Gestion de projet multimédia

CM2 : Méthodes agiles

Mickaël Martin Nevot

V2.0.0

Gestion de projet multimédia

Méthodes agiles

- Méthodes de génie logiciel de **conduite de projet**
- Le plus souvent utilisées en **conception logicielle**
- Plus **pragmatique** que les méthodes traditionnelles
- Approche **adaptative** (réactive) plutôt que prédictive
- Implication maximum du **client**
- Favorise plus la **satisfaction du besoin** et moins des spécifications contractuelles
- Orienté vers les **personnes** plutôt que vers les processus

Concepts récents (officialisé en 2001 par un manifeste : le manifeste agile)

Valeurs

Méthodes agiles

- **Équipe** :
  Personnes et interactions
- **Application** :
  Logiciel fonctionnel
- **Collaboration** :
  Discussion de confiance avec le client
- **Acceptation du changement** :
  Réagir au changement

Méthodes « traditionnelles »


Processus et outils


Documentation complète


Négociation de contrats


Suivre le plan établi

Principes

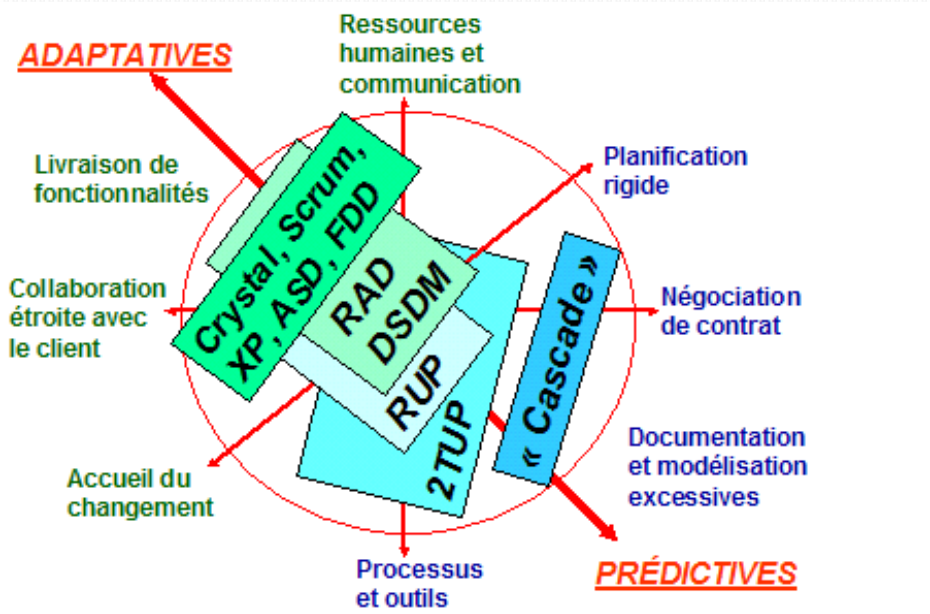
Les quatre valeurs se déclinent en douze principes généraux communs à toutes les méthodes agiles

1. Notre priorité est de satisfaire le client en livrant tôt et régulièrement des logiciels utiles
2. Le changement est bienvenu, même tardivement dans le développement ; les processus agiles exploitent le changement comme avantage compétitif pour le client
3. Livrer fréquemment une application fonctionnelle, toutes les deux semaines à deux mois, avec une tendance pour la période la plus courte
4. Les gens de l'art et les développeurs doivent collaborer quotidiennement au projet
5. Bâtissez le projet autour de personnes motivées ; donnez leur l'environnement et le soutien dont elles ont besoin, et croyez en leur capacité à faire le travail
6. Le plus efficace pour transmettre l'information est une conversation en face à face
7. Un logiciel fonctionnel est la meilleure unité de mesure de la progression du projet
8. Les processus agiles promeuvent un rythme de développement durable ; commanditaires, développeurs et utilisateurs devraient pouvoir maintenir le rythme indéfiniment
9. Une attention continue à l'excellence technique/qualité de la conception améliore l'agilité
10. La simplicité, l'art de maximiser la quantité de travail à ne pas faire, est essentielle
11. Une meilleure architecture/spécification/conception est due à une équipe auto-organisée
12. Régulièrement, l'équipe réfléchit pour devenir plus efficace, puis ajuste son comportement

Principales méthodes agiles

- RAD (*rapid application development*)
- DSDM (*dynamic systems development method)*
- **Scrum**
- **Extreme programming**

Date de création



Pratiques communes

- **Ressources humaines** :
  - Participation de l'**utilisateur final** aux groupes de travail
  - Groupes de travail disposant du **pouvoir de décision**
  - **Autonomie** et organisation centralisée de l'équipe (joue sur la motivation)
  - **Spécification et validation permanente** des Exigences

Pratiques communes

- **Pilotage du projet** :
  - Niveau **méthodologique variable** en fonction des enjeux
  - Pilotage par les **enjeux** et les **risques**
  - Planification stratégique globale basée sur des **itérations rapides** (scénario utilisateur ou *user story*)
  - Réalisation en **jalons** par prototypage itératif et incrémental
  - Recherche continue d'amélioration des pratiques

