Introduction aux Méthodes de Développement
GL

3ième année S.I.

Dr. Kouah S.

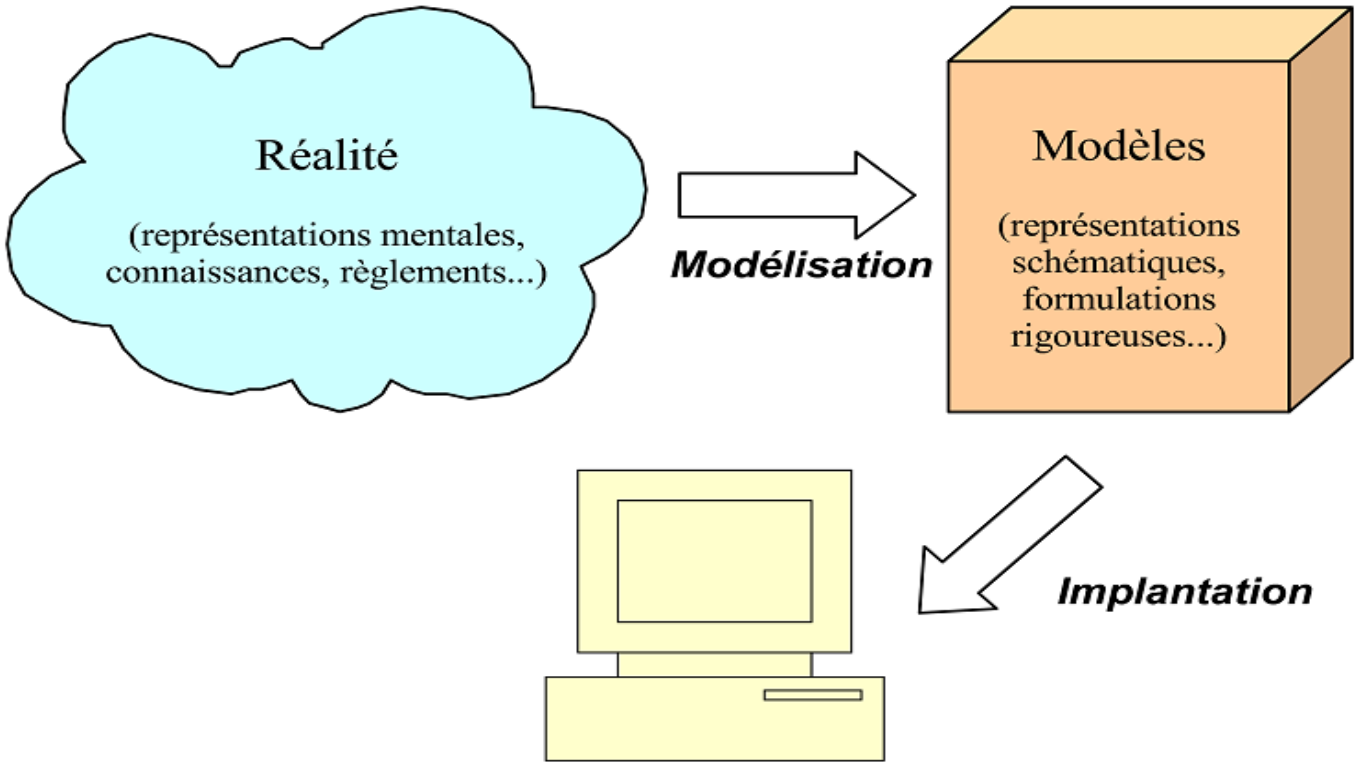
2019- 2020

Plan du chapitre

- **Introduction**
- **Rappel:**
  - **Importance de la modélisation**
  - **Méthodes de développement**
- **Unified Process (UP)**
- **Rational Unified Process(RUP)**

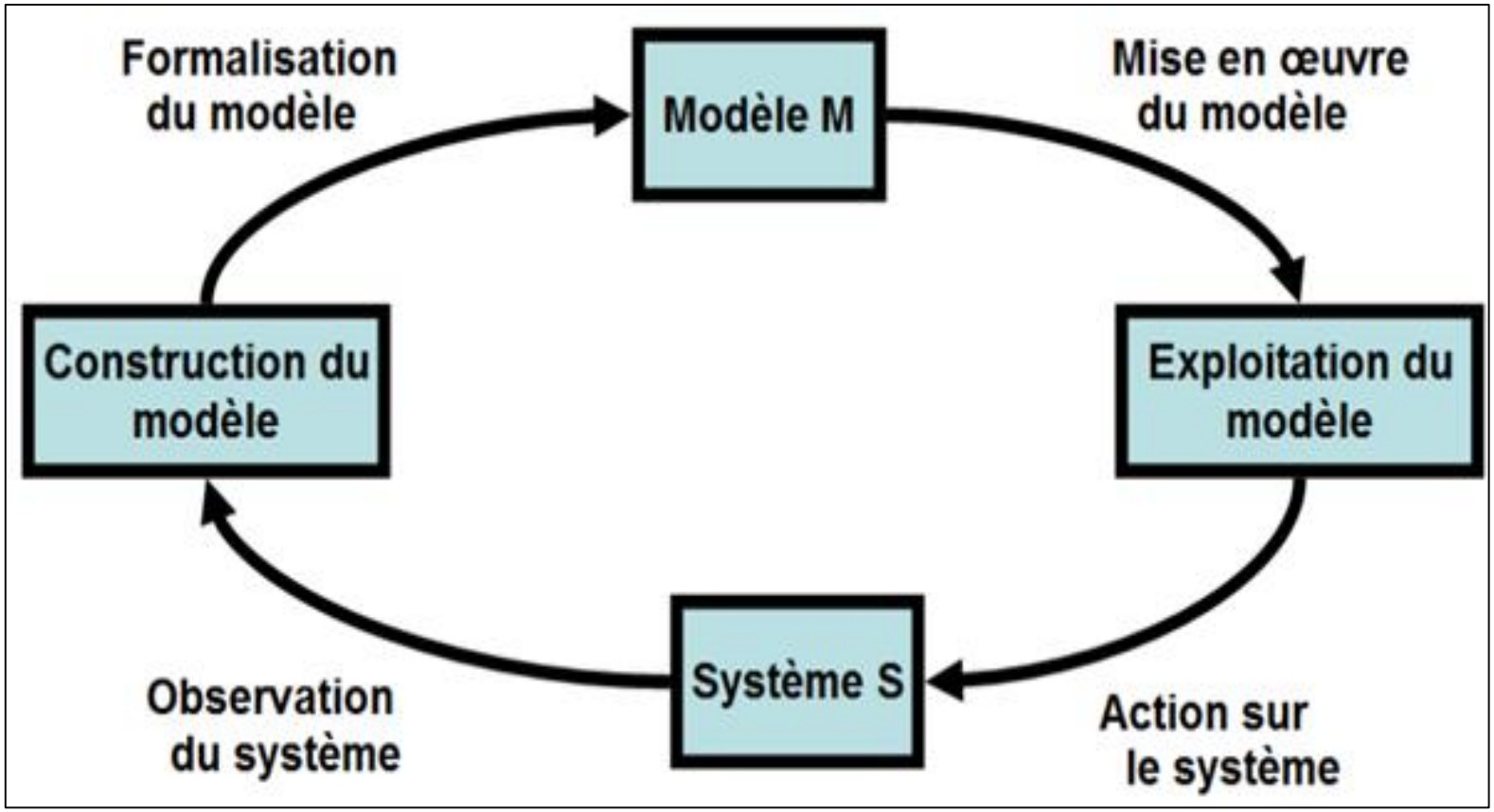
Rappel: Importance de la modélisation

- La modélisation d'un système informatique est le processus par lequel on **construit un modèle** pour ce système.



