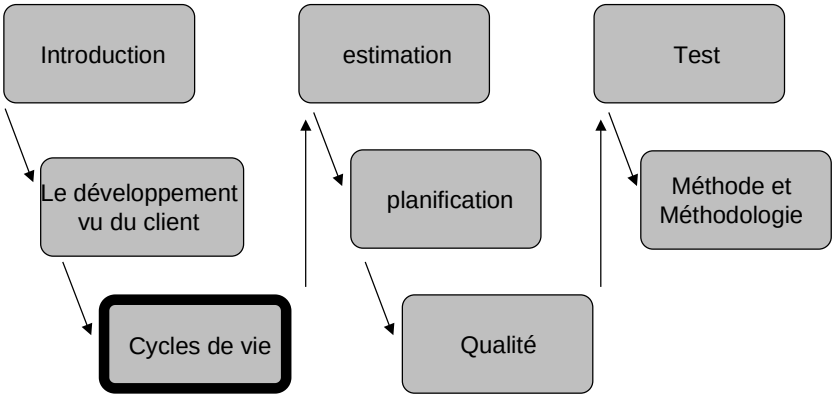
Introduction au génie logiciel # 2

plan



Cycles de vie

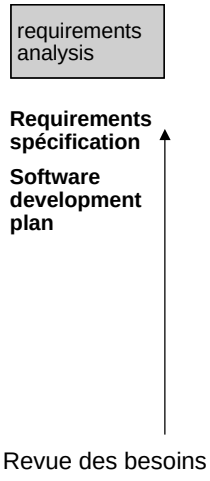
- **Modèles de cycle de vie**
- Principes généraux
- Les phases

Cycles de vie

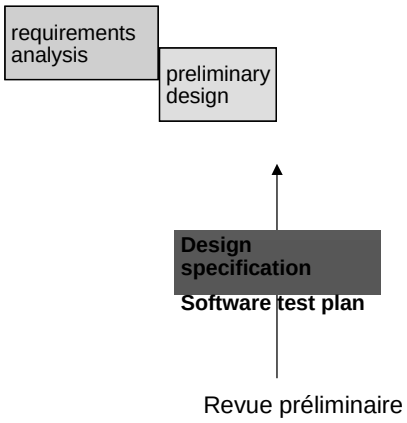
- Les cycles de vie servent à modéliser le déroulement dans le temps d'un processus complexe.
- Permettent de définir le vocabulaire employé
- Permettent de planifier des activités

