

Gestion de projets

analyse des besoins et génie logiciel

Philippe Collet

Licence 3 MIAGE
2014-2015

[http://miageprojet2.unice.fr/User:PhilippeCollet/
Gestion_de_projet_2014-2015](http://miageprojet2.unice.fr/User:PhilippeCollet/Gestion_de_projet_2014-2015)

Objectif

- Appréhender et appliquer les concepts de l'analyse des besoins et de la gestion des projets informatiques à grande échelle.

Evaluation

- Participation aux ateliers et réalisation du projet en TD : 20 %
- DoW de projet à réaliser en équipe : 40 %
- Contrôle terminal sur papier : 40 %
 - Support de cours autorisé



TD et projet : fonctionnement

- 24/10 : premier cours / atelier
- 07/11 : distribution des projets
 - Date limite pour retour des formations d'équipe par mail (4 par équipes)
- 05/12 : dernier TD
 - Dernier point d'avancement de votre DoW sur le projet
- 06/01/2015 : rendu du projet (DoW + planning sur site de gestion)

Plan

- Introduction : mythes et réalités
- Génie logiciel et gestion de projet
- Analyse des besoins, cahier des charges
- Cycle de vie du logiciel, organisation
- Gestion de projets et agilité
- Analyse des risques, estimation
- Vérification et Validation

Introduction

- Pourquoi ?
- Génie logiciel, projet : définition(s)
- Pourquoi c'est difficile ?

Pourquoi le Génie logiciel ?

- pour passer du développement logiciel *ad hoc* et *imprévisible*

à

- un développement logiciel *systematique* et *réfléchi*

Génie logiciel : historique

- Histoire drôle : la facture à 0 euro
- Réponse à la crise du logiciel, il y a 40 ans
- Conférence OTAN 1968

La crise du logiciel

- Grosses erreurs :
 - Les sondes perdues (Vénus dans les années 60, Mars en 99)
 - La fausse attaque de missiles (1979)
 - Les missiles Patriotes (1991)
 - 1er vol d'Ariane 5 (1996)
 - L'aéroport de Denver (1994-96)
 - L'an 2000...
- Les projets logiciels
 - ne livrent pas le produit dans les temps
 - coûtent beaucoup plus chers que prévu.
 - délivrent un produit de qualité très faible
 - échouent dans la majorité des cas !!!
 - Étude américaine de 1995 : 81 milliard \$ / an en échec

Pourquoi ne construit-on pas les logiciels comme on construit des ponts ?

- Génie civil
 - Échecs moins nombreux
 - L 'écroulement est grave et met en danger l 'utilisateur
 - On ne répare pas un pont « buggé », on reconstruit un pont qui s'écroule.
 - On inspecte tous les ponts construits sur le même modèle
 - Les ponts résistent à toutes les conditions (à 99 %...)
- Génie logiciel
 - Échecs très nombreux
 - *Crash* système pas considéré comme inhabituel
 - Cause du bug pas directement identifiable
 - Dommages mineurs
 - A part dans les systèmes critiques, on considère que le logiciel ne peut anticiper **TOUTES** les situations

☞ ***Différence d 'approche face à l 'échec, face aux pannes***

Pourquoi ne construit-on pas les logiciels comme on construit des ponts ?

- Génie civil

- Plusieurs milliers d 'années d 'expérience dans la construction des ponts
- Les ponts sont des systèmes continus et analogiques
- On repeint un pont, on change l 'enrobée de la route...
- On ne reconstruit pas la moitié d 'un pont

- Génie logiciel

- Les systèmes informatiques se complexifient trop vite
- Les logiciels passent par des états discrets, dont certains ne sont pas prévus
- Ajouts, changements de fonctionnalités, de plateformes...

 ***Différence dans la complexité et dans la maintenance***

Génie logiciel : définition (ou presque)

- Discipline (= méthodes, techniques et outils)
 - basée sur le savoir (théorique)
 - le savoir-faire (pragmatique)
 - et le faire savoir (communication)
 - pour produire (développement)
 - de façon industrielle (taille, diffusion)
 - des logiciels (= les produits)
 - *de qualité au meilleur prix...*

