

L'AGILITÉ AU QUOTIDIEN

Formation 2 jours – Juillet 2019

2



RETOUR SUR LA 1ÈRE JOURNÉE

S Q L I 2 0 1 9



➤ SÉLECTION DE 3 IMAGES

Chacune représentant les méthodes agiles et leurs bénéfices.

➤ 3 ITÉRATIONS

LES RÈGLES



LOGISTIQUE

➤ HORAIRES PROPOSÉS

Jour 1 : 9:30 – 17:30

Jour 2 : 9:30 – 17:30

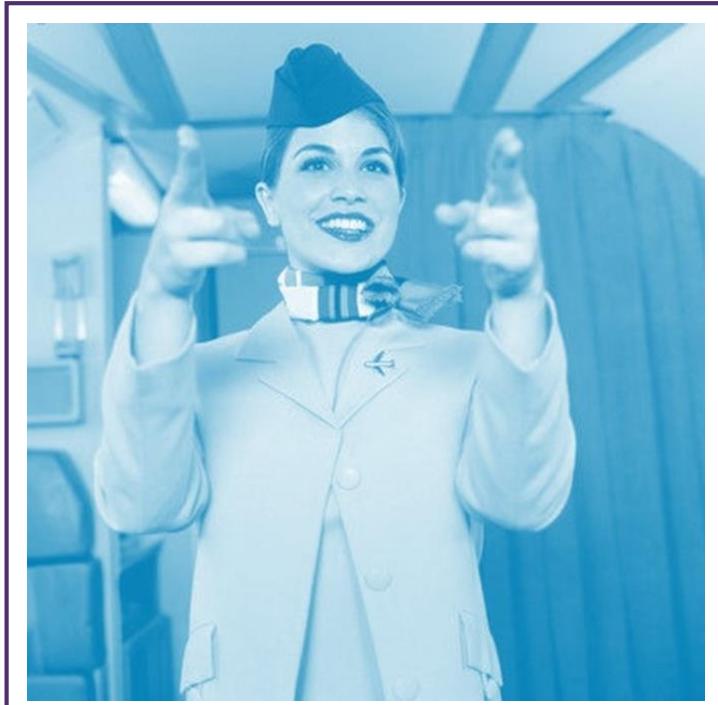
➤ DÉJEUNER

12:00 – 13:00

➤ AUTRES INTERRUPTIONS

Une pause matin et après-midi

➤ CONTRAINTES PARTICULIÈRES ?



SOMMAIRE

➤ Jour 1

- Pourquoi l'agilité
- Le Manifeste Agile
- Changer de paradigme
- Pilotage par la valeur
- La galaxie Agile
- Scrum
- Les rôles Scrum
- La gestion du backlog produit
- L'item du backlog
- La communication au sein de l'équipe Agile

➤ Jour 2

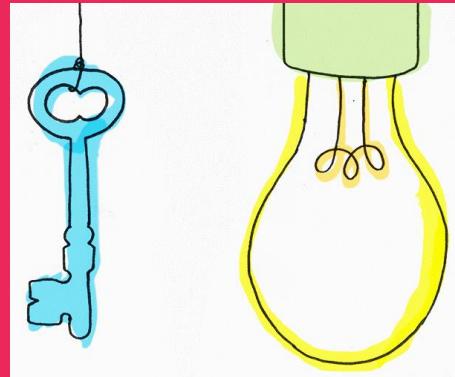
- Retour sur la 1^{ère} journée
- Gérer les priorités
- La planification de sprint
- L'estimation
- L'itération Scrum
- Le daily Scrum
- La revue de sprint
- La rétrospective
- Management visuel & indicateurs
- Vivre Scrum !
- Kanban
- XP
- La dette technique
- Pour conclure
- Annexes



LES IDÉES CLÉS

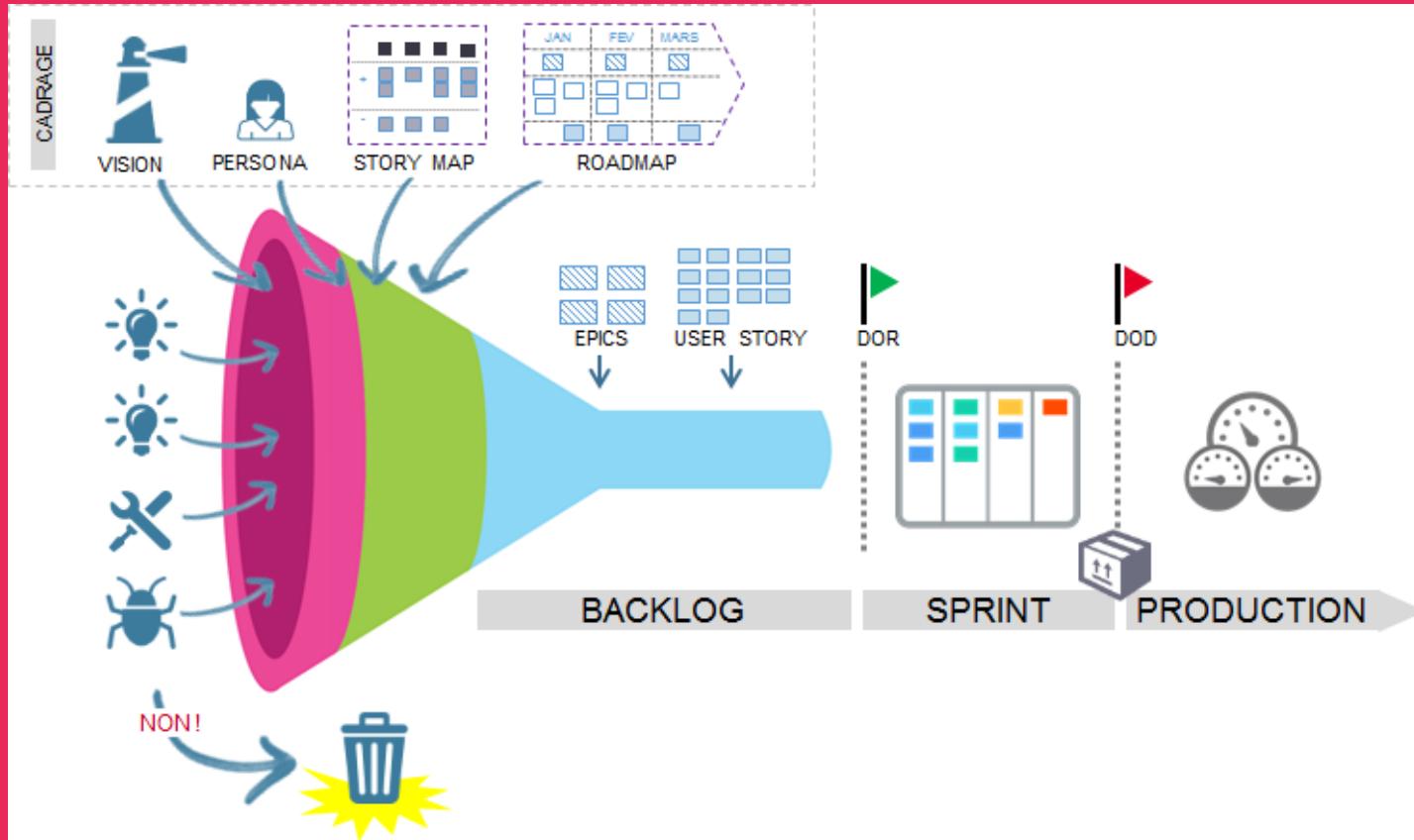
SQLI 2019

- LE CLIENT AU CŒUR DU PRODUIT
- L'ESPRIT D'ÉQUIPE
- LA COMMUNICATION EST LA CLÉ
- LA SIMPLICITÉ, L'EFFICACITÉ ET LA QUALITÉ
- LA FLEXIBILITÉ AUX CHANGEMENTS
- L'AVANCEMENT BASÉ SUR LE CONCRET





— AVANCER PAS À PAS ET PRÉCISER AU FUR À MESURE



GÉRER LES PRIORITÉS

UNE HABITUDE QUI NE DATE PAS D'HIER

S Q L I | 2 0 1 9



Source : Jean-Claude Grosjean – www.qualitystreet.fr



LA PRIORISATION DU QUOTIDIEN

S Q L I 2 0 1 9



➤ VOUS AVEZ MIS VOTRE RÉVEIL À 7H DU MATIN, ET VOUS DEVEZ PARTIR À 8H DE CHEZ VOUS POUR ARRIVER À L'HEURE AU TRAVAIL

1. Lister l'ensemble des activités habituelles que vous faites le matin
2. Affecter une durée à chaque activité
3. Prioriser les dans le temps

COMMENT PRIORISER LE BACKLOG ?

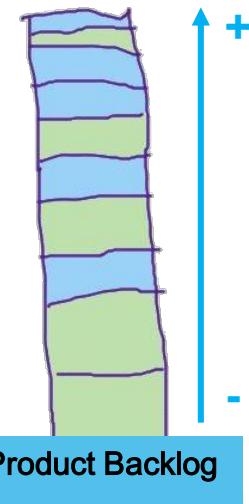
- LA VALEUR MÉTIER EST GÉNÉRALEMENT LA MESURE INITIALE UTILISÉE POUR PRIORISER LE PRODUCT BACKLOG
- DEUX ÉLÉMENTS EN HAUT DU BACKLOG NE PEUVENT AVOIR LA MÊME PRIORITÉ
- LES QUESTIONS POUR MODULER LA PRIORITÉ AU-DELÀ DE LA VALEUR MÉTIER

Risques et Incertitudes :

- Augmenter la priorité des éléments dont la mise en œuvre peut réduire les risques

Dépendances externes :

- Réduire la priorité des éléments ayant des dépendances externes, tant que celles-ci ne sont pas levées

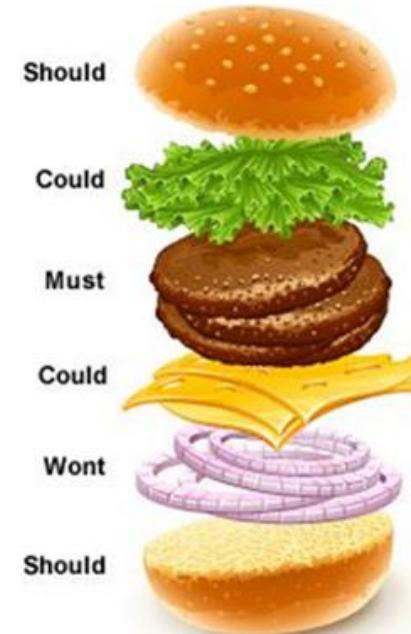
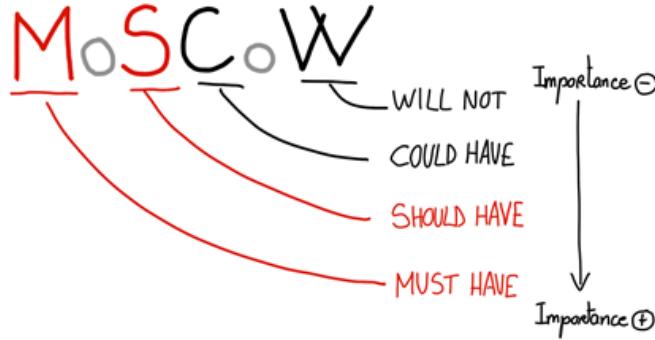


LE MODÈLE MOSCOW

SQLI 2019

➤ MOSCOW EST ISSU DU LEAN MANAGEMENT

Must Have – Should Have – Could Have – Won't Have (cette fois-ci)



LA VALEUR BUSINESS

➤ LA BUSINESS VALUE POSITIVE

Le gain pour le produit si la Story est livrée

➤ LA BUSINESS VALUE NÉGATIVE

La perte pour le produit si la Story n'est pas livrée

➤ LE RISQUE

➤ LES DÉPENDANCES

Le nombre de relations avec les autres stories

➤ L'EFFORT

$$\text{Priorité} = \frac{(\text{Valeur positive} - \text{Valeur négative}) \times \text{Risque} \times \text{Dépendances}}{\text{Effort}}$$

MATRICE DE PRIORISATION (1/2)

➤ CHOISIR DEUX AXES SIGNIFICATIFS AVEC 5 NIVEAUX

Urgence, Effort/Complexité/Coût, Risque, Valeur Métier

	Urgence	Valeur Métier
5	<ul style="list-style-type: none"> + Très forte contrainte de temps. + Très fortes dépendances avec les autres éléments. + Si pas fait immédiatement, il n'y a un faible intérêt de le faire. 	<ul style="list-style-type: none"> + Extrêmement Important pour la plupart ou tous les clients + Très fort impact sur la marque ou la réputation + Critique pour le succès du business
4	<ul style="list-style-type: none"> + Forte contrainte de temps. + Fortes dépendances avec les autres éléments. + A faire dans la prochaine itération à cause des clients ou des exigences contractuelles 	<ul style="list-style-type: none"> + Important pour beaucoup de clients + Impact significatif sur la marque ou la réputation + Avantage compétitif significatif
3	<ul style="list-style-type: none"> + Contrainte de temps modérée. + Dépendances modérée avec les autres éléments. + Préférable d'être terminé dans les prochaines itérations 	<ul style="list-style-type: none"> + Important pour plusieurs clients + Impact modéré sur la marque ou la réputation + Avantage compétitif modérément important
2	<ul style="list-style-type: none"> + Faible contrainte de temps. + Faible dépendance avec les autres éléments. + Acceptable d'être terminé dans les 2-3 prochaines itérations 	<ul style="list-style-type: none"> + Important pour seulement quelques clients + Faible Impact sur la marque ou la réputation + Avantage compétitif faible
1	<ul style="list-style-type: none"> + Pas de contraintes de temps. + Pas de dépendances avec les autres éléments. + Peu ou pas d'impact 	<ul style="list-style-type: none"> + Important pour peu ou aucun clients + Peu ou pas d'impact sur la marque ou la réputation + Peu ou pas d'avantage compétitif

MATRICE DE PRIORISATION (2/2)

➤ AFFECTER UNE VALEUR SELON LA CLASSIFICATION CHOISIE

Ici, urgence et valeur métier

➤ CALCULER LA PRIORITÉ SELON LA FORMULE

Priorité = produit des deux axes

Ici, Priorité = Urgence x Valeur Métier

Valeur Métier	5	5	10	15	20	25
4	4	8	12	16	20	
3	3	6	9	12	15	
2	2	4	6	8	10	
1	1	2	3	4	5	
	1	2	3	4	5	

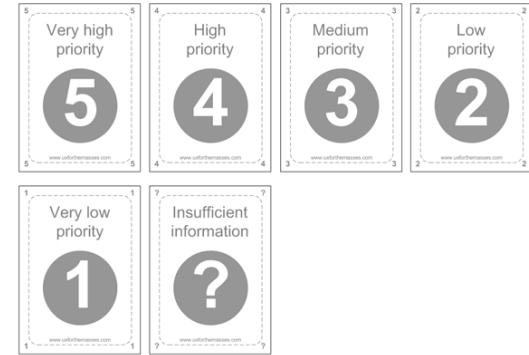
Urgence

➤ ON OBTIENT AINSI 4 NIVEAUX DE PRIORITÉ

➤ C'EST SUFFISANT POUR GÉRER LES 2-3 PROCHAINS ITÉRATION

25	Critique Doit être fait immédiatement
15-20	Important A inclure dans la prochaine itération
6-12	Modérément important A prévoir pour les 2-3 prochaines itérations
1-5	Nice to have Faible priorité Planification peu importante

CARTES DE PRIORITY POKER



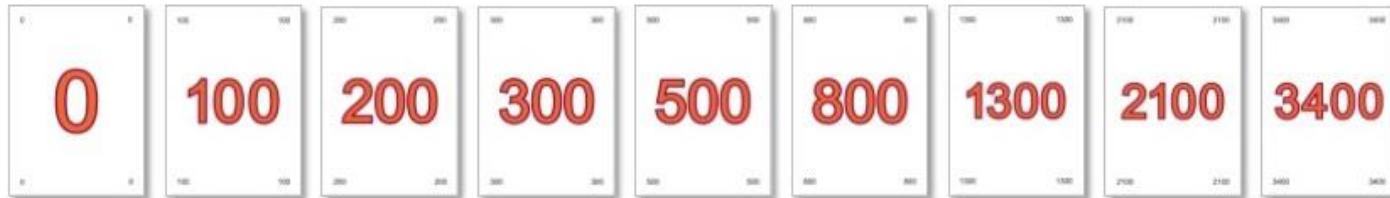
➤ POUR CHAQUE USER STORY :

Chaque participant évalue le handicap infligé au produit par l'absence de cette fonctionnalité

Chaque participant évalue l'avantage donné au produit par la présence de cette fonctionnalité

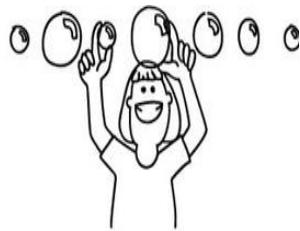
Le cumul des deux fournit une priorité globale du projet

Une pondération/coefficient Handicap/Avantage peut être utilisée



PRIORISATION COMPARATIVE

SQLI 2019



➤ BUBBLE SORT

On compare des éléments les uns aux autres, par paire d'éléments voisins



➤ ATTRIBUTION DE POINTS

Combien de points sur ma réserve de 1000 points dois-je attribuer à cet élément ?



➤ LE CANOT DE SAUVETAGE

On identifie l'élément le moins important, puis le suivant....