- Quelques exemples :
  - Waterfall
  - V
  - Spirale

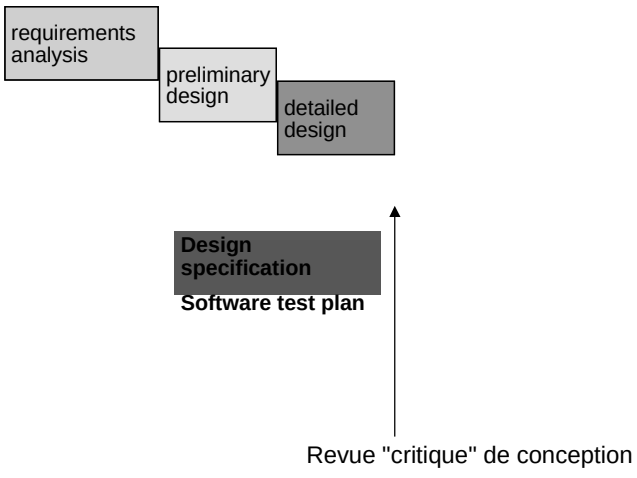
Cycle de vie standard



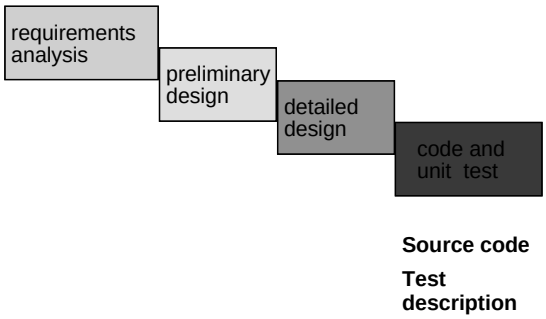
Cycle de vie standard



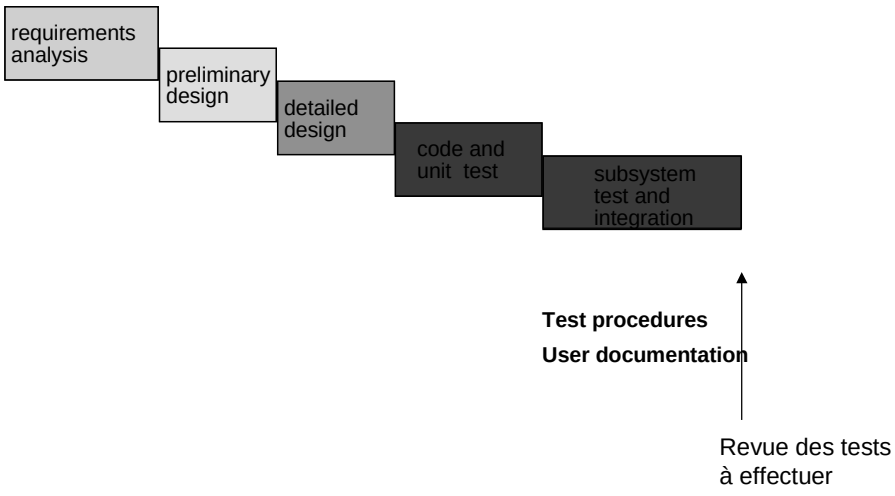
Cycle de vie standard



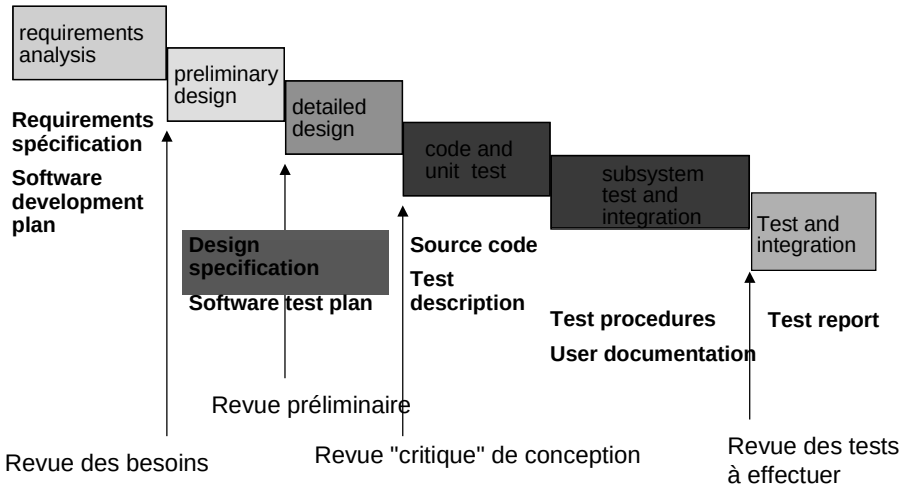
Cycle de vie standard



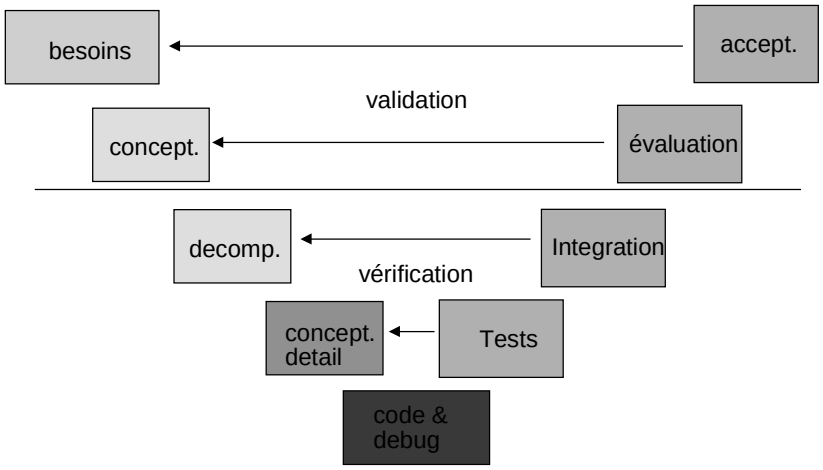
Cycle de vie standard



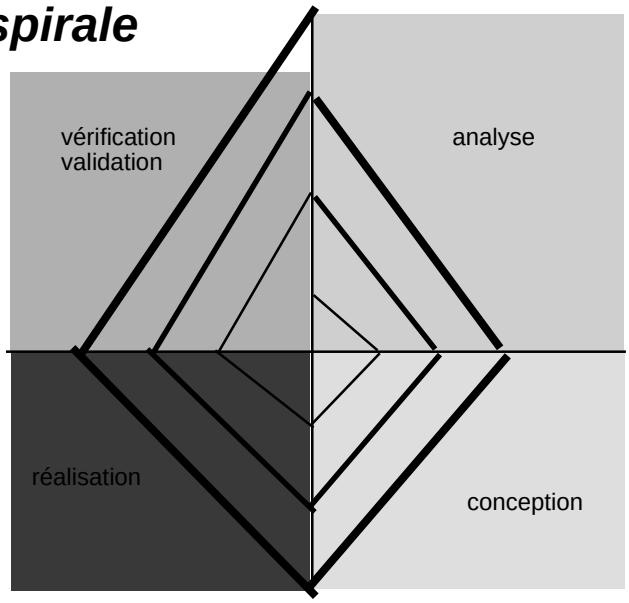
Cycle de vie standard



Le cycle en V

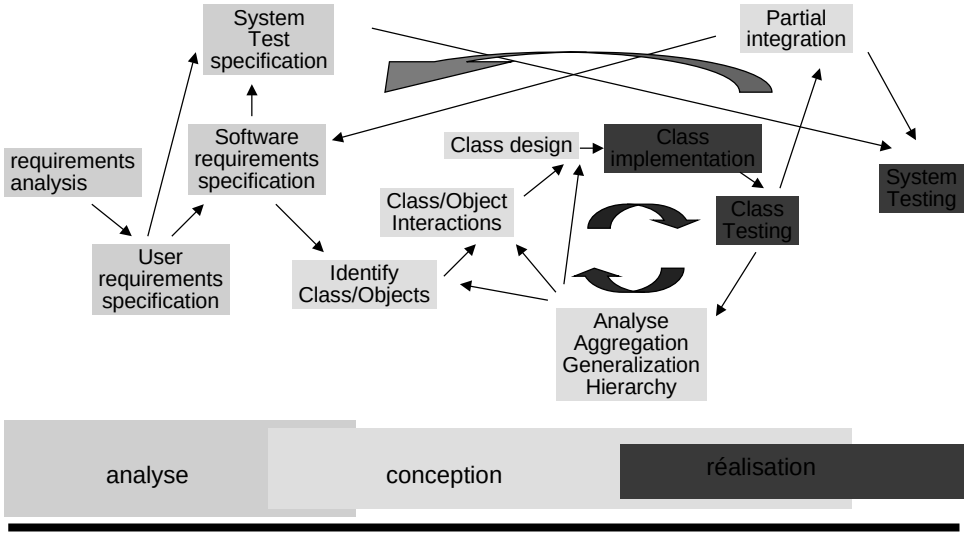


Cycle spirale



Succession de prototypes.
Approche objet.

Cycle objet (exemple)



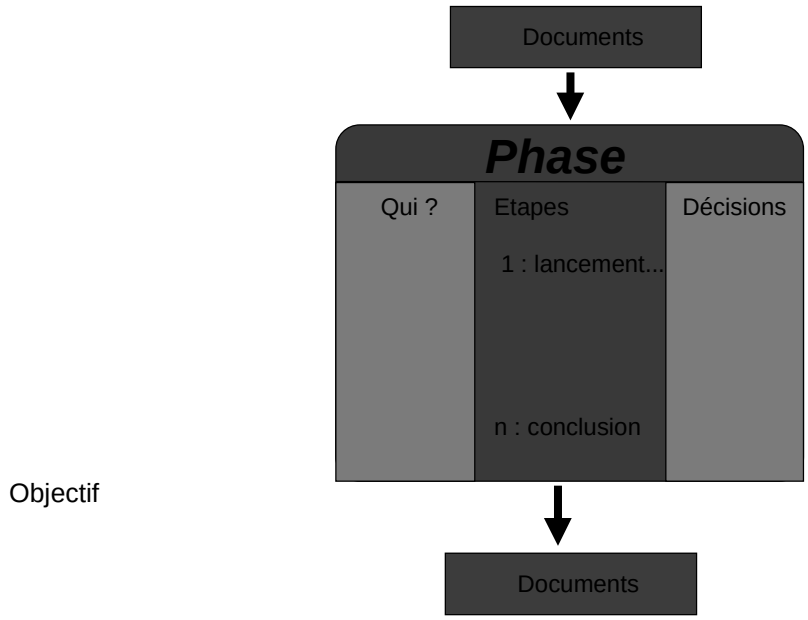
Cycles de vie

- Modèles de cycle de vie
- **Principes généraux**
- Les phases

Limite des cycles de vie

- La réalité suit rarement le modèle…retours arrière
- Il peut y avoir plusieurs flux en parallèle
- trop rigide

Mais offre des guides...



