UML
2/5 – Processus Unifié

Aurélien Tabard
Département Informatique
Université Claude Bernard Lyon 1

Basé sur le cours de Yannick Prié

Objectifs du cours global

Notion de méthode de conception de Système Informatique (SI)

Plan des 5 séances :

.: Généralités sur les méthodes

.: Processus unifié

.: UML : Cas d’utilisation

.: UML : Diagrammes de séquence

.: UML : Diagrammes d’état

.: Processus agiles

Plan

Trame du processus unifié

Caractéristiques essentielles

Plan

Trame du processus unifié

Caractéristiques essentielles

Unified Software Development Process

.: Développé dans les années 90 par Rumbaugh, Booch, Jacobson (les concepteurs originaux d'UML)

.: Purement objet

.: Gérer un projet logiciel de bout en bout

.: Les généralités présentées sur USDP s'appliquent à beaucoup de méthodes objets

Unified Software Development Process

Un processus capable de

.: dicter l'organisation des activités de l'équipe

.: diriger les tâches de chaque individu et de l'équipe dans son ensemble

.: spécifier les artefacts à produire

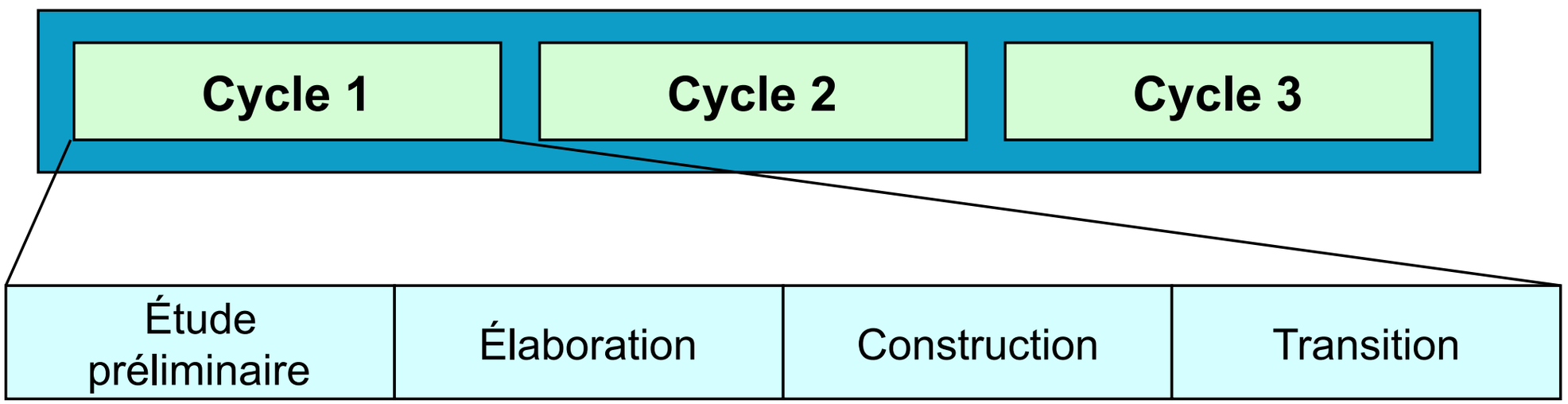
.: proposer des critères pour le contrôle de produits et des activités de l'équipe

Regroupement de bonnes pratiques, mais

.: non figé

.: générique (hautement adaptable : individus, cultures, …)

Les 4 phases du cycle de vie



.: Considérer un produit logiciel par rapport à ses versions

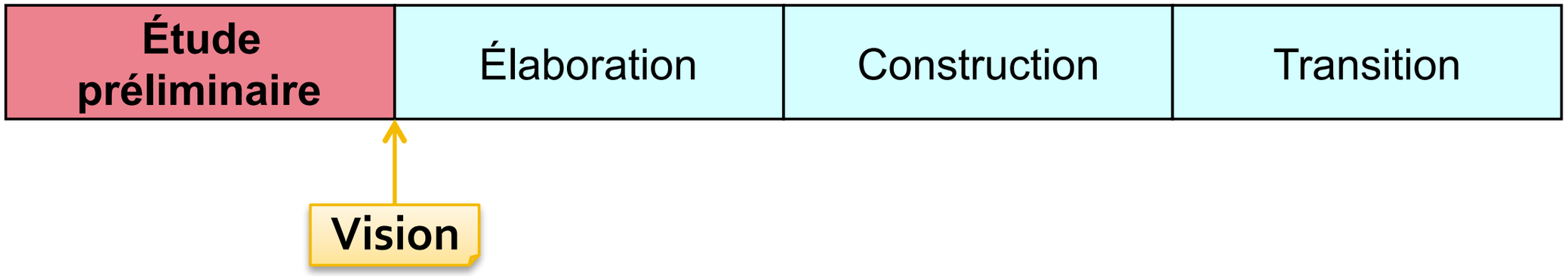
  .: un cycle produit une version

.: Gérer chaque cycle de développement comme un projet ayant quatre phases

  .: vue gestionnaire (manager)

  .: chaque phase se termine par un point de contrôle (ou jalon) permettant aux chefs de projet de prendre des décisions

