User stories et Backlog de produit

User stories ou scénarios :

Une user story ou un scénario est une exigence du système à développer formulée en une ou deux phrases dans le langage des utilisateurs pour servir un but. Sa granularité doit permettre à l'équipe de réalisation, d'estimer son coût en effort et de la réaliser entièrement à l'intérieur d'un sprint (ou une itération).

L'avantage des user-stories est qu'elles facilitent la démarche en deux temps : **générales** et **grossières** au début, elles s'enrichissent ensuite d'exemples et de tests d'acceptation. Elles facilitent la communication, l'ajout ou la suppression de détails.

Une user story s'écrit comme suit;

En tant que <rôle>

Je veux <liste de tâches>

Afin de <valeur ajoutée ou résultat>

Exemples

En tant qu'acheteur en ligne, **je veux** pouvoir ajouter et supprimer des items à mon panier **afin de** pouvoir n'acheter que ce dont je suis vraiment certain.

En tant que client, **je veux** consulter la liste des factures émises **afin d'en** sélectionner une.

En tant que fournisseur, **je veux** consulter la liste de mes clients **afin d'en** sélectionner un.

En tant que client, **je veux** connaître le montant total des factures impayées **afin de** connaitre le montant à payer.

INVEST

Une bonne user story doit respecter l'acronyme INVEST (indépendant, négociable, « valuable », estimable, « small » petite, testable)

« I » pour **Indépendante** : lorsque le client peut en toute liberté décider de l'ordre dans lequel les scénarios (story) sont implémentés sans qu'interviennent des contraintes techniques. Si des stories sont fortement liées, alors il faut peut-être les combiner. Une user story se doit d'être indépendante de toutes les autres stories. En prenant cette définition à l'envers on la comprend mieux, une user story se doit de n'avoir aucune dépendance envers les autres stories, elle doit pouvoir être réalisée individuellement. Ce besoin découle du fait qu'à tout moment on veut pouvoir réorganiser, « reprioriser » le « product Backlog ». Si les stories ont des dépendances, cette « repriorisation » est plus limitée.

Bon exemple :

En tant que client, je veux consulter la liste des factures émises afin d'en sélectionner une.

Mauvais exemple :

En tant que client, je veux créer la liste des factures afin des consulter.

« N » pour **Négociable** : L'équipe de développement n'est pas contrainte par la manière dont sera implémenté le scénario. Elle a la latitude d'imaginer une solution efficace. Toute story qui se trouve dans le **backlog de produit**, peut être réécrite, repensée ou même supprimée à tout moment en raison de changements du marché, de modification de l'orientation de besoins techniques ou tous autres motifs valables provenant de membre de l'équipe ou du client.

Bon exemple :

En tant que client, je peux connaître le montant total des factures impayées afin de connaitre le montant à payer.

Mauvais exemple

En tant que client, je veux, lorsque je clique sur le bouton «calculer» une ligne s'ajoute et on affiche sur cette ligne le montant total des factures impayées afin de connaitre le montant à payer.

« V » pour « **Valuable** » (génère de la valeur pour l'utilisateur final): un bon scénario ou story décrit une fonctionnalité complète de l'application dont le client apprécie l'intérêt.

Bon exemple :

Un bon découpage qui fait apparaitre les scénarios distincts rendus par le service de facturation :

- **Consulter la liste des factures émises**
- **Afficher la liste des factures ordonnées par date**
- **Afficher la liste des factures par adresse de livraison**
- **Consulter une facture client**
- **…**

Mauvais exemple :

Créer la base de données pour le module de facturation. Créer l'interface graphique pour la facturation…

« E » pour **Estimé** : pour que le scénario puisse servir de base à la planification, on doit connaître au moins une estimation du coût d'implémentation. Une user story doit pouvoir être évaluée en terme de complexité par l'équipe. Si une user story ne peut être évaluée, elle ne sera jamais planifiée ni incluse dans un « sprint backlog » (développement fait lors d'une itération) ou découpée en taches. Habituellement elle ne peut l'être lorsqu'elle manque d'informations de support, ou lorsque la description faite par le « Product Owner » (client) est insuffisante ou vague.

Bon exemple :

Implémenter les règles métier R1, et R2

R1 : si le nom de l'utilisateur existe déjà, lorsque l'acheteur en ligne essaye de créer son compte, alors le message « ce compte existe déjà » doit être affiché.

R2 : un rabais de 5% est octroyé pour tous les clients dont le montant total de la facture dépasse les 1000$.

Mauvais exemple :

Garantir le respect des règles de gestion en vigueur.

