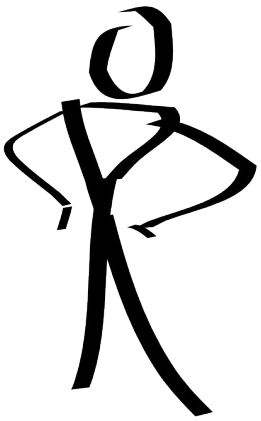
Méthodes

M1 INFO GL CM

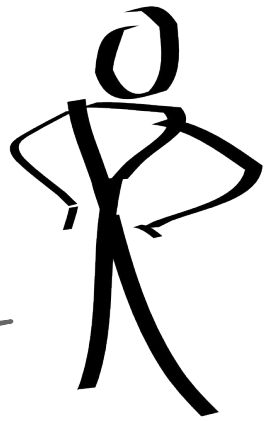
Les acteurs

Les acteurs : utilisateurs



- Fonctionnalités
- Facilité d’utilisation
- Performance
- Sécurité
- Robustesse

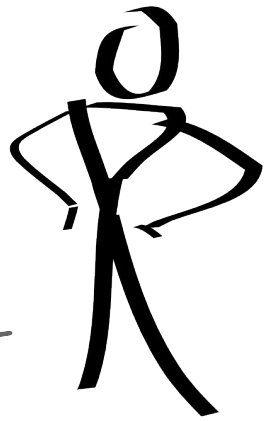
Les acteurs : client



Maîtrise d’ouvrage

- Respect des coûts et des délais
- Garantie de fonctionnement
- Pérennité
- Réutilisation

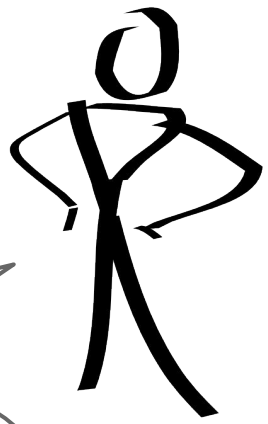
Les acteurs : manager



Maîtrise d’oeuvre

- Relation client
- Bénéfice
- Responsabilité limitée/maîtrisée

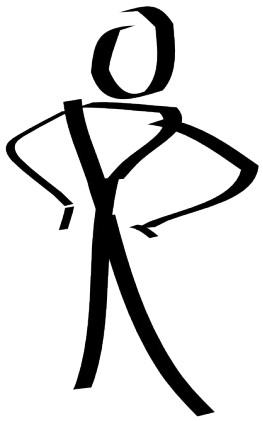
Les acteurs : chef de projet



Maîtrise d'oeuvre

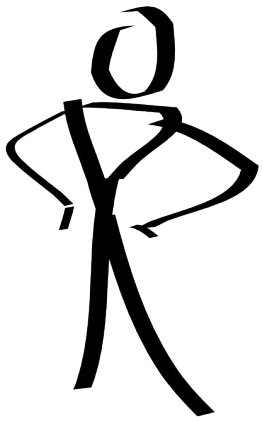
- Possibilité d'évaluer les progrès
- Bonne expression des contraintes
- Technologies maîtrisées
- Structuration correspondant à ses équipes
- Réutilisation des composants internes
- Maîtrise des risques
- Maîtrise des coûts et des délais
- Simplicité de l'ensemble

Les acteurs : analyste



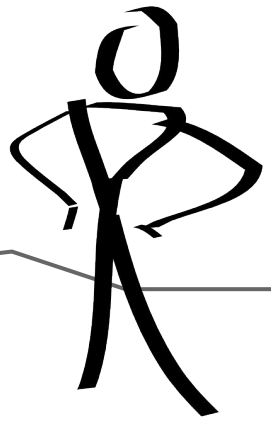
- Accès aux utilisateurs
- Clarté du domaine
- Indépendance technologique

Les acteurs : architecte



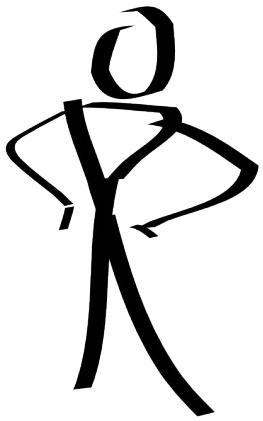
- Cohérence de l’architecture
- Simplicité
- Pérennité
- Réutilisation de l’existant

Les acteurs : développeur



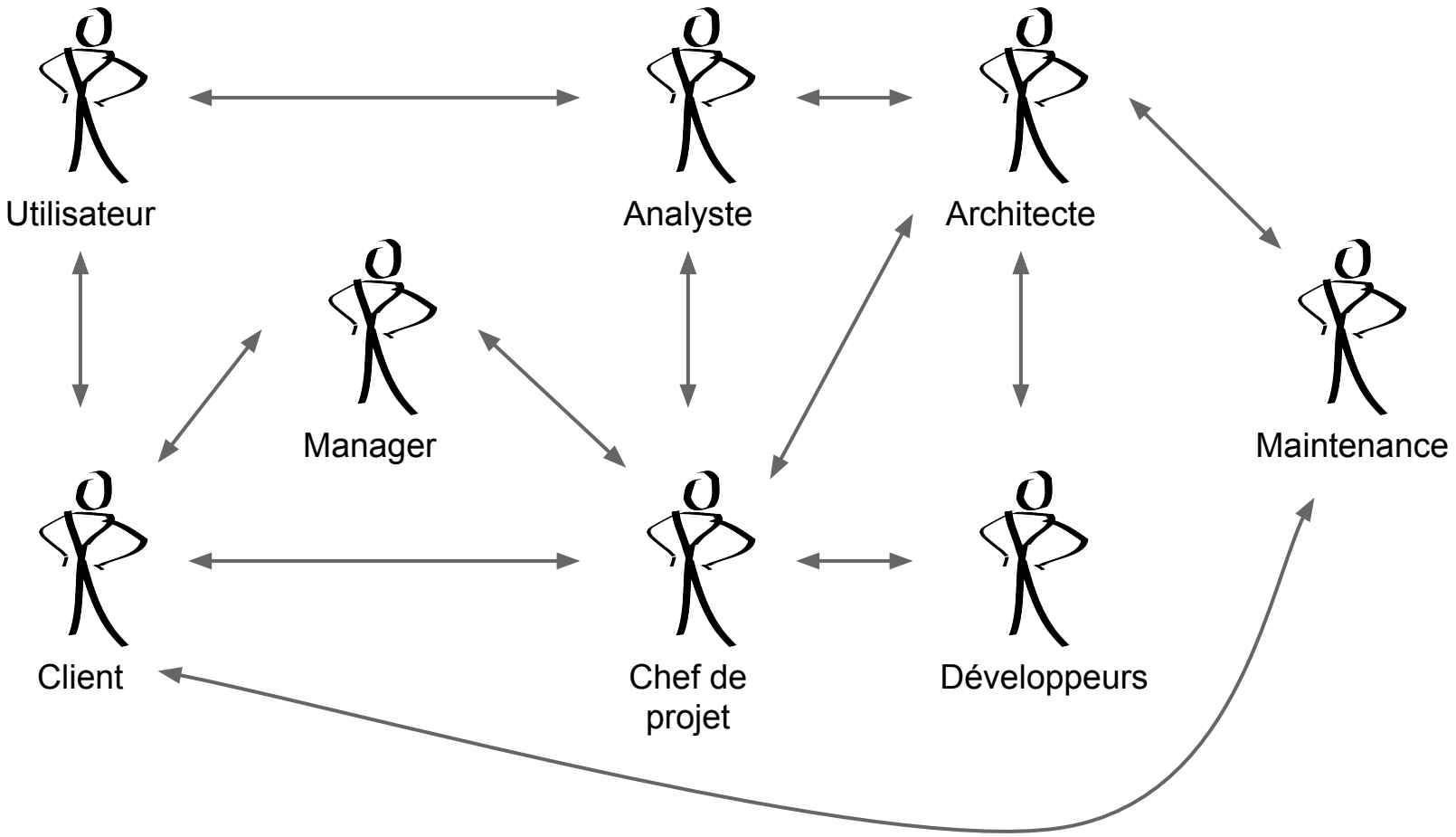
- Technologie connue ou « d'avenir »
- Simplicité de l'interface de ses composants
- Pas trop de contraintes imposées sur ses composants

Les acteurs : maintenance

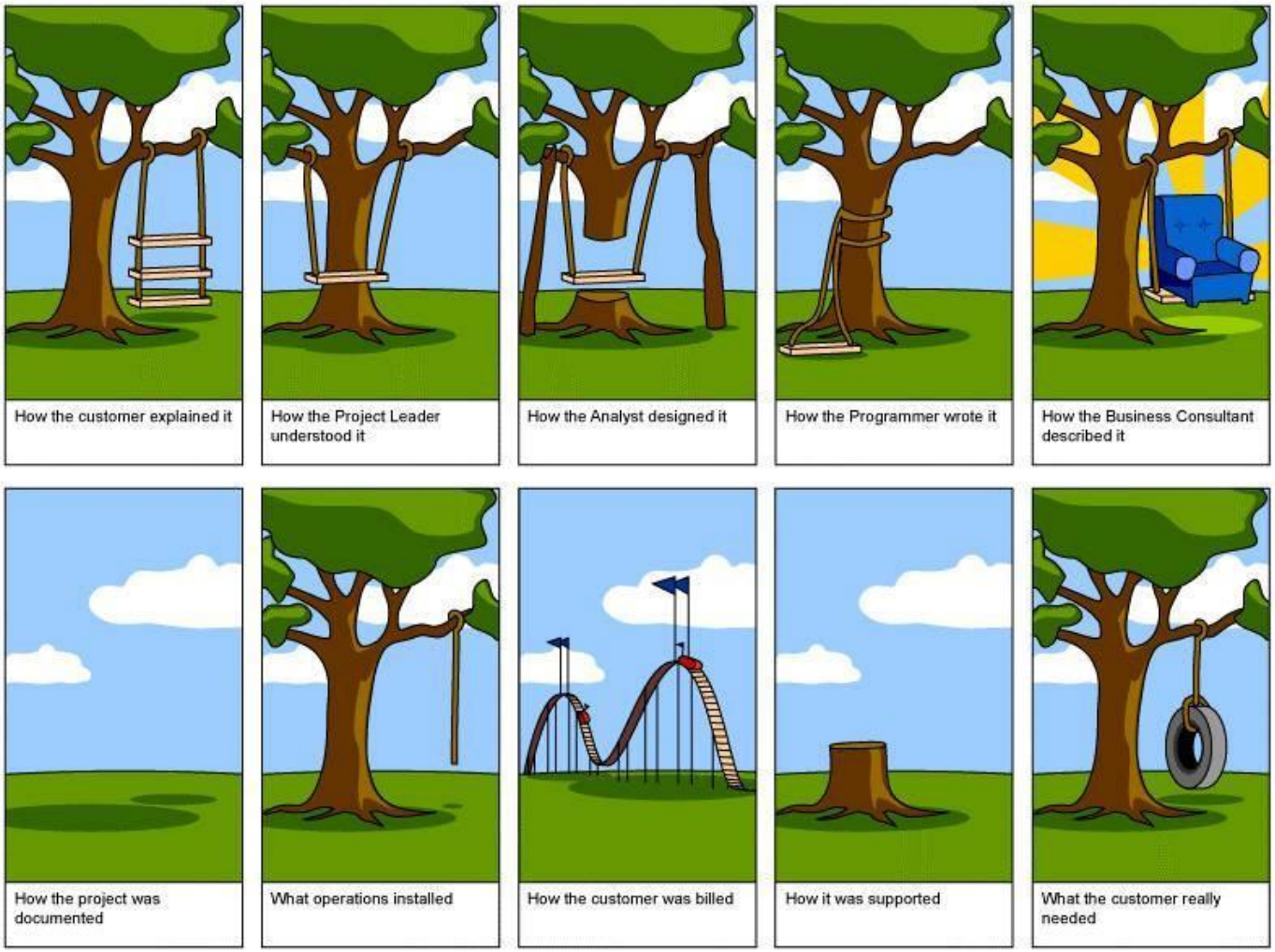


- Facilité de modification
- Isolation des composants
- Existence d’interfaces d’administration
- Technologies connues et pérennes

Les acteurs : interactions



Les acteurs : interactions



Les phases

Les phases : analyse des besoins

Un préalable si le client n'a qu'une idée peu précise du système à réaliser.

- Étude informelle des fonctionnalités externes, sans considérations techniques
- Point de vue métier / utilisateur
- Dialogue fournisseur / client en termes intelligibles pour ce dernier

Que veut le client? Lien utilisateur - analyste.

Les phases : analyse des besoins

Cf. CM UML/Diagrammes CU/Processus Unifié.

Cf. CM Documents/Cahier des Charges.

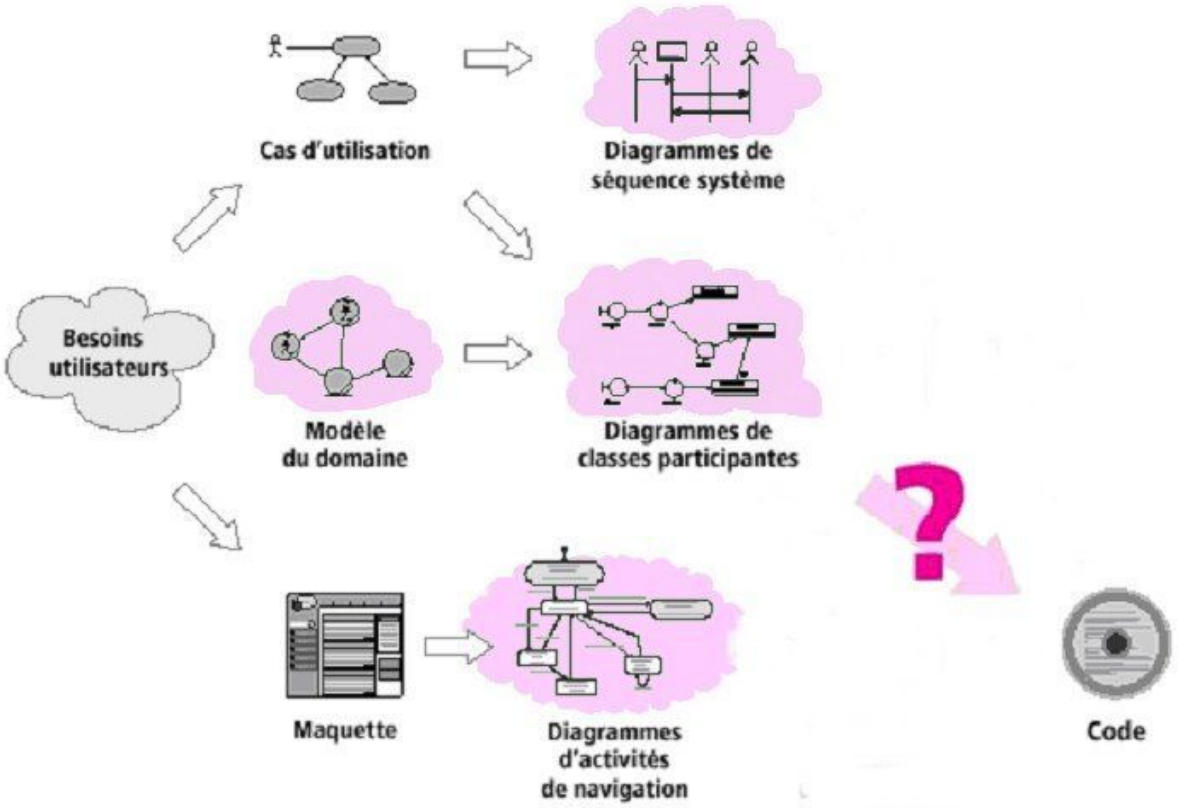
Les phases : spécification

Spécifier précisément les fonctionnalités attendues: le système comme une boîte noire, ses contours.