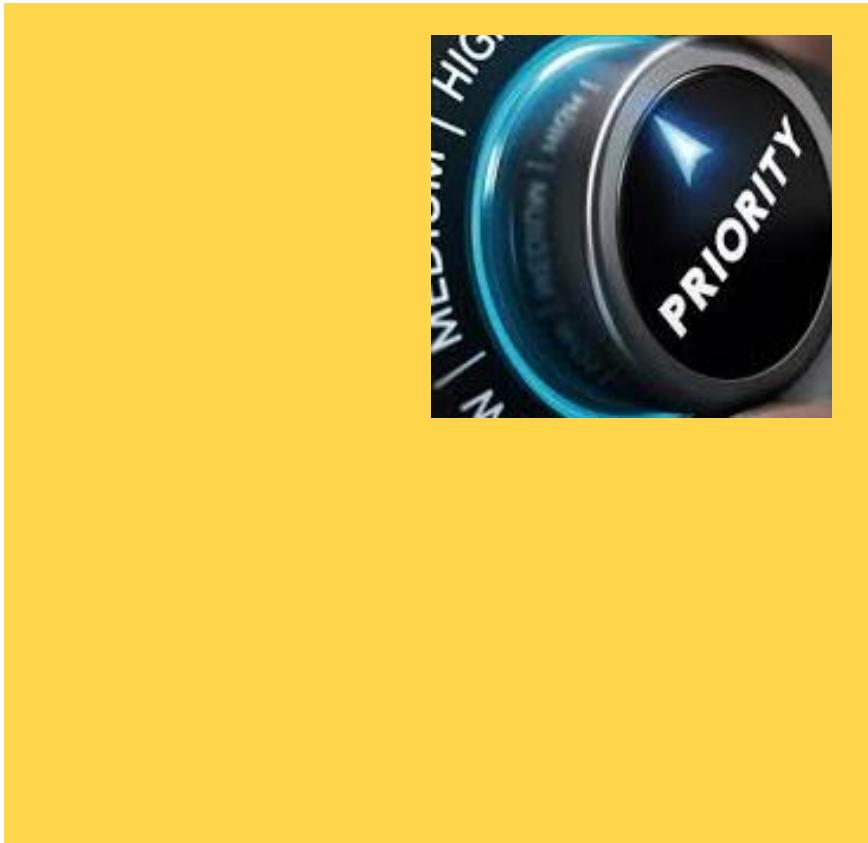
Pause





PRIORISATION DE BACKLOG (1/2)

SQLI 2019



➤ CONTEXTE :

Développer une application mobile sur le trekking permettre aux utilisateurs de trouver des itinéraires appropriés à leurs envies

➤ REPRENDRE LA STORY MAP INITIÉE

➤ PRIORISER AVEC LE JEU « BUY A FEATURE »

Utilisation des jetons de poker / billets de banque



le fil rouge



PRIORISATION DE BACKLOG (2/2)

S Q L I 2 0 1 9

Buy A Feature



- Acheter des fonctionnalités en disposant d'un montant d'argent limité
- Faire discuter les différents interlocuteurs Métier
- Obtenir un alignement sur la stratégie produit
- Obtenir une roadmap / release suivante / MVP (etc.)

➤ BUY A FEATURE

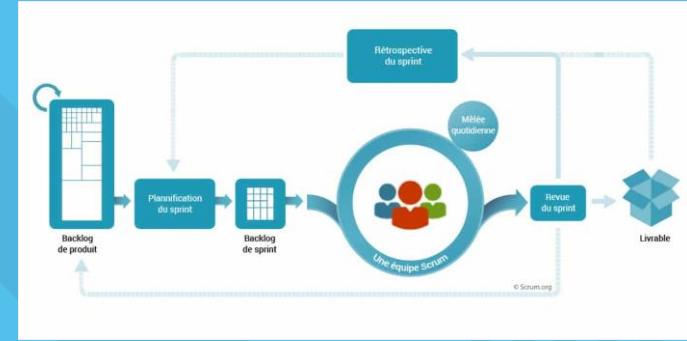
Identifier et se mettre d'accord sur les fonctionnalités à créer en premier

Matériel : billets de Monopoly ou jetons de jeu de cartes; liste de fonctions estimées en € ou en points

Chacun reçoit une somme insuffisante pour acheter toutes les fonctions dont il a besoin => il faut négocier

Innovation Games®

Source : www.innovationgames.com



LA PLANIFICATION DE SPRINT

PRINCIPES DE LA CÉRÉMONIE

> OBJECTIF :

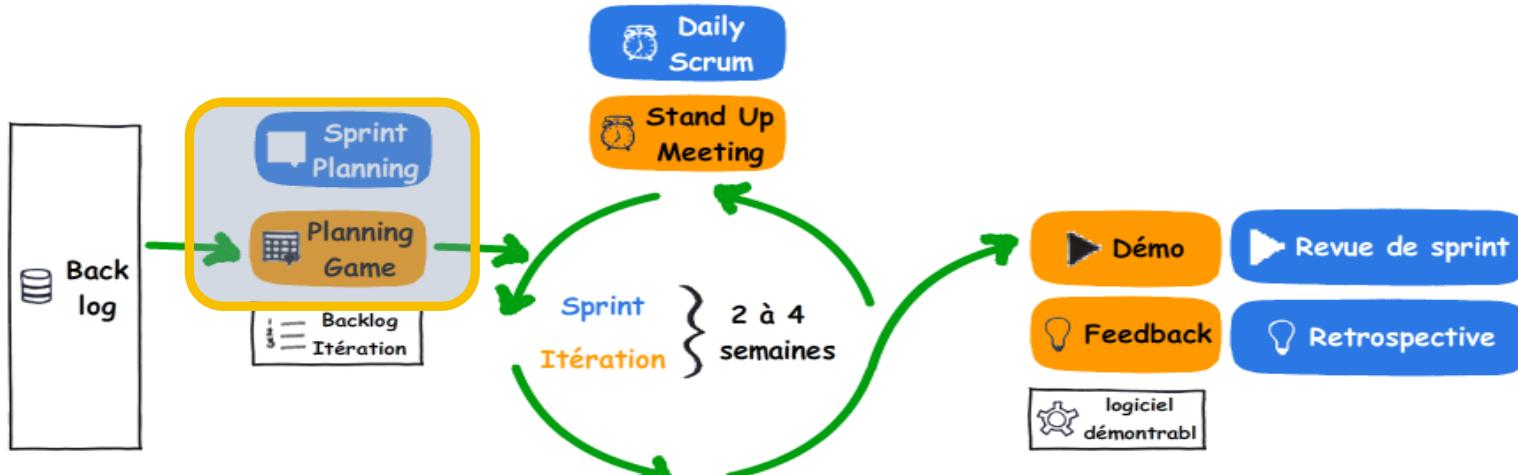
Élaborer, estimer et attribuer les tâches ajoutées au Backlog du Sprint.

> DURÉE MAXIMALE :

8h max pour un Sprint de 1 mois (en deux parties)

> PARTICIPANTS :

Equipe de développement, Product Owner, Scrum Master.



AVANT LE SPRINT PLANNING



➤ LE PRODUCT OWNER A DÉFINI

Les **priorités** du Product Backlog par rapport à leurs importances métier (valorisation)

Les **fonctionnalités** qu'il souhaite être développées pendant ce sprint



➤ LE SCRUM MASTER A VÉRIFIÉ QUE :

Le Product Owner est **prêt**

- au besoin, il l'a aidé à rédiger, à raffiner, à prioriser ...

L'équipe de développement a **préparé la réunion** :

- elle connaît les éléments à venir et est capable de les estimer.

Les dates des cérémonies du sprint sont connues :

- Date de début
- Heure de mêlée quotidienne
- Date et heure de démonstration
- Date et heure de rétrospective

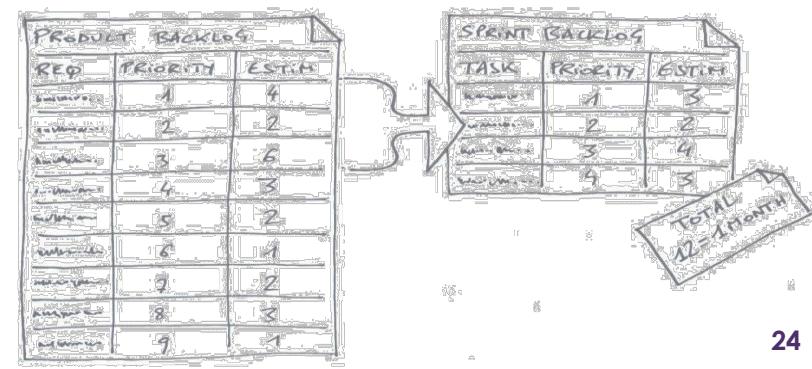
La capacité de l'équipe est connue.

2 ÉTAPES

La planification de Sprint répond aux deux questions suivantes :

1. QU'EST-CE QUI PEUT-ÊTRE TERMINÉ AU COURS DE CE SPRINT ?
2. COMMENT SERA EFFECTUÉ LE TRAVAIL CHOISI ?

L'ARTEFACT PRODUIT EST LE BACKLOG DE SPRINT



1 - QU'EST-CE QUI PEUT ÊTRE TERMINÉ AU COURS DE CE SPRINT ?

- LE PRODUCT OWNER PRÉSENTE À L'ÉQUIPE DE DÉVELOPPEMENT LES ÉLÉMENTS PRIORISÉS DU PRODUCT BACKLOG

- L'ÉQUIPE DE DÉVELOPPEMENT CHOISIT LES FONCTIONNALITÉS QUI POURRONT ÊTRE LIVRÉES AU COURS DU SPRINT

Toute l'équipe Scrum travaille alors ensemble sur la compréhension du travail à effectuer dans le Sprint

Une fois les items du Sprint Backlog construits, l'Équipe Scrum détermine l'objectif du Sprint.



DÉFINITION DU PRÊT (DOR)

➤ DEFINITION OF READY (DOR)

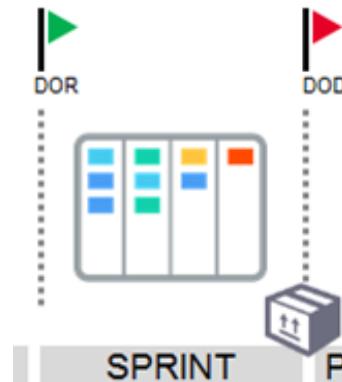
- UN ÉLÉMENT EST CONSIDÉRÉ PRÊT QUAND IL PEUT ÊTRE FINI EN UNE ITÉRATION
- IL DOIT

Faire partie des plus prioritaires
Être estimé suffisamment petit
Être décrit
Posséder des critères d'acceptation
Être compris et accepté par l'équipe

➤ L'ÉLÉMENT EST AFFINÉ EN SÉANCE DE TOILETTAGE (GROOMING)

➤ LA DÉFINITION DU PRÊT

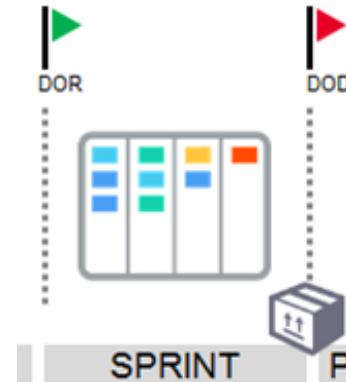
Est une convention entre le Product Owner et l'équipe de développement
Aide le Product Owner à proposer des éléments consistants



DÉFINITION DU PRÊT (DOR)

➤ EXEMPLE

- L'item est décrit sous la forme « EN TANT QUE ... JE VEUX ... AFIN DE ... »
- L'item est clair et compris par toute l'équipe. L'équipe dispose de suffisamment d'informations pour réaliser la solution correspondante.
- Le test d'acceptation lors de la démonstration est défini
- La valeur Métier est définie
- Les tâches associées sont définies et estimées
- Les cas de tests associés (TU et UAT) sont identifiés.



DÉFINITION DU TERMINÉ (DOD)

- **DEFINITION OF DONE (DOD)**
- **UN ÉLÉMENT EST CONSIDÉRÉ TERMINÉ QUAND IL NE RESTE PLUS RIEN À FAIRE SUR L'INCREMENT.**

➤ LA DÉFINITION DU TERMINÉ

Est déterminée par l'équipe de développement
Est partagée par l'équipe.

Respecte à minima les standards de l'entreprise

Est propre à chaque équipe SCRUM

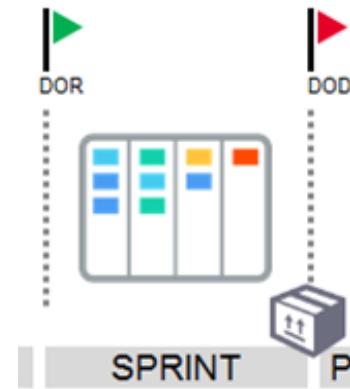
- Selon ses besoins
- Selon sa maturité

Est évolutive dans le temps

- Dès que l'équipe gagne en maturité, elle a intérêt à renforcer sa définition du Terminé

➤ SUGGESTIONS

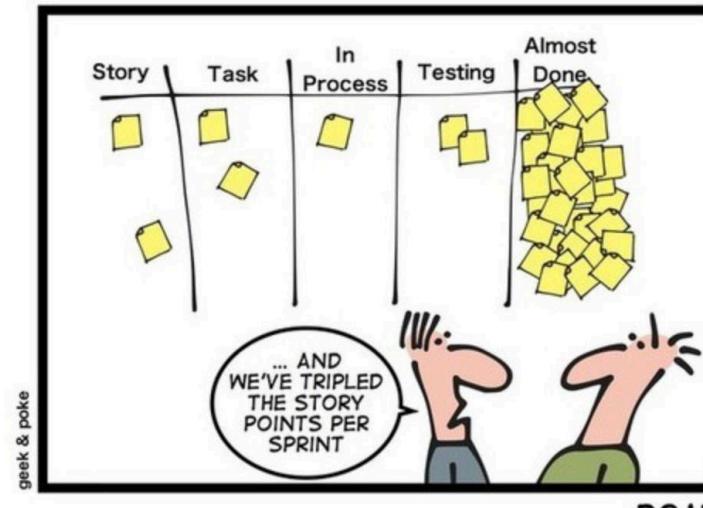
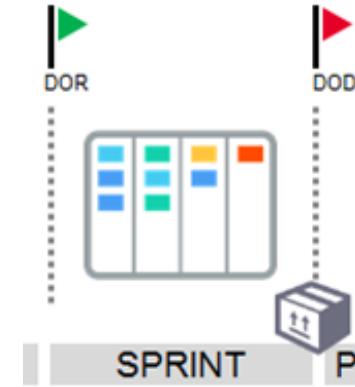
L'afficher comme check-list sur le Scrum Board



DÉFINITION DU TERMINÉ (DOD)

➤ EXEMPLE

- Le code est propre et conforme aux standards
- Le logiciel compile !
- Les Tests Unitaires existent et sont OK
- La recette utilisateur est validée
- Toute la documentation est mise à jour
- La démonstration a été répétée



2 - COMMENT LE TRAVAIL SÉLECTIONNÉ SERA-T-IL ACCOMPLI ?

- L'ÉQUIPE DE DÉVELOPPEMENT DÉCIDE COMMENT ELLE VA RÉALISER, DURANT LE SPRINT, UN INCRÉMENT DE PRODUIT « TERMINÉ »

L'équipe décompose le travail (en tâches) pour les premiers jours du Sprint

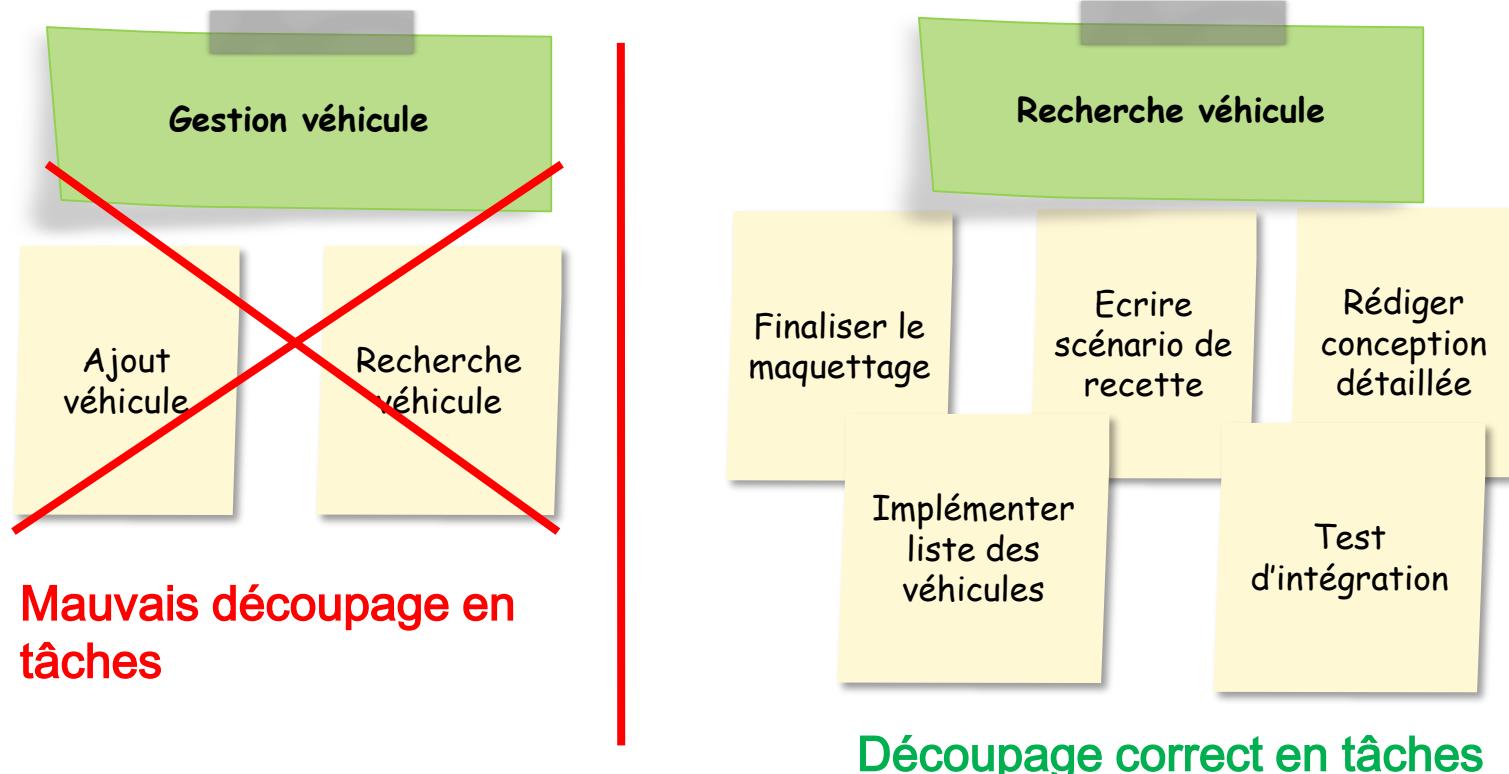
Le Product Owner peut être présent lors de la deuxième partie de la réunion de planification de Sprint pour clarifier les éléments sélectionnés du Product Backlog

Si l'équipe de développement détermine qu'elle a trop de travail, ou bien qu'elle n'en a pas assez, elle peut renégocier avec le Product Owner



Comment

L'IMPORTANCE DE BIEN IDENTIFIER LES TÂCHES



LA NÉGOCIATION

➤ LA NÉGOCIATION EST UN MOMENT PRIVILÉGIÉ ENTRE L'ÉQUIPE DE DÉVELOPPEMENT ET LE PRODUCT OWNER

Le Product Owner peut changer d'avis sur les priorités des éléments du Backlog et donc modifier le contenu de l'itération.

Le Product Owner sélectionne les éléments, éventuellement raffinés, en fonction des priorités et du temps disponible.

➤ LA QUALITÉ N'EST PAS NÉGOCIABLE !