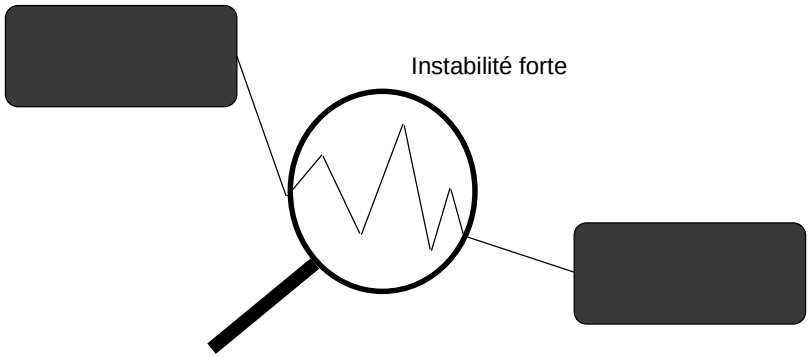
Rôles

La même personne peut jouer plusieurs rôles

- Chef de projet
- Chef de produits
- Architecte
- Designer
- Documentation
- Programmeur
- Intégrateur
- Rédacteur des tests
- Testeur
- Ingénieur Système
- Administrateur Système
- Assurance Qualité
- Experts
- Supports Outils
- Bêta Administration
- Sites Bêta
- Responsable des ventes
- Comptable

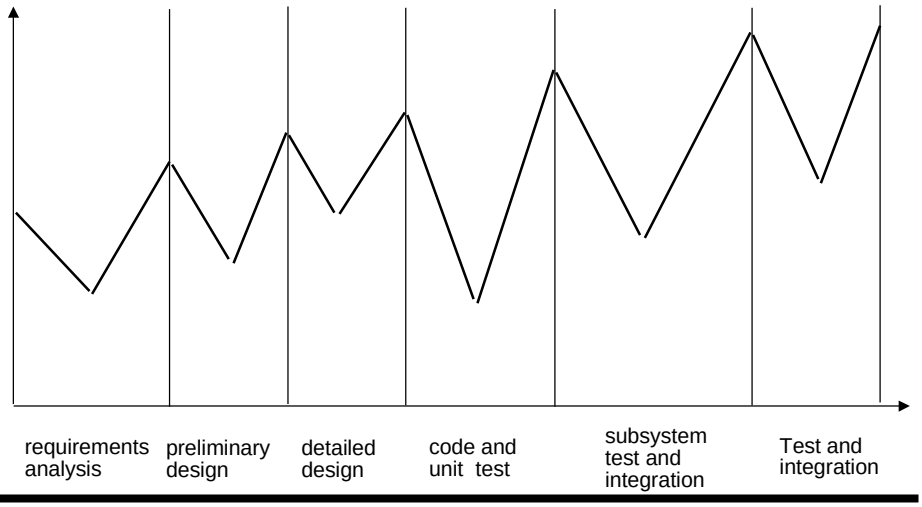
Certains rôles peuvent être extérieurs à l 'organisation, mais des liens de communication sont indispensables

Changement de phase



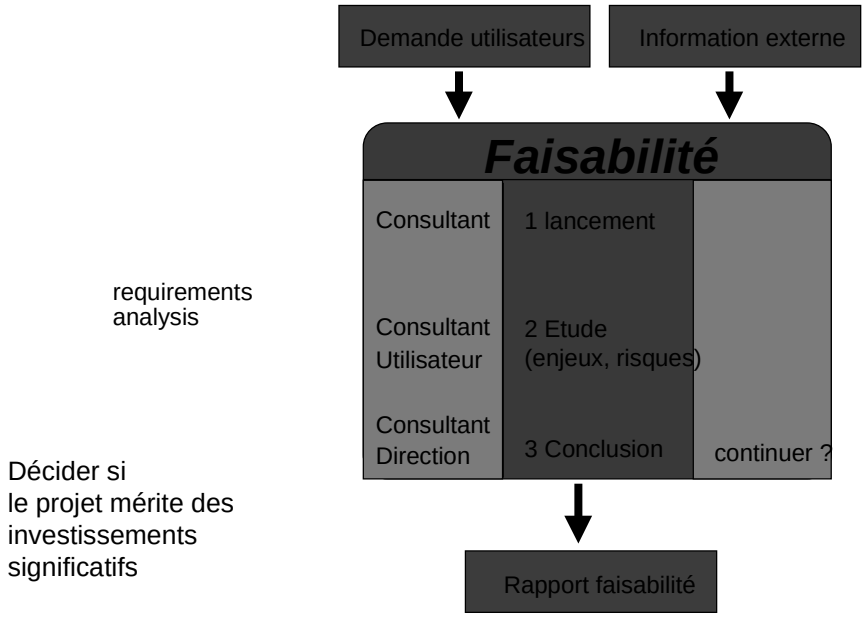
Communication forte...