Modélisation de l'environnement dans lequel évoluera le logiciel. Modélisation des processus métiers du client, hors de toute considération technique. Modèle du système niveau métier. Nécessite des interaction avec le client / utilisateur, auquel il convient de les faire valider.

Que va-t-on faire pour le client? Lien analyste - architecte.

Les phases : spécification



Cf. CM UML.

Cf. CM Documents/Spécifications.

* * *

Les phases : conception globale

(Aka Analyse du système)

Approche statique: identifier les éléments intervenants et dans le système : fonctionnalités, structures et relations.

Approche dynamique: identifier les états par lesquels ils passent suivant certains événements.

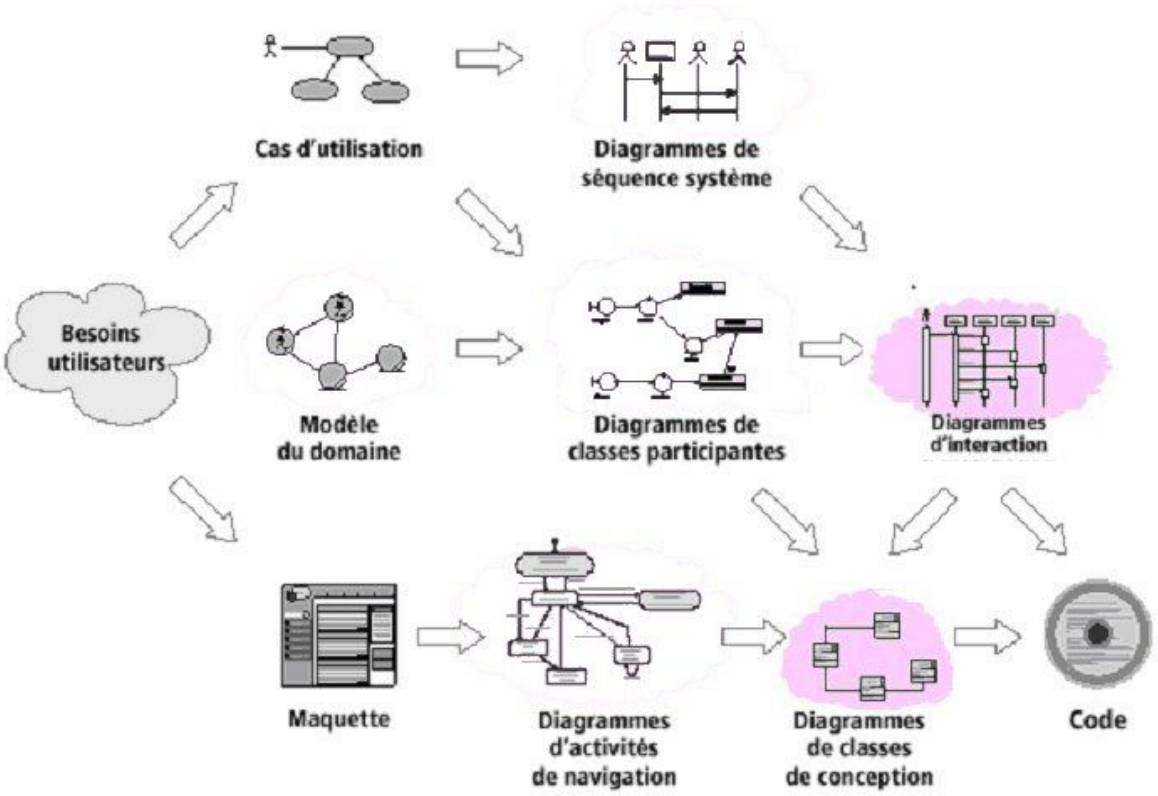
Modèliser de l'architecture logicielle du système.

Décomposition en packages et grands sous-systèmes.

Peut rester un support de discussion avec le client.

Lien architecte - développeurs.

Les phases : conception globale



Cf. CM UML

Cf. CM Documents/Dossier d'Analyse Fonctionnelle.

Les phases : conception détaillée

Comment faire le système.

Choix de patrons éventuels, choix techniques.

Décomposition en sous-systèmes : interfaces, APIs.

Permet l'organisation de l'implémentation.

Expertise informatique hors compréhension du client.

Cf. CM Conception

Cf. CM Documents/Rapport de Conception Détaillée.

Les phases : implémentation

Souvent trop de temps consacré au codage au détriment des phases d'analyse et de conception.

Savoir user de la réutilisation de composants, voire d'outils de génération de code (ex: mise en place automatique du squelette du code à partir du modèle système):

L'activité de développement est de plus en plus tournée vers la réutilisation de composants existants.

Les phases : vérification

Vérification de la robustesse et cohérence du des composantes du système en particulier dans le cas d'exception.

Idéalement testeur ≠ concepteur ou programmeur

Assurer un bonne couverture

Tests unitaires, tests d'intégration.

Est-ce que le code fonctionne?

Cf. CM Tests, CM Documents/Dossier de tests.

Les phases : validation

Vérification la conformité aux spécifications et donc valider la conformité à ce qu'on voulait faire ; ce qu'on avait compris des besoins utilisateurs.

Tests systèmes.

A-t-on fait ce qu'on s'était fixé ?

Cf. CM Tests, CM Documents/Dossier de validation.

Les phases : recette

Validation avec le client d’un rendu final ou intermédiaire.

Tests d'acceptation.

Le client est-il satisfait ?

Cf. CM Tests.

Cf. CM Documents/Dossier de recette.

Les phases : maintenance

- Correction des erreurs du système
- Demande d'évolution (modification de l'environnement technique, nouvelle fonctionnalité)

Facteurs de qualité essentiels:

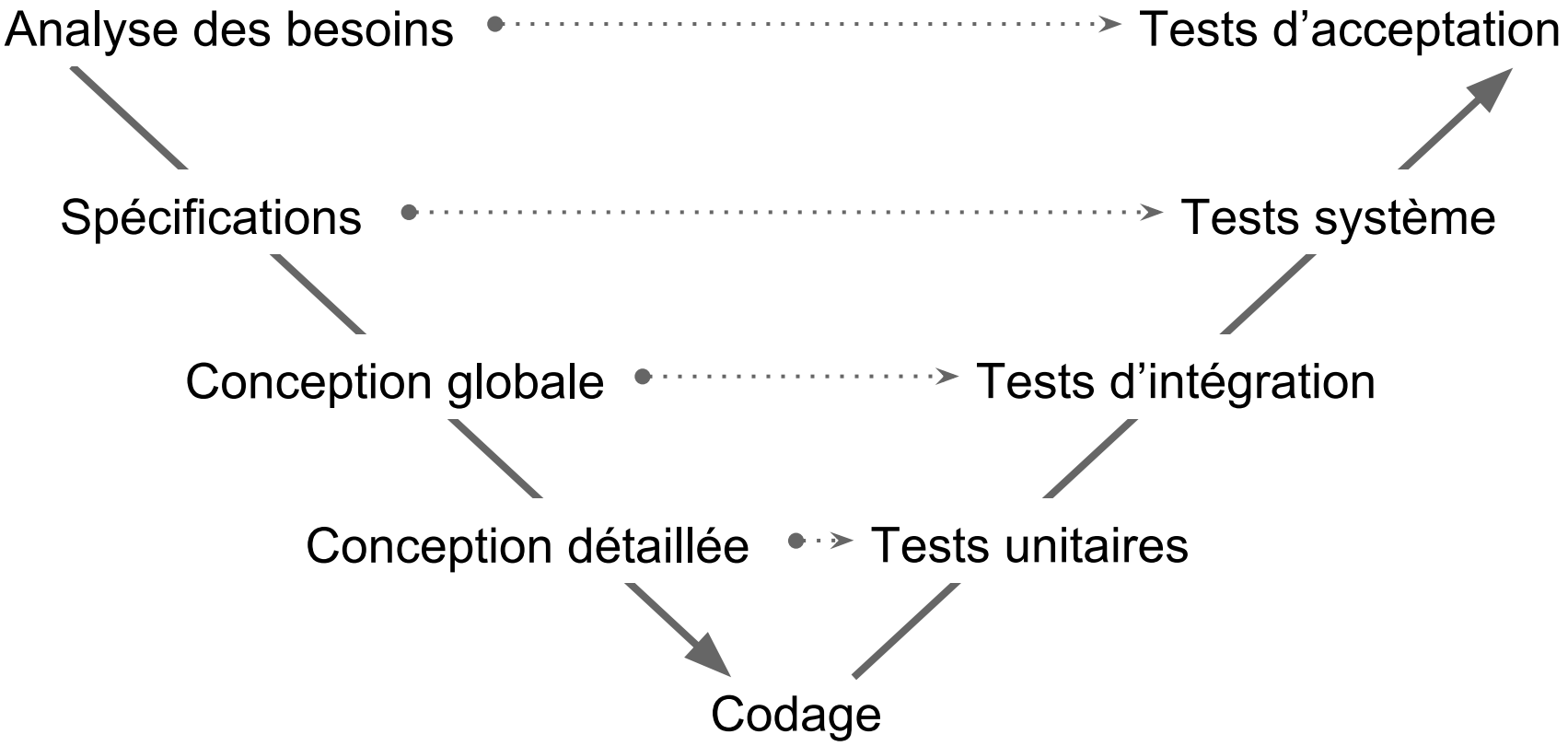
- Corrections : robustesse
- Evolutions : modifiabilité, portabilité

70 % de l'effort (problème de pratiques? ou critère de succès: nature du logiciel...)

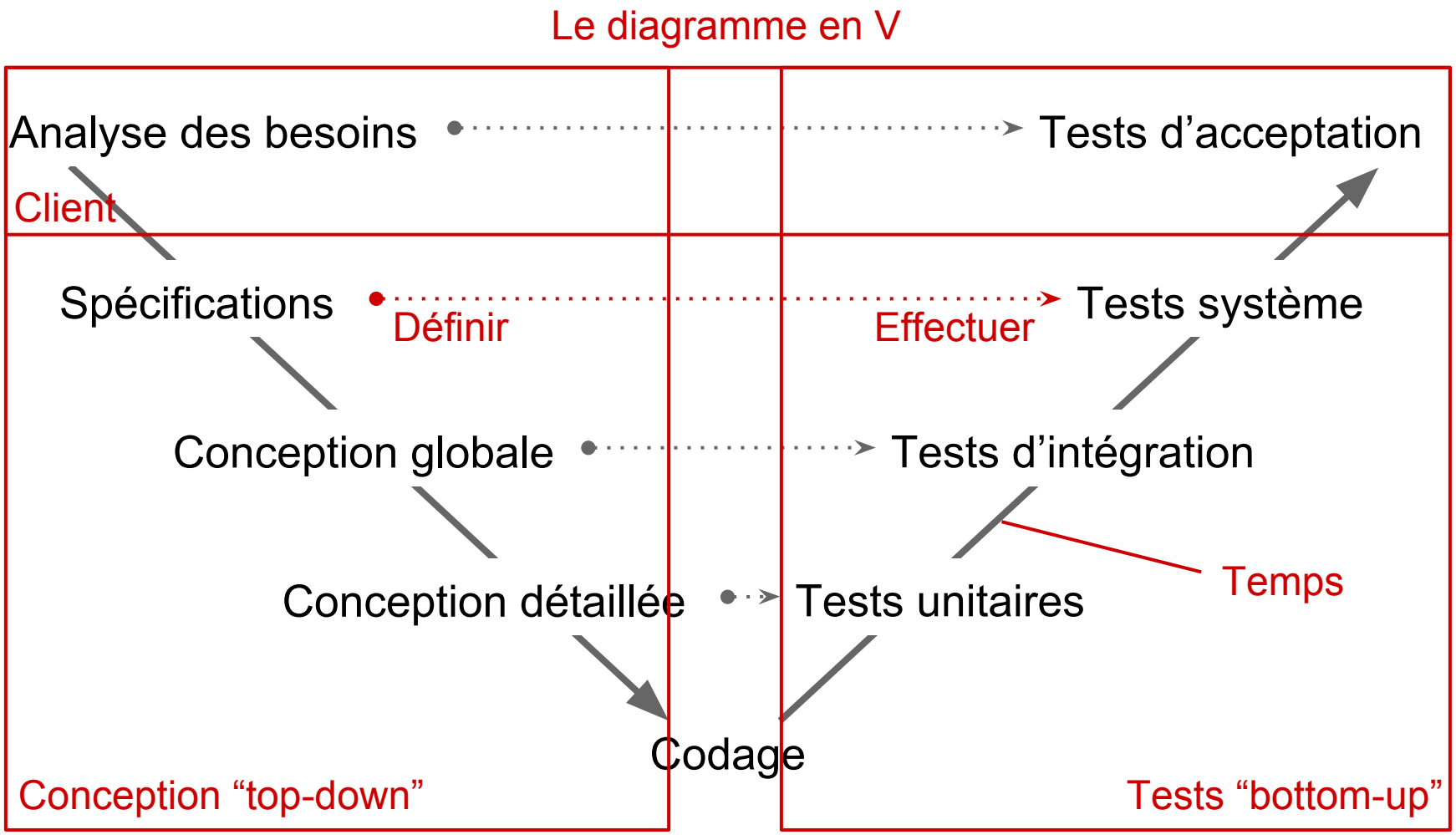
* * *

Méthode en V

Méthode en V


Analyse des besoins
Tests d'acceptation
Spécifications
Tests système
Conception globale
Tests d'intégration
Conception détaillée
Tests unitaires
Codage

