lab-sticc.univ-brest.fr/~babau/

Méthodes de conception pour les logiciels

Jean-Philippe Babau

Département Informatique, UFR Sciences, UBO
Laboratoire Lab-STICC

Plan

- Introduction
  - Le projet
  - Objectifs d'une méthode
- Les méthodes de développement
  - Introduction
  - Le cycle en V
  - Compléments
- Gestion de projet

UBO

Plan

- Introduction
  - Le projet
  - Objectifs d'une méthode
- Les méthodes de développement
  - Introduction
  - Le cycle en V
  - Compléments
- Gestion de projet

UBO

Qu'est-ce qu'un projet

- Ensemble d'activités pour atteindre un objectif précisément défini
- Un projet
  - Un objectif clairement identifié et quantifié
  - Un délai de réalisation
  - Avec des contraintes de réalisation
    - Moyens humains
    - Compétences techniques
    - Matériel disponibles
    - Budgets

UBO

De la gestion de projet

- Les projets informatiques n'atteignent pas souvent leurs objectifs
  - Dépassement de délais
  - Surcouts
  - Qualité insuffisante
  - Maitriser les délais, les coûts et la qualité
- Les projets sont menés à plusieurs
  - Acteurs divers (informaticiens et non informaticiens)
  - Sous-traitance (offshore)
  - Maitriser la gestion de projet

UBO

Principes généraux de la gestion de projet

- Un projet répond à une demande
  - Identifier les objectifs : bien comprendre les besoins
  - Les besoins sont clairement explicités
- Établir les actions à mener pour répondre aux besoins
  - Mobiliser des personnes et des moyens
  - Organiser cette mobilisation
    - Plan d'actions, délais, coûts
  - Gérer des ressources humaines et matérielles
    - évaluation et suivi des tâches
- Repérer les éléments critiques
- Maitriser la communication
  - Être clair

=> Formaliser et adapter les techniques de gestion de projet

UBO

Les intervenants du projet

- Maître d'ouvrage (MOA) : le client
  - Définit les besoins
  - Déclenche le financement
  - Fait l'interface entre les utilisateurs et le MOE
  - Valide les livrables
  - Fait le suivi d'exploitation

Il faut que ça serve

- Maître d'œuvre (MOE) : le chef de projet
  - Coordonne la réalisation
  - Définit une référentiel qualité
  - Suit les actions et gère les risques
  - Qualifie les livrables

Il faut que ça avance

- Equipe projet
  - Réalise les travaux

Il faut que ça marche

jean-philippe.babau@univ-brest.fr

UBO

Les intervenants du projet

- Maître d'ouvrage (MOA) : le client
  - Respect des besoins utilisateurs
  - Respect des contraintes techniques

Il faut que ça serve

- Maître d'œuvre (MOE) : le chef de projet
  - Maitriser la qualité, les délais et les coûts

Il faut que ça avance

- Equipe projet
  - Résoudre les problèmes techniques

Il faut que ça marche

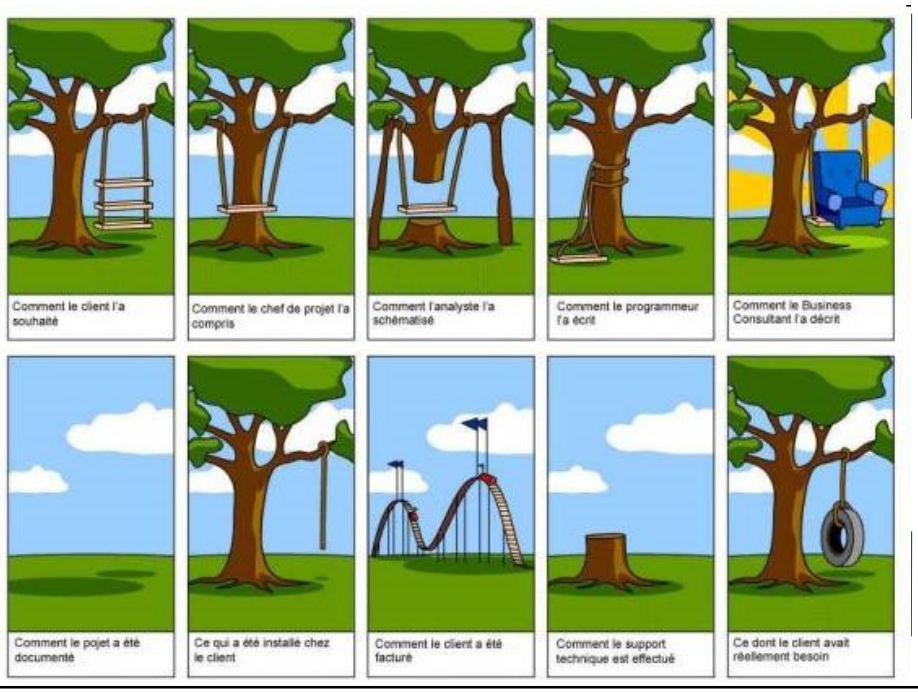
jean-philippe.babau@univ-brest.fr

UBO

Conduire un projet

- *S'appuyer sur des méthodes*
  - Spécifiant des processus de développement
    - Qui, Quand, Quoi, Comment
  - Organisant les activités du projet
  - Définissant les artefacts (résultats) du projet
  - Se basant sur des modèles clairs et précis

jean-philippe.babau@univ-brest.fr



UBO

Plan

- Introduction
  - Le projet
  - Objectifs d'une méthode
- Les méthodes de développement
  - Introduction
  - Le cycle en V
  - Compléments
- Gestion de projet