EN SORTIE DE CÉRÉMONIE

➤ UNE FOIS QUE LES ITEMS DU SPRINT BACKLOG SONT CONSTRUITS, L'ÉQUIPE SCRUM DÉTERMINE L'OBJECTIF DU SPRINT

Il s'agit de définir la raison d'être du Sprint

L'objectif peut adresser plusieurs types de domaines :

- Fonctionnel : Ex : Réaliser la totalité de la gestion des droits des utilisateurs
- Organisationnels : Ex : Arriver à l'heure à chaque mélée quotidienne
- Techniques : Ex : Finaliser le calibrage de l'application

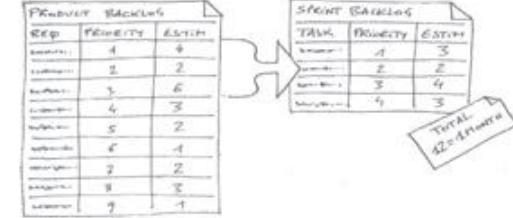
Il doit être défini de façon collégiale.

L'objectif du Sprint doit se concrétiser par une simple phrase, défini en termes métier, compréhensible par les membres non informaticiens de l'équipe.

➤ ET S'ENGAGE SUR L'INCRÉMENT À PRODUIRE.



LE SPRINT BACKLOG



- LE SPRINT BACKLOG SE CONSTRUIT PENDANT LE SPRINT PLANNING.
- IL REND VISIBLE TOUT LE TRAVAIL QUE L'ÉQUIPE DE DÉVELOPPEMENT IDENTIFIE COMME NÉCESSAIRE À L'ATTEINTE DE L'OBJECTIF DU SPRINT
- CONSIGNES À RESPECTER :

Les tâches de réalisation ne devraient pas dépasser environ 2 jours de travail pour une personne.

Inclure les tâches de test dans votre Sprint Backlog, y compris les tests d'acceptation.

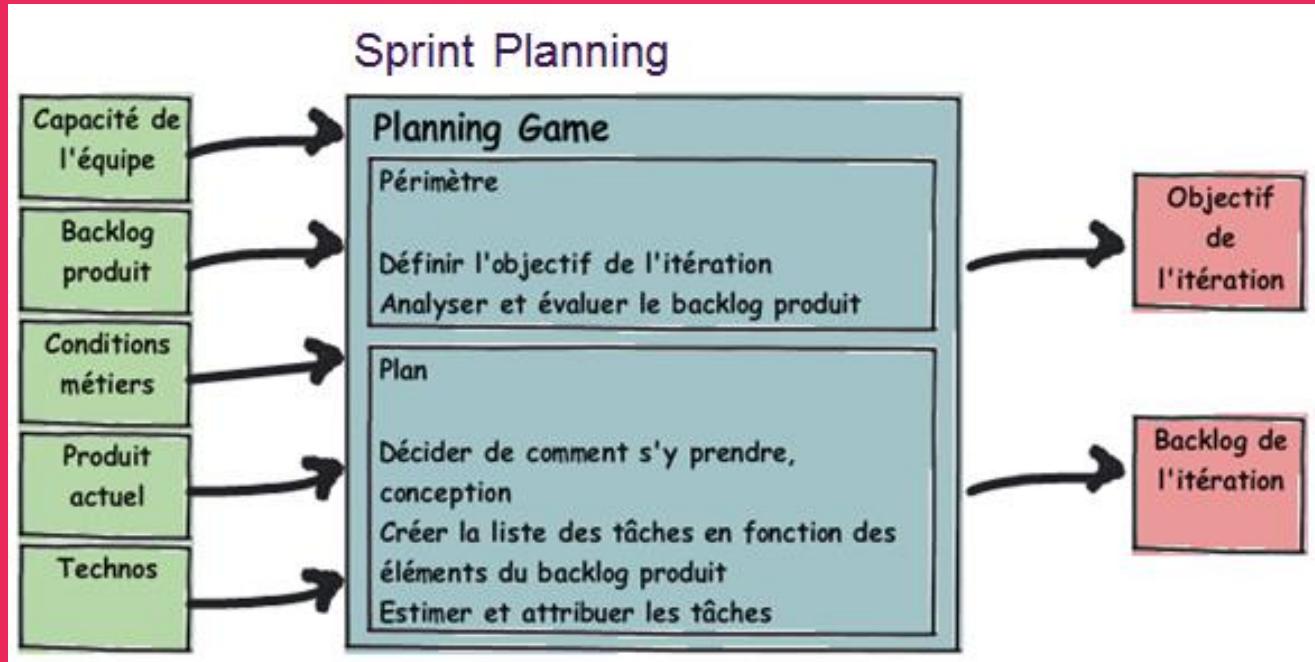
Il ne s'agit pas d'y faire figurer les tâches de gestion de projet (hors Scrum) ni les événements de Scrum.

Chaque Sprint a son propre Sprint Backlog.

Il n'est pas obligatoire d'affecter toutes les tâches dès le début du Sprint, elles peuvent être sélectionnées par chaque membre au fur à mesure de l'avancement quotidien.



LE SPRINT PLANNING



L'ESTIMATION



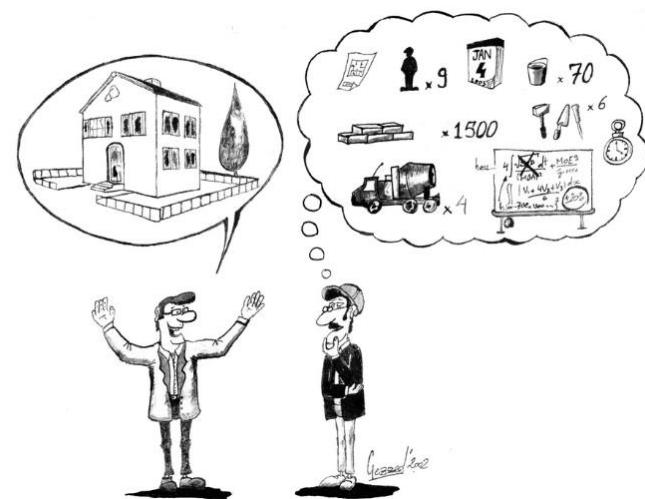
TOUS ESTIMÉS

> POURQUOI ESTIMER ?

- Pour se rendre compte de la complexité d'un travail
- Pour évaluer si la somme de travail est réaliste pour un Sprint (effort)
- Pour aider à la planification
- Pour mesurer les performances de l'équipe (cf. vélocité)

> QUOI ESTIMER ?

- Les items du Product Backlog
- Les tâches du Sprint Backlog



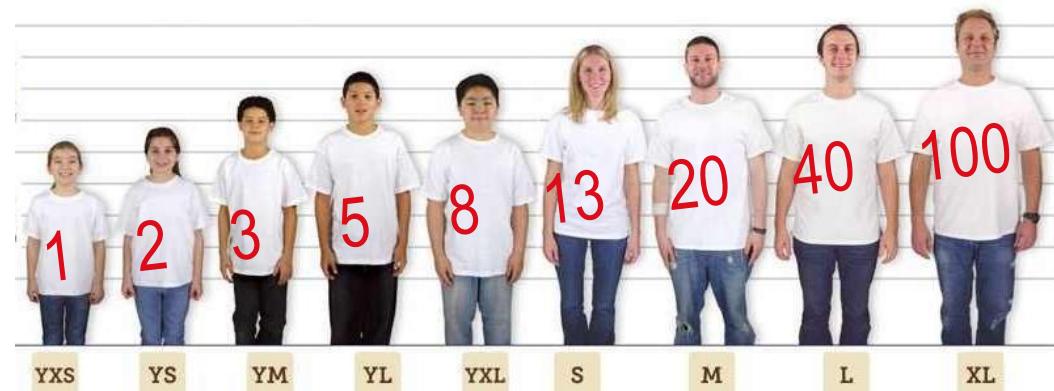
OUI MAIS ... COMMENT ?

➤ ESTIMER PAR COMPARAISON PLUTÔT QU'EN VALEUR ABSOLUE

Grouper les éléments selon les tailles similaires

- Macro : analogie avec des tailles de T-Shirt
- Raffiné : avec des chiffres
 - Jours/Homme
 - Suite de Fibonacci

Pour faciliter les calculs



ESTIMER AVEC LE PLANNING POKER

➤ LE POKER " FIBONACCI" NÉCESSITE UN JEU DE 13 CARTES DE VALEURS :

0 – 0,5 – 1 – 2 – 3 – 5 – 8 – 13 – 20 – 40 – 100 et l'infini

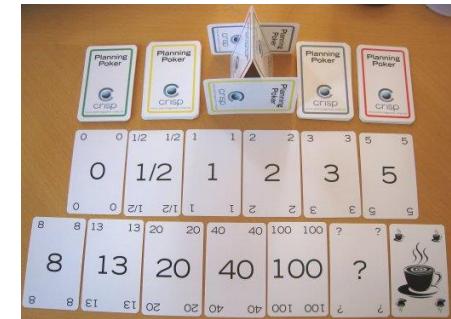
➤ ON TROUVE DES JEUX DE CARTES

- à commander sur internet,
- à découper à partir de planches de cartes téléchargées,
- sous forme d'applications pour smartphone, certaines en ligne pour synthétiser les résultats (pratique pour les équipes distantes)

➤ AVANTAGES :

- Permettre à tous de s'exprimer librement, en minimisant l'influence des autres membres de l'équipe. L'estimation sera meilleure parce que plusieurs personnes l'auront validée : des participants avec des niveaux d'expérience et d'expertise différents
- Favoriser les échanges entre le responsable de produits et l'équipe de développement

➤ L'ÉQUIVALENT EN JOURS-HOMMES EST PROPRE À CHACUN, SELON SES COMPÉTENCES, SON EXPÉRIENCE ET SA CONNAISSANCE DU DOMAINE.



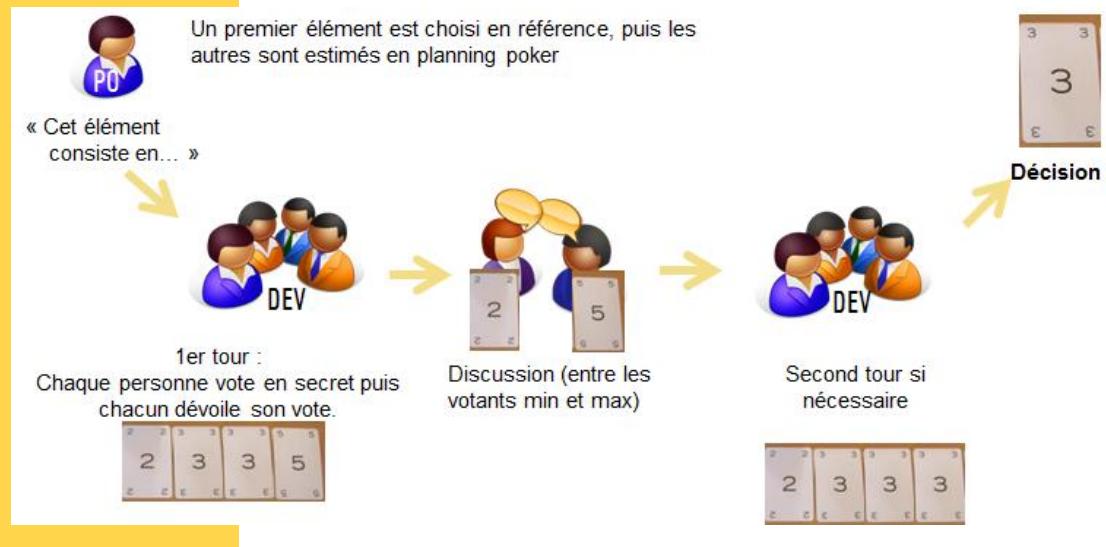


ESTIMER UN BACKLOG

SQLI 2019



- AVEC DES CARTES DE PLANNING POKER, ESTIMER L'EFFORT À FOURNIR POUR LAVER DES MODÈLES DIFFÉRENTS DE VOITURE.





Kangoo



Smart



Twingo



Ford T



Formule 1



Limousine



4 x 4



Monospace



V W Golf

LA VÉLOCITÉ DE L'ÉQUIPE DE DEV

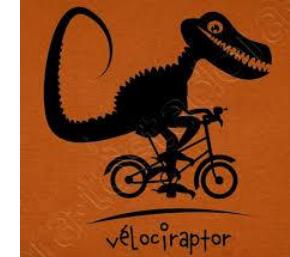
➤ LA VÉLOCITÉ EXPRIME LE NOMBRE DE POINTS TRAITÉS EN UNE ITÉRATION ET VALIDÉS PAR LE PRODUCT OWNER

➤ ELLE TRADUIT LA CAPACITÉ DE L'ÉQUIPE

En nombre de points par Sprint

Permet de fixer un périmètre réaliste au Sprint

Repère quantitatif



➤ MODE DE CALCUL

Moyenne des itérations

Pour un premier Sprint, 40 % consommés pour rendre l'équipe opérationnelle

Les itérations deux et trois permettent de dégager une tendance

➤ POINTS DE VIGILANCE :

La vélocité n'est pas un outil de management pour optimiser la capacité de l'équipe

L'augmentation de la vélocité n'est pas toujours une bonne nouvelle

➤ C'EST UN OUTIL DE PLANIFICATION POUR LE RESTE DU PROJET



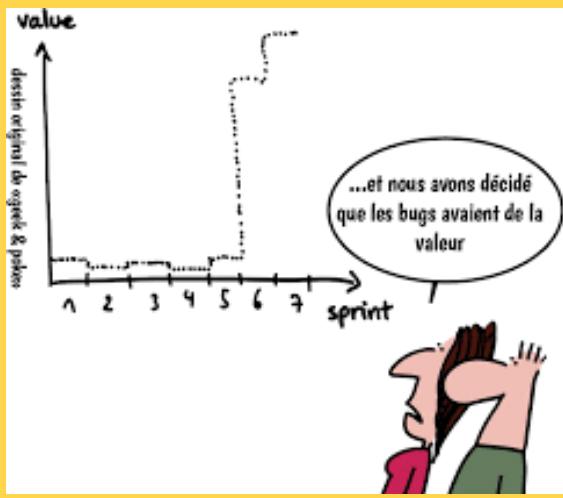
CALCULER LA VÉLOCITÉ

SQLI 2019

- LES RELEVÉS DE VÉLOCITÉ D'UNE ÉQUIPE SONT LES SUIVANTS :

	Vélocité	Points restants
Sprint 1:	55 pts	560
Sprint 2:	65 pts	495
Sprint 3:	50 pts	445
Sprint 4:	75 pts	370

- QUELLE EST LA VÉLOCITÉ MOYENNE SUR TROIS ITÉRATIONS ?
- COMBIEN DE SPRINTS SERONT NÉCESSAIRES POUR RÉALISER LES POINTS RESTANTS ?





ENERGIZER

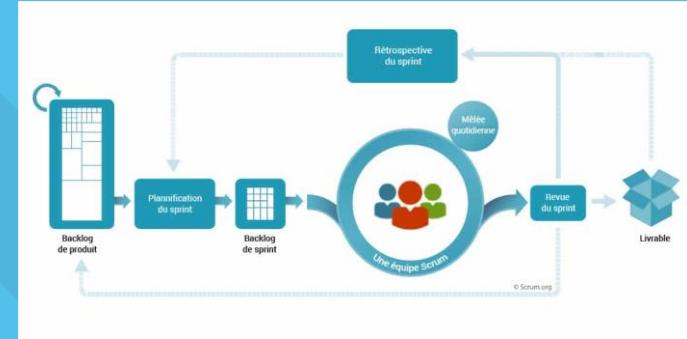


➤ SI VOUS ÉTIEZ UN ANIMAL ...

➤ IMITEZ LE !



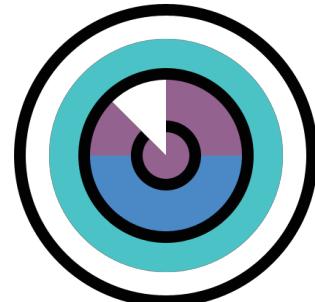
L'ITÉRATION SCRUM



PENDANT L'ITÉRATION

➤ LE SPRINT EST UTILISÉ POUR RÉALISER UN OBJECTIF :

- **L'objectif** du sprint est **fixe**; les changements qui le remettent en cause ne sont donc pas permis
- Les objectifs de **qualité** sont **maintenus**; ils ne sont jamais revus à la baisse
- Le périmètre peut être **clarifié et renégocié** entre le Product Owner et l'Équipe de Développement selon ce que l'Équipe Scrum apprend





DÉROULEMENT

➤ PENDANT UNE ITÉRATION, ON NE CHANGE PAS LES ÉLÉMENTS DE BACKLOG D'ITÉRATION À RÉALISER !

Et si l'équipe est en retard ou en avance ?

Le Product Owner peut alors décider d'enlever ou d'ajouter un élément
Dans son organisation, l'équipe doit prévoir :

- Les urgences
- Le refactoring
- La préparation de la démonstration

➤ PENDANT LE DÉROULEMENT DE L'ITÉRATION

L'équipe auto-organisée décide de l'affectation des tâches et met à jour son "reste à faire" pour chaque tâche

➤ SUIVI DE L'ITÉRATION

La somme en heures des tâches du Backlog d'itération constitue le reste à faire global de l'itération

Chaque jour, chaque reste à faire de tâche doit être mis à jour pour voir les heures restantes à produire

EN FIN D'ITÉRATION : L'INCRÉMENT

➤ L'INCRÉMENT EST CONSTITUÉ DES ÉLÉMENTS DU PRODUCT BACKLOG TERMINÉS PENDANT LE SPRINT AINSI QUE DE LA VALEUR CUMULATIVE DES INCRÉMENTS LIVRÉS DANS LES SPRINTS PRÉCÉDENTS.

➤ LE PRODUIT EST POTENTIELLEMENT UTILISABLE

Le produit, même si rudimentaire, est complet.

Il comprend donc :

- La documentation : design, installation, manuel utilisateur...
- Le produit lui-même
- Ce qui est nécessaire à sa mise en œuvre : installateurs, configurations...