Méthode en V


Le diagramme en V
Analyse des besoins
Tests d'acceptation
Client
Spécifications
Définir
Effectuer
Tests système
Conception globale
Tests d'intégration
Temps
Conception détaillée
Tests unitaires
Codage
Conception "top-down"
Tests "bottom-up"

* * *

Méthode en V : avantages

- Étapes de vérification et validations intermédiaires
- Points de mesure concrets de l'avancement du travail
- Favorise la décomposition fonctionnelle de l'activité
- Modèle éprouvé très utilisé pour de grands projets
- Existence d'outils supports
- Éventuellement surévaluation coût
- Paiement au forfait

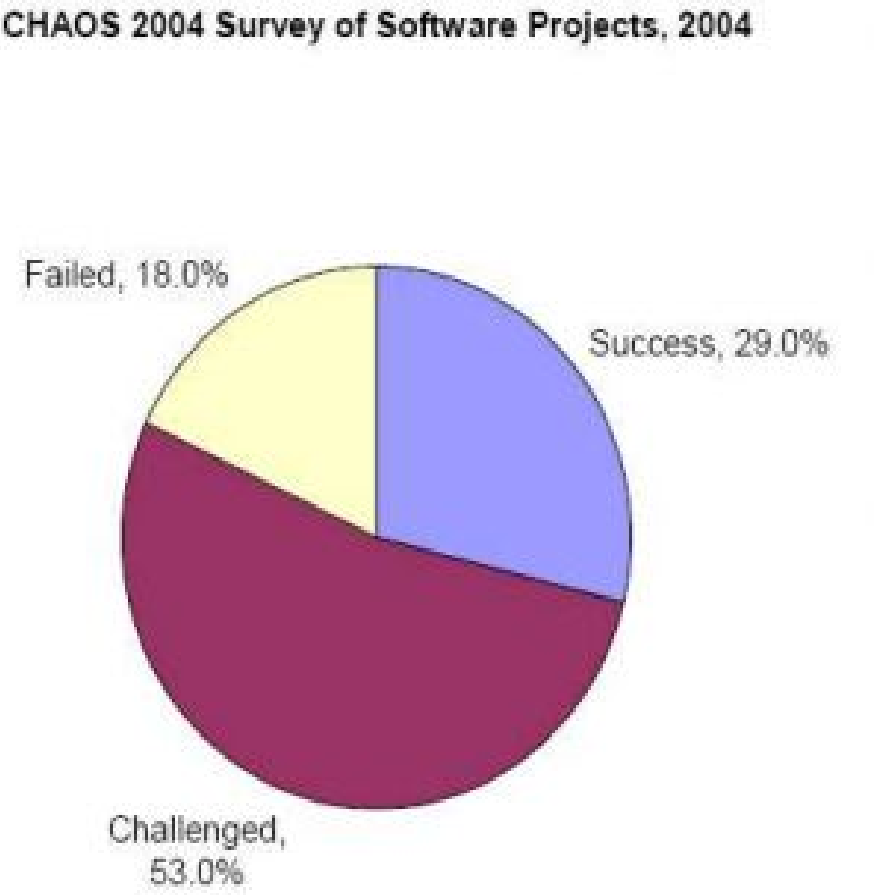
Idéal quand les besoins sont bien connus et figés, quand l'analyse et la conception sont claires.

Méthode en V : désavantages

- Prédominance de la documentation sur l'intégration : validation tardive du système global.
- Les validations intermédiaires n'empêchent pas la transmission des insuffisances.
- Peu d'adaptabilité: comme de construire un pont.
- Maintenance non intégrée : logiciel jetable?
- Probablement sous-évaluation coût (concurrence)

A éviter en situation dynamique, qui nécessite d'exploiter la malléabilité du logiciel.

Risques : importance



Risques : nature

Risques : nature

Risques financiers (cher, long, pas sûr)

Risques d'incompréhension:

- Manque d'implication des utilisateurs
- Besoins client incomplets
- Besoins client changeants

Risques techniques (perte d'expertise, complexité, incompatibilité…)

Risques commerciaux (placement sur le marché, concurrence, ...)

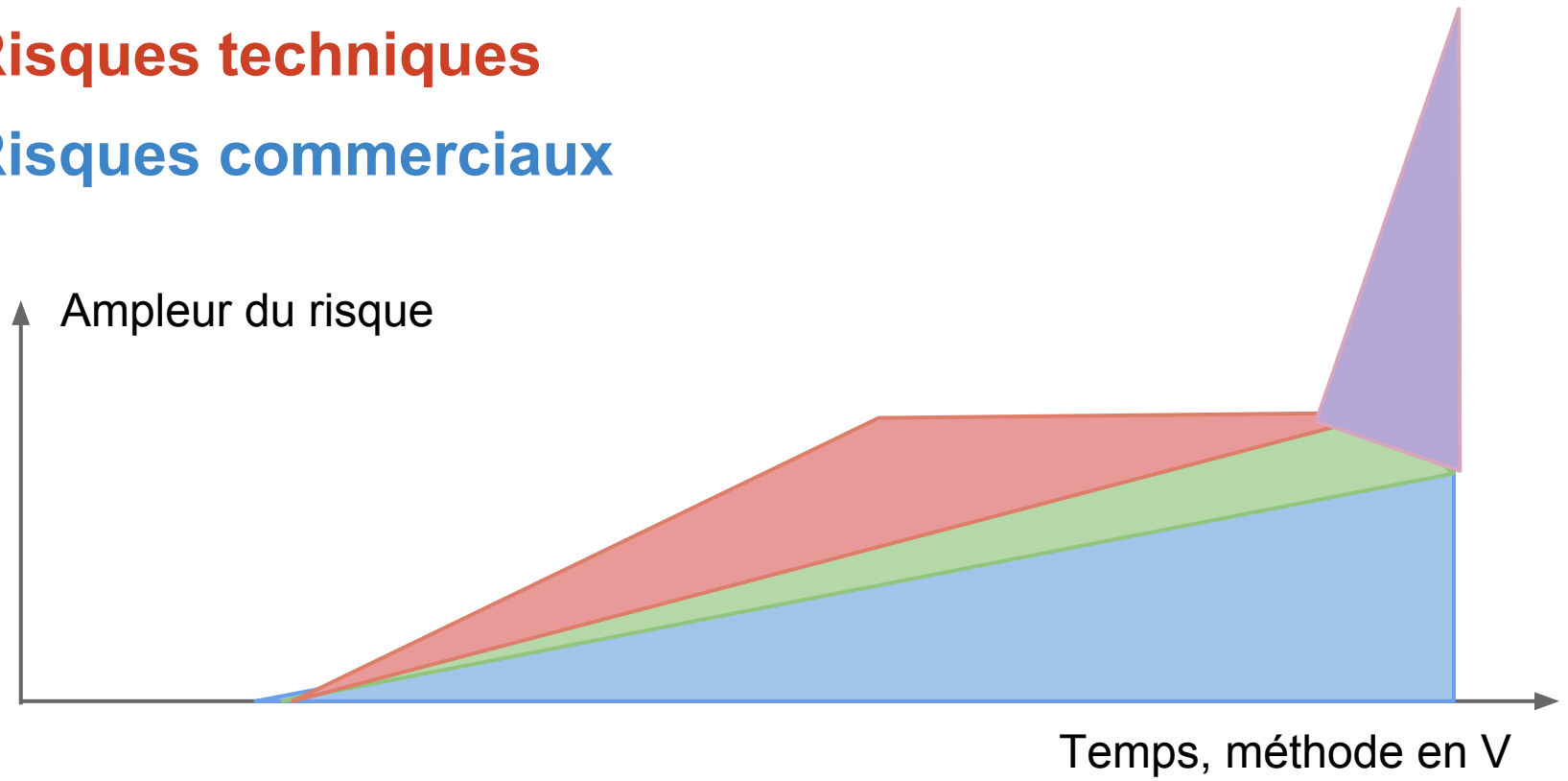
Risques : nature et répartition

Risques financiers

Risques d'incompréhension

Risques techniques

Risques commerciaux



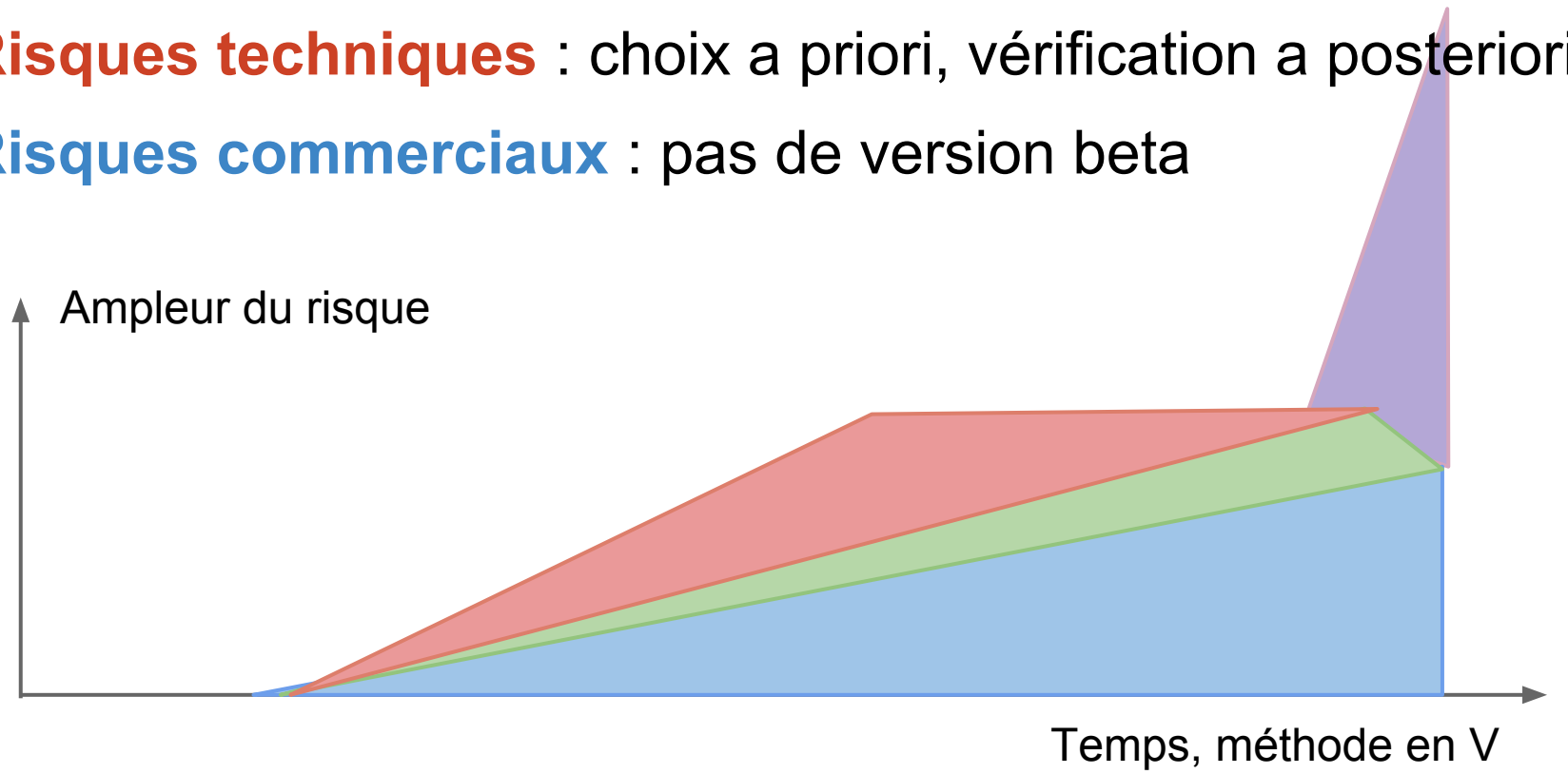
Risques : nature et répartition

Risques financiers : pas de possibilité d'interrompre

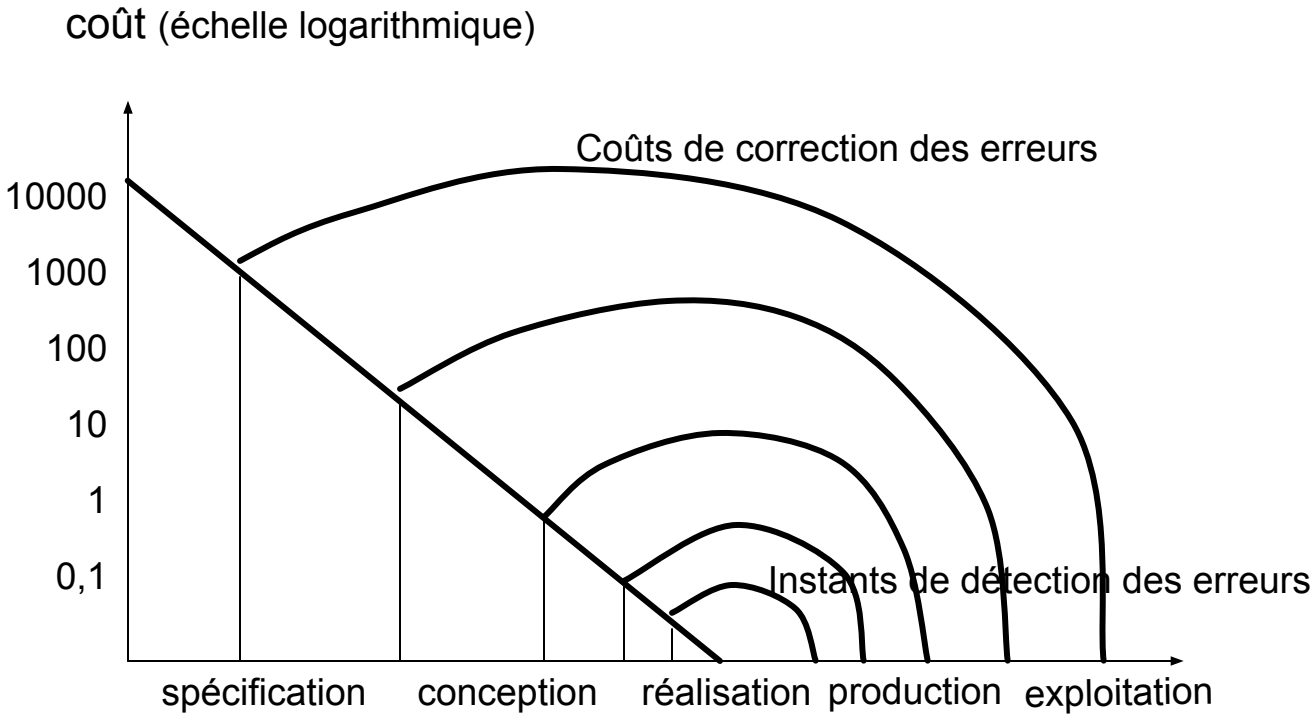
Risques d'incompréhension : validation trop tardive