UBO

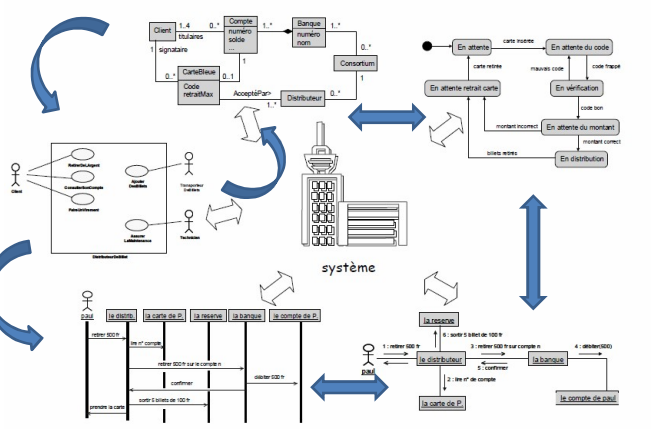
Rappels sur les modèles

- Besoin de modèles pour
  - Echanger des informations
  - Intégrer les concepts et les approches spécifiques métier au logiciel
  - Expliquer l'impact du traitement automatique de l'information au métier
  - Maitriser le développement
- Des diagrammes différents
  - 13 types de diagrammes UML
- Des utilisations diverses
  - Documentation
  - Conception
  - Programmation
  - Vérification
- Rechercher une approche intégrée

UBO

Intégration de modèles par les méthodes

- Définir les modèles, les principes de mise en œuvre et assurer un enchainement dans la construction des modèles



UBO

UML n'est pas une méthode

- UML est un langage
  - Connaitre UML (ou maitriser un AGL) n'est pas suffisant pour réaliser de bonnes conceptions

  « Maitriser l'orthographe (lexique) et la grammaire (syntaxe) de la langue française, et un éditeur de texte, ne suffisent pas pour être un écrivain de talent »

- Pour réussir un projet de développement informatique, il faut aussi
  - Maitriser les techniques de conception et de programmation (objet)
  - Avoir un certain nombre de qualités « projet »

- Et, suivre des méthodes de conception
  - Propositions de cheminements à suivre pour concevoir
  - Cheminements reproductibles et avérés

UBO

Méthode ou processus

- Une définition
  - guide ou démarche, plus ou moins formalisé, reproductible permettant d'obtenir des solutions fiables à un problème
  - capitalise l'expérience de projets antérieurs et les règles dans le domaine du projet
- Une méthode définit
  - Une chronologie des activités
  - Des concepts de modélisation
  - Un ensemble de règles et de conseils pour tous les participants au projet
- Décrire **qui** (des personnes) fait **quoi** (des artefacts) à quel moment (**quand**) et de quelle façon (**comment**) pour atteindre un certain **objectif**
  - Ensemble de bonnes pratiques issues de l'état de l'art, de l'expérience

jean-philippe.babau@univ-brest.fr

UBO

Les méthodes

- Grandes classes de méthodes
  - Méthodes pour l'organisation stratégique
  - **Méthodes de développement**
  - Méthodes de conduite de projet
  - Méthodes d'assurance et de contrôle qualité
- Méthodes de développement
  - Construire des systèmes opérationnels
  - Organiser le travail dans le projet
  - Gérer le cycle de vie complet
  - Gérer les risques
  - Obtenir de manière répétitive des produits de qualité constante

jean-philippe.babau@univ-brest.fr

UBO

Bilan : méthode, projet, génie logiciel

- Génie Logiciel
  - Appliquer de « bons » principes
    - Pour produire de « bons » artefacts
- Méthode : décrire (expliciter et formaliser) les activités
  - Qui fait l'activité et comment
  - Définir les productions, le quoi (artefacts)
  - Valider les artefacts produits (qualité)
  - Planification (quand)
- Gestion de projet
  - Assurer le suivi du projet (mise en œuvre de la méthode) pour la maitrise des activités
    - Et donc du coût, des délais et des moyens

UBO

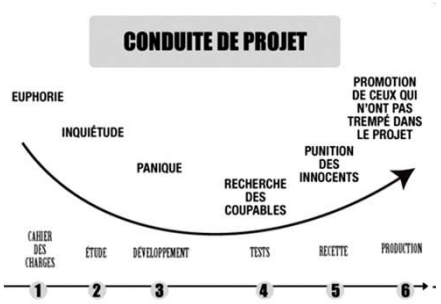
Plan

- Introduction
  - Le projet
  - Objectifs d'une méthode
- Les méthodes de développement
  - Introduction
  - Le cycle en V
  - Compléments
- Gestion de projet

UBO

Les phases d'un projet de développement

- Lancement / avant projet
  - Expression du besoin/cahier des charges (MOA)
  - Réponse à appel d'offres (MOE : DSI ou prestataires)
- Définition/Etude
  - Mise en place référentiel projet, outils, méthodes, équipe, planning
  - Spécifications détaillées
- Réalisation
  - Conception
  - Développement
  - Tests, intégration
  - Recette
- Capitalisation
  - Bilan, retour d'expérience



jean-philippe.babau@univ-brest.fr

UBO

Plan

- Introduction
  - Pourquoi une méthode ?
  - Objectifs d'une méthode
- Les méthodes de développement
  - Introduction
  - Le cycle en V
  - Compléments
- Gestion de projet

jean-philippe.babau@univ-brest.fr

UBO

Phases classiques du développement (1/4)

- Phases amont : compréhension et analyse des besoins

- (1) Expression des besoins
  - Comprendre le client : analyse métier (les données et les processus de l'entreprise)
  - Comprendre les besoins du client (quel logiciel pour le client « Que veut-on ? »)
- (2) Etude de faisabilité
  - Estimation de la faisabilité des besoins
  - Cela doit répondre aux questions « Est-ce réalisable ? » et « À quel coût ? ».
- (3) Analyse des besoins ou spécification fonctionnelle
  - Traduire les besoins du client en besoins pour l'informatique
  - C'est le cahier des charges du produit final, tel que le désire le client. Il doit couvrir l'intégralité des cas d'utilisation du produit, en expliquant ce qu'il *doit faire* et non pas *comment il va le faire*.