Rappel: Importance de la modélisation

- Pratiquement, la modélisation passe par les étapes suivantes:

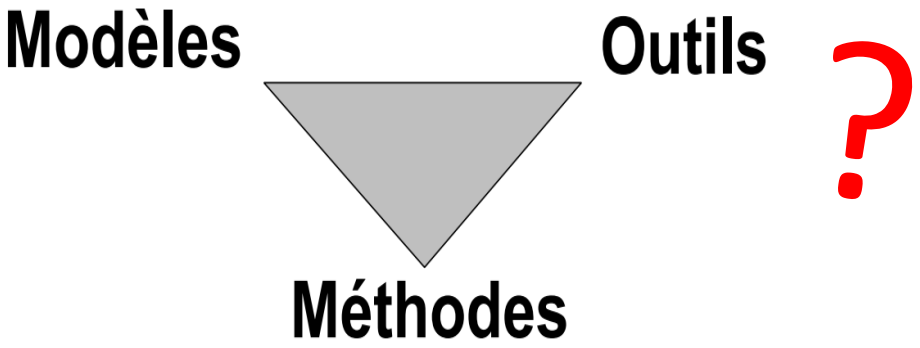


Rappel: Importance de la modélisation

Pourquoi modéliser?

- Comprendre le système à informatiser.
- *« Un modèle est une simplification de la réalité. »* ***Grady BOOCH.***
- Communiquer avec les membres de l'équipe.
- Maîtriser la complexité
- Automatiser la production de logiciel.
  - Documentation.
  - Code.

Modéliser = comprendre et représenter



- **Modèle**: représentation abstraite de tout ou partie du réel. Un modèle ne représente pas une **réalité absolue** mais reflète des aspects importants de la réalité, il en donne donc une vue juste et pertinente. (**vue subjective)**
- **Outil**: support automatisé ou semi automatisé pour supporter les méthodes.
- **Méthode**: manière dont on va conduire une activité.

Méthode = {modèle , outils} + démarche de mise en œuvre.

Une méthode sert à canaliser et ordonner les étapes de la modélisation

Classification des méthodes de développement

- Plusieurs classifications , selon certains critères :

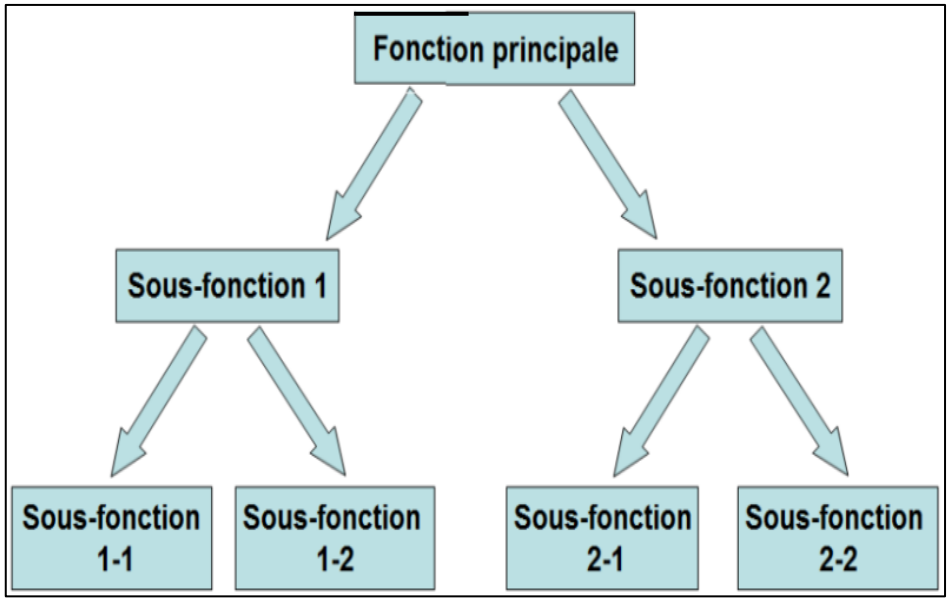
1) Décomposition Descendante/Ascendante

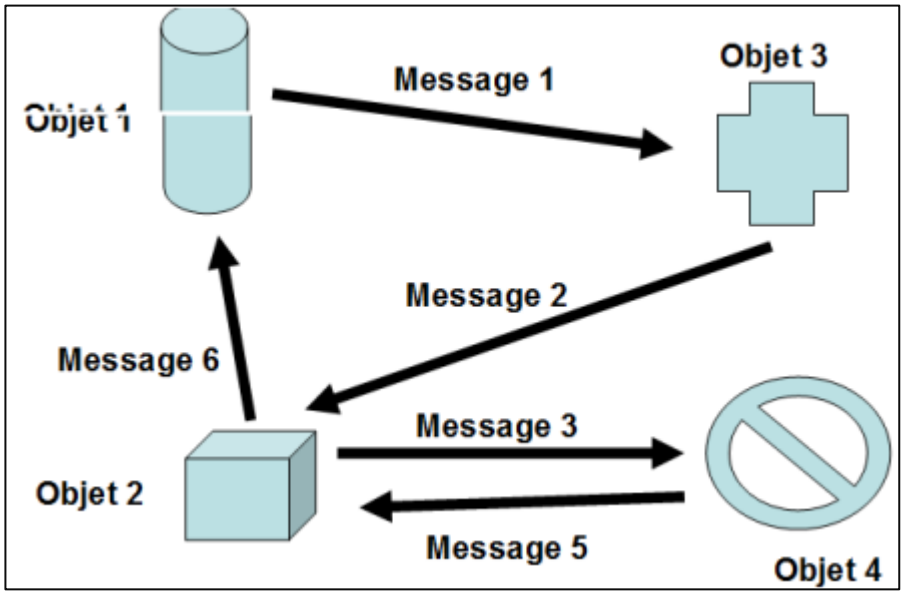
- **Méthodes descendantes (Top Down).**
- **Méthodes ascendantes (Bottom-up).**

2) Décomposition Fonctionnelle/Objet

Les méthodes fonctionnelles **Les méthodes orientée Objet**

SADT,SA, SD. **OOA, OOD, HOOD, OMT, OOSE**





Modélisation orientée objets

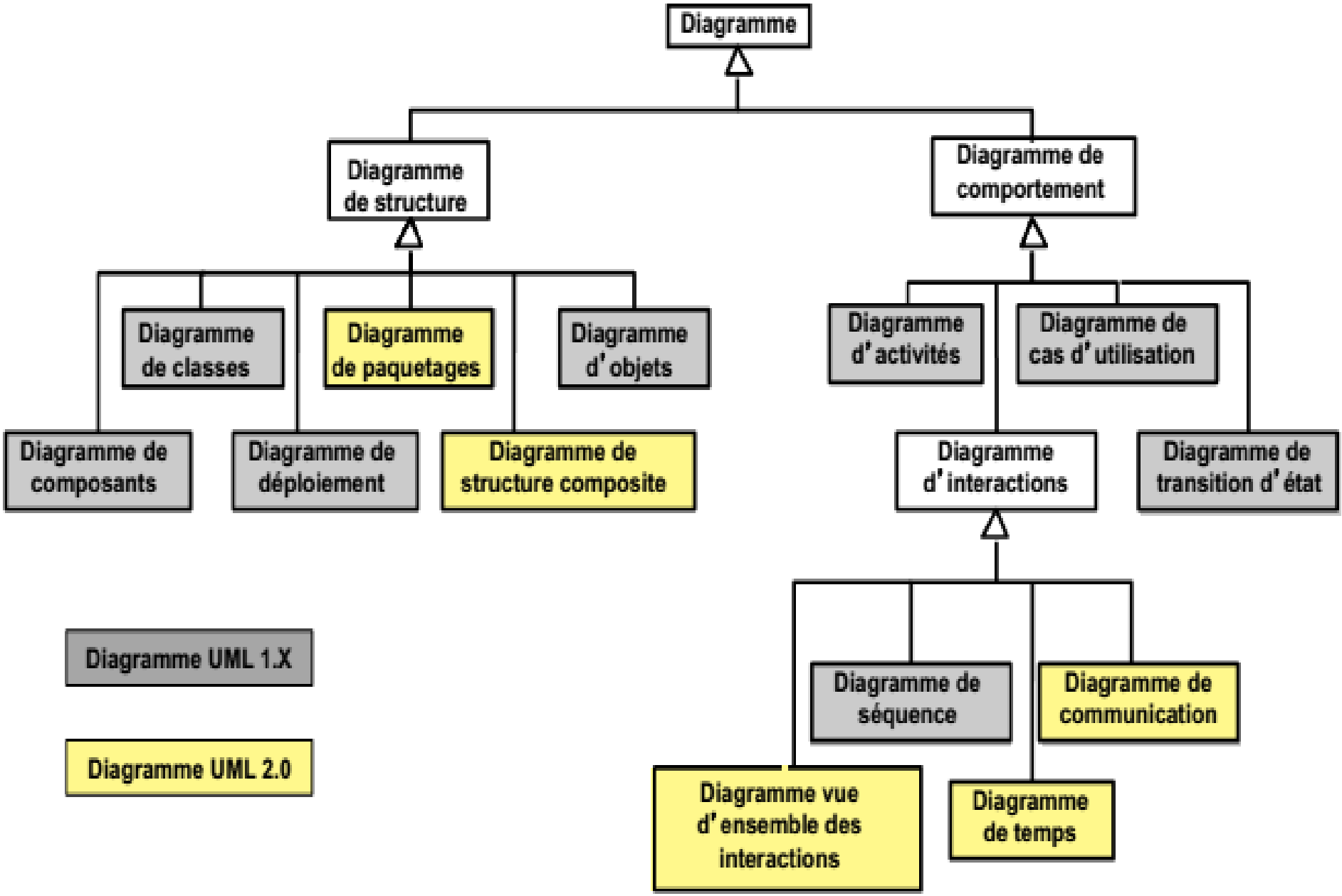
- La conception orientée objets est la méthode qui conduit à des architectures logicielles fondées sur les objets du système, plutôt que sur une décomposition fonctionnelle.
- C'est la structure du système qui lui donne sa forme.
- On peut partir des objets du domaine (briques de base) et remonter vers le système global: approche ascendante.
- Attention: l'approche objet n'est pas seulement ascendante.

