Faire du ScrumBan avec iceScrum, en utilisant les stories

Scrum et Kanban, il existe de nombreuses possibilités pour tirer le meilleur des 2 (pour reprendre le titre de la traduction que nous avons faite du mini-livre de Kniberg et Skarin, voir http://www.aubryconseil.com/post/2eme-version-francaise-de-Kanban-et-Scrum).

Je vais en décrire une autre qui fait un grand usage des stories, après celle basée sur les tâches. Attention il ne s'agit pas de Kanban, mais de ScrumBan.

Pour qui ?

Cette approche est destinée à ceux qui, ayant essayé Scrum, ont trouvé que le cadre des sprints avec son cérémonial était un peu trop contraignant pour leur contexte.
Généralement, ce contexte est caractérisé par une équipe légère qui reçoit un flux relativement continu d'entrées, comme c'est typiquement le cas pour le support d'un produit déjà en production.
Ces entrées qui peuvent être des demandes d'évolution, des bugs ou des questions, il n'est pas toujours nécessaire de les décomposer, comme c'est le cas avec Scrum avec les notions de story et de tâche.

La technique que je vais présenter peut se mettre en oeuvre sur un nouveau projet, ou sur une nouvelle release, voire un nouveau sprint d'un projet en cours.

Configuration

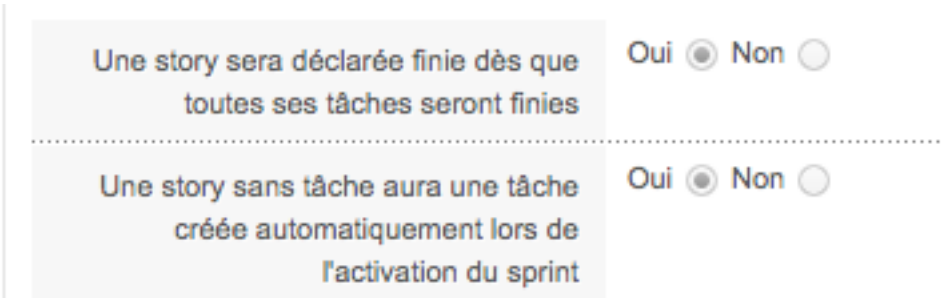
Avec du Scrum classique géré avec iceScrum, le déroulement le plus direct pour finir une story est le suivant :

1. création d'une story dans le bac à sable
2. acceptation de la story qui passe dans le backlog
3. estimation de la story
4. planification de la story qui la fait passer dans le plan de release
5. lors de la planification du sprint, création de tâches associées à la story
6. avancement des tâches dans le plan de sprint afin qu'elles soient finies
7. déclaration de la story comme finie (par le PO)

La technique présentée permet d'éliminer les étapes 3, 5 et 7.

Pour cela, il faut configurer iceScrum en choisissant les bonnes options dans les pratiques du projet :

En effet, nous n'allons pas estimer les stories, simplement les compter. Cela nous permettre aussi de sauter l'étape 3 du workflow.