UBO

Phases classiques du développement (2/4)

- Phases de développement : réaliser le produit

- (4) Conception générale
  - Définition de l'architecture générale du logiciel
  - Choix techniques généraux
- (5) Conception détaillée
  - Découpage en modules élémentaires ou sous-ensembles du logiciel
  - Spécification des entrées/sorties de chaque module
  - Choix techniques pointus
- (6) Réalisation ou codage
  - Produire des modules codés
- (7) Intégration
  - Les modules codés sont assemblés

UBO

Phases classiques du développement (3/4)

- Phases d'évaluation : vérification
- (8) Tests unitaires
  - Tests des modules : le module correspond à sa spécification
- (9) Tests d'intégration
  - Tests d'intégration des modules : l'interfaçage des modules est correct, l'assemblage est correct
- (10) Validation
  - Vérifier la conformité vis-à-vis des spécifications fonctionnelles
- (11) Qualification ou recette
  - Vérification du respect du contrat initial avec le client
  - Dernière vérification avant mise en production

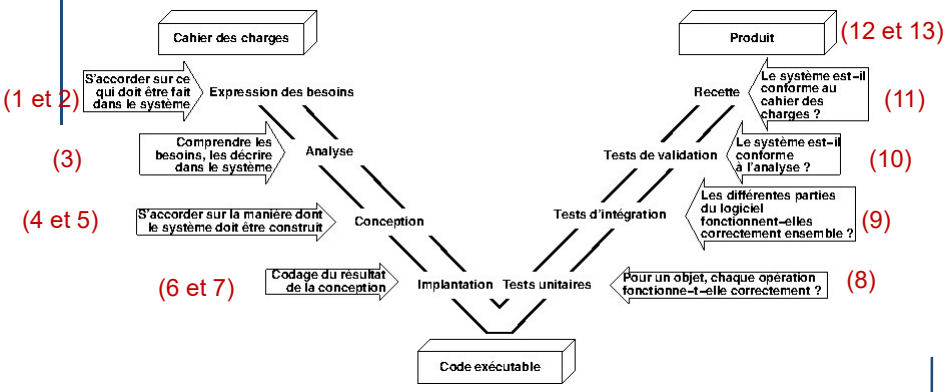
UBO

Phases classiques du développement (4/4)

- Phases de suivi : accompagnement
- (12) Mise en production
  - Déploiement du service
- (13) Maintenance
  - Correction des anomalies résiduelles (maintenance corrective) et des évolutions (maintenances évolutives)
- (14) Documentation
  - Produire les informations nécessaires pour l'utilisation du logiciel et pour les développements ultérieurs

UBO

Le cycle en V

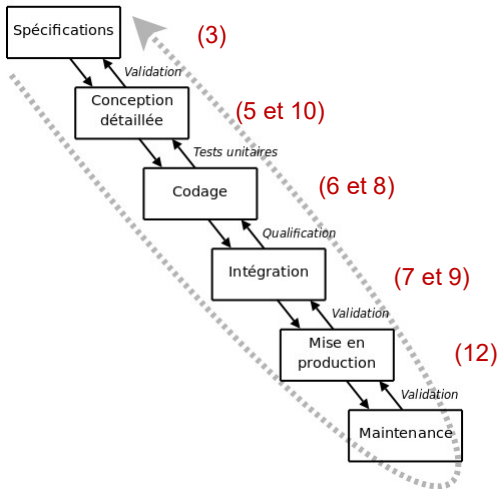


jean-philippe.babau@univ-brest.fr

25

UBO

Cycle en cascade



jean-philippe.babau@univ-brest.fr

26

UBO

Expression des besoins

- Permettre une meilleure compréhension du système
  - Comprendre le domaine
  - Aspects fonctionnels et non fonctionnels (sécurité, …)
- Comprendre et structurer les besoins du client
  - Clarifier, filtrer et organiser les besoins
  - Ne pas chercher l'exhaustivité
- Une fois identifiés et structurés, ces besoins :
  - Définissent le contour du système à modéliser
    - Précisent le but à atteindre
  - Permettent d'identifier les fonctionnalités principales (critiques) du système
- Définir le domaine : diagramme de classe et diagrammes d'activité
- Le besoin : diagramme de cas d'utilisation

jean-philippe.babau@univ-brest.fr

UBO

Spécifications fonctionnelles

- Objectif
  - Décrire le problème (ce que doit faire le système et comment il le fait tel que vu d'un point de vue métier) sans spécifier la solution technique
- Préciser les besoins
  - Traduire dans un langage qui se rapproche *doucement* de celui des informaticiens les modèles exprimés dans l'expression des besoins
  - Pour rester compréhensible par les clients ou utilisateurs, on ne prend en considération que des entités du domaine (métier)
- Elément de référence
  - Entre le clients et les concepteurs du logiciel
  - L'analyse doit servir de support pour la conception, l'implantation et la maintenance
- Définir les exigences et les contraintes
- Le besoin : diagramme de cas d'utilisation et diagrammes de séquence

jean-philippe.babau@univ-brest.fr

UBO

Conception

- Le modèle de la conception décrit la solution (comment le problème est résolu)
  - Modélisation architecturale
- Faire des choix
  - Le but de la conception est de fixer les choix techniques et de préparer l'implantation
- Modèle pivot pour les informaticiens
  - La conception doit servir de support pour l'implantation et la maintenance
  - Le modèle de la conception n'est pas destiné à être compréhensible par les utilisateurs mais par les développeurs
- Bonne structuration (Design Pattern) et bonnes pratiques (styles architecturaux)
- Conception: diagrammes de classe, de package et de composants, diagrammes de séquence

UBO

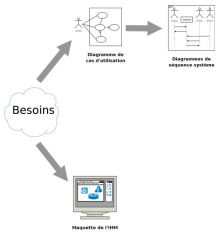
Qualités pour la conception

- Observer et expérimenter
  - Une conception est rarement pertinente au premier jet
  - Prendre du recul, se remettre en question
  - Il n'y a pas de recettes toutes faites
- Abstraire
  - Ne pas se plonger/s'enliser dans les détails technologiques
  - Ne pas se focaliser sur un détail
  - Savoir extraire un aspect du problème (via un modèle spécifique)
- Etre guidé par le résultat attendu
  - Le client doit être satisfait (enjeux financiers)
  - C'est le client qui décide de la validité du résultat

UBO

IHM

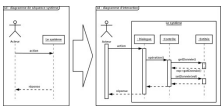
- Maquette de l'IHM de l'application (non couvert par UML)
- Indépendant de la conception
- Souvent nécessaire dès le début, parfois intégré dans la spécification