Les mythes de gestion de projet

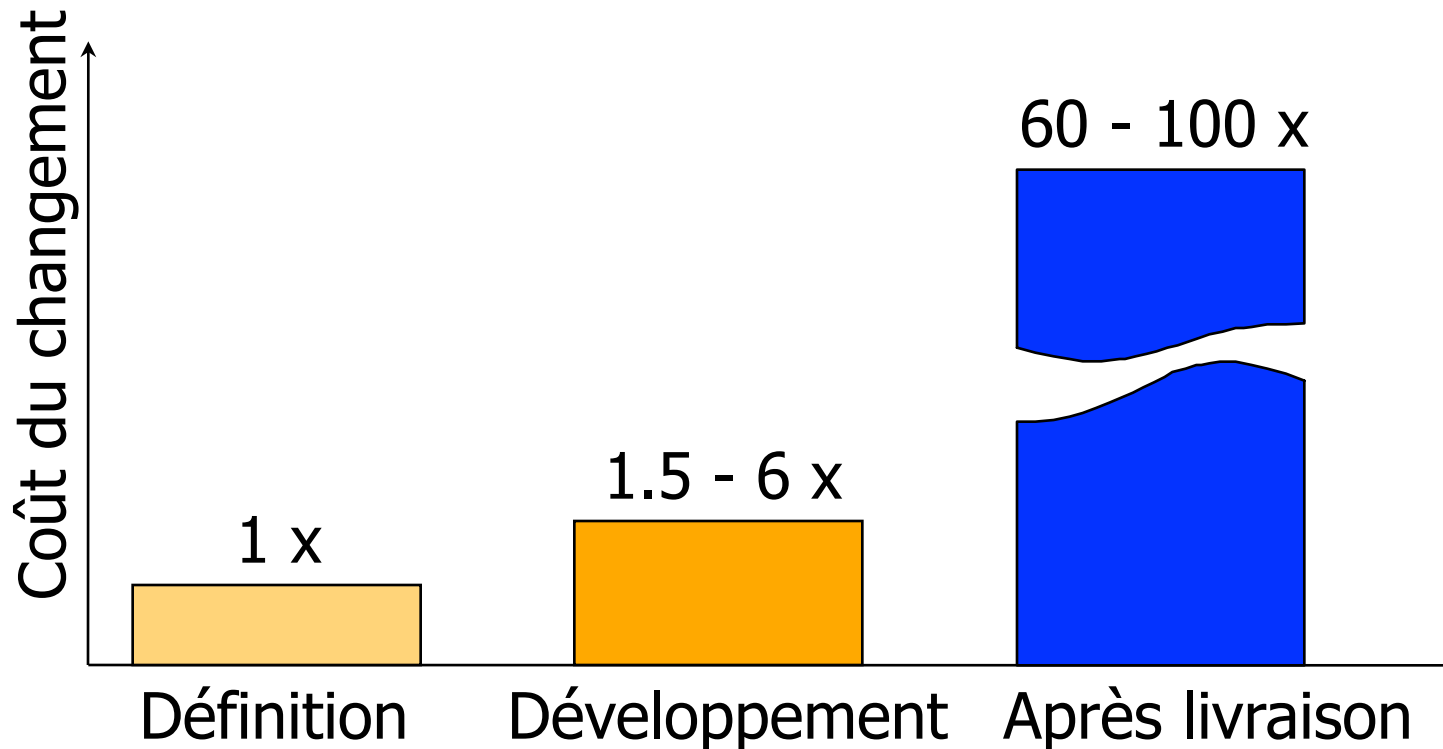
- Les outils actuels sont la solution
 - un *nul* avec un outil est toujours un *nul*
- Si on est en retard, on ajoutera du personnel



Les mythes du client

- Une idée générale des objectifs est suffisante pour commencer le codage – on ajoutera les détails plus tard
 - Une forte communication entre clients et développeurs est toujours nécessaire
- Les changements peuvent être facilement répercutés parce que le logiciel est flexible
 - Les changements ne peuvent être évités, c'est la vie...
 - Les changements tardifs coûtent très chers

L'impact des changements



Les mythes des développeurs

- Une fois que le programme est écrit et qu'il *tourne*, le travail est terminé
 - 50-70% de l'effort est réalisé après la livraison
- Jusqu'à ce que le programme *tourne*, il n'y a aucun moyen d'évaluer sa qualité
 - Inspections & revues
- La seule chose à livrer pour un projet réussi est un programme qui marche
 - Documentation (utilisateur, maintenance)

Pourquoi c'est difficile ?