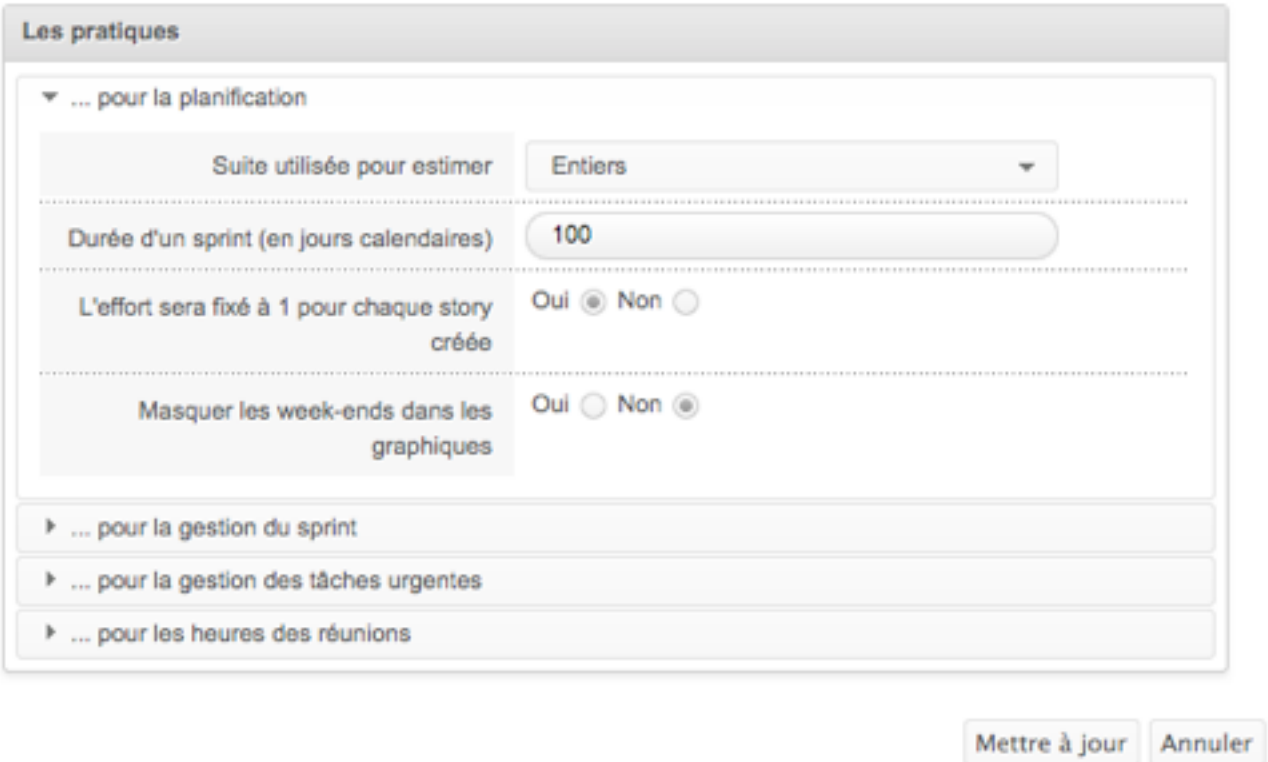
La deuxième option nous permettra d'éviter l'étape 7.
La dernière consiste à créer une tâche dès que la story est mise dans le plan de sprint. Comme cela, nous n'aurons pas besoin de créer une tâche explicitement (étape 5).

Sprint ?

Dans une approche orientée flux, les sprints ne nous intéressent pas beaucoup. Avec iceScrum nous allons quand même créer un sprint, mais de très longue durée, par exemple 100 jours.
On pourra toujours modifier la date de fin de sprint plus tard, par exemple pour la prolonger.

Le sprint est créé à la création du projet ou plus tard, à partir du plan de release.

On peut aussi le faire, lors de l'assistant de création ou en accédant aux pratiques du projet, définir une durée longue, comme ci-dessous :



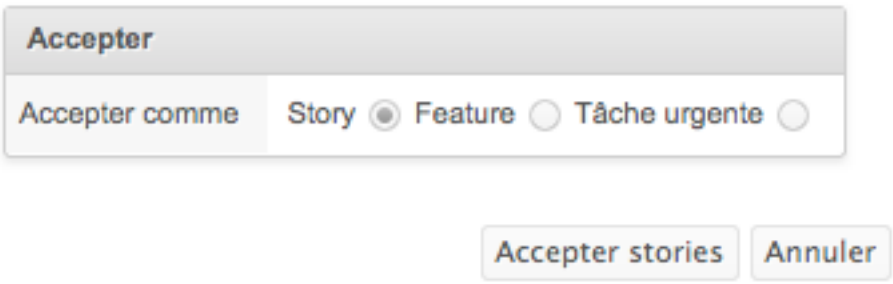
Eléments utilisés

Dans Scrum -et dans iceScrum- on distingue la story de la tâche. La tâche est un travail qui contribue à la réalisation d'une story. Cette distinction est très utile dans les projets de développement, mais avoir ces 2 notions n'est pas nécessaire pour gérer des travaux orientés flux.