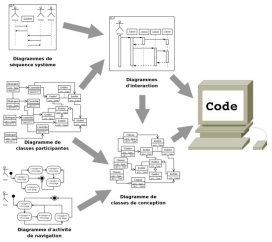


UBO

Conception détaillée

- Passage du monde « idéal » au monde « réel »
- Diagrammes de classe
  - Intégration des spécificités du langage d'implémentation
  - Intégration des spécificités des outils
  - Mise en œuvre des design-pattern
- Diagrammes d'interaction
  - Diagramme de séquence pour les objets
- Diagrammes de déploiement
  - Architecture matérielle et logique de l'application





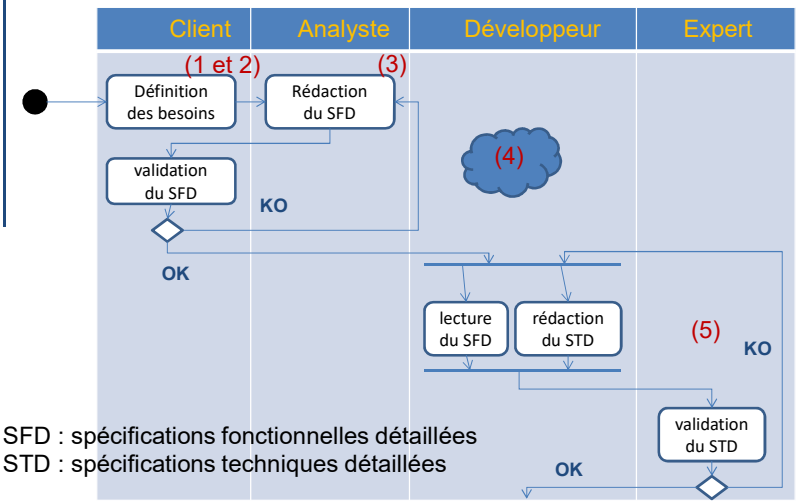
UBO

Limites du cycle en V

- Cycle idéal
  - dans la réalité, il est quasiment impossible d'obtenir des spécifications fonctionnelles complètes et qui ne seront pas modifiées par la suite
  - dans la réalité, la conception n'est pas idéale et lors de l'implémentation, on identifie des cas limites et des problèmes conceptuels
- Cycle pour un développement entier
  - Dans les projets, on traite souvent d'évolutions de logiciels existants
- Pas d'élément sur le comment (comment réaliser chaque étape)

UBO

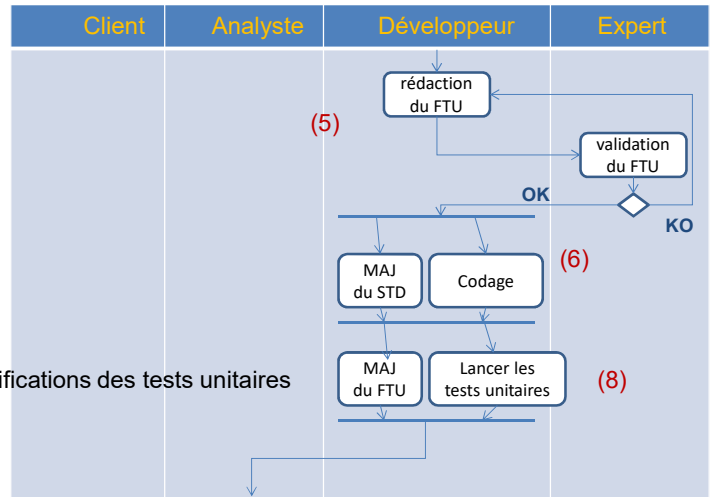
Un cycle en V adapté et détaillé



SFD : spécifications fonctionnelles détaillées
STD : spécifications techniques détaillées

UBO

Un cycle en V adapté et détaillé

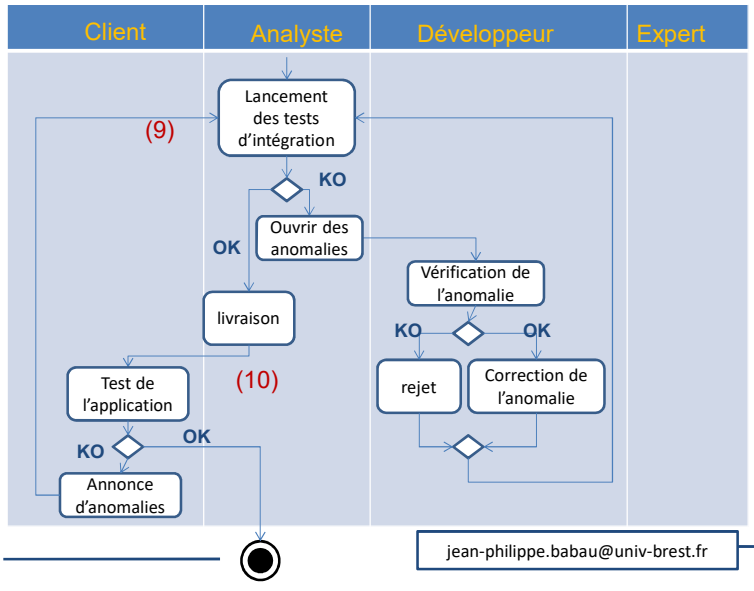


FTU: spécifications des tests unitaires

jean-philippe.babau@univ-brest.fr

UBO

Un cycle en V adapté et détaillé



jean-philippe.babau@univ-brest.fr

UBO

Mise en place d'un processus de développement

- Pour une même étape
  - Compréhension du problème
    - Mise en place de la solution
    - Vérification de la solution
  - Livraison de la solution
- Une étape n'est pas liée à une seule personne
- Une étape correspond à plusieurs tâches
- La reprise d'erreurs est possible tout au long du cycle de vie
- Un processus strict à respecter
  - Communication tout au long du projet
- Tout est formalisé dans des documents

