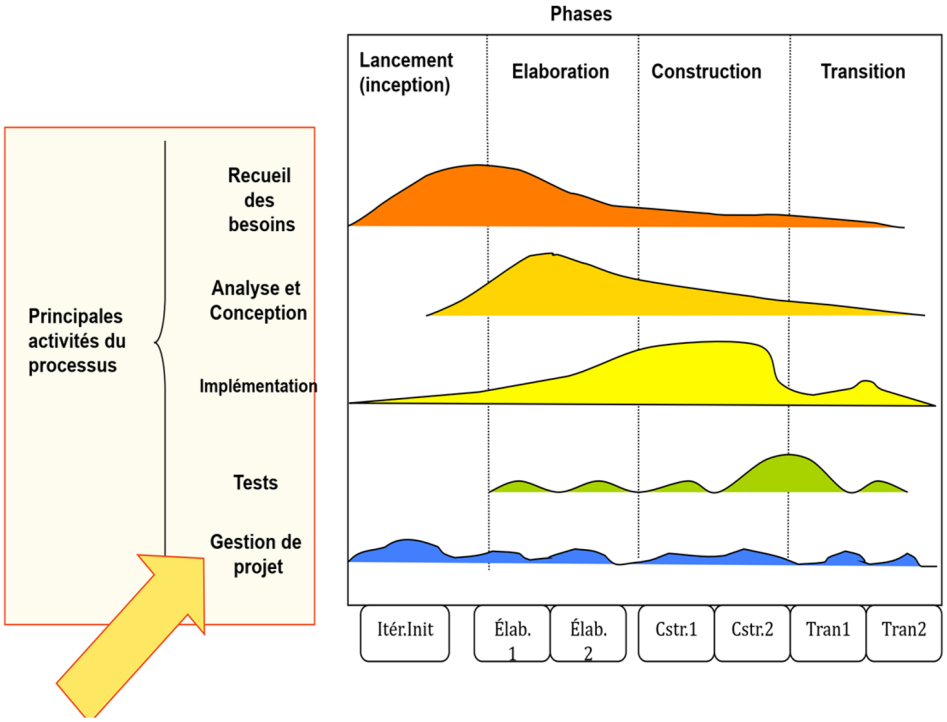

Figure 1: Architecture bidimensionnelle d'UP.

3.1 Activités

3.1.1 Recueil des besoins

Cette activité consiste en étude du domaine concerné pour le système à développer (nouveau système) et une évaluation du système existant (ancien). Ainsi, elle permet de recenser les besoins fonctionnels qui conduisent à l'élaboration des cas d'utilisation et les besoins non fonctionnels (techniques) qui aboutissent à la rédaction d'une matrice des exigences.

3.1.2 Analyse

L'objectif de l'analyse est à comprendre les besoins et les exigences formulés par le client. Il s'agit de réaliser des spécifications permettant de concevoir la solution. L'analyse se concrétise par l'élaboration du modèle d'analyse qui permet de modéliser les aspects statique et dynamique du système.

3.1.3 Conception

La conception prend en compte les choix d'architecture technique liées au langage de programmation, à l'utilisation des composants et au système d'exploitation. La conception étoffe le modèle d'analyse en y intégrant les aspects techniques plus proches des préoccupations physiques.

3.1.4 Implémentation

Cette phase correspond à la production du logiciel sous forme de composants, c'est-à-dire, de code source, de scripts, de binaires, d'exécutables et d'autres éléments du même type.

3.1.5 Test

Les tests permettent de vérifier le bon fonctionnement du logiciel implémente, sa performance et sa qualité. Ils comprennent la vérification des composants et leur intégration ainsi que l'implémentation de touts les besoins (fonctionnelles et techniques). Différents niveaux de tests sont réalisés dans cette activité : test unitaire, test d'intégration, test de réception, test de performance et test de non-régression.

Le tableau ci-dessous résume l'objectif de chaque activité, les intervenants dans sa réalisation et les artefacts qu'elle produit.

Activité	Objectif	Rôles	Artefacts
Recueil des besoins	Capturer les besoins des utilisateurs et clients.	Analystes	• Modèle de cas d'utilisations • Spécifications non fonctionnelles • Listes des Risques • Maquette IHM
Analyse	Raffiner et analyser les besoins	Analystes	• Modèle d'analyse
Conception	Concevoir l'architecture du système	Concepteurs et Architectes	• Modèle de conception • Modèle de déploiement
Implémentation	Raffiner le modèle de conception et produire des composants logiciels	Développeurs et Intégrateurs	• Modèle de composants • Prototype ou version du logiciel
Test	Test des composants logiciels développés	Testeurs	• Plan du test • Script de test

Table 1: Activités du processus UP

3.2 Phases

UP Considère un produit logiciel par rapport à ses versions. Une version du logiciel correspond à un cycle de 4 phases successives. Chaque phase peut-être divisée en itérations, et chaque itération comprend les cinq activités de base. Chaque phase se termine par un jalon qui correspond à une étape d'évaluation de la phase terminée, et de lancement de la phase suivante.