Risques techniques : choix a priori, vérification a posteriori

Risques commerciaux : pas de version beta



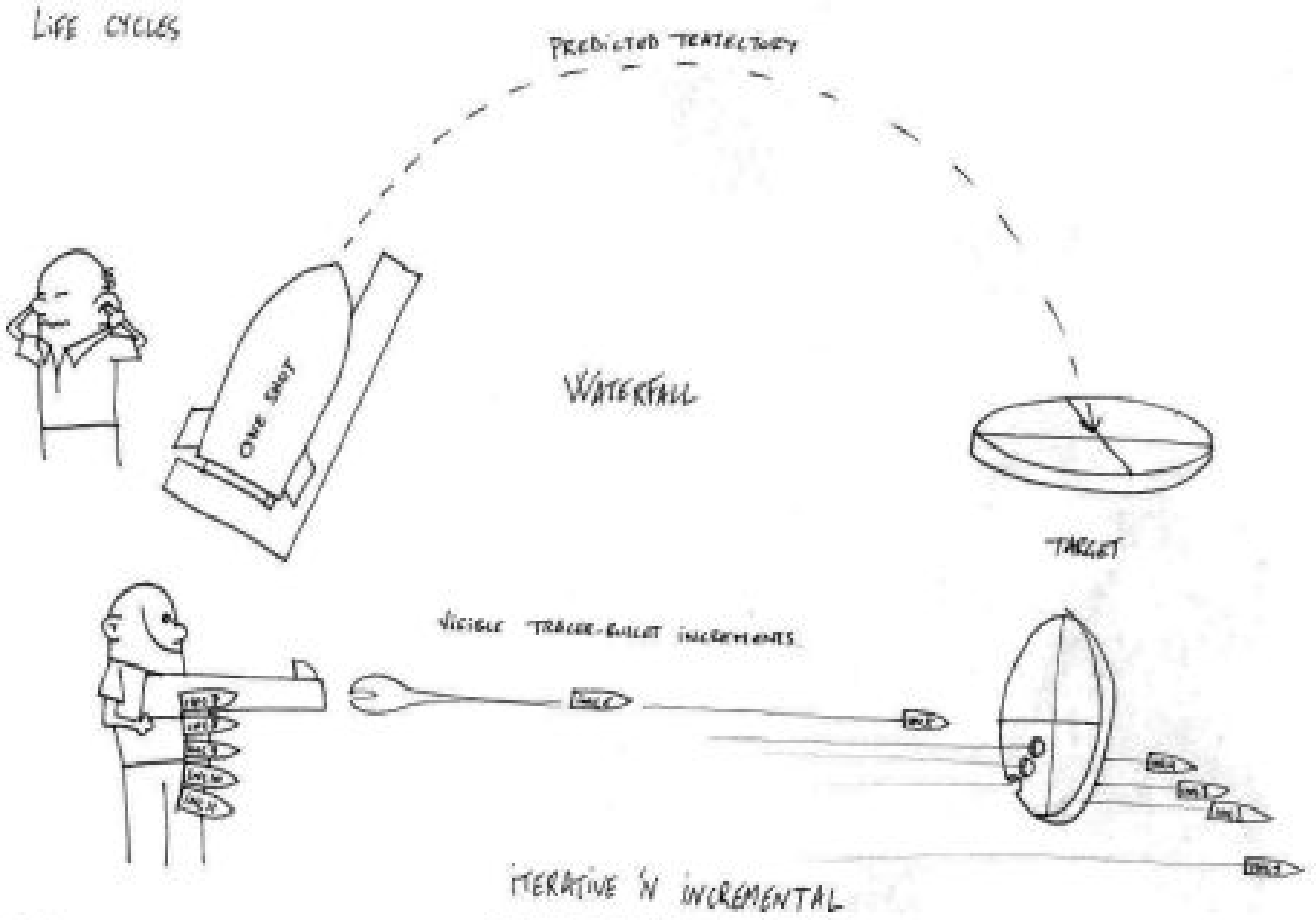
Risques : répartition et coûts



Mieux vaut se heurter aux difficultés le plus tôt possible.

Méthodes adaptives

Méthodes adaptives versus en V



Méthodes adaptives versus en V

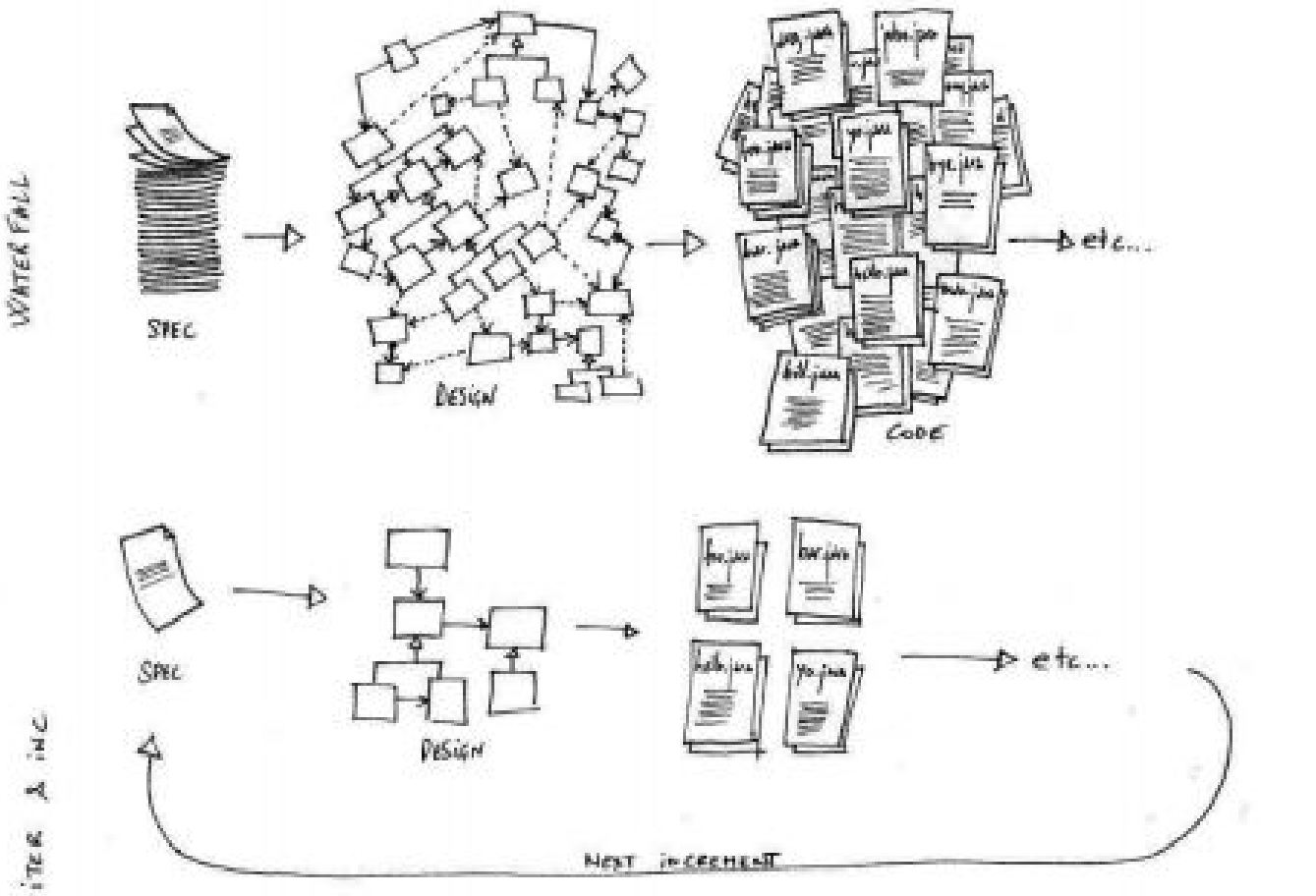
Méthodes adaptives, synonymes:

- Itératives, incrémentales
- Agiles, adaptables
- Cycle / méthode / diagramme en spirale
- iter & inc

Méthode en V, synonymes:

- Cycle / méthode / diagramme en V
- Méthodes prédictives
- Différent, mais proche du modèle en cascade (waterfall)

Méthodes adaptives



Méthodes adaptives : Vocabulaire

Itération

- L'application est divisée en plusieurs applications plus petites: "divide and conquer".

- Permettent la validation régulière par les utilisateurs.
- Le contenu d'une itération *n+1* comporte de nouvelles fonctionnalités et la prise en compte des améliorations de l'itération *n*

Méthodes adaptives : Vocabulaire

Itérations

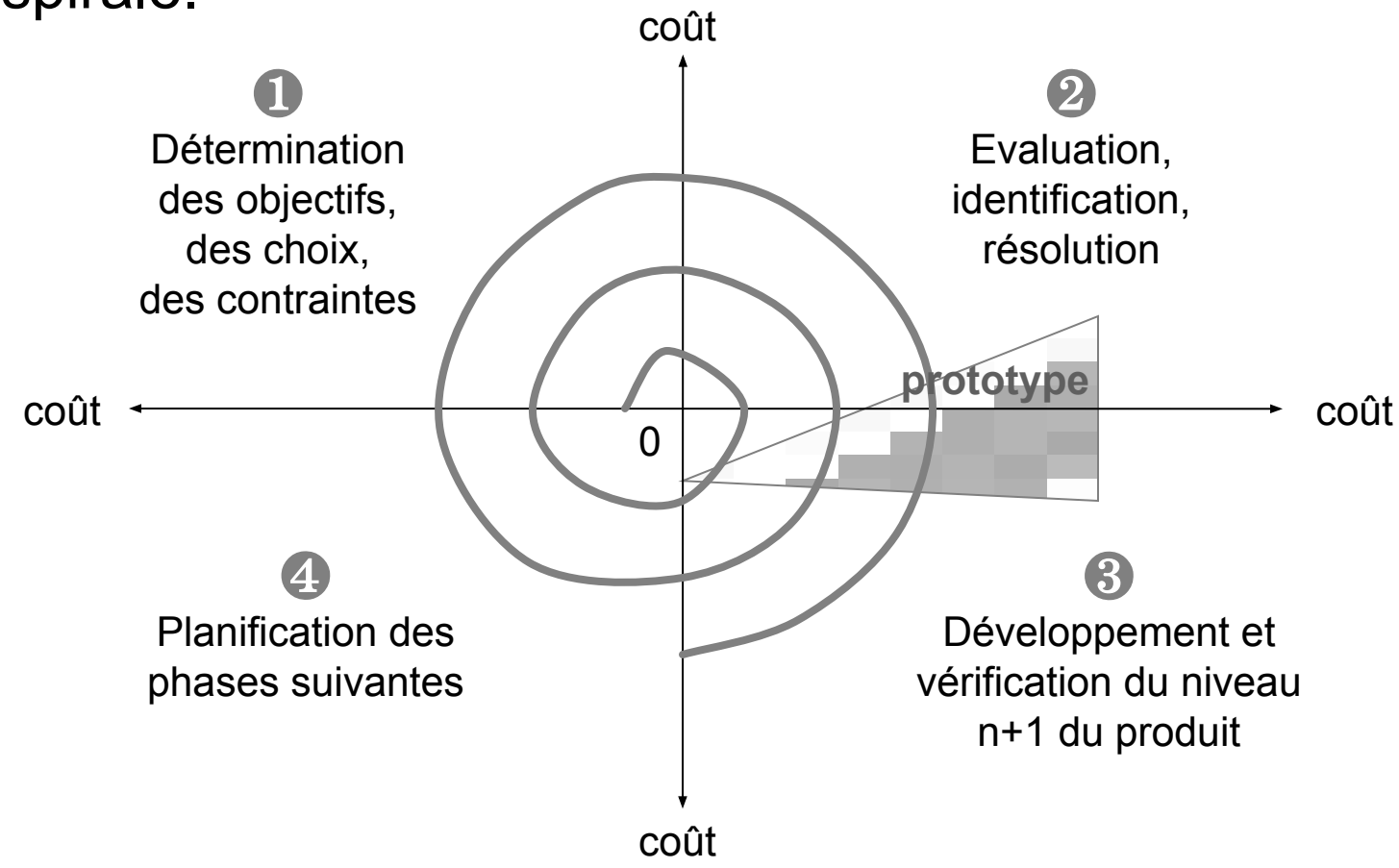
But: rectifier au plus tôt les erreurs et même prendre en compte l'évolution des besoins.

Prototypage

Livraison d'un exécutable permettant une validation concrète.

Méthodes adaptives

La spirale.