UBO

Quelques chiffres

- Répartition des activités
  - Analyse et conception 45%
  - Réalisation et tests unitaires : 35%
    - Codage 15 à 20% du total
  - Intégration et validation : 25%
- Dérives dans les estimations
  - Étude préalable : de 10 à 25 %
  - Conception : de 10% à 35 %
  - Réalisation : de 30% à 40%
  - Mise en œuvre : de 5% à 20%

UBO

Plan

- Introduction
  - Le projet
  - Objectifs d'une méthode
- Les méthodes de développement
  - Introduction
  - Le cycle en V
  - Compléments
- Gestion de projet

UBO

Qualité de code

- CheckStyle
  - Proposé avec 126 règles de codage
  - Règles programmables
  - Utilisés par les entreprises pour assurer la qualité du code
  - Règles de présentation du code
  - Règles de nommage
  - Complexité du code
    - Nombre max de méthodes par classe
    - Longueur max d'une ligne de code
    - Longueurs max de lignes dans une méthode
    - Nombre maximal de chemins dans une méthode
  - Règles de programmation
    - Initialisation des données …
  - …
- checkstyle.sourceforge.net

UBO

La gestion des risques

- Risque = événement indésirable, incertain, pouvant mettre en danger le projet en impactant
  - La date de fin
  - Les coûts
  - Le respect des spécifications (techniques, qualité, performance, stabilité)
  - L'image d'entreprise, l'environnement, …
- Préoccupation essentielle de la gestion de projet
- Objectifs
  - Anticiper les risques
  - Réduire l'impact d'événements négatifs
  - Profiter des opportunités qui se présentent

UBO

Types de risques

- Différentes natures de risques
  - Besoins
    - Mauvaise interprétation des besoins, du domaine
  - Technique
    - Mauvaise maitrise des technologies
    - Architecture technique inadaptée (performances insuffisantes)
  - Développement
    - Impossibilité de fournir un résultat de qualité
    - Développement trop complexe
  - Autres
    - Outil inutilisable (formation, administration)
    - Risques non techniques (personnel de développement insuffisant, problèmes commerciaux ou financiers)
- Tout risque fatal pour le projet est à découvrir au plus tôt

UBO

Caractériser le risque

- **P**robabilité : Echelle de 1 à 4 : de *très peu probable* à *quasi-inévitable*
- Gravité/**I**mpact : Echelle de 1 à 4 : de *sans impact réel* à *vie humaine en danger*
- **D**étectabilité : Echelle de 1 à 4 : *détectable systématiquement* à *indétectable*
- Poids ou criticité = P * I (ou P * I * D)
  - Echelle de 1 à 16 (ou de 1 à 64)
  - Entre 9 et 12 : criticité moyenne, impactant les objectifs du projet, sans le remettre en cause dans son ensemble
  - Entre 13 et 16 : conséquences pouvant interrompre le projet

UBO

Réduction des risques

- Supprimer la cause : ne pas utiliser la technologie X non maîtrisée
- Modifier le projet : abandon d'une fonction
- Partage/Transfert de responsabilité : sous-traiter certains travaux à un spécialiste
- Limiter les conséquences : prévoir un système de secours, affecter une ressource spécifique…
- Surveiller le risque : méthode de détection (augmenter la fréquence des tests), mise en place d'indicateurs spécifiques

UBO

Gestion de risques et pilotage de projet

- Suivre les risques principaux (criticité ≥ 9)
- Etablir un tableau des risques et réévaluer régulièrement
- Identifier de nouveaux risques
- Capitaliser les retours d'expérience
- Evaluer risques résiduels en fin de projet

jean-philippe.babau@univ-brest.fr

UBO

Approches itératives et incrémentales

- Gestion de la complexité
  - Pas « tout en même temps »
  - Etalement 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)
- Prise en compte des modifications de besoins
  - Feedback, implication des utilisateurs et adaptation précoce
- Apprentissage rapide et adaptation de la méthode
  - Amélioration de la productivité et de la qualité du logiciel
  - 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

jean-philippe.babau@univ-brest.fr

UBO

Développement UP

- Un cycle de vie produit une version
- Un cycle de vie = 4 phases
  - Lancement
  - Elaboration
  - Construction (50% du cycle de vie : développement principal et mise au point)
  - Transition
- Une phase se découpe en itérations successives
  - Approche incrémentale
- Une phase et une itération se termine par un jalon

UBO

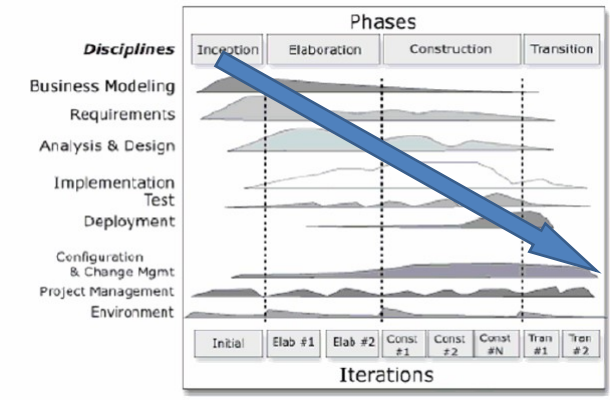
Un jalon

- Un jalon marque la fin de la phase
  - La phase est finie
  - Restitution des résultats obtenus lors de la phase
  - Prise de décision
    - Est-ce qu'on continue le projet ?
    - Est-ce qu'on abandonne le projet ?
    - Est-ce que l'on recommence la phase ? ⟸ Echec de la phase

UBO

La méthode UP : phases et activités

- 4 phases, plusieurs itération par phase
- Dans chaque phase /itération : toutes les activités de développement



UBO

Les phases UP

- Lancement : étude prospective
  - Quel est le domaine d'application?
  - Que doit faire le système ?
  - Quels sont les risques ?
  - Quel sont les coûts, les délais, les ressources et les moyens pour le projet ?
  - Comment le planifier ?
  - Jalon : « objectifs du projet »
  - Prise de décision: Est-ce qu'on accepte le projet ?
- Elaboration : mise en place d'un prototype
  - Spécification de la plupart des cas d'utilisation
  - Conception de l'architecture de référence (squelette du système)
  - Mise en œuvre de cette architecture (CU critiques, <10 % des besoins)
  - Planification complète
  - Les besoins sont-ils considérés? L'architecture stable? Les risques sont contrôlés ?
  - Jalon : « architecture»
  - Prise de décision: Est-ce qu'on peut passer à la réalisation?