« S » pour **Suffisamment petit** (Size appropriately) : toujours, pour planifier sa réalisation un scénario doit être réalisable en peu de temps pour que l'équipe de projet puisse en planifier plusieurs dans un même sprint (une itération de développement). Une user story doit être suffisamment petite pour permettre d'être évaluée avec le plus de précision et certitude possible. Si l'équipe évalue la complexité relativement élevée, cela démontre en général un niveau d'incertitude élevé et cela implique qu'elle devrait être découpée en plusieurs stories.

Bon exemple :

Voir « V »

Mauvais exemple :

Réaliser le module de facturation – *il faudra découper* – …

« T » pour **Testable** : planifier un test (ou des tests) d'acceptation est ***un excellent moyen pour vérifier que tout le monde a bien compris le scénario ou la story***. Garder en tête qu'une story ne peut être considérée finie sans qu'elle ait été testée avec succès. S'il n'est pas possible de tester la story due à un manque d'information, elle ne devrait surtout pas être considérée pour être incluse dans un sprint Backlog. Ceci est encore plus vrai pour les équipes intégrant les techniques XP et particulièrement le TDD (développement piloté par les Tests).

Exemple de test d'acceptation :

En tant que client, je veux connaître la liste des factures par date afin d'en connaitre l'historique.

Test d'acceptation : voir si on peut afficher la liste des factures ordonnées par la date de facturation.

Si une user story répond à tous ces critères (INVEST) sa place dans le product backlog est tout à fait justifiée. Il ne restera donc plus qu'à la prioriser.

Élément à considérer pour enrichir les users stories :

En plus de l'histoire elle-même, certaines informations utiles peuvent être ajoutées pour la définition de la story ce qui donnera comme ensemble :

Thème : Le thème auquel cette story est reliée. Utiliser un « postit » de la couleur définie dans le thème permet de rendre le backlog plus visuel.

Acteur : L'acteur ou « personas » utilisé pour rédiger la user story. Là encore si c'est possible utiliser l'icône rend l'ensemble plus visuel.

Story : …

Description : utilisé lorsqu'il s'agit d'une story technique ou d'un bug.

Prototype : Le prototype est un modèle, un dessin d'écran ou de formulaire voulu. Une image vaut mille mots.

Valeur : La valeur métier, on pourra utiliser le même genre de principe que celui présenté pour les thèmes.

Complexité : Sera évaluée par l'équipe lors des sprints plannings ou des rencontres de travail du backlog de produit (optionnelles).

ROI : Le Retour sur Investissement se calcule simplement = Valeur/Complexité. C'est un indicateur intéressant pour prioriser le backlog.

Tests d'acceptation : Permet de définir quels seraient les tests d'acceptation de la story.

Commentaires : Autres informations pertinentes… Attention de « surdocumenter ».

Le backlog de produit :

Le **backlog de produit** est la liste des fonctionnalités (**story ou un cas d'utilisation**) attendues d'un produit.

Scrum n'impose pas de pratique pour identifier et nommer les éléments du **backlog**. L'usage le plus courant est de définir un élément comme étant une **story ou un cas d'utilisation**.

Dans un **backlog de produit**, les stories sont rangées (classées) selon l'ordre envisagé pour leur réalisation. Cette notion de priorité prend une grande importance dans le développement itératif.

Exemple simplifié de backlog de produit

Ici la colonne **Effort ou charge** désigne le nombre de jours/homme.

5jours/homme : a une durée de 5 jours si le nombre de ressources qui travaillent en parallèle et à temps plein est égal à 1. A une durée de 10 jours si le nombre de ressources qui travaillent en parallèle et à **mi-temps** est égal à 1. A une durée de 1 jours si le nombre de ressources qui travaillent en parallèle et à temps plein est égal à 5.

La colonne **risque** : indique le niveau de complexité technique.

Scénario ou story	Priorité	Effort ou charge	Risque
En tant qu'administrateur, je dois pouvoir supprimer un usager du système sans le faire planter afin d'isoler du système cet utilisateur.	M	1	1
En tant que client, je veux que mon robot puisse trier les pièces selon leur forme afin de facilité leur placement dans les étagères	M	2	1
En tant que client, je veux que mon robot se déplace entre le point A et le point B selon le chemin le plus court afin d'optimiser son temps de travail.	M	10	3
En tant que candidat, je dois pouvoir soumettre mon CV par voie interne afin de facilité ma mise en candidature.	M	3	1

*Exemple 1, plus détaillé sous forme de tableau (approprié au **backlog de sprint**)*