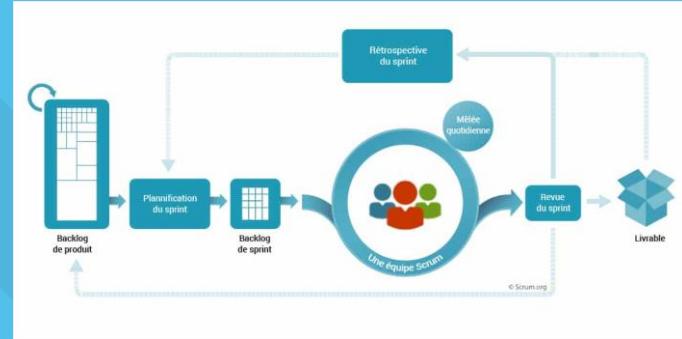


L'ANNULATION DE SPRINT

- UN SPRINT PEUT ÊTRE ANNULÉ AVANT LA FIN DE SON BLOC DE TEMPS LORSQUE SON OBJECTIF DEVIENT CADUC
- SEUL LE PRODUCT OWNER POSSÈDE LE DROIT D'ANNULER LE SPRINT, MÊME S'IL PEUT LE FAIRE À LA DEMANDE DES PARTIES PRENANTES, DE L'ÉQUIPE DE DÉVELOPPEMENT, OU DU SCRUM MASTER.
- QUAND UN SPRINT EST ANNULÉ :
 - Tous les éléments du Product Backlog « terminés » sont passés en revue
 - Tous les éléments incomplets du Product Backlog doivent être ré-estimés et remis dans le Product Backlog

ANNULÉ

LE DAILY SCRUM



PRINCIPES DE LA CÉRÉMONIE

> OBJECTIF :

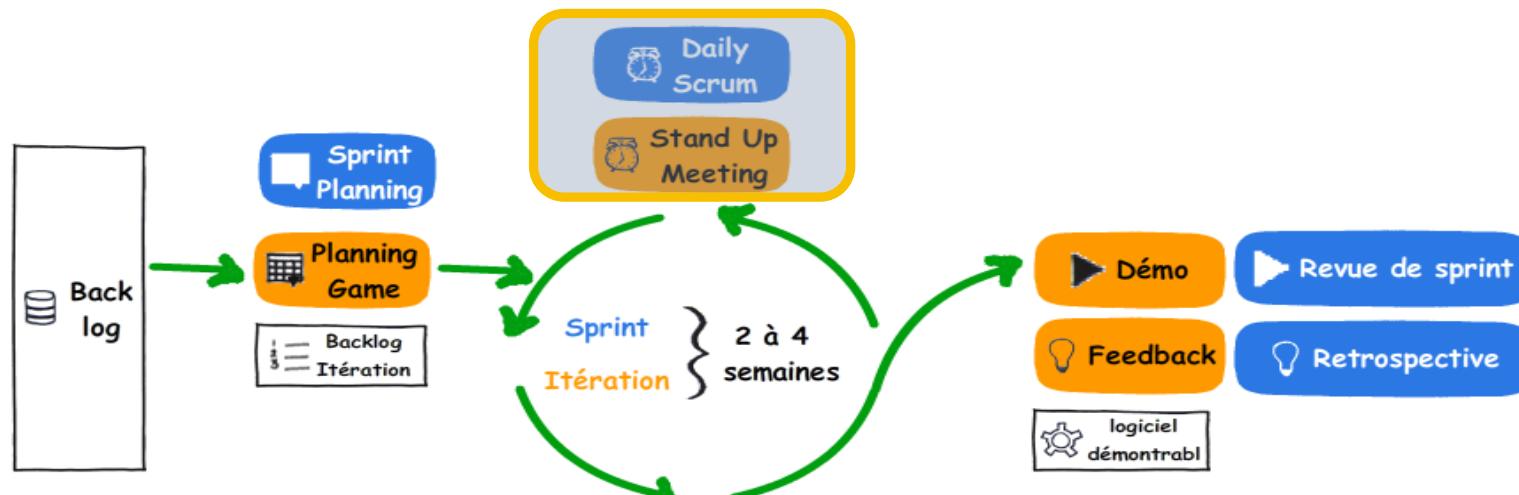
Synchronisation quotidienne de l'équipe

> DURÉE :

15 minutes

> PARTICIPANTS OBLIGATOIRES:

Toute l'équipe de développement



POURQUOI FAIRE ?



➤ OBJECTIFS :

Synchroniser l'équipe de réalisation, fluidifier les problèmes, créer l'esprit d'équipe.

- Rendre transparent, Inspecter, S'adapter

➤ DÉROULEMENT :

Tous les jours, pendant 15 minutes maximum au même endroit, afin de réduire la complexité de fonctionnement

➤ QUI :

Seules les personnes travaillant à la réalisation des tâches du Sprint Backlog doivent participer au Daily Scrum.

Si le Scrum Master ou Product Owner ont aussi des tâches à réaliser figurant dans le Sprint Backlog, ils participent au Daily Scrum.

JUSTE 3 QUESTIONS À SE POSER

1. QU'AI-JE FAIT HIER QUI A AIDÉ L'ÉQUIPE DE DÉVELOPPEMENT À ATTEINDRE L'OBJECTIF DU SPRINT ?
2. QUE PRÉVOIS-JE DE FAIRE AUJOURD'HUI POUR AIDER L'ÉQUIPE DE DÉVELOPPEMENT À ATTEINDRE L'OBJECTIF DU SPRINT ?
3. VOIS-JE DES OBSTACLES QUI M'EMPÊCHENT OU EMPÊCHENT L'ÉQUIPE DE DÉVELOPPEMENT D'ATTEINDRE L'OBJECTIF DU SPRINT ?





LE DAILY SCRUM FROM HELL

S Q L I 2 0 1 9



- **EFFECTUER SA 1ÈRE STAND UP MEETING**
- **3 QUESTIONS :**
 - 1- Qu'ai-je fait **hier qui a aidé** l'équipe de développement à atteindre l'objectif du Sprint ?
 - 2- Que prévois-je de faire **aujourd'hui pour aider** l'équipe de développement à atteindre l'objectif du Sprint ?
 - 3- Vois-je des **obstacles** qui m'empêchent ou empêchent l'équipe de développement d'**atteindre** l'objectif du Sprint ?

le fil rouge



LE DAILY SCRUM

SQLI 2019

➤ AVANTAGES

- Visibilité partagée de l'avancement
- Motivation forte et implication de l'équipe
- Résolution rapide des problèmes ou blocages

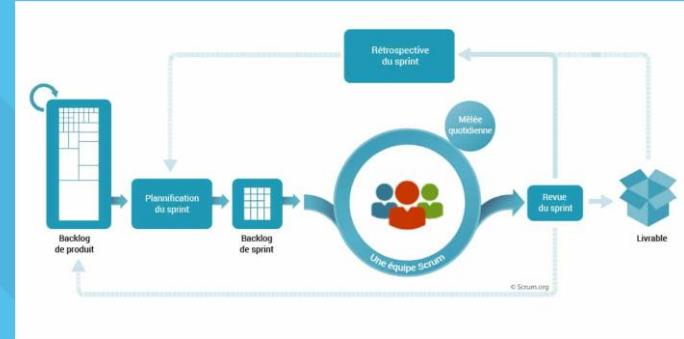
➤ POINTS D'ATTENTION

- Ne résoudre les problèmes qu'après la mêlée !
- La présence d'autres personnes (ex. managers) peut perturber l'équipe.



Le Scrum Master doit simplement s'assurer que l'équipe de développement sait comment mener un Daily Scrum et le fait.

LA REVUE DE SPRINT



PRINCIPES DE LA CÉRÉMONIE

> OBJECTIF :

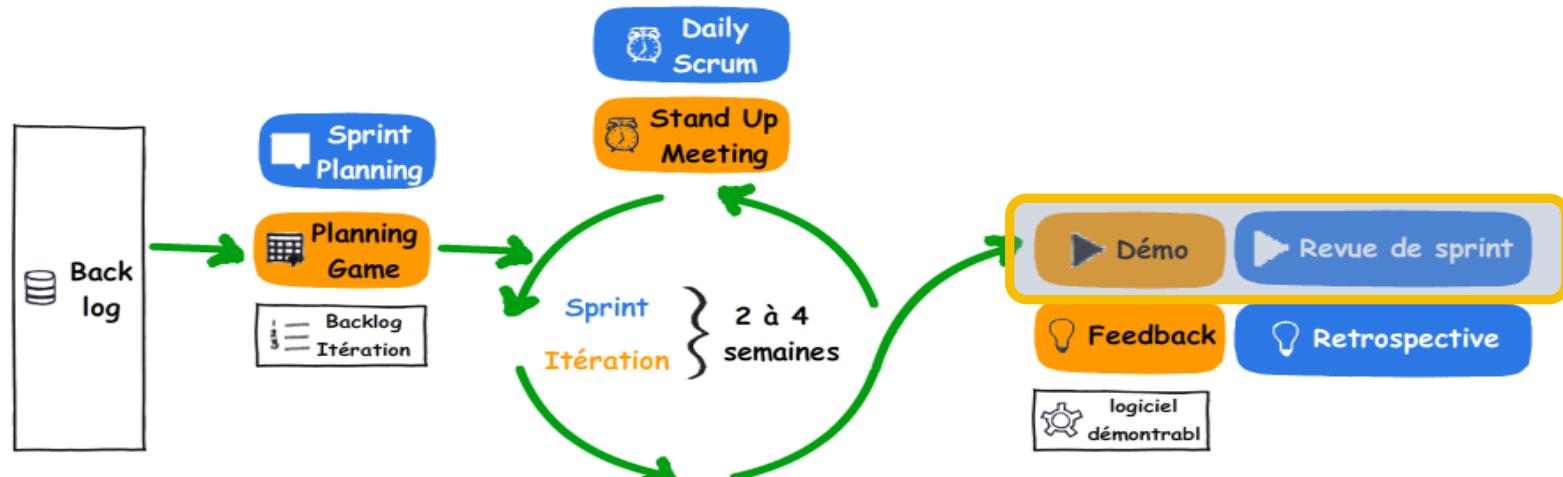
Inspecter l'incrément du produit et adapter le Product Backlog si nécessaire

> DURÉE :

4h pour un Sprint de 1 mois , 2h pour un Sprint de 2 semaines

> PARTICIPANTS :

Equipe de développement, Product Owner, Scrum Master, parties prenantes



DÉROULEMENT

➤ LE PRODUCT OWNER

Identifie ce qui a été terminé et ce qui n'a pas été terminé.

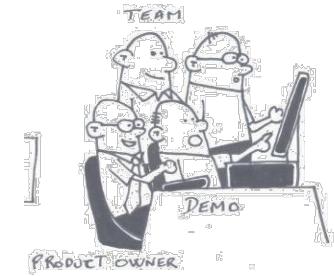
Discute du Product Backlog tel qu'il est. Il détermine des dates probables d'achèvement en fonction des progrès à ce jour.

➤ L'ÉQUIPE DE DÉVELOPPEMENT

Discute de ce qui s'est bien déroulé durant le Sprint, quels problèmes ont été rencontrés, et comment ces problèmes ont été résolus.

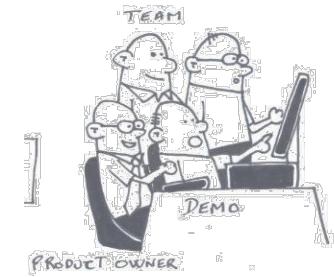
Démontre le travail « terminé » et répond aux questions sur l'incrément.

➤ L'ENSEMBLE DU GROUPE CONVIENT DE CE QU'IL FAUT FAIRE POUR LA SUITE, DE SORTE QUE LA REVUE DE SPRINT FOURNISSE UNE CONTRIBUTION PRÉCIEUSE AUX RÉUNIONS SUBSÉQUENTES DE PLANIFICATION DE SPRINT



PRINCIPES

- BASÉ SUR CETTE ANALYSE, ET EN CONSIDÉRANT LES CHANGEMENTS APPORTÉS AU PRODUCT BACKLOG DURANT LE SPRINT, LES PARTICIPANTS COLLABORENT POUR DÉTERMINER LES PROCHAINS ÉLÉMENTS QUI POURRAIENT ÊTRE FAITS
- CETTE RÉUNION SE VEUT INFORMELLE, LA PRÉSENTATION DE L'INCRÉMENT EST DESTINÉE À SUSCITER DES RÉACTIONS ET À FAVORISER LA COLLABORATION
- LE RÉSULTAT DE LA REVUE DE SPRINT EST UN PRODUCT BACKLOG RÉVISÉ QUI DÉFINIT LES ÉLÉMENTS ENVISAGÉS POUR LE SPRINT SUIVANT



PRÉPARATION ET DÉROULÉ

➤ PRÉPARER LA DÉMONSTRATION

Imaginer un scénario pour dérouler les éléments qui seront présentés

➤ EFFECTUER LA DÉMONSTRATION

Seuls les cas d'utilisation complètement finis sont présentés.

➤ EVALUER LES RÉSULTATS DU SPRINT

Le Backlog de produit est enrichi en fonction des retours

CONSEILS (1/2)

- **NE PAS MONTRER DE DIAPOS : LA DÉMONSTRATION DOIT PRÉSENTER LE PRODUIT DÉVELOPPÉ AVEC LES ÉLÉMENTS IMPLÉMENTÉS DE BOUT EN BOUT**
- **EXÉCUTION DES TÂCHES MÉTIER : NE PAS PARLER DE LA FAÇON DONT LES CHOSES ONT ÉTÉ IMPLÉMENTÉES. SE FOCALISER SUR UNE PERSPECTIVE UTILISATEUR**

Se concentrer sur le feedback : aller d'un bout à l'autre d'une tâche métier (présenter un user story)

- Ne montrez pas les tas de corrections de bugs mineurs et de fonctionnalités insignifiantes. Mentionnez-les mais ne les montrez pas, ça prend trop de temps et ça écarte l'attention des choses plus importantes
- A noter : le feedback utilisateur peut induire de nouvelles exigences qui devront être confrontées aux existantes pour ajuster les priorités

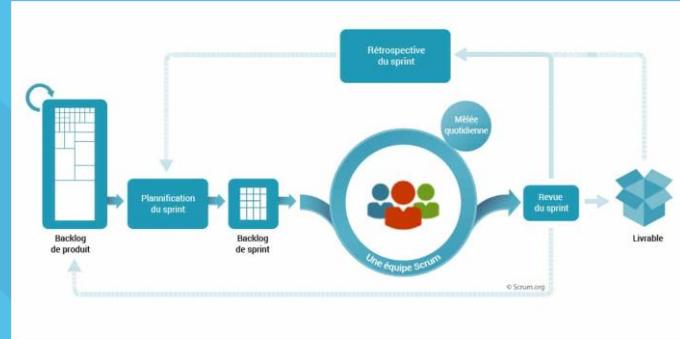


CONSEILS (2/2)

- GARDEZ UN RYTHME ÉLEVÉ, AUTREMENT DIT ESSAYEZ DE PRÉPARER UNE DÉMONSTRATION RYTHMÉE PLUTÔT QU'UNE JOLIE DÉMONSTRATION
- CONCENTREZ-VOUS SUR « CE QUE NOUS AVONS FAIT » PLUTÔT QUE SUR « COMMENT NOUS L'AVONS FAIT »
 - Ne montrez pas les tas de corrections de bugs mineurs et de fonctionnalités insignifiantes. Mentionnez-les mais ne les montrez pas, ça prend trop de temps et ça écarte l'attention des choses plus importantes
 - Attention : le feedback utilisateur peut induire de nouvelles exigences qui devront être confrontées aux existantes pour ajuster les priorités



LA RÉTROSPECTIVE



PRINCIPES DE LA CÉRÉMONIE

SQLI 2019

➤ OBJECTIF :

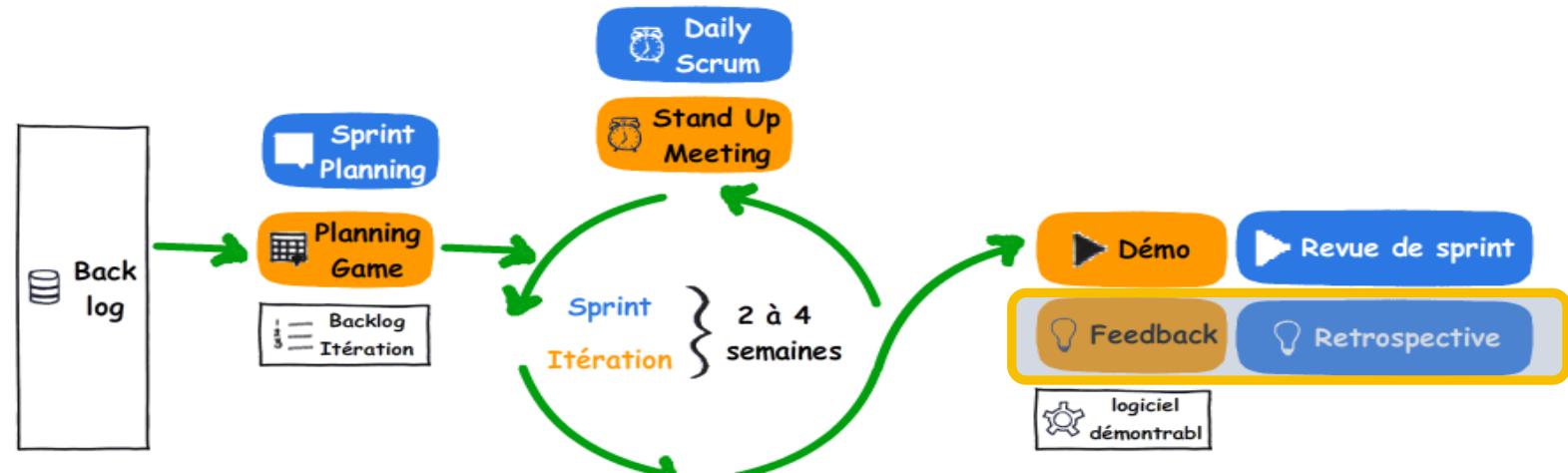
Inspecter le Sprint passé et créer un plan d'améliorations à mettre en place dès le Sprint suivant

➤ DURÉE :

3h pour un Sprint de 1 mois sinon plus court si le Sprint est plus court

➤ PARTICIPANTS :

Equipe de développement, Product Owner, Scrum Master.



L'OBJECTIF

➤ LE BUT DE LA RÉTROSPECTIVE DE SPRINT EST

D'inspecter la manière dont le dernier Sprint s'est déroulé en ce qui concerne **les personnes, les relations, les processus et les outils.**

D'**identifier et ordonner** les éléments majeurs qui se sont bien déroulés et les améliorations potentielles

De **créer un plan pour améliorer** les processus de travail de l'équipe Scrum

➤ ET CÉLÉBRER !