Dans cette approche ScrumBan, nous allons utiliser intensivement la notion de story. Cependant nous allons passer par la vue Plan de sprint pour bénéficier de son intérêt visuel avec les colonnes. Ce qui avance dans les colonnes, ce sont uniquement des tâches. Mais comme l'option de création automatique de tâche a été activée, nous ferons avancer cette tâche qui sera l'image temporaire de la story

Utilisation du bac à sable

Le bac à sable, qui offre une ouverture vers le monde extérieur et des possibilités de discussion sera utilisé pour collecter les demandes.
Une fois les «stories» entrées dans le bac à sable et après un éventuel dialogue avec les créateurs, le Product Owner peut les accepter comme stories et elles passent dans le backlog.



Utilisation du backlog

Grâce à l'option que nous avons choisie, les stories mises dans le backlog sont directement estimées à un point. Cela revient à les compter et fait qu'elles sont directement utilisables dans un sprint.

L'utilisation du plan de release n'est nécessaire avec cette approche : nous allons ouvrir directement le plan de sprint, ce qui a pour effet de mettre le backlog dans la partie gauche.
On peut aussi afficher le backlog en widget en faisant glisser le bouton backlog dans la zone de gauche.

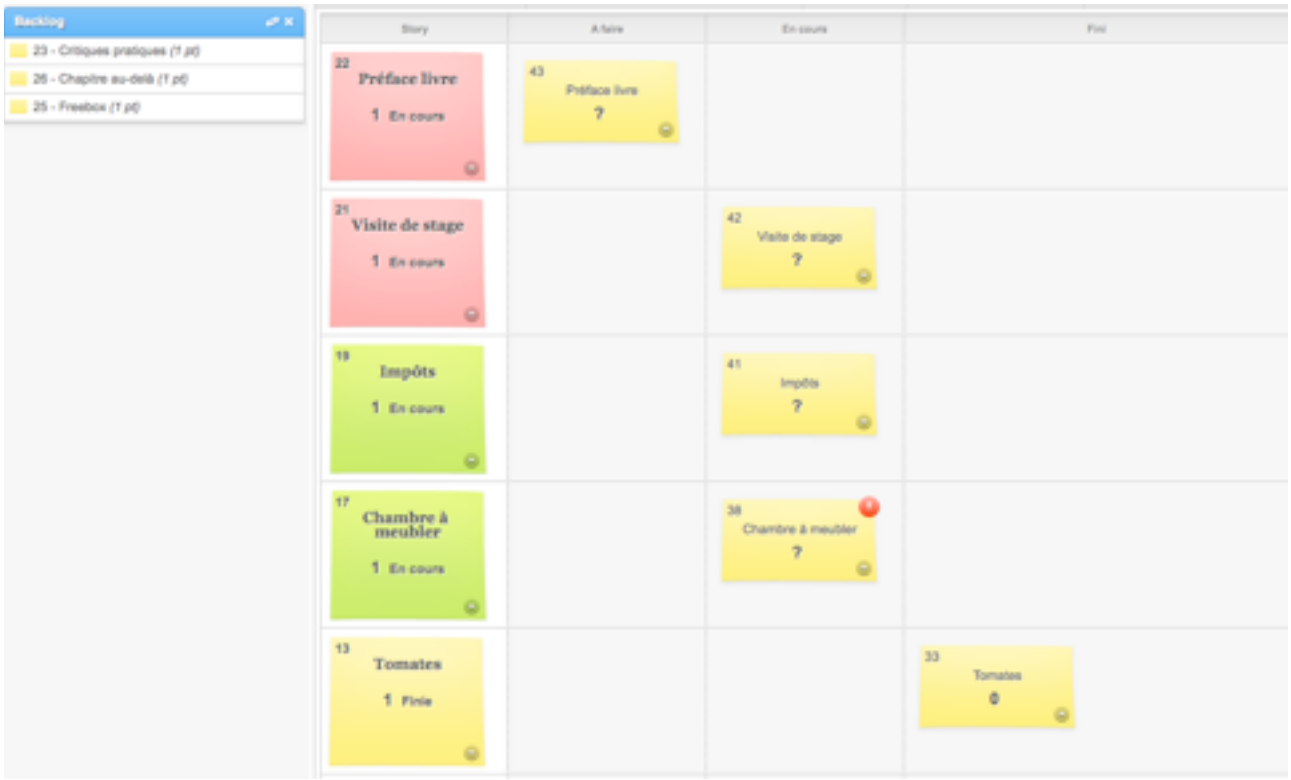
Lorsque le backlog est minimisé à gauche, seules les stories estimées sont présentées. Comme les nôtres ont été automatiquement estimées à 1, elles y seront.