Typologie des méthodes d'analyse et de conception (2)

Une autre classification distingue:

- Les méthodes dirigées par les données.
- Les méthodes fonctionnelles (dirigées par les traitements)
- Les méthodes systémiques
- Les méthodes orientées objets.

Les diagrammes UML



Le Processus Unifié ou UP (Unified Process) 1

- Le Processus Unifié est une méthode générique de développement de logiciel développée par les concepteurs d'UML.
- **Générique** signifie qu'il est **nécessaire d'adapter UP au** contexte du projet, de l'équipe, du domaine et/ou de l'organisation.
- Il existe donc un certain nombre de méthodes issues de UP comme par exemple: **RUP (Rational Unified Process), 2TUP (Two Track Unified Process)**.
- Il existe d'autres approches (qui ne sont en général pas complètement contradictoires), comme par ex. les méthodes « agile », beaucoup **moins orientées modèle**, tel que **XP (eXtreme Programming).**

Le Processus Unifié ou UP (Unified Process) 2

- Le processus unifié permet **d'affecter des tâches et des responsabilités** au sein d'une organisation de développement logiciel
- Approche disciplinée pour **des gros projets** (chef de projet, analystes, intégrateur, intervenants, etc.)
- Approche **à adapter** pour des petits projets .
- Pas particulièrement conçu pour le développement de systèmes embarqués

Le Processus Unifié ou UP (Unified Process) 3

- Le processus unifié utilise **le langage UML.**
- Le processus unifié est **piloté par les cas d'utilisation**
- Centré sur **l'architecture**
- **Itératif** et **incrémental**
- Orienté **risques**

UP est piloté par les cas d'utilisation :

- Système analysé, conçu et développé pour des utilisateurs.
- Tout doit donc être fait en adoptant le point de vue utilisateur.

UP est centré sur l'architecture :

- L'architecture du système est décrite à l'aide de **différentes vues**.
- L'architecte procède de manière incrémentale :
  - il commence par définir une **architecture simplifiée** qui répond aux besoins classés comme prioritaires;
  - Puis définit à partir de l'architecture simplifiée les sous-systèmes de manière beaucoup plus précise.

UP est itératif et incrémental :

- En procédant de manière itérative, il est possible de **découvrir les erreurs et les incompréhensions plus tôt.**
- Le **feedback de l'utilisateur** est aussi encouragé et les tests effectués à chaque utilisation permettent d'avoir une vision plus objective de l'avancement du projet.
- Le **travail itératif** permet à l'équipe de capitaliser à chaque cycle les enseignements du cycle précédent.

UP est orienté risques :

- Identifier les risques.
- Maintenir une liste de risques tout au long du projet.

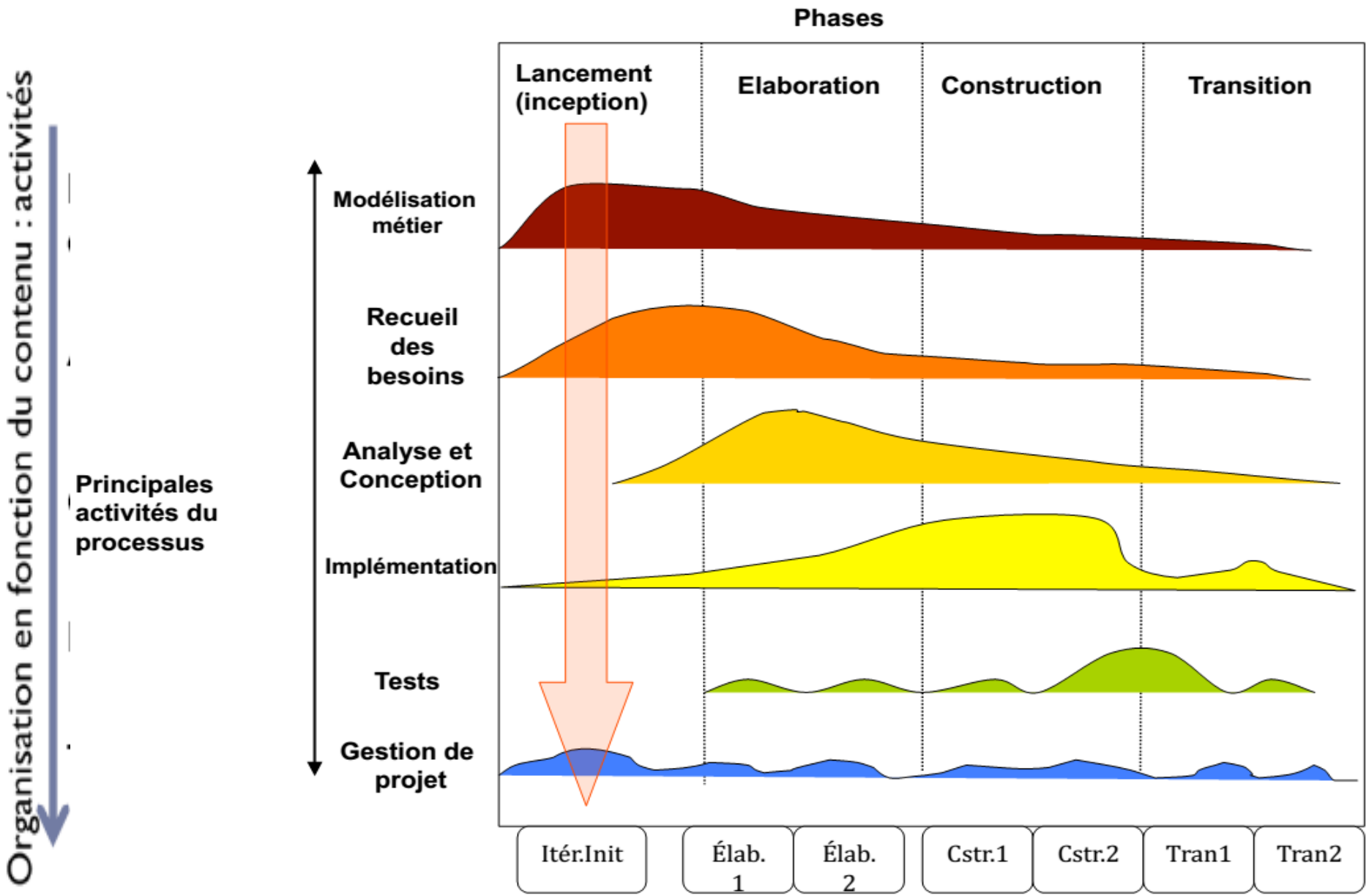
Exemple d'implémentation de UP: RUP (Rational Unified Process)

- **RUP** est l'une des plus célèbres implémentations de la méthode **PU** permettant de donner un cadre au développement logiciel.