#	Scénario ou story	Niveau utilisateur (cas d'utilisation)	Niveau détaillé	Priorité	Risque	Effort
1	En tant que recruteur je veux recevoir les candidatures via notre site internet.	Réaliser un formulaire de mise en candidature	Construire le formulaire avec tous les champs.	M	1	1
			Indiquer les champs obligatoires.	M	1	1
			Sélectionner dans une liste les champs de compétence pour le poste. La liste est triée par ordre alphabétique.	M	1	1
			Avoir une zone de texte de 500 caractères pour les motivations pour le poste.	S	2	1
			Trier la liste des compétences selon le poste.	C	2	1
			Enregistrer le formulaire correctement.	M	2	2
2	En tant que recruteur je souhaite afficher la liste des candidats par domaine de compétences.	Lister les candidats par domaine de compétences	Afficher la liste des candidats selon les trois premières compétences.	M	1	1
			Lister les candidats selon un résumé d'expérience (à partir de la liste précédente).	M	1	1
			En plus des compétences, afficher les candidats par niveau de scolarisation.	C	1	1
			En plus des compétences, afficher les candidats par ordre d'éloignement.	W	2	2

*Exemple 2, plus détaillé sous forme texte (approprié au **backlog de sprint**)*

1 (# du scénario)	
Acteur ou rôle :	Recruteur
Scénario ou story :	En tant que recruteur je veux recevoir les candidatures via notre site web afin de facilité le recrutement et de limiter les erreurs.
Détail ou description :	1. Construire le formulaire avec tous les champs. 2. Indiquer les champs obligatoires. 3. Sélectionner dans une liste les champs de compétence pour le poste. La liste est triée par ordre alphabétique.

	4. Avoir une zone de texte de 500 caractères pour les motivations pour le poste. 5. Trier la liste des compétences selon le poste. 6. Enregistrer les valeurs du formulaire correctement.
Tests d'acceptation :	Inscrire un candidat potentiel à partir du formulaire du site web et vérifier si l'enregistrement des données s'est bien fait.
Complexité :	2
Effort :	5j/homme **ou** 3 (effort relative selon les autres scénarios)
Commentaires :	
2 (# du scénario)	
Acteur ou rôle :	Recruteur
Scénario ou story :	En tant que recruteur je souhaite afficher la liste des candidats par domaine de compétences afin de facilité le choix d'un candidat pour un domaine donné
Détail ou description :	1. Afficher la liste des candidats selon les trois premières compétences. 2. Lister les candidats selon un résumé d'expérience (à partir de la liste précédente). 3. En plus des compétences, afficher les candidats par niveau de scolarisation. 4. En plus des compétences, afficher les candidats par ordre d'éloignement.
Tests d'acceptation :	**Pré-condition** : Avoir une liste étalon de plusieurs candidats de différents domaines déjà enregistrés dans le système. Faire afficher la liste des candidats et vérifier s'ils sont affichés par domaine de compétence.
Complexité :	2
Effort :	3j/homme **ou** 2 (effort relative selon les autres scénarios)
Commentaires :	Il faut penser de mettre des données dans le fichier des candidats pour faire les tests.

La méthode MoSCow pour les priorités :

La technique utilisée pour prioriser les besoins dans un contexte itératif est celle de MoSCoW. L'avantage de la méthode *MoSCoW* réside dans la signification de l'acronyme, qui est plus compréhensible que d'autres techniques de priorisation comme élevé/moyen/faible

« M » pour **Must Have** : ***DOIT*** être fait. L'exigence est essentielle. Si elle n'est pas faîte le projet échoue. On peut dire également priorité haute.

« S » pour **Should Have** : Il s'agit d'une exigence essentielle, qu'il faut faire dans la mesure du possible (***DEVRAIT***). Mais si elle n'est pas faîte, on peut la contourner et la livrer plus tard.

« C » pour **Could Have**: Il s'agit d'une exigence souhaitable. Elle ***POURRAIT*** être faîte dans la mesure où elle n'a pas d'impact sur les autres tâches.

« W » pour **Won't Have :** Il s'agit d'une exigence «Luxe». ***NE SERA PAS*** faîte cette fois mais plus tard, mais intéressante et à garder pour la prochaine version.

Ce qu'il faut faire :

- Cultiver le backlog : le backlog de produit évolue dans le temps, il faudra le mettre à jour.
- Partager le backlog de produit avec toute l'équipe
- Surveiller la taille du backlog de produit : ne pas avoir plus de 150 éléments à faire dans le backlog

Éviter :

- D'avoir plusieurs backlog de produit pour le même de produit

- De ne pas avoir de backlog de produit
- De confondre le **backlog de produit** avec le **backlog de sprint**

Sources :

Gestion de projet agile 3e Édition , Véronique Messager Rota, Eyrolles 2011.

SCRUM : le guide pratique de la méthode agile la plus populaire., Claude Aubrey, DUNOD 2011.

Préparé par Saliha Yacoub, automne 2013

À lire aussi :

https://sebastienboyer.wordpress.com/tag/user-story/