Phase 1 : Etude préliminaire



.: Déterminer

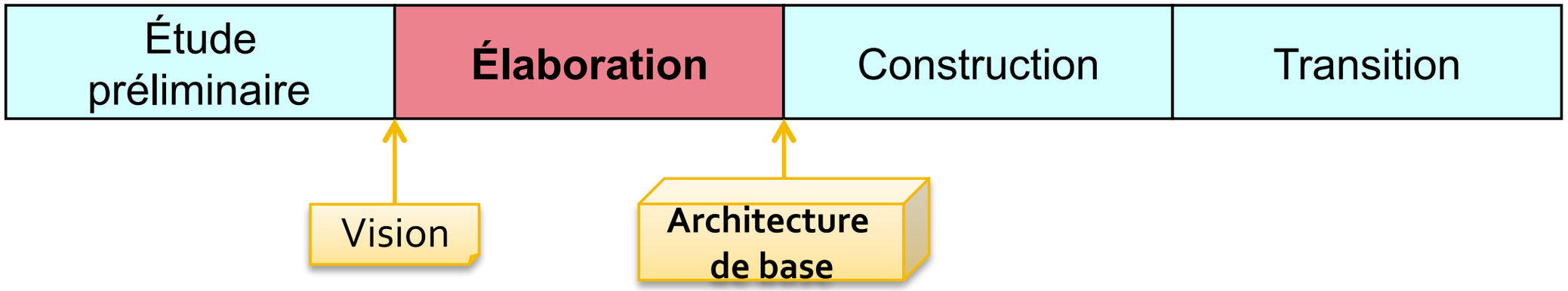
- .: que fait le système ?
- .: à quoi pourrait ressembler l'architecture ?
- .: quels sont les risques ?
- .: quel est le coût estimé du projet ? Comment le planifier ?

.: Jalon

- .: « vision du projet » = document
- .: Accepter le projet ?

.: Coût faible

Phase 2 : Elaboration



.: Spécifier la plupart des cas d'utilisation

.: Concevoir l'architecture de base (squelette du système)

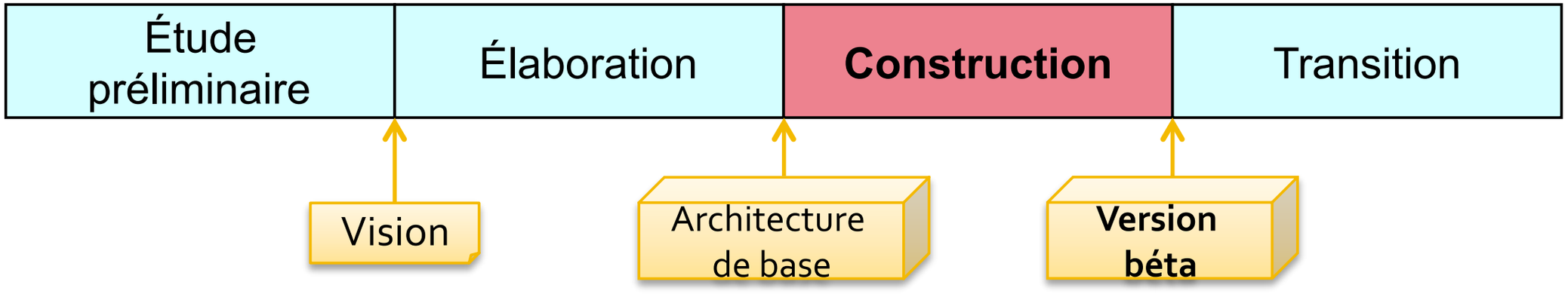
.: Mettre en œuvre cette architecture

.: Faire une planification complète

.: Jalon

- .: « architecture du cycle de vie » = premier prototype qui démontre la validité des choix architecturaux
- .: Peut-on passer à la construction ? (besoins, architecture, planning stables ? Risques contrôlés ?)

Phase 3 : Construction



.: Développer par incréments

- .: l'architecture reste stable malgré des changements mineurs
- .: le produit contient au final tout ce qui avait été planifié, il reste quelques erreurs

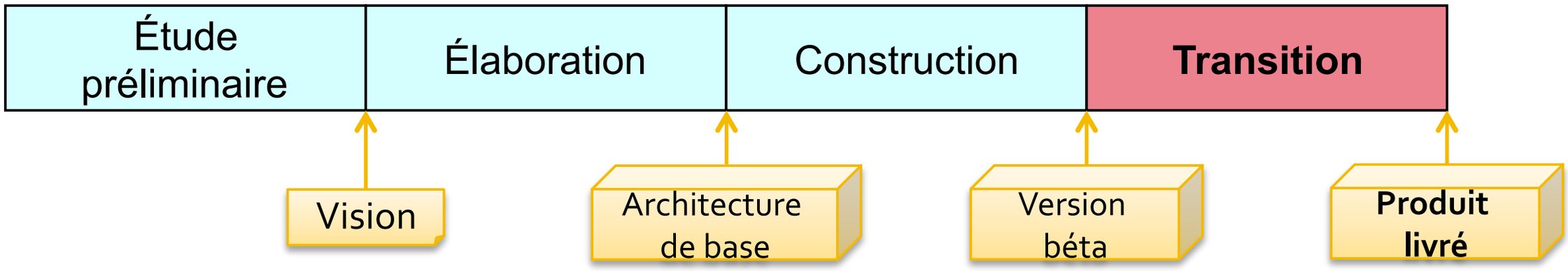
.: Jalon

- .: « capacité opérationnelle initiale » = version béta
- .: Le produit est-il suffisamment correct pour être installé chez le / un client ?

.: Phase la plus coûteuse

- .: > 50% du cycle
- .: englobe conception, codage, tests, intégration, etc.

Phase 4 : Transition



.: Transition

- .: livrer / déployer le produit
- .: corriger le reliquat d'erreurs
- .: améliorer le produit
- .: former les utilisateurs
- .: lettre en place l'assistance en ligne

.: Jalon

- .: « livraison du produit » = produit déployé chez le client
- .: tests suffisants ? Produit satisfaisant ? Manuels prêts ?