Pratiques communes

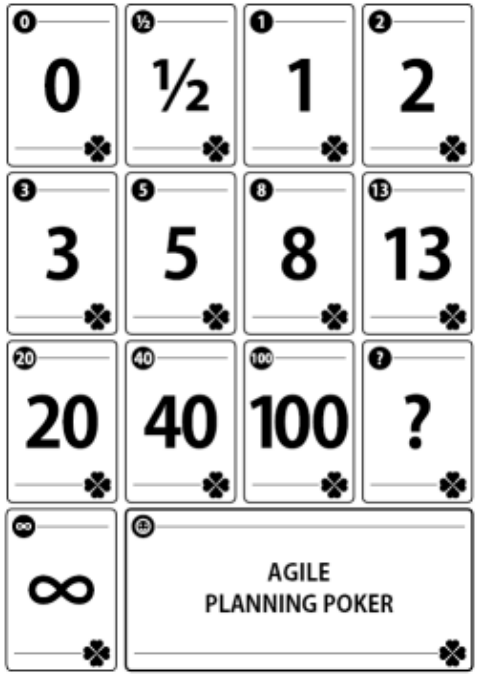
- **Qualité de la production** :
  - Recherche d'**excellence technique et de conception**
  - Vision graphique d'une **modélisation nécessaire et suffisante**
  - Vision de la **documentation nécessaire et suffisante**
  - **Normes et techniques raisonnables** de qualité du code
  - Architecture à base de **composants**
  - **Gestion des changements** automatisée

Critères d'éligibilité

- **Favorisant** :
  - Besoin rapide de mise à disposition du produit
  - Imprévisibilité des besoins du client
  - Nécessité de changements fréquents
  - Besoin de visibilité du client sur l'avancement des développements
  - Présence de l'utilisateur assurant un *feedback* immédiat
- **Défavorisant** :
  - Indisponibilité du client ou de l'utilisateur
  - Dispersion géographique des ressources humaines
  - Inertie des acteurs du projet ou refus des changements

Planning poker

- Méthode **ludique et efficace** pour avoir des estimations sur la complexité des fonctionnalités à produire
- **Expression libre**
- **Validation multiple** à des niveaux d'expérience différents
- Véritable mesure de **complexité relative** (suite de Fibonacci pour les évaluations)



Meilleures pratiques du *planning poker* en ligne sur le même site Web que ce cours

Méthode Scrum

- Rubgy plutôt que course de relais !

« … L'approche course de relais pour le développement de produits… peut être en conflit avec les objectifs de vitesse et de flexibilité maximum. A l'inverse, une approche holistique comme au rugby – quand une équipe essaie d'avancer en restant unie, en se passant le ballon de main en main – peut mieux servir les exigences de compétitivité d'aujourd'hui »
— H. Takeuchi et I. Nonaka

Références

Utilisateurs

- Google
- Microsoft
- Yahoo
- Electronic Arts
- Philips
- Siemens
- Nokia
- BBC
- Time Warner
- Etc.

Utilisations

- Développement de jeux vidéo
- Site Web
- Téléphonie mobile
- Développement interne
- Projet au forfait
- Infrastructure de réseaux
- Etc.