- Invisibilité du logiciel
- Facilité apparente d'écriture et de modification
- Le produit fini est différent du programme :
 - produit logiciel : généralisation, tests, documentation, maintenance * **3**
 - programme intégré dans un système (interfaces) * **3**
 - ☞ produit logiciel intégré dans un système : * **9**
 - ☞ *The mythical man-month* de Frédéric Brooks (1975)

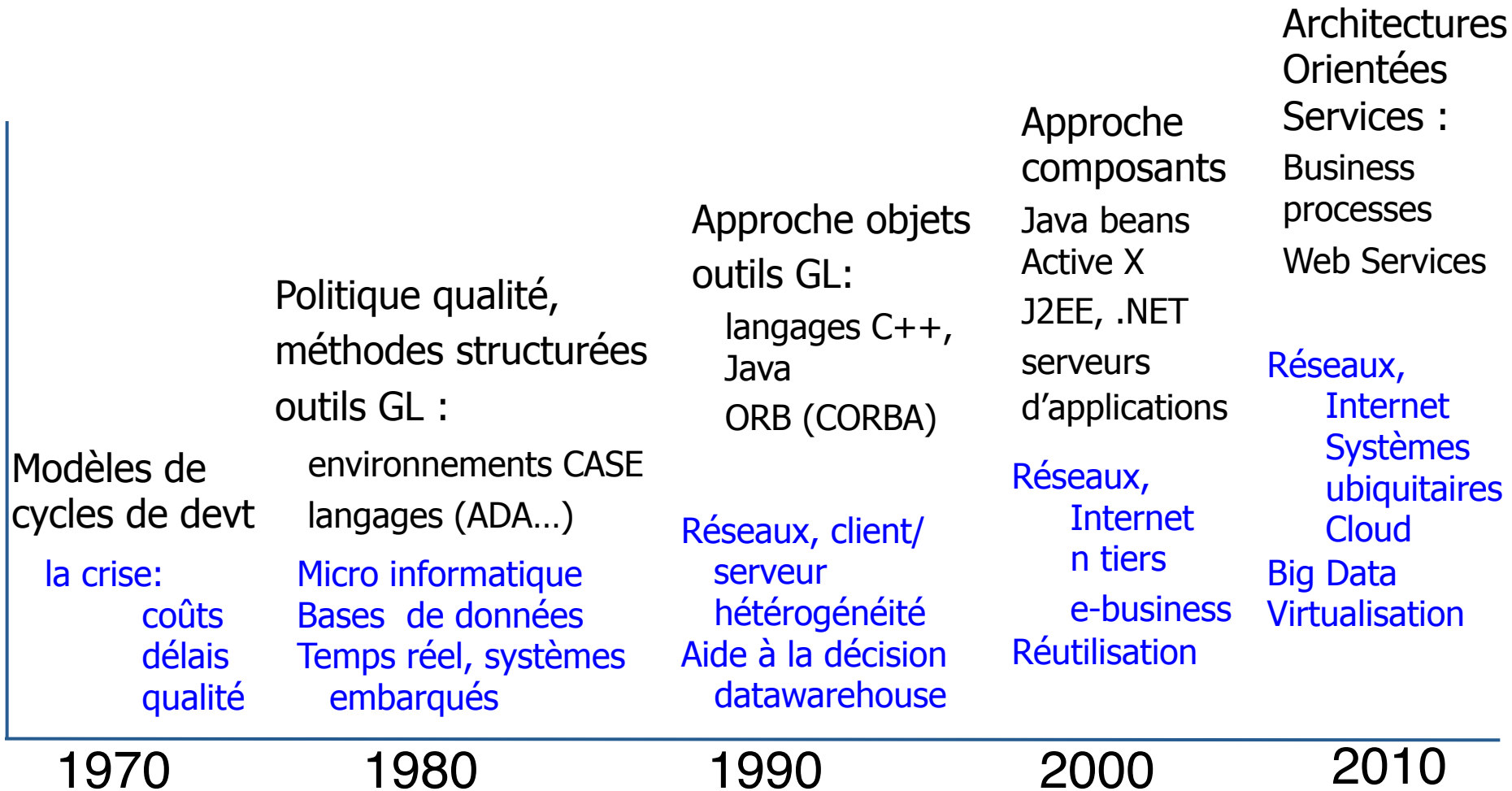
Pourquoi c'est difficile ? (suite)

- La spécification :
 - Le logiciel modifie son environnement
- La maintenance (67 % du coût total)
 - corrective (50 %) : 60 % des défauts proviennent d'erreurs de spécification ou de conception.
 - évolutive : se méfier de l'effet 2ème version...
- L'optimisme !
 - *Ajouter du personnel à un projet en retard ne fait que le retarder plus* Loi de Brooks

Les réponses à la crise

- Recherche du concept de qualité
 - Maîtrise du processus de développement
 - Méthodes et outils structurés (CASE)
 - Programmes de recherche
- Approche solo
 - Prolog, Lisp, Smalltalk, etc.
- Approche par objets
 - ☞ Réutilisation théorique
- Approche par composants
 - ☞ Réutilisation quasi-effective