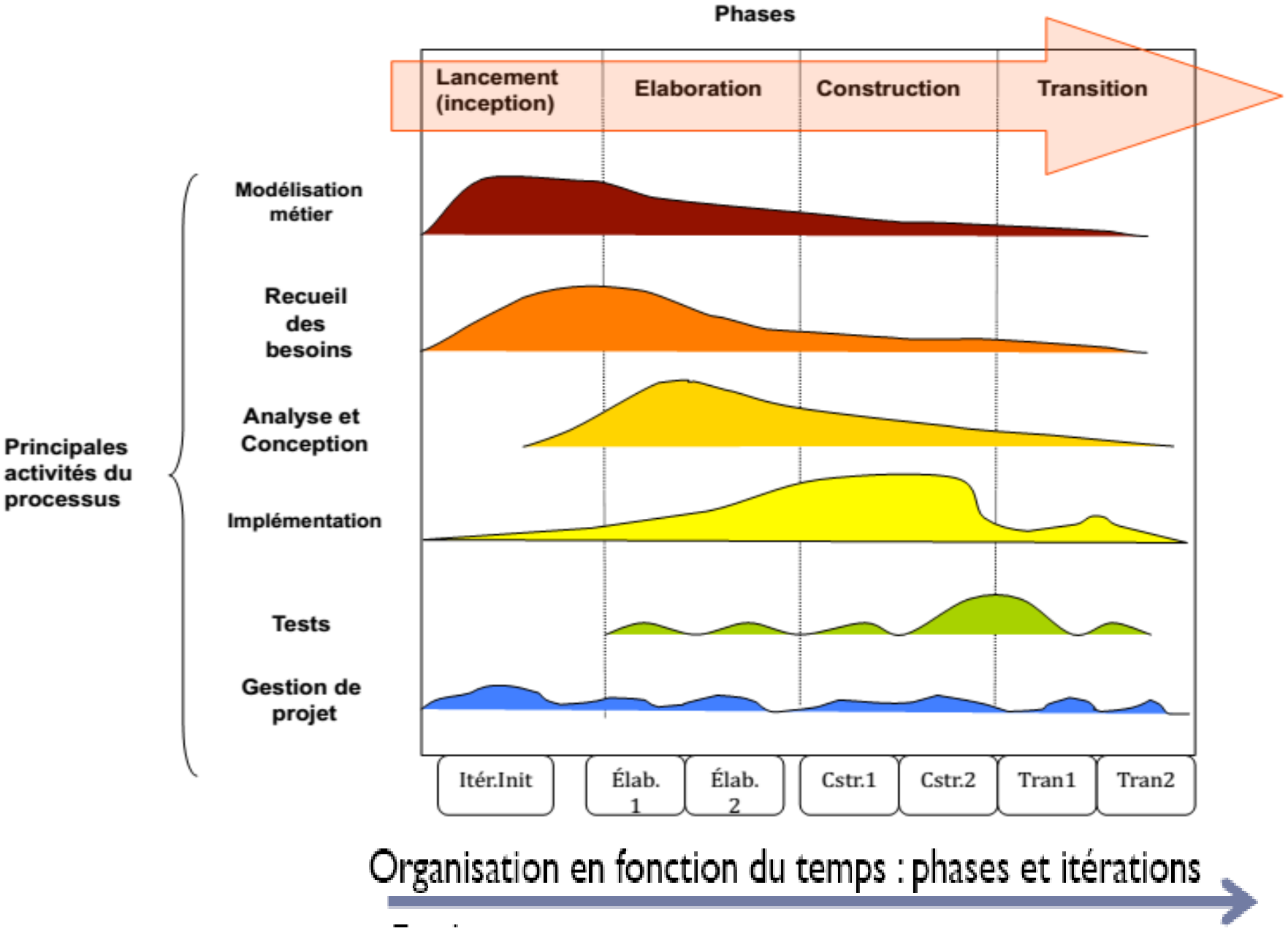
Composantes de RUP

- **4 phases :**
  - Etude d'opportunité.
  - Elaboration.
  - Construction.
  - Transition.
- **5 activités:**
  - Expression des besoins.
  - Analyse.
  - Conception.
  - Implémentation.
  - Test.
- **Ensemble d'itérations :** circuit de développement aboutissant à un délivrable.

RUP: Structure logique du processus



RUP: les quatre phases (1)



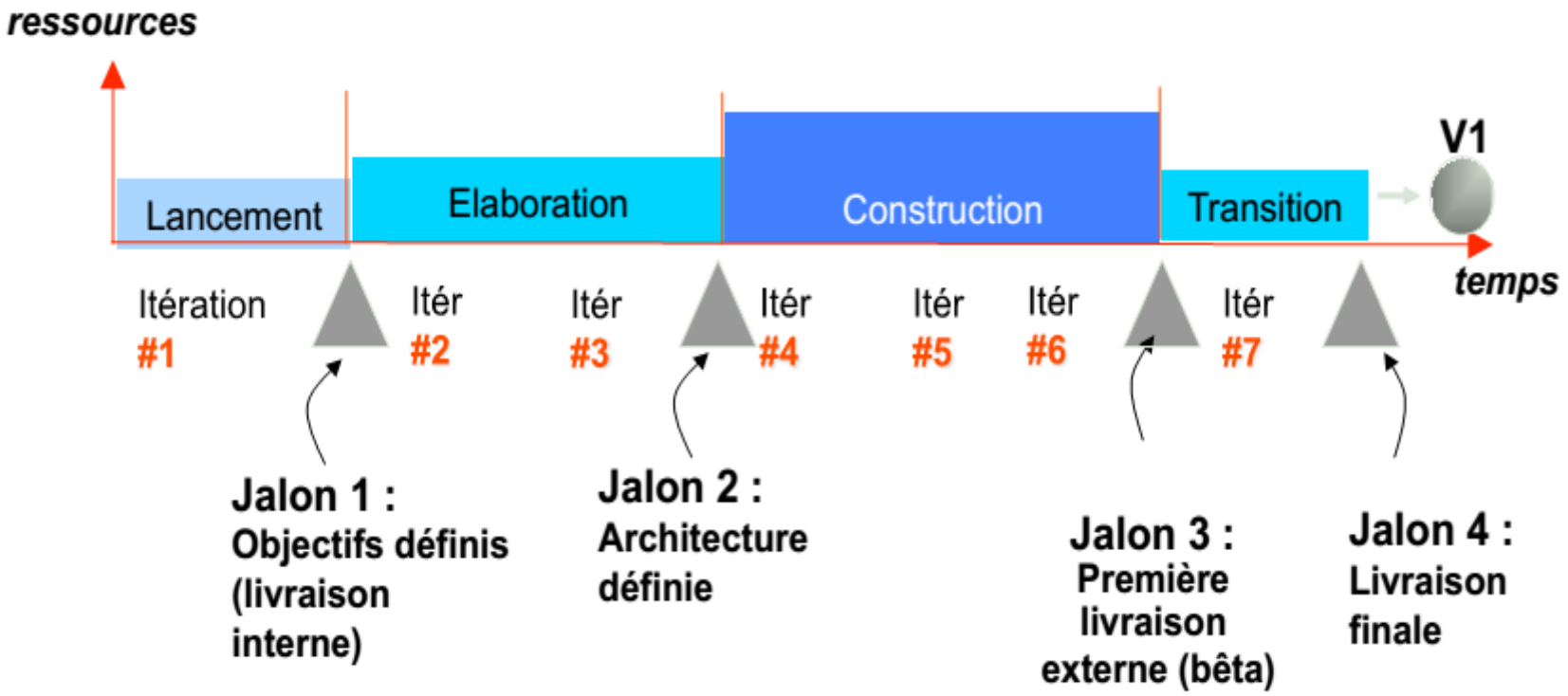
RUP: les quatre phases (2)

- **Initialisation** :
  - Définition du problème.
- **Elaboration** :
  - Planification des activités, affectation des ressources, analyse.
- **Construction** :
  - Développement du logiciel par incréments successifs.
- **Transition** :
  - Recettage et déploiement.