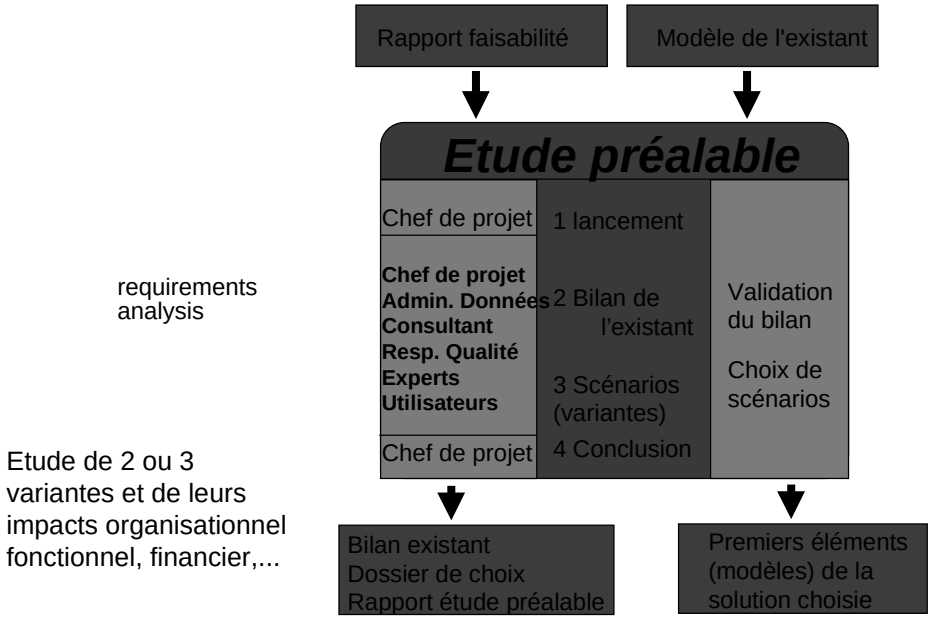
Communication



Cycles de vie

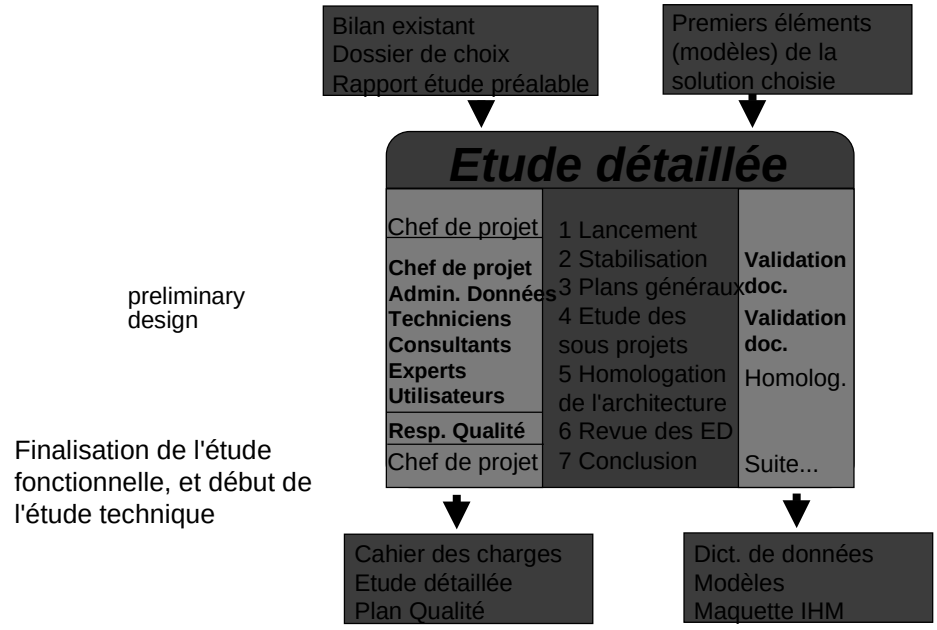
- Modèles de cycle de vie
- Principes généraux
- **Les phases**





Comprendre le domaine

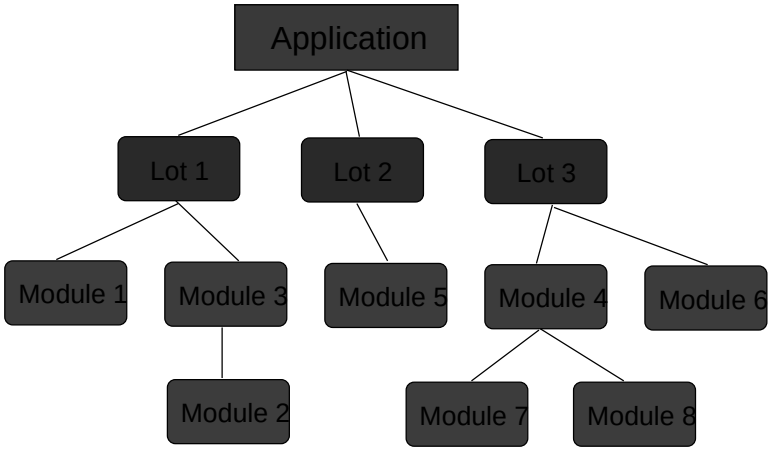
- Etudier le contexte dans lequel va s'insérer le système
- Qu'est-ce qui compose l'environnement du système ?
  - acteurs
  - objets
  - flux (d'information, de messages, de matériel, etc)
- Mickael Jackson : "le modèle du monde réel est plus stable que le modèle su système !"
- Quels besoins fonctionnels...
- Quelle qualité...



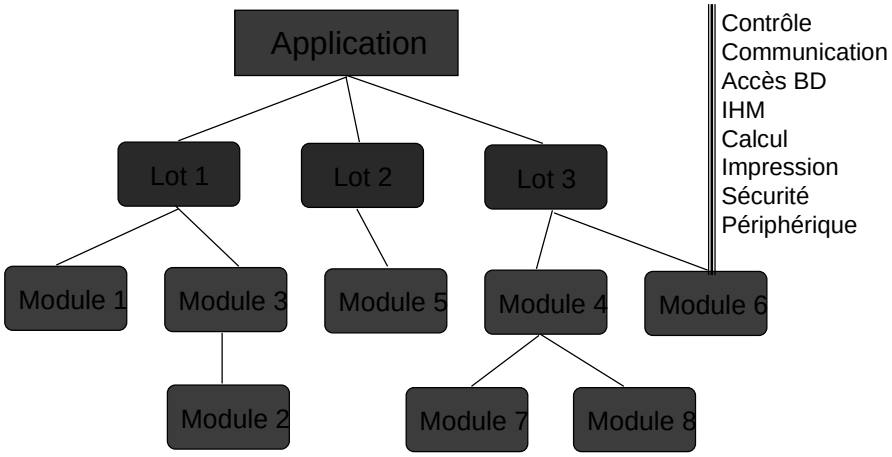
Délimiter le système

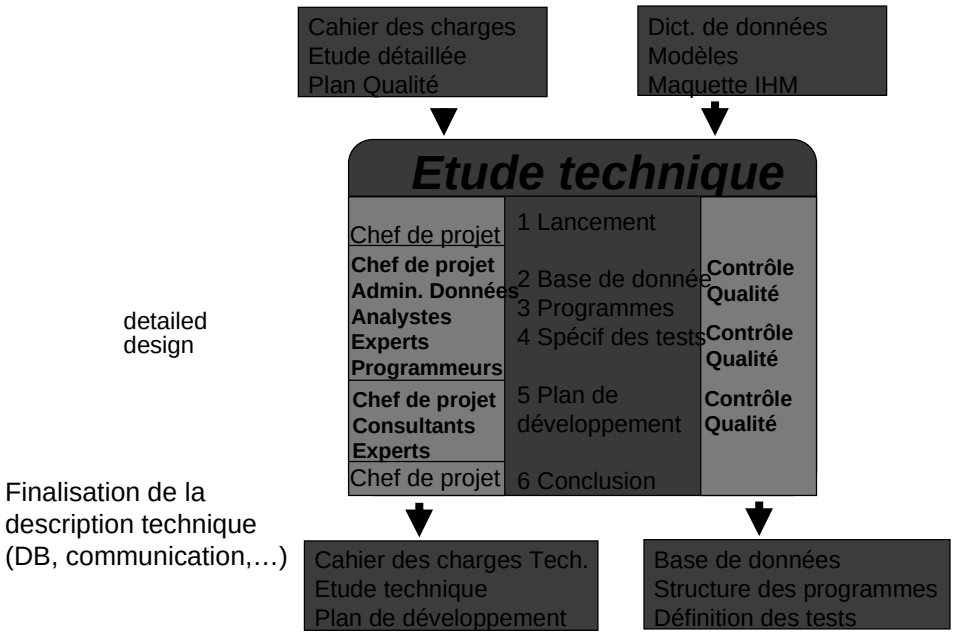
- Le contexte étant bien compris, on se concentre sur la réalisation du système
- On étudie les différentes solutions techniques (architecture des logiciels - décomposition)
- On évalue par rapport à la qualité et aux besoins requis