40 ans de Génie logiciel



Le logiciel, fin 2014

- Fiabilité meilleure mais...
- **partout**, sous toutes les formes
- gros, **très très** gros, cher, **très très** cher !
- Types :
 - Sur mesure (à partir de composants, de services)
 - Générique (les progiciels)
 - Interconnectés, en constante évolution...
- Acteurs : constructeurs, SSII, utilisateurs

Liste (non-exhaustive) des problèmes


- Productivité
 - Coûts et délais
- Simplicité
 - Uniformité, orthogonalité, unicité, normalisation
- Communication H/M
 - Ergonomie, interactivité, multimédia, simplicité, rapidité, documentation (contextuelle)
- Fonctionnels
 - Étendue et pertinence des services, fiabilité (correction, robustesse)

Liste des problèmes (suite)

- Matériau
 - Logiciel, structure, langage, modularité...
- Organisation
 - Gestion de projet visibilité, protections, contrôles
- Réalisme
 - Adéquation aux besoins, évolutivité
- Économique
 - Réutilisabilité, transportabilité
- Diversité
 - BD, IA, Calcul, Parallélisme, Réseau, Internet, intranet
- Divers
 - Juridique, psychologique

Il faut donc...

- Développer des
 - nouveaux produits
 - nouvelles fonctions
 - nouveaux portages
- A partir
 - d'un cahier des charges
 - d'applications existantes
 - de composants existants
- En
 - interne
 - sous-traitance

 ***avec des objectifs
de qualité et
de productivité***

Génie logiciel : les besoins

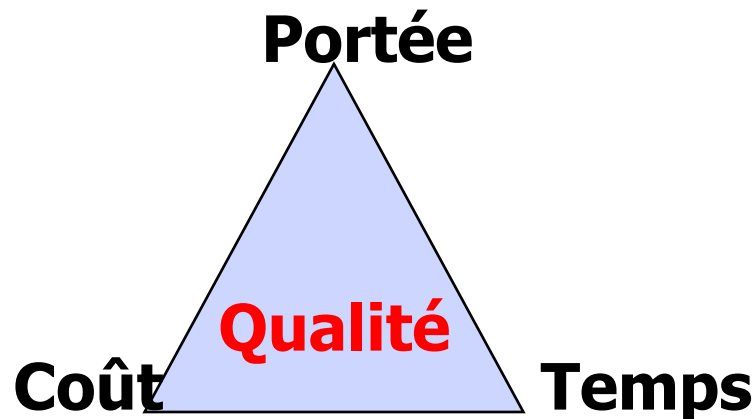
- Langages *pour décrire*
- Outils *pour manipuler*
- Méthodes *pour décider*
- Théories *pour démontrer*
- Professionnels *pour réaliser*
- Logistique *pour supporter*

Qu'est qu'un projet ?

- Définition
 - Un effort temporaire
 - qui est progressivement planifié, contrôlé et exécuté
 - par des personnes travaillant avec des contraintes de ressources
 - pour créer un produit, service ou résultat unique
- Temporaire
 - Début et fin sont définies
 - Pas forcément court, mais fini
- Planifié, contrôlé et exécuté
 - Nécessité d'une planification initiale et d'un suivi
 - Le travail s'organise pour accomplir des objectifs (exécution)
 - Le travail nécessite des vérifications pour être correctement exécuté
 - Et tout cela, progressivement, en étapes, en affinant au fur et à mesure

Qu'est qu'un projet ? (suite)

- Par des personnes
 - La dimension humaine est primordiale
- Avec des contraintes de ressources
 - Contraintes de temps, de coût
 - Toute limitation ou frontière du projet est une contrainte
- *Gérer un projet, c'est essentiellement gérer continuellement ces contraintes, pour atteindre des critères de qualité prédéfinis*



Qu'est qu'un projet ? (suite)

- Pour créer un produit, service ou résultat unique
 - Le projet crée quelque chose de nouveau
 - Quelque chose de tangible (produit) ou non (service, résultat)
 - Exemple : Diminuer le temps d'attente au téléphone de 20 %
- Comment déterminer l'objectif du projet ?
 - L'objectif du projet est quelque chose que l'organisation ne peut obtenir par son fonctionnement normal
 - Exemple de fonctionnement normal : Produire les fiches de paie mensuelles
- Questions
 - Pour un constructeur de maisons, chaque chantier est-il un projet ?