Les phases du développement sont les grandes étapes du développement du logiciel

- Le projet commence en phase d'initialisation et termine en phase de transition

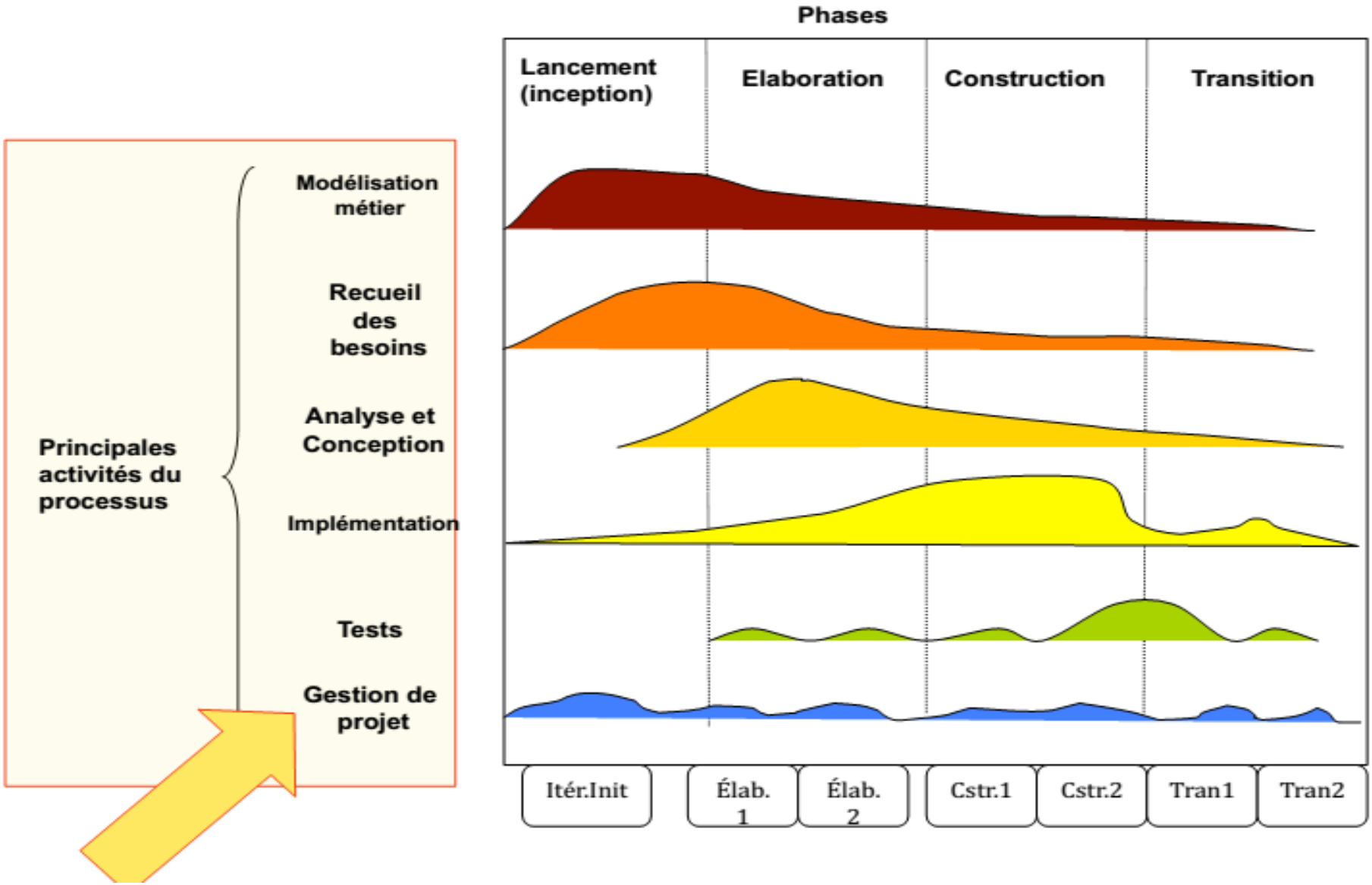
RUP: Déroulement des phases



- Les phases
- Les itérations
- Les jalons (#n)

Les Jalons correspondent à des étapes d'évaluation de la phase terminée, et de lancement de la phase suivante

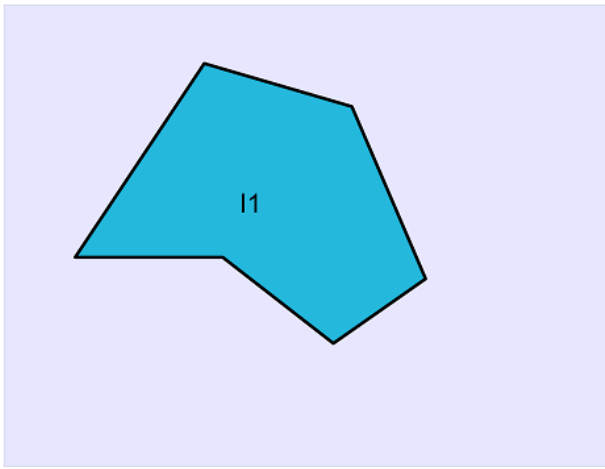
UP : L'enchainement d'activités



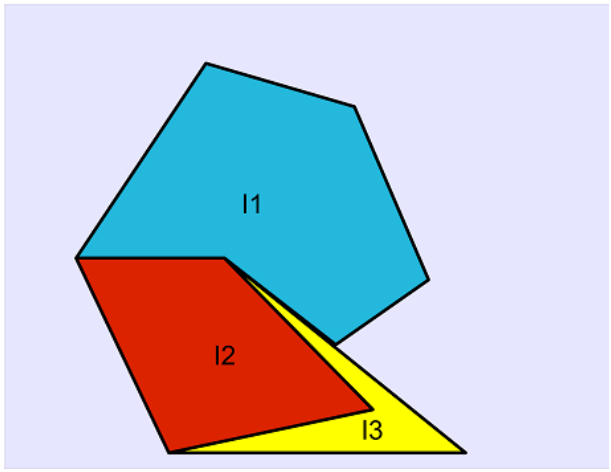
Les itération : exemple (1)

Itérations 0

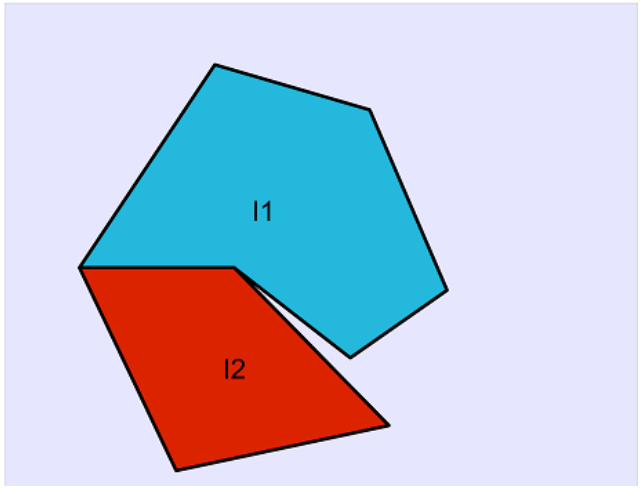
Itérations 1



Itérations 3

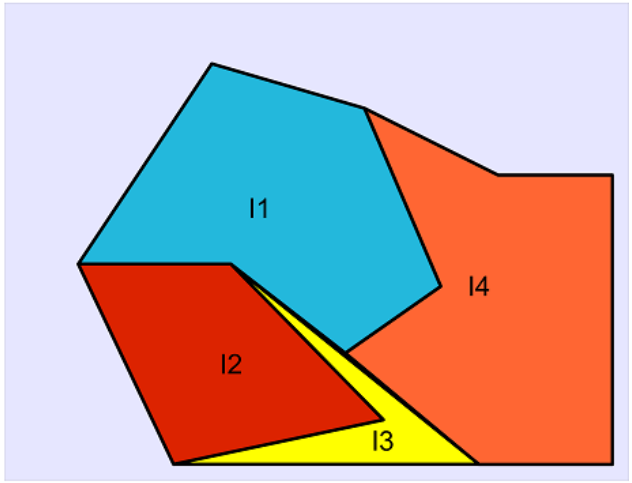


Itérations 2

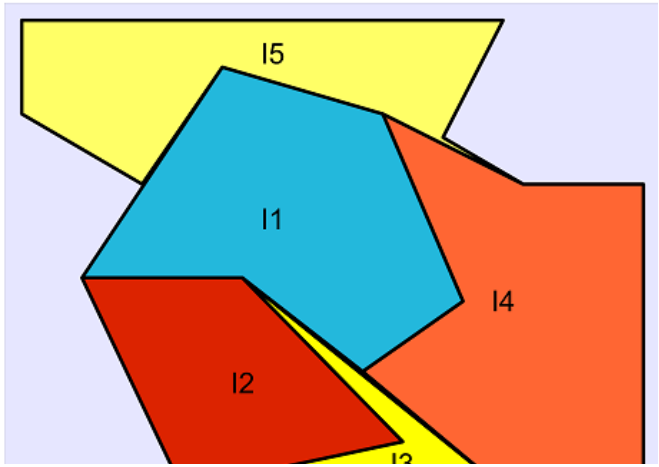


Les itérations: Exemple (2)