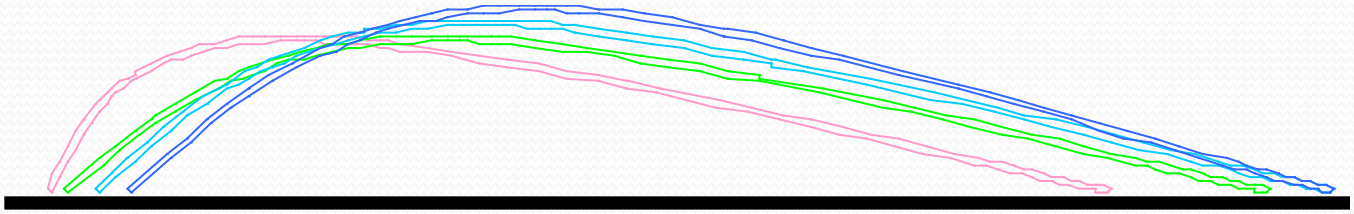
Séquencement des tâches

Exigences

Conception

Code

Test

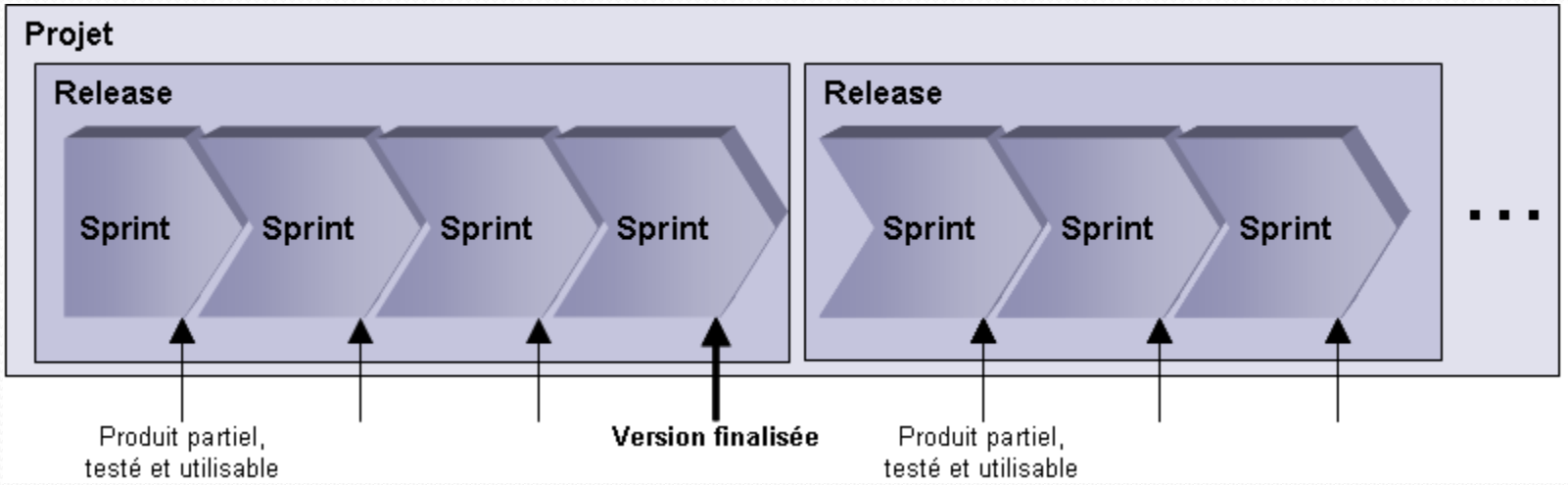


Points d'histoire

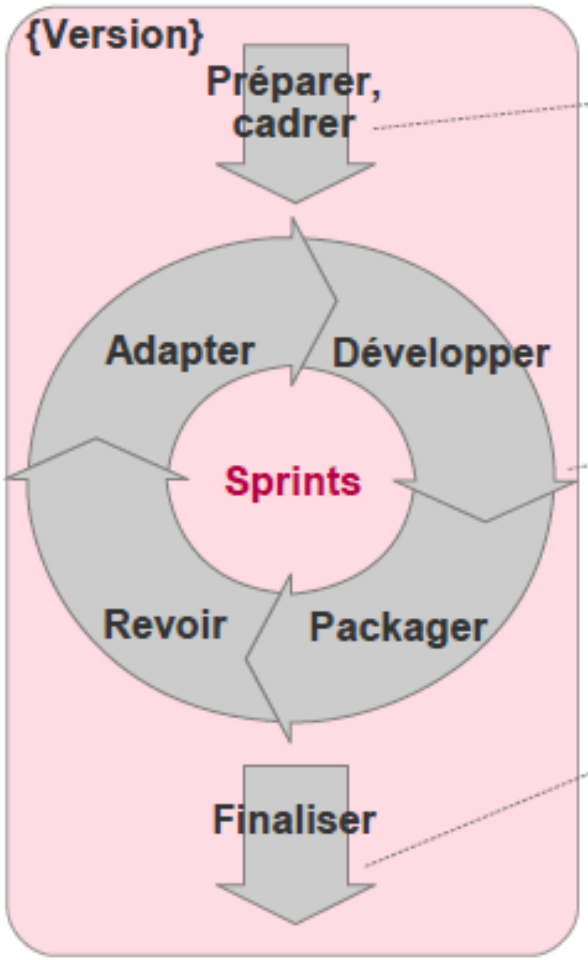
- **Unité d'estimation** temporelle Scrum
- Équivalent à des **jours-homme idéaux** :
  - Jour de travail parfaitement efficace
  - Sans perturbation

Phases d'une version de projet

- Le projet est découpé en versions (***releases***)
- Chaque version est divisée en itérations (***sprints***)



Phases d'une version de projet



- **Pré-jeu** :
  - *Planning* :
    - Définition du *backlog*, des jetons, de l'équipe
    - Identification des facteurs de risques
    - Choix des outils et infrastructure, logique
    - Budget, chiffrage et estimation du coût
  - Architecture générale, conception générale
- **Jeu** :
  - *Sprints*
  - Développement (analyse, conception, codage)
- **Post-jeu** :
  - Packaging global de la version
  - Intégration et jeux de tests
  - Documentation et manuels utilisateur globaux, accompagnement au changement

Sprint

- Durée : **2 à 4 semaines**
- Une **durée constante** apporte un meilleur rythme
- **But** : défini à l'avance à partir du *backlog* de produit
- **Tâches identifiées et estimées** entre **1 et 16 heures collectivement**
- **Pas de conception bas niveau** abordée
- **Stabilité** : différer la prise en compte d'un changement jusqu'au prochain *sprint*

Les rôles et artéfacts

Rôles

PO

Product Owner

Représentant du client et des utilisateurs. Formalise et priorise le backlog du produit

SM

Scrum Master

Meneur d'équipe, facilitateur, travaillant de façon rapprochée avec le Product Owner et l'équipe

È

Équipe

+/- 7 personnes

Elle doit tout faire pour délivrer le produit (organisation, profils, ...). La responsabilité est collective.