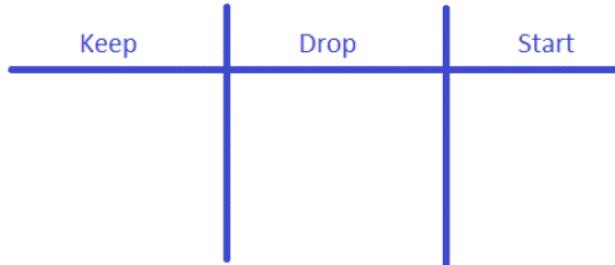


COMMENT L'ANIMER

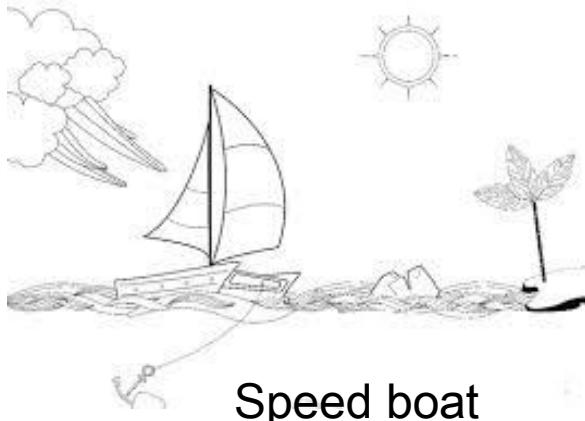
- LORS DE LA RÉTROSPECTIVE, TOUTE L'ÉQUIPE SCRUM DOIT ÊTRE PRÉSENT
- CHAQUE MEMBRE ÉCRIT SES COMMENTAIRES ET LES PRÉSENTE À TOUR DE RÔLE
- LE SCRUM MASTER ENGAGE L'ÉQUIPE À TROUVER DES MOYENS CONCRETS POUR AMÉLIORER CE QUI DOIT L'ÊTRE ET D'EN APPLIQUER LES PLUS PERTINENTS DÈS LE PROCHAIN SPRINT

Points positifs	Points améliorables	Comment améliorer

QUELQUES EXEMPLES

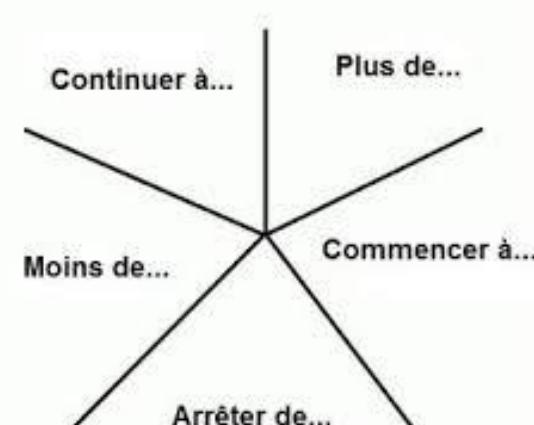


Keep / Drop / Start



Speed boat

Etoile de mer

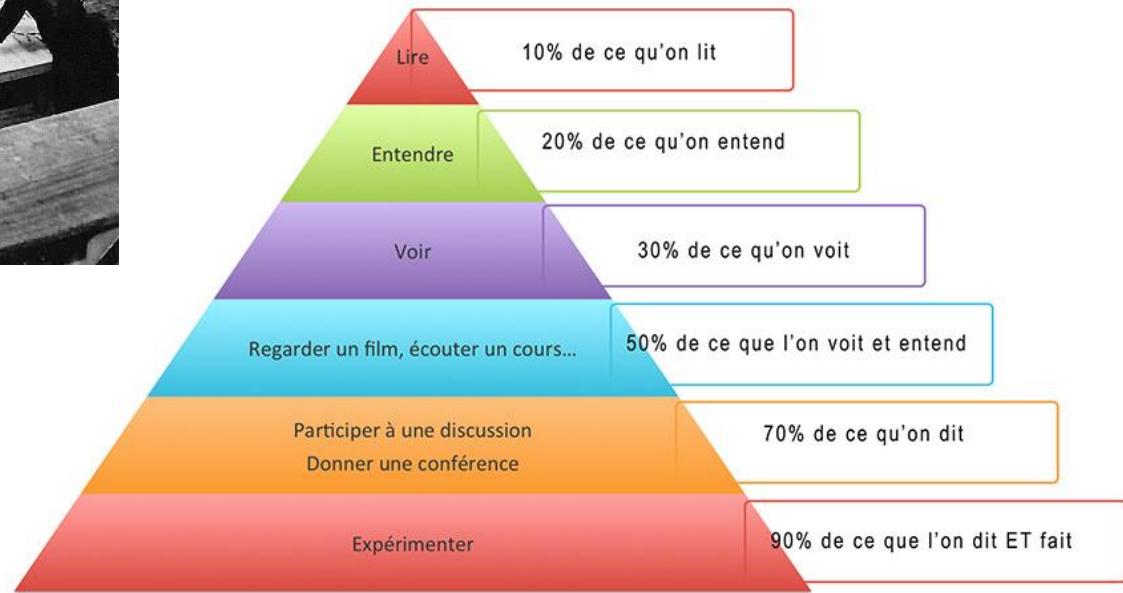


Blanche neige & les 7 nains

MANAGEMENT VISUEL & INDICATEURS

ON COMPREND MIEUX QUAND ON VOIT

S Q L I 2 0 1 9



LE BACKLOG PRODUIT VISUEL

➤ OBJECTIFS

Avoir une vue exhaustive du Backlog Produit
Suivre l'avancement du Produit

➤ COMMENT

Tableau à plusieurs colonnes représentant la vie de l'élément :

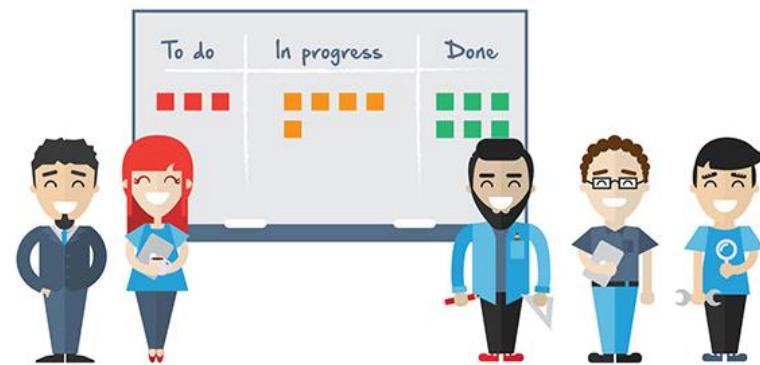
- **A faire, en cours, (en validation), fait**

Les éléments sont mis au mur sur des cartes

On peut indiquer l'équipe en charge si plusieurs équipes

Toute information pertinente est bonne à mettre

- **Blocages, contraintes...**



LE SCRUM BOARD (TASK BOARD)

➤ OBJECTIFS

Partager l'avancement de l'équipe durant le sprint

Lieu d'échanges pour la mêlée quotidienne



➤ COMMENT

Tableau à 4 colonnes : éléments, à faire, en cours, fait

Les éléments du Backlog d'itération avec leurs tâches correspondantes

Pour chaque carte/post-it de tâche, on trouve

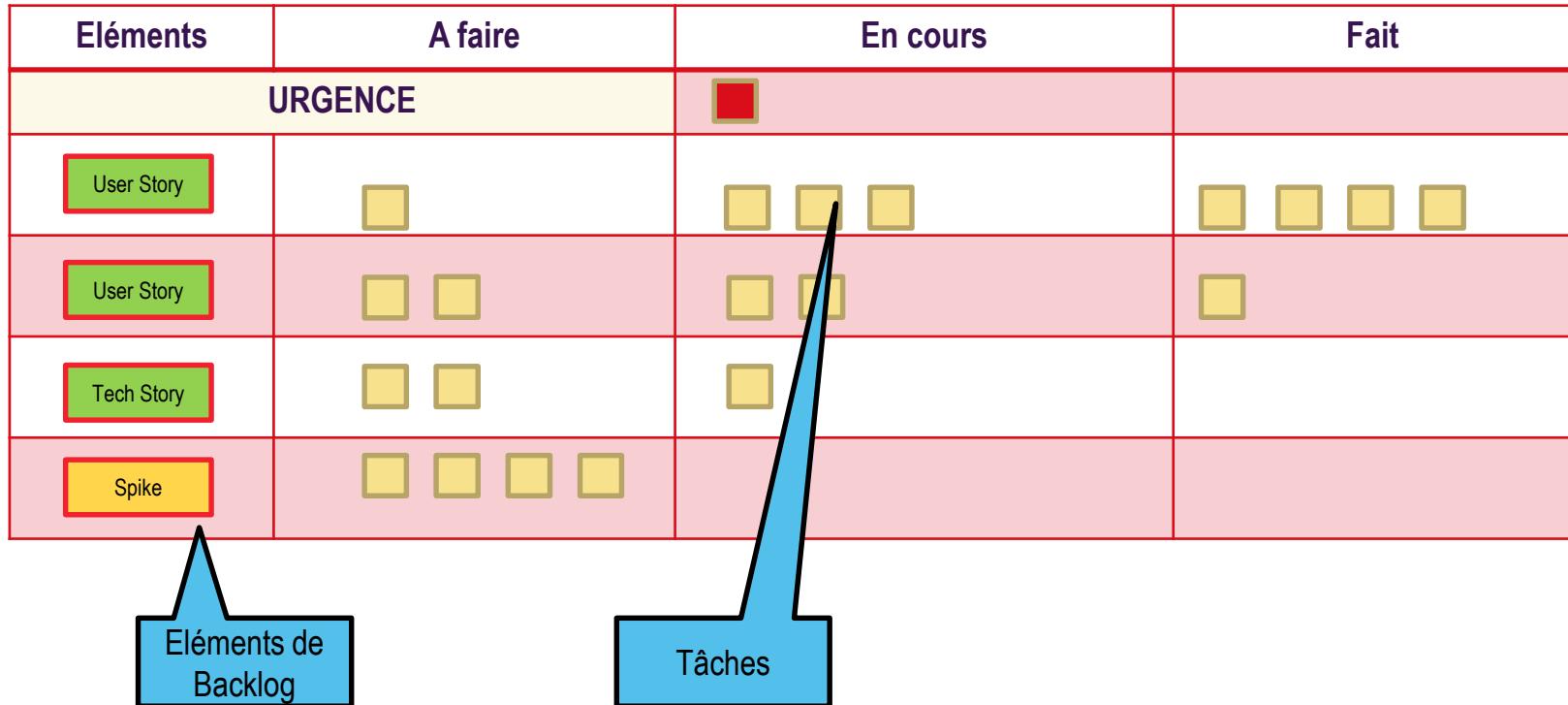
- La référence à l'élément de Backlog
- Le libellé
- Le temps estimé

➤ QUAND LA TÂCHE EST DÉMARRÉE (COLONNE "À FAIRE"), ON AJOUTE

- Le nom du développeur
- Les blocages éventuels
- Le Reste à faire

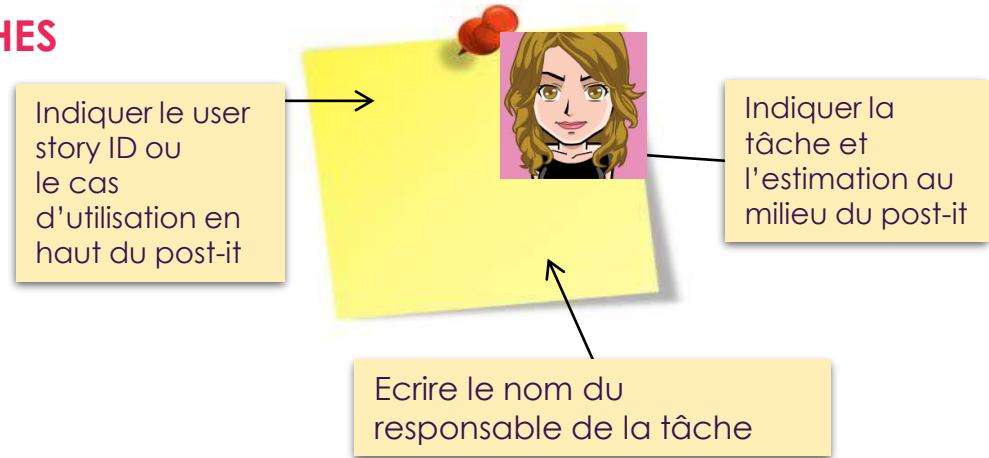
LE SCRUM BOARD (TASK BOARD)

SQLI 2019

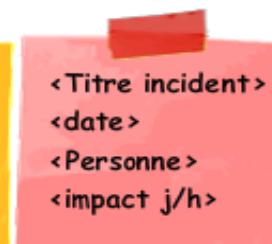
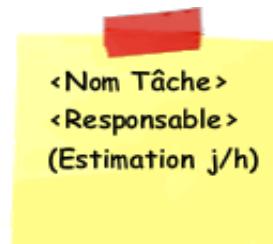
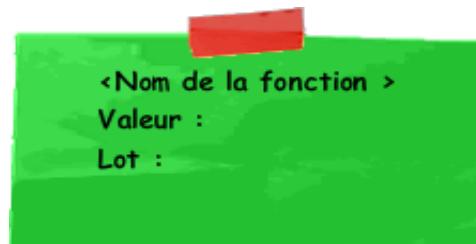


LES POST-IT

➤ UN POST-IT POUR TOUTES LES TÂCHES



➤ CHOISIR UN CODE COULEUR STANDARDISÉ POUR LA GESTION DES POST-ITS.



Elément de backlog

Tâche

Tâche imprévue

Incident / Problème

LE SUIVI DU SPRINT

- A TOUT MOMENT, LE TRAVAIL GLOBAL RESTANT À FAIRE POUR ATTEINDRE L'OBJECTIF PEUT ÊTRE CALCULÉ.

Le Product Owner suit la somme du travail restant à faire au moins à chaque Revue de Sprint.

Le Product Owner compare cette quantité au travail restant à faire lors des Revues de Sprint précédentes pour évaluer les progrès vers l'achèvement des travaux prévus dans le temps souhaité pour l'objectif.

- CETTE INFORMATION EST RENDUE TRANSPARENTE POUR TOUTES LES PARTIES PRENANTES.

- DIVERSES TECHNIQUES DE PROJECTIONS DE TENDANCES POUR ANTICIPER L'AVANCEMENT :

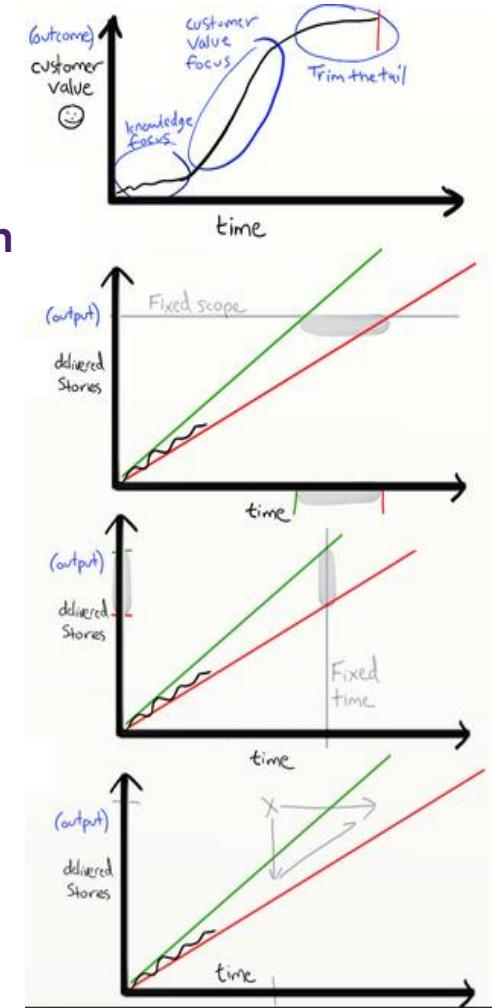
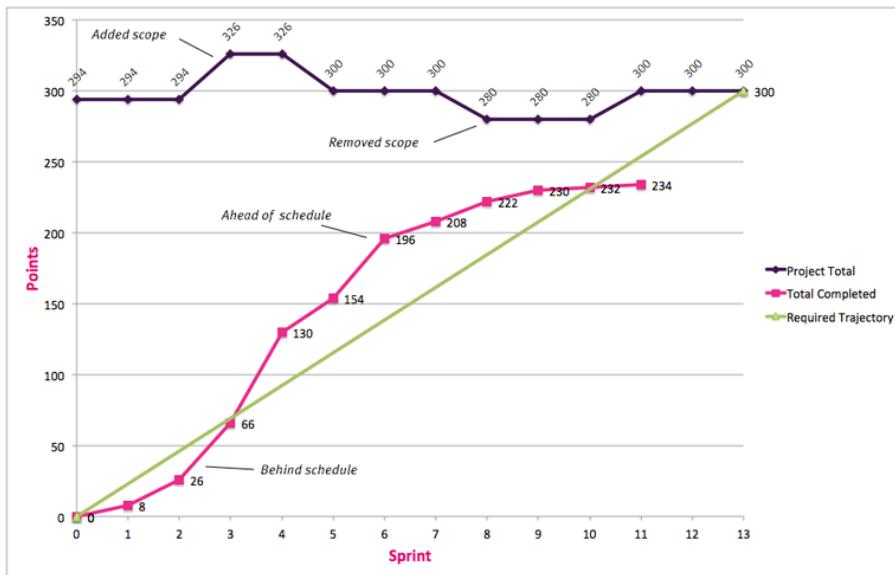
- burn-downs,
- burn-ups, ou
- flots cumulatifs.

BURN-UP

> DONNE LA VISIBILITÉ DE L'AVANCEMENT DU PROJET.

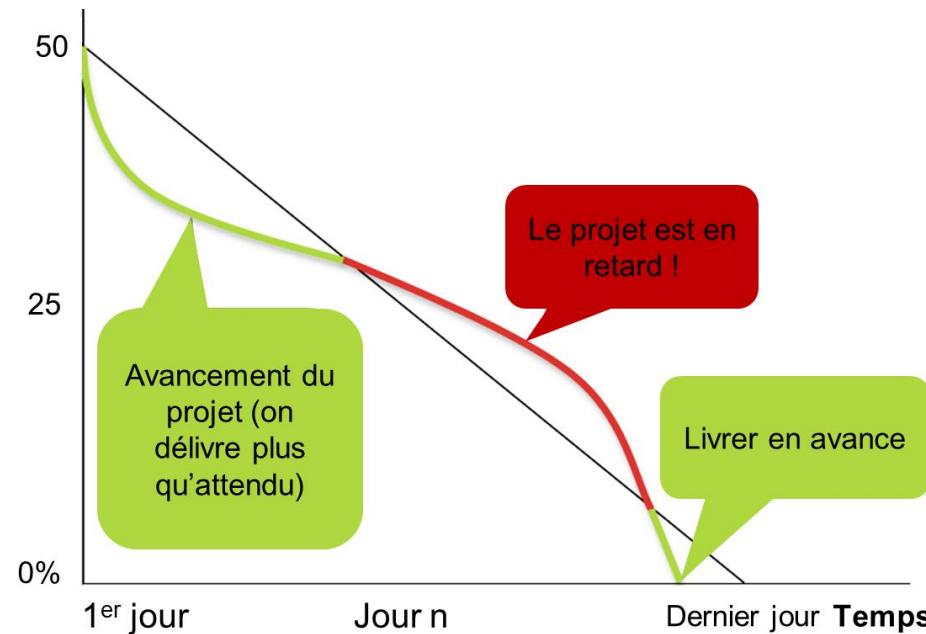
Efficacité de la communication visuelle et de la dimension collective au niveau projet.