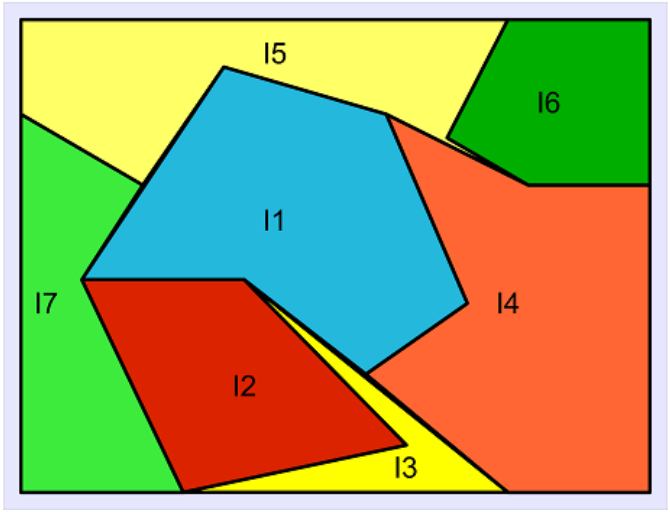
Itérations 4



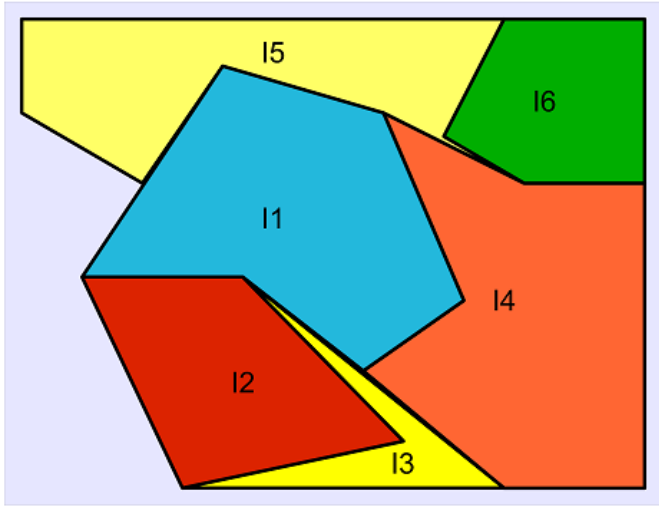
Itérations 5



Itérations 7



Itérations 6



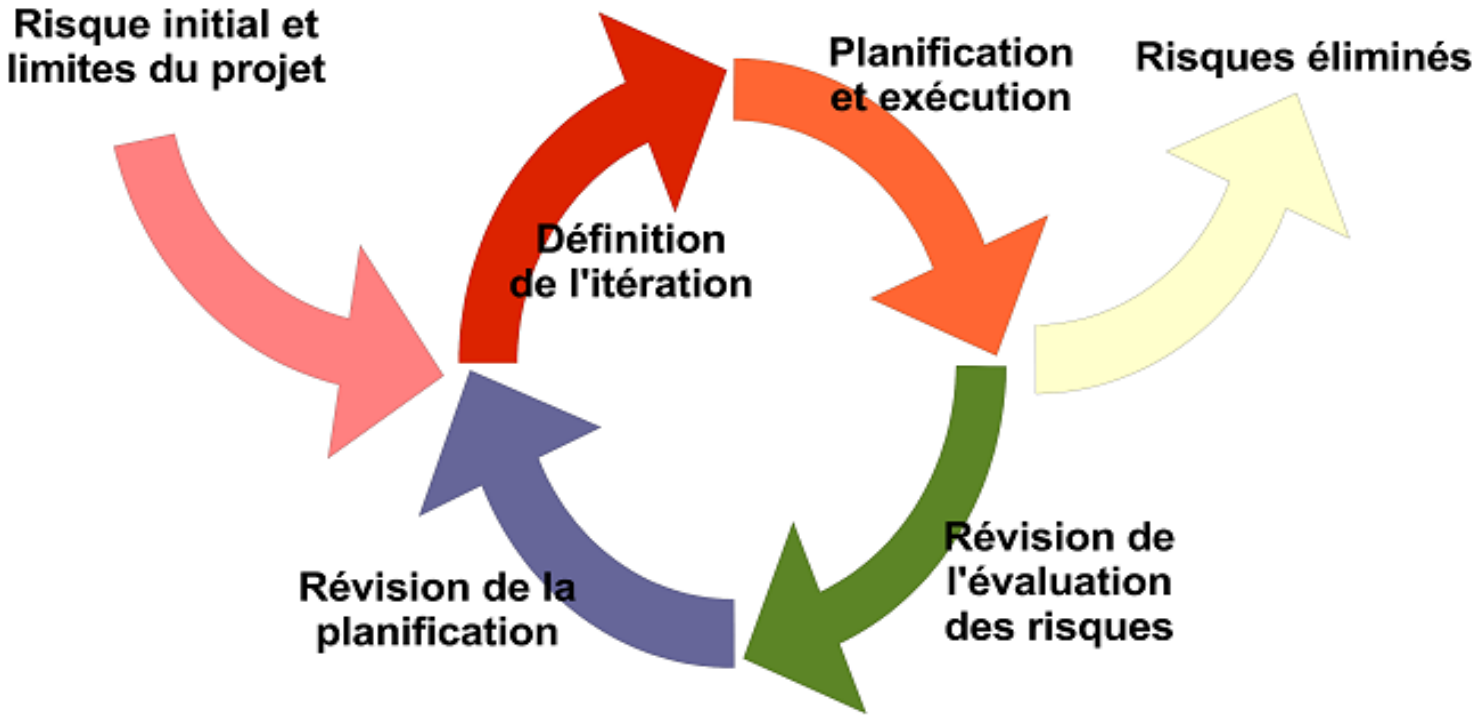
Les itérations

A chaque itération, on refait :

1. Spécification ;
2. Conception ;
3. Implémentation ;
4. Tests.

RUP: élimination de risques à chaque itération(1)

On peut voir le développement d'un logiciel comme un processus graduel d'élimination de risques.



- C'est pendant « **Planification et éxécution** » qu'on répète Spécification → Conception → Implémentation → Tests.

RUP: élimination de risques à chaque itération (2)

Chaque itération prend en compte un certain nombre de cas d'utilisation.