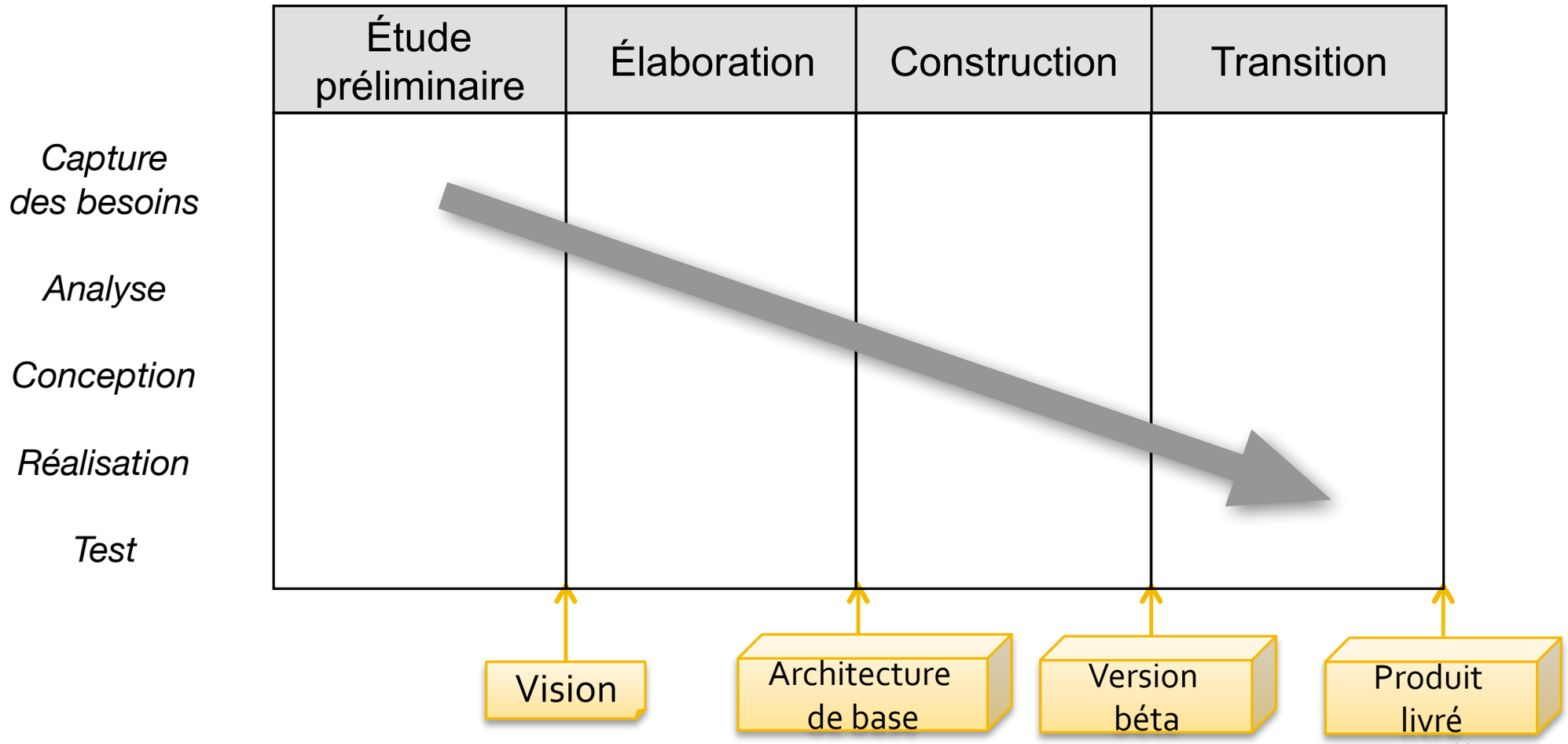
Phases et Activités

Un cycle met en jeu des activités

.: vue du développement sous l'angle technique (développeur)

.: les activités sont réalisées au cours des phases

.: les activités



Plan

Trame du processus unifié

Processus unifié : caractéristiques essentielles

.: itératif et incrémental

.: piloté par les besoins

.: piloté par les risques

.: centré sur l’architecture

Itérations et incréments

.: Anti-cascade, spirale

.: Construction progressive du système

  .: processus incrémental

.: Chaque itération permet de maîtriser une partie des risques et apporte une preuve tangible de faisabilité

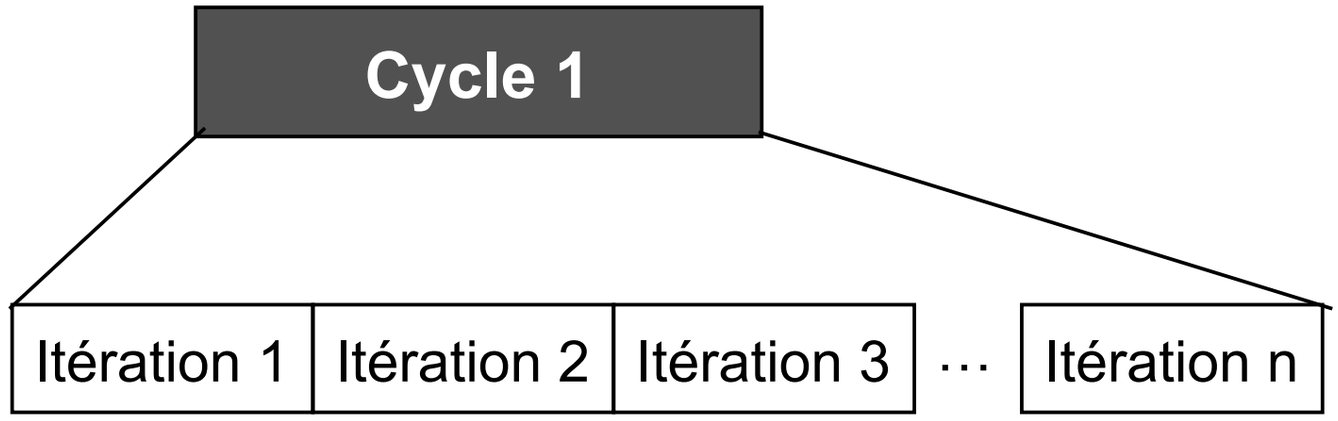
  .: produit un système partiel opérationnel (exécutable, testé et intégré) avec une qualité égale à celle d'un produit fini (alpha)