Découpage



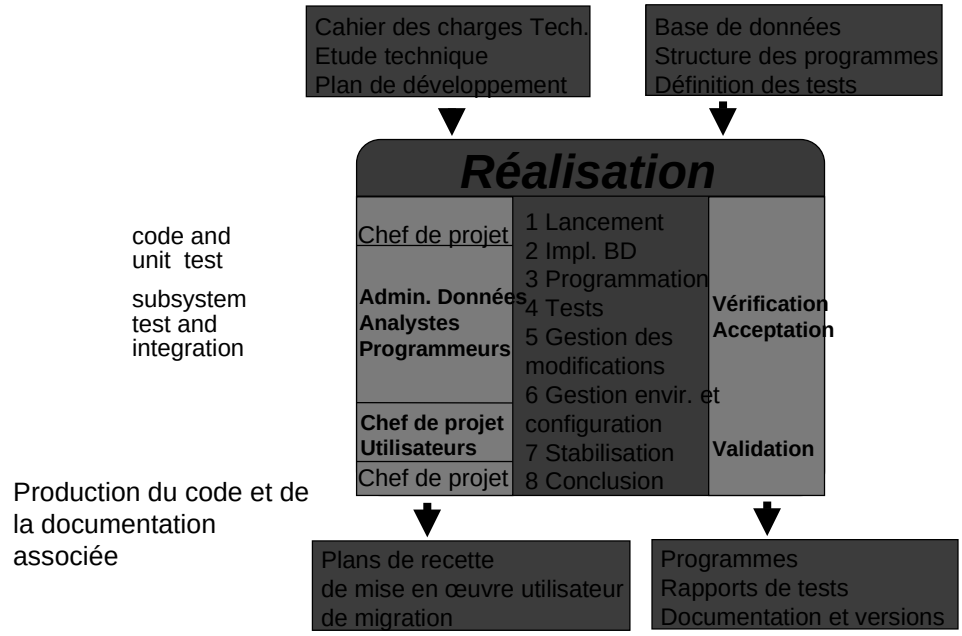
Découpage



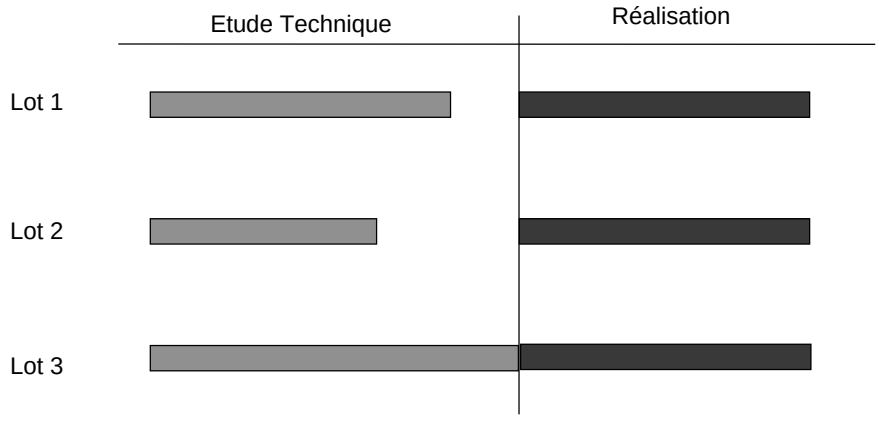


Détail de la solution retenue

- Une solution technique a été retenue
- Elle est détaillée ; on décrit précisément le contenu de chaque module/lot
- On devrait rester indépendant du langage et du système, mais certaines contraintes ou caractéristiques peuvent influer sur la conception (objet ou non, base de donnée ou non, etc.)

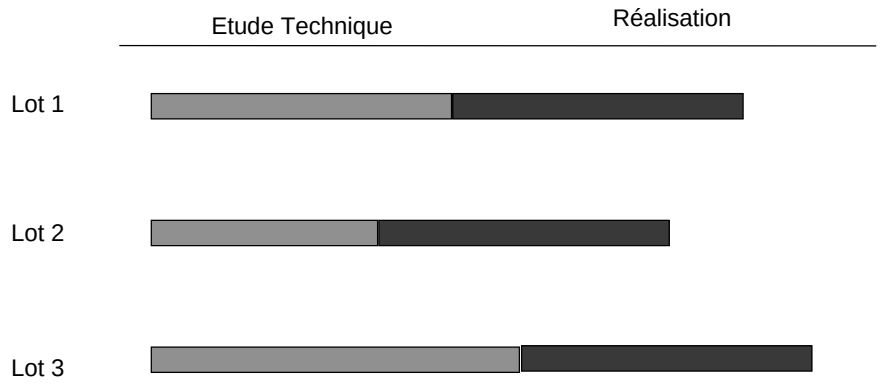


Stratégie au plus tard



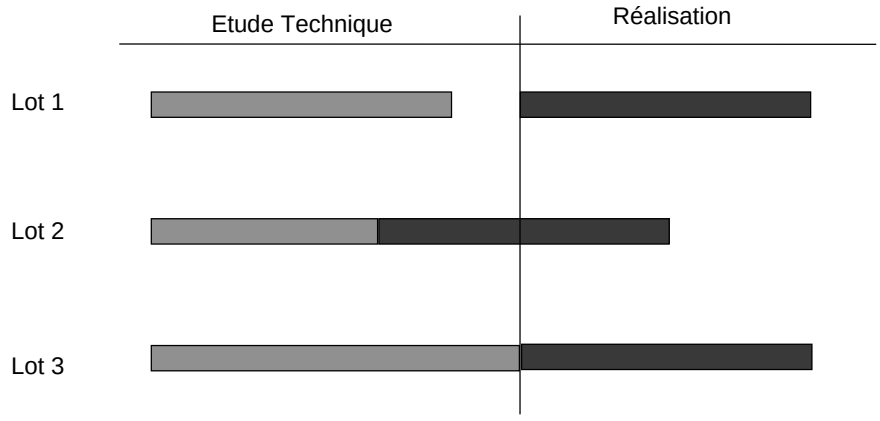
plus de cohérence et de prudence, mais moins d'anticipation des aléas et plus d'inactivité