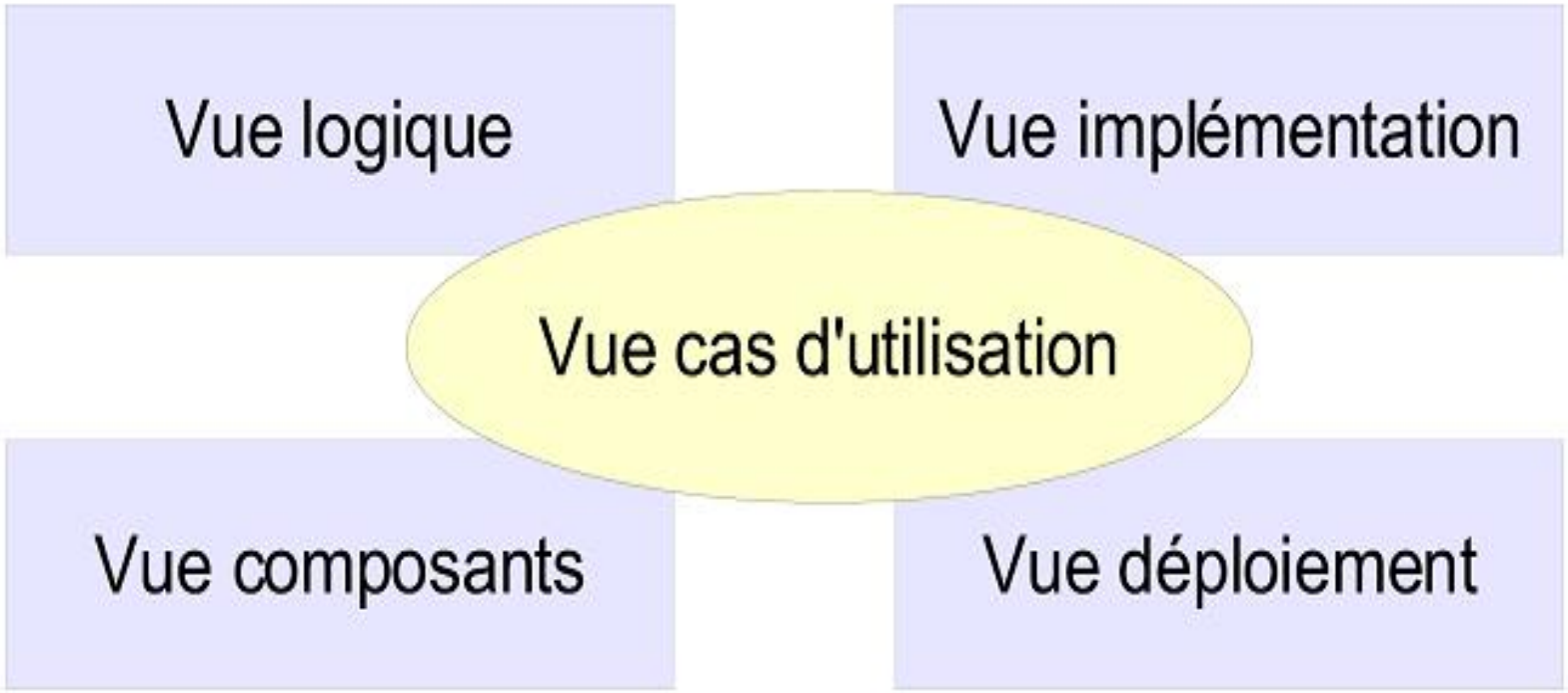
Les risques majeurs sont traités en priorité.

Chaque itération donne lieu à un incrément et produit une nouvelle version exécutable.

RUP: Centré sur l'architecture

Modélisation de différentes pespectives indépendantes et complémentaires.

Architecture en couches et vues de Krutchen.



RUP: Vues du système

- **Vue cas d'utilisation** :
  - Description du système comme un ensemble de transactions du point de vue de l'utilisateur.
- **Vue logique** :
  - Créée lors de la phase d'élaboration et raffinée lors de la phase de construction ;
  - Utilisation de diagrammes de classes, de séquences...
- **Vue composants** :
  - Description de l'architecture logicielle.
- **Vue déploiement** :
  - Description de l'architecture matérielle du système.
- **Vue implémentation** :
  - Description des algorithmes, code source.

Phase1: Initialisation (Objectif)

- Définition du cadre du projet, son concept, et inventaire du contenu ;
- Elaboration des cas d'utilisation critiques ayant le plus d'influence sur l'architecture et la conception ;
- Réalisation d'un ou de plusieurs prototypes démontrant les fonctionnalités décrites par les cas d'utilisation principaux ;
- Estimation détaillée de la charge de travail, du coût et du planning général ainsi que de la phase suivante d'élaboration Estimation des risques.

Phase1: Initialisation (Activités)

- Formulation du cadre du projet, des besoins, des contraintes et des critères d'acceptation ;
- Planification et préparation de la justification économique du projet et évaluation des alternatives en termes de gestion des risques, ressources, planification ;
- Synthèse des architectures candidates, évaluation des coûts.

Phase1: Initialisation (Livrable)

- Un document de vision présentant les besoins de base, les contraintes et fonctionnalités principales ;
- Une première version du modèle de cas d'utilisation ;
- Un glossaire de projet ;
- Un document de justification économique incluant le contexte général de réalisation, les facteurs de succès et la prévision financière ;
- Une évaluation des risques ;
- Un plan de projet présentant phases et itérations ;
- Un ou plusieurs prototypes.

Phase1: Initialisation (Critères d'évaluation)

- Un consensus sur :
  - La planification ;
  - L'estimation des coûts ;
  - La définition de l'ensemble des projets des parties concernées.
- La compréhension commune des besoins.

Phase 2: Elaboration (objectif)

- Définir, valider et arrêter l'architecture ;
- Démontrer l'efficacité de cette architecture à répondre à notre besoin ;
- Planifier la phase de construction.

Phase 2: Elaboration (Activités)

- Elaboration de la vision générale du système, les cas d'utilisation principaux sont compris et validés ;
- Le processus de projet, l'infrastructure, les outils et l'environnement de développement sont établis et mis en place ;
- Elaboration de l'architecture et sélection des composants.

Phase 2: Elaboration (Livrable)

- Le modèle de cas d'utilisation est produit au moins à 80 % ;
- La liste des exigences et contraintes non fonctionnelles identifiées ;
- Une description de l'architecture ;
- Un exécutable permettant de valider l'architecture du logiciel à travers certaines fonctionnalités complexes ;
- La liste des risques revue et la mise à jour de la justification économique du projet ;
- Le plan de réalisation, y compris un plan de développement présentant les phases, les itérations et les critères d'évaluation ;

Phase 2: Elaboration (Critère d'évaluation)

- La stabilité de la vision du produit final ;
- La stabilité de l'architecture ;
- La prise en charge des risques princip aux est adressée par le(s) prototype(s) ;
- La définition et le détail du plan de projet pour la phase de construction ;
- Un consensus, par toutes les parties prenantes, sur la réactualisation de la planification, des coûts et de la définition de projet.

Phase 3 : Construction (Objectif)

- La minimisation des coûts de développement par :
  - L'optimisation des ressources ;
  - La minimisation des travaux non nécessaires.
- Le maintien de la qualité ;
- Réalisation des versions exécutables.

Phase 3 : Construction (Activités)

- Gestion et le contrôle des ressources et l'optimisation du processus de projet ;
- Evaluation des versions produites en regard des critères d'acceptation définis.

Phase 3 : Construction (Livrables)

- Les versions exécutables du logiciel correspondant à l'enrichissement itération par itération des fonctionnalités ;
- Les manuels d'utilisation réalisés en parallèle à la livraison incrémentale des exécutables ;
- Une description des versions produites.

Phase 3 : Construction (Critères d'évaluation)

- La stabilité et la qualité des exécutables ;
- La préparation des parties prenantes ;
- La situation financière du projet en regard du budget initial.

Phase 4: Transition (Objectifs)

- Le déploiement du logiciel dans l'environnement d'exploitation des utilisateurs ;
- La prise en charge des problèmes liés à la transition ;
- Atteindre un niveau de stabilité tel que l'utilisateur est indépendant ;
- Atteindre un niveau de stabilité et qualité tel que les parties prenantes considèrent le projet comme terminé.

Phase 4: Transition (Activités)

- Activités de « packaging » du logiciel pour le mettre à disposition des utilisateurs et de l'équipe d'exploitation ;
- Correction des erreurs résiduelles et amélioration de la performance et du champ d'utilisation ;
- Evaluation du produit final en regard des critères d'acceptation définis.

Phase 4: Transition (Livrable)

- La version finale du logiciel ;
- Les manuels d'utilisation.

Phase 4: Transition (Critères d'évaluation)

- La satisfaction des utilisateurs ;
- La situation financière du projet en regard du budget initial.

Les activités

- Chaque phase comprend plusieurs itérations
- Pour chacune des itérations, on se livre à plusieurs activités :
  - Modélisation métier ;
  - **Expression des besoins** ;
  - **Analyse** ;
  - **Conception** ;
  - **Implémentation** ;
  - **Test** ;
  - Déploiement.