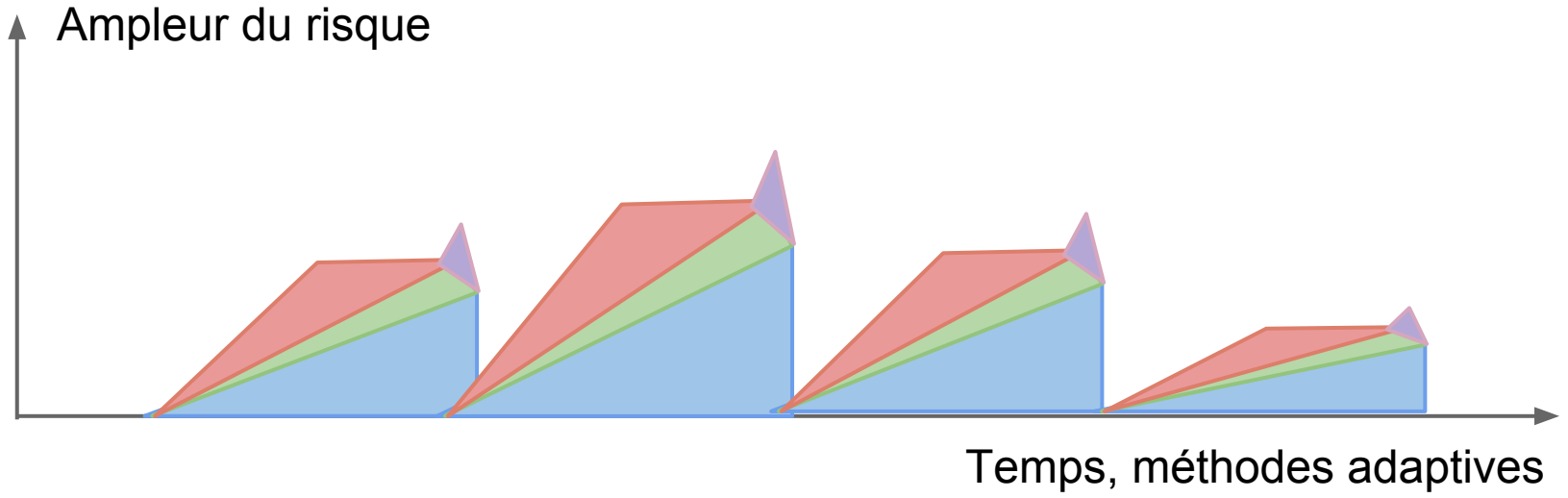
Méthodes adaptives : risques

Risques commerciaux : version beta, veille technologique

Risques d'incompréhension : validation régulière

Risques techniques : incrémentaux, évolutifs

Risques financiers : prototype déjà là



Méthodes adaptives : risques

"*Un tiens vaut mieux que deux tu l'auras.*"

"*Un système complexe qui fonctionne a toujours évolué à partir d'un système simple qui a fonctionné… Un système complexe construit à partir de zéro ne fonctionne jamais et ne peut être rapiécé pour qu'il fonctionne. Il faut tout recommencer, à partir d'un système simple qui fonctionne.*"

-- John Gall.

Méthodes adaptives : Planification

Constituer un **product backlog**, liste du travail à accomplir.

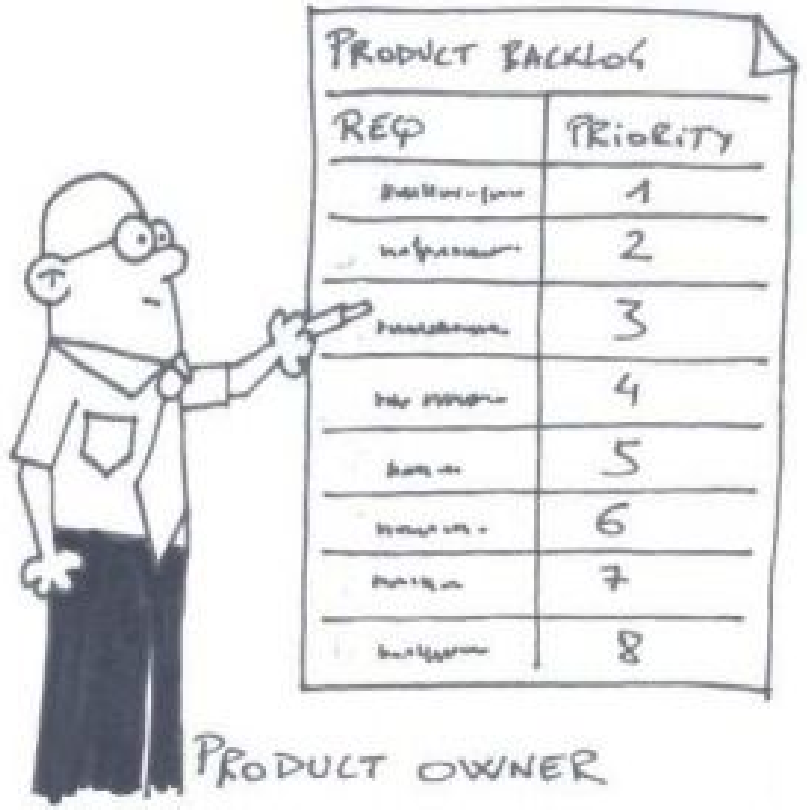
Évaluer chaque item en:

- Valeur (par et pour le client): indisp. / significatif / joli.
- Charge (par et pour le développeur)
- Risques (nature, dépendances)

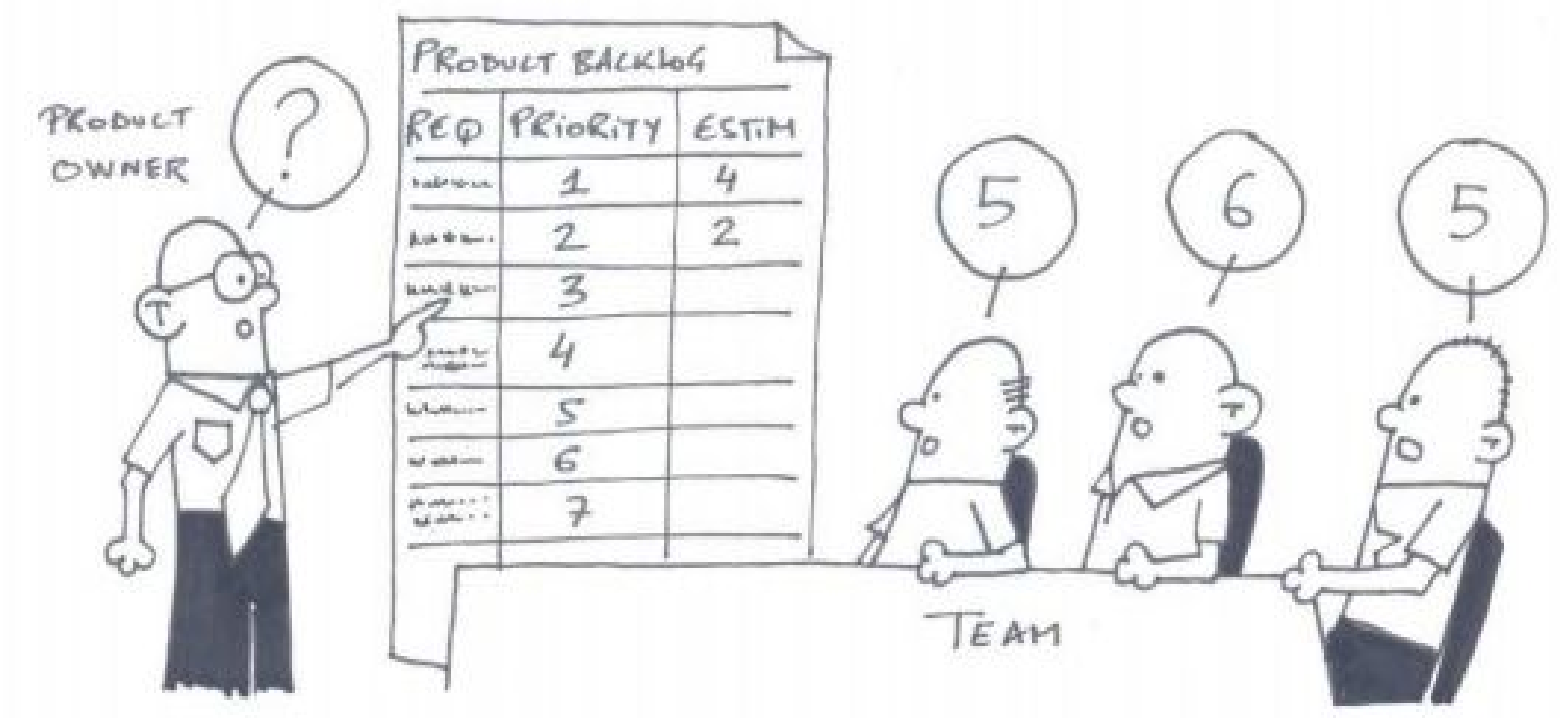
En réunion.

- Avec le client ou son représentant.
- Avec les développeurs.
- En mode exploratoire, dynamique.

Méthodes adaptives : Planification



Méthodes adaptives : Planification



Méthodes adaptives : Planification

Remarques.

Plus les items sont prioritaires, plus ils sont:

- Sous-divisés en tâches pour une meilleure évaluation
- Décrits sous diverses formes (CU, Scénarios…)

Règle des 20% de travail - 80% de bénéfices.

Ex. d'items:

- Features, bugs
- Conception, codage, tests
- Acquisition de connaissance

Méthodes adaptives : Planification

Constituer un **sprint backlog** en une itération de durée fixe.

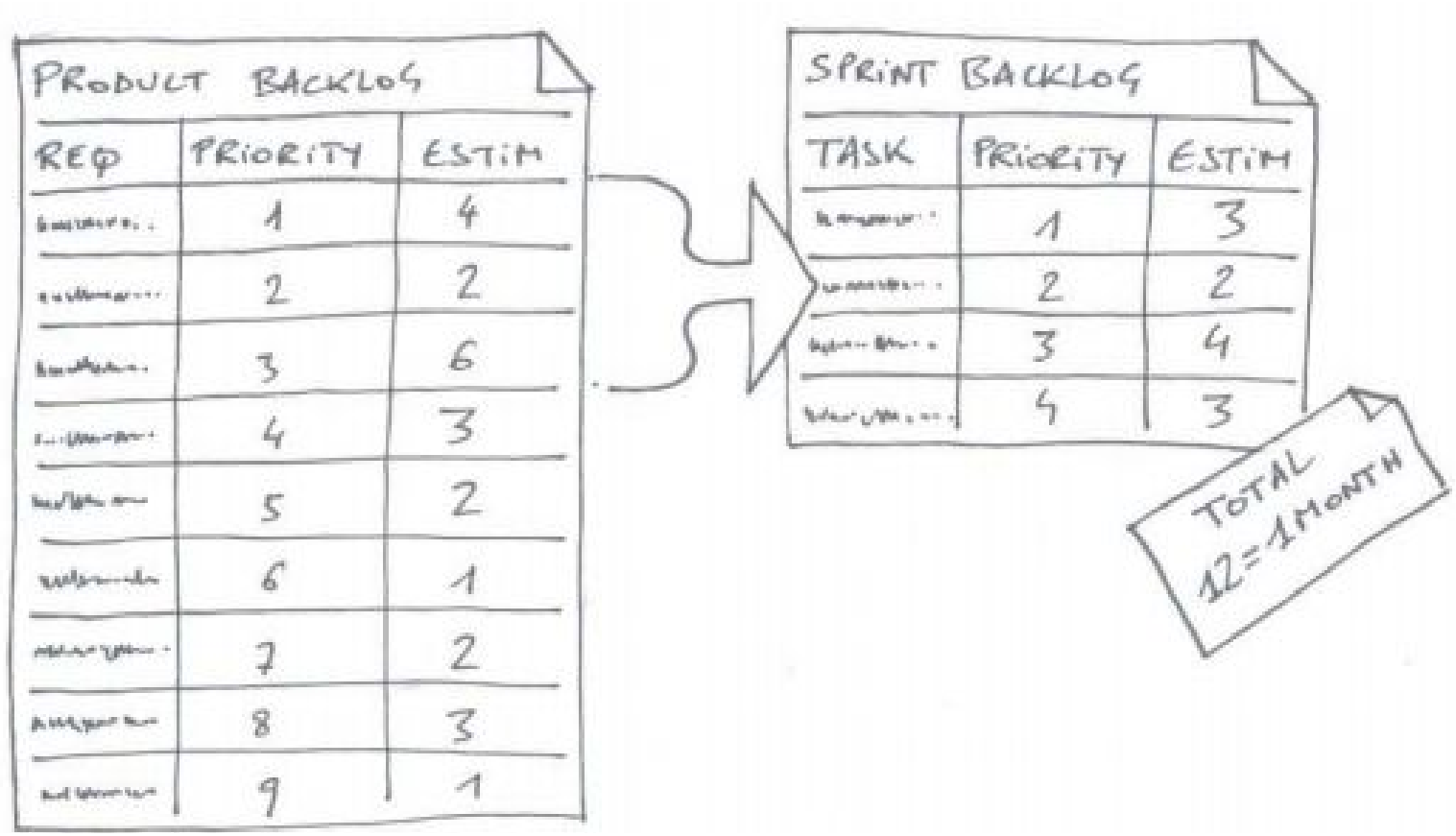
C'est une sous-liste d'items, fonction de:

- la valeur (prototyper) : laisser le client piloter.
- la charge (faisabilité)
- les risques (à évacuer tôt)

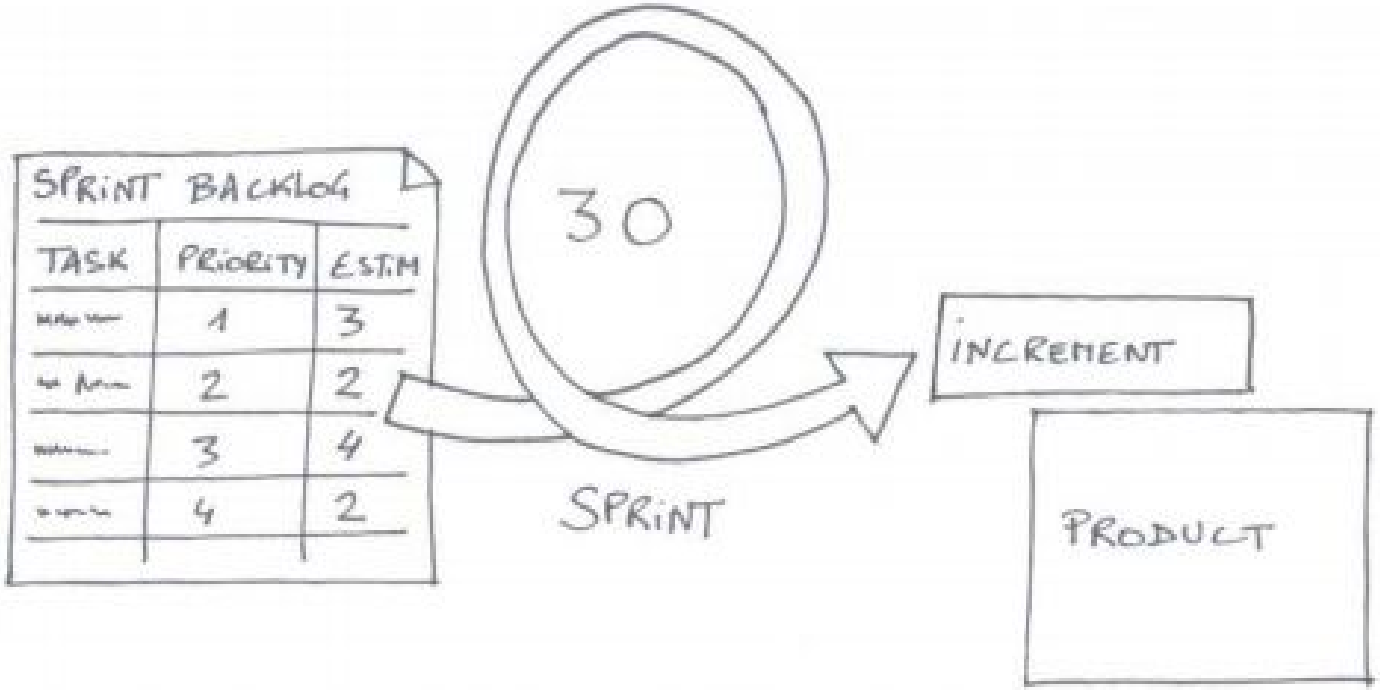
En réunion.