Stratégie au plus tôt

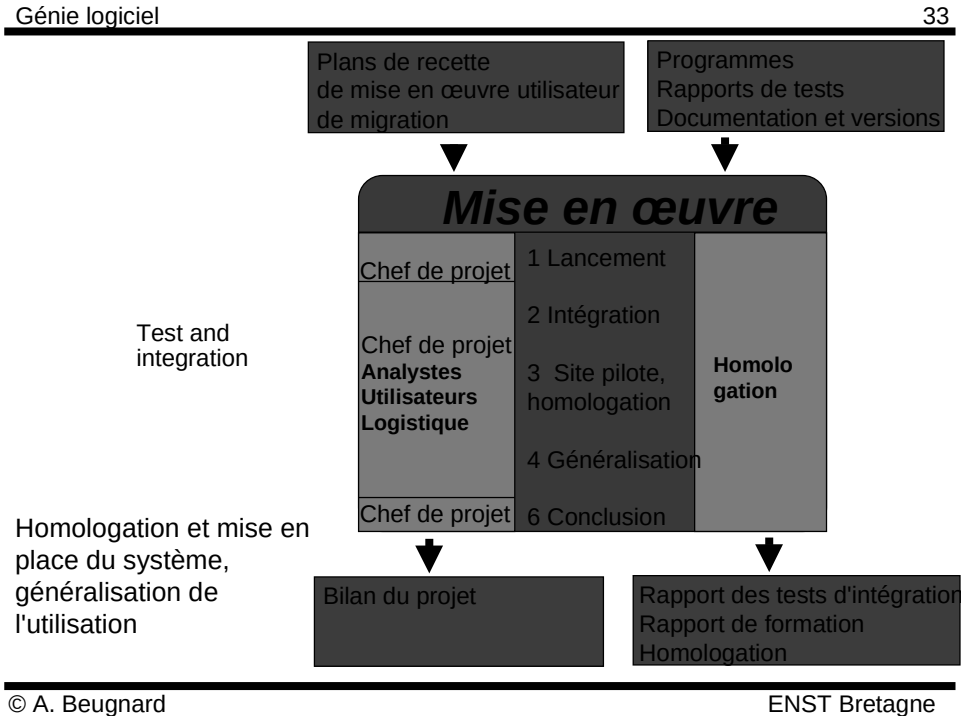


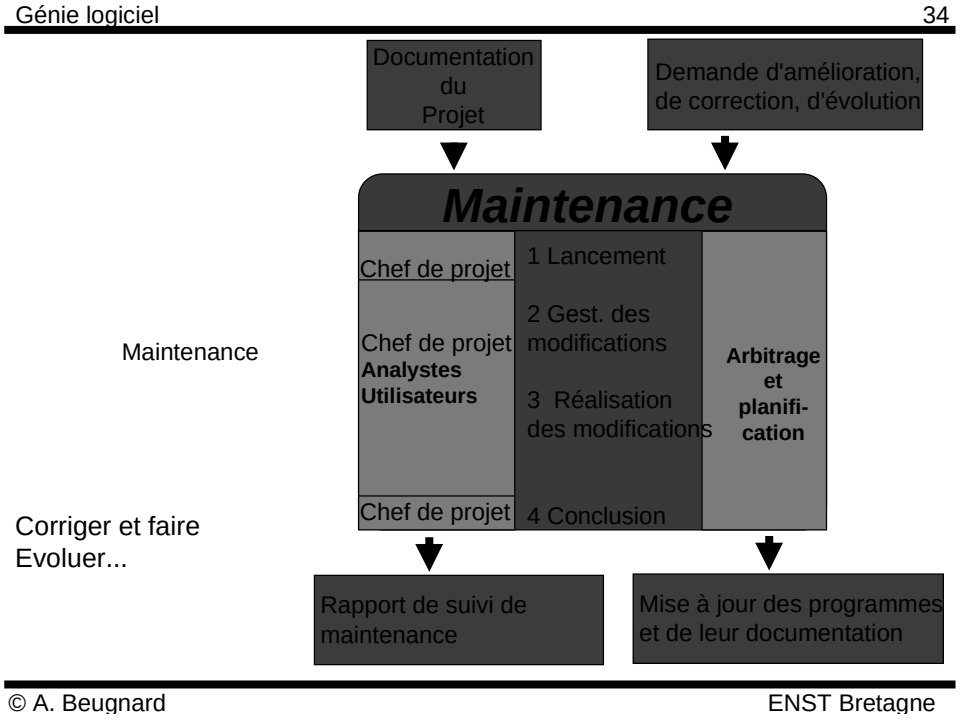
moins de cohérence, mais moins d'inactivité et détection plus précoce d'éventuels aléas

Stratégie du projet Pilote



un compromis classique ...

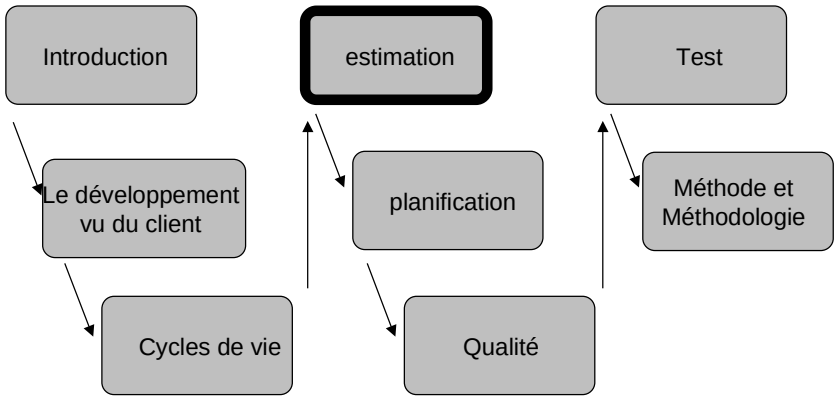

Génie logiciel
33
Plans de recette
de mise en œuvre utilisateur
de migration
Programmes
Rapports de tests
Documentation et versions
Mise en œuvre
Chef de projet
1 Lancement
2 Intégration
Chef de projet
Analystes
Utilisateurs
Logistique
3 Site pilote, homologation
4 Généralisation
Homologation
Chef de projet
6 Conclusion
Test and integration
Homologation et mise en place du système, généralisation de l'utilisation
Bilan du projet
Rapport des tests d'intégration
Rapport de formation
Homologation
© A. Beugnard
ENST Bretagne


Génie logiciel
34
Documentation du Projet
Demande d'amélioration, de correction, d'évolution
Maintenance
Chef de projet
1 Lancement
2 Gest. des modifications
Chef de projet
Analystes
Utilisateurs
3 Réalisation des modifications
Arbitrage et planifi-cation
Chef de projet
4 Conclusion
Maintenance
Corriger et faire Evoluer...
Rapport de suivi de maintenance
Mise à jour des programmes et de leur documentation
© A. Beugnard
ENST Bretagne

Bibliographie

- Cyrille Chartier-Kastler, *Précis de conduite de projet informatique,* Les éditions d'organisation, 1995
- John J. Marciniak, *Acquisition Management*, in Encyclopædia of Software Engineering, Vol 1, pp 4--24, John Wiley & Sons, 1994
- John J. Marciniak and D.J Reifer, *Software Acquisition Management*, John Wiley &Sons, Inc, New York, 1990
- George Wilkie, *Object-Oriented Software Engineering*, Addison-Wesley, 1993.

plan



Estimation

- **Principes généraux**
- Quelques techniques
- La méthode Cocomo
- Les points de fonction