Dans cette zone réduite le nombre de stories affichées est limité à 6 (au-delà il faut scroller). Dans le cadre de cette approche ScrumBan, c'est une limite largement suffisante.

Utilisation du plan de sprint

Nous pouvons prendre une story du backlog minimisé dans la partie gauche et la faire glisser vers le plan de sprint, dans la colonne story (le sprint étant activé).
Une tâche est créée automatiquement, qui reprend le nom de la story et se place dans la colonne à faire. La story passe dans l'état en cours.

Il est possible d'ajouter d'autres stories. Il n'y a pas de limite.
La tâche-story évolue ensuite dans le tableau, dans la colonne en cours. Quand on la déplace dans la colonne fini, la story est automatiquement déclarée finie et passe en dessous des stories en cours.

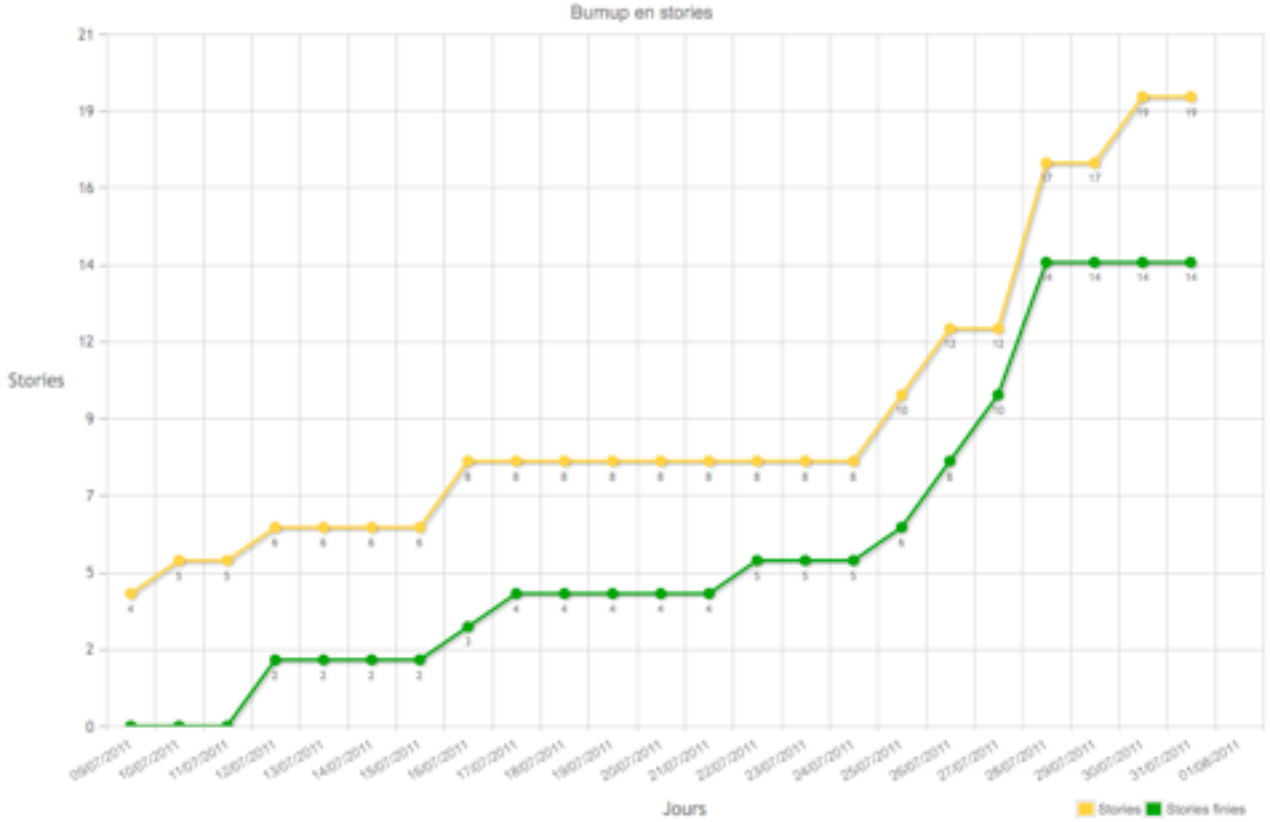


On peut distinguer les tâches bloquées de façon visuelle (symbole rouge ci-dessus). Il est facile de savoir si tous les membres de l'équipe ont du travail, en filtrant les tâches selon les membres.
Dans la version actuelle d'iceScrum, il n'y a pas de limite sur les stories. Mais dans la colonne Story, comme celles finies passent en bas, il est facile de voir le nombre de stories en cours.

Indicateurs

Les indicateurs portant sur les stories ou sur les features ne seront d'aucun intérêt puisqu'ils ne sont pas mis à jour tant que le sprint n'est pas fini.
Le burndown de sprint en heures ne sera pas significatif, puisqu'on est dans un contexte où il n'aura pas nécessairement une tendance à descendre.

L'indicateur le plus intéressant sera le burnup de sprint en stories, qui montre 2 courbes :



une pour le total de stories et l'autre pour celles qui sont finies, actualisées tous les jours.

Si on a toujours une tâche par story, le burnup en tâches est équivalent.

Mesures Kanban

Une mesure essentielle en kanban est le temps de cycle (ou lead time, voir http://www.fabrice-aimetti.fr/dotclear/index.php?post/2010/11/07/Kanban-%3A-definition-du-Lead-Time-et-du-Cycle-Time). iceScrum a déjà les outils permettant de le calculer. Pour une story, on conserve les dates de changement d'état, visibles dans la vue détail :

Dates	