- Chacun choisit ses tâches: se donner confiance.
- Identifier les dépendances de développement.
- Fournit une référence permettant d'évaluer l'itération, et de mieux planifier la prochaine.

Méthodes adaptives : Planification



Méthodes adaptives : Planification



Méthodes adaptives : XP

- **Un représentant du client** est intégré à plein temps pour une réactivité optimale en termes de définition des besoins et validation des livraisons.
- **Programmation en binômes** pour partage de l'expertise, meilleure appropriation du code, revue de code permanente.
- **Tests unitaires avant codage**, par tous.
- **Intégration et refactoring en continu.**
- **Livraisons fréquentes** pour accélérer la convergence du produit par rapport aux besoins des utilisateurs.

Méthodes adaptives : XP

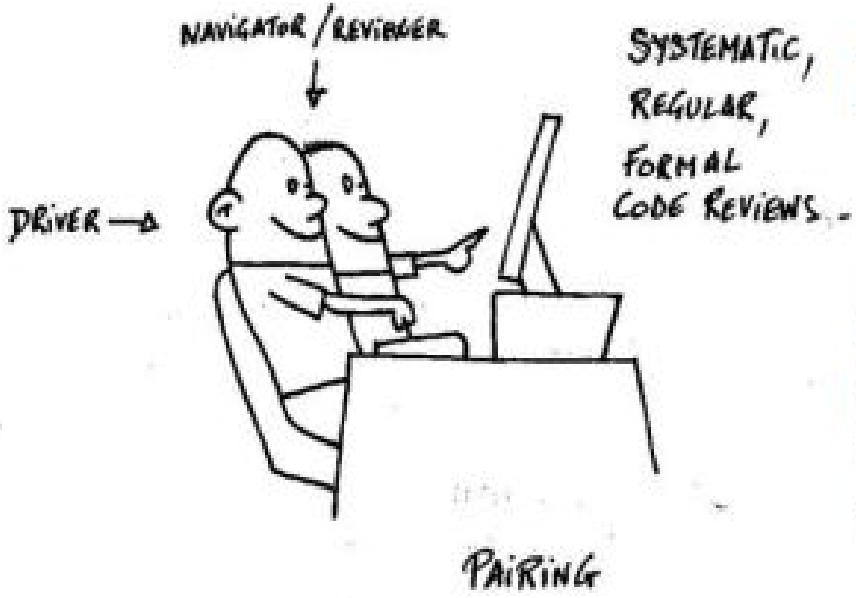
Avantages.

Dégraissé, efficace, peu exposé au risque:

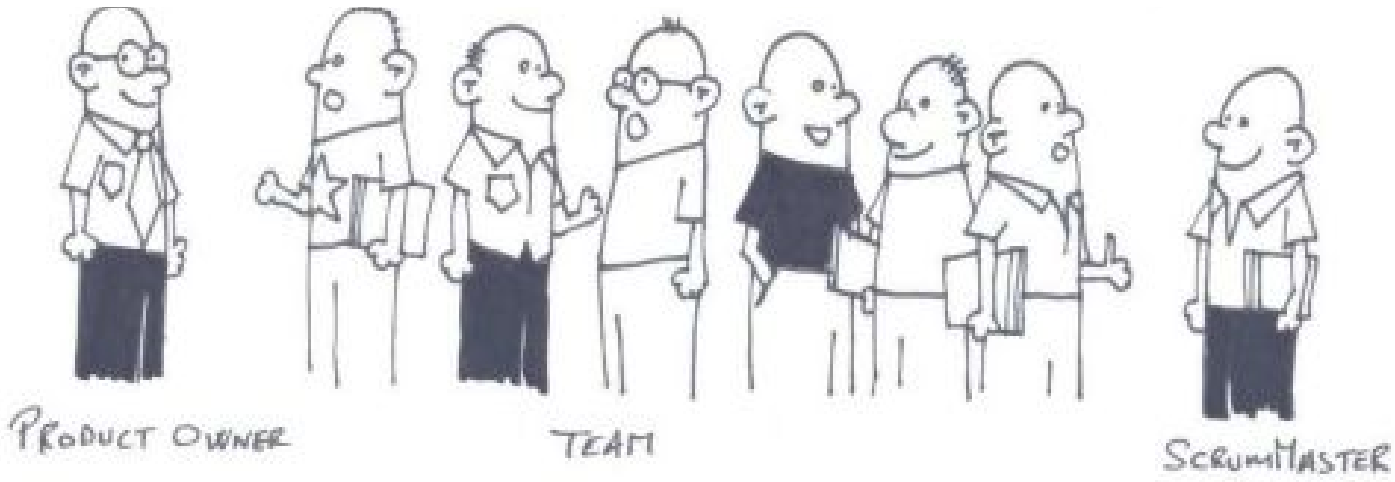
- Défaillances? Tests routine par routine et client.
- Incompréhension besoin? Le client est là.
- Fausse fonctionnalité? Priorité à la valeur.
- Changement des besoins? Raccourcir les cycles.
- Dépassement de délais? cycle court.
- Abandon de projet? petite livraison.

Équipes de petite et moyenne taille, confrontée à des besoins vagues ou changeants.

Méthodes adaptives : XP



Méthodes adaptives : Scrum



Méthodes adaptives : Scrum

Rôle: Product owner.

En interne :

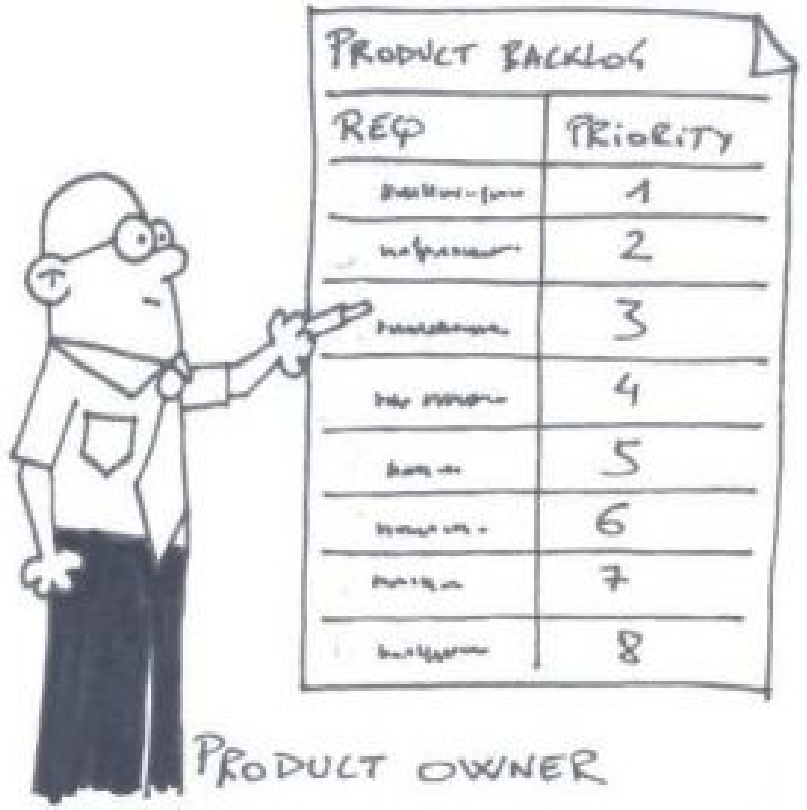
- Incarne, ou est le client.
- Fait évoluer le “product backlog”.
- Évalue la valeur client, les priorités.

En externe:

- Organise les réunions client
- Les livraisons
- Les négociations

Empathique, communique au niveau de détail adéquat.

Méthodes adaptives : Scrum



Méthodes adaptives : Scrum

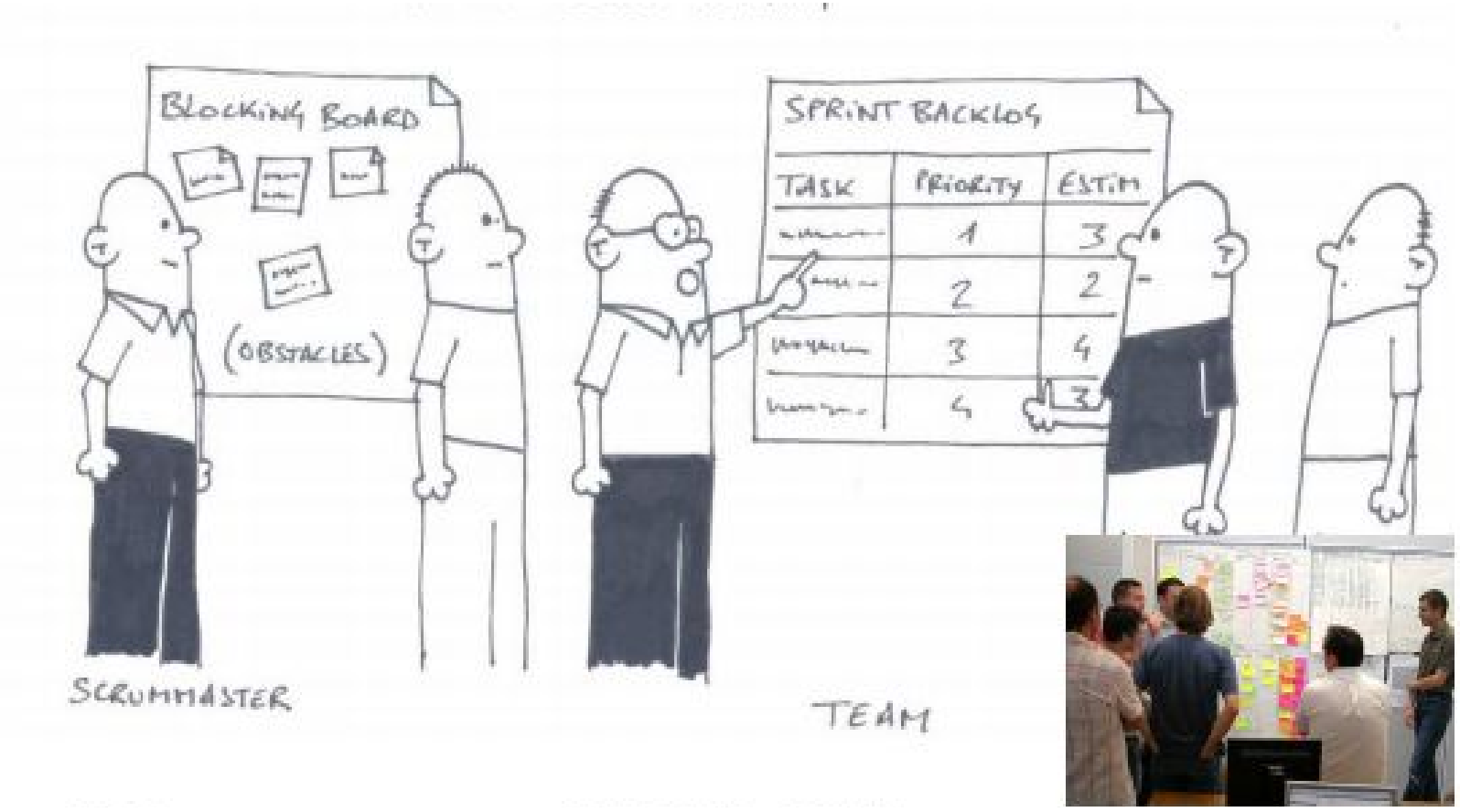
Rôle: Scrum master.

Protège le processus Scrum des obstacles qu'il peut rencontrer. Responsable de la mise en place les règles du Scrum.

N'est pas un manager ou chef de projet pour éviter :

- Conflit d'intérêt
- Remise en cause de l'autorité

Méthodes adaptives : Scrum



Méthodes adaptives : Scrum

Rôle: Development team.

4-8 personnes de compétences mixtes, auto-organisée.

Rôle optionnel: Hooker.

Remontée des bugs dans le backlog.

Rôle optionnel: Mr Freeze.

Responsable de l’arrêt du développement pour livraison.

Méthodes adaptives : Scrum

Sprint.

Itération durée fixée: 1 à 4 sem., typiquement 2.

But : aboutir à un “potentially shippable increment”,

c’est-à-dire un incrément potentiellement livrable.

Definition of Done.

Critère commun de complétude d’un item du backlog.