I

Intervenants ("chickens & pigs")

Observent & conseillent

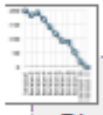
Artéfacts (éléments produits)

Backlog produit

Liste des macro-fonctionnalités de l'application, priorisées par leur valeur métier

Backlog de Sprint

Liste des tâches à implémenter sur un Sprint, classées par importance et état



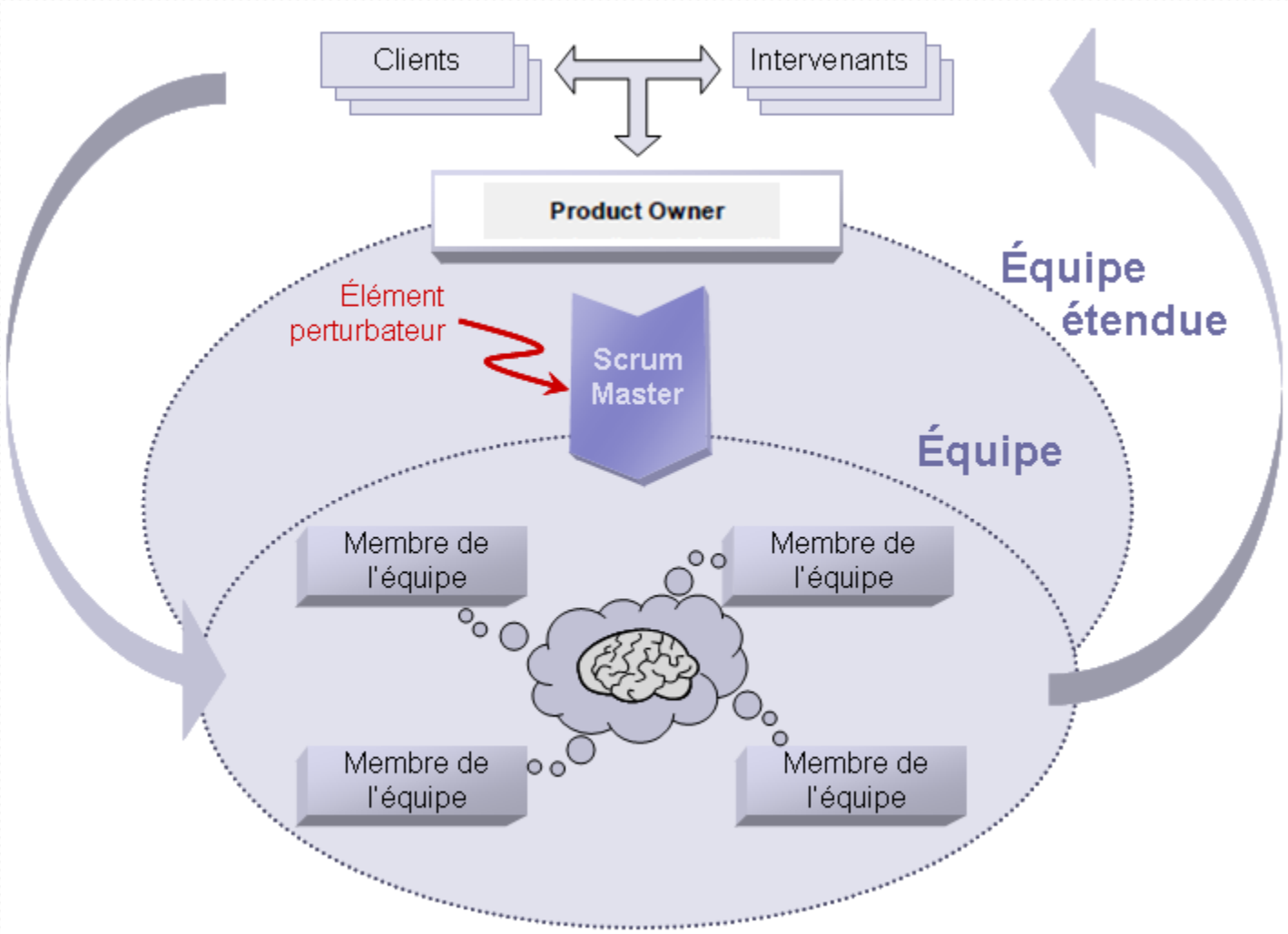
Burn-down chart

Diagramme permettant de suivre l'avancement du projet (représentation du Reste à Faire)

Produit partiel

Résultat du Sprint, testé et potentiellement livrable.

Organigramme



Activités

- Organisation de l'équipe :
  - **Auto-organisation**
  - **Environnement optimisé** :
    - Le *ScrumMaster* veille au quotidien au bienêtre de l'équipe
    - Informations partagées (tableaux blancs, wiki, etc.)
- Estimations :
  - Par **analogie**, **intuition** ou **point de fonction**
- Planification :
  - Prévision d'une **décontraction entre deux *sprints*** (veille technologique, projet personnel, etc.) sur un demi / un jour

Vélocité estimée

- **Vélocité :**

Réelle, après application congés, temps partiel, etc.