Naissance du projet

- Pourquoi démarre-t-on un projet ?
 - Besoin, demande, idée, inspiration...
- Besoin organisationnel
 - Amélioration dans le processus métier ou création d'un nouveau
- Demande du marché
 - Opportunité pour un produit ou un service
- Demande d'un client
- Avance technologie (ou obsolescence)
- Nécessité légale
- Besoin social

Fin du projet

- Quand
 - Les objectifs sont atteints
 - Il devient clair qu'on ne pourra atteindre les objectifs
 - Le besoin n'existe plus

Projet et production : ne pas confondre

- Production
 - Efforts de l'organisation pour soutenir son métier principal
 - C'est une activité récurrente
- Points communs avec le projet
 - Deadlines (dates limites), personnes, contraintes de temps et de coût
 - Planification, contrôle
- Différences avec le projet
 - Toujours en cours, pas d'objectif fixé, ni de terminaison, organisation stable
 - Incertitude faible, retour sur investissement positif

Caractéristiques du projet

- Livrables
 - La partie la plus importante d'un projet, souvent multiples
 - On parle parfois d'artefact, comme quelque chose qu'il est nécessaire de produire, sans que ce soit un livrable
- Portée du produit
 - Caractéristiques et fonctionnalités du produit
- Portée du projet
 - Comment les objectifs vont être atteints
 - Donc, le travail, et uniquement le travail, pour réaliser les... livrables
 - Donc, directement impacté par le temps et le coût
- Impossible de définir la portée du projet sans la portée du produit

Qualités du logiciel

- Il faut bien distinguer
 - Les qualités utiles à l'utilisateur, donc *a priori* souhaitées par le client
 - Phases d'exploitation
 - Les qualités utiles au développeur
 - Phases de construction et de maintenance

Qualités pour l'utilisateur

- Fiabilité = Validité + Robustesse
 - Validité (Efficacité) \equiv correction, exactitude
 - *Efficacité : qualité d'une chose ou d'une personne qui donne le résultat escompté*
 - ☞ Assurer exactement les fonctions attendues, définies dans le cahier des charges et la spécification, en supposant son environnement fiable
 - ☞ Adéquation aux besoins

Qualités pour l'utilisateur (suite)

- Robustesse: faire tout ce qu'il est utile et possible de faire en cas de défaillance: pannes matérielles, erreurs humaines ou logicielles, malveillances...
- Performance (parfois appelée efficacité)
 - Utiliser de manière optimale les ressources matérielles : temps d'utilisation des processeurs, place en mémoire, précision...
- Convivialité
 - Réaliser tout ce qui est utile à l'utilisateur, de manière **simple, ergonomique, agréable** (documentation, aide contextuelle...

Qualités pour le développeur

- Documentation
 - Tout ce qu'il faut, rien que ce qu'il faut, là où il faut, quand il faut, correcte et adaptée au lecteur : **crucial !**
- Modularité = Fonctionnalité + Interchangeabilité + Évolutivité + Réutilisabilité
 - Fonctionnalité
 - Localiser un phénomène unique, facile à comprendre et à spécifier

Qualités pour le développeur (suite)

– Interchangeabilité

- Pouvoir substituer une variante d'implémentation sans conséquence fonctionnelle (et souvent non-fonctionnelle) sur les autres parties

– Évolutivité

- Facilité avec laquelle un logiciel peut être adapté à un changement ou une extension de sa spécification

– Réutilisabilité

- Aptitude à être réutilisé, en tout ou en partie, tel que ou par adaptation, dans un autre contexte : autre application, machine, système...

Qualités pour l'entité de développement

- Client satisfait (*est-ce possible ?*)
- Coût minimum de développement
 - Nombre de développeurs
 - Formation des développeurs
 - Nombre de jours de réalisation
 - Environnement
 - Réutilisation maximale

Génie logiciel : le défi

- Contradictions apparentes
 - Qualités vs coût du logiciel
 - Qualités pour l'utilisateur vs qualités pour le développeur
 - Contrôler vs produire
- Conséquences
 - ☞ Chercher sans cesse le meilleur compromis
 - ☞ Amortir les coûts
 - Premier exemplaire de composant coûteux à produire ou à acheter, puis amortissement...

Objectifs de qualité

- Réduire le nombre d'erreurs résiduelles
- Maîtriser coût et durée du développement
- sans nuire à la créativité et à l'innovation
- Adéquation aux besoins
- Efficacité temps/espace
- Fiabilité
- Testabilité, Traçabilité
- Adaptabilité
- Maintenabilité
- Convivialité (interface et documentation)

 ***doivent rejoindre les objectifs de productivité***