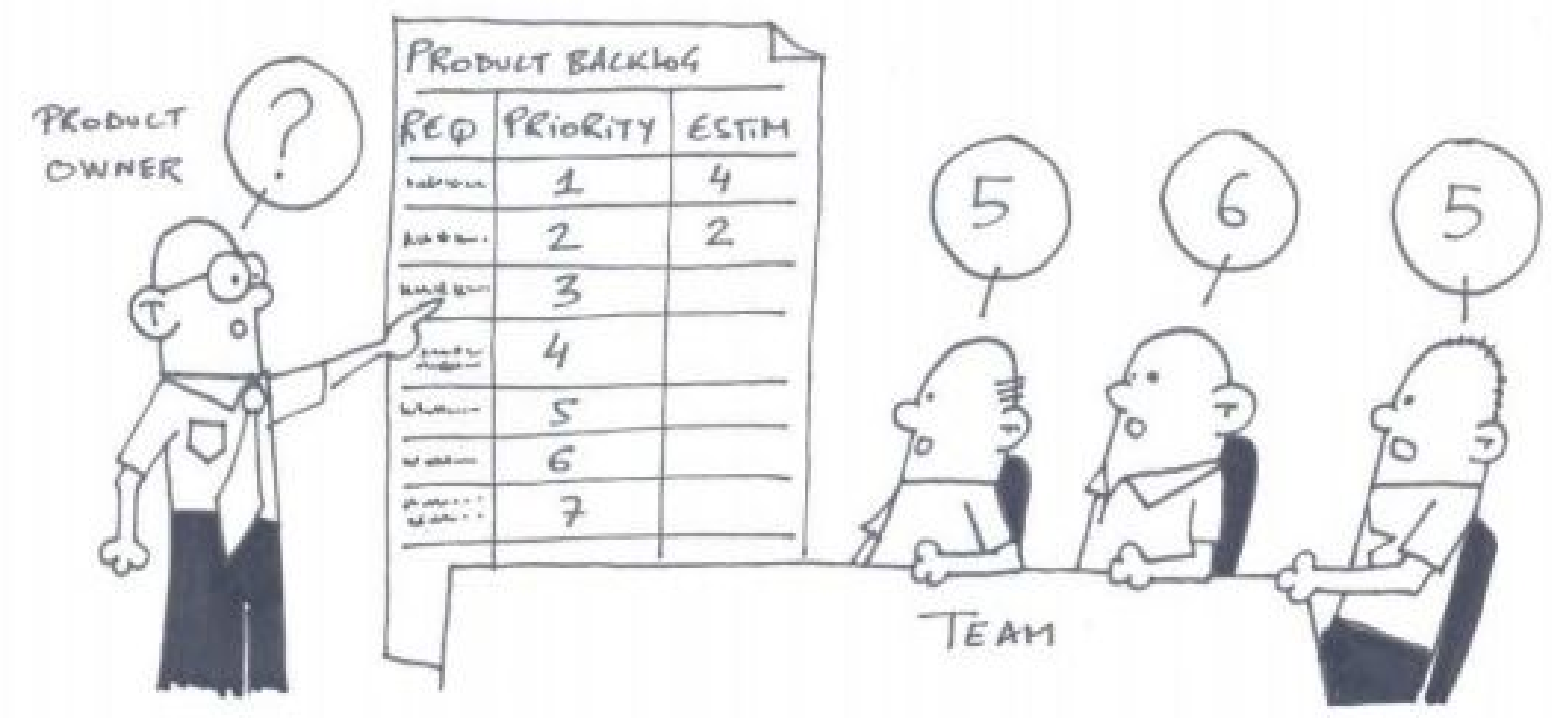
Ex: tests de non-régression passés. Ex. Mario

Méthodes adaptives : Scrum

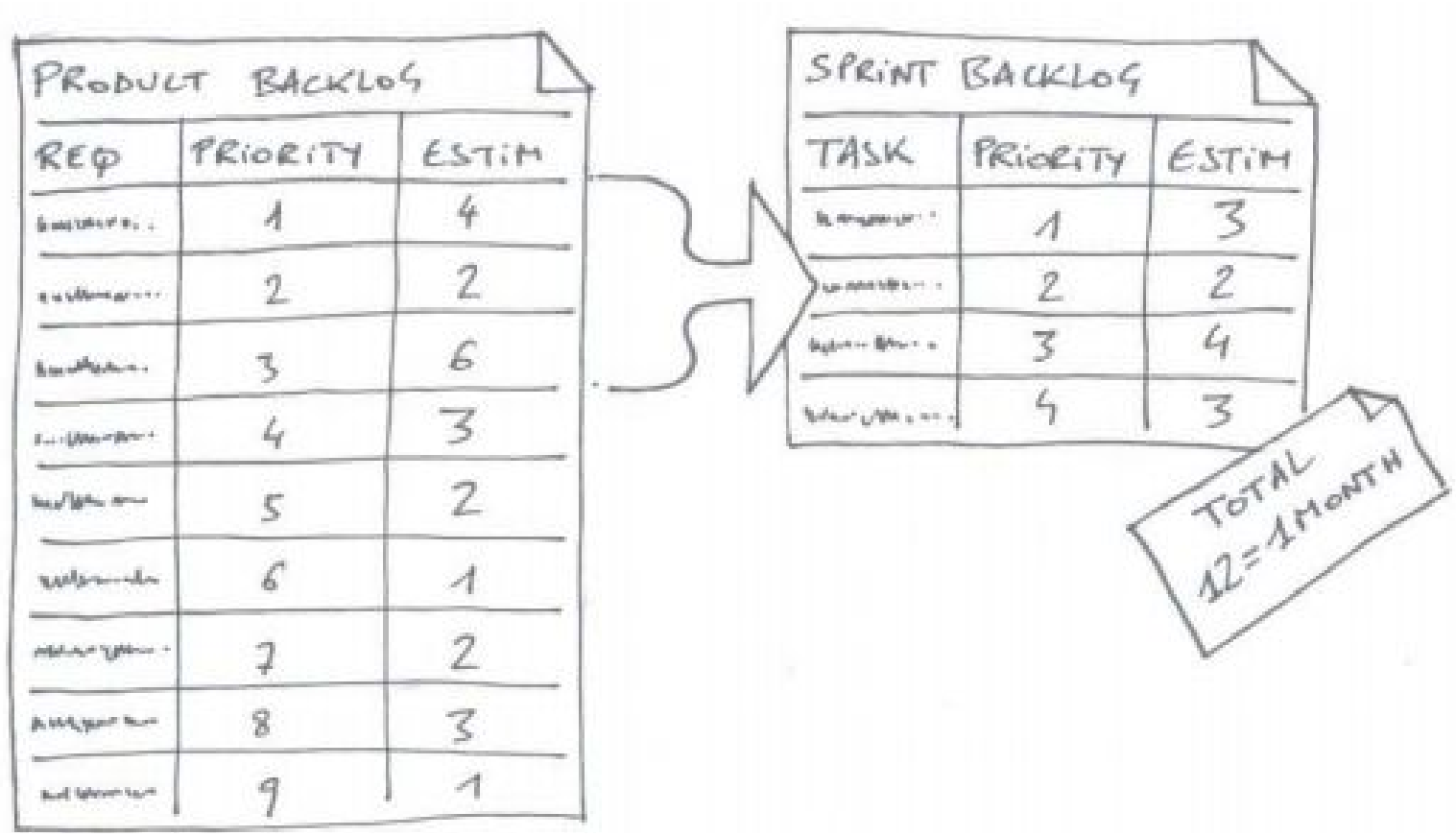
Sprint planning meeting (<2-8h)

- Priorétiser le product backlog
- Constituer le sprint backlog
- “Signing-up to tasks”: auto-organisation.

Méthodes adaptives : Scrum



Méthodes adaptives : Scrum

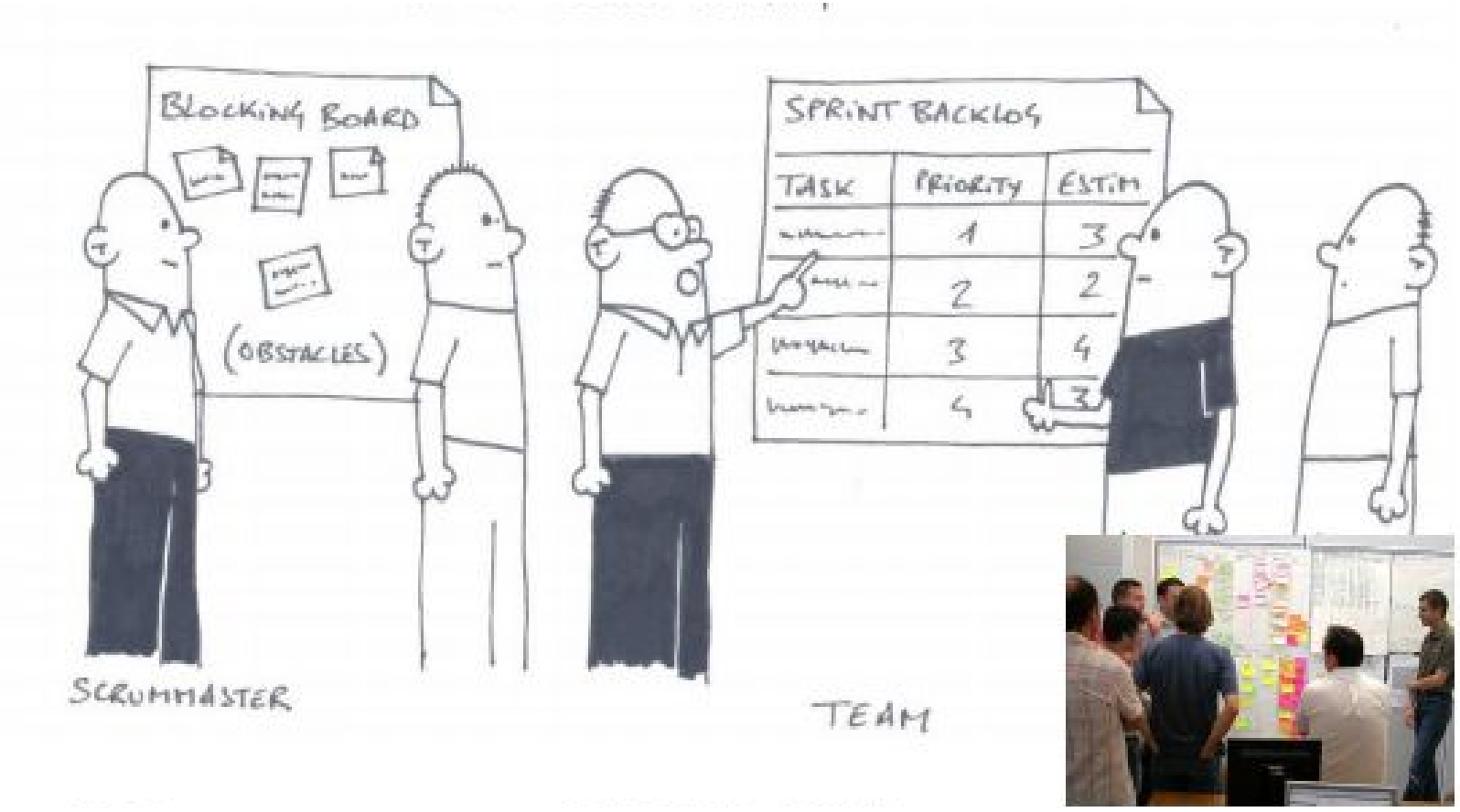


Méthodes adaptives : Scrum

Daily scrum meeting (<15mn)

- Même heure (exacte même si absents)
- Même lieu
- Fait hier, à faire aujourd'hui?
- Obstacles? A gérer par le scrum master.

Méthodes adaptives : Scrum



Méthodes adaptives : Scrum

Méthodes adaptives : Scrum

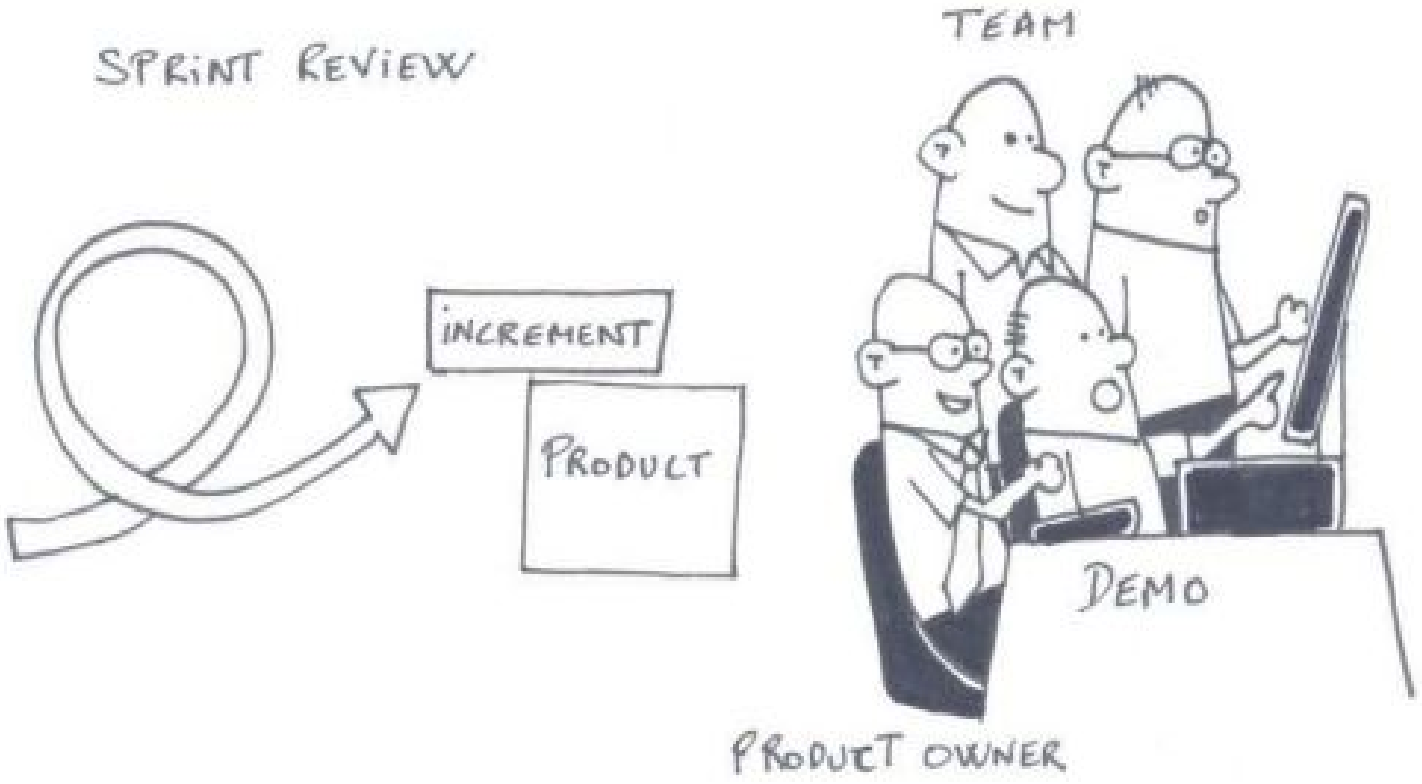
Sprint review meeting (<1-4h)