(VELOCITE ESTIMEE) = (JOURS-HOMME DISPONIBLES) x (FACTEUR FOCALISATION)

- **Facteur de focalisation** :
  - Estimation sur la concentration de l'équipe
  - Correction d'anomalie peut être comprise
  - Peut être évalué à partir des *sprints* précédents :

FACTEUR FOCALISATION DU SPRINT PRECEDENT :

(FACTEUR FOCALISATION) = (VELOCITE REELLE) / (JOURS-HOMME DISPONIBLES)

FACTEUR FOCALISATION DU DERNIER SPRINT :

40 % = 18 POINTS D'HISTOIRE / 45 JOURS-HOMME

VELOCITE ESTIMEE DE CE SPRINT :

50 JOURS-HOMME x 40 % = 20 POINTS D'HISTOIRE

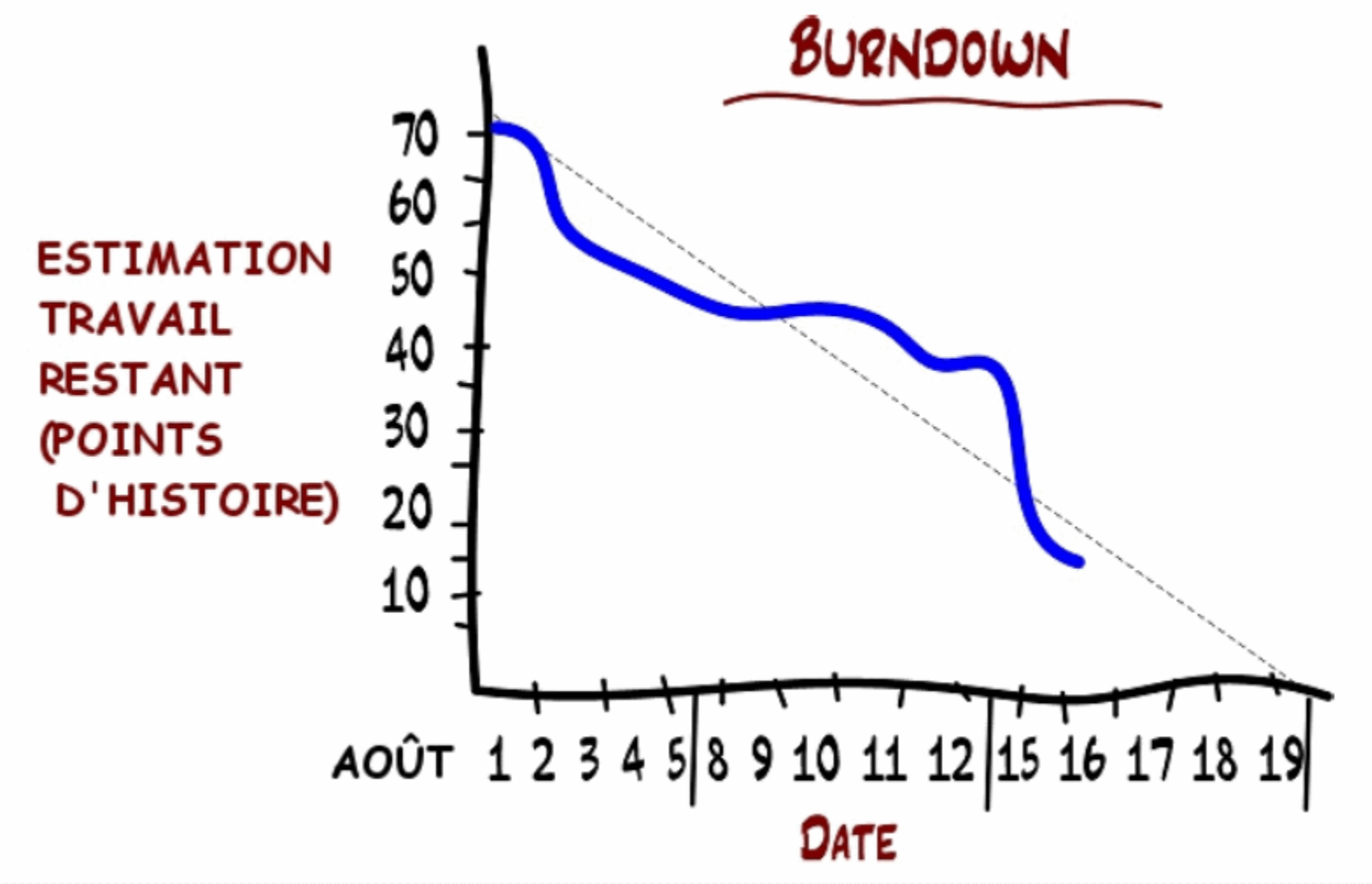
Exemple de backlog produit

Élément de *backlog*	Estimation
Connexion au jeu	1
Gestion d'un compte joueur	5
Élevage de chevaux	13
Gestion des tâches automatisées	8
Mise en place d'un concours	4
Course d'endurance	20
Générer le journal du jour	20
Écriture de l'aide	8
Gestion des exceptions	13
…	40
…	100