  .: qui peut être évalué : permet de savoir si on va dans la bonne direction ou non

.: Série de prototypes qui vont en s'améliorant

  .: de plus en plus de fonctionnalités disponibles

  .: retours utilisateurs



Une itération

.: est un mini-projet :

  .: plan pré-établi et objectifs pour le prototype, critères d'évaluation,

.: comporte toutes les activités (mini-cascade)

.: est terminée par un point de contrôle

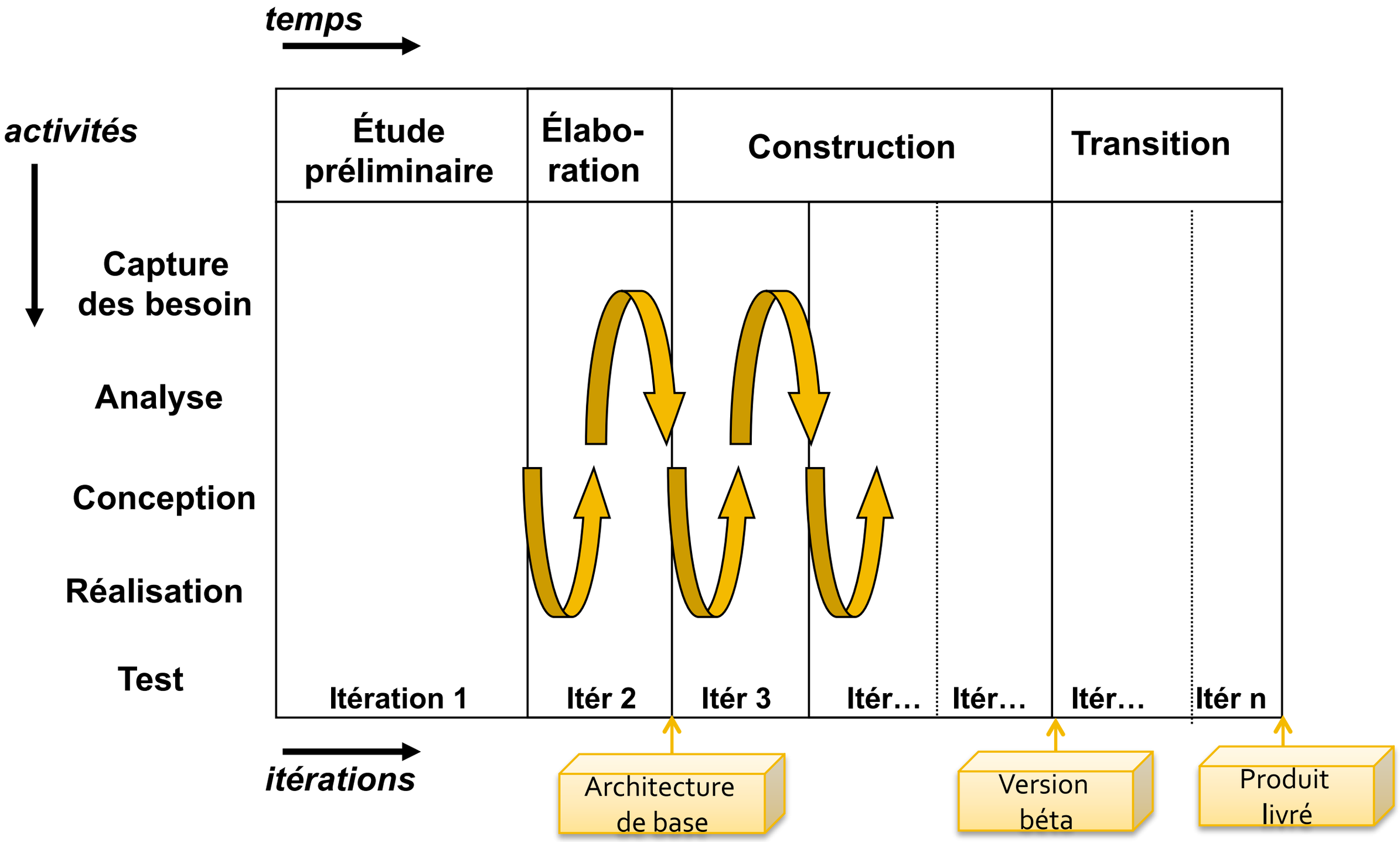
  .: ensemble de modèles agréés, décisions pour les itérations suivantes

  .: conduit à une version montrable implémentant un certain nombre de Cas d'Utilisation (CU).