UBO

Les phases UP

- Construction : développement
  - Développement par itérations / incréments
    - Pour aboutir à une architecture stable
  - Le produit contient tout ce qui avait été planifié
    - Il peut rester quelques erreurs non détectées …
  - Jalon : «opérationnel»
  - Prise de décision: le produit est-il suffisamment correct pour être installé chez un client ?
- Transition : mise en place et suivi
  - Produit livré (version béta)
  - Correction du reliquat d'erreurs
  - Essais et améliorations du produit
  - Formation des utilisateurs
  - Installation de l'assistance en ligne
  - Le produit est-il satisfaisant ? Les manuels sont-ils prêts ?
  - Jalon : « release du produit »
  - Prise de décision: le projet est fini ?

jean-philippe.babau@univ-brest.fr

UBO

Jalon de lancement « objectifs du projet »

- Interlocuteurs présents
  - Client : resp. projet, resp. qualité, resp. achats
  - Prestataire : chef de projet, resp. qualité, chargé d'affaires
- Objectifs du jalon
  - Rappeler les objectifs et le contexte du projet
  - Présenter l'équipe en charge de la réalisation
  - Lister
    - les données d'entrée, les éléments manquants
    - les éléments à produire pendant le projet
  - Préciser le planning/les jalons mis à jour selon la date réelle T0
  - Identifier les premières actions à mener par le client et le prestataire
  - Lister et évaluer les premiers risques

jean-philippe.babau@univ-brest.fr

UBO

Livrables de la phase de lancement

- Première version du modèle du domaine ou de contexte de l'entreprise
- Liste des besoins fonctionnels et non fonctionnels
- Ébauche des modèles de cas, d'analyse et de conception
- Esquisse d'une architecture
- Liste ordonnée de risques et liste ordonnée de cas
- Grandes lignes d'un planning pour un projet complet
- Première évaluation du projet, estimation grossière des coûts
- Glossaire

UBO

Livrables de la phase d 'élaboration

- Un modèle de l'entreprise (processus, glossaire) complet
- Les cas d'utilisation et l'expression des besoins (80%)
- La description des architectures
  - Technique, composants, classes
  - Une version des modèles
- Conception (<10%), implémentation (<10%) et déploiement
  - Une architecture de base exécutable
  - Un manuel utilisateur préliminaire
- Une liste des risques mise à jour
- Un projet de planning pour les phases suivantes
  - Itérations
- Évaluation du coût du projet

UBO

Livrables de la phase de construction

- Un plan du projet pour la phase de transition
- L'exécutable
- Tous les documents et les modèles du système
- Une description à jour de l'architecture
- Un manuel utilisateur suffisamment détaillé pour les tests

UBO

Livrables de la phase de transition

- L'exécutable et son programme d'installation
  - Selon les cibles
- Les documents légaux
  - Contrat, licences, garanties
- Les documents de développement
- Les manuels utilisateur, administrateur et opérateur
- Le matériel de formation
  - Cours, exercices
- Les références pour le support utilisateur
  - Suivi de bug, assistance
  - Site Web

UBO

Plan

- Introduction
  - Pourquoi une méthode ?
  - Objectifs d'une méthode
- Les méthodes de développement
  - Introduction
  - Le cycle en V
  - Compléments
- Gestion de projet

UBO

Piloter un projet

- Comment piloter un projet logiciel ?
  - Prise de décision : chef de projet
    - Rôle : suivre et guider le projet
      - Attribuer (arrêter) une activité, préparer les évaluations, suivre le planning
      - Le droit à l'erreur
    - Ne décide pas de tout, ne fait pas le travail à la place de
      - Mais décide de l'organisation du projet
      - Peut revenir en arrière
      - Développer la notion de responsabilité
  - Suivi du projet :analyse, *reporting*, synthèse
    - Le *reporting* concerne un type de rapports qui répond aux questions
      - « Que s'est-il passé ? » ou « Que se passe-t-il en ce moment ? »
      - Différent d'un tableau de bord

UBO

Organiser le projet en lots

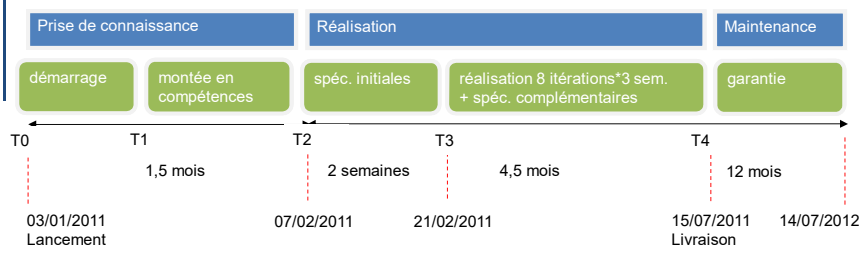
- Les informations caractérisant chaque lot
  - Les objectifs : ce qui doit être réalisé lorsque le lot est terminé
  - Les ou les responsables
  - Les livrables : matériels, logiciels, rapports, études, données…
  - Les données d'entrée : informations, documents, données nécessaires à la réalisation du lot
  - Les ressources : moyens matériels/humains, outils, fournitures…
  - La durée de réalisation tenant compte des disponibilités et capacités des ressources
  - Le budget prévisionnel fonction des ressources allouées et de la durée de travail
  - La performance : indicateurs de performance et/ou de respect des normes et standards pour assurer le suivi de la réalisation

jean-philippe.babau@univ-brest.fr

UBO

Planifier la réalisation des lots

- Structurer le projet
  - Planning macro – Jalons principaux



  - Pour chaque lot/composant du projet, identifier la phase
  - Documentation
  - Traitement des anomalies

jean-philippe.babau@univ-brest.fr

UBO

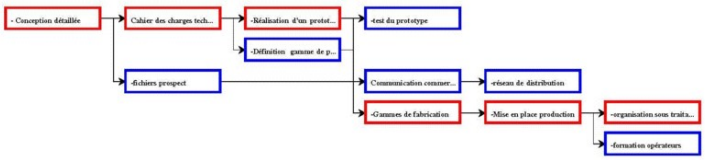
Associer des tâches aux lots

- Définition des tâches, pour chaque lot

  Tâche = opération élémentaire indispensable à la réalisation du lot

  – Identifier des tâches unitaires, courtes (charge = 10 jours max.)
  – Prévoir l'enchaînement des tâches : contrainte d'antériorité
  – Identifier le chemin critique