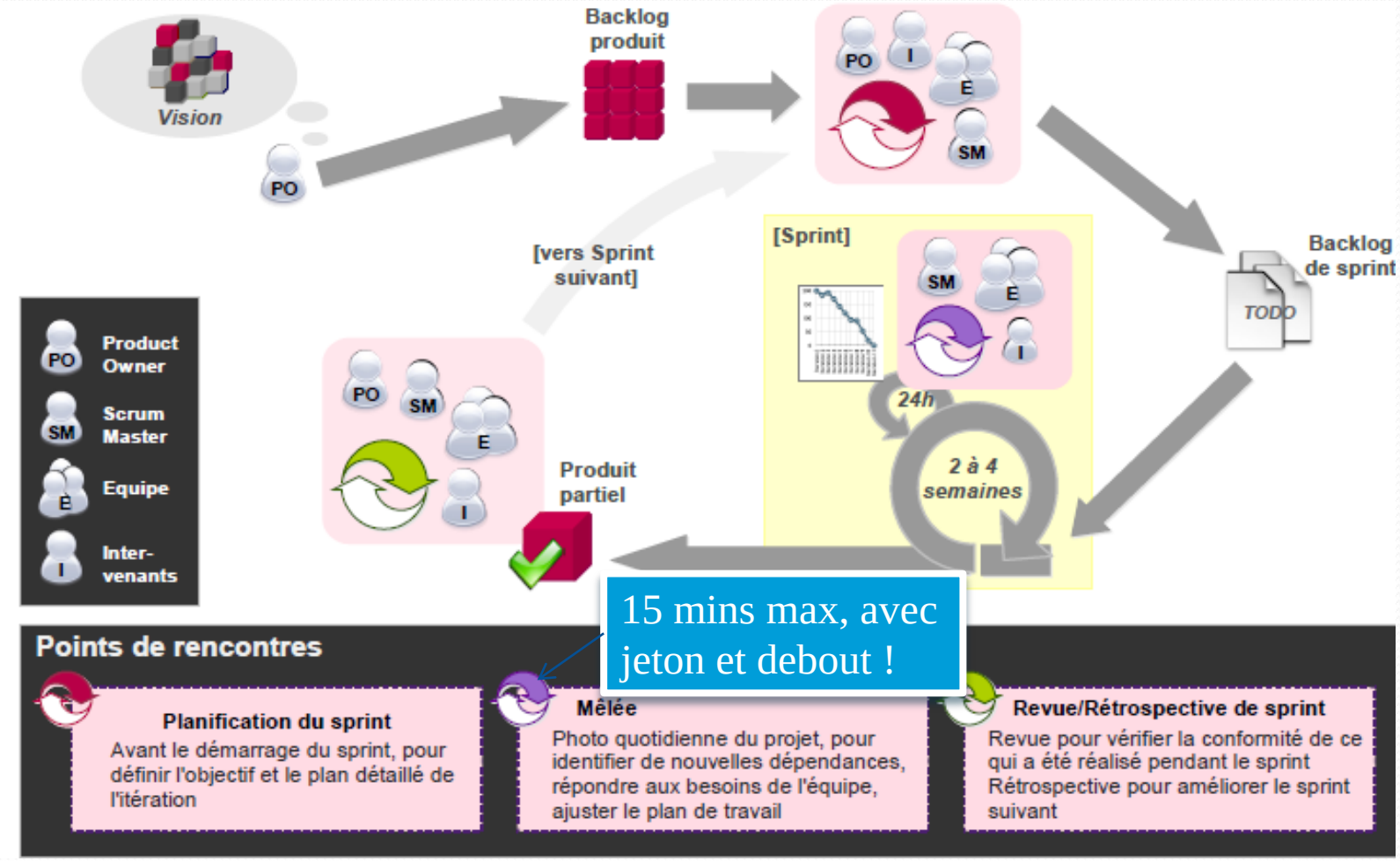
Exemple de backlog de sprint

Tâche	Lundi	Mardi	Mercredi	Jeudi	Vendredi
Coder l'IHM	8	4	8		
Coder la couche métier	16	12	10	4	
Tester l'intégration	8	16	16	11	8
Écrire l'aide	12				
Écrire la classe `Cheval`	8	8	8	8	8
Tracer les erreurs	8	8	8	8	8
Générer le journal			8	4	
Générer une tâche	10	8		12	4
Modifier un compte			10	10	
…	8	8	8	8	
…			16	14	