Estimation

Il est difficile de prévoir...surtout l'avenir
B.Shaw

Nécessité de précision pour :

- Identifier, justifier l'usage des ressources (personnes, temps, capital) et leur priorité
- Négocier les budgets et les plannings
- Optimiser les coûts et la productivité
- Mesurer les risques et prendre les bonnes décisions

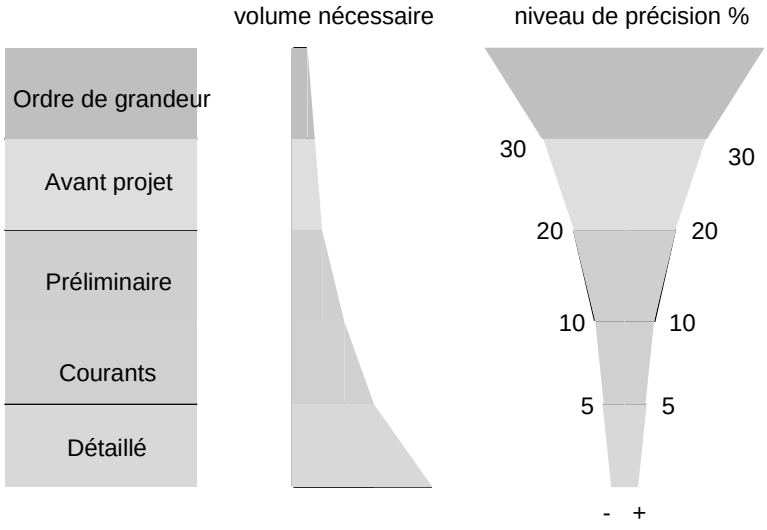
Analyse de risque

- Gérer l'impact des modifications
- Gérer les aléas

Qu'estime-t-on ?

- La durée d'un projet
- Le nombre de personnes impliquées
- Le coût
- La taille du système

Niveau d'information

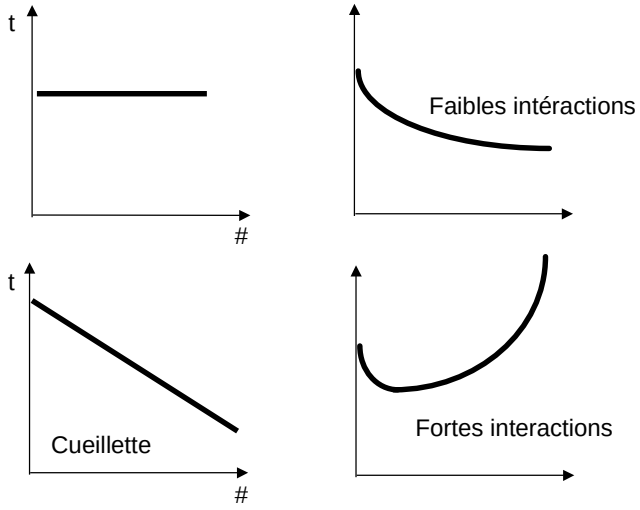


Pourquoi de mauvaises estimations ?

- Les besoins sont vagues, et la qualité des résultats difficile à évaluer
- Les "managers" sont trop optimistes et pensent que tout ira bien ; Ils espèrent le succès parce qu'ils veulent le travail.
- On ne tire pas assez expérience du passé, on utilise trop le « pif »; Il faut de l'expérience pour bien estimer
- Imprécision des notions d'homme-mois, de ligne de code source

Alors, comment faire ...

Mythe de l'homme-mois



Pièges à éviter

- Faire trop précis
  - travailler avec des marges d'erreur importantes
- Sous-estimer
  - être exhaustif dans la liste des choses à estimer
- Sur-estimer
  - ne pas intégrer systématiquement tous les coûts des aléas possibles
- Confondre objectif et estimation
  - résister à "Il ne faut pas que ça coûte plus de… »
- Vouloir tout estimer
  - savoir avouer son ignorance

Qualités de l'estimation

- Rendue dans les délais
- Homogène en précision
- Honnête
- Complète
- Afficher les hypothèses
- Réaliste
- Proche du coût réel

Best Estimate

Qualités de l'estimateur

- Utile au client
- Organisé
- Objectif
- Compétent
- Créatif
- Réaliste
- Manier l'analogie

Peut faire faux, pas idiot

Processus d'estimation

- Définir le vocabulaire
- Identifier les composants (WBS) et leurs versions
- Estimer la taille des composants
- Estimer la précision, la porté et la difficulté
- Estimer les ressources nécessaires
- Valider les estimations - Evaluer les risques (What-if)
- Allouer les ressources
- Suivre et affiner les estimations

Estimation

- Principes généraux
- **Quelques techniques**
- La méthode Cocomo
- Les points de fonction

Méthodes d'estimations

- Par analogie
- Modèle paramétrique
- Oracle
- PERT
- Bottom-Up

Par analogie

- Exploitation des expériences passées
- Catalogue des projets et estimations passés

- Ce qui est analysé :
  - Taille
  - Durée
  - Effort
  - Complexité
  - Coût

- On rapproche ce qui se ressemble...

Modèles paramétriques

- Les estimations sont basées sur des modèles mathématiques reposant sur divers paramètres.

- Elles sont largement répandus
  - COCOMO
  - SLIM
  - PRICE-S
  - SoftCost
- Elles disposent d'outils

Oracle

- Equipe d'experts
- Atteinte d'un consensus par négociation

PERT

- Estimations reposant sur l'hypothèse d'une répartition normale des estimations.
- On réalise plusieurs* estimations avec une méthode "par analogie" ou "oracle" :
  - la pire (l)
  - la moyenne (m)
  - la meilleure (h)
- Effort = (l+4m+h)/6

* pour être valide, les estimations (l, m, h) doivent être non corrélées (sources différentes

Bottom-Up

- Les estimations par analogie, PERT, paramétrique, oracle, sont faites par
  - activité ou
  - composant élémentaire
- Puis consolidés (en suivant le WBS, par exemple) jusqu'au sommet du projet.