  ➔ Graphe PERT (Program Evaluation and Review Technique)

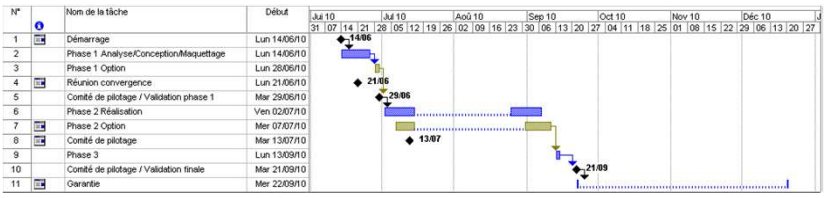


UBO

Planifier les tâches

- Planification des tâches et des ressources
  – Le PERT ne prend pas en compte la disponibilité des ressources (moyens humains et techniques)
  – Pour chaque tâche
    - Préciser la charge prévue
    - Allouer une/plusieurs ressources (en tenant compte des disponibilités)
    - Planifier la tâche, en tenant des contraintes d'antériorité du PERT

  ➔ Planning de référence du projet : diagramme de GANTT



UBO

Adapter le planning

- Adapter le planning selon les ressources disponibles
  - Eviter les surcharges (heures sup.) à un moment et les sous-charge à d'autres moments
  - Deux méthodes
    - Lissage : supprimer les surcharges par ajouts de ressources ponctuelles
    - Nivellement : répartir l'utilisation des ressources tout au long du projet en décalant des tâches selon la marge disponible
  - Une contrainte : respecter la date prévue de fin de projet

jean-philippe.babau@univ-brest.fr

UBO

Evaluation de la charge

- Premier modèle d'estimation COCOMO
  - KLS = 1000 lignes de code source
    - Par exemple 2 000 lignes 2 KLS
  - Effort global en homme-mois
    - pour un programme de complexité simple (2) $2.4 * KLS^{1.05}$ 5 h/mois
    - pour un programme de complexité moyenne (32) $3 * KLS^{1.12}$ 145 h-mois
    - pour un programme de complexité forte (2) $3.6 * KLS^{1.2}$ 8.27 h/mois
  - Temps de développement en mois
    - pour un programme de complexité simple (2) $2.5 * effort^{0.38}$ 4.6 mois
    - pour un programme de complexité moyenne (32) $2.5 * effort^{0.35}$ 14.27 mois
    - pour un programme de complexité forte (2) $2.5 * effort^{0.32}$ 4.91 mois
  - Moyens en personnel (effort/temps)
    - pour un programme de complexité simple (2) 1.1 personne
    - pour un programme de complexité moyenne (32) 10.17 personnes
    - pour un programme de complexité forte (2) 1.68 personne

jean-philippe.babau@univ-brest.fr

UBO

Evaluation de la charge

- Premier modèle d'estimation *COCOMO*
  - Productivité en ligne de code
    - pour un programme de complexité simple (2) 400 lignes/mois
    - pour un programme de complexité moyenne (32) 220 lignes/mois
    - pour un programme de complexité forte (2) 241 lignes/mois
    - Entre 10 et 20 lignes de code utiles par jour
  - Répartition (complexité moyenne et 32 KLS)
    - *Expression des besoins et planification* *7%*
    - *Conception générale* *17%*
    - *Conception détaillée* *25%*
    - *Programmation et tests unitaires* *33%*
    - *Tests et intégration* *25%*

UBO

Evaluation du coût

- Coûts en matériel
  - PC, …
- Coûts en logiciel
  - Outils de développement
  - Licence
- Couts
  - Salaires
    - cout total employeur : salaire net + charges (salariales et patronales)
    - Environ 2.25 fois le salaire net
    - Brut chargé pour un ingénieur expert (ANR) 8550 €/mois
    - Brut chargé pour un ingénieur junior (ANR) de 4500 à 6450 €/mois
  - Couts de structure (ANR : 80% du brut chargé)

UBO

Agilité dans le développement

- Méthodes Agiles
  - Le manifeste Agile 2001
  - Pratique de pilotage et de réalisation
  - Des frameworks (XP, Scrum)
- Quatre valeurs fondamentales
  - Les individus et leurs **interactions**
  - Des logiciels **opérationnels**
  - La collaboration avec le **client**
  - **L'adaptation** au changement

UBO

Agilité dans le développement

- Douze principes généraux
  - Satisfaire le client en priorité
  - Accueillir favorablement les demandes de changement
  - Livrer le plus souvent possible des versions opérationnelles de l'application
  - Assurer une coopération permanente entre le client et l'équipe projet
  - Construire des projets autour d'individus motivés
  - Privilégier la conversation en face à face
  - Mesurer l'avancement du projet en termes de fonctionnalités de l'application
  - Faire avancer le projet à un rythme soutenable et constant
  - Porter une attention continue à l'excellence technique et à la conception
  - Faire simple
  - Responsabiliser les équipes
  - Ajuster à intervalles réguliers son comportement et ses processus pour être plus efficace

UBO

Méthode Scrum

- Roles défini pour chacun
  - Le product owner : celui qui donne le point de vue du client
  - Le scrum master : le leader, l'animateur
  - Les développeurs (3 à 9)
  - Pas de chef de projet
  - Équipe auto-organisée, partage des tâches

- Réunions courtes
- Daily meeting
  - 15 minutes maximum en début de journée
  - Ce qui a été fait, ce qui va être fait, les obstacles

- Toute l'équipe dans la même pièce
- Utilisation de visuels pour suivre l'avancement du travail
  - A faire, en-cours, fait, validé
  - Post-it