Exemple de burn-down chart



Le cycle de développement



Aspect contractuel

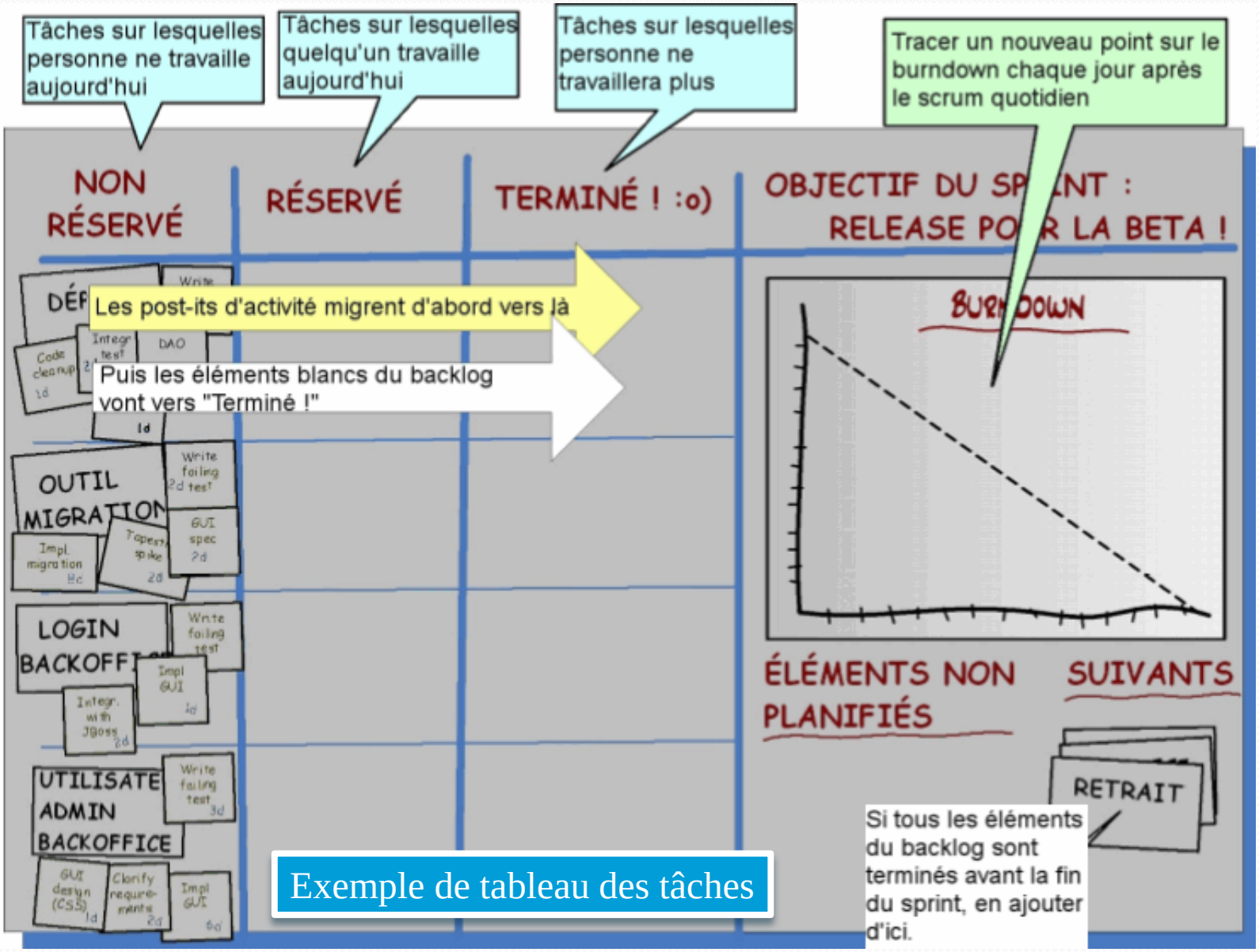
Sujet par définition problématique

- **Engagement sur les moyens** (compétences, etc.)
- Périmètre (initial) tracé de manière **collaborative**
- **Type de contrat :**
  - ***Co-sourcing*** :
    - Equipe mixte, sur le site du client
  - **Forfait collaboratif** :
    - Budget et dates de livraison fixés par le client
    - Le périmètre varie en fonction des contraintes
  - **Forfait avec engagement mesurable** :
    - Définition de points de mesure concrets

Estimation du rapport qualité/prix plutôt que prix le plus bas à tout prix

Outillage d'un projet Scrum

Tâche	Outils	Notes
***Backlog* (produit, *sprint*)**	• Microsoft Excel • Système de suivi de bogues • Spécifique agile	Souvent suffisant sauf pour le suivi des anomalies
Communication et gestion documentaire	• Wiki • Tableau de tâches, Post-it • Espace de stockage	Imprimer les cartes du *backlog* pour les afficher au regard de tous
Gestion de configuration et intégration continue	• Subversion • Hudson, Continuum, etc.	
Pilotage (tableaux de bord)	• Microsoft Excel • Spécifique agile	À homogénéiser avec la gestion des *backlogs*
Pilotage (suivis)	• Microsoft Excel • Spécifique agile	À homogénéiser avec la gestion des *backlogs*



Exemple de tableau des tâches

Scalabilité

- **Scalabilité** (extensibilité) : propriété d'un système, d'un réseau ou d'un processus qui témoigne de sa capacité à gérer des charges de travail importantes en toute souplesse ou à être agrandi sans difficulté
- Possibilité de faire un Scrum de Scrums
- Possibilité de faire un Scrum de Scrums de Scrums !
- Etc.