.: dure entre quelques semaines et 9 mois (au delà : danger)

.: butée temporelle qui oblige à prendre des décisions

Itérations et phases



Avantages d'un processus itératif & incrémental

Gestion de la complexité

.: pas tout en même temps, étalement des décisions importantes

Maîtrise des risques élevés précoce

.: diminution de l'échec

.: architecture mise à l'épreuve rapidement (prototype réel)

Intégration continue

.: progrès immédiatement visibles

.: maintien de l'intérêt des équipes
(court terme, prototypes vs documents)

Avantages d'un processus itératif & incrémental

Prise en compte des modifications de besoins

.: feedback, implication des utilisateurs et adaptation précoce

Apprentissage rapide de la méthode

.: amélioration de la productivité et de la qualité du logiciel

Adaptation de la méthode

.: possibilité d'explorer méthodiquement les leçons tirées d'une itération (élément à conserver, problèmes, éléments à essayer…)

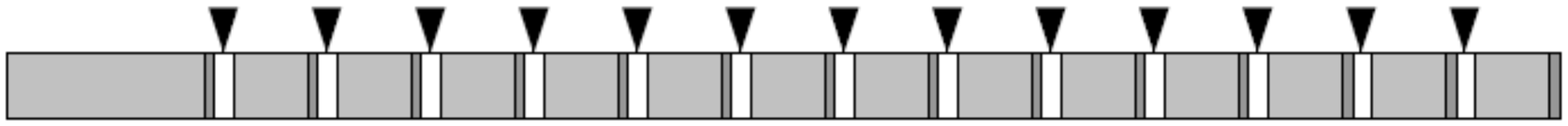
Mais gestion de projet plus complexe : planification adaptative

Résumé

Démarche “en bloc”

Validation effective

Démarche itérative



Plan

Trame du processus unifié

Processus unifié : caractéristiques essentielles

.: itératif et incrémental

.: piloté par les besoins

.: piloté par les risques

.: centré sur l’architecture

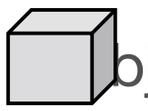
Un processus piloté par les besoins

Objectif du processus

.: construction d'un système qui réponde à des besoins

.: par construction complexe de modèles

Cas d'utilisation = expression / spécification des besoins

.: CU portée système  objectifs utilisateurs 

CU utilisés tout au long du cycle

.: validation des besoins / utilisateurs

.: point de départ pour l'analyse (découverte des objets, de leurs relations, de leur comportement) et la conception (sous-systèmes)