Client Burnup Chart



BURN-DOWN

- PRÉSENTE L'ESTIMATION DU TRAVAIL À FAIRE (EN US, POINTS OU HEURES)



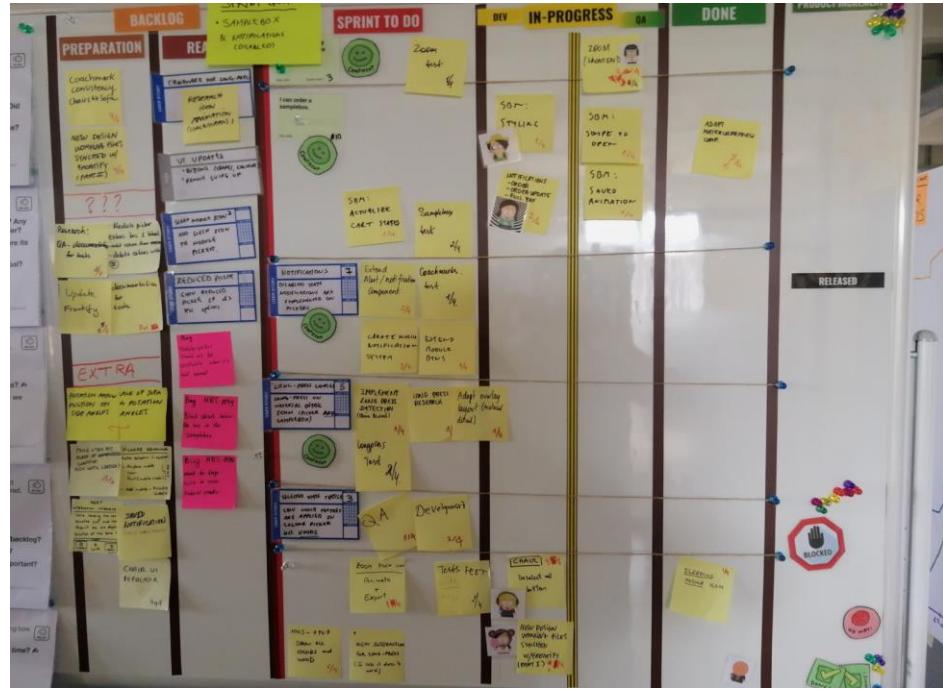
RADIATEUR D'INFORMATION

- L'ÉQUIPE TRAVAILLE DANS LA MÊME PIÈCE ET PARTAGE TOUTES LES INFORMATIONS NÉCESSAIRES AU PROJET – LIEU DU DAILY SCRUM.
- LE RADIATEUR D'INFORMATION EST VISIBLE POUR TOUTE L'ÉQUIPE
mais aussi pour les différents intervenants (architectes, commerciaux, managers, ...)
- IL CONTIENT AU MINIMUM LE BURNDOWN CHART ET L'AVANCEMENT DU SPRINT EN FONCTIONNALITÉS « TERMINÉES »
- IL PEUT CONTENIR ÉGALEMENT :
 - Les risques / Problèmes
 - Les objectifs du projet / du Sprint
 - Les éléments du Backlog en attente
 - Un Planning Release
 - Des indicateurs de qualité
 - Le moral de l'équipe
 - Toutes autres informations utiles au projet



EXAMPLES

SQLI 2019





CE QUI EST IMPORTANT EST VISIBLE

➤ **DESCRIPTION DU PRODUIT**

- Diagramme du système, des cas d'utilisation
- Représentation de l'architecture
- Maquettes IHM
- Un extrait du document de vision
- Extrait des backlogs

➤ **DESCRIPTION DE L'ORGANISATION DU PROJET**

- Nom des membres
- Processus de développement
- Règles de travail
- Définition du "Terminé" ou "Done"

➤ **ORGANISATION OPÉRATIONNELLE DU PROJET**

- Task board (Kanban)
- Courbes d'avancement projet
- Jalons projet
- Tableau des risques, problèmes et incidents
- Idées d'amélioration



Pause



VIVRE SCRUM !





SIMULATION DE SCRUM

S Q L I 2 0 1 9



➤ LEGO 4 SCRUM

20

KANBAN



C'EST QUOI ?

- MÉTHODE NON ITÉRATIVE BASÉE SUR LE FLUX TIRÉ CONTINU

- **KANBAN =**

terme japonais signifiant panneau ou fiche cartonnée

Chaque sujet est inscrit sur une carte Kanban et parcours le flux des activités

- ON MESURE LE TEMPS DE CYCLE (TEMPS MIS ENTRE L'ENTRÉE ET LA SORTIE DANS LE FLUX) ET LE TEMPS DE TRAVAIL EFFECTIF. EN TRAÇANT LA DIFFÉRENCE, ON EN TIRE DES ENSEIGNEMENTS POUR OPTIMISER LE FLUX

- ON LIMITÉ LE WORK IN PROGRESS (WIP) À CHAQUE ÉTAPE POUR FIABILISER LE PROCESSUS

- **ON NE LAISSE PAS S'INSTALLER UN GOULOT D'ÉTRANGLEMENT**

Dès qu'une limite est atteinte, les membres de l'équipe disponibles vont aider à enlever le point de blocage dans le processus

Pour définir une limite pertinente, il faut expérimenter



6 PRATIQUES (1/2)

Visualiser le flux



Avez-vous un flux des travaux ?
Marquer une étape pour chaque phase

Limiter le Work in Progress (WIP)



Où s'entassent les activités ?
Définir une limite pour le WIP
Attacher une limite de WIP à chaque étape

Gérer le Flux



Normaliser le délai de mise en œuvre

On utilise un système pointilleux qui récompense les produits finis et punit les non-finis

6 PRATIQUES (2/2)

Mettre en place des boucles de Feedback



Regardez-vous régulièrement vos travaux et vos processus ?

Ayez des mêlées debout et des rétrospectives

Rendre explicites les règles de processus



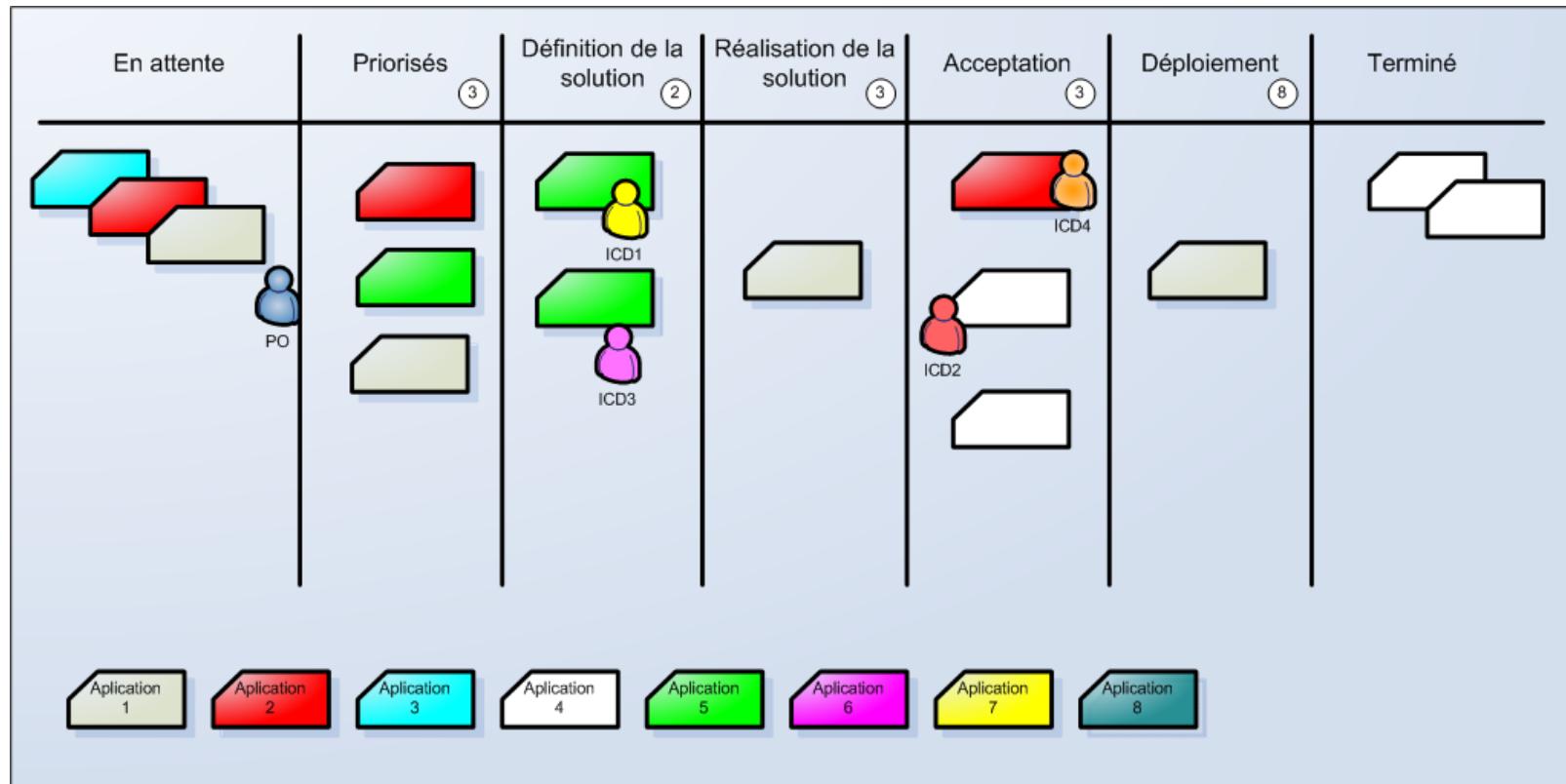
Rendre facile aux personnes de faire les bonnes choses, et de les faire bien

S'améliorer collaborativement



Utiliser des métriques et des modèles pour s'améliorer en permanence

EXEMPLE



SCRUM OU KANBAN ?

➤ AVEC SCRUM, LES ITÉRATIONS ONT DES DURÉES FIXES

2 à 3 semaines en général

Ce rythme impulsé fait partie de la méthode

Besoin d'ajuster les éléments pour les faire « rentrer »

L'équipe est de taille limitée

➤ EN KANBAN, SEULE COMpte LA PRIORITé, PAS LA TAILLE

On regarde le temps de cycle

On peut séquencer le travail de plusieurs équipes

➤ ON RECOURT AU KANBAN

Pour des équipes très matures

Pour des processus plus complexes

En maintenance

21

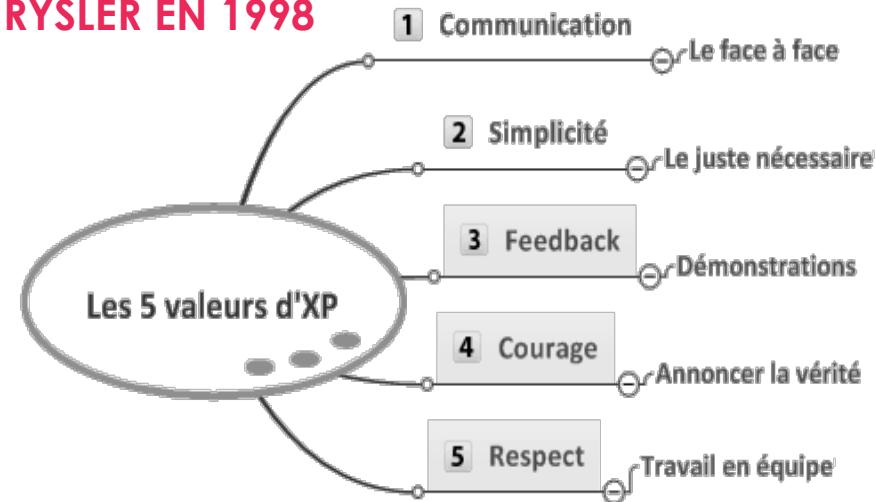
XP



FONDEMENTS (1/4)

- XP EST UN ENSEMBLE DE PRATIQUES D'INGÉNIERIE LOGICIELLE ISSU D'UN PROJET MENÉ PAR KENT BECK, WARD CUNNINGHAM ET RON JEFFRIES POUR CHRYSLER EN 1998

- BASÉ SUR LES 5 VALEURS



- FONCTIONNE EN METTANT EN PRÉSENCE L'ÉQUIPE AUTOUR DE PRATIQUES SIMPLES, AVEC UN FEEDBACK QUI LUI PERMET :

De visualiser son avancement

D'adapter les pratiques au contexte du projet

FONDEMENTS (2/4)

➤ LES PRINCIPES DE CETTE MÉTHODE NE SONT PAS NOUVEAUX. L'ORIGINALITÉ DE LA MÉTHODE EST DE LES POUSSER À L'EXTRÊME PUISQUE

La revue de code est une bonne pratique, elle sera faite en permanence (par un binôme)

Les tests sont utiles, ils seront faits systématiquement avant chaque implantation

La conception est importante, elle sera faite tout au long du projet (refactoring)

La simplicité permet d'avancer plus vite, nous choisirons toujours la solution la plus simple

La compréhension est importante, nous définirons et ferons évoluer ensemble des métaphores

L'intégration des modifications est cruciale, nous l'effectuerons plusieurs fois par jour

Les besoins évoluent vite, nous ferons des cycles de développement très rapides pour nous adapter au changement

FONDEMENTS (3/4)

➤ PRATIQUES DE DÉVELOPPEMENT

Conception simple

Remaniement du code (refactoring)

Développements pilotés par les tests (TDD) - "Tests Unitaires Automatisés"

Tests de recette

➤ PRATIQUES DE COLLABORATION

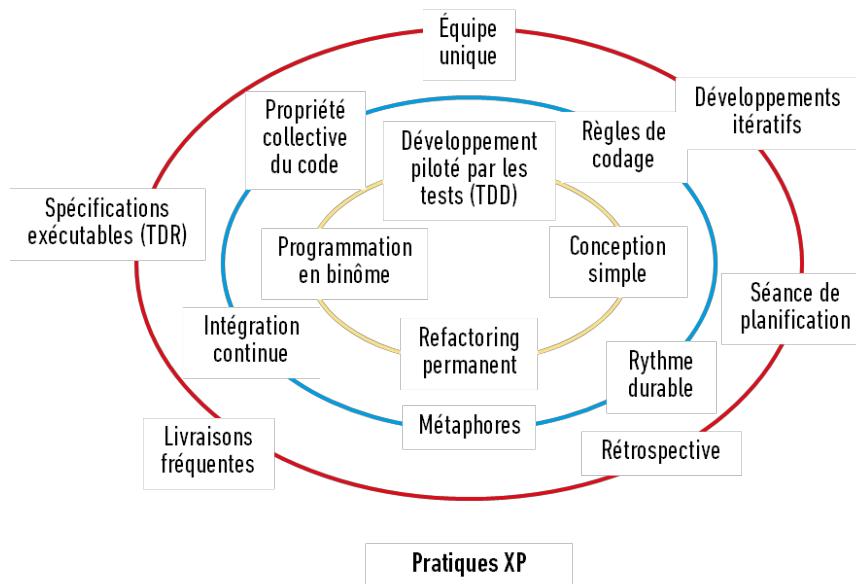
Programmation en binôme

Appropriation du code par l'équipe

Standards de développement

Intégration continue

Utilisation de métaphores



FONDEMENTS (4/4)

➤ PRATIQUES DE GESTION DE PROJET

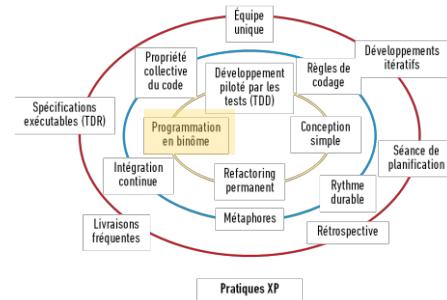
Livraisons fréquentes
Planification itérative
Client sur site
Rythme soutenable

Nom dans XP	Équivalent en Scrum
Équipe complète Le client devrait être présent sur site	Scrum Team Le Product Owner est le représentant du client
Coach d'équipe	Scrum Master
Itération	Sprint
Planning Game	Sprint Planning
Stand-up meeting	Daily Scrum
Démonstration	Sprint Review

➤ CES PRATIQUES STIMULENT

L'esprit d'équipe
La diffusion accélérée des connaissances
L'exigence de qualité

TRAVAIL EN BINÔME



➤ LORS DU CODAGE, DISTINGUER LES DEUX RÔLES

Un **conducteur** (pilote) tient le clavier, il s'occupe de la syntaxe du code, de la compilation et du jeu des tests

Le **navigateur** (copilote) s'occupe de ce que le code doit faire, de la conception et la modularité et la relecture du code

➤ AVANTAGES

Contribue à souder l'équipe

Favorise la propriété collective du code

Vecteur de qualité logicielle (simplicité, modularité, refactoring)

Accélère l'intégration de nouveaux

Développe la pluridisciplinarité

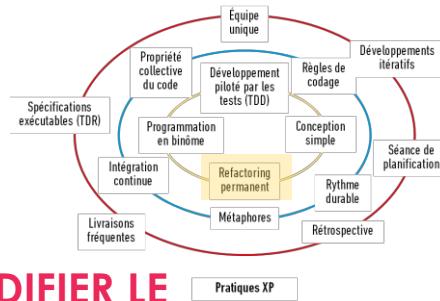
Renforce la dimension collaborative et l'entraide

Retour sur investissement très important



Non systématique : nécessite une forte concentration

REFACTORING



- AMÉLIORATION RÉGULIÈRE DE LA QUALITÉ DU CODE SANS EN MODIFIER LE COMPORTEMENT
- ON RETRAVAILLE LE CODE POUR REPARTIR SUR DE MEILLEURES BASES TOUT EN GARDANT LES MÊMES FONCTIONNALITÉS
- LES PHASES DE REFACTORING N'APPORTENT RIEN AU CLIENT MAIS PERMETTENT AUX DÉVELOPPEURS D'AVANCER DANS DE MEILLEURES CONDITIONS ET DONC PLUS VITE
- NOTE : LES TESTS AUTOMATISÉS ÉVITERONT TOUTES RÉGRESSIONS

Cette pratique est particulièrement nécessaire dans le cadre d'une conception itérative, car il faut régulièrement généraliser pour éviter des doublons, factoriser, etc.