Forces et problèmes

Forces

- Propice à une confiance réciproque
- Transparence sur l'avancement (résultats rapides)
- Adaptabilité : le client peut modifier son besoin

Problèmes

- Le client est-il prêt à s'impliquer ?
- Comment gérer le changement par rapport au mode classique (organisation, contrat, etc.) ?
- Comment éviter de retourner dans le mode classique au premier coup dur ?

Co-pilotage avec le client

Forces et problèmes

Forces

- Plus de responsabilisation, plus de motivation
- Pas d'individualisme
- Amélioration permanente

Problèmes

- Capacité à travailler en équipe, à partager ?
- Capacité à se responsabiliser, à s'impliquer (envie ?)
- Capacité à faire l'effort d'être polyvalent (alternance codage, tests, correctif) ?
- Comment gérer les ressources non adaptées à ce mode de fonctionnement ?

Équipe et responsabilité collective

Forces et problèmes

Forces

- Orienté productivité et qualité (pas de travail superflu)
- Plus de souplesse et de créativité
- Recherche de la simplicité
- Pilotage au quotidien

Problèmes

- En cas de *framework* minimaliste : risque d'hétérogénéité ? (si mauvaise communication dans l'équipe, etc.)
- Pour un premier Scrum, comment estimer la charge de pilotage (accrue par rapport à forfait "classique" + coût des outils si spécifiques agile) ?

Cadre de production, méthodologie

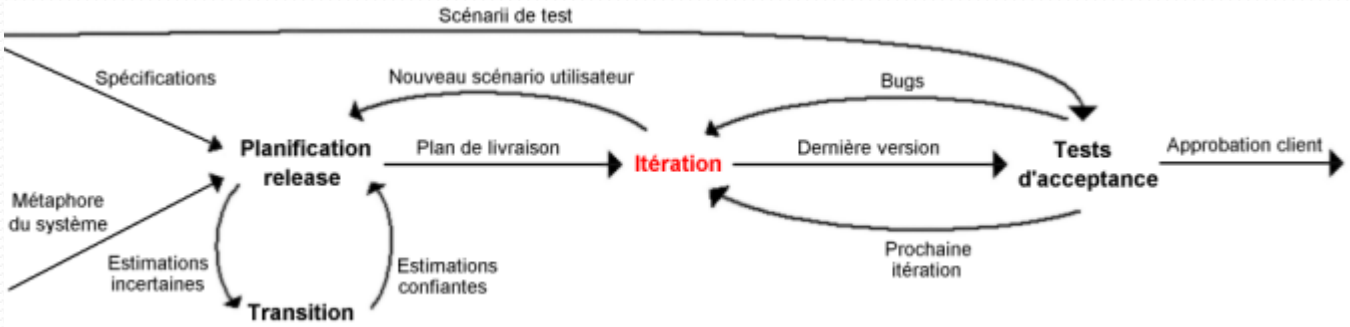
Extreme Programming

- Pousse à l'extrême des principes et pratiques simples :
  - Travail en **binôme**
  - **Rythme soutenable** (pas d'heure supplémentaire !)
  - Utilisation de **métaphores** !
  - **Convention de nommage**
  - **Client sur site**
  - **Conception simple**
  - **Intégration continue**
  - **Petites livraisons**
  - **Test unitaires** (Développement Piloté par les Tests)
  - **Tests de recette** (ou tests fonctionnels)
  - ***Refactoring*** (remaniement du code)

Cycle de développement XP

Scénarios utilisateurs

Transition architecturale



Environnements défavorables

- **Blocage culturel** (accroissement de la productivité)
- **Équipe de 20 développeurs ou plus**
- Coût de changement important (code propre et simple)
- ***Feedback* long** ou difficile à obtenir
- Infrastructure **non *open space***
- Non respect d'une discipline collective
- Résistance au changement

Aller plus loin

- Poulets et cochons Scrum
- Autres méthodes agiles :
  - RAD
  - DSDM
- Étude de méthodes en rapport avec les méthodes agiles :
  - 2TUP
  - RUP
  - Etc.
- Management agile / Entreprise agile
- Gestion de production *lean*

Liens

- Documents électroniques :
  - http://henrik-kniberg.developpez.com/livre/scrum-xp
  - http://www.aubryconseil.com
  - http://www.planningpoker.com
- Documents classiques :
  - Cours :
    - Mike Cohn. *Introduction à Scrum.*
    - Claude Aubry. *Scrum, Agilité... Et Rock'n roll.*
    - TimWi Consulting. *Scrum, Tour d'horizon de la méthode.*
    - Audrey Braconi. *Introduction à Scrum.*
  - Livres :
    - Henrik Kniberg. *Scrum et XP depuis les tranchées.*

Crédits

Auteur

Mickaël Martin Nevot
mmartin.nevot@gmail.com

Carte de visite électronique

Relecteurs

Cours en ligne sur : mickael.martin.nevot.free.fr