Comparaison

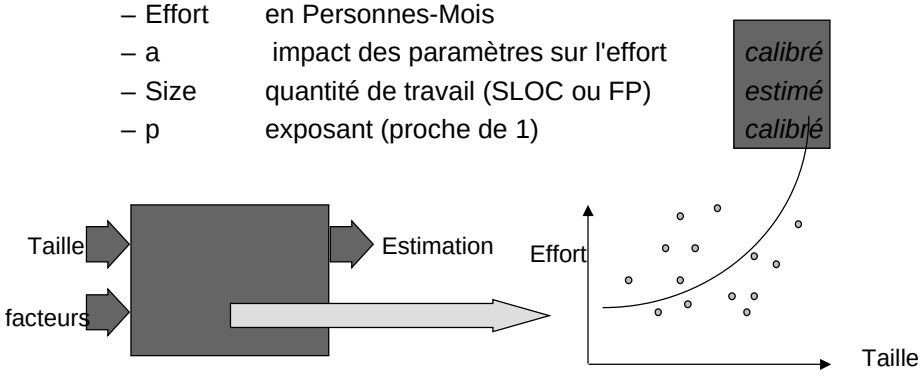
Méthode	Forces	Faiblesses
Analogie	basé sur l'expérience	les expériences passées peuvent être inappropriées
Paramétrique	objective et répétable beaucoup de facteurs	calibration difficile subjectivité des facteurs
Oracle	analyse croisée	dépend de la qualité des experts
PERT	borne le risque	difficile d'avoir de bonnes entrées
Bottom-Up	très détaillée	long et beaucoup d'efforts

Principe des modèles paramétriques

- Effort = a (Size)^p
  Avec :
  - Effort en Personnes-Mois
  - a impact des paramètres sur l'effort *calibré*
  - Size quantité de travail (SLOC ou FP) *estimé*
  - p exposant (proche de 1) *calibré*

Principe des modèles paramétriques

- Effort = a (Size)^p
  Avec :
  - Effort en Personnes-Mois
  - a impact des paramètres sur l'effort *calibré*
  - Size quantité de travail (SLOC ou FP) *estimé*
  - p exposant (proche de 1) *calibré*



Estimation

- Principes généraux
- Quelques techniques
- **La méthode Cocomo**
- Les points de fonction

COCOMO

- Modèle paramétrique
- Facteurs dans le domaine public
- 3 modes de bases
  - organique — petite équipe, environnement stable
  - semi-détaché — équipe de taille moyenne
  - détaché — grande équipe, répartie, nouvel environnement

COCOMO simple

• mode organique :	HM = 2,4 (KLSL) $^{1.05}$	Effort
• semi-détaché :	HM = 3.0 (KLSL) $^{1.12}$	
• détaché :	HM = 3,6 (KLSL) $^{1.20}$	
• mode organique :	TDEV = 2.5 (HM) $^{0.38}$	Durée
• semi-détaché:	TDEV = 2.5 (HM) $^{0.35}$	
• détaché :	TDEV = 2.5 (HM) $^{0.32}$	

N = HM / TDEV

HM : Hommes-Mois (152heures)
KLSL : Kilo de Ligne de Source Livrées

COCOMO intermédiaire

- Quinze facteur correctifs sont introduits, valués de VeryLow à XtraHigh
- Pour le projet :
  - fiabilité requise du logiciel
  - taille de la base de donnée
  - complexité du produit
- Pour les contraintes de l'environnement :
  - contraintes de temps d'exécution
  - contraintes de place mémoire
  - stabilité de la machine virtuelle
  - système de développement interactif ou non

COCOMO intermédiaire

- Pour le personnel :
  - aptitude à l'analyse
  - expérience du domaine
  - expérience de la machine virtuelle
  - aptitude à la programmation
  - expérience du langage
- Pour les méthodes :
  - méthode de programmation moderne
  - outils logiciels
  - durée du développement

COCOMO détaillé

- Les facteurs correctifs dépendent de la taille (KLSL)
- Une répartition de l'effort sur les phases de développement est réalisée

Estimation de la taille

- Les "Function points" (Albrecht, 1979, 1984)
  - Composant identifiable et unique (fonction)
  - 5 types de fonction :
    - » Input
    - » Output
    - » Inquiry
    - » Internal Logical File
    - » External Interface File

Estimation

- Principes généraux
- Quelques techniques
- La méthode Cocomo
- **Les points de fonction**

Function Point

- Compter le nombre de fonctions (FC)
- Ajuster selon leur complexité (c_i)
  14 facteurs notés de 0 (pas d'influence) à 5 (fondamental)
  - communication par message
  - distribution de données ou de fonctions
  - haut taux de transaction
  - calcul complexe
  - conception multi sites
  - conception facilement maintenable

$$FP = FC * PCA$$

$$PCA = 0.65 + 0.01 \sum c_i$$

$$KLSL = -5 + 0.2\ FP$$

Comparaison de SLOC et de FP

	Case A ASM (100K)	Case B ADA (30K)	Difference	
requirements	20	20	0	
analysis and design	30	30	0	
coding	100	30	-70	
testing	50	30	-20	
documentation	20	20	0	
management	30	20	-10	
total effort	250	150	-100	pro High Level lang.
total cost	$1,250,000	$750,000	-$500,000	
cost per line	$12.50	$25	+$12.5	Apparently pro asm !
lines per month	400	200	-200	

Comparaison de SLOC et de FP

Count	Element	Weight	Totals
8	input	X 4 =	32
17	output	X 5 =	85
12	inquiry	X 4 =	48
5	internal	X 10 =	50
5	external	X 7 =	35
total			*250*
complexity adjusts			*1.2*
FP			*300*

Comparaison de SLOC et de FP

	Case A ASM (100K)	Case B ADA (30K)	Difference	
requirements	20	20	0	
analysis and design	30	30	0	
coding	100	30	-70	
testing	50	30	-20	
documentation	20	20	0	
management	30	20	-10	
total effort	250	150	-100	pro High Level lang.
total cost	$1,250,000	$750,000	-$500,000	
cost per FP	$4167	$2500	+$1667	
FP per month	1.2	2	+0.8	

Conseils

- Définir et normaliser son propre processus d'estimation
- Utiliser plusieurs techniques pour les recouper
- Calibrer le modèle avec ses propres expériences
- Evaluer les risques et les différentes options
- Stocker le données passées
- Etre réaliste

Quelques trucs...

- Pour les petits projets
  - utiliser l'analogie
  - paramétrique au moins 3-5 personnes et 5KSLOC (sensibilité au personnes affectées)
- Pour augmenter la précision des modèles paramétriques, utilisez vos propres données
- Pour gérer le risque
  - estimer sans en tenir compte
  - introduisez les facteurs de risque un par un

Bibliographie

- D.J. Reifer, *Cost estimation*, Encyclopædia of Software Engineering, pp209-220, J.J. Marciniak ed, John Wiley & Sons, Inc, New York, 1994
- B.W Boehm, *Software Engineering Economics*, Prentice Hall, Inc, Englewood Cliffs, NJ, 1981
- D.V. Ferens, *COCOMO*, Encyclopædia of Software Engineering, pp103-110, J.J. Marciniak ed, John Wiley & Sons, Inc, New York, 1994
- J.B Dreger, *Function Point Analysis,* Prentice Hall, Inc, Englewood Cliffs, NJ, 1989
- C.S. Fugate, Estimating the cost of Object-Oriented Programming, *Journal of Parametrics,* **XI**(1), Aug 1991
- C. Jones, *Productivity*, Encyclopædia of Software Engineering, pp869-872, J.J. Marciniak ed, John Wiley & Sons, Inc, New York, 1994