CONCEPTION SIMPLE

➤ ÉVITER LE SYNDROME DU "AU CAS OÙ"

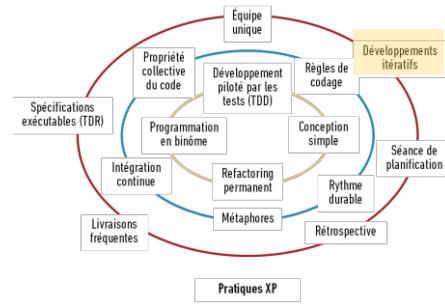
Envisager les prochaines évolutions ferait perdre du temps sans avoir la garantie d'un gain ultérieur

➤ PLUS L'APPLICATION EST SIMPLE, PLUS IL SERA FACILE DE LA FAIRE ÉVOLUER LORS DES PROCHAINES ITÉRATIONS

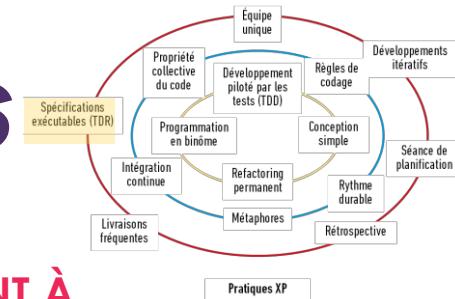
➤ DE MÊME, LA DOCUMENTATION DOIT ÊTRE MINIMALE

On préfèrera un programme simple qui nécessite peu d'explications à un système complexe

➤ UN CODE LISIBLE PERMET DE RÉDUIRE, VOIRE DE S'AFFRANCHIR DE LA DOCUMENTATION



SPÉCIFICATIONS EXÉCUTABLES



➤ TEST DRIVEN REQUIREMENT (TDR) EST UNE TECHNIQUE CONSISTANT À

Rédiger les tests en tant que première étape de la formalisation du besoin
 Détailler l'exigence à travers la description des cas de test

➤ AVANTAGES

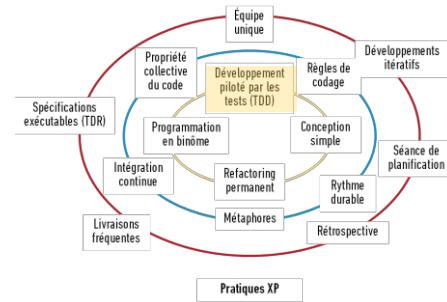
Fluidifie la phase amont de recueil de besoin et de spécifications fonctionnelles
 Mutualise les étapes de spécifications et de tests fonctionnels

➤ MISE EN ŒUVRE

S'appuie sur les échanges entre le Product Owner et l'équipe qui réalise
 Intégration des testeurs à la phase de définition pour plus de valeur ajoutée
 Couplé à un outil type Fit, FitNesse ou GreenPepper, chaque exemple devient un cas de test automatique

- Fluidifie la phase amont de recueil de besoin et de spécifications fonctionnelle
- Mutualisation des étapes de spécifications et de tests fonctionnels

DÉVELOPPEMENT PILOTÉ PAR LES TESTS (TDD)



➤ TEST DRIVEN DEVELOPMENT

La rédaction des tests est la première étape de la formalisation du codage

Axe d'orientation des développements : chaque élément de code n'est écrit que pour permettre de passer le test

À chaque modification du code on :

- Lance tous les tests écrits par tous les développeurs
- Sait immédiatement si quelque chose ne fonctionne plus

Les tests sont conservés et maintenus jusqu'à la fin du projet (tant que la fonctionnalité est requise)

Le code est remanié (supprimer les duplications, améliorer sa lisibilité...)

➤ AVANTAGES

Interaction entre les cas de test et la compréhension fine des besoins fonctionnels

La compétence de tests est couplée à celle de développement

Adhérence du code aux tests

- Intégration forte de la qualité logicielle

Le code écrit en TDD est plus maintenable, plus découpé

Sécurise le développement (par exemple, pour quelqu'un qui ne connaît pas une partie du code)

DÉVELOPPEMENT PILOTÉ PAR LES TESTS (TDD)

➤ MISE EN ŒUVRE

Apprentissage de la démarche

S'appuie sur les échanges entre le Product Owner et l'équipe qui réalise
Interaction testeur / codeur

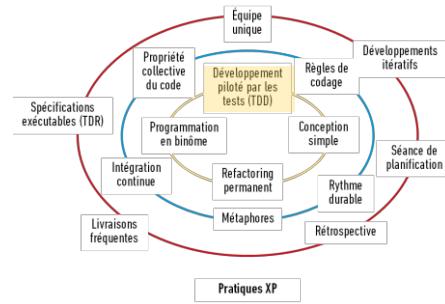
➤ OUTILLAGE

xUnit

Selenium (tests d'interface)

Couplé à un système d'intégration continue

➤ L'ESSAYER, C'EST L'ADOPTER !



RED-GREEN-REFACTOR

1. Ecrire un test pour la nouvelle fonctionnalité

Le test DOIT échoué : la fonctionnalité n'est pas encore codée !

2. Ecrire du code pour faire passer le test

Le code peut être de mauvaise qualité, l'important est de rendre le test OK !

3. Vérifier que le test passe maintenant

4. Refactoriser le code

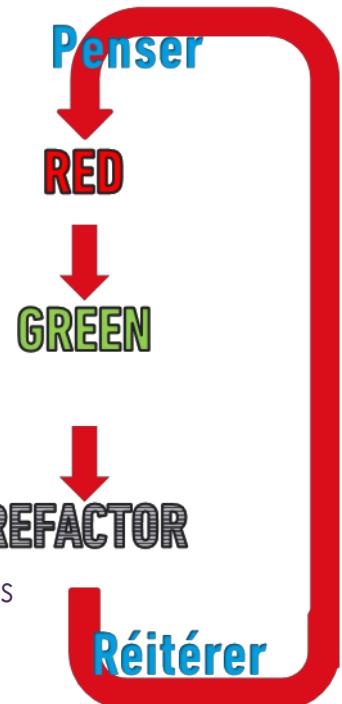
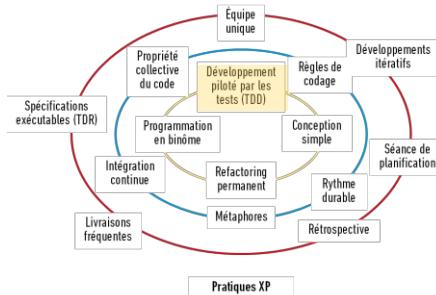
Supprimer les duplications

Faire attention aux noms de méthodes et de variables

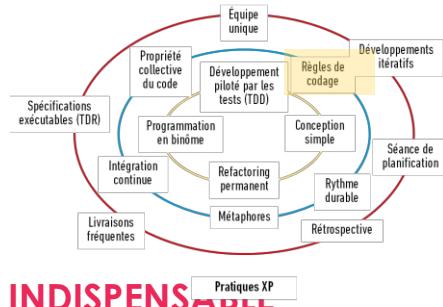
5. Repasser les tests pour vérifier

6. Réitérer avec un nouveau test

- “On s'est aperçu que les étudiants qui testaient d'abord écrivent en moyenne plus de tests et à leur tour, les étudiants qui écrivent plus de tests tendent à être plus productifs.” IEEE Transactions on Software Engineering, 31 Janvier 2005



RÈGLES DE CODAGE



➤ PUISQUE TOUS LES DÉVELOPPEURS INTERVIENNENT SUR TOUT LE CODE, IL EST INDISPENSABLE D'ESTABLIR ET DE RESPECTER DES NORMES

➤ LES RÈGLES DÉFINIES SONT

Format du code

Design patterns à utiliser ou interdits

Convention de nommage, utilisation de normes reconnues, etc.

➤ AVANTAGES

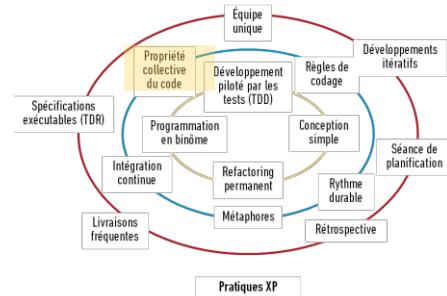
Permet à tout le monde d'avoir un code familier (même si ce l'on n'est pas celui qui l'a écrit)

Favorise l'appropriation collective

➤ LE CONTENU DES NORMES DE DÉVELOPPEMENT N'EST PAS CE QU'IL Y A DE PLUS IMPORTANT

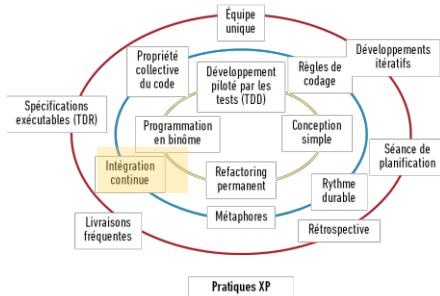
➤ LE FAIT QUE TOUS LES DÉVELOPPEURS DE L'ÉQUIPE SUIVENT LE MÊME STANDARD EST LE PLUS IMPORTANT

APPROPRIATION COLLECTIVE DU CODE



- L'ÉQUIPE EST COLLECTIVEMENT RESPONSABLE DE L'APPLICATION
- CHAQUE DÉVELOPPEUR PEUT FAIRE DES MODIFICATIONS DANS TOUTES LES PORTIONS DU CODE, MÊME CELLES QU'IL N'A PAS ÉCRITES
- LES TESTS DIRONT SI QUELQUE CHOSE NE FONCTIONNE PLUS
- NOTE : LA PROGRAMMATION PAR BINÔME RENFORCE CET ASPECT

INTÉGRATION CONTINUE



➤ TECHNIQUE CONSISTANT À VÉRIFIER POUR CHACUNE DES MODIFICATIONS DE CODE SOURCE QU'ELLE NE PRODUIT PAS DE RÉGRESSION DES DÉVELOPPEMENTS EXISTANTS

Lorsqu'une tâche est terminée, les modifications sont immédiatement intégrées dans le produit complet

Chacun des membres intègre au minimum une fois par jour

Chaque intégration à la "baseline" est conditionnée par une vérification automatisée

On évite ainsi la surcharge de travail liée à l'intégration de tous les éléments avant la livraison

➤ LES TESTS FACILITENT GRANDEMENT CETTE INTÉGRATION

Quand tous les tests passent, l'intégration est terminée

➤ PRÉREQUIS

Un point d'entrée unique dans le référentiel, à jour et accessible

Automatisation des tests unitaires

Automatisation du processus de build

INTÉGRATION CONTINUE – OUTILLAGE DE BASE

SQLI 2019

➤ OUTILS DE GESTION DE CONFIGURATION

Subversion, Git, etc.

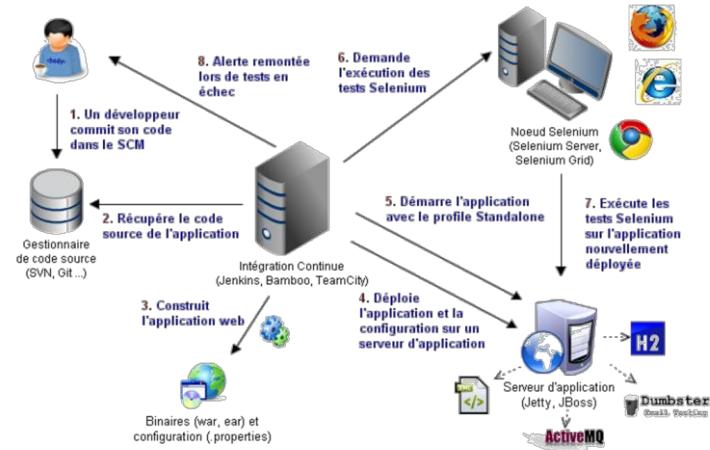
➤ OUTILS DE GÉNÉRATION

Make, Ant, Maven

➤ DIVERS

Qualimétrie – Sonar, splint

Préparation de la production – Mise en paquets Linux (deb, rpm), génération d'un installateur Windows



INTÉGRATION CONTINUE – OUTILLAGE DES TESTS

➤ **XUNIT (JUNIT, CPPUNIT, PHPUNIT...) – FAMILLE DE FRAMEWORK DES TESTS UNITAIRES POUR PLUSIEURS LANGAGES**

➤ **FITNESSE – AUTOMATISATION DES TESTS D'ACCEPTANCE**

Tableaux d'entrées et de sorties attendues, accessibles aux non-techniques (clients, MOA...)
FitNesse possède un wiki intégré et son propre système de versioning
L'unique outil open source gratuit offrant ce type de fonctionnalités

➤ **SELENIUM – AUTOMATISATION DES TESTS D'APPLICATIONS WEB**

Solution open-source de tests fonctionnels des interfaces utilisateurs d'applications web :

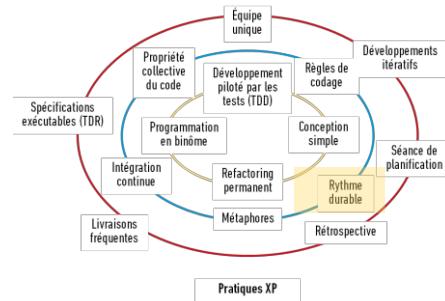
- Test de la compatibilité avec des environnements client hétérogènes (browser, OS, etc.)

➤ **GREENPEPPER**

Moteur de test open source s'utilisant avec un wiki externe : Confluence (payant) ou XWiki (Open Source)

Semblé mort depuis l'automne 2013

RYTHME DURABLE



➤ UNE ÉQUIPE DOIT POUVOIR TRAVAILLER CONTINUELLEMENT SUR LE MÊME RYTHME

➤ SI LE RYTHME EST SUPÉRIEUR

- La discipline décline
- Les développeurs fatigués font des erreurs
- Des raccourcis sont pris
- La qualité est dégradée
- La dette technique s'accumule

➤ UN RYTHME SOUTENABLE PEUT ALTERNER DE PETITES ITÉRATIONS AVEC DES ÉVÉNEMENTS

- Minuteur : Pomodoro technique
- Pauses sans occupation : slack time

LIVRAISONS FRÉQUENTES

➤ LIVRER TÔT ET SOUVENT

L'intégration continue et les tests automatiques réduisent considérablement le coût de livraison

➤ INTÉRÊT

Lever tôt les problèmes d'environnement et d'intégration

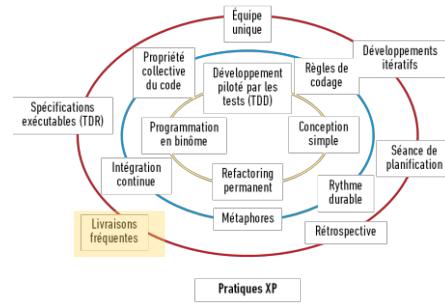
Découper la complexité du projet en morceaux plus faciles à appréhender par l'équipe (facteur de concentration plus important sur chaque sujet)

Créer de la valeur "concrète" le plus souvent possible

Faciliter le partage avec le client et le recueil du feedback

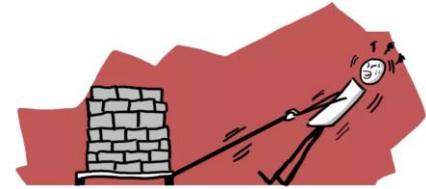
Faciliter ainsi la "matérialisation progressive" et les retours sur les fonctionnalités déjà développées

Créer un rythme, du mouvement et entraîner la motivation



LA DETTE TECHNIQUE

QU'EST CE QUE C'EST ?



➤ ORIGINE & DÉFINITION DU TERME :

Concept (#technical debt) introduit par Ward Cunningham

Comme une dette financière, plus on retarde son règlement, plus les intérêts à payer sont importants.

Définition du Glossaire Scrum :

« A shared understanding of the expectations that software must live up to in order to be releasable into production, with a purpose of providing transparency over the software created. Managed by the development team. »

➤ LES SOURCES DE LA DETTE TECHNIQUE :

Le non respect des conventions de codes #clean code

Le code hérité (« legacy code »): il s'agit code sans test unitaires ou code non maintenu par son auteur ou code qui « fait peur » aux dévs 😊

Vision Produit, long terme, absente : on a une vision court terme, une vision projet.

QUELQUES EXEMPLES

➤ **TOUT CE QUI GÉNÈRE DES DIFFICULTÉS DE MAINTENANCE**

➤ **QUALITÉ DU CODE**

Appréhension d'un design, nommage flou ou inappropriée

➤ **QUANTITÉ DE CODE**

Il peut y avoir trop de code pour la fonction à réaliser

➤ **TESTS : QUANTITÉ, QUALITÉ, PERTINENCE**

➤ **DOCUMENTATION**

Inexistante, décorrélée, peu pertinente



QUELQUES SYMPTOMES

- L'insatisfaction des utilisateurs lors de la revue ou de la livraison (non-acceptation)
- Un ratio de plus en plus important entre la durée de la livraison et la complexité d'un item
- Le nombre de bugs et d'incidents relevés sur une période donnée
- La durée des investigations sur des produits, composants ou code hérités.
- La dépendance de l'équipe des rôles spécialistes
- Le nombre d'heures supplémentaires que travaille l'équipe de développement subissant de la pression
- Le score que peut donner l'équipe de développement quant au plaisir qu'elle a à travailler sur le projet

D'OU VIENNT-ELLE ?

Elle intervient en cas de non respect des règles et conventions ou en l'absence d'optimisation.

Pour l'éviter il faut avoir une **vision Produit, donc une vision à long terme**, et non pas une vision Sprint ou Projet. « You don't build code, you grow code ».

La dette technique est une **responsabilité commune** – toute l'équipe devrait se préoccuper.

Elle induit une **augmentation du coût final du produit**. Comme une dette financière, plus on retarde son règlement, plus les intérêts à payer sont importants.

Si on attend trop, on finit par se retrouver avec un produit pour lequel tous les coûts sont absorbés par la maintenance. Le produit ne peut plus évoluer.



Souvent la dette technique n'est pas visible. Elle est détectée lors d'un audit du code par exemple ou seulement lorsque la stabilité de l'application est dégradée.

COMMENT LA PRÉVENIR ?

Respecter la « #definition of done » : le « done » est 100% « done » ou il ne l'est pas!

Instaurer et respecter une exigence de qualité de développement au sein de l'organisation : principes SOLID, revue de code, **#TDD, #BDD, #pair programming**, code générique...

Utilisez des outils reconnus dans l'état de l'art du marché pour avoir un code mutualisé, versionné,...

Evitez les « héros » ou les « pompiers » : partagez les connaissances ...