.: guide pour la construction des interfaces

.: guide pour la mise au point des plans de tests

Les CU lient les modèles

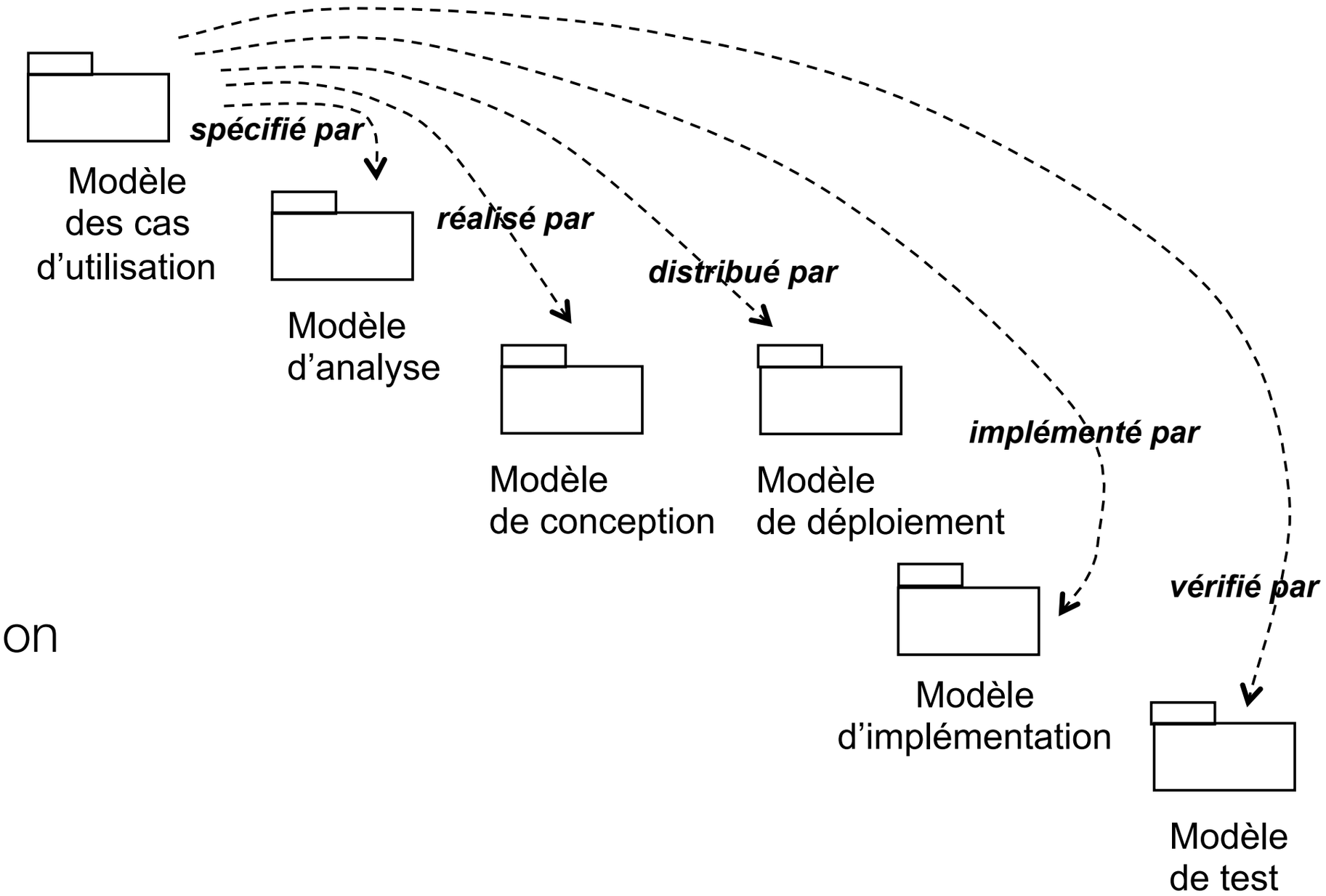
Besoins

Analyse

Conception

Implémentation

Tests



Avantages des cas d'utilisation

- Centrés utilisateurs
  - support de communication en langue naturelle entre utilisateurs et concepteurs basé sur les scénarios (et non liste de fonctions)
  - dimension satisfaction d'un objectif utilisateur
  - également pour les utilisateurs informaticiens (administration)
    - besoins fonctionnels, pour acteurs humains et non humains à identifier précisément
- Assurent la traçabilité par rapports aux besoins de toute décision de conception sur l'ensemble du projet
  - tout modèle peut se référer in fine à un CU
  - Fournissent une vision commune aux participants du projet
  - management, conception, qualité, marketing, etc.
- Attention :
  - créer de bons CU est un art (voir prochain cours)

Plan

Trame du processus unifié

Processus unifié : caractéristiques essentielles

.: itératif et incrémental

.: piloté par les besoins

.: piloté par les risques

.: centré sur l’architecture

Risque ?

Risque que le projet de construction du système soit un échec

.: Différentes natures de risques pour un projet

.: besoins / technique / autres

Exemples

.: le système construit n’est pas le bon

.: architecture inadaptée, utilisation de technologies mal maîtrisées, performances insuffisantes

.: personnel insuffisant, problèmes commerciaux ou financiers (risques non techniques, mais bien réels)

Gérer les risques

Identifier et classer les risques par importance

Agir pour diminuer les risques

.: ex. changer les besoins, confiner la portée à une petite partie du projet,
faire des test pour vérifier leur présence et les éliminer