UBO

Méthode Scrum

- Carnet de produit
  - Liste priorisée des besoins et fonctions à développer
  - Chaque tache : une description, une estimation de l'effort

- User stories
  - Un scénario utilisateur pour illustrer une fonctionnalité
  - Une procédure de test

- Tests
  - Tests unitaires
    - Rédigés avant le codage
    - Tests automatisés
  - Tests fonctionnels
    - Tests couvrant les fonctionnalités mises en œuvre
    - Mise à jour d'un référentiel pour assurer la non-régression
  - Test d'IHM
    - Ergonomie et performance

UBO

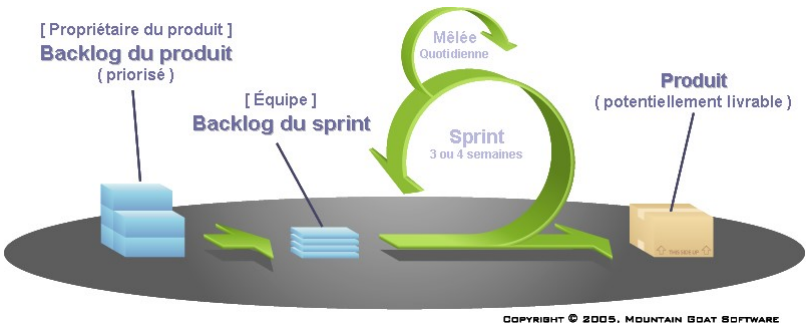
Méthode Scrum

- Sprint
  - De l'ordre de quelques semaines (2 ou3)
    - Développement, tests, documentation
  - Sprint planning
    - ½ à 1 journée
    - Listes des besoins fonctionnels à considérer
  - Revue
    - Quelques heures à ½ journée
    - Blian et démonstration
  - Livraison
    - Toutes les demandes validées par le product Owner et le client sont livrées
    - Manuel d'installation, d'exploitation, doc technique
  - Rétrospective
    - 2 à 3 heures
    - Chaque membre de l' équipe exprime son avis via des post-it
    - Les points positifs, des questions ou des remarques, les points négatifs
    - Echanger avec l'équipe pour mettre en avant les points positifs et éliminer les points négatifs

jean-philippe.babau@univ-brest.fr

UBO

Soyez agiles, soyez vigilants



jean-philippe.babau@univ-brest.fr

UBO

Gestion de problèmes

- Un document d'entrée n'est pas clair
  - Le document est incomplet
  - Le document est ambigüe
- Une tâche n'avance pas
  - Une difficulté non prévue survient
  - On ne trouve pas de solution simple
  - la personne est malade, …

REAGIR AU PLUS TOT : bien avant la fin de la tâche

- Consulter celui qui a produit les entrées
  - pour préciser un point
- Consulter un expert
  - Technique pour solutionner un problème
  - Du domaine pour préciser un point

- Toujours aviser le chef de projet qui pourra réagir
  - En affectant de nouvelles ressources
  - En adaptant son planning
  - En modifiant les spécifications fonctionnelle
  - En positionnant une alerte

jean-philippe.babau@univ-brest.fr

UBO

Quelques règles

- En début de projet : bien poser le problème
  - que doit-on faire ? l'objectif du projet (sans ambiguïté)
  - comment ? les moyens (budget/ressources)
  - avec qui ? MOA, MOE, équipe projet et rôles
- Ne pas être dans « l'à peu près »
  - périmètre du projet bien défini et figé
  - plannings réalistes
  - maîtriser la complexité
    - projet découpé en entités et en tâches claires
    - points de contrôle (jalons tests, qualification, documentation)
  - équipe bien adaptée, chef de projet impliqué
  - facteurs relationnels : communiquer efficacement, acteurs formés
  - anticiper les besoins et les difficultés
  - coûts suivis et maîtrisés
  - outils de gestion appropriés

jean-philippe.babau@univ-brest.fr

UBO

Qualités relationnelles pour l 'équipe projet

- Etre tourné vers le monde extérieur
  - Et non vers le (son ?!) code
- Dialoguer et communiquer avec les gens
  - Qui utiliseront le système
    - Savoir communiquer avec le(s) client(s) et/ou utilisateur(s)
  - Qui font le système
    - Savoir communiquer en interne
  - Faire des modèles clairs, précis, exhaustifs
    - À destination du lecteur
- Travailler à plusieurs
  - Un projet n'est jamais réalisé tout seul dans son coin
  - Accepter les conseils et propositions des autres et ne pas s'imposer

UBO

Conclusion

- Une méthode
  - Des règles pragmatiques
    - l'organisation doit être au service de la production
  - Des règles communes
    - d'organisation, de présentation
  - Des règles éprouvées et reconnues
    - expérience, standards
  - Des adaptations selon le contexte
    - pas de rigidité
- Expliciter et formaliser toutes les activités liées au développement
  - Développer n'est pas simplement coder
  - Pas d'activités « invisibles »
    - Associée à un exécutant, avec des objectifs, des artefacts
  - Pas d'activité « isolée »
    - Liens avec les autres activités au sein d'un processus global
  - Une activité est intégrée dans un processus

UBO

Conclusion

- Qualités humaines
  - Être tourné vers le client
    - Comprendre le client (métier et domaine)
    - Comprendre les besoins du client (besoins)
    - Comprendre les capacités d'intégration de suivi du client (déploiement)
  - Savoir travailler en équipe
    - Savoir communiquer
      - Utiliser des modèles
      - Respecter des conventions
    - Produire des documents réutilisables
- Qualités projet
  - Respecter la planification
  - Respecter les délais
  - Être guidé par des objectifs
  - Faire valider chaque production
  - Réagir au plus vite en cas de problème

UBO

Bibliographie

- http://www-inf.int-edu.eu/COURS/CSC4002/EnLigne/Cours/CoursUML/3.7.html
- Cours de Yannick Prié
  - http://liris.cnrs.fr/~yprie/
- Rational Unified Process Best Practices for Software Development Teams
  - IBM Rational Unified Process Web Site