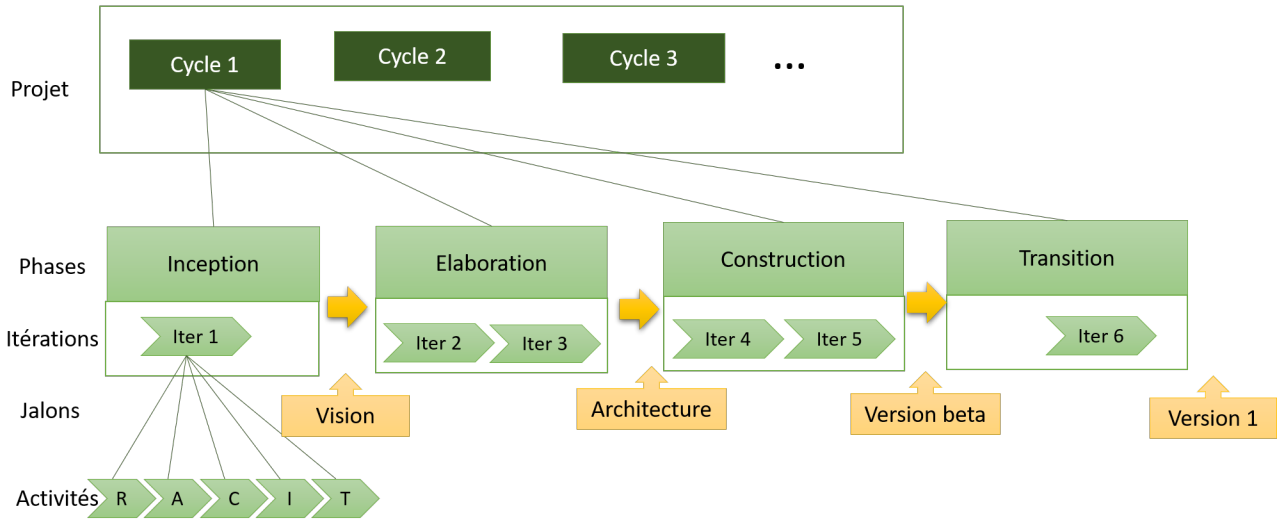

Figure 2: Cycle de vie du processus UP.

3.2.1 Inception (Lancement)

L'objectif de cette phase est d'initialiser le projet en menant:

- une étude d'opportunité et de faisabilité du système à construire.
- une évaluation des risques.
- un recensement les besoins fonctionnels (du point de vue de l'utilisateur) et les besoins non fonctionnels (techniques).
- une estimation du budget, de la durée et des ressources nécessaires pour le projet.

Les activités du recueil des besoins et d'analyse représentent l'essentiel de la phase de l'inception. Cependant, une certaine conception et implémentation peuvent également être effectuées s'il est décidé de construire un prototype technique ou de preuve de concept. Le test n'est généralement pas abordé à cette phase. L'inception produit un document qui donne une vision sur les principales exigences, caractéristiques et contraintes du projet (Cahier de charges). Ce n'est qu'à l'issue de cette première phase que l'on peut considérer que le projet est véritablement lancé.

3.2.2 Élaboration

L'élaboration reprend les éléments de la phase d'analyse des besoins et les précise pour arriver à une spécification détaillée de la solution à mettre en œuvre. Cette phase a pour but de :

- élargir l'appréciation de la faisabilité sur la quasi-totalité des cas d'utilisation.
- analyser le domaine technique du système à développer.
- concevoir une architecture de référence.
- définir les niveaux de qualité à atteindre.
- faire une planification complète abordant les questions de calendrier, de personnel et de budget.

L'élaboration se concentre principalement sur le recueil des besoins, l'analyse. La conception et l'implémentation sont plus à la fin de la phase où une architecture exécutable est produite.

3.2.3 Construction

Cette phase est fortement centrée sur les activités de conception, d'implémentation et de test. Elle a pour but :

- la production d'une version bêta du logiciel.
- le développement complet de composantes et le test par rapport au critères d'évaluation définis.

- l'appréciation des produits par rapport aux visions définies.