S’ils sont inévitables, les évaluer rapidement

.: tout risque fatal pour le projet est à découvrir au plus tôt

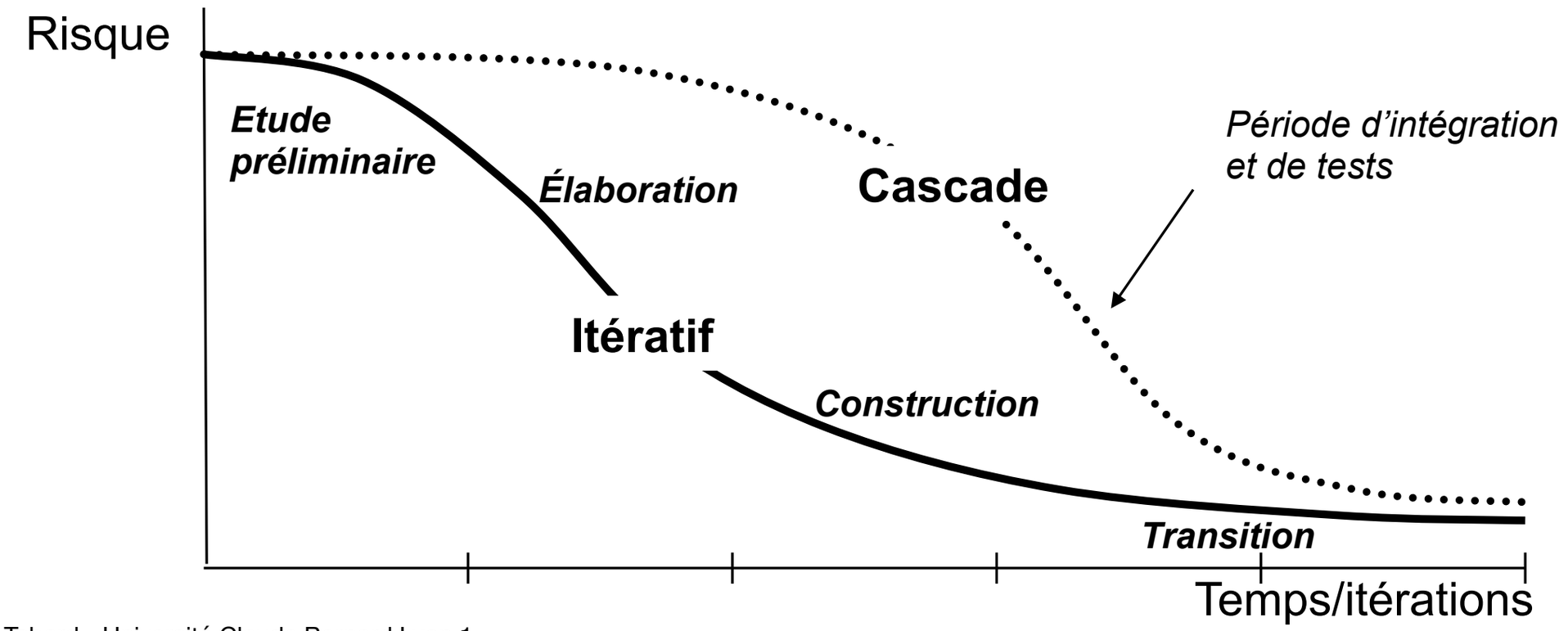
Pilotage par les risques et itérations

.: Identifier, lister et évaluer la dangerosité des risques

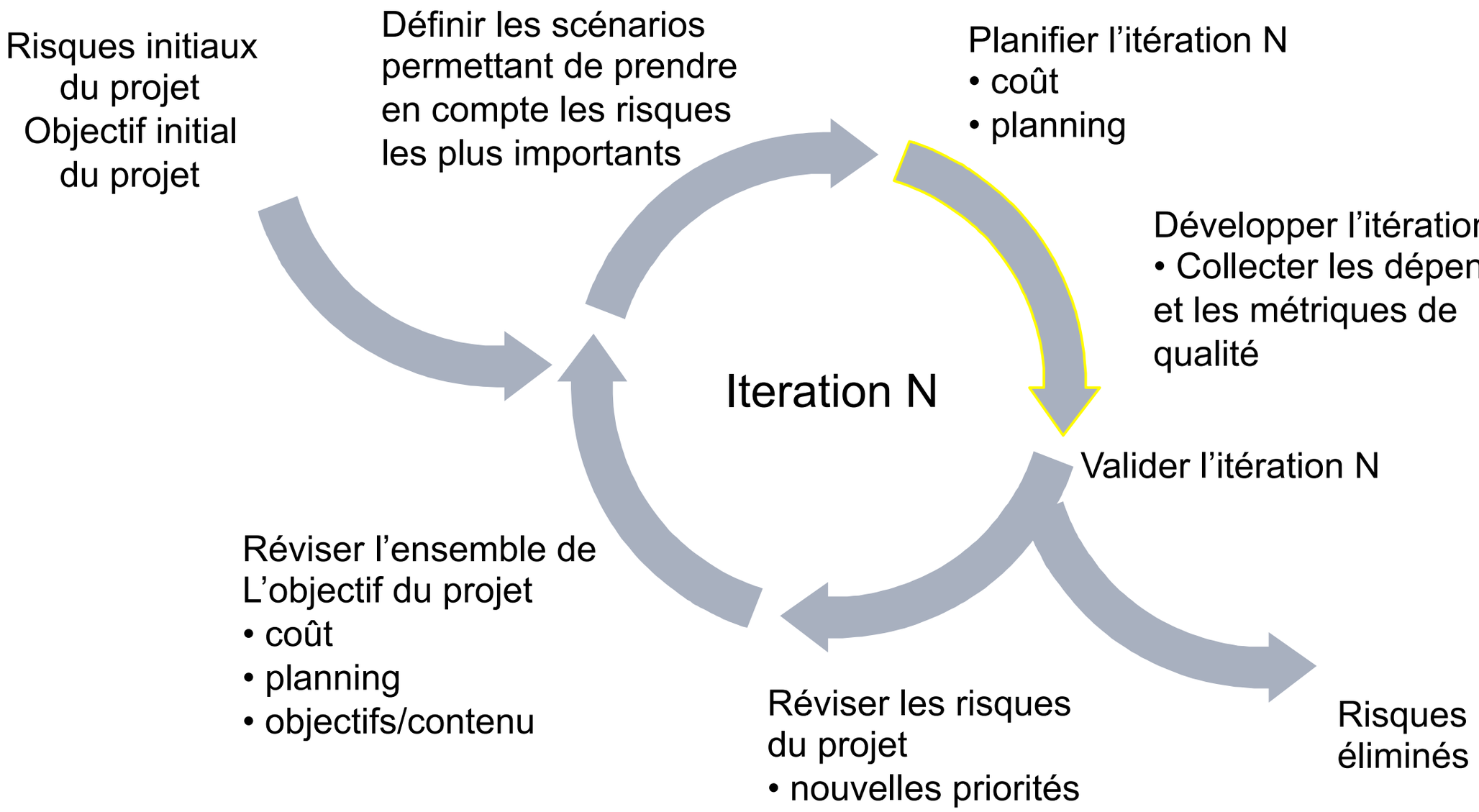
.: Construire les itérations en fonction des risques : s'attaquer en priorité aux risques les plus importants qui menacent le plus la réussite du projet

  .: « provoquer des changements précoces »

  .: ex. stabiliser l'architecture et les besoins liés le plus tôt possible



Itération pilotées par la réduction des risques



Itérations et risque

On ordonne les itérations à partir des priorités établies pour les cas d'utilisation et de l'étude du risque

.: plan des itérations

.: chaque prototype réduit une part du risque
et est évalué comme tel

.: les priorités et l'ordonnancement de construction des prototypes peuvent changer avec le déroulement du plan

Planification des itérations

- Planification des itérations dès l'élaboration
  - précise pour l'itération suivante, imprécise pour la fin
  - la planification générale est adaptée en fonction des risques identifiés/traités et des résultats obtenus dans les itérations
- Planification adaptative (vs. prédictive)
  - fixer des butées temporelles
  - gérer l'adaptation avec le client
- Itération
  - toutes les activités des besoins aux tests, résultats différents en fonction de la phase dans laquelle on se trouve
  - mini-projet : planning, ressources, revue
  - premières itérations : suivre une méthode
  - à la fin d'une itération : retrospective
    - éléments à conserver, problèmes, éléments à essayer

Plan

Trame du processus unifié

Processus unifié : caractéristiques essentielles

.: itératif et incrémental

.: piloté par les besoins

.: piloté par les risques

.: centré sur l’architecture

Processus centré sur l'architecture

.: L'architecture sert de lien pour l'ensemble des membres du projet

- .: réalisation concrète de prototypes incrémentaux qui « démontrent » les décisions prises
- .: vient compléter les cas d'utilisation comme « socle commun »