- Bilan
- Démonstration

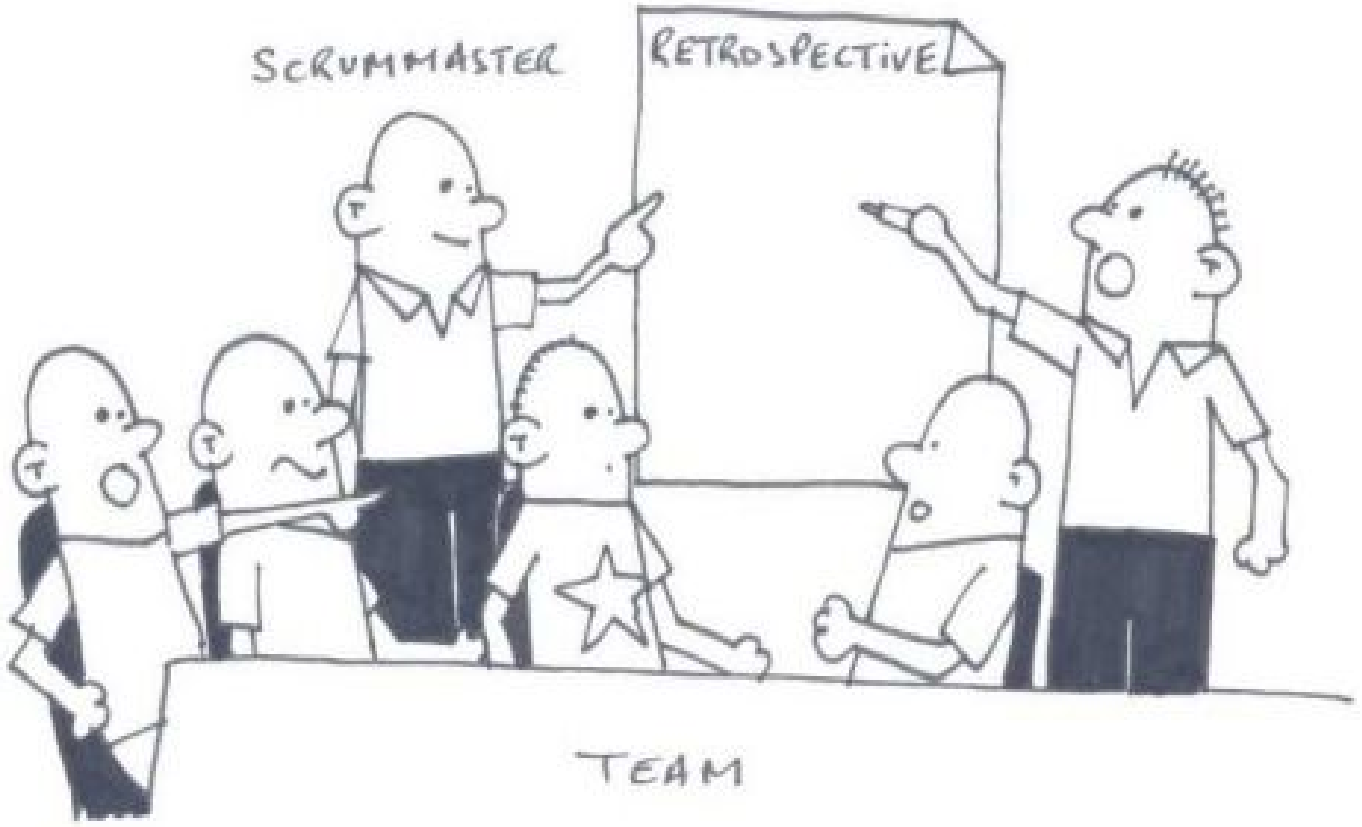
Sprint retrospective meeting (<1-3h)

- Organisée par le scrum master
- Qu’est-ce qui s’est bien passé? Que peut-on ameillorer?

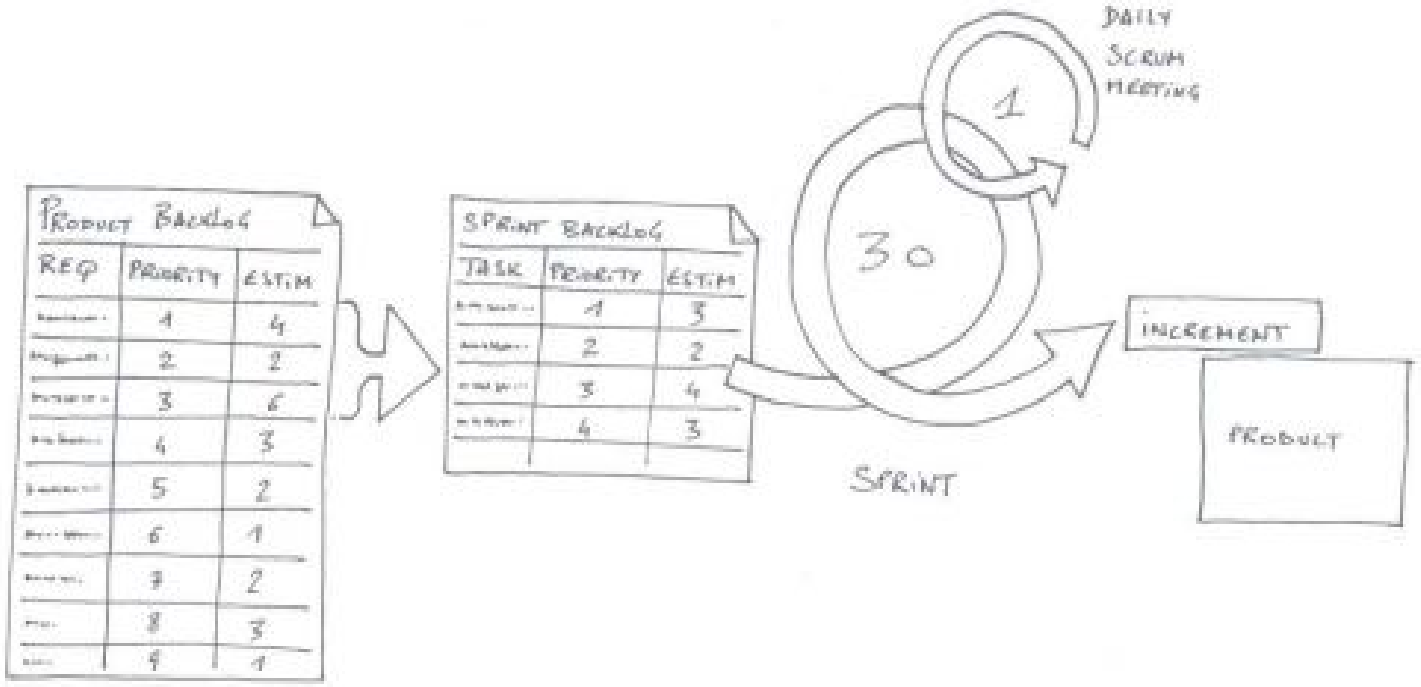
Méthodes adaptives : Scrum



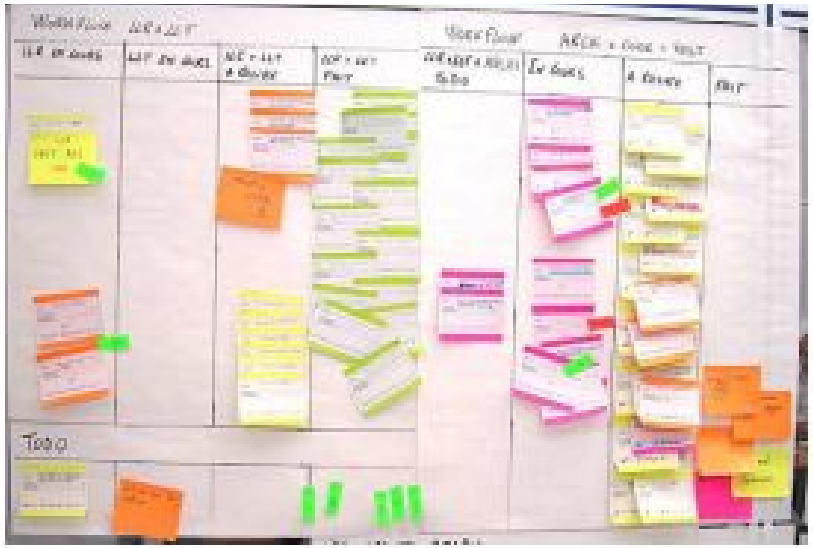
Méthodes adaptives : Scrum



Méthodes adaptives : Scrum



Méthodes adaptives : Task board *



Méthodes adaptives : Task board *

Outils gratuits :

- Trello.com
- Kanban.com
- https://taiga.io
- IceScrum

Méthodes adaptives : Task board

Product backlog

Post-its par thèmes

Sprint backlog

Une ligne et un post-it pour une user story.

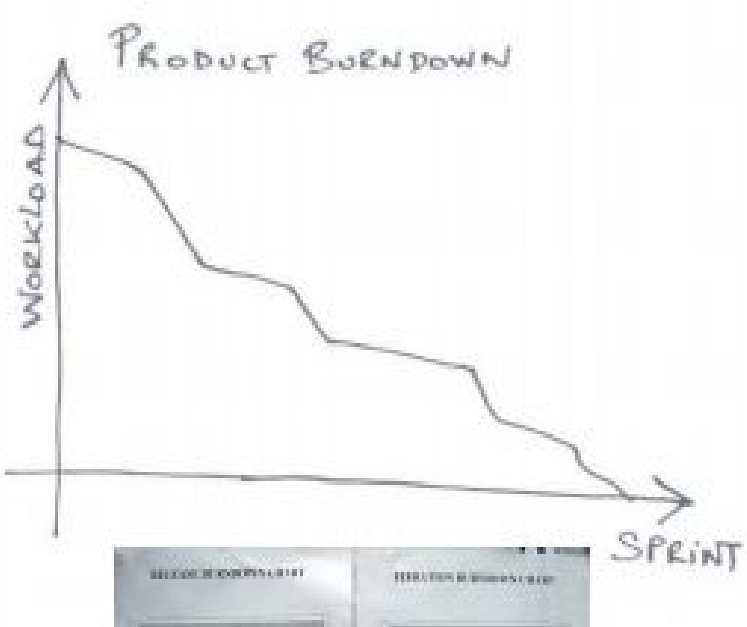
Un train de post-it par tâches qui la constituant, avançant le long de TODO, en cours, à relire, fait.

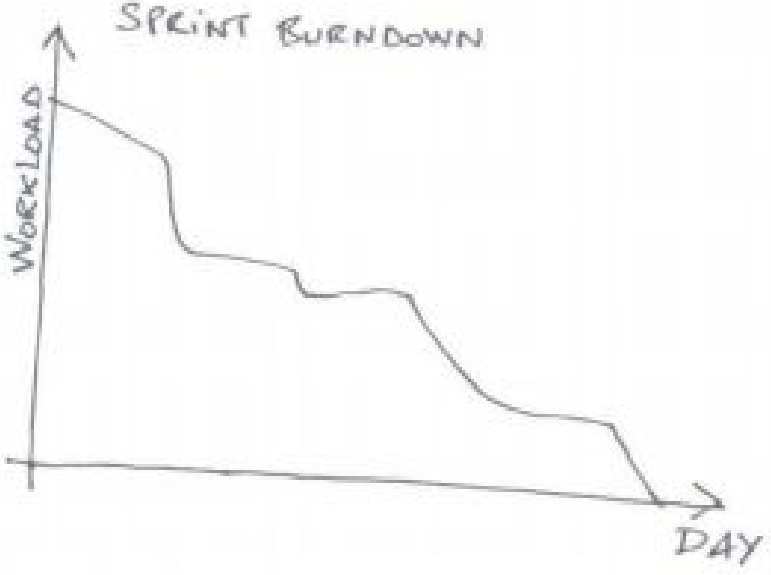
Daily agenda

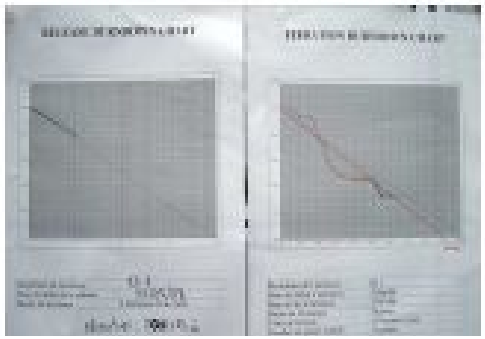
Post-it sur bureau

Méthodes adaptives : Task board

Méthodes adaptives : Burndown chart







Méthodes adaptives : Burndown chart

Tous les matins, on marque le nombre de points d'efforts restants.

Permet de :

- se situer dans le sprint
- rassurer hiérarchie et client !

Souvent en dents de scies: phases de tests.

Méthodes adaptives : Mood board

Mood board

Post-its that when wrong, post-its that went good

Groupés par thèmes durant la rétrospective

Pour alimenter cette réunion et ses résolutions

Mood calendar

Gommettes par membre d’équipe, rouge-jaune-verte, par jour.

Méthodes adaptives : avantages

- Validation concrète et non sur documents
- Limitation du risque à chaque itération
- Risque à la charge du client
- Conformité effort - prix
- Paiement progressif : au jour
- Client partenaire : retour rapide sur ses attentes
- Progressions : pas d'explosion des besoins à l'approche de la livraison : pas de « n'importe quoi pourvu que ça marche »
- Flexibilité : Modification des spécifications = nouvelle itération
- Maintenance = forme d'itération

Méthodes adaptives : avantages

A utiliser en situation dynamique, qui nécessite d'exploiter la malléabilité du logiciel.

Méthodes adaptives : désavantages

- Demande des efforts de planifications incessants et rigoureux.
- Demande la mise en place d'un cadre plus fin, pour éviter le grand mélange.

Un trop grand risque à prendre quand les besoins sont bien connus et figés, quand l'analyse et la conception sont claires.

Méthodes adaptives : Agile…

[Manifesto for Agile Software development]

4 valeurs:

- Priorité aux personnes et aux interactions sur les procédures et les outils
- Priorité aux applications fonctionnelles sur une documentation pléthorique
- Priorité de la collaboration avec le client sur la négociation de contrat
- Priorité de l’acceptation du changement sur la planification

Méthodes adaptives : Agile...

12 principes:

- Livraison ASAP de versions fonctionnelles
- Le changement comme avantage concurrentiel
- Livraisons intermédiaires aussi souvent que possible
- Coopération quotidienne entre utilisateurs et développeurs
- Construction d'une équipe motivée
- Favoriser l'échange oral sur l'écrit
- 1er indicateur d'avancement du projet : le fonct. de l'application
- Rythme soutenable pour les utilisateurs et les développeurs
- Attention continue à l'excellence technique et à la conception
- Toujours favoriser la simplicité
- Responsabilité confiée à l'équipe entière et volontariat
- Auto-ajustement de l'équipe pour améliorer son efficacité

Références

http://manu40k.free.fr/CoursGenieLogiciel.pdf

http://scrum.org