- Les **activités** sont des étapes dans le développement d'un logiciel, mais à un niveau de granularité beaucoup plus fin que les phases.
- Chaque activité est répétée autant de fois qu'il y a d'itérations.

Activité de Modélisation métiers

- Objectif : Mieux comprendre la structure et la dynamique de l'organisation :
  - Proposer la meilleure solution dans le contexte de l'organisation cliente ;
  - Réalisation d'un glossaire des termes métiers ;
  - Cartographie des processus métier de l'organisation cliente.
- Activité coûteuse mais qui permet d'accélérer la compréhension d'un problème.

Activité d'expression des besoins

- **Objectif :** Cibler les besoins des utilisateurs et du clients grâce à une série d'interviews.
  - L'ensemble des parties prenantes du projet, maîtrise d'oeuvre et maîtrise d'ouvrage, est acteur de cette activité.
- L'activité de recueil et d'expression des besoins débouche sur ce que doit faire le système (question « **QUOI ?** »)

Activité d'expression des besoins

- Utilisation des **cas d'utilisation** pour :
  - Schématiser les besoins ;
  - Structurer les documents de spécifications fonctionnelles.
- Les cas d'utilisation sont décomposés en scénarios d'usage du système, dans lesquels l'utilisateur « raconte » ce qu'il fait grâce au système et ses interactions avec le système.
- Un maquettage est réalisable pour mieux « immerger » l'utilisateur dans le futur système.
- Une fois posées les limites fonctionnelles, le projet est planifié et une prévision des coûts est réalisée.

Activité d'analyse

- **Objectif :** Transformer les besoins utilisateurs en modèles UML.
  - Analyse objet servant de base à une réflexion sur les mécanismes internes du système.
- Principaux livrables :
  - Modèles d'analyse, neutre vis à vis d'une technologie ;
  - Livre une spécification plus précise des besoins.
- Peut envisagé comme une première ébauche du modèle de conception.

Activité de Conception

- **Objectif :** Modéliser **comment** le système va fonctionner :
  - Exigences non fonctionnelles ;
  - Choix technologiques.
- Le système est analysé et on produit :
  - Une proposition d'architecture ;
  - Un découpage en composants.

Activité d'Implémentation

- **Objectif :** Implémenter le système par composants.
  - Le système est développé par morceaux dépendant les uns des autres.
  - Optimisation de l'utilisation des ressources selon leurs expertises.
- Les découpages fonctionnel et en couches sont indispensable pour cette activité.
- Il est tout à fait envisageable de retoucher les modèles d'analyse et de conception à ce stade.

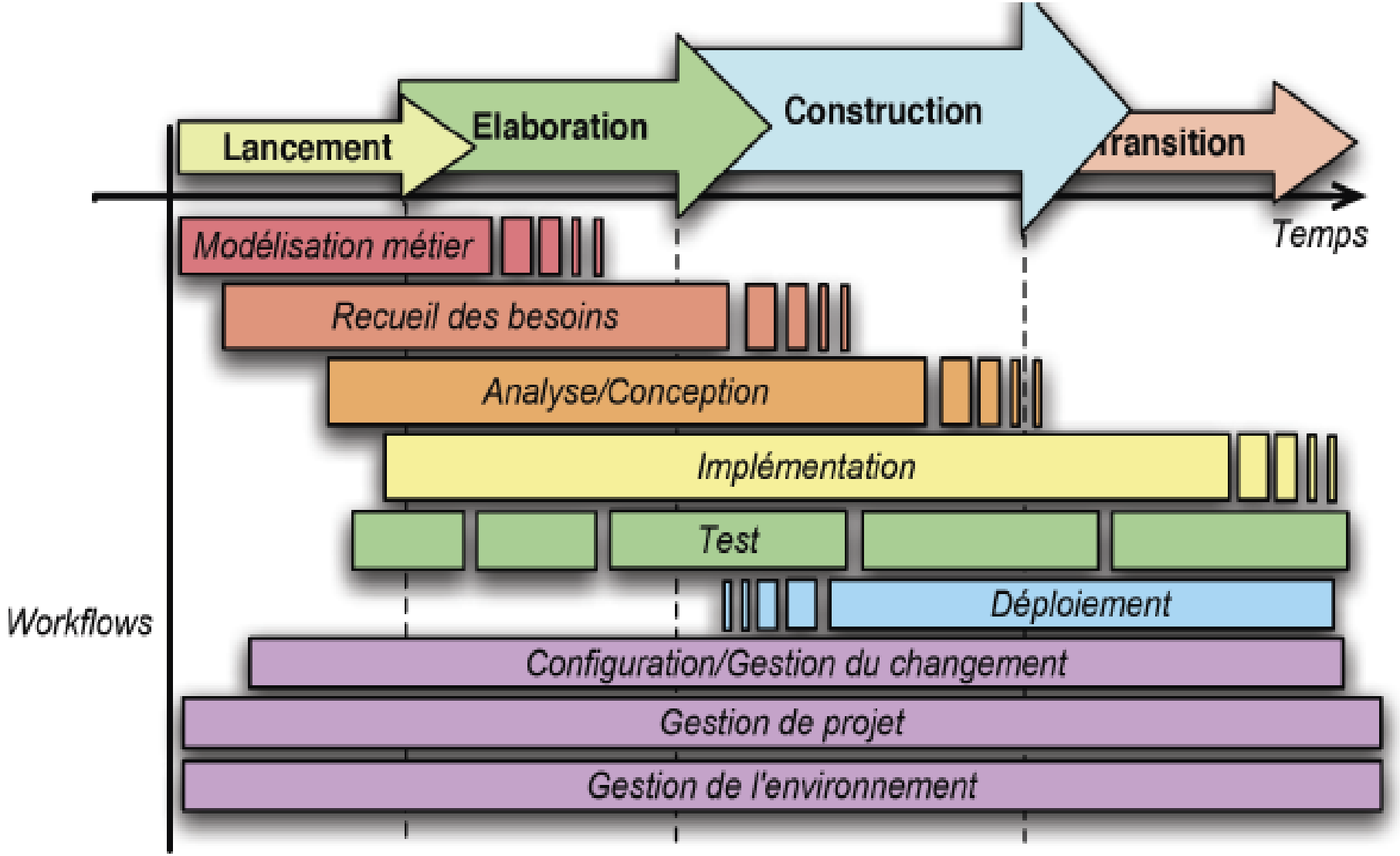
Activité de Test

- **Objectif :** Vérifier des résultats de l'implémentation en testant la construction :
  - **Tests unitaires :** tests composants par composants ;
  - **Tests d'intégration :** tests de l'interaction de composants préalablement testés individuellement.
- Méthode :
  - Planification pour chaque itération ;
  - Implémentation des tests en créant des cas de tests ;
  - Exécuter les tests ;
  - Prendre en compte le résultat de chacun.

Activité de déploiement

- **Objectif :** Déployer les développements une fois réalisés.
  - Peut être réalisé très tôt dans le processus dans une sousactivité de prototypage dont l'objectif est de valider :
    - L'architecture physique ;
    - Les choix technologiques.

Répartition et importance des activités dans chaque phase



Les principaux diagrammes par activité

- **Expression des besoins et modélisation métier :**
  - Modèles métier, domaine, cas d'utilisation ;
  - Diagramme de séquences ;
  - Diagramme d'activité.
- **Analyse**
  - Modèles métier, cas d'utilisation ;
  - Diagramme des classes, de séquences et de déploiement.
- **Conception**
  - Diagramme des classes, de séquences ;
  - Diagramme état/transition ;
  - Diagramme d'activité ;
  - Diagramme de déploiement et de composant.

RUP: Conclusion

- Le RUP est à la fois une méthodologie et un outil prêt à l'emploi (référentiel Web)
- Cible des projets de plus de 10 personnes
- Points forts
  - Spécifie le dialogue entre les différents intervenants du projet : les livrables, les plannings, les prototypes…
  - Propose des modèles de documents, et des canevas pour des projets types
- Points faibles
  - Coûteux à personnaliser : batterie de consultants
  - Très axé processus, au détriment du développement