.: Contrainte de l'architecture

- .: plus le projet avance, plus l'architecture est difficile à modifier
- .: -> les risques liés à l'architecture sont très élevés, car très coûteux

.: Objectif pour le projet

- .: établir dès la phase d'élaboration des fondations solides et évolutives pour le système à développer, en favorisant la réutilisation
- .: l'architecture s'impose à tous, contrôle les développements ultérieurs, permet de comprendre le système et d'en gérer la complexité

.: L'architecture est contrôlée et réalisée par l'architecte du projet

Objectif / architecture

Construire une architecture

.: comme forme dans laquelle le système doit s'incarner

.: les CU réalisés doivent y trouver leur place

.: la réalisation des CU suivant doit s'appuyer sur l'architecture.

.: qui permette de promouvoir la réutilisation

.: qui ne change pas trop

.: converger vers une bonne architecture rapidement

Analyse architecturale

Identifier et traiter les besoins non fonctionnels dans le contexte des besoins fonctionnels

Identifier les points de variation et d'évolution les plus probables

- .: points de variation : variations dans le système existant ou dans les besoins
- .: points d’évolution : points de variation spéculatifs, actuellement absents des besoins

Construction de l'architecture : principe (1)

Pour commencer

.: choix d'une architecture de haut-niveau et construction des parties générales de l'application

.: ébauche à partir

- .: de solutions existantes
- .: de la compréhension du domaine
- .: de parties générales aux applications du domaine (quasi-indépendant des CU)
- .: des choix de déploiement

Construction de l’architecture : principe (2)

Construction de l’architecture de référence :

.: confrontation à l’ébauche des cas d’utilisation les plus significatifs (un à un)

.: construction de parties de l’application réelle (sous-systèmes spécifiques)

.: stabilisation de l’architecture autour des fonctions essentielles (sous-ensemble des CU).

.: traitement des besoins non fonctionnel dans le contexte des besoins fonctionnels

.: identification des points de variation et les points d'évolution les plus probables.

Construction de l’architecture : principe (3)

Réalisation incrémentale des cas d’utilisation

.: itérations successives

.: l’architecture continue à se stabiliser sans changement majeur

Remarque : maturation des cas d’utilisation

.: plus faciles à exprimer et plus précis quand un début d’application est disponible

L’architecture est conçue et réalisée pendant la phase d’élaboration

.: Phase d’élaboration

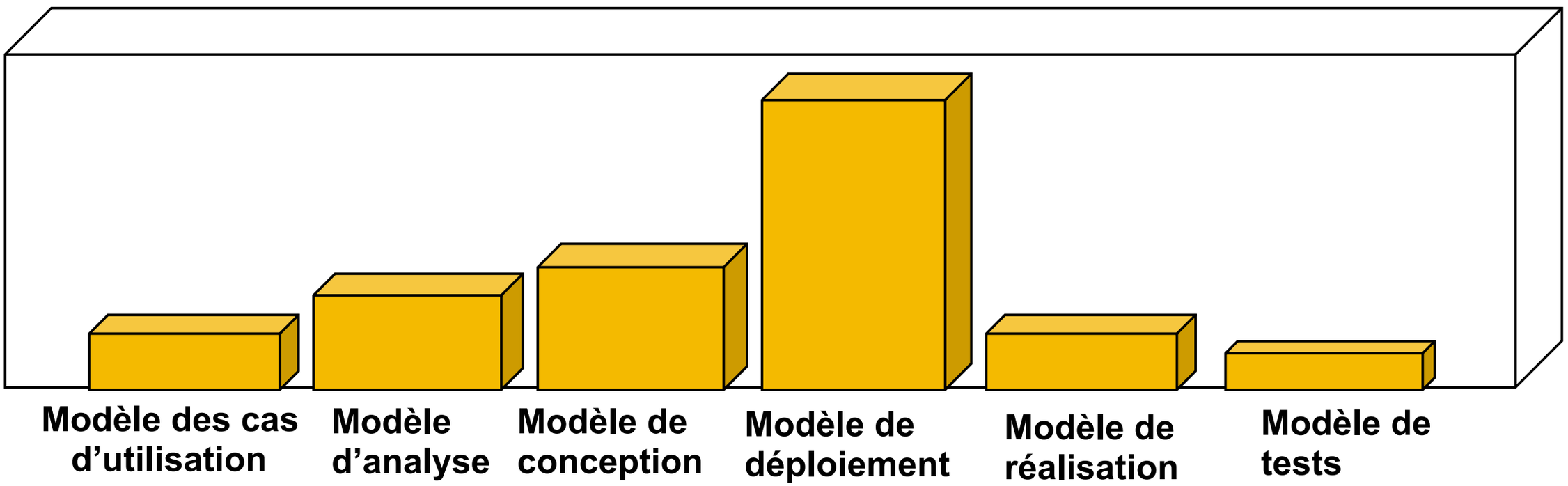
.: aller directement vers une architecture robuste, à coût limité, appelée « architecture de référence »

.: 10% des classes suffisent

.: L’architecture de référence

.: permettra d’intégrer les CU incrémentalement pendant la construction

.: guidera le raffinement et l’expression des CU pas encore détaillés



Conclusion

Les 4 grandes caractéristiques du processus unifié sont liées

Processus itératif et incrémental

Piloté par les risques

Piloté par les cas d’utilisation

Centré sur l’architecture