3.2.4 Transition

La phase de transition commence après le test de la version bêta du produit (mené dans la phase précédente). Il s'agit de:

- livrer le produit pour une exploitation réelle.
- valider le nouveau système auprès des utilisateurs.
- préparer le déploiement du logiciel sur tous les sites utilisateurs.
- créer des manuels d'utilisation et d'autres documents.
- mettre en œuvre d'un service d'assistance ;

Dans cette phase, l'accent est mis sur l'implémentation et le test. Une conception suffisante est effectuée pour corriger les erreurs de conception trouvées dans les tests bêta. Espérons qu'à ce stade du cycle de vie du projet, très peu de travail devrait être effectué dans les premières activités.

4 Caractéristiques de UP

Les caractéristiques essentielles du processus unifié sont les suivants:

- Processus associé à UML,
- Processus guidé par les cas d'utilisation,
- Processus itératif et incrémental,
- Processus centré sur l'architecture,
- Processus orienté par la réduction des risques.

4.1 Processus guidé par les cas d'utilisation

L'objectif du processus est de construire un système qui réponde aux besoins des futurs utilisateurs. Il est donc centré sur les cas d'utilisation qui permettent de spécifier les besoins fonctionnels (du point de vue de l'utilisateur). Le modèle des cas d'utilisation est utilisé tout au long du cycle de développement permettant aux analystes, concepteurs et développeurs de créer une série de modèles d'analyse, de conception et d'implémentation réalisant les cas d'utilisation (voir la figure 3). En effet, le modèle des cas d'utilisation:

- représente un point de départ pour l'analyse (découverte des objets, leurs relations, et leur comportement)
- permet de garantir que la conception évolutive est toujours pertinente par rapport aux besoins de l'utilisateur.
- permet de Guider la construction des interfaces front-end.
- permet de Guider la mise au point des plans de tests.

4.2 Processus centré sur l'architecture

L'architecture d'un système logiciel peut être décrite comme les différentes vues du système qui doit être construit. Elle comprend les aspects fonctionnel, structurel, comportemental ainsi que technique du système. Il est important de définir le plus tôt possible l'architecture type qui sera retenue pour le développement, l'implémentation et ensuite le déploiement du système. Une architecture d'un système doit prévoir la réalisation de tous les cas d'utilisation et elle est souvent influencée par :

- Les langages et frameworks envisagés pour le développement;
- La plate-forme sur laquelle devra s'exécuter le système;
- Les composants et les bibliothèques réutilisables disponibles pour le développement;

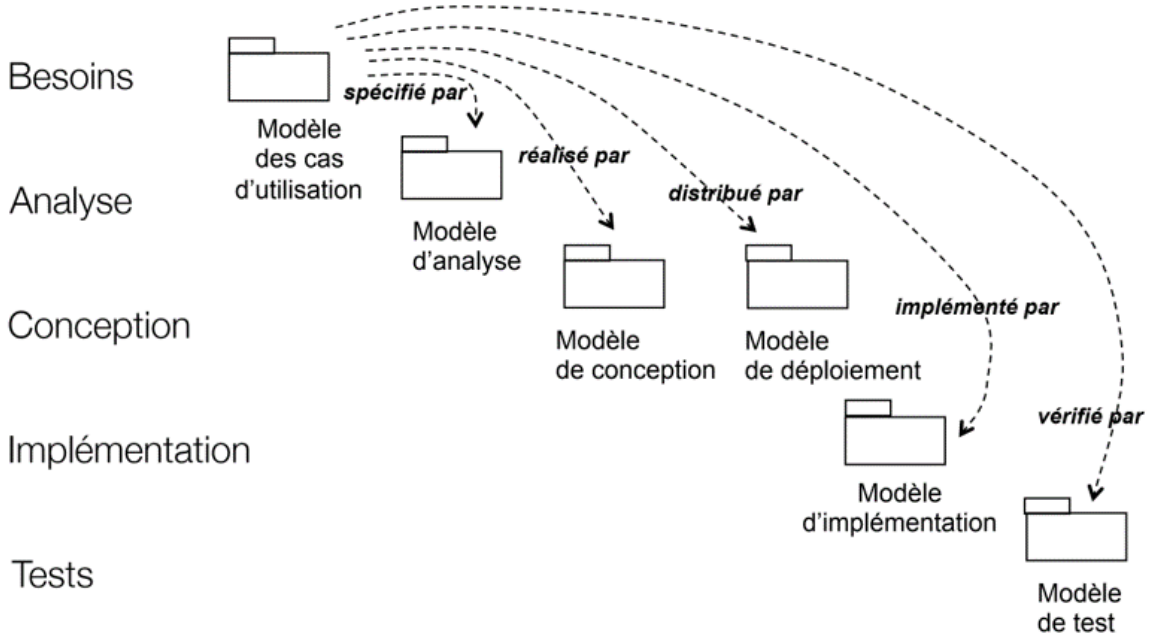

Figure 3: UP piloté par les cas d'utilisations.