Proposée le	22/04/2011 10:36:29
Acceptée le	22/04/2011 10:37:14
Estimée le	22/04/2011 10:37:14
Planifiée le	23/04/2011 23:29:07
En cours depuis le	23/04/2011 23:29:07
Finie le	23/04/2011 23:29:20

Les dates pour Accepté et Estimée sont les mêmes, et pareil pour Planifiée et En cours, c'est la conséquence de nos options.

Les mesures Kanban en découlent :
- Le Lead time correspond à la durée entre Proposée et Finie.
- Le temps de réaction correspond à la durée entre Proposée et Acceptée.
- Le temps de cycle correspond à la durée ente En cours et Finie.

Un peu de Scrum

Avec cette approche ScrumBan, on a la possibilité de conserver des pratiques Scrum :
•le rôle de Product Owner
•le backlog (et le bac à sable dans iceScrum)
•la définition de fini pour indiquer la stratégie d'utilisation du kanban
•la release, comme jalon. Un exemple est d'avoir une release de 3 mois avec un seul sprint, nécessaire pour avoir un plan de sprint kanbanisé.

Il reste bien entendu possible d'ajouter des tâches urgentes et récurrentes dans le sprint, ainsi que de créer d'autres tâches en plus de celle créée automatiquement.

C'est bien du ScrumBan et pas du Scrum pur, dans lequel il serait, par exemple, interdit d'ajouter des stories à un sprint commencé.

Evolutions dans iceScrum pour un meilleur ScrumBan

On a vu une utilisation d'iceScrum qui peut convenir quand les travaux à suivre rentrent bien dans ce workflow basique.
Evidemment une amélioration intéressante dans l'esprit Kanban serait de pouvoir modifier le workflow. Dans l'approche proposée ici, on a un workflow imposé, à 4 états :
•en attente dans le bac à sable
•estimée (à 1)
•en cours, avec la tâche à faire
•en cours, avec la tâche en cours
•fini (lorsque la tâche est finie)

Une autre évolution serait d'ajouter une limite, par exemple sur le nombre de tâches dans l'état en cours pour l'ensemble des stories.

L'outil autorise actuellement du flux remontant : une tâche en cours peut repasser dans la colonne à faire (mais une tâche finie ne peut pas remonter le flux) et une story finie peut redevenir en cours. Une option pourrait être ajoutée dans la configuration des pratiques, pour l'autoriser ou pas.