COMMENT S'Y ATTAQUER

➤ AU FIL DE L'EAU

- **Tous, comme des #scouts** : à chaque intervention sur un bout de code, on essaie de l'améliorer même juste un peu.
- **Budgétée & planifiée à chaque itération**, sous la responsabilité du tech-lead.

➤ CRÉNEAUX PRÉCIS AVEC DES OBJECTIFS PRÉCIS

- Toutes les X semaines, l'équipe bloque sa journée sur des tâches de désendettement, ce qui donne en plus l'occasion d'échanger collectivement sur le sujet.
- Organisez le **#refactoring** d'une partie spécifique de l'application si nécessaire.

➤ MISSION DE SAUVETAGE ! Commencez de zéro...

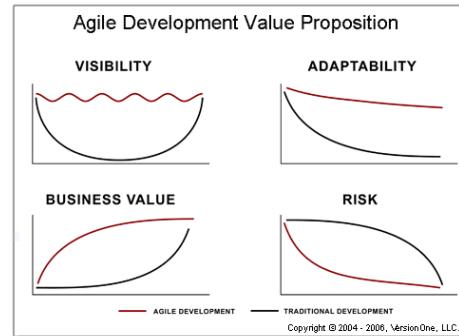


POUR CONCLURE

BÉNÉFICES DE L'AGILITÉ

POUR LE MÉTIER

- Visibilité de l'avancement réel du produit/service (vs avancement du projet)
 - Feedback rapide et possibilité d'interaction (vs effet tunnel et périmètre rigide)
 - Dispose plus tôt et plus vite de livrables utilisables
 - Meilleure adéquation au besoin métier / business
 - Meilleure qualité



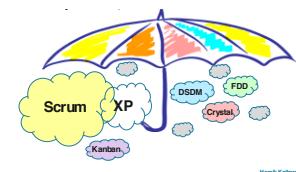
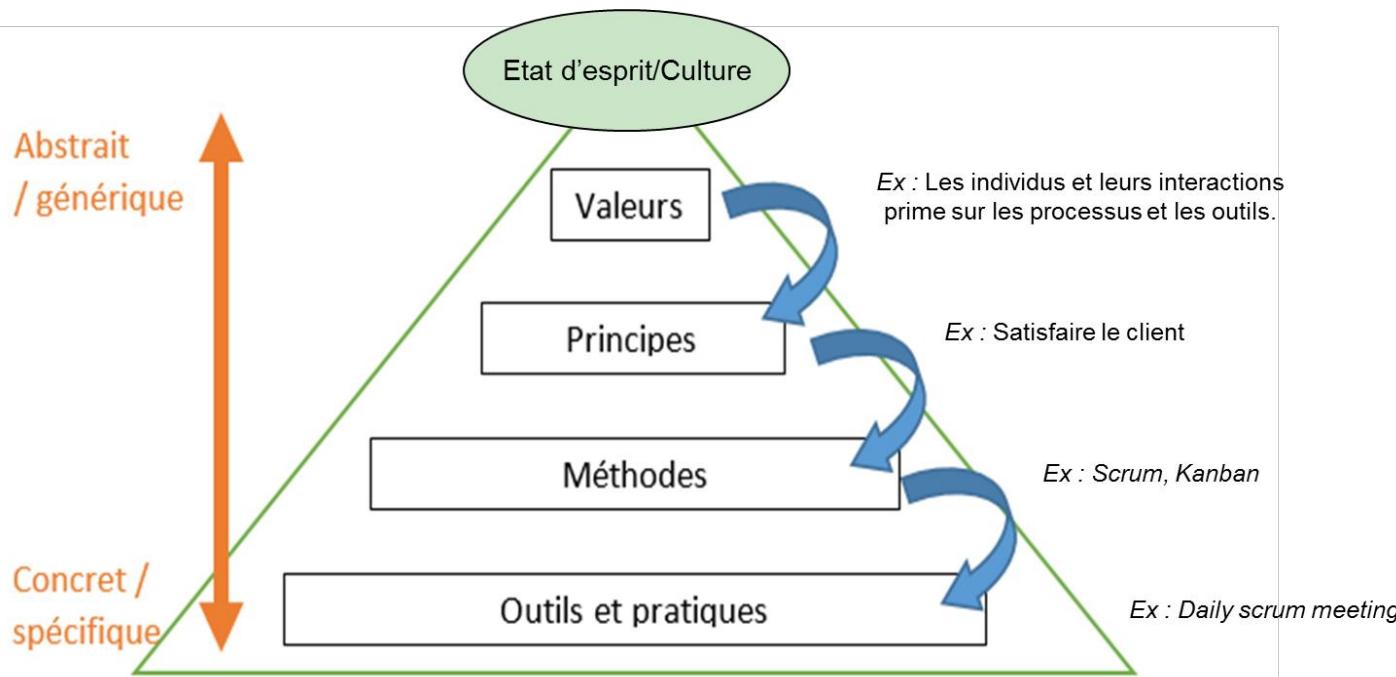
POUR L'ÉQUIPE PROJET

- Développement rapide de la polyvalence des équipes
 - Meilleure affectation des ressources
 - Facteur réel de motivation
 - Synergie d'une équipe motivée qui porte la responsabilité du projet.

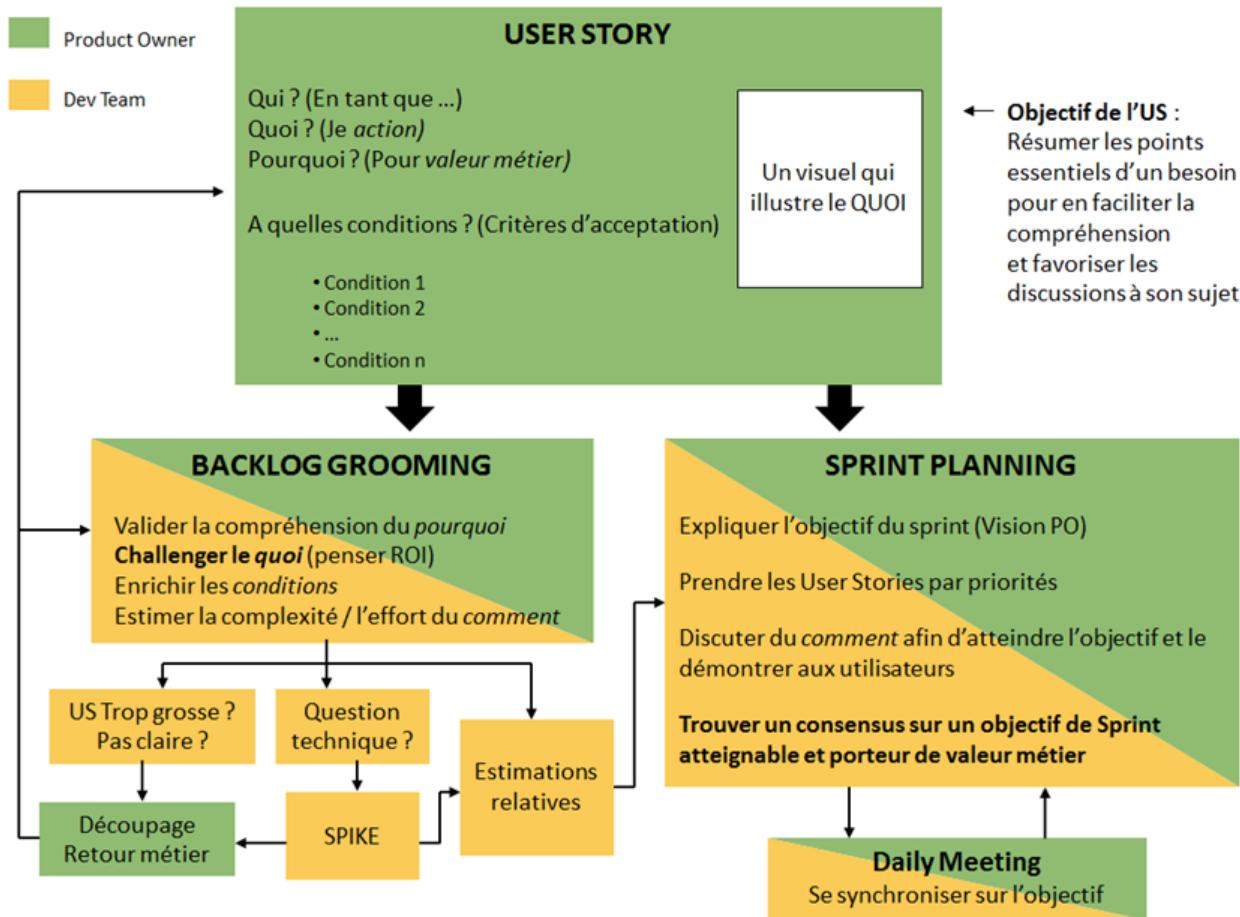
POUR LA HIÉRARCHIE PROJET

- Visibilité de l'avancement réel du produit/service
 - Satisfaction client et relation client collaborative
 - Respect des délais et des couts.

L'AGILITÉ EN RÉSUMÉ



SCRUM EN SYNTHÈSE



FIN DE LA FORMATION

➤ QUESTIONS



➤ RETOUR SUR LES ATTENTES

➤ FEEDBACK – LE ROTI



"**Excellente.** Voila une super réunion dont je vais bénéficier. Ça valait bien plus que le temps qu'on y a passé"



"**Bonne.** Voilà une réunion au dessus de la moyenne. J'ai gagné plus que le temps que j'y ai passé "



"**Juste Moyenne.** Je n'ai pas perdu mon temps, sans plus"



"**Utile** mais ça ne valait pas à 100% le temps que j'y ai passé. J'ai donc perdu du temps "



"**Inutile.** Je n'ai rien gagné, rien appris. J'ai vraiment perdu 1 heure ! "

Jean Claude Grosjean - www.qualitystreet.fr

ANNEXES





GLOSSAIRE

Terme	Définition
BurnDown Chart	Graphique généralement utilisé pour suivre l'avancement du Sprint (horizon court terme). Il permet de suivre l'évolution du travail restant en fonction du temps. Le but consiste donc à atteindre le niveau zéro le plus tôt possible, d'où le terme "Down"
BurnUp Chart	Graphique d'avancement généralement utilisé pour suivre l'avancement d'une Release (horizon moyen ou long terme). Il permet de suivre : <ul style="list-style-type: none">- l'évolution de la quantité de travail terminée en fonction du temps- l'évolution de la quantité de valeur métier terminée en fonction du temps. Le but consiste donc à atteindre la cible (haut du graphique) le plus tôt possible, d'où le terme "Up"
Grooming	Cérémonie Agile : réunion à mis-sprint de préparation du prochain sprint planning meeting . Souvent entre le product owner et le scrum master
Kanban	Méthode basée sur le management visuel des tâches. On appelle aussi « kanban » (avec un 'k' minuscule), le tableau des tâches utilisé pour suivre visuellement l'activité du Sprint
Mêlée	Cérémonie Agile : réunion quotidienne de moins de 15 minutes permettant à la feature team de se synchroniser, identifier les obstacles éventuels et mesurer son avancement sur le Sprint en cours
Quotidienne/ Daily Meeting / Stand-Up Meeting	



GLOSSAIRE

Terme	Définition
Minimum Viable Product (MVP)	Produit minimum apportant de la valeur aux utilisateurs et permettant de valider une hypothèse
Planning Poker	Technique d'estimation collective. Chaque élément du Product Backlog est estimé collectivement en se basant généralement sur une unité relative (Story Point ou taille de tee-shirt). Cette estimation va notamment aider le Product Owner à prioriser son Product Backlog
Product Backlog / Backlog	Liste ordonnancée (priorisée) des besoins (généralement formulés sous forme de User story) du projet
Product Owner (PO)	Représentant des utilisateurs qui porte la vision du produit que l'on souhaite réaliser dans le cadre du projet. Il est responsable du Product Backlog et en interaction directe et quotidienne avec la feature team
Release / Version	Une release est une nouvelle version du produit, livrée aux utilisateurs. Elle est le fruit de plusieurs Sprints
Rétrospective de Sprint	Cérémonie Agile : réunion de travail de la feature team ayant pour but de tirer les leçons du Sprint échu afin de les mettre au profit des suivants. Instance dédiée à l'amélioration continue collective
Scrum	Scrum est processus agile de gestion de projet pour construire un logiciel de façon incrémentale, itérative et adaptative



GLOSSAIRE

Terme	Définition
Scrum Master (SM)	Le scrum master est : <ul style="list-style-type: none">- facilitateur permettant à l'équipe de travailler dans les meilleures conditions- porte parole des développeurs et interlocuteur privilégié du product owner- responsable de l'application de la méthode scrum
Sprint / Itération	Bloc de temps (2 ou 3 semaines, souvent appelé itération), pendant lequel la feature team va créer un incrément du produit potentiellement livrable et qui apporte de la valeur métier
Sprint backlog	C'est le contenu d'un sprint , sur lequel la feature team s'engage. Il contient notamment un extrait du product backlog . Ce contenu est discuté lors du sprint planning meeting
Sprint Planning meeting / Planification de Sprint	Cérémonie Agile : réunion déterminant l'ensemble des choses à réaliser pour un sprint et pour lesquelles l'équipe s'engage
Sprint Review / Revue de Sprint	Cérémonie Agile : réunion de travail consistant à présenter aux parties prenantes les fonctionnalités terminées au cours du sprint afin de recueillir leurs feedbacks. Et à faire le point sur l'avancement global du projet



GLOSSAIRE

Terme	Définition
Tests d'acceptation	Contrat de validation des user story entre le product owner et les développeurs
User Story / Story	Besoin exprimé sous la forme d'un scénario utilisateur Une User Story contient a minima : <ul style="list-style-type: none">- titre qui commence par un verbe d'action à l'infinitif- valeur métier- description : En tant que ..., afin de ..., je veux ...- Test(s) d'acceptation

BIBLIOGRAPHIE – AGILITÉ (1/2)

➤ VÉRONIQUE MESSAGER-ROTA

Gestion de projet agile
Coacher une équipe agile



➤ CHROMATIC

Extreme Programming (Poche)



➤ LAURENT MORISSEAU

Kanban pour l'IT



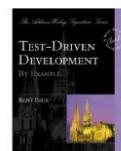
➤ MIKE COHN

Succeeding with Agile
Agile Estimating and Planning



➤ KENT BECK

Test Driven Development



BIBLIOGRAPHIE – AGILITÉ (2/2)

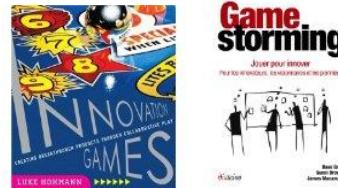
➤ MARY ET TOM POPPENDIECK :

Lean Software Development, An Agile Toolkit
Implementing Lean Software development, From Concept to Cash



➤ ANIMATION – CRÉATIVITÉ

Luke Hohmann - Innovation Games
Gamestorming



➤ ET AUSSI :

Kent Beck et Cynthia Andres - Extreme Programming Explained, embrace change

BIBLIOGRAPHIE - SCRUM

- **KEN SCHWABER ET MARK BEEDLE :**
Agile Software Development With Scrum



- **KEN SCHWABER, CO-FONDATEUR DE SCRUM**
Agile Project Management with Scrum



- **CLAUDE AUBRY**
Scrum - Le guide pratique de la méthode agile la plus populaire



- **SCRUM ET XP DEPUIS LES TRANCHÉES**
Un retour d'expérience sur leur mise en place
<http://henrik-kniberg.developpez.com/livre/scrum-xp/>



QUELQUES SITES ...

- [HTTP://AGILEMANIFESTO.ORG/](http://AGILEMANIFESTO.ORG/)
- [HTTP://WWW.SCRUMALLIANCE.ORG/](http://WWW.SCRUMALLIANCE.ORG/)
- [HTTP://WWW.SCRUM.ORG/](http://WWW.SCRUM.ORG/)
- [HTTP://WWW.XPROGRAMMING.COM/](http://WWW.XPROGRAMMING.COM/)
- [HTTP://WWW.AGILEGAMESFRANCE.FR/INDEX.PHP?TITLE=JEUX](http://WWW.AGILEGAMESFRANCE.FR/INDEX.PHP?TITLE=JEUX)
- [HTTP://WWW.INNOVATIONGAMES.COM/](http://WWW.INNOVATIONGAMES.COM/)
- [HTTP://SCRUMTRAININGSERIES.COM/](http://SCRUMTRAININGSERIES.COM/)
- [HTTP://WWW.LES-TRADUCTEURS-AGILES.ORG/](http://WWW.LES-TRADUCTEURS-AGILES.ORG/)
- [HTTP://HENRIK-KNIBERG.DEVELOPPEZ.COM/](http://HENRIK-KNIBERG.DEVELOPPEZ.COM/)

QUELQUES BLOGS ...

- KEN SCHWABER
[HTTP://WWW.CONTROLCHAOS.COM/](http://WWW.CONTROLCHAOS.COM/)
- JEFF SUTHERLAND
[HTTP://WWW.JEFFSUTHERLAND.COM/SCRUM](http://WWW.JEFFSUTHERLAND.COM/SCRUM)
- SCOTT AMBLER
[HTTP://WWW.AMBYSOFT.COM/](http://WWW.AMBYSOFT.COM/)
- MARTIN FOWLER
[HTTP://MARTINFOWLER.COM/ARTICLES/](http://MARTINFOWLER.COM/ARTICLES/)
- EMMANUEL CHENU
[HTTP://EMMANUELCHENU.BLOGSPOT.COM/](http://EMMANUELCHENU.BLOGSPOT.COM/)
- CLAUDE AUBRY
[HTTP://WWW.AUBRYCONSEIL.COM](http://WWW.AUBRYCONSEIL.COM)

- THIERRY CROS
[HTTP://ETREAGILE.THIERRYCROS.NET](http://ETREAGILE.THIERRYCROS.NET)
- FLORENT LOTHON
[HTTP://WWW.AGILISTE.FR](http://WWW.AGILISTE.FR)
- ALEXANDRE BOUTIN
[HTTP://WWW.AGILEX.FR/](http://WWW.AGILEX.FR/)
- BRUNO ORSIER
[HTTP://BRUNO-ORSIER.DEVELOPPEZ.COM](http://BRUNO-ORSIER.DEVELOPPEZ.COM)
- JEAN-CLAUDE GROSJEAN
[HTTP://WWW.QUALITYSTREET.FR/](http://WWW.QUALITYSTREET.FR/)
- EMMANUEL ETASSE (UT7)
[HTTP://HOMOAGILIS.BLOGSPOT.COM](http://HOMOAGILIS.BLOGSPOT.COM)
- HENRIK KNIBERG
[HTTP://BLOG.CRISP.SE/HENRIKNIBERG/](http://BLOG.CRISP.SE/HENRIKNIBERG/)



www.SQLI.COM

**THANK
YOU**