- Les considérations de déploiement, les systèmes existants et les besoins non fonctionnels (performance, fiabilité, sécurité, etc.).

4.3 Processus itératif et incrémental

UP préconise l'utilisation du principe de développement par itérations successives, car il n'essaie pas de tout développer d'un coup mais plutôt de découper un projet en des itérations en s'appuyant sur la décomposition du système en cas d'utilisation. Une itération prend en compte un certain nombre de cas d'utilisation, en considérant en priorité ceux exposant des risques majeurs; et produit un incrément (prototype) implémentant ces cas en enrichissant les incréments des itérations précédente (Figures 2 et 4). Les avantages du développement itératif se caractérisent comme suit :

- Gestion de la complexité
- Accélération du rythme de développement grâce à des objectifs clairs et à court terme.
- Évaluation et traitement au fur et à mesure des itérations
  - Diminution de l'échec
  - Architecture mise à l'épreuve rapidement (prototype réel)
- Intégration continue et progrès immédiatement visibles
- Prise en compte des modifications de besoins feedback, implication des utilisateurs et adaptation précoce

4.4 Processus orienté par la réduction des risques

Un projet de développement peut toujours subir des risques de différentes natures (fonctionnels, techniques, financiers, ou autres) tels que une architecture mal conçue, performances insuffisantes, problèmes commerciaux et financiers ou compétences des développeurs.
L'analyse des risques doit être présente à tous les phases et activités du processus de développement. Il est important de bien évaluer et s'attaquer en priorité aux risques les plus cruciaux qui menacent le plus la réussite du projet. Du fait de l'application du processus itératif permet de diminuer les risques au fur et à mesure du déroulement des itérations successives (voir figure 4).

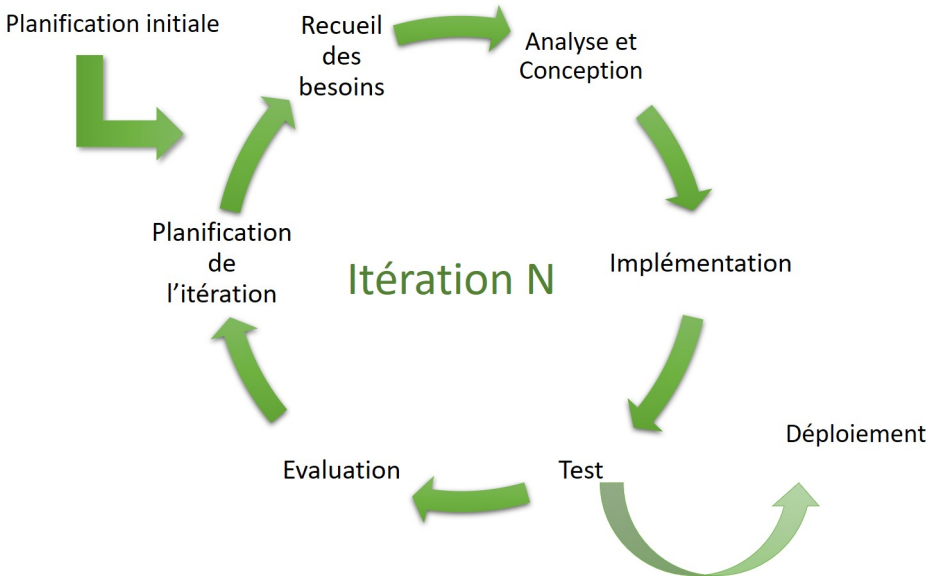

Figure 4: Itérations et gestion des risques

5 Conclusion

Le processus unifié regroupe de bonnes pratiques de l'ingénierie logicielle pour développer et maintenir des logiciels de qualité. Il s'agit d'une approche générique et hautement adaptable aux besoins du projet en cours (taille, personnels, entreprise, compréhension du processus, etc.). Ces principes fondamentaux sont valables pour toute conception orientée objets. Les adaptations d'UP les plus connues sont : RUP (Rational Unified Process), XP (eXtreme Programming), 2TUP (Two Tracks Unified Process).

Références

- GABAY, Joseph et GABAY, David. UML 2 Analyse et conception: Mise en œuvre guidée avec études de cas. Dunod, 2008.
- ARLOW, Jim et NEUSTADT, Ila. UML 2 and the unified process: practical object-oriented analysis and design. Pearson Education, 2005.
- Pascal Roques. Les cahiers du programmeur, UML2 Modéliser une application web. 4eme édition. EYROLLES, 2008.
- UP : Unified Process, ÉDITIONS EYROLLES https://sabricole.developpez.com/uml/tutoriel/unifiedProcess consulté